POISONING CROSS-VIEW CORRESPONDENCE IN VERTICAL FEDERATED LEARNING VIA CONSISTENT LATENT-CLUSTER REASSIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Vertical Federated Learning (VFL) is a privacy-preserving collaborative training paradigm in which participants holding disjoint feature sets over a shared sample space jointly train a global model without exposing raw data. Although VFL security has attracted growing research interest, the literature remains heavily skewed toward targeted threats, particularly backdoor and reconstruction attacks. Byzantine attacks, a family of untargeted poisoning threats that degrade global model accuracy, have been extensively studied in horizontal federated learning but remain largely unexplored in the vertical setting. This gap is especially consequential: the classical two-party design of many VFL systems allows a single compromised passive party to unilaterally destabilize the entire system. In this paper, we introduce a novel Byzantine attack strategy for vertical settings that performs consistent cluster-based swapping, constructing a poisoned cross-view association during training. The global model gets corrupted and catastrophically fails on clean, correctly aligned data at inference time. Since existing VFL defenses fail to detect this attack without incurring substantial performance penalties, we propose a targeted defense that attributes and repairs the corrupted associations. Our findings establish untargeted Byzantine attacks as a real, underappreciated availability threat to VFL and motivate robustness mechanisms tailored to its unique cross-view learning dynamics.

1 INTRODUCTION

Federated learning (FL) enables multiple data owners to train a joint model without sharing their raw private data (McMahan et al., 2023; Yang et al., 2019). Unlike horizontal FL, where participants hold different samples in a common feature space, vertical federated learning (VFL) connects organizations that observe the same entities through disjoint feature sets (Cheng et al., 2021; Liu et al., 2024; Ye et al., 2025). A bank and an e-commerce platform, for example, may hold complementary information about overlapping customers but be unable to pool it directly. VFL allows them to learn jointly while keeping the underlying features local.

A standard neural VFL pipeline first aligns the entities shared across the participating organizations through a privacy-preserving protocol (Lu & Ding, 2020; Xi et al., 2025). Each party then transforms its local features into an intermediate representation, while the label-owning active party or server combines the representations, computes the loss, and returns gradients (Liu et al., 2024; Ye et al., 2025). The model therefore learns not only from the information contained in each feature view, but also from the relationships among views associated with the same entity. Once alignment is completed, however, training generally assumes that every party continues to provide the correct local representation for each aligned sample.

This assumption creates an overlooked integrity risk. A compromised passive party controls its local data, bottom model, and transmitted representations, allowing it to manipulate collaborative training without changing labels or accessing honest features. Existing VFL security research has predominantly examined confidentiality leakage or targeted integrity attacks, including backdoors, model hijacking, and adversarial inference (Fu et al., 2022; Liu et al., 2024; Yu et al., 2024; Bai et al., 2023; Chen et al., 2023; Xuan et al., 2023; He et al., 2024a; Naseri et al., 2024; Chen et al., 2024a;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Shen et al., 2025). These attacks recover private information or manipulate selected inputs toward attacker-chosen outputs. They do not fully characterize an availability threat in which a passive party seeks to undermine the model's general utility on ordinary, correctly aligned data.

Untargeted Byzantine poisoning provides a natural framework for this threat, but existing solutions do not directly address the structure of VFL. Horizontal FL defenses rely on comparable client updates over a shared parameter space (Blanchard et al., 2017; Yin et al., 2018; Fang et al., 2020; Cao et al., 2021), an assumption VFL's disjoint feature spaces do not satisfy. The VFL-specific studies that do exist (Yuan et al., 2022; Xu et al., 2024; Oonishi & Nakai, 2025) still locate corruption within a single transmitted value, update, or contribution rather than the correspondence between parties, which is generally assumed to remain correct once alignment is complete.

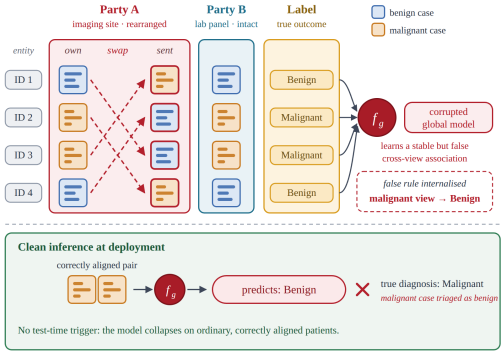


Figure 1: Overview of the Consistent Cross-View Swap (CCVS) attack.

We study this correspondence itself as the attack surface. A malicious party can provide a genuine local representation from the wrong semantic region while maintaining a persistent false association with the honest view and label. Unlike random reassignment, which mainly introduces unstructured noise, a semantically organized and consistently repeated reassignment creates a stable false dependency that the model can learn. The global model is consequently optimized for a corrupted cross-view distribution and fails when correctly aligned views are restored at inference time, causing broad degradation without a trigger or test-time intervention.

Based on this observation, we propose CONSISTENT CROSS-VIEW SWAP POISONING (CCVS), an untargeted Byzantine attack designed around VFL's dependence on aligned views. The same setting creates a difficult recovery problem. Detecting and discarding the malicious party may suppress the attack, but in the classical and widely used two-party VFL configuration, it also removes the only complementary feature view (Fu et al., 2022; Liu et al., 2024). We therefore introduce REFERENCE-GUIDED ATTRIBUTION AND RECONSTRUCTION (RGAR), a repair-oriented defense that uses a small authenticated reference set to model honest cross-view relationships. RGAR accumulates sample-level inconsistencies to attribute the corruption and reconstructs the expected representation of the malicious party rather than discarding its entire contribution. Our main contributions are as follows:

- We formulate *cross-view association poisoning* as an untargeted Byzantine threat in VFL, distinguishing corruption of sample correspondence from targeted backdoors and value-level perturbations.
- We develop CCVS, which combines limited auxiliary supervision, semi-supervised clustering, and persistent cluster reassignment to degrade clean inference without a trigger, target class, label modification, or test-time control.
- We evaluate CCVS across vision, sensor, and tabular tasks against alternative poisoning strategies and representative horizontal and vertical FL defenses, reporting multi-seed variance and sensitivity to swap coverage, auxiliary label budget, and reference-set size.
- We introduce RGAR, which attributes the corrupted party and reconstructs its expected contribution from a small authenticated reference set, preserving useful feature views without full retraining.

2 RELATED WORK

Targeted attacks on VFL. Most attacks studied in VFL revolve around targeted objectives. Backdoor methods construct a trigger-target association using whatever information is available to the passive party, including labelled auxiliary data and local latent representations (Gu & Bai, 2023), poisoned representations injected at the split-learning boundary (Bai et al., 2023), gradient-traced target-

class samples (Xuan et al., 2023), or clean-label triggers under more restricted information (Chen et al., 2023). Practical backdoor injection by a single malicious party has been demonstrated against both classical and split-neural-network VFL architectures (Naseri et al., 2024; He et al., 2024a; Chen et al., 2024a; Shen et al., 2025). A related line of work hijacks the model toward chosen predictions rather than a fixed trigger, generating adversarial embeddings from inferred labels (He & Chang, 2025) or corrupting selected clients' inputs after deployment (Yao et al., 2024). Despite their different mechanisms, these methods are evaluated on inputs or outputs selected by the attacker and are generally designed to preserve ordinary model behaviour. CCVS pursues a fundamentally different objective: broad failure on correctly aligned clean data, without a trigger, target class, or test-time action.

Untargeted Byzantine attacks that degrade global accuracy without pursuing a specific target are well studied in HFL (Fang et al., 2020; Blanchard et al., 2017; Yin et al., 2018). Krum measures distances among client updates, coordinate-wise rules suppress anomalous parameters, and FLTrust compares each update with a trusted server direction (Cao et al., 2021). These defenses rely on update homogeneity. Because every HFL client trains over the same feature space, client updates are directly comparable, making it possible to identify an update as outlying or insufficiently aligned with the trusted direction (Shi et al., 2022; So et al., 2020). This assumption does not hold in VFL. Parties encode disjoint feature spaces by design, so their embeddings naturally differ in distribution, dimension, and semantic meaning. Consequently, applying Krum-style or FLTrust-style filtering to VFL may penalize ordinary feature heterogeneity as readily as malicious behaviour.

Byzantine attacks in VFL: practical gaps. A small number of studies adapt Byzantine threat modelling directly to VFL (Yuan et al., 2022; Xu et al., 2024; Oonishi & Nakai, 2025). Yuan et al. (2022) propose a Byzantine-resilient subgradient method tolerant to arbitrary corrupted values. Xu et al. (2024) recalculate gradients and cross-validate models to detect Byzantine clients, while Oonishi & Nakai (2025) estimate client contributions to identify sign-flipping and related manipulations. Chen et al. (2024b) generate adversarial perturbations against a surrogate anomaly detector. These methods locate Byzantine behaviour in a transmitted value, such as a corrupted quantity, flipped sign, or adversarial perturbation, which can be examined through one party's contribution in isolation. None of them targets the association learned between the parties' views, a quantity unique to VFL because it fuses heterogeneous representations rather than aggregating comparable updates. Noisy-correspondence methods in multi-view learning treat misaligned pairs as random noise to be tolerated; CCVS instead constructs structured misalignment chosen to maximize clean-test loss. In CCVS, individual embeddings remain locally plausible; the corruption lies in the persistent false correspondence between those embeddings, the honest view, and the label. Such an attack offers little direct abnormality for a value-level detector to identify.

Defenses against VFL poisoning. Existing VFL defenses mainly detect, filter, or purify suspicious representations. VFedAD identifies poisoned samples through information-theoretic and contrastive analysis (Lai et al., 2023). VFLIP detects backdoor-triggered embeddings at inference time and reconstructs them from the remaining honest views (Cho et al., 2024). It is the closest existing defense to RGAR in spirit, but it operates after training on a fixed model rather than addressing corruption as it accumulates during training. SVFLDetector and CC-VFed instead detect abnormal optimization behaviour or client contribution without repairing the corrupted information (Xu et al., 2024; Oonishi & Nakai, 2025).

These approaches do not naturally address cross-view association poisoning. Filtering a suspicious instance discards all of its feature views, while removing a malicious party eliminates an entire feature block, which is especially costly when no alternative passive view exists. Similarly, purifying an isolated embedding does not restore a correspondence that has been repeatedly corrupted throughout training. RGAR is designed for this gap. A small authenticated reference set provides trustworthy cross-view relationships, accumulated inconsistency identifies the responsible party, and the party's expected representation is reconstructed rather than discarded. Unlike VFLIP, RGAR attributes and repairs corruption while training is still underway. Unlike FLTrust, which evaluates homogeneous horizontal updates against a trusted direction, RGAR models the conditional relationship between heterogeneous vertical views.

A preliminary short study (Tasneem, 2026) demonstrated distant-cluster feature swapping in VFL; we formalize the threat, optimize the swap, evaluate it systematically, and introduce the first repair-based defense in this context.

3 PRELIMINARIES

3.1 VERTICAL FEDERATED LEARNING

Let K clients be indexed by $c \in \{1, \dots, K\}$ over N aligned samples. For sample i, client c holds a private feature view $x_i^{(c)} \in \mathbb{R}^{d_c}$, encoded through a bottom model $h_i^{(c)} = f_c(x_i^{(c)}; \theta_c)$. The server or active party concatenates the embeddings and applies a head model, $\hat{y}_i = g\Big(\|_{c=1}^{K} h_i^{(c)}; \theta_g\Big)$ and trains the system end-to-end by minimizing a supervised loss $\mathcal{L}(\hat{y}_i, y_i)$ over $\Theta = \{\theta_1, \dots, \theta_K, \theta_g\}$; only the server holds labels $y_i \in \mathcal{Y}$. During back-propagation the server returns each client its embedding gradient $\nabla_{h_i^{(c)}} \mathcal{L}$, so VFL exposes only intermediate representations and gradients, never raw features.

3.2 BYZANTINE ATTACKS IN VFL

A Byzantine attack is an adversarial deviation by a subset of malicious clients $\mathcal{B} \subset \{1, \dots, K\}$ that corrupts the global model. In HFL, Byzantine clients manipulate local model updates before aggregation; in VFL, clients instead submit intermediate representations from heterogeneous feature spaces, so a malicious client corrupts training by poisoning its local input view or by perturbing its transmitted embedding directly. The threat considered here is *untargeted*: the adversary seeks to degrade clean inference-time performance after training.

3.3 THREAT MODEL

Our threat model follows all the classic VFL protocols and carries out most experiments in the classic two-party ($K = 2$) VFL setting, the dominant real-world deployment (Yang et al., 2023; Vepakomma et al., 2018) and the adversarially worst case: the malicious party controls half the joint representation. The adversarial party A holds $X^{(A)} \in \mathbb{R}^{N \times d_A}$ and only has access to their own bottom model f_A. Party A cannot access the feature-set or local model of other benign parties, label $\mathbf{y}$, or any message between the server and benign parties. Party A additionally holds a small stratified auxiliary set

$$\mathcal{D}_L = \big\{(x_i^{(A)}, y_i)\big\}_{i \in \mathcal{I}_L}, \quad |\mathcal{I}_L| = \lfloor \varepsilon N \rfloor, \quad \varepsilon \in [0.02, 0.03], \tag{1}$$

with at least one sample per class. This is strictly weaker than full label access, and is consistent with realistic partial-label leakage scenarios in VFL (Fu et al., 2022; He et al., 2024b), contractual outcome sharing between collaborating institutions, and partially public outcome records.

Adversarial objective. Party A replaces its training features with a rearranged set $\widetilde{X}^{(A)}$, where $\widetilde{x}_i^{(A)} = x_{j(i)}^{(A)}$ for some $j(i) \in [N]$. The server, oblivious to the poisoning, fits:

$$g^\star = \operatorname*{argmin}_g \ \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\big(g\big(\widetilde{h}_i^{(A)} \| \, h_i^{(B)}\big), \, y_i\big), \quad \widetilde{h}_i^{(A)} = f_A(\widetilde{x}_i^{(A)}). \tag{2}$$

$$\max_{\widetilde{X}^{(A)}} \ \mathbb{E}_{P_{\text{clean}}}\Big[\mathcal{L}\big(g^\star\big(f_A(x^{(A)}) \| \, f_B(x^{(B)})\big), \, y\big)\Big], \tag{3}$$

The attacker maximises the *clean-test* loss of $g^\star$ forcing $g^\star$ to internalize a spurious cross-view association that collapses at inference on clean, correctly aligned data.

RGAR: one-time reference setup. RGAR requires the server to hold a small statically authenticated reference set before training begins:

$$\mathcal{R} \subset [N], \quad |\mathcal{R}| = \lfloor r_{\text{ref}} N \rfloor, \quad r_{\text{ref}} \in [0.05, 0.10], \quad |\mathcal{R} \cap \{i : y_i = c\}| \geq 1 \ \ \forall c \in \mathcal{Y}. \tag{4}$$

Party A submits $\{x_i^{(A)}\}_{i \in \mathcal{R}}$ to the server once at onboarding before compromise or under third-party audit as a condition of participation, analogous to the root dataset of FLTrust (Cao et al., 2021), so the stored copy reflects its true features; the server keeps it immutable and no labels are disclosed to Party A. During training, the server obtains reference embeddings $h_i^A = f_A(x_i^{(A)})$ by returning this copy to Party A for a forward pass. Following Eq. (2), the adversary manipulates its inputs but executes f_A faithfully, which attested execution can enforce; forged reference embeddings are outside

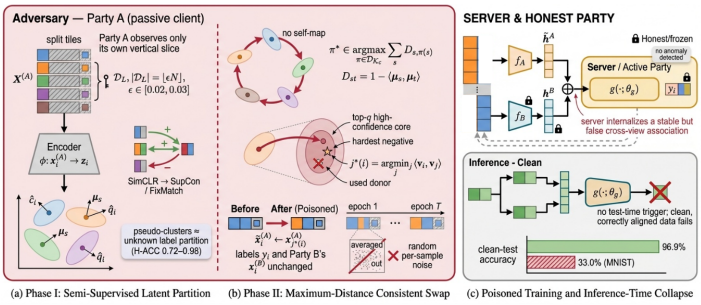


Figure 2: Detailed methodology of the Consistent Cross-View Swap (CCVS) attack.

our scope. We adopt the conservative assumption that Party A knows $\mathcal{R}$. Sharing raw features for this small subset is a trust concession beyond standard VFL; the exchange is operationally comparable to the pre-training entity alignment mandated in production VFL systems (Yu et al., 2024), and provides the calibration anchor for RGAR's scoring, attribution, and reconstruction stages.

4 Methodology

To achieve the Byzantine objective, we design a two-stage attack pipeline: the first stage constructs high-fidelity clusters of the training samples without observing the true labels, and the second uses these clusters to guide a consistent feature-swapping attack that distorts the association between feature subsets throughout training.

4.1 Phase I: Semi-supervised Clustering via Latent Representation

Phase I infers a latent partition of the adversary's local feature view. Given the adversarial training view $X^{(A)} = \{x_i^{(A)}\}_{i=1}^N$ and a small auxiliary labeled subset $\mathcal{D}_L$, the adversary learns an embedding function and outputs pseudo-cluster assignments and confidence scores: $\phi : \; x_i^{(A)} \mapsto z_i \in \mathbb{R}^p$, $\hat{c}_i \in \{1, \ldots, K_c\}$, and $\hat{q}_i \in [0, 1]$. The goal is to recover enough class-correlated structure from the adversary's own view so that Phase II can construct semantically misleading class-consistent cross-view pairs.

Representation learning. Because the adversary observes only a partial vertical view, Phase I uses modality-matched representation learning. Vision datasets pretrain an encoder with SimCLR's NT-Xent objective (Chen et al., 2020) and refine it on $\mathcal{D}_L$ with supervised contrastive learning (Khosla et al., 2020); harder half-view and tabular settings instead use a FixMatch-style student–teacher pipeline (Sohn et al., 2020), where an EMA teacher supplies stable pseudo-labels. Concretely, MNIST and Fashion-MNIST use SimCLR→SupCon, CIFAR-10 substitutes FixMatch for SupCon, UCI-HAR uses tabular FixMatch, UCI-Mushroom fits a Bernoulli mixture with pseudo-label refinement, and UCI-Bank uses an unsupervised GMM tuned for donor diversity rather than label alignment (Appendix D). The cluster count K_c equals the number of classes, known from the task definition, except UCI-Bank ($K_c = 16$), where two clusters give too few distinct donors.

Clustering and confidence. Embeddings are L_2-normalized and clustered with a Gaussian or Bernoulli mixture model (McLachlan & Peel, 2000), or, for classifier-based pipelines, assigned by the teacher. The cluster identity $\hat{c}_i$ is the maximum-responsibility component after prototype-based merging and the confidence $\hat{q}_i$ is the corresponding soft membership. These pass to Phase II as a surrogate for the unknown label partition.

4.2 Phase II: Class-Consistent View Swap

Core mechanism. Guided by the predicted clusters from Phase I, the adversary constructs a poisoned training set by consistently exchanging its local feature view across clusters: because VFL depends on stable cross-party correlations between aligned views, disrupting them during training

teaches the head model a spurious association that fails to generalize. Let $\mathcal{G} = \{C_1, \ldots, C_{K_c}\}$ be the cluster partition from Phase I and $c(i) \in [K_c]$ sample i's assignment. For each source cluster C_s, the adversary identifies a disjoint destination cluster $C_{t(s)}$ and replaces every victim $i \in C_s$ with a *donor* sample $j(i) \in C_{t(s)}$:

$$\widetilde{x}_i^{(A)} \leftarrow x_{j(i)}^{(A)}, \quad j(i) \in C_{t(c(i))}, \quad \forall\, i \in [N]. \tag{5}$$

Server labels y_i and Party B's features $x_i^{(B)}$ remain unchanged. The server therefore trains on triples $(\widetilde{x}_i^{(A)}, x_i^{(B)}, y_i)$ whose two feature views are drawn from different class-conditioned manifolds, internalizing a stable but semantically inverted cross-view association.

Algorithm 1 CCVS: Two-Phase Cluster-Swap Attack on Party A's View

Require: Party A's features $\{x_i^{(A)}\}_{i=1}^N$; auxiliary set $\mathcal{D}_L$ ($\approx 2\%$–3% labelled); modality $\mathcal{M}$; confidence quantile q; top-k donor scope.

Ensure: Poisoned training view $\{\widetilde{x}_i^{(A)}\}_{i=1}^N$.

// Phase I: Semi-supervised clustering
Extract embeddings via modality pipeline $V \leftarrow \mathrm{Embed}_{\mathcal{M}}(X^{(A)}, \mathcal{D}_L)$.
Fit clusters $\{C_s\}_{s=1}^{K_c}$ and confidences $\hat{q}$ from V.
Compute centroids: $\mu_s \leftarrow \frac{1}{|C_s|} \sum_{i \in C_s} v_i \,/\, \|\cdot\|, \forall s$.
// Phase II: Maximum-distance cluster swap
Build distance matrix $D_{st} = 1 - \langle \mu_s, \mu_t \rangle$.
Solve $\pi^\star \in \mathrm{argmax}_{\pi \in \mathfrak{D}_{K_c}} \sum_s D_{s,\pi(s)}$ via Eq. (7).
for $s = 1$ to K_c **do**
  Union top-k donor cores: $U_s = \bigcup_{t \in \mathcal{T}(s)} \mathcal{D}_t^{(q)}$, where $\mathcal{T}(s)$ are the k farthest clusters.
  for each victim $i \in C_s$ **do**
    $j^\star(i) = \mathrm{argmin}_{j \in U_s,\, j \text{ unused}} \langle v_i, v_j \rangle$ (Eq. (8)).
    $\widetilde{x}_i^{(A)} \leftarrow x_{j^\star(i)}^{(A)}$.
  end for
end for
return $\{\widetilde{x}_i^{(A)}\}_{i=1}^N$.

Why cluster-level consistency is essential. Let $c_i \in [K_c]$ be the Phase I cluster of sample i and $c_i^{(B)}$ the corresponding structure in the honest view. Independent per-sample corruption destroys the association, $\mathrm{MI}\big(\tilde{c}^{(A)}; c^{(B)}\big) = 0$, so it averages out over mini-batches and leaves no stable signal to fit. A cluster-wise permutation instead applies the same map π to every sample in a cluster and in every epoch, and because π is a bijection it preserves the association exactly:

$$\mathrm{MI}\big(\pi(c^{(A)}); c^{(B)}\big) = \mathrm{MI}\big(c^{(A)}; c^{(B)}\big), \qquad \mathrm{MI}\big(\tilde{c}^{(A)}_{\text{noise}}; c^{(B)}\big) = 0. \tag{6}$$

Since Phase I clusters are a surrogate for the label partition, the poisoned view retains full predictive structure with respect to $X^{(B)}$ and y, but that structure now points to the wrong class. The adversarial view therefore stays locally valid, and the server fits the fabricated association rather than discarding it as noise.

Optimal cluster-to-cluster mapping. CCVS chooses the derangement $\pi \in \mathfrak{D}_{K_c}$ that maximizes total centroid distance,

$$\pi^\star \in \arg\max_{\pi \in \mathfrak{D}_{K_c}} \sum_{s=1}^{K_c} D_{s,\pi(s)}, \qquad D_{st} = 1 - \langle \mu_s, \mu_t \rangle, \qquad \mu_s = \frac{\sum_{i \in C_s} v_i}{\big\|\sum_{i \in C_s} v_i\big\|_2}, \tag{7}$$

solved exactly as a linear assignment problem on D with the diagonal excluded. The objective is the total distance, so an individual cluster need not receive its single farthest partner. Given $\pi^\star$, each victim $i \in C_s$ receives the donor of lowest cosine similarity within the high-confidence core of the destination cluster $t = \pi^\star(s)$:

$$j^\star(i) = \arg\min_{j \in \mathcal{D}_t^{(q)}} \langle v_i, v_j \rangle, \qquad \mathcal{D}_t^{(q)} = \{j \in C_t : \hat{q}_j \geq \mathrm{quant}(\hat{q}_{C_t}, q)\}. \tag{8}$$

Donors are drawn without replacement until the core is exhausted and then reused, so the sample-level mapping need not be bijective; the cluster-level map $\pi^\star$ is fixed for all epochs (Appendix E). At inference time, clean test pairs are correctly aligned and no longer follow the reinforced cross-cluster pattern the head model learned, causing the decision rule to fail. Algorithm 1 details the full two-phase pipeline.

5 DEFENSE

RGAR: Reference-Guided Attribution and Reconstruction. Byzantine defenses typically *detect and drop*, discarding, clipping, or zeroing suspicious embeddings (Blanchard et al., 2017; Cao et al., 2021; Cho et al., 2024). In VFL, where deployments have few parties, suppressing one removes a substantial share of the joint representation. RGAR instead detects unreliable cross-view pairs, attributes fault to a party, and repairs that party's view.

Calibration. From the trusted set $\mathcal{R}$ (Eq. (4)), the server forms reference triples (h_i^A, h_i^B, y_i), with h_i^A computed on the authenticated copy (Sec. 3.3). It fits class prototypes $\mathbf{p}_c^A, \mathbf{p}_c^B$, joint prototypes $\mathbf{p}_c^{\text{jnt}}$ over concatenated embeddings, and diagonal Mahalanobis variances (Mahalanobis, 1936), and trains a reconstructor $g_\theta(h_i^B, y_i) \to \hat{h}_i^A$, then frozen, with a SmoothL1 loss (Girshick, 2015):

$$\mathcal{L}_{\text{recon}} = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \text{SmoothL1}\big(g_\theta(h_i^B, y_i), h_i^A\big). \tag{9}$$

Scoring and attribution. Each sample receives

$$s_i^{\text{proto}} = \alpha \cdot \tfrac{\bar{d}_i^A + \bar{d}_i^B}{2} + (1-\alpha) s_i^{\text{jnt}}, \tag{10}$$

$$s_i^{\text{jnt}} = 1 - \cos\Big(\tfrac{[h_i^A; h_i^B]}{\|[h_i^A; h_i^B]\|}, \mathbf{p}_{y_i}^{\text{jnt}}\Big), \tag{11}$$

where $\bar{d}_i^k$ is the scaled diagonal Mahalanobis distance to $\mathbf{p}_{y_i}^k$, so the joint term catches cross-view inconsistency even when each view is locally plausible. Party evidence e_i^k adds temporal drift $1 - \cos(h_i^k, \text{EMA}_i^k)$ (Tarvainen & Valpola, 2017). For samples exceeding τ_{pair}, RGAR accumulates signed evidence $e_i^A - e_i^B$ per epoch; after burn-in, a mean above τ_{global} attributes corruption to Party A and decays its trust weight ρ_A toward a floor $\rho_{\min}$. The attribution is revoked if later evidence reverses.

Repair. The attributed party's embedding is then replaced by a blend

$$\begin{aligned} \hat{h}_i^A &= \beta\, g_\theta(h_i^B, y_i) + (1-\beta)\mathbf{p}_{y_i}^A, \\ \tilde{h}_i^A &= (1-w_i) h_i^A + w_i \hat{h}_i^A, \end{aligned} \tag{12}$$

with w_i growing in s_i^{proto} and raised after attribution, while Party B keeps $\rho_B = 1$. For the asymmetric HAR and Bank models, whose honest embeddings have 8 and 2 dimensions, RGAR omits the reconstructor and uses $\hat{h}_i^A = \mathbf{p}_{y_i}^A$, freezing the suspected encoder. For $K > 2$, one reconstructor per party is trained from the remaining views, with attribution to the party of largest accumulated evidence; collusion is out of scope. Appendix G gives the implemented form.

6 EXPERIMENTAL EVALUATION

6.1 EXPERIMENTAL SETUP

Datasets and partitions. We evaluate on six datasets spanning vision, sensor, and tabular modalities: MNIST (LeCun et al., 1998), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky, 2009), UCI-HAR (Anguita et al., 2013), UCI-Mushroom (Schlimmer, 1987), and UCI-Bank (Moro et al., 2014). Per-dataset partitions, K_c, and ε are listed in Appendix C.

Models and training. MNIST and Fashion-MNIST use lightweight local encoders with a two-layer MLP server head; CIFAR-10 uses ResNet-based bottom models (He et al., 2016) with concatenation fusion; tabular datasets use MLP bottom models. Details of all models and with specified settings are described in F.

Attacks and defenses. **CCVS** is our full method: maximum-distance derangement with hardest-negative donors; on UCI-Bank, whose 16 clusters are not label-aligned, it uses a class-aware variant that restricts destinations to clusters of opposite predicted class tendency. We compare against **Round-Robin** (fixed cyclic mapping), **Random Cluster** (uniform random derangement), and **Random Sample** (independent donors, no Phase I structure), all at equal coverage and training budget. As defenses, we compare RGAR against three detect-and-drop baselines that suppress rather than repair: **Batch Krum**, adapted from Krum aggregation (Blanchard et al., 2017); **Cosine Gate**, following FLTrust's trusted-reference cosine scoring (Cao et al., 2021); and **AE Gate**, following reconstruction-based VFL defense (Cho et al., 2024).

Metrics and protocol. Phase I quality is reported as H-ACC (Kuhn, 1955), NMI, ARI (Hubert & Arabie, 1985), and purity. Attack strength is clean-test accuracy after poisoned training (lower is stronger) and defense performance is clean-test accuracy after defended training; detection and suspicion rates are defined in Appendix H. Results are mean ± std over three VFL training seeds with Phase I clustering fixed, so the variance isolates training stochasticity; † marks single-seed entries. Hardware and runtimes are in Appendix I.

Table 1: Phase I clustering quality on the training split.

Dataset	N	K_c	H-ACC	NMI	ARI	Purity
MNIST	60,000	10	0.881	0.787	0.756	0.881
Fashion-MNIST	60,000	10	0.751	0.699	0.610	0.751
CIFAR-10	50,000	10	0.723	0.544	0.507	0.723
UCI-HAR	8,754	6	0.874	0.787	0.743	0.874
UCI-Mushroom	6,905	2	0.978	0.861	0.913	0.978
UCI-Bank	36,168	16	0.914	0.812	0.795	0.883

6.2 Phase I: Latent Partition Quality

Table 1 shows that Phase I recovers useful latent partitions from limited auxiliary supervision. Partition quality bounds attack quality but does not gate it: Phase II requires a geometrically coherent partition rather than an accurate class partition, which is why UCI-Bank remains an effective attack surface using $K_c = 16$ clusters chosen for donor diversity rather than label alignment.

Table 2: Post-attack global model accuracy (%) on clean test data. Lower is stronger. CCVS values are mean ± std over three training seeds; comparison strategies are single-seed point estimates (†).

Dataset	Clean	CCVS	Round Robin†	Rnd. Cluster†	Rnd. Sample†
MNIST	96.5±0.1	**32.5±2.2**	82.0	75.0	89.0
Fashion-MNIST	87.5±0.4	**26.2±1.9**	67.1	78.0	81.6
CIFAR-10	89.2±0.7	**52.2±1.4**	80.5	81.5	83.2
UCI-HAR	97.1±0.9	**15.7±2.1**	43.2	44.9	56.3
UCI-Mushroom	97.9±0.5	**32.8±2.6**	95.2	94.7	91.0
UCI-Bank	89.3±0.1	**39.7±1.2**	84.6	83.8	86.4

6.3 Phase II: Attack Effectiveness

Table 2 reports clean-test accuracy after poisoned training at full swap coverage. CCVS degrades global model accuracy across all six datasets without any test-time trigger, and multi-seed standard deviations below 2.6 points confirm the collapse is a property of the attack rather than a favorable initialization. The strategy comparison shows that Round-Robin and Random Cluster retain cluster-level consistency but discard the geometry of the mapping, and land in between. Only the full mapping, combining maximum-distance cluster assignment with hardest-negative donors, produces collapse.

Table 3: Defense comparison on clean-test accuracy (%) and detection rate (%). RGAR repairs corrupted representations, while baselines suppress flagged embeddings and fail to save the model from catastrophe. UCI-M refers to UCI-Mushroom

Dataset	Clean	No Defense	Krum		Cosine Gate		AE Gate		RGAR (ours)	
			Acc.	Det.	Acc.	Det.	Acc.	Det.	Acc.	Susp.
MNIST	96.8	33.3	38.2	6	46.9	100	33.6	3	**83.5**	100
F-MNIST	87.2	26.2	41.5	4	13.3	100	32.9	18	**73.6**	100
CIFAR-10	89.2	52.2	47.6	9	41.4	64	36.8	12	**84.7**	94
UCI-HAR	98.3	15.5	15.5	5	61.2	97	29.1	10	**89.4**	100
UCI-M	98.4	32.8	30.7	7	63.2	78	22.4	8	**92.4**	100
UCI-Bank	89.6	39.9	32.7	4	88.3	57	38.2	4	**85.3**	97

Table 4: Ablations (attack accuracy %, lower is stronger). (a) party count K. (b) auxiliary label fraction ε.

(a) Party count K

K	MNIST	HAR	Bank
2	32.5	15.7	39.7
4	57.0	25.2	66.3
8	96.2	37.6	74.4
10	97.0	44.9	81.3
12	97.9	49.6	89.9

(b) Aux. fraction ε

ε	MNIST	HAR	Bank
0.5%	72.7	57.8	62.2
1%	65.2	41.8	50.8
2%	32.5	37.2	39.7
3%	27.8	15.7	36.2
5%	24.5	13.6	31.4

6.4 Defense Evaluation

Table 3 shows the two ways detect-and-drop fails. RGAR is the only method that recovers on every dataset, because it reconstructs the corrupted view instead of suppressing it. Across three training seeds this recovery is stable, with standard deviations from 0.3 points on MNIST to 2.0 on UCI-HAR. This supports our central defense claim: in few-party VFL, representation repair dominates representation removal.

6.5 Ablation Studies

Party count and auxiliary labels. Table 4 shows that with a single attacker, CCVS becomes less effective as the number of parties increases. Our focus, however, is the classical two-party VFL setting, where we show that one compromised party can severely degrade the global model. Extending CCVS to colluding attackers in multi-party settings remains future work. The auxiliary label budget also affects attack strength, highlighting the value of limited supervision for constructing effective swaps. Such access to auxiliary labels is consistent with assumptions used in prior VFL attacks (Fu et al., 2022; Gu & Bai, 2023).

Reference-set size and integrity. Table 5 shows that reducing r_{ref} from 10% to 1% lowers recovery accuracy from 82.7% to 78.9% on MNIST and from 83.2% to 68.9% on HAR, both above their undefended baselines. With 10% reference corruption, accuracy is 82.2% and 90.2%, respectively. These results suggest tolerance to limited reference corruption.

7 Conclusion

We identify untargeted Byzantine poisoning as a distinct availability threat in VFL. CCVS corrupts cross-view correspondence through consistent cluster-level swaps, degrading clean-test accuracy without a trigger, while detect-and-drop defenses either miss it or discard critical features. RGAR instead attributes the malicious party and reconstructs its view. Our characterization is empirical; formal analysis of when consistent reassignment defeats a learned head remains future work.

8 Reproducibility and Code Availability

Data. All six datasets are public benchmarks obtained from their standard sources: MNIST, Fashion-MNIST, and CIFAR-10 through `torchvision`, and UCI-HAR, UCI-Mushroom, and UCI-Bank from the UCI Machine Learning Repository. No private, licensed, or newly collected data is used. Preprocessing, the vertical feature partitions, and the mutual-information ranking that defines them are specified in Appendix C; sample counts, cluster counts K_c, and auxiliary-label budgets ε are listed per dataset in Table 6.

Attack. Phase I representation learning and clustering are specified per modality in Appendix D, including the contrastive, supervised-contrastive, and FixMatch objectives and how cluster identities and confidences are derived. Phase II is fully determined by the maximum-distance derangement, the high-confidence donor core, and the hardest-negative rule of Appendix E; the donor-reuse convention and the fact that the sample-level mapping is built once and fixed for all epochs are stated there. The comparison strategies differ from CCVS only in the cluster mapping and the donor rule, with coverage, architectures, and training budget held fixed.

Defense. RGAR's trust boundary, calibration, suspicion score, party-evidence accumulation, attribution and revocation rules, and repair are given in Appendix G, with the implemented form collected in Algorithm 2. Two properties that affect reproduction are stated explicitly rather than left implicit: RGAR uses class-prototype replacement instead of the reconstructor for the asymmetric UCI-HAR and UCI-Bank models, and repair is disabled at inference, so all reported test accuracies are computed on unmodified embeddings.

Sources of variation. Reported standard deviations are over three VFL training seeds with the Phase I partition and clustering artifacts held fixed, so they measure training variability conditional on a single clustering run rather than variability of the full attack pipeline. Entries marked † are single-seed. Phase I is computed once per dataset and reused across all attack and defense conditions, which removes clustering variation as a confound between conditions but also means clustering variance is not reported. Hyperparameters were set from training-time diagnostics rather than a shared grid, as described in Appendix G; this is a limitation of the current protocol.

Compute. Hardware, software versions, Phase I clustering runtimes, and per-run wall-clock times are reported in Appendix I. All experiments run on a single GPU or CPU without distributed training, and the most expensive component, CIFAR-10 Phase I clustering, takes roughly seven hours on a single consumer GPU. The full study is therefore reproducible on commodity hardware.

Code release. Upon acceptance, we will publicly release the complete code repository with the exact commands needed to reproduce every reported result. The release will cover Phase I clustering, CCVS and the comparison strategies, the defense baselines, and RGAR, together with dataset preparation scripts, per-table configuration files, the training seeds used, and the Phase I artifacts.

References

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *European Symposium on Artificial Neural Networks*, 2013.

Yijie Bai, Yanjiao Chen, Hanlei Zhang, Wenyuan Xu, Haiqin Weng, and Dou Goodman. VILLAIN: Backdoor attacks against vertical split learning. In *32nd USENIX Security Symposium*, pp. 2743–2760. USENIX Association, 2023.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, 2017.

Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. FLTrust: Byzantine-robust federated learning via trust bootstrapping. In *Network and Distributed System Security Symposium*, 2021.

Peng Chen, Jirui Yang, Junxiong Lin, Zhihui Lu, Qiang Duan, and Hongfeng Chai. A practical clean-label backdoor attack with limited information in vertical federated learning. In *2023 IEEE International Conference on Data Mining*, pp. 41–50. IEEE, 2023. doi: 10.1109/ICDM58522.2023.00013.

Peng Chen, Xin Du, Zhihui Lu, and Hongfeng Chai. Universal adversarial backdoor attacks to fool vertical federated learning. *Computers & Security*, 137:103601, 2024a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *PMLR*, pp. 1597–1607, 2020.

Xiaolin Chen, Daoguang Zan, Wei Li, Bei Guan, and Yongji Wang. A GAN-based data poisoning framework against anomaly detection in vertical federated learning. In *2024 IEEE International Conference on Communications*, pp. 3982–3987. IEEE, 2024b. doi: 10.1109/ICC51166.2024.10623031.

Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. Secureboost: A lossless federated learning framework, 2021. URL `https://arxiv.org/abs/1901.08755`.

Yungi Cho, Woorim Han, Miseon Yu, Younghan Lee, Ho Bae, and Yunheung Paek. VFLIP: A backdoor defense for vertical federated learning via identification and purification. In *Computer Security – ESORICS 2024*, volume 14985 of *Lecture Notes in Computer Science*, pp. 291–312. Springer, 2024. doi: 10.1007/978-3-031-70903-6_15.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, pp. 1605–1622, 2020.

Chong Fu, Xuhong Zhang, Shouling Ji, Jinyin Chen, Jingzheng Wu, Shanqing Guo, Jun Zhou, Alex X Liu, and Ting Wang. Label inference attacks against vertical federated learning. In *31st USENIX security symposium (USENIX Security 22)*, pp. 1397–1414, 2022.

Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.

Yuhao Gu and Yuebin Bai. Lr-ba: Backdoor attack against vertical federated learning using local latent representations. *Computers & Security*, 129:103193, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Weiyang He and Chip-Hong Chang. HASSLE: A self-supervised learning enhanced hijacking attack on vertical federated learning, 2025.

Ying He, Zhili Shen, Jingyu Hua, Qixuan Dong, Jiacheng Niu, Wei Tong, Xu Huang, Chen Li, and Sheng Zhong. Backdoor attack against split neural network-based vertical federated learning. *IEEE Transactions on Information Forensics and Security*, 19:748–763, 2024a. doi: 10.1109/TIFS.2023.3327853.

Ying He, Zhili Shen, Jingyu Hua, Qixuan Dong, Jiacheng Niu, Wei Tong, Xu Huang, Chen Li, and Sheng Zhong. Backdoor attack against split neural network-based vertical federated learning. *IEEE Transactions on Information Forensics and Security*, 19:748–763, 2024b. doi: 10.1109/TIFS.2023.3327853.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

Prannay Khosla, Pang Wei Tian, Chen Wang, Aaron Cartwright, Phillip Isola, Dilip Krishnan, Francois Barbalat, Ling Liu, Dhruv Krishnamurthy, Joshua Chang, Yuandong Tian, Phillip Isola, Dilip Krishnan, Carl Vondrick, Deva Ramanan, Cordelia Schmid, and Yonglong Tian. Supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 18661–18673, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

Jinrong Lai, Tong Wang, Chuan Chen, Yihao Li, and Zibin Zheng. VFedAD: A defense method based on the information mechanism behind the vertical federated data poisoning attack. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 1148–1157. ACM, 2023. doi: 10.1145/3583780.3615106.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. Vertical federated learning: Concepts, advances, and challenges. *IEEE transactions on knowledge and data engineering*, 36(7):3615–3634, 2024.

Linpeng Lu and Ning Ding. Multi-party private set intersection in vertical federated learning. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 707–714. IEEE, 2020. doi: 10.1109/TrustCom50675.2020.00098.

Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55, 1936.

Geoffrey McLachlan and David Peel. *Finite Mixture Models*. Wiley, 2000.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2023. URL `https://arxiv.org/abs/1602.05629`.

Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

Mohammad Naseri, Yufei Han, and Emiliano De Cristofaro. BadVFL: Backdoor attacks in vertical federated learning. In *2024 IEEE Symposium on Security and Privacy*, pp. 2013–2028. IEEE, 2024. doi: 10.1109/SP54263.2024.00008.

Kento Oonishi and Tsunato Nakai. CC-VFed: Client contribution detects byzantine attacks in vertical federated learning, 2025. URL `https://openreview.net/forum?id=E3qIInyTgL`.

Jeff Schlimmer. Mushroom dataset. UCI Machine Learning Repository, 1987.

Wei Shen, Wenke Huang, Guancheng Wan, and Mang Ye. Label-free backdoor attacks in vertical federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20389–20397, 2025. doi: 10.1609/aaai.v39i19.34246.

Junyu Shi, Wei Wan, Shengshan Hu, Jianrong Lu, and Leo Yu Zhang. Challenges and approaches for mitigating byzantine attacks in federated learning. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 139–146. IEEE, 2022.

Jinhyun So, Başak Güler, and A Salman Avestimehr. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communications*, 39(7):2168–2181, 2020.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. FixMatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 596–608, 2020.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, 2017.

Jarin Tasneem. Breaking cross-view associations: Byzantine model poisoning attack against vertical federated learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(48):41522–41524, 2026. doi: 10.1609/aaai.v40i48.42327. URL https://ojs.aaai.org/index.php/AAAI/article/view/42327.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data, 2018.

Yuxin Xi, Yu Guo, Shiyuan Xu, Chengjun Cai, and Xiaohua Jia. Private sample alignment for vertical federated learning: An efficient and reliable realization. *IEEE Transactions on Information Forensics and Security*, 20:3834–3848, 2025. doi: 10.1109/TIFS.2025.3555794.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jiuyun Xu, Yinyue Jiang, Hanfei Fan, and Qiqi Wang. Svfldetector: a decentralized client detection method for byzantine problem in vertical federated learning. *Computing*, 106(5):1659–1679, 2024.

Yuexin Xuan, Xiaojun Chen, Zhendong Zhao, Bisheng Tang, and Ye Dong. Practical and general backdoor attacks against vertical federated learning. In *Machine Learning and Knowledge Discovery in Databases: Research Track*, pp. 402–417. Springer, 2023. doi: 10.1007/978-3-031-43415-0_24.

Liu Yang, Di Chai, Junxue Zhang, Yilun Jin, Leye Wang, Hao Liu, Han Tian, Qian Xu, and Kai Chen. A survey on vertical federated learning: From a layered perspective, 2023. URL https://arxiv.org/abs/2304.01829.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, 2019. doi: 10.1145/3298981.

Duanyi Yao, Songze Li, Ye Xue, and Jin Liu. Constructing adversarial examples for vertical federated learning: Optimal client corruption through multi-armed bandit. In *International Conference on Learning Representations*, 2024.

Mang Ye, Wei Shen, Bo Du, Eduard Snezhko, Vassili Kovalev, and Pong C. Yuen. Vertical federated learning for effectiveness, security, applicability: A survey. *ACM Computing Surveys*, 57(9):1–32, 2025. doi: 10.1145/3720539.

Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5650–5659. PMLR, 2018.

Lei Yu, Meng Han, Yiming Li, Changting Lin, Yao Zhang, Mingyang Zhang, Yan Liu, Haiqin Weng, Yuseok Jeon, Ka-Ho Chow, and Stacy Patterson. A survey of privacy threats and defense in vertical federated learning: From model life cycle perspective, 2024. URL https://arxiv.org/abs/2402.03688.

Kun Yuan, Zhaoxian Wu, and Qing Ling. A byzantine-resilient dual subgradient method for vertical federated learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4273–4277. IEEE, 2022.

A AI USE STATEMENT

In this work, we used generative AI tools to identify literature relevant to our methodological assumptions, provide feedback on those assumptions, and assist with the interpretation and reporting of experimental results. Additionally, we used these tools for literature discovery, reference formatting, LaTeX formatting, and suggestions on manuscript organization and wording. These suggestions served as inputs to the authors' own assessment and revision.

All AI suggestions served as inputs to the authors' own judgment. Research questions, attack and defense design, implementation, experiment execution, and the choice of what to report were decided

by the authors. We have not used generative AI tools for generating synthetic datasets, and proving mathematical claims is not applicable to this work, as it contains no formal proofs. We reviewed all AI-assisted material, verified each reference against its original publication, and checked that the cited sources support the associated claims. We assessed methodological suggestions against the implemented procedures and checked experimental descriptions and numerical claims against the codebase and experiment logs. We independently composed the final text and take responsibility for the final content of this work, including all citations, claims, and artifacts produced with the aid of generative AI.

B Ablation Results

Table 5: RGAR clean-test accuracy (%) vs. reference fraction r_{ref}, and with 10% of $\mathcal{R}$ poisoned. Single seed. Undefended baselines: 33.3% (MNIST), 15.5% (HAR).

Dataset	r**=0.01**	r**=0.02**	r**=0.05**	r**=0.10**	**10% corrupt**
MNIST	78.9	80.6	83.9	82.7	82.2
UCI-HAR	68.9	66.3	78.3	83.2	90.2

C Data Partitioning and Auxiliary Supervision

C.1 Vertical feature partitions

All parties operate on aligned sample identifiers. In the two-party setting, Party A holds the adversarial view, Party B holds the complementary honest view, and the server holds the training labels. Partitions are fixed per dataset and shared by all clean, attacked, and defended runs, except where stated below for UCI-Bank.

Image datasets. Images are split along the pixel-width axis. For MNIST and Fashion-MNIST, Party A receives the left 14 grayscale columns and Party B the right 14. For CIFAR-10, Party A receives the left 16 RGB columns and Party B the right 16; RGB images are rescaled to $[0, 1]$ before contrastive augmentation. These splits are controlled spatial views. They do not represent parties holding semantically different modalities, which the tabular datasets approximate more closely.

Tabular datasets. Features are ranked by the estimated mutual information $\widehat{\mathrm{MI}}(x_d; y)$, computed only on the adversary's auxiliary labeled subset $\mathcal{D}_L$. The same ranking defines the Phase I input and the Phase II swap view.

- **UCI-HAR.** Party A receives the top $\eta_A = 0.88$ fraction of ranked features and Party B the remainder.
- **UCI-Mushroom.** Categorical attributes are one-hot encoded; Party A receives the top-MI block of the resulting binary features.
- **UCI-Bank.** Party A receives a skewed MI-ranked share ($\eta_A = 0.92$), concentrating high-information features on the adversarial side to provide diverse donor regions in Phase II.

Feature share η_A and embedding dimension (Appendix F) are distinct quantities. For UCI-Bank, the clean baseline reported in the main text uses the same skewed partition as the attacked runs.

C.2 Auxiliary labels and cluster counts

Let $\varepsilon = |\mathcal{D}_L|/N$. The auxiliary set is stratified with at least one sample per class. Auxiliary labels are used for representation refinement, pseudo-label initialization, cluster merging, and tabular feature ranking. The attacker never receives the remaining training labels. Table 6 lists the per-dataset configuration used for the main-text results.

The cluster count K_c equals the number of classes, which is known from the task definition, for all datasets except UCI-Bank. Bank is binary; two clusters give too few geometrically distinct donor regions, so Bank uses $K_c = 16$. Bank's clusters are therefore donor regions, not predicted classes.

Table 6: Per-dataset configuration for main-text results.

Dataset	N (train)	K_c	ε	Phase I pipeline
MNIST	60,000	10	2%	SimCLR → SupCon → GMM merge
Fashion-MNIST	60,000	10	2%	SimCLR → SupCon → GMM merge
CIFAR-10	50,000	10	3%	SimCLR → FixMatch → GMM/teacher selection
UCI-HAR	8,754	6	3%	Tabular FixMatch (EMA teacher)
UCI-Mushroom	6,905	2	2%	Bernoulli mixture + pseudo-label refinement
UCI-Bank	36,168	16	2%	Unsupervised GMM (donor regions)

D Phase I: Representation Learning and Clustering

D.1 Objectives

Contrastive pretraining. For image datasets, an encoder $e(\cdot)$ is pretrained on Party A's local view with the SimCLR NT-Xent objective (Chen et al., 2020). For two stochastic augmentations t_1, t_2, let $z_i^{(1)} = e(t_1(x_i))$ and $z_i^{(2)} = e(t_2(x_i))$:

$$\mathcal{L}_{\text{SimCLR}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp\big(\text{sim}(z_i^{(1)}, z_i^{(2)})/\tau\big)}{\sum_{k\neq i} \exp\big(\text{sim}(z_i^{(1)}, z_k^{(2)})/\tau\big)}, \tag{13}$$

where B is the batch size, τ the temperature, and $\text{sim}(\cdot,\cdot)$ cosine similarity.

Supervised contrastive refinement. For MNIST and Fashion-MNIST, the encoder is refined on $\mathcal{D}_L$ with supervised contrastive learning (Khosla et al., 2020). With $z_i = e(u_i)$ and $P(i)$ the labeled positives sharing sample i's class:

$$\mathcal{L}_{\text{SupCon}} = -\sum_{i\in\mathcal{D}_L} \frac{1}{|P(i)|} \sum_{p\in P(i)} \log \frac{\exp\big(\text{sim}(z_i, z_p)/\tau\big)}{\sum_{a\neq i} \exp\big(\text{sim}(z_i, z_a)/\tau\big)}. \tag{14}$$

FixMatch-style self-training. For CIFAR-10 and UCI-HAR, a student p_θ is trained on $\mathcal{D}_L$ and on confident pseudo-labels from an EMA teacher $\theta_t \leftarrow m\,\theta_t + (1-m)\,\theta_s$ (Sohn et al., 2020):

$$\mathcal{L}_{\text{fix}} = \frac{1}{\mu B}\sum_{u} \mathbb{1}\Big[\max_y p_{\theta_t}(y \mid a_w(u)) \geq \tau_c\Big]\; H\big(\hat{y}_u,\, p_\theta(y \mid a_s(u))\big), \tag{15}$$

where a_w, a_s are weak and strong augmentations, $\hat{y}_u$ the teacher pseudo-label, μ the unlabeled batch multiplier, and τ_c the confidence threshold.

D.2 Per-dataset pipelines

Grayscale images (MNIST, Fashion-MNIST). SimCLR pretraining, SupCon refinement on $\mathcal{D}_L$, and one self-training pass that produces pseudo-labels. An over-specified GMM is then fit to L_2-normalized embeddings, and components are merged into K_c clusters by assigning each component to its nearest auxiliary-label class prototype.

RGB images (CIFAR-10). A 16-column RGB half-view carries limited semantic context, so SupCon is replaced by FixMatch. A linear probe trained on $\mathcal{D}_L$ initializes the classifier. A final selection step chooses between teacher predictions and GMM-based assignments by their accuracy on $\mathcal{D}_L$.

Continuous tabular (UCI-HAR). A student–teacher MLP trained with the FixMatch objective, using inverse-frequency class weights estimated from $\mathcal{D}_L$. Cluster identities are the teacher's hard predictions.

Discrete tabular (UCI-Mushroom). A Bernoulli mixture model on binary features, initialized with deterministic-annealing EM, followed by pseudo-label refinement rounds, graph-based smoothing over nearest neighbours, and a spread-based post-refinement step.

Attack-oriented clustering (UCI-Bank). An unsupervised GMM with $K_c = 16$ components on Party A's selected features. The objective is geometrically diverse donor regions for Phase II, not binary label alignment.

D.3 Cluster assignment and confidence

For mixture-based pipelines with responsibilities r_{ik}, and for classifier-based pipelines with teacher probabilities $p_{\theta_t}(k \mid x_i^{(A)})$, the cluster identity and confidence are

$$\hat{c}_i = \arg\max_k r_{ik}, \qquad \hat{q}_i = \max_k r_{ik}, \tag{16}$$

with r_{ik} replaced by $p_{\theta_t}(k \mid x_i^{(A)})$ for classifier-based pipelines. Assignments are computed after any component merging.

E Phase II: Swap Construction

E.1 Cluster mapping and donor pool

Let v_i be the L_2-normalized Phase I representation of sample i and μ_s the normalized centroid of cluster C_s. Cluster distances are $D_{st} = 1 - \langle \mu_s, \mu_t \rangle$. The maximum-distance derangement $\pi^\star$ (Eq. 7 of the main text) is solved exactly as a linear assignment problem on D with the diagonal excluded.

For each source cluster s, donors are drawn from the high-confidence core

$$\mathcal{D}_t^{(q)} = \{\, j \in C_t : \hat{q}_j \geq \mathrm{quant}(\hat{q}_{C_t}, q) \,\} \tag{17}$$

of the destination cluster $t = \pi^\star(s)$. Within this pool, each victim $i \in C_s$ receives the hardest negative $j^\star(i) = \arg\min_j \langle v_i, v_j \rangle$ among unused donors.

E.2 Donor reuse and consistency

Selected donors are marked as used until the destination core is exhausted, after which the core is refilled (greedy diversity cycling). The sample-level mapping is therefore not bijective when $|C_s| > |\mathcal{D}_t^{(q)}|$. Consistency refers to the cluster-level rule $s \mapsto \pi^\star(s)$, which is fixed for all epochs. The poisoned view $\widetilde{X}^{(A)}$ is constructed once before training and reused in every epoch; Party B's features and the server labels keep their original alignment. Evaluation always uses correctly aligned clean test views.

E.3 Comparison strategies

All strategies rearrange only Party A's training features and use the same coverage, architecture, and training budget.

- **CCVS**: maximum-distance derangement with hardest-negative donors from the destination core.
- **Round-Robin**: fixed cyclic mapping $s \mapsto (s+1) \bmod K_c$, random donors within the destination cluster.
- **Random Cluster**: uniformly random derangement, random donors within the destination cluster.
- **Random Sample**: each victim receives an independent random donor from the full training set, without Phase I structure.

Class-aware variant (UCI-Bank). Because Bank's 16 clusters are not label-aligned, the Bank run uses an auxiliary classifier trained on $\mathcal{D}_L$ to estimate each cluster's class tendency, and restricts destinations to clusters with the opposite predicted tendency. The rearrangement constraint is unchanged: donors are still Party A's own genuine feature rows. This uses predicted class information from $\mathcal{D}_L$ only, not the server's training labels.

E.4 Swap coverage

Coverage is the fraction of each cluster's victims that are reassigned. Table 7 shows a threshold response: at partial coverage the correctly aligned majority of each cluster dominates training, and collapse occurs only when the fabricated association is the only one available for a cluster. Coverage is the attacker's choice, since Party A controls every row of its own view.

Table 7: CCVS clean-test accuracy (%) vs. swap coverage. Lower is stronger. The 100% column is mean $\pm$ std over three seeds and matches Table 2; other columns are single-seed.

Dataset	25%	50%	75%	100%
MNIST	96.0	95.4	93.5	32.5 ± 2.2
Fashion-MNIST	86.8	86.5	85.7	26.2 ± 1.9
UCI-HAR	95.8	93.3	93.0	15.7 ± 2.1

E.5 Donor-scope sensitivity

Table 8 varies the donor scope k (number of farthest destination clusters pooled) on MNIST. This sweep was run on an earlier configuration of the donor assignment (clean accuracy 96.9%), so absolute values differ from Table 2; the trend is the result of interest. Pooling more distant clusters weakens the attack.

Table 8: Attack accuracy (%) vs. donor scope k on MNIST (earlier configuration, single seed).

k	1	2	3	5	7
Attack acc.	33.0	34.5	36.3	50.9	61.4
Drop (pp)	63.9	62.4	60.6	46.0	35.5

F Model Architectures and Training

MNIST and Fashion-MNIST. Each party uses a lightweight local convolutional encoder over its half-image. The server concatenates party embeddings and applies a two-layer MLP head with hidden dimension 32.

CIFAR-10. Each party uses a ResNet-based bottom model (He et al., 2016) over its half-image, with a concatenation fusion head. RandAugment is used in Phase I representation learning and in VFL training.

Tabular datasets. MLP bottom models and MLP head. UCI-HAR and UCI-Bank use asymmetric embeddings: 96 (adversarial) and 8 (honest) for HAR, 96 and 2 for Bank.

Optimization. All VFL models train for 80 epochs with Adam or AdamW. Optimizer settings, batch sizes, and augmentations are fixed per dataset across clean, attacked, and defended runs. Phase I artifacts (cluster assignments and confidences) are computed once per dataset and reused. Reported standard deviations are over three VFL training seeds with the Phase I partition fixed, so they measure training variability conditional on that partition.

G RGAR: CALIBRATION, ATTRIBUTION, AND REPAIR

G.1 REFERENCE ACQUISITION AND TRUST BOUNDARY

RGAR assumes the server holds true Party-A features for a small reference subset $\mathcal{R}$, collected at onboarding before compromise or under third-party audit. Authentication protects integrity in transit; it does not by itself make the features true, which is why the onboarding or audit assumption is required. The server stores an immutable copy and uses its own labels to form calibration triples. Party A never receives labels.

Following Eq. 2, the adversary manipulates its inputs but executes f_A faithfully on server-supplied inputs. Forged responses to reference queries, colluding parties, and poisoning of $\mathcal{R}$ at collection time are outside the evaluated setting. Sharing raw reference features is a privacy concession beyond standard VFL and is not a cryptographic guarantee.

Embedding-only variant. Where raw features cannot be shared, RGAR can be calibrated on frozen clean reference embeddings instead. This preserves the calibration role of $\mathcal{R}$ while reducing raw-feature exposure, at the cost of not tracking encoder updates. A privacy-preserving implementation using secure enclaves or cryptographic computation is left to future work.

G.2 SUSPICION SCORE

The main text writes the pair score as a convex blend (Eq. 10). The implementation uses the unnormalized form

$$s_i^{\text{proto}} = w_{\text{proto}} \cdot \frac{\bar{d}_i^A + \bar{d}_i^B}{2} + w_{\text{jnt}} \cdot s_i^{\text{jnt}}, \qquad w_{\text{proto}} = 1, \tag{18}$$

where $\bar{d}_i^k$ is the diagonal Mahalanobis distance of h_i^k to $p_{y_i}^k$, scaled by the median distance on $\mathcal{R}$, and s_i^{jnt} is the cosine distance to the joint prototype (Eq. 11). The two forms are equivalent up to a positive rescaling: Eq. 10 with weight α equals α times Eq. (18) with $w_{\text{jnt}} = (1-\alpha)/\alpha$. The reported range $w_{\text{jnt}} \in [0.06, 0.55]$ corresponds to $\alpha \in [0.65, 0.94]$.

G.3 PARTY EVIDENCE AND ATTRIBUTION

For party $k \in \{A, B\}$, the per-sample evidence combines prototype distance and temporal drift against an exponential moving average of that sample's embedding (Tarvainen & Valpola, 2017):

$$e_i^k = \bar{d}_i^k + \big(1 - \cos(h_i^k, \text{EMA}_i^k)\big). \tag{19}$$

Samples with $s_i^{\text{proto}} > \tau_{\text{pair}}$ contribute the signed difference $e_i^A - e_i^B$ to an epoch buffer E. After a burn-in of W epochs, if $\text{mean}(E) > \tau_{\text{global}}$, corruption is attributed to Party A and its trust weight decays multiplicatively, $\rho_A \leftarrow \max(\rho_{\min}, \gamma\rho_A)$. If $\text{mean}(E) < -\tau_{\text{global}}$, attribution is revoked and trust recovers by a fixed factor, $\rho_A \leftarrow \min(1, 1.05\,\rho_A)$. An empty buffer leaves ρ_A unchanged. The buffer is cleared every epoch. Party B keeps $\rho_B = 1$.

G.4 REPAIR

After attribution, the live embedding is repaired as in Eq. 12:

$$\hat{h}_i^A = \beta\, g_\theta(h_i^B, y_i) + (1-\beta)\, p_{y_i}^A, \qquad \tilde{h}_i^A = (1-w_i)\, h_i^A + w_i\, \hat{h}_i^A, \tag{20}$$

and the head receives $(\rho_A \tilde{h}_i^A,\ \rho_B h_i^B)$. The blend weight w_i increases with s_i^{proto} and is raised after attribution.

Asymmetric architectures. For UCI-HAR and UCI-Bank, where the honest embedding has only 8 and 2 dimensions, the reconstructor is not used: RGAR sets $\hat{h}_i^A = p_{y_i}^A$ and freezes the suspected encoder after attribution. β is therefore not applicable for these datasets. Their recovery in Table 3 comes from class-prototype replacement.

Reconstructor. g_θ takes the concatenation of h_i^B and a one-hot encoding of y_i, and is trained on $\mathcal{R}$ with the SmoothL1 objective (Eq. 9), then frozen.

G.5 Algorithm

Algorithm 2 RGAR

Require: Reference set $\mathcal{R}$; thresholds $\tau_{\text{pair}}, \tau_{\text{global}}$; burn-in W; weight w_{jnt}; blend β; decay γ; floor $\rho_{\min}$

Fit $\{p_c^A, p_c^B, p_c^{\text{jnt}}, \Sigma_c^A, \Sigma_c^B\}$ on $(h_i^A, h_i^B, y_i)_{i\in\mathcal{R}}$
Train g_θ on $\mathcal{R}$ with $\mathcal{L}_{\text{recon}}$, then freeze {skipped for asymmetric models}
$\rho_A \leftarrow 1$; $\rho_B \leftarrow 1$; attr $\leftarrow$ FALSE; $E \leftarrow \emptyset$
for $t = 1$ **to** T_{epochs} **do**
  for all mini-batches $\mathcal{B}$ **do**
    for all $i \in \mathcal{B}$ **do**
      Compute s_i^{proto} (Eq. 18) and e_i^A, e_i^B; update EMAs
      if $s_i^{\text{proto}} > \tau_{\text{pair}}$ **then**
        $E \leftarrow E \cup \{e_i^A - e_i^B\}$
      end if
      if asymmetric model **then**
        $\hat{h}_i^A \leftarrow p_{y_i}^A$
      else
        $\hat{h}_i^A \leftarrow \beta\, g_\theta(h_i^B, y_i) + (1-\beta)\, p_{y_i}^A$
      end if
      $w_i \leftarrow w(s_i^{\text{proto}}, \text{attr})$ {increasing in s_i^{proto}, raised once attr}
      $\tilde{h}_i^A \leftarrow (1-w_i)\, h_i^A + w_i\, \hat{h}_i^A$
    end for
    Forward $(\rho_A \tilde{h}_i^A,\ \rho_B h_i^B)_{i\in\mathcal{B}}$ to the head; update models
  end for
  if $t \geq W$ **and** $E \neq \emptyset$ **then**
    if $\text{mean}(E) > \tau_{\text{global}}$ **then**
      attr $\leftarrow$ TRUE; $\rho_A \leftarrow \max(\rho_{\min}, \gamma\rho_A)$ {freeze f_A for asymmetric models}
    else if $\text{mean}(E) < -\tau_{\text{global}}$ **then**
      attr $\leftarrow$ FALSE; $\rho_A \leftarrow \min(1, 1.05\,\rho_A)$
    end if
  end if
  $E \leftarrow \emptyset$
end for
return trained head and bottom models {repair disabled at inference}

G.6 Training labels and inference

The repair uses y_i, which the server legitimately holds during supervised training. At inference, labels are unavailable and repair is disabled: the trained head receives the unmodified embeddings (h_i^A, h_i^B) from both parties. All reported test accuracies use this protocol.

G.7 Scope and adaptive adversaries

The evaluated defense covers two parties and one malicious input view. For $K > 2$, one reconstructor per party can be trained from the remaining views, with attribution to the party with the largest accumulated evidence; its reliability depends on the remaining views being honest and predictive. An adaptive adversary aware of RGAR could spare reference samples, alternate clean and poisoned submissions, or drift slowly to stay below τ_{global}. Joint-prototype deviation, temporal evidence, and revocable attribution are design responses to these strategies, not a demonstrated robustness guarantee; an adaptive evaluation is future work.

H Evaluation Metrics

Let n_{sc} count training samples in cluster s with class c.

$$\text{Purity} = \frac{1}{N}\sum_{s}\max_{c} n_{sc}, \qquad \text{H-ACC} = \frac{1}{N}\max_{m\in\mathcal{M}}\sum_{s} n_{s,m(s)}, \tag{21}$$

where $\mathcal{M}$ is the set of one-to-one maps from clusters to classes (Hungarian algorithm (Kuhn, 1955)). On the same contingency table, H-ACC $\leq$ Purity, with equality when the majority mapping is a bijection.

Attack and defense metrics. Attack strength is clean-test accuracy after poisoned training (lower is stronger). Defense performance is clean-test accuracy after defended training. For the baselines, Det. is the percentage of poisoned training samples flagged in the final epoch; for RGAR, Susp. is the percentage of poisoned training samples with $s_i^{\text{proto}} > \tau_{\text{pair}}$ in the final epoch.

Imbalanced data. UCI-Bank's majority class is about 88% of the data, so accuracy near that value can reflect majority-class prediction. Bank results should be read against the majority-class baseline on the test split.

I Computational Resources

Experiments used two machines, each with a single GPU and no distributed training. Machine A: 16-core CPU, 14 GB RAM, NVIDIA RTX 4060 Laptop GPU (8 GB), Ubuntu 22.04.5, PyTorch 2.6.0 with CUDA 12.4. Machine B: 24-core CPU, 31 GB RAM, NVIDIA RTX 4090 (24 GB). CIFAR-10 ran on Machine B.

Table 9: Phase I clustering runtime (run once per dataset, reused by all Phase II runs).

Dataset	Pipeline	Runtime (min)
MNIST	SimCLR + SupCon + GMM	≈122
Fashion-MNIST	SimCLR + SupCon + GMM	≈135
CIFAR-10	SimCLR + FixMatch	≈436
UCI-HAR	Tabular FixMatch	≈70
UCI-Mushroom	Bernoulli mixture	≈60
UCI-Bank	GMM ($K_c = 16$)	≈100

Table 10: Wall-clock seconds per 80-epoch VFL run (GPU). Naked: undefended poisoned training. RGAR includes reconstructor training, which dominates its cost and does not scale with epoch count.

Dataset	Clean	Attack	Naked	RGAR
MNIST	99	206	47	3165
Fashion-MNIST	81	115	47	2964

RGAR overhead. RGAR is substantially more expensive than undefended training: on MNIST, about 3,165 s versus 47 s, almost entirely from reconstructor training. Tabular datasets train on CPU; clean and attacked runs finish in under a minute, and RGAR takes about 2–15 minutes depending on reconstructor size. Reference queries add communication proportional to $|\mathcal{R}| \times d_A$ per query. RGAR runs during training and does not require a second training pass, but its cost should be compared with retraining without the attributed party, which costs one undefended run.