

Federated Credit Card Fraud Detection Under Simulated Operating Conditions: Encrypted Aggregation, Partial Participation and Byzantine Resilience

El Mehdi Adnani^{1*}, Mohammed Erritali²

Data Science for Sustainable Earth Laboratory (DataEarth)-Faculty of Science and Technology,
Sultan Moulay Slimane University, Beni Mellal, Morocco^{1,2}

Abstract—Credit card fraud is rare, costly and continuously evolving, and the data required to detect it are distributed across institutions that cannot share them. This study presents a privacy-preserving federated framework in which several banks jointly train a fraud detector without exchanging raw transactions, and it quantifies the cost of each privacy and robustness mechanism. The evaluation uses ten simulated institutions obtained by partitioning a public benchmark of 284807 transactions. A federated autoencoder learns legitimate spending without fraud labels, and a nearest neighbour classifier operating on the per-feature reconstruction errors provides the decision, using labelled error vectors. The placement of this classifier determines the trade-off between privacy and accuracy. Fitted on pooled error vectors, it attains an F1-score of 0.8625 over five seeds, comparable to the centralized model at 0.8667. Fitted locally at each bank, it attains 0.7254 with no exchange of data, and a head trained by federated averaging, which exchanges only parameter updates, attains 0.7655, close to a federated supervised network on raw features at 0.783. On the same split, label-free detectors remain below 0.45 and a centralized random forest reaches 0.835. CKKS homomorphic encryption conceals every update from the aggregation server and reproduces plaintext averaging to within $1.8e-7$, at 25.6 times the ciphertext volume and 1.011 times the computation. Robust aggregation operates on plaintext updates and is evaluated separately: the coordinate-wise trimmed mean sustains an F1-score of 0.8402 when a fifth of the institutions mount a gradient inversion attack, and fails at two fifths. A participation rate of 60 percent remains within seed variation of full participation while reducing traffic by 40 percent. These results provide a consortium with a quantified trade-off between privacy, robustness and detection quality.

Keywords—Anomaly detection; byzantine robustness; credit card fraud detection; federated learning; homomorphic encryption

I. INTRODUCTION

Payment card fraud cost issuers, merchants and acquirers 33.41 billion US dollars worldwide in 2024, and the Nilson Report projects 403.88 billion in cumulative losses over the following decade [1]. Behind these totals lies a detection problem with an awkward statistical shape. Fraud is rare: in the public European cardholder benchmark used throughout this work, 492 of 284807 transactions are fraudulent, a positive

rate of 0.173 percent [2]. A model that labels every transaction as legitimate is therefore 99.83 percent accurate and useless, which is why detection quality must be judged on F1-score and on the area under the receiver operating characteristic curve rather than on accuracy, and why results reported as raw accuracy on this dataset are difficult to interpret.

Fraud also moves. Patterns that a supervised classifier learns from last quarter's labelled cases lose value as attackers adapt, and labelled fraud arrives late, after chargebacks and investigation. Anomaly detection offers a different bargain: model what normal spending looks like, then flag what the model cannot reconstruct. A prior study of this benchmark followed the same route in a centralized setting, using autoencoders and boosting [3]. The present work keeps that detection principle and confronts the constraint that a centralized study can ignore, namely that the data cannot be gathered in one place.

Two forces prevent pooling. The first is regulatory: the General Data Protection Regulation [59] restricts transfers of personal data between controllers, and the Payment Card Industry Data Security Standard [60] constrains how cardholder data is stored and moved, so a shared training corpus across banks is rarely lawful and never simple. The second is operational: a central repository of transaction histories is a concentrated target, and breaches of such repositories have exposed millions of records. Institutions consequently train on their own data alone, and each of them sees too few fraud cases to characterize the phenomenon well.

Federated learning removes the pooling requirement. Clients train locally and transmit only model updates, which a server aggregates, classically by weighted averaging [4], into a shared global model [5]. For a consortium of banks the relevant variant is cross-silo federation: few participants, large local datasets, unreliable availability, and mutual distrust [6].

Applying it to fraud detection has an active literature [7], [8], [9], [10]. Read together, however, these studies share a set of evaluation habits that limit what a practitioner can conclude from them. Reported performance can be inflated by the evaluation protocol rather than earned by the method, because oversampling applied before the train and test split leaks synthetic neighbours of test frauds into training. Privacy

*Corresponding author

mechanisms, when present, are described but not costed: homomorphic encryption is introduced as a property of the protocol, while its bandwidth and compute price is rarely quantified. Deployment conditions are idealized, with full client participation each round, data distributed identically across silos, and every client honest. Where heterogeneity is simulated, it is summarized by a Dirichlet concentration parameter alone, without disclosing how the training mass actually landed across clients. Results are typically reported from a single run, on a task whose F1-score varies by more than a point across random seeds.

None of these habits is fatal in isolation. Together they mean the published numbers do not answer the questions a bank actually asks: what does privacy cost in bandwidth, how much does detection degrade when half the members are offline, and what happens when one member is compromised.

Each individual axis has a mature literature. Communication-efficient homomorphic encryption [11], [12], non-IID federated optimization [13], [14], [15] and Byzantine-robust aggregation [16], [17] are all well studied, but these results are established on vision benchmarks and generic classifiers. No published study evaluates them jointly, on one fraud detection pipeline and one benchmark, so how they behave side by side on a single detector remains unmeasured. Two narrower gaps follow from the same evaluation culture. Federated studies conflate two different measurements when a downstream classifier sits on top of the federated model, namely the quality of the learned representation and the benefit of refitting the classifier on that representation. And heterogeneity is reported through the parameter that generated it rather than through the partition it produced, which allows a non-IID result to be driven by a single dominant client holding most of the data.

The components of the framework, namely autoencoders, federated averaging, CKKS encryption, nearest neighbour classification, FedProx and trimmed-mean aggregation, are established techniques. The contribution is twofold. At the system level, a federated two-stage detector is proposed in which the classifier operates on per-feature reconstruction error vectors, and the placement of that classifier is evaluated from fully local to pooled. Empirically, encryption cost, partial participation, heterogeneity and Byzantine attacks are evaluated jointly on one pipeline and one public benchmark. Fig. 1 summarizes the pipeline and the conditions under which it is tested. The contributions are as follows.

- A two-stage federated pipeline with explicit label usage The autoencoder is trained across silos on legitimate transactions only and therefore requires no fraud labels. The classifier stage operates on labelled error vectors, and four placements of it are compared. Fitted on pooled error vectors, the pipeline attains an F1-score of 0.8625 over five seeds, statistically indistinguishable from 0.8667 for the centralized model. Fitted locally it attains 0.7254, and a federated head that exchanges only parameter updates attains 0.7655. Label-free, centralized supervised and federated supervised baselines are scored on the same split. Frozen and refitted classifiers are both reported so that representation quality and classifier adaptation remain separable.

- Quantified encryption cost CKKS aggregation is implemented with TenSEAL and measured on the trained model. The ciphertext expansion is 25.6 times, the per-round compute overhead factor is 1.011, and the encrypted aggregate matches plaintext averaging to within $1.8e-7$. Total communication rises from 22.4 MB to 574.0 MB over 30 rounds.
- A Byzantine defence efficacy curve Under the canonical boosted gradient inversion attack, undefended averaging collapses to an F1-score of zero with 20 percent malicious clients, coordinate-wise trimmed mean restores 0.8402, and both fail at 40 percent. The curve locates the point at which the defence ceases to hold on plaintext updates.
- Heterogeneity reported by partition, not by parameter Alongside the Dirichlet sweep, the study publishes per-client data mass, fraud ratios and an entropy-based effective client count, which shows that a concentration of 0.1 over ten clients is in practice a two-client federation.
- A communication budget supported by evidence Participation at 60 percent reduces communication by 40 percent with detection quality within seed variation of full participation, and the participation, drift and adversarial results are reported over multiple seeds with confidence intervals.

These measurements are reported together on one pipeline and one public benchmark, which Table I positions against the federated fraud detection literature. Taken together, they quantify the bandwidth price of encryption, the accuracy price of keeping labelled error vectors local, the quality obtainable under partial participation and the point at which a compromised member degrades the model.

The remainder of the study is organized as follows. Section II reviews the related work and positions this study against it. Section III formalizes the pipeline, the federated protocol, the encryption model and the threat model. Section IV reports the experimental results. Section V discusses their mechanisms and their deployment implications. Section VI concludes.

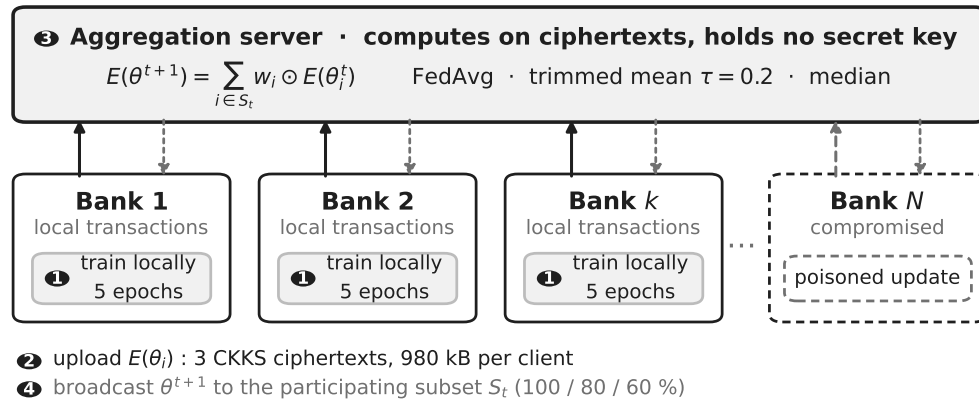
II. RELATED WORK

A. From Rules to Learned Models in Card Fraud Detection

The first generation of deployed fraud systems was rule based, using thresholds on transaction amount, merchant category, geography or velocity [18], [19]. Such systems are auditable and fast, which is why they survive in production, but they encode fraud as it was understood when the rules were written, and tightening them as attackers adapt raises the false alarm rate faster than the detection rate [20], [21]. Supervised learning replaced them with decision boundaries learned from labelled histories. Ensembles of trees proved particularly effective on tabular transaction data [22], and comparative studies report accuracies above 99 percent for this family of models [23], [24], [25], while convolutional architectures have been applied to transaction sequences [26].

These accuracy figures deserve a caution that shapes the rest of this study. At a positive rate of 0.173 percent, a

(a) One federated round, repeated 30 times over 10 institutions



(b) Detection with the trained global model

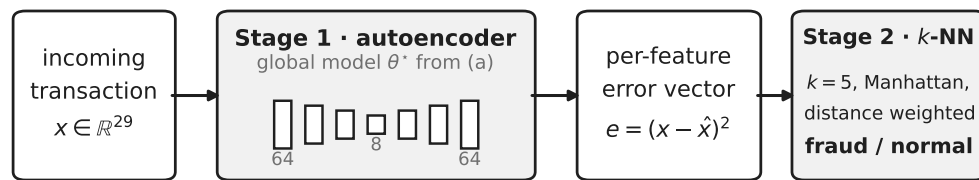


Fig. 1. The proposed pipeline. (a) One round proceeds in four steps: each participating bank trains the shared autoencoder on its own legitimate transactions (1), encrypts the resulting parameters under CKKS and uploads them (2), the server aggregates the ciphertexts without decrypting them (3), and the new global model is broadcast to the subset participating in the next round (4). One client is drawn as compromised to indicate the Byzantine threat model; encrypted aggregation and Byzantine-robust aggregation are evaluated separately (Section III-F). (b) At detection time the global autoencoder reconstructs an incoming transaction, and the per-feature squared error vector, not a scalar error, is classified by a distance-weighted Manhattan nearest neighbour rule.

classifier that predicts legitimate for every transaction is 99.83 percent accurate, so accuracy above 99 percent is consistent with a useful detector and equally consistent with a useless one. Reported results are comparable only through metrics that respond to the minority class. This is not a criticism of individual studies, several of which do report such metrics, but it explains why headline accuracies across this literature cannot be ranked against one another.

The scarcity of labelled fraud also motivates a different formulation. Unsupervised and semi-supervised anomaly detection avoids the dependency on labels by modelling legitimate behaviour and treating deviation as evidence [27], [28], [29]. Autoencoders are the natural instrument, because a network trained to reconstruct normal transactions reconstructs fraudulent ones poorly, and the reconstruction error becomes the detection signal [30], [31], [32]. One study of this benchmark compares an autoencoder against supervised baselines in a centralized setting [3]. A related centralized line of work does not stop at the error itself but feeds it into a second supervised classifier, which recovers structure that a single threshold discards [33]. Every study cited in this subsection assumes that training data can be assembled in one place, and none quantifies what it would cost to relax that assumption.

B. Federated Learning and its Open Problems

Federated learning removes the assumption of a central corpus. Yang et al. [5] organize the field into horizontal federated learning, where participants hold the same features

for different subjects, vertical federated learning, where they hold different features for overlapping subjects, and federated transfer learning for the case where neither aligns. Kairouz et al. [6] draw the distinction that matters operationally, between cross-device federation over millions of unreliable phones and cross-silo federation over a handful of institutions with large datasets and persistent infrastructure. A consortium of banks is a horizontal, cross-silo setting.

The advantages and difficulties of the paradigm are well surveyed [34], [35], [36], [37]. Confidentiality improves because raw records never move, which also removes the single high-value target that a shared repository creates [38]. The recurring difficulties are equally documented [39], [40]: communication cost, statistical heterogeneity, partial availability, and the fact that model updates are not themselves free of information about the data that produced them.

An adjacent line of work replaces the coordinating server with a distributed ledger, so that aggregation is verifiable without a trusted coordinator [41]. This addresses a different trust assumption from the one considered here, since a coordinating server is retained here and its ability to read updates is removed, rather than removing the server itself, and the ledger approach is treated as a complementary direction. The literature on federation across edge and internet-of-things devices [42] likewise targets the cross-device regime, where the binding constraints are battery, connectivity and device count rather than institutional trust.

TABLE I. POSITION OF THIS WORK RELATIVE TO FEDERATED FRAUD DETECTION STUDIES

Study	Contribution	Added in this work
Fanai [33]	centralized autoencoder with a deep classifier	the two-stage design is federated, and the cost of doing so is measured
Aurna [7]	sampling strategies for federated deep classifiers	encrypted aggregation, partial participation and Byzantine resilience
Abdul Salam [8]	data balancing inside a federated pipeline	evaluation at the natural class ratio, with system-level metrics
D'souza [43]	scalability of privacy-preserving federated fraud detection	measured encryption cost and an explicit defence boundary
Awosika [9]	federation combined with explainable decisions	communication cost, heterogeneity disclosure and defended aggregation
FED-SPFD [10]	federated transfer learning across heterogeneous banks	the realized partition and its effective client count, plus encryption
This work	federated autoencoder with nearest neighbour detection on per-feature errors, evaluated on detection quality, communication, measured CKKS overhead, participation, disclosed heterogeneity, Byzantine resilience and multi-seed stability	

C. Federated Learning for Card Fraud Detection

Federated fraud detection is an active area. Aurna et al. [7] analyse the interaction between sampling strategies and deep classifiers under federated averaging on the public benchmark, and a related study evaluates balancing techniques inside a federated pipeline [8]. D'souza et al. [43] target the scalability of privacy-preserving federated fraud detection. A federated transfer learning framework addresses heterogeneous banks through segmented private features [10]. Awosika et al. [9] combine federation with explainability so that federated decisions remain auditable.

These studies establish that federated fraud detection is feasible and that federated models can approach centralized quality. Each addresses one facet of the problem: the interaction with sampling, the scalability of the protocol, heterogeneity between banks, or the interpretability of the resulting decisions. Table I places the present framework alongside them and states what it contributes on top of each. The pattern that emerges is the motivation for this study: the operating cost of federation, its behaviour when members are absent or compromised, and the composition of the partitions used to model heterogeneity are the dimensions this study measures and reports.

D. Privacy Mechanisms and the Choice of Encryption

Federation hides raw records but not the information carried by model updates, from which properties of the training data can be inferred [44], [45]. Three families of mechanism close that residual channel, and they trade off differently.

Differential privacy perturbs updates with calibrated noise so that no individual record measurably changes what the server observes [46], an approach whose federated instantiation is well established [65]. The guarantee is formal and the cost is accuracy, which is particularly damaging when the phenomenon of interest occupies 0.173 percent of the data.

TABLE II. PRIVACY PARADIGMS FOR COLLABORATIVE FRAUD DETECTION

Paradigm	Hidden from the server	Behaviour in this pipeline
Differential privacy [46]	individual contributions	noise reduces accuracy, most visibly on rare classes
SMPC [47]	individual updates	lossless; requires clients to be online together
Split learning [51]	raw inputs	lossless; the detector needs the full input reconstructed at the client
SecureML [53]	inputs, via two servers	lossless; requires two non-colluding servers
CKKS [49]	updates and the aggregate	lossless; cost is bandwidth, quantified in Section IV-D

Secure multi-party computation distributes the aggregation so that no single party observes an individual update [47], [48]. It is lossless but introduces multi-round interaction between participants. Homomorphic encryption allows the server to compute the weighted sum directly on ciphertexts. The CKKS scheme [49] supports approximate arithmetic on real vectors and is therefore the natural fit for averaging model parameters, and TenSEAL [50] provides a practical implementation. Encryption leaves the aggregate unchanged up to bounded noise, so predictive quality is expected to be untouched, and the price is transmission volume. BatchCrypt [11] documents roughly hundredfold inflation for naive homomorphic aggregation and reduces it by batching, and Pack [12] reports CKKS overhead near 17 times as its reference baseline. These figures vary by an order of magnitude across parameter choices, which is why the expansion factor is measured for the present configuration in Section IV-D.

Two further paradigms deserve mention: split learning partitions the network itself so that raw inputs never leave the client [51], [52], and SecureML combines secret sharing with two-server computation [53]. Table II compares them on the dimensions that decide the choice here. Homomorphic encryption is selected for a reason that matters to the experimental design as much as to deployment: because it does not alter the aggregate, any difference between the centralized and federated results is attributable to federation alone and not to privacy noise. Split learning is excluded on technical grounds rather than preference, since the second stage consumes a per-feature reconstruction error vector and therefore needs the complete input reconstructed on the client side.

E. Heterogeneity and Byzantine Robustness

Two properties of real federations break the assumptions under which federated averaging was analysed. Data across silos is not identically distributed. Hsu et al. [13] introduced the Dirichlet partitioning protocol that has become the standard way to synthesize heterogeneity. FedProx [14] constrains local updates with a proximal term to limit client drift, NIID-Bench [15] compares such methods across skew types, and FedBN [54] documents a specific incompatibility, namely that batch normalization statistics are non-linear functions of local data and are not correct to average.

Participants are also not necessarily honest. Blanchard et al. [16] showed that a single malicious client can steer an averaged

model arbitrarily, and Yin et al. [17] established statistical rates for the coordinate-wise trimmed mean and median, which tolerate a bounded fraction of adversaries by construction. This literature is mature, but it is established on image classification with generic architectures. Whether a robust aggregation rule preserves a reconstruction error geometry on which a nearest neighbour classifier subsequently operates is not answered by those results, and neither is the question of how heterogeneity affects a model that learns normality rather than a class boundary. Section IV answers both empirically.

III. PROPOSED METHOD

A. Design Rationale

Four choices shape the framework, and each of them answers a constraint that the others cannot.

Representation learning is unsupervised rather than supervised. Fraud labels are scarce, they arrive weeks after the event, and they describe attacks that have already succeeded, so a boundary learned from them ages quickly. An autoencoder trained on legitimate transactions learns the behaviour that every institution has in common, and fraud is detected as a departure from it, which keeps the representation useful as attack patterns change. Fraud labels enter only at the classifier stage.

Detection is performed on the per-feature error vector rather than on a scalar reconstruction error. Two transactions may fail to reconstruct by the same total amount while failing on entirely different components of the input, and only the vector preserves that distinction. Retaining it is what allows a nearest neighbour rule to recognize a fraud signature instead of merely detecting a large residual.

Training is federated because the institutions that hold the data cannot exchange it. What they do exchange, the model updates, still carries information about that data. Updates are therefore encrypted before they leave the institution, and the server aggregates them without decryption.

Finally, the second stage uses fraud labels, and the study evaluates four placements of its labelled error vectors, from fully local to pooled, which separates the quality of the federated representation from the benefit of adapting the classifier to it.

B. Problem Formulation

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^d$ a transaction and $y_i \in \{0, 1\}$ its label, where $y_i = 1$ denotes fraud. Here $d = 29$. The data is partitioned across N institutions, $\mathcal{D} = \bigcup_{i=1}^N \mathcal{D}_i$, with $n_i = |\mathcal{D}_i|$ and $n = \sum_i n_i$, and no institution may transmit any element of $\mathcal{D}_i$. Federated training minimizes the weighted sum of local objectives given in Eq. (1),

$$\min_{\theta} \mathcal{L}(\theta) = \sum_{i=1}^N \frac{n_i^0}{n^0} \mathcal{L}_i(\theta), \quad (1)$$

$$\mathcal{L}_i(\theta) = \frac{1}{n_i^0} \sum_{x \in \mathcal{D}_i^0} \|x - f_{\theta}(x)\|_2^2, \quad (2)$$

where, $\mathcal{D}_i^0$ is the legitimate subset held by institution i , n_i^0 its size, $n^0 = \sum_i n_i^0$, and f_{θ} the autoencoder. Institutions are weighted by the volume of legitimate traffic they contribute, since that is the data on which the autoencoder is trained. Local objectives use legitimate transactions only; fraud labels are used exclusively downstream and never during the federated training of the autoencoder.

C. Stage One: The Federated Autoencoder

The encoder maps 29 inputs through 64, 32 and 16 units to an 8-unit bottleneck, and the decoder mirrors it back to 29 outputs. Hidden layers use rectified linear activations [55] with dropout of 0.15, the output layer uses a sigmoid to match the range of the min-max scaled inputs, and the loss is the mean squared reconstruction error. The model has 9797 trainable parameters, or 0.037 MB at single precision. The bottleneck forces the network to retain only the structure common to legitimate transactions, so that a fraudulent transaction survives the round trip poorly.

One architectural decision is specific to federation. Batch normalization maintains running estimates of the mean and variance of each activation, and those estimates are non-linear functions of the local data, so averaging them across clients produces statistics that correspond to no client's data. This is the incompatibility documented as the FedBN problem [54]. Layer normalization normalizes each sample across its own features, keeps no running state, and therefore has nothing to aggregate incorrectly, so it is used throughout.

Each selected client trains for five epochs with the AdamW optimizer at a learning rate of 0.001 and weight decay of 0.01, gradient norm clipping at 1.0, a warmup-cosine schedule over the local epochs, and a batch size of 512. Weight decay and clipping bound how far a client can move the parameters within a round, which limits both benign client drift and the leverage a malicious client obtains from one round of training.

D. Stage Two: Per-Feature Errors and Nearest Neighbours

Given a trained autoencoder and a transaction x , the per-feature squared error vector is formed as in Eq. (3),

$$e(x) = ((x_1 - \hat{x}_1)^2, \dots, (x_d - \hat{x}_d)^2) \in \mathbb{R}^d, \quad \hat{x} = f_{\theta}(x). \quad (3)$$

Threshold-based detectors collapse this vector to its mean and compare it against a cut-off, which discards the information that distinguishes fraud types: two transactions with the same total error but errors concentrated in different features are different events. Classification uses a k nearest neighbour vote over error vectors with $k = 5$, the Manhattan metric and inverse distance weighting. Let $\mathcal{N}_k(x)$ denote the k training samples nearest to $e(x)$ under the Manhattan distance, and let $w_j = 1/\|e(x) - e_j\|_1$ be the weight of neighbour j . The fraud score of a transaction is

$$s(x) = \frac{\sum_{j \in \mathcal{N}_k(x)} w_j y_j}{\sum_{j \in \mathcal{N}_k(x)} w_j} \quad (4)$$

and the transaction is flagged as fraudulent when the score of Eq. (4) satisfies $s(x) > 0.5$. The Manhattan distance sums

per-feature deviations without letting one large component dominate, which suits an error vector whose informative signal may be spread across several components.

The autoencoder is federated, whereas the classifier requires labelled error vectors, and four placements of it are evaluated. Under the *frozen* protocol, the classifier is fitted once on errors produced by the centralized autoencoder and applied unchanged to the federated model, which fixes the decision geometry and isolates the quality of the representation. Under *pooled refit*, it is fitted on the labelled error vectors of all institutions computed with the federated model, which requires each institution to transmit a per-feature transform of its transactions together with the label. Under the *local* protocol, each institution fits the same rule on its own error vectors and nothing is transmitted. Under the *federated head* protocol, a small classifier replaces the nearest neighbour rule and is trained by federated averaging, so that only parameter updates leave the institution. The local and federated head placements therefore satisfy the constraint of Section III-B, whereas pooled refit trades it for accuracy and the frozen protocol serves as an analytical control.

The head maps the error vector through the fixed transform $z_j = (\log_{10}(e_j + 10^{-7}) + 7)/7$, which needs no statistics shared across institutions, then through one hidden layer of 64 rectified units to a fraud probability, for 1985 parameters, which fit in a single CKKS ciphertext. It is trained with class-weighted binary cross-entropy, a weight of 3 on the fraud class, for 30 rounds of five local epochs with the Adam optimizer at a learning rate of 0.03. Its decision threshold is chosen without sharing any vector. Each institution scores its own validation shard, a random tenth of the validation data, and reports only the counts $TP_i(\tau)$, $FP_i(\tau)$ and $FN_i(\tau)$ on a grid of thresholds from 0.02 to 0.98 in steps of 0.01. The server sums the counts and selects the threshold of Eq. (5).

$$\tau^* = \arg \max_{\tau} \frac{2 TP(\tau)}{2 TP(\tau) + FP(\tau) + FN(\tau)}, \quad (5)$$

where, $TP(\tau) = \sum_i TP_i(\tau)$, and $FP(\tau)$ and $FN(\tau)$ are summed in the same way. The head architecture and its optimizer settings were chosen on validation counts for one seed and were not tuned on the test split.

E. Federated Protocol

With S_t the set of clients participating in round t , federated averaging forms the global model as in Eq. (6),

$$\theta^{t+1} = \sum_{i \in S_t} \frac{n_i^0}{\sum_{j \in S_t} n_j^0} \theta_i^t, \quad (6)$$

weighting each client by the count of legitimate transactions it trained on, so that institutions contribute in proportion to the evidence they hold. At the start of each round the server samples $|S_t| = \max(1, \lfloor pN \rfloor)$ clients uniformly without replacement, with participation rate p . This models the availability of institutions and is a sampling mechanism rather than a straggler cut-off, since the server selects the round's cohort in advance.

As an ablation, the FedProx proximal term is added [14], which anchors each local solution to the current global model,

$$\mathcal{L}_i^{\text{prox}}(\theta) = \mathcal{L}_i(\theta) + \frac{\mu}{2} \|\theta - \theta^t\|_2^2, \quad \mu = 0.01. \quad (7)$$

Federated averaging has no defence against a client that reports an arbitrary parameter vector, since the mean is unbounded in each coordinate. The two coordinate-wise rules of Yin et al. [17] are therefore also implemented. The trimmed mean with trim fraction $\tau = 0.2$ sorts the client values in each coordinate, discards the $\lfloor \tau |S_t| \rfloor$ largest and smallest, and averages the remainder, which tolerates up to $\lfloor \tau |S_t| \rfloor$ Byzantine clients by construction. The coordinate-wise median takes the median across participating clients. Algorithm 1 states the complete round.

Algorithm 1 Privacy-preserving federated training, with encrypted weighted-sum aggregation; the robust rules of Section III-E operate on plaintext updates

```

Server initializes  $\theta^0$ , publishes public and evaluation keys
for  $t = 0$  to  $T - 1$  do
   $S_t \leftarrow$  sample  $\max(1, \lfloor pN \rfloor)$  clients without replacement
  Broadcast  $\theta^t$  to every  $i \in S_t$ 
  for each client  $i \in S_t$  in parallel do
     $\theta_i \leftarrow \theta^t$ 
    for  $E$  local epochs over minibatches of  $\mathcal{D}_i^0$  do
      AdamW step on Eq. (2) plus Eq. (7), clipped
    end for
    Transmit  $E(\theta_i)$  as CKKS ciphertexts
  end for
   $\theta^{t+1} \leftarrow \mathcal{A}(\{E(\theta_i)\}_{i \in S_t})$ 
end for
Fit the nearest neighbour stage on per-feature errors

```

F. Encrypted Aggregation

The CKKS scheme [49] is used, implemented with TenSEAL [50]. The polynomial modulus degree is 8192, the coefficient modulus bit sizes are 60, 40, 40 and 60, and the global scale is 2^{40} . That configuration packs 4096 slots into a ciphertext, so each client transmits three ciphertexts per round for its 9797 parameters. Each client flattens its updated parameters into a real vector, packs it into ciphertexts and transmits them. The server holds the public and evaluation keys only, and computes the weighted sum of Eq. (8),

$$E(\theta^{t+1}) = \sum_{i \in S_t} w_i \odot E(\theta_i^t), \quad w_i = \frac{n_i^0}{\sum_{j \in S_t} n_j^0}, \quad (8)$$

using the two operations of Eq. (9), namely addition of two ciphertexts and multiplication of a ciphertext by a plaintext scalar,

$$E(a) \oplus E(b) = E(a + b), \quad w \odot E(a) = E(w \cdot a). \quad (9)$$

Key custody defines the guarantee. The consortium holds the CKKS secret key, while the aggregation server holds only

the public key, which lets clients encrypt, and the evaluation keys, which let it compute. The server therefore reads neither an individual update nor the aggregate it produces, and the returned ciphertext is decrypted by the institutions before the next round. This single-key arrangement secures the framework against a server that follows the protocol and inspects what it receives, which is the adversary of Section III-G. A server acting together with a key-holding institution falls outside that model, and threshold or multi-key custody addresses it by requiring several institutions to cooperate before anything can be decrypted, at an expected ciphertext size and aggregation cost that are essentially unchanged, although this was not measured.

Encryption protects the confidentiality of the updates, and robust aggregation protects the integrity of the model they build. The two mechanisms are evaluated separately: encrypted aggregation with federated averaging in Section IV-D, and the coordinate-wise rules on plaintext updates in Section IV-G. The trimmed mean and the median compare individual client values in each coordinate, an operation that a server holding only ciphertexts under single-key CKKS cannot perform, since CKKS supports the weighted sum but not the comparison.

G. Threat Model

The assets are raw transaction records, which never leave their institution, and model updates, which do. Two adversaries are considered. An honest-but-curious server follows the protocol but attempts to infer properties of local data from what it receives [44], [40]; encrypted aggregation removes its access to individual updates and to the aggregate. Byzantine clients are compromised institutions that submit arbitrary updates. Let θ^t be the broadcast model and θ_i the honestly trained result. Three attacks are evaluated,

$$\theta_i \leftarrow \theta_i + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 I), \quad \sigma = 0.5, \quad (10)$$

$$\theta_i \leftarrow \theta^t - (\theta_i - \theta^t), \quad (11)$$

$$\theta_i \leftarrow \theta^t - \kappa(\theta_i - \theta^t), \quad \kappa = 5. \quad (12)$$

Eq. (10) models a faulty or noisy participant, Eq. (11) inverts the update without amplifying it, and Eq. (12) is the canonical gradient inversion attack [16], which drives the global model away from every honest optimum. Byzantine fractions of 10, 20 and 40 percent are evaluated, spanning the tolerance bound of the trimmed mean from below and above. The security guarantee is stated under three assumptions. The server is honest-but-curious, so it follows the protocol and does not collude with a key-holding institution. The aggregate is decrypted by the institutions and released to all participants, as in standard federated averaging. The institutions' own infrastructure is trusted. Within these assumptions, CKKS aggregation keeps every individual update confidential from the server. It does not by itself provide record-level privacy, which would require differential privacy, and membership or property inference from the released model remains possible for any participant [44], [45]. A server that deviates from the protocol, for example by returning a tampered model, lies outside the

model, as does collusion between the server and a key-holder, which threshold or multi-key custody would address (Section III-F). A compromised institution is considered only as a Byzantine contributor. The pooled classifier of Section III-D shares labelled error vectors, a deterministic transform of the transactions together with their labels, and should be governed like any shared labelled data, whereas the local and federated-head placements avoid this exposure. Inference-time evasion and record-level poisoning are left to other defence layers.

H. Heterogeneity Model and Cost Models

Heterogeneity is synthesized with the Dirichlet protocol [13], where proportions $(p_1, \dots, p_N) \sim \text{Dir}(\alpha \mathbf{1}_N)$ determine each client's share and $\alpha \in \{1.0, 0.5, 0.1\}$. Reporting α alone is insufficient, because a single draw at small α can concentrate most of the data in one silo. The realized partition is therefore also reported and its effective client count, defined in Eq. (13),

$$N_{\text{eff}} = \exp(H(p)), \quad H(p) = - \sum_{i=1}^N \frac{n_i}{n} \log \frac{n_i}{n}, \quad (13)$$

which equals N for a balanced partition and falls toward one as a client dominates. As a control, $\alpha = 0.1$ is also run with a floor of 500 samples per client, imposed by clipping the proportions from below and renormalizing.

With $|\theta|$ the parameter count, four bytes per value, m participants and T rounds, plaintext communication is $C_{\text{plain}} = 8Tm|\theta|$ bytes, the factor of two covering upload and broadcast. Under encryption both directions carry ciphertexts, so $C_{\text{HE}} = \alpha_c C_{\text{plain}}$ and $T_{\text{HE}} = \beta T_{\text{plain}}$, where α_c is the ciphertext expansion factor and β the compute overhead factor. The constants α_c and β are measured on this configuration and extrapolated the measured constants across the experiment matrix, which is valid because the parameter count is identical across configurations and CKKS addition is correct regardless of how many clients participate.

IV. EXPERIMENTAL RESULTS

A. Setup

The experiments use the European cardholder dataset [2]: 284807 transactions collected over two days in September 2013, of which 492 are fraudulent. Features are 28 principal components published in place of the raw attributes for confidentiality, plus the transaction amount. The time column is discarded, since it encodes seconds elapsed within the collection window.

Features are scaled to the unit interval with a min-max transform fitted on the training split alone. The data is split 70, 10 and 20 percent into training, validation and test, stratified on the label, giving 199364 training transactions with 345 frauds, 28481 validation transactions with 49 frauds, and 56962 test transactions with 98 frauds.

No oversampling is applied. Synthetic minority oversampling and its relatives are near-universal in this literature, and when applied before the split they scatter interpolated fraud points across training and test, so a model is evaluated partly

on synthetic neighbours of its own training data. Preserving the natural rate in every split keeps the evaluation faithful. Every test-set result reported here therefore describes performance on transactions the pipeline has never seen, in the proportion it would meet in production.

The consortium is simulated: ten institutions, each holding a disjoint partition of the training data drawn at random, or by the Dirichlet scheme of Section III-H, from this single dataset. All institutions therefore share the same principal-component feature space, and differences in customer population or fraud pattern between institutions are not represented. Training runs for 30 rounds of five local epochs at participation rates of 100, 80 and 60 percent. The autoencoder study comprises 19 federated configurations, trained across seeds for a total of 49 federated runs, together with the centralized baseline; the comparisons of Sections IV-H and IV-J add federated head and federated supervised runs and five centralized seeds. The data split is fixed once, with the same 98 test frauds in every run, and a seed changes the client partition, the weight initialization and the order of training batches. The stability, classifier-placement and neighbour-sensitivity analyses and the comparison with the centralized model use five seeds, and the participation, client-drift and adversarial studies use three seeds each, so their comparisons carry confidence intervals. The heterogeneity sweep is reported on the reference seed, with the multi-seed statistics in Tables IV and VII. Because the test set is fixed, seed intervals capture training variability and not the sampling variability of a test set with 98 frauds, which a bootstrap over test transactions addresses in Section IV-B.

Accuracy is reported once and used to rank nothing: at a positive rate of 0.173 percent it cannot separate a working detector from a broken one, and the centralized model scores 0.9996 for that reason. The F1-score is the headline metric. The area under the receiver operating characteristic curve and the area under the precision-recall curve (PR-AUC) accompany it, the latter being the more informative of the two at this class ratio, and confusion counts, precision and recall are given for the headline comparisons. Communication volume, round latency and wall-clock training time complete the picture.

B. Centralized, Federated and Encrypted Models

Table III reports the main comparison on the reference seed together with the confusion counts underlying each F1-score. The frozen and refit protocols address different questions. Under the frozen protocol the federated model attains 0.8391, 2.6 points below the centralized 0.8652, which indicates that federated training displaces the reconstruction errors relative to the geometry on which the centralized classifier was fitted. Under pooled refit, in which the classifier is fitted on the federated errors, the reference seed attains 0.8778, corresponding to 79 of 98 frauds detected with three false positives against 77 with three for the centralized model.

Across five seeds, the pooled refit attains 0.8625 ± 0.0134 and the centralized model, trained on the same seeds, 0.8667 ± 0.0079 . The paired difference is -0.0042 , with a 95 percent confidence interval of -0.0212 to 0.0128 ($t = -0.68$, $p = 0.53$), so the two are statistically indistinguishable. A bootstrap over test transactions, which captures the sampling

variability of 98 frauds that seeds cannot, gives a reference-seed difference of $+0.0127$ with an interval of -0.0154 to 0.0431 . The 0.8778 of the reference seed is the highest of the five federated seeds, and the five-seed comparison indicates parity between the pooled-refit and centralized pipelines.

The reference centralized run selects its epoch using a classifier fitted on training and validation errors jointly, which stops training at epoch 5. The five-seed centralized runs select the epoch with a classifier fitted on training errors alone and stop between epochs 25 and 90. Both rules yield comparable scores, 0.8652 and 0.8667.

The confusion counts localize the differences. The centralized model detects 77 of 98 frauds with three false positives, and the federated model at full participation detects 73 under the frozen protocol and 79 under pooled refit, again with three false positives. All configurations generate between three and five false alerts per 57000 transactions, so the variation in F1 is attributable almost entirely to missed frauds, between 19 and 28 of them. An institution can price this trade-off from the relative cost of an investigation and of a chargeback. PR-AUC ranks the models consistently with F1, from 0.8147 to 0.8449, with the centralized model at 0.8425.

With pooled error vectors, the federated pipeline therefore matches the centralized one within seed variation while raw records remain at their institutions. Encryption adds communication cost.

ROC-AUC is of limited informativeness for this pipeline. The nearest neighbour stage assigns a score of exactly zero to 99.74 percent of test transactions, including 12.2 percent of the frauds, so ROC-AUC primarily reflects the fraction of frauds with a fraudulent neighbour and varies little across seeds. F1 and PR-AUC are accordingly treated as the primary measures.

C. Participation Rate

Table IV reports the participation sweep over three seeds. Detection quality under the frozen protocol is stable across participation rates: the three confidence intervals overlap, with 80 percent participation spanning 0.8264 to 0.8553 and 60 percent spanning 0.8205 to 0.8351, so reducing attendance by two fifths leaves the detector's quality within the resolution of three seeds.

The refit column is flat within seed variation, at 0.8694, 0.8607 and 0.8715 for 100, 80 and 60 percent participation, against a five-seed standard deviation of 0.0134 at full participation (Section IV-H). Participation of 60 percent therefore costs no measurable refit quality, at 60 percent of the communication of full participation. In this configuration, three fifths of the ten institutions online per round is enough. The cohort is drawn uniformly at random in each round.

Round latency is the quantity that governs elapsed time in deployment, because institutions train in parallel and the server waits for the slowest of them. At 1.11 to 1.13 seconds per round, a full thirty-round training cycle occupies roughly 34 seconds of coordinated computation, to which transmission is added. Training the detector is therefore a short operation for the consortium, and the cost that matters is the traffic it generates.

TABLE III. MAIN COMPARISON OF CENTRALIZED, FEDERATED AND ENCRYPTED FEDERATED MODELS, REFERENCE SEED

Method	Classifier	TP	FP	FN	Prec.	Recall	F1	ROC-AUC	PR-AUC	Time (s)	Comm. (MB)
Centralized	refit	77	3	21	0.9625	0.7857	0.8652	0.9387	0.8425	154.0	0
Federated, $p = 100\%$	frozen	73	3	25	0.9605	0.7449	0.8391	0.9336	0.8293	381.2	22.4
	pooled refit	79	3	19	0.9634	0.8061	0.8778	0.9336	0.8439		
Federated, $p = 80\%$	frozen	73	5	25	0.9359	0.7449	0.8295	0.9285	0.8147	316.2	17.9
	pooled refit	73	3	25	0.9605	0.7449	0.8391	0.9387	0.8444		
Federated, $p = 60\%$	frozen	70	3	28	0.9589	0.7143	0.8187	0.9336	0.8290	253.7	13.5
	pooled refit	77	3	21	0.9625	0.7857	0.8652	0.9387	0.8449		
Fed. with CKKS, $p = 100\%$	predictive columns as the plaintext rows (derived, not re-run)									385.4	574.0
Fed. with CKKS, $p = 80\%$	predictive columns as the plaintext rows (derived, not re-run)									319.7	459.2
Fed. with CKKS, $p = 60\%$	predictive columns as the plaintext rows (derived, not re-run)									256.5	344.4

a. Reference configuration, seed 42, 98 test frauds among 56962 transactions; three-seed means for the same federated settings are in Table IV and five-seed means in Table VIII. The pooled refit requires labelled error vectors to be pooled across institutions (Section III-D). Training time aggregates the work of all participating clients, and per-round latency is given in Table IV. The encrypted rows apply the measured constants of Table V to the plaintext run, so their predictive columns are identical by construction, and encryption was not executed inside the 30-round training loop.

TABLE IV. PARTICIPATION SWEEP, MEAN AND STANDARD DEVIATION OVER THREE SEEDS

Part.	F1 (frozen)	F1 (refit)	Latency (s)	Comm. (MB)
100%	0.8337 $\pm$ 0.0047	0.8694 $\pm$ 0.0059	1.13	22.4
80%	0.8409 $\pm$ 0.0128	0.8607 $\pm$ 0.0161	1.11	17.9
60%	0.8278 $\pm$ 0.0064	0.8715 $\pm$ 0.0051	1.11	13.5

TABLE V. MEASURED COST OF CKKS AGGREGATION

Quantity	Measured value
Ciphertext expansion α_c	25.6 $\times$
Compute overhead β	1.011 $\times$
Plaintext payload per client per round	38.3 kB
Ciphertext payload per client per round	979.6 kB
Encryption, per client	10.7 ms
Homomorphic aggregation, server	19.5 ms
Decryption of the aggregate	3.7 ms
One-time key bundle	33.8 MB
Error against plaintext averaging	1.8e-7

D. Cost of Encrypted Aggregation

The cost of encrypted aggregation was measured directly. A standalone CKKS implementation encrypts the trained model's parameters as ten client updates, performs the weighted aggregation homomorphically, decrypts the result and compares it against the plaintext average. Table V reports the measurement.

Two results matter more than the headline factor. Encryption is bandwidth bound rather than compute bound: the whole cryptographic pipeline adds about 130 ms to a round in which local training already takes seconds, a 1.1 percent increase in training time, while the same round grows from 0.75 MB to 19.1 MB of traffic. A deployment therefore budgets network capacity rather than processor time. Correctness is verified element-wise, with the encrypted aggregate matching the plaintext aggregate to 1.8e-7, seven orders of magnitude below the scale of the reconstruction errors and therefore incapable of changing a neighbour ordering. A deployment

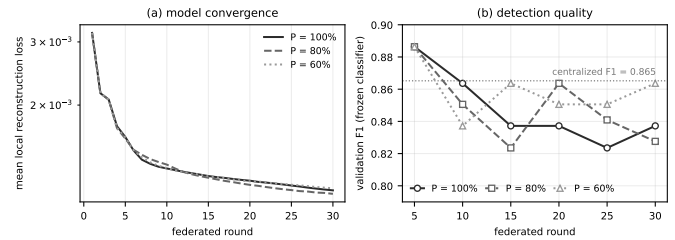


Fig. 2. (a) Mean local reconstruction loss per round, logarithmic scale, and (b) validation F1-score under the frozen classifier. The model converges; the detection metric does not improve with it.

planner should also account for the 33.8 MB key bundle transmitted once per session, which adds six percent over a thirty-round run.

The practical reading is that encryption here is purchased with bandwidth, on terms a banking network can meet, and that one aggregation leaves the average unchanged to within the measured error.

Fig. 2 separates two quantities that a single curve would confuse. The reconstruction loss falls steeply over the first ten rounds and flattens after fifteen, so the autoencoder converges. The validation F1-score under the frozen classifier does not follow it: detection quality is at its highest at the first evaluation point, round five, and thereafter oscillates within about three points with no upward trend at any participation rate. Part of that oscillation is measurement noise, since the validation split contains 49 frauds and a single sample moves the F1-score by roughly one point, but the absence of a trend is not noise. A deployment that stopped at ten rounds would therefore transmit a third of the 574 MB while retaining the detection quality of the full schedule, which is the largest single saving available in this design.

E. Data Heterogeneity

Table VI reports detection quality together with the realized partition, and this pairing is the point of the subsection. Taken at face value, the F1 column says heterogeneity is harmless

TABLE VI. NON-IID RESULTS WITH THE
REALIZED PARTITION

Partition	Largest share	N_{eff} of 10	F1 (frozen)	F1 (refit)
IID	10.0%	10.00	0.8391	0.8778
$\alpha = 1.0$	29.3%	6.92	0.8539	0.8729
$\alpha = 0.5$	42.0%	4.78	0.8409	0.8715
$\alpha = 0.1$	75.9%	1.92	0.8492	0.8652
$\alpha = 0.1$, floor 500	74.7%	2.12	0.8343	0.8778
$\alpha = 0.1$, $p = 60\%$	75.9%	1.92	0.8092	0.8840

a. Reference configuration, seed 42, on the partitions drawn for that seed; the participation, drift and adversarial studies of Tables IV, VII and Section IV-F report three-seed means.

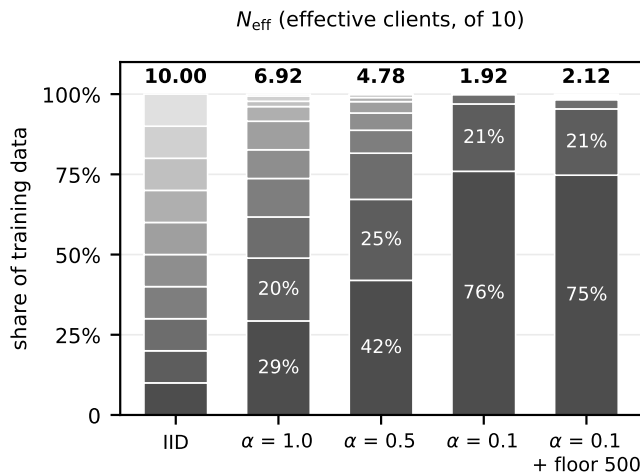


Fig. 3. Realized client mass under each partition, largest client first, with the effective client count above each bar. The two most heterogeneous partitions are two-client federations in all but name.

and occasionally helpful, since $\alpha = 0.1$ scores above the IID baseline on the frozen protocol. That reading does not hold, and the effective client count shows why. At $\alpha = 0.1$ a single client holds 75.9 percent of the training data and four clients hold one sample each, so the effective client count is 1.92 out of ten and the global model is approximating centralized training on one large silo, which is an easier problem. The apparent benefit of extreme heterogeneity reflects the partition rather than the method.

This was tested directly by rerunning $\alpha = 0.1$ with a floor of 500 samples per client. The floor removes the degenerate clients and the frozen F1-score falls by 1.5 points, consistent with the dominant client effect having inflated the unfloored result. The floor does not rebalance the partition, however: the largest client still holds 74.7 percent and the effective client count rises only to 2.12. Both arms remain, in substance, two-client federations, as Fig. 3 shows.

This is why the realized partition is reported alongside the concentration parameter. At ten institutions a single Dirichlet draw determines how the data actually lands, and N_{eff} captures that outcome in one number, so a reader can see immediately how many institutions carry the training mass in each arm. The practice costs a single column and makes every heterogeneity

TABLE VII. BYZANTINE ROBUSTNESS, MEAN AND
STANDARD DEVIATION OVER THREE SEEDS

Attack	Aggregation	Byz.	F1	$\Delta F1$
Gaussian	averaging	10%	0.8618 ± 0.0170	+0.0282
Gaussian	averaging	20%	0.8049 ± 0.0381	-0.0287
Gaussian	trimmed mean	20%	0.8412 ± 0.0083	+0.0076
Sign-flip	averaging	10%	0.8357 ± 0.0129	+0.0020
Sign-flip	averaging	20%	0.8378 ± 0.0159	+0.0042
Boosted	averaging	20%	0.0000	-0.8337
Boosted	trimmed mean	20%	0.8402 ± 0.0062	+0.0066
Boosted	averaging	40%	0.0000	-0.8337
Boosted	trimmed mean	40%	0.0000	-0.8337

claim in this study verifiable, and the practice is recommended for federated studies at comparable client counts.

The last row of Table VI combines extreme heterogeneity with partial participation. It yields the lowest frozen F1-score in the study, 0.8092, whereas its refit F1-score of 0.8840 lies within the seed standard deviation of 0.0134 of the other rows. The protocols diverge because this configuration displaces the error distribution furthest from the centralized one, which the frozen classifier penalizes, while the representation remains separable and a classifier fitted to it recovers the quality of the other rows.

F. Client Drift Correction

FedProx improves the frozen F1-score on IID participation at 80 percent from 0.8409 ± 0.0128 to 0.8609 ± 0.0030 over three seeds, and on the most heterogeneous partition from 0.8492 to 0.8566 ± 0.0031 . The first of these is the only pairwise difference in the optimizer and participation sweeps whose confidence intervals do not overlap, and the proximal term also cuts the seed-to-seed spread by a factor of four. The gain is larger on the IID partition than on the heterogeneous one. This is attributed to the way the proximal term operates in this setting: with five local epochs on large shards, unconstrained local training overshoots the shared optimum, and anchoring each client to the broadcast model damps that overshoot. Under $\alpha = 0.1$ the average is already dominated by one institution, so there is correspondingly less drift for the term to correct. FedProx is therefore worth enabling for its stabilizing effect on local optimization, independently of the heterogeneity it was designed to address.

G. Byzantine Robustness

Table VII reports three attacks at 10, 20 and 40 percent compromised clients under undefended averaging and under the trimmed mean. Deltas are computed against the clean baseline averaged over the same three seeds, 0.8337 ± 0.0047 , and not against a single clean run.

Unstructured attacks do not work, and saying so is part of the result. Gaussian perturbation and plain sign inversion at 10 to 20 percent leave the F1-score statistically unchanged, because weighted averaging attenuates an additive perturbation by roughly the attacker's data share and a negated update is cancelled by the honest majority. A study that tests only these attacks will conclude that federated averaging is robust.

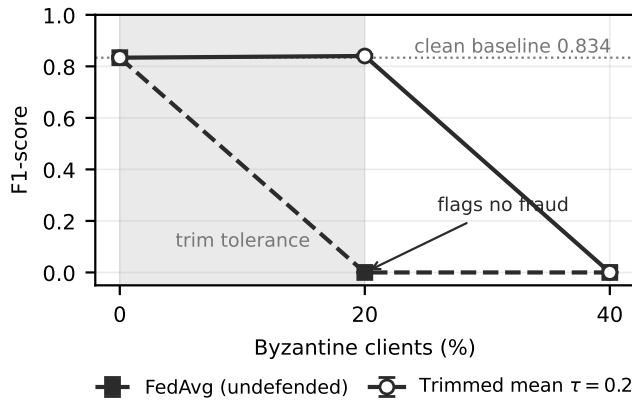


Fig. 4. Defence efficacy under the boosted sign-flip attack. Undefended averaging collapses to zero at 20 percent compromised clients, while the trimmed mean holds the clean baseline until the trim tolerance is exceeded and both fail at 40 percent. Error bars are one standard deviation over three seeds.

The canonical attack destroys the model completely. Boosting the inverted update by a factor of five drives the F1-score to exactly zero at 20 percent compromised clients. The global model no longer flags a single fraudulent transaction, and the same holds at 40 percent. Two malicious members out of ten are enough.

The defence works, and its stopping point can be stated precisely. The trimmed mean restores the F1-score to 0.8402 ± 0.0062 under the attack that annihilates averaging, statistically indistinguishable from the clean baseline, and fails completely at 40 percent, consistent with the tolerance bound. It also stabilizes the Gaussian arm, raising 0.8049 ± 0.0381 to 0.8412 ± 0.0083 and cutting the seed-to-seed spread by a factor of four. The result therefore delimits a boundary and does not assert general robustness. With ten clients and a trim fraction of 0.2, the rule discards the two largest and two smallest values in every coordinate and by construction tolerates up to two compromised clients. The experiments are consistent with this bound, with detection preserved at two of ten and lost at four of ten. Fig. 4 plots this curve.

Under clean conditions the trimmed mean incurs no measurable cost, and it preserved the detector when two of ten members executed the attack.

One result in Table VII is worth a mechanism. Gaussian noise on a single client raises the frozen F1-score by 2.8 points, reproducibly and on every seed. The effect is regularization rather than robustness: a perturbation on one client enters the average scaled by that client's weight of roughly one tenth, and the resulting jitter widens the reconstruction error distribution in a way that the frozen classifier, whose decision geometry was fitted on centralized errors, exploits. It appears under the frozen protocol only, which is a further illustration of the distinction the two protocols are designed to expose.

H. Stability and the Placement of the Classifier

Over five seeds at full participation and under the frozen protocol the F1-score is 0.8322 ± 0.0043 , with a 95 percent

TABLE VIII. PLACEMENT OF THE CLASSIFIER STAGE, FIVE-SEED MEAN AND STANDARD DEVIATION

Classifier	Leaves the bank	F1	Prec.	Recall	ROC	PR
Local, per bank	nothing	0.7254 ± 0.0117	0.853	0.634	0.899	0.686
Federated head	parameter updates	0.7655 ± 0.0157	0.737	0.798	0.964	0.721
Pooled refit	labelled error vectors	0.8625 ± 0.0134	0.962	0.782	0.938	0.852
Centralized	raw records	0.8667 ± 0.0079	0.960	0.790	0.936	0.845

a. The local row averages the ten banks, each scored on the full test set; the head is trained with plaintext federated averaging, and its parameter updates fit in one CKKS ciphertext but were not encrypted in this run. The centralized row uses the epoch selection described in Section IV-B.

confidence interval of 0.8284 to 0.8360, while the area under the curve is 0.9336 with a standard deviation below 0.00001. The contrast is informative: the federated autoencoder learns an essentially identical ranking every time, and what varies is where the nearest neighbour vote places the decision boundary within that ranking. The representation is therefore highly reproducible. F1 differences are meaningful at a resolution of about half a point under the frozen protocol, and about one point once the classifier is refitted, where the five-seed standard deviation is 0.0134.

The classifier stage requires labelled error vectors, and their location is the principal privacy decision of the pipeline. Table VIII compares the four placements of Section III-D over the same five seeds, the same federated autoencoders and the same test set.

Local fitting attains a mean F1-score of 0.7254 across seeds. On the reference seed the ten banks range from 0.6780 to 0.7486, against 0.8778 for the pooled fit, and over all seeds and banks the range is 0.6225 to 0.8068. Each bank holds approximately 19900 training transactions but only 23 to 46 frauds, and a five-neighbour vote over so few positive exemplars generalizes poorly to unseen fraud types. The local classifier is precise and attains a recall of 0.634, which is the scarcity argument for federation recurring at the second stage.

The federated head recovers part of this gap without any institution transmitting an error vector. Its F1-score of 0.7655 lies 4.0 points above the local classifier and 9.7 points below the pooled refit. The difference with the pooled refit is concentrated in precision: the head detects 78.2 of 98 frauds on average, slightly more than the 76.6 of the pooled classifier, but generates 28.2 false alerts per 56962 transactions against 3.0. Its ROC-AUC of 0.964 is the highest in the table and its PR-AUC of 0.721 is markedly below the 0.852 of the pooled classifier, consistent with a head that ranks frauds well but lacks a sharp operating point at this class ratio. A consortium that cannot share labelled error vectors may therefore expect an F1-score near 0.77, and one that can, near 0.86. The head was tuned on validation counts for one seed, and further tuning may narrow this difference.

I. Nearest Neighbour Hyperparameters

The default of $k = 5$, the Manhattan metric, inverse-distance weighting and a decision threshold of 0.5 was fixed when the pipeline was designed, and the sweep of Table IX

TABLE IX. TEST F1-SCORE OF THE POOLED REFIT BY
NEIGHBOUR COUNT AND METRIC, THRESHOLD 0.5,
FIVE-SEED MEAN

Metric	$k = 1$	$k = 3$	$k = 5$	$k = 7$	$k = 11$	$k = 21$
Manhattan	0.8385	0.8649	0.8625	0.8570	0.8482	0.8342
Euclidean	0.8329	0.8460	0.8546	0.8529	0.8438	0.8282
Cosine	0.7651	0.8017	0.7930	0.7897	0.7908	0.7926

evaluates it after the fact, over the same five federated models and the pooled-refit protocol.

The F1-score is stable with respect to k and within a metric family. With the Manhattan metric it equals 0.8649, 0.8625 and 0.8570 at $k = 3, 5$ and 7 , and the Euclidean metric lies within 0.8 points at $k = 5$. The cosine distance is approximately 7 points lower, since it discards the magnitude of the errors on which detection relies. The decision threshold exerts a larger effect. At $k = 5$ with the Manhattan metric, lowering the threshold from 0.5 to 0.4 raises the F1-score from 0.8625 to 0.8884, as recall increases from 0.782 to 0.837 while precision decreases only from 0.962 to 0.947. A threshold of 0.3 yields 0.8889, and 0.2 reduces it to 0.8249 as false alerts accumulate. Selecting the metric, k and threshold on the validation split for each seed gives a mean test F1-score of 0.8803, 1.8 points above the default, with the Euclidean metric, $k = 5$ and a threshold of 0.3 chosen in four of five seeds. The default of $k = 5$, Manhattan distance and a threshold of 0.5 was fixed at design time and is retained throughout as a high-precision operating point.

J. Comparison with Alternative Detectors

Table X scores alternative detectors on the identical split, with the same five seeds for stochastic methods, the same metrics, and thresholds selected on the validation split. The baselines received light tuning only. The nearest neighbour stage retains its fixed threshold of 0.5, whereas the baseline thresholds are tuned on validation (Section IV-I).

Detectors that use no labels attain substantially lower F1-scores: 0.245 for Isolation Forest and 0.443 and 0.434 for the autoencoder with a scalar threshold, federated and centralized respectively, against 0.8625 for the same autoencoder followed by the per-feature nearest neighbour stage. The error-vector classifier therefore contributes most of the discriminative power. The pooled pipeline also exceeds the centralized supervised models, at 0.8625 against 0.835 for the random forest and 0.802 for gradient boosting, a margin of 2.8 points that corresponds to roughly two standard deviations of the pipeline. Among methods that exchange only parameter updates, a federated multilayer perceptron on raw features attains 0.783, 1.7 points above the federated head and within about one standard deviation of it, so the two perform comparably on these IID partitions. The label-free representation, which a supervised model does not provide, is intended for settings in which labels are scarce, a regime that this comparison, in which all labels are available, does not exercise.

K. Limitations

The evaluation relies on ten simulated institutions derived from a single public dataset spanning two days of transactions

TABLE X. ALTERNATIVE DETECTORS ON THE IDENTICAL
SPLIT, FIVE-SEED MEAN AND STANDARD DEVIATION

Detector	Leaves the bank	F1	Prec.	Recall	PR
Isolation Forest	raw records	0.245 ± 0.029	0.196	0.337	0.146
AE, scalar threshold, federated	updates	0.443 ± 0.005	0.481	0.410	0.404
AE, scalar threshold, centralized	raw records	0.434 ± 0.007	0.411	0.478	0.402
Logistic regression	raw records	0.710	0.647	0.786	0.701
Gradient boosting	raw records	0.802 ± 0.035	0.951	0.696	0.833
Random forest	raw records	0.835 ± 0.005	0.953	0.743	0.860
Federated logistic regression	updates	0.764 ± 0.002	0.726	0.806	0.693
Federated MLP	updates	0.783 ± 0.006	0.737	0.835	0.724
AE and federated head (this work)	updates	0.766 ± 0.016	0.737	0.798	0.721
AE and pooled k-NN (this work)	error vectors	0.863 ± 0.013	0.962	0.782	0.852

a. The first group is label-free, the second centralized supervised on pooled raw records, the third federated supervised on raw features with the same ten banks and the same threshold rule as the head. Logistic regression is deterministic, so it has no spread.

and described by anonymized principal components. Differences between banks in customer population, transaction mix and fraud pattern are therefore not represented, and no second dataset is used. The split is random rather than chronological, so temporal drift is not assessed. With 98 test frauds, F1 comparisons have limited resolution, as the bootstrap of Section IV-B shows. Three non-adaptive attacks are considered, at 10, 20 and 40 percent, which places the breaking point of the trimmed mean between two and four of ten clients and leaves adaptive attackers unevaluated. The encryption constants originate from a standalone benchmark of a 9797-parameter model across ten members and are applied to the plaintext training runs, and the costs for larger models, larger federations and wide-area links are extrapolated from a linear model. The baselines are lightly tuned and use IID partitions only. The findings accordingly apply to this configuration.

V. DISCUSSION

A. Why the Results Behave as They Do

Federating the autoencoder is inexpensive because the task is to learn what is normal. A supervised fraud classifier is expected to federate badly when fraud types differ across institutions, since each client pulls the decision boundary toward the attacks it has seen. The clients here do not learn a boundary; they learn to reconstruct legitimate spending, which is the part of the distribution that institutions have most in common, and averaging ten reconstructions of a shared distribution is a benign operation. The IID partitions of Section IV-J do not test that expectation, since a federated supervised model performed comparably to the federated head there, so the argument remains a hypothesis for heterogeneous silos.

Encryption is bandwidth bound because the model is small and the training is not. Cryptographic cost scales with the

TABLE XI. WHAT EACH CHOICE BUYS AND WHAT IT COSTS

Choice	What it buys	What it costs	Evidence
Classifier on pooled error vectors	F1 of 0.8625, level with the centralized 0.8667	each bank shares labelled error vectors, a transformed form of its data	Table VIII
Federated classifier head	only parameter updates leave the bank; F1 of 0.7655	about 10 F1 points below pooled; 28 false alerts per 56962 transactions against 3	Table VIII
Local classifier at each bank	nothing leaves the bank	F1 of 0.7254; recall of 0.634, so more than a third of frauds are missed	Table VIII
CKKS encrypted aggregation	the server never sees an update	25.6 times the traffic (22.4 MB to 574.0 MB per run) and 1.1 percent more time; no trimmed mean on ciphertexts	Table V
Trimmed mean aggregation	F1 of 0.8402 with 2 of 10 banks attacking, where averaging drops to 0	works on plaintext updates only; fails at 4 of 10	Table VII
Participation at 60 percent	40 percent less traffic with F1 inside seed variation	only random absences were tested	Table IV
Stopping at round ten	a third of the traffic	validation F1 showed no trend after round five, so the saving rests on validation only	Fig. 2
Threshold of 0.3 to 0.4 instead of 0.5	F1 of about 0.889 instead of 0.8625	precision falls from 0.962 to about 0.92 to 0.95	Section IV-I
FedProx with $\mu = 0.01$	steadier training; seed spread cut about fourfold at 80 percent participation	one more hyperparameter, and a smaller gain on skewed partitions	Section IV-F

parameter count, so the cryptographic term stays small while the training term does not. The figure of 1.1 percent is therefore a property of small models and would not survive porting the pipeline to a network with millions of parameters, whereas the factor of 25.6 would. Likewise, unstructured attacks fail against weighted averaging because a perturbation enters scaled by the attacker's data share, while boosting the inverted update defeats that mechanism and points away from every honest optimum at the same time. The choice of attack therefore determines the conclusion, and a robustness claim tested only against noise is not a robustness claim.

B. The Tradeoffs in Practice

Each design choice carries a measurable benefit and cost, summarized in Table XI. The placement of the classifier is the dominant trade-off: pooling labelled error vectors yields a detector on par with the centralized one, whereas keeping them local or exchanging only parameter updates lowers F1 by roughly 10 points, mainly through additional false alerts. Encryption conceals every update from the server at roughly 25 times the traffic and precludes the trimmed mean. The remaining choices are inexpensive: reduced participation, an early stop and a lower decision threshold each reduce traffic or raise F1 at modest cost.

C. Deployment and Hardware

The measurements are enough to size a deployment. A thirty-round run at full participation moves 574 MB of ciphertext plus a one-time 33.8 MB key bundle, or 19.1 MB per round across the consortium, which a cross-silo banking network can be expected to absorb. The multiplier of 25.6 sounds alarming and is operationally modest at cross-silo scale, whereas ten million handsets on cellular links would be a different verdict entirely, and this is where cross-silo and cross-device federation part company [6]. The conclusion about the affordability of homomorphic encryption is a conclusion about the cross-silo regime.

Homomorphic aggregation of ten client updates takes 19.5 ms. If server cost scales linearly with membership, a hundred-member consortium would need roughly 200 ms of aggrega-

tion per round for about 98 MB of ingest, an extrapolation that was not measured here. The workload is strongly duty-cycled, seconds of activity per round followed by idleness while institutions train, which suits on-demand provisioning, one of the defining characteristics of cloud computing [56].

Homomorphic encryption also makes third-party hosting defensible, since the operator computes on ciphertexts and holds no key, whereas a member-operated aggregator reintroduces the trust asymmetry federation was meant to remove. A secure enclave reaches a comparable property by computing on plaintext inside hardware-isolated memory [57]. Enclaves have no ciphertext expansion and support the comparisons that the trimmed mean requires, but their security rests on a hardware vendor's implementation, which transient-execution attacks have repeatedly broken [58]. An enclave that performs robust aggregation on data that arrives encrypted would combine both properties and is the most promising route past the composition limitation of Section III-F.

D. Comparison with Prior Work

Placing these results beside the published literature requires care, because the most cited numbers are not comparable to these. Studies on this benchmark routinely report accuracies above 99 percent [23], and federated studies report similar figures [7], [8]. At a positive rate of 0.173 percent, however, 99.83 percent accuracy is what a classifier achieves by predicting legitimate every time, and where oversampling is applied before the split the reported score partly measures memorization. Both were avoided here, and the cost of that discipline is a five-seed F1-score of 0.8667 centralized and 0.8625 federated, rather than a number in the nineties.

A cross-paper comparison of F1-scores on this dataset is meaningful only when the splits, the sampling policy and the metric are identical, which is rarely the case, so the comparison is made on the present split (Section IV-J). The federated pipeline with a pooled classifier matches the centralized model within seed variation and, among the baselines tried, exceeds the centralized supervised models by up to 2.8 points, and encrypted transport, partial participation, skewed silos and compromised members were each evaluated against it. Six

evaluation dimensions are reported on a single pipeline: communication cost, measured encryption overhead, partial participation, the realized non-IID partition, Byzantine participants and multi-seed confidence intervals. The measurements also sharpen three widely held expectations. Homomorphic encryption is bandwidth bound rather than compute bound at this model size. Federated averaging needs a robust aggregation rule to withstand a gradient inversion attack. And a Dirichlet parameter alone does not describe the partition it generates. Relative to the centralized two-stage line of work that this detector extends [33], the pipeline survives federation with its detection quality intact when the classifier stage pools labelled error vectors, and loses about ten points of F1-score when it does not, at quantified costs.

VI. CONCLUSION

Financial institutions cannot pool transaction records, and no single institution observes enough fraud to characterize it alone. This study evaluated a federated detector and quantified the cost of each design choice.

Detection quality is governed mainly by the placement of the classifier. With pooled labelled error vectors, the federated pipeline attains an F1-score of 0.8625 over five seeds against 0.8667 for the centralized model, with a paired difference of -0.0042 and a 95 percent interval from -0.0212 to 0.0128 . A federated head that exchanges only parameter updates attains 0.7655, a locally fitted classifier 0.7254, and a federated network on raw features 0.783. On the same split, label-free detectors remain below 0.45 and a centralized random forest attains 0.835. The representation is learned without fraud labels, and the classifier stage uses them.

Encryption is bandwidth-bound rather than compute-bound. CKKS aggregation expands traffic by a factor of 25.6, to 574 MB over 30 rounds, and increases training time by about one percent, while a single encrypted aggregation reproduces plaintext averaging to $1.8e-7$.

Undefended averaging collapses to an F1-score of zero when two of ten institutions mount a boosted gradient inversion attack. The trimmed mean sustains 0.8402 and fails at four of ten, in line with its trim tolerance of two clients. It operates on plaintext updates and is not combined with encrypted aggregation.

A participation rate of 60 percent reduces traffic by 40 percent without measurable loss, and validation F1 shows no gain after the fifth round. Reporting the realized partition shows that a Dirichlet concentration of 0.1 corresponds to an effective client count of 1.92 out of ten.

The study thus provides a quantified set of trade-offs (Table XI) from which a consortium can select an operating point between privacy, robustness and accuracy, obtained on ten simulated institutions and one public dataset.

Three extensions are natural. Threshold or multi-key CKKS [61] would distribute key custody so that several institutions must cooperate to decrypt. The additional cost is expected to be small, since ciphertext size and aggregation time do not change in principle, but it has not been measured. Running robust aggregation inside a trusted execution environment [57] could combine encrypted transport with the coordinate-wise rules.

The defence boundary of Section IV-G also invites evaluation against adaptive adversaries [62] and stronger aggregation rules [16], [63], [64].

DECLARATIONS

The authors declare no conflict of interest. The dataset analysed in this study is the publicly available European cardholder credit card fraud dataset [2]. No funding was received for this work.

REFERENCES

- [1] Nilson Report, "Card fraud losses worldwide 2024," Nilson Report, Carpinteria, CA, Jan. 2026. [Online]. Available: <https://nilsonreport.com>
- [2] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *Proc. IEEE Symp. Series Comput. Intell. (SSCI)*, 2015, pp. 159–166.
- [3] I. Karkaba, E. M. Adnani, and M. Erritali, "Deep learning detecting fraud in credit card transactions," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 9, pp. 3557–3565, 2023.
- [4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist. (AISTATS)*, PMLR 54, 2017, pp. 1273–1282.
- [5] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.
- [6] P. Kairouz *et al.*, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021.
- [7] N. F. Aurna *et al.*, "Federated learning-based credit card fraud detection: Performance analysis with sampling methods and deep learning algorithms," in *Proc. IEEE Int. Conf. Cyber Security and Resilience (CSR)*, 2023, pp. 180–186.
- [8] M. Abdul Salam, K. M. Fouad, D. L. Elbably, and S. M. Elsayed, "Federated learning model for credit card fraud detection with data balancing techniques," *Neural Comput. Appl.*, vol. 36, no. 11, pp. 6231–6256, 2024.
- [9] T. Awosika, R. M. Shukla, and B. Pranggono, "Transparency and privacy: The role of explainable AI and federated learning in financial fraud detection," *IEEE Access*, vol. 12, pp. 64551–64560, 2024.
- [10] Y. Chen, K. Zhang, H. Zhu, and Z. Qiu, "A novel federated transfer learning framework for credit card fraud detection under heterogeneous data conditions," *Risks*, vol. 13, no. 11, p. 208, 2025.
- [11] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: Efficient homomorphic encryption for cross-silo federated learning," in *Proc. USENIX Annu. Tech. Conf. (ATC)*, 2020, pp. 493–506.
- [12] Y. Zuo *et al.*, "Pack: Towards communication-efficient homomorphic encryption in federated learning," in *Proc. ACM Symp. Cloud Comput. (SoCC)*, 2024, pp. 980–997.
- [13] T. M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv:1909.06335*, 2019.
- [14] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst. (MLSys)*, vol. 2, 2020, pp. 429–450.
- [15] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-IID data silos: An experimental study," in *Proc. IEEE 38th Int. Conf. Data Eng. (ICDE)*, 2022, pp. 965–978.
- [16] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017, pp. 119–129.
- [17] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, PMLR 80, 2018, pp. 5650–5659.
- [18] K. Modi and R. Dayma, "Review on fraud detection methods in credit card transactions," in *Proc. Int. Conf. Intell. Comput. Control (I2C2)*, 2017, pp. 103–107.

- [19] A. Aftab, I. Shahzad, A. Sajid, M. Anwar, and N. Anwar, "Fraud detection of credit cards using supervised machine learning techniques," *Pakistan J. Eng. Sci. Technol.*, vol. 4, no. 3, pp. 38–51, 2023.
- [20] S. Dhankhad, E. Mohammed, and B. Far, "Supervised machine learning algorithms for credit card fraudulent transaction detection: A comparative study," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, 2018, pp. 122–125.
- [21] S. Khatri, A. Arora, and A. P. Agrawal, "Supervised machine learning algorithms for credit card fraud detection: A comparison," in *Proc. 10th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence)*, 2020, pp. 680–683.
- [22] Y. Liu, Y. Wang, and J. Zhang, "New machine learning algorithm: Random forest," in *Proc. Int. Conf. Inf. Comput. Appl. (ICICA)*, 2012, pp. 246–252.
- [23] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, and A. Anderla, "Credit card fraud detection: Machine learning methods," in *Proc. 18th Int. Symp. INFOTEH-JAHORINA*, 2019, pp. 1–5.
- [24] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, "Random forest for credit card fraud detection," in *Proc. IEEE 15th Int. Conf. Netw., Sens. Control (ICNSC)*, 2018, pp. 1–6.
- [25] V. Vinodhini and A. Sivaraj, "Credit card fraud detection using random forest algorithm," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 7, no. 3, 2019.
- [26] K. Fu, D. Cheng, Y. Tu, and L. Zhang, "Credit card fraud detection using convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. (ICONIP)*, 2016, pp. 483–490.
- [27] M. Rezapour, "Anomaly detection using unsupervised methods: Credit card fraud case study," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 11, 2019.
- [28] A. K. Rai and R. K. Dwivedi, "Fraud detection in credit card data using unsupervised machine learning based scheme," in *Proc. Int. Conf. Electron. Sustain. Commun. Syst. (ICESC)*, 2020, pp. 421–426.
- [29] G. Montavon, J. Kauffmann, W. Samek, and K.-R. Müller, "Explaining the predictions of unsupervised learning models," in *xxAI - Beyond Explainable AI*, Springer, 2022, pp. 117–138.
- [30] M. Zamini and G. Montazer, "Credit card fraud detection using autoencoder based clustering," in *Proc. 9th Int. Symp. Telecommun. (IST)*, 2018, pp. 486–491.
- [31] T.-H. Lin and J.-R. Jiang, "Credit card fraud detection with autoencoder and probabilistic random forest," *Mathematics*, vol. 9, no. 21, p. 2683, 2021.
- [32] S. Chen and W. Guo, "Auto-encoders in deep learning: A review with new perspectives," *Mathematics*, vol. 11, no. 8, p. 1777, 2023.
- [33] H. Fanai and H. Abbasimehr, "A novel combined approach based on deep autoencoder and deep classifiers for credit card fraud detection," *Expert Syst. Appl.*, vol. 217, p. 119562, 2023.
- [34] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, p. 106775, 2021.
- [35] S. Bharati, M. R. H. Mondal, P. Podder, and V. B. S. Prasath, "Federated learning: Applications, challenges and future directions," *Int. J. Hybrid Intell. Syst.*, vol. 18, no. 1–2, pp. 19–35, 2022.
- [36] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.
- [37] S. M. Mathews and S. A. Assefa, "Federated learning: Balancing the thin line between data intelligence and privacy," *arXiv:2204.13697*, 2022.
- [38] Noerlina, T. N. Mursitama, R. Kurniawan, and A. Bandur, "Centralized versus decentralized technology in the financial sector," in *Proc. Int. Conf. Inf. Manage. Technol. (ICIMTech)*, 2023, pp. 296–301.
- [39] S. Aliyev, "A survey on challenges of federated learning," *Azerbaijan J. High Perform. Comput.*, vol. 5, no. 2, pp. 273–285, 2022.
- [40] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantaha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021.
- [41] E. Ntizikira, W. Lei, F. Alblehai, K. Saleem, and M. A. Lodhi, "Secure and privacy-preserving intrusion detection and prevention in the internet of unmanned aerial vehicles," *Sensors*, vol. 23, no. 19, p. 8077, 2023.
- [42] A. Brecko, E. Kajati, J. Koziorek, and I. Zolotova, "Federated learning for edge computing: A survey," *Appl. Sci.*, vol. 12, no. 18, p. 9124, 2022.
- [43] R. D'souza, S. Sharma, and A. Chaudhari, "Scalable federated learning for privacy-preserving credit card fraud detection," in *Proc. IEEE Int. Carnahan Conf. Security Technol. (ICCST)*, 2023, pp. 1–6.
- [44] B. Rao, J. Zhang, D. Wu, C. Zhu, X. Sun, and B. Chen, "Privacy inference attack and defense in centralized and federated learning: A comprehensive survey," *IEEE Trans. Artif. Intell.*, vol. 6, no. 2, pp. 333–353, 2025.
- [45] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Federated Learning: Privacy and Incentive*, LNCS 12500, Springer, 2020, pp. 17–31.
- [46] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory of Cryptography Conf. (TCC)*, LNCS 3876, 2006, pp. 265–284.
- [47] W. Mou, C. Fu, Y. Lei, and C. Hu, "A verifiable federated learning scheme based on secure multi-party computation," in *Proc. Int. Conf. Wireless Algorithms, Syst., Appl. (WASA)*, LNCS 12938, 2021, pp. 198–209.
- [48] C. Fang, Y. Guo, Y. Hu, B. Ma, L. Feng, and A. Yin, "Highly efficient federated learning with strong privacy preservation in cloud computing," *Comput. Secur.*, vol. 96, p. 101889, 2020.
- [49] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. ASIACRYPT*, LNCS 10624, 2017, pp. 409–437.
- [50] A. Benaissa, B. Retiat, B. Cebere, and A. E. Belfedhal, "TenSEAL: A library for encrypted tensor operations using homomorphic encryption," *arXiv:2104.03152*, 2021.
- [51] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *J. Netw. Comput. Appl.*, vol. 116, pp. 1–8, 2018.
- [52] P. Vepakomma, T. Swedish, R. Raskar, O. Gupta, and A. Dubey, "No peek: A survey of private distributed deep learning," *arXiv:1812.03288*, 2018.
- [53] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Security and Privacy (SP)*, 2017, pp. 19–38.
- [54] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated learning on non-IID features via local batch normalization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.
- [55] A. Zilouchian, "Fundamentals of neural networks," in *Intelligent Control Systems Using Soft Computing Methodologies*, CRC Press, 2001, pp. 17–38.
- [56] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, USA, Special Publication 800-145, Sep. 2011.
- [57] Y. Chen, F. Luo, T. Li, T. Xiang, Z. Liu, and J. Li, "A training-integrity privacy-preserving federated learning scheme with trusted execution environment," *Inf. Sci.*, vol. 522, pp. 69–79, 2020.
- [58] J. Zhou, Y. Xiao, and Y. Zhang, "ENCLYZER: Automated analysis of transient data leaks on Intel SGX," in *Proc. IEEE Int. Symp. Secure and Private Execution Environment Design (SEED)*, 2022, pp. 145–156.
- [59] European Parliament and Council, "Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data (General Data Protection Regulation)," *Official Journal of the European Union*, L 119, pp. 1–88, May 2016.
- [60] PCI Security Standards Council, *Payment Card Industry Data Security Standard: Requirements and Testing Procedures*, version 4.0.1, Jun. 2024.
- [61] A. Al Omar, H. Wang, and M. Rahman, "Efficient privacy-preserving cross-silo federated learning with multi-key homomorphic encryption," *arXiv:2505.14797*, 2025.
- [62] M. Baruch, G. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 8635–8645.
- [63] J. Xu, S.-L. Huang, L. Song, and T. Lan, "SignGuard: Byzantine-robust federated learning through collaborative malicious gradient filtering," *arXiv:2109.05872*, 2021.

- [64] E. Kritharakis, D. Katsaros, and N. Tziritas, "FedGreed: A Byzantine-robust loss-based aggregation method for federated learning," *arXiv:2508.18060*, 2025.
- [65] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.