RELIABILITY-AWARE GENERATOR CONSOLIDATION WITH FEATURE-GATED ADVERSARIAL REPLAY FOR FEDERATED CLASS-INCREMENTAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated class-incremental learning must retain earlier knowledge while combining clients with different class distributions and task histories. We study synchronized stage transitions with client-specific task orders and spatial label skew. FedRUA organizes replay into two server stages. First, a conditional generator learns from cached client classifiers whose real-data histories include the requested class. Predictive disagreement attenuates multi-expert supervision, while a singleton fallback combines predictive confidence, feature-statistic consistency, and update deviation. Second, the server fine-tunes the global classifier using weighted clean replay and feature-gated adversarial replay. The gate controls the additional adversarial loss while retaining clean replay. Clients use the frozen shared generator for local replay without retaining historical raw examples. On five-task CIFAR-100, FedRUA improves mean final accuracy by 0.88 percentage points over DGR-FCL and reduces mean forgetting by 1.70 points. Under Dirichlet label skew with $\alpha = 0.5$, the accuracy margin increases to 1.45 points. On 64×64 ImageNet-1K, the corresponding accuracy and forgetting margins are 0.30 and 0.70 points. These results support the integrated procedure for a trusted server under the evaluated task-order and spatial heterogeneity.

1 INTRODUCTION

Federated learning (FL) trains a shared model from decentralized data without transferring raw client examples (McMahan et al., 2017; Li et al., 2020). In many deployments, however, each institution receives a sequence of new classes or domains and cannot retain all earlier data. Federated continual learning (FCL) must therefore handle catastrophic forgetting together with client drift (Yoon et al., 2021; Dong et al., 2022). We study federated class-incremental learning with two sources of heterogeneity: clients may receive different class proportions within a task and encounter the same task groups in different orders. Clients nevertheless advance under a shared stage clock. Thus, task-order heterogeneity concerns which task a client encounters at a given stage, whereas task-stage heterogeneity concerns whether clients occupy different stage indices at the same round. Our experiments evaluate task-order and spatial heterogeneity under synchronized stage progression.

Generative replay is attractive when old raw examples cannot be retained: a generator supplies pseudo-examples while clients learn new data. Heterogeneity makes both the production and the use of these examples consequential. A client that has never observed the requested class has no real-data basis for supervising its synthesis. Prior exposure is necessary for eligibility, but does not ensure that the client's current classifier still predicts that class well. Even a confidently classified synthetic example constrains the global classifier only at the sampled input unless replay also trains its nearby behavior. Selecting supervision for synthesis and regularizing classifier updates are thus separate, connected decisions: errors in the first stage can propagate through the second.

We introduce FedRUA, which assigns distinct roles to class coverage, predictive agreement, and feature statistics. Real-data coverage determines which cached client experts can supervise a requested label. Disagreement controls the strength of their collective predictive guidance, with a separate fallback when only one expert is available. Feature statistics regularize synthesis and determine where classifier replay receives an additional adversarial loss. The server first consolidates a conditional generator and then trains the global classifier on clean and gated adversarial replay; clients use the

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.

shared generator only for inference. The contribution is this organization of distributed supervision and replay regularization in a trusted-server setting. Specifically:

- We formulate class-conditioned supervision from cached client experts, separating real-data eligibility from the strength of predictive guidance.
- We combine generator consolidation with weighted clean replay and feature-gated adversarial classifier training, while keeping generator optimization on the server.
- We evaluate the complete procedure using final accuracy and forgetting, major-loss and branch comparisons, backbone and initialization studies, replay sensitivity, participation regimes, and a task-boundary server profile.

2 Related Work

Federated continual learning. FCL methods inherit continual-learning choices and additionally coordinate knowledge across clients: regularization constrains damaging updates (Bakman et al., 2024; Li et al., 2025b; Dong et al., 2022); replay retains or reconstructs earlier information (Li et al., 2025a; Qi et al., 2023); parameter isolation separates shared, local, or task-adaptive components (Yoon et al., 2021; Guo et al., 2024; Zhang et al., 2025); and prototype methods communicate compact class structure (Gao et al., 2024; Yu et al., 2025). These trade memory, plasticity, personalization, and server coordination differently; FedRUA instead retains a single adaptive global classifier and moves synthetic consolidation to the server rather than training generators on clients.

Generative replay in FCL. These methods differ in whose information guides synthesis and which knowledge is selected for retention. TARGET trains a generator to approximate accumulated global knowledge for exemplar-free distillation (Zhang et al., 2023); FedCIL combines model consolidation and consistency enforcement (Qi et al., 2023); and DGR-FCL uses diffusion-based generative replay (He et al., 2026). CAN already uses client expertise to navigate synthesis and client-specific forgetting to direct replay (Rong et al., 2025). AF-FCL already uses a feature-space normalizing flow to assess and selectively forget biased prior knowledge (Wuerkaixi et al., 2024). FedRUA contributes a specific coupling of these established directions. Real-data class coverage selects the synthesis teachers; disagreement attenuates their collective predictive loss; and a feature-statistic gate controls an additional adversarial loss during the subsequent classifier stage. Clean replay remains active when the gate is closed. This separation makes teacher eligibility, supervision strength, and classifier regularization explicit within one server-trained replay procedure. The empirical comparisons assess the resulting system and its major objective terms.

Data-free synthesis and distillation. DAFL distills a student from a fixed teacher's responses (Chen et al., 2019); DeepInversion matches feature moments in the teacher's batch-normalization layers (Yin et al., 2020). FedRUA uses BN-moment matching as a feature-statistic proxy, not evidence of the true data manifold, alongside class guidance, cross-client distillation, and reliability weighting. The BN-only row in Section 5.4 evaluates these additions jointly; it does not isolate every selection rule.

Spatial versus task-stage heterogeneity. Non-IID FL typically describes a spatial allocation at one training time (Li et al., 2020; Karimireddy et al., 2020); HFCL can also permit client task clocks to differ (Wuerkaixi et al., 2024; Yu et al., 2025). We reserve *task-stage heterogeneity* for differing positions in a task stream and do not equate it with delayed or asynchronous communication. In our evaluation, a shared stage clock coexists with client-specific task orders and IID or Dirichlet within-task allocations. These settings characterize task-order and spatial heterogeneity; different client stage clocks are outside the reported comparison.

Adversarial replay. FGSM constructs a local input perturbation from the classifier's loss gradient (Goodfellow et al., 2015). FedRUA applies this established operation during server-side replay, with proxy reliability weights and a feature-statistic gate on clean class sub-batches. The gate controls where the additional replay loss is applied. It does not establish the semantic validity of each perturbed input or preservation of an earlier decision boundary.

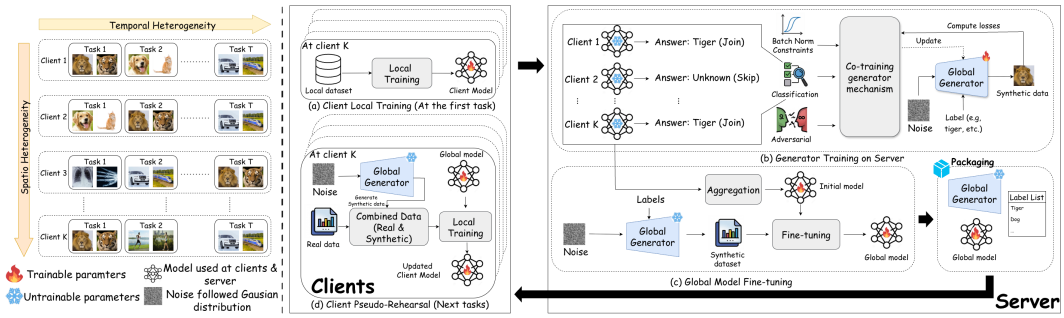


Figure 1: FEDRUA maintains one server generator; clients use it for local replay and upload classifiers/feature statistics, never optimizing generators. At each task boundary the server (1) consolidates the generator from class-eligible experts and (2) fine-tunes the classifier with reliability-aware, feature-gated adversarial replay.

3 PROBLEM SETTING

Consider K clients coordinated by a trusted server. Let $\pi_k(r)$ be client k's local stage at communication round r, and let $\rho_k(s)$ identify the task assigned to its stage s. The data $\mathcal{D}_k^s$ and class set $\mathcal{C}_k^s$ correspond to task $\rho_k(s)$. In the reported experiments, $\pi_k(r) = \pi(r)$ for all clients, while the permutations ρ_k may differ. A task that is historical for one client can therefore remain current for another. Such global label availability does not restore a client's earlier examples or make the federation's current training distribution stationary: the participating clients, class proportions, and local objectives still change. The evaluated setting is continual learning under client-specific task orders and spatial allocation, with synchronized stage progression.

Let $\mathcal{C}_{\text{all}}$ be the dataset labels and $\mathcal{C}_{\text{seen}}(r)$ the union of labels introduced across clients through round r. The classifier has $|\mathcal{C}_{\text{all}}|$ outputs. Evaluation selects the predicted label from $\mathcal{C}_{\text{pred}}(r) = \mathcal{C}_{\text{all}}$ without masking labels not yet introduced. The expert softmax distributions, entropies, disagreement, and KL terms in Section 4.2 also use $\mathcal{C}_{\text{all}}$. Prediction support and replay-label sampling support are distinct.

Class imbalance changes the amount of evidence available to an eligible expert, whereas missing real-data coverage means that the expert has no direct experience of the requested label. Different task orders also change how long ago a client encountered a class and which subsequent updates affected it. Coverage therefore records exposure rather than current competence. A conceptual retention objective combines each client's current and historical task risks:

$$\min_{\theta} \sum_{k=1}^{K} \frac{n_k}{\sum_j n_j} \left[\mathcal{L}_k^{\pi_k(r)}(\theta) + \sum_{s<\pi_k(r)} \mathcal{L}_k^s(\theta) \right]. \tag{1}$$

Here $n_k = |\mathcal{D}_k^{\pi_k(r)}|$ is client k's current-task sample count, and $\mathcal{L}_k^s$ is its expected loss on the task encountered at local stage s. Equation 1 assigns unit weight to the stage-specific risks inside each bracket and weights clients by current-task size. It expresses the retention goal; it is not an empirical risk computed from stored historical examples. FedRUA instead optimizes the local and server replay objectives below using one conditional generator $G_\phi(z, y)$. Class-balanced replay provides exposure to previously observed labels, while synthetic ones never expand a client's real-data coverage mask.

4 METHOD

4.1 END-TO-END WORKFLOW AND LOCAL REPLAY

Figure 1 ummarizes the procedure. The server maintains a global classifier f_θ and a conditional generator G_ϕ. Participating clients receive the current classifier and cache the latest shared generator. Write $\hat{x}(z, y) = \mathcal{N}((G_\phi(z, y) + 1)/2)$ for a generated image in classifier coordinates, where $\mathcal{N}$ is channelwise normalization. Client k samples replay labels from its earlier real-data history, $\mathcal{C}_{k,\text{real}}^{<\pi_k(r)}$. Let q_k^{rep} be the replay-label sampling distribution supported on this set and p_z the generator's latent

prior. The local objective is

$$\mathcal{L}_k^{\text{local}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_k^{\pi_k(r)}}\ell(f_{\theta_k}(x), y) + \lambda_{\text{rep}}\mathbb{E}_{y\sim q_k^{\text{rep}},\, z\sim p_z}\ell(f_{\theta_k}(\hat{x}(z,y)), y). \tag{2}$$

The first expectation denotes the empirical average over current local data. The replay term is zero when the earlier real-data label set is empty. Equation 2 separates the real and replay averages, so λ_{rep} is a loss coefficient rather than a sample-count ratio.

Clients optimize their classifiers and use the shared generator only for inference. Classifiers are aggregated after each communication round. At a task boundary, generator consolidation precedes global-classifier replay. The generator objective treats the expert and global-classifier parameters and BN buffers as fixed quantities; input gradients through these classifiers update the generator. The subsequent classifier stage uses the frozen consolidated generator.

The server caches each client's latest uploaded classifier, BN buffers, real-data coverage mask, and update-deviation factor. An entry is refreshed on the client's next upload and has no age-based expiry. This retains access to absent clients' recorded class coverage, although the latest snapshot need not be the client's best historical classifier for every class.

For a batch $\{(\hat{x}_i, y_i)\}_{i=1}^B$, the class-specific predictive terms query $\sum_i |K_{y_i}|$ image–expert pairs. The statistic term processes the full mixed batch through every expert in $\bigcup_i K_{y_i}$. Eligibility therefore restricts predictive supervision, while the BN term supplies an aggregate feature prior. Server-side generator optimization removes generator training from clients, but clients still download, store, and execute the shared generator.

4.2 Reliability-Aware Generator Consolidation

At each task boundary, the cached client classifiers are frozen. For label y, let K_y contain usable cached experts whose real-data coverage masks include y; synthetic exposure never expands this mask. Optional hard rejection in the corrupted-update study can remove experts. Both server objectives sample labels from $\mathcal{C}_{\text{use}} = \{y \in \mathcal{C}_{\text{seen}} : |K_y| > 0\}$. Labels without usable experts are excluded from replay sampling. The objectives below are defined for nonempty $\mathcal{C}_{\text{use}}$; an empty support provides no teacher-defined replay objective. For a generated image $\hat{x}$, let $p_k(\cdot \mid \hat{x})$ be expert k's softmax distribution on $\mathcal{C}_{\text{all}}$, and let $\bar{p} = |K_y|^{-1}\sum_{k\in K_y} p_k$. For $|K_y| \geq 2$, we use generalized Jensen–Shannon disagreement:

$$u(\hat{x}, y) = H(\bar{p}) - \frac{1}{|K_y|}\sum_{k\in\mathcal{K}_y} H(p_k), \qquad w(\hat{x}, y) = \exp[-\beta u(\hat{x}, y)]. \tag{3}$$

With natural logarithms, $u = |K_y|^{-1}\sum_{k\in K_y}\mathrm{KL}(p_k\|\bar{p})$ satisfies $0 \leq u \leq \min\{\log|K_y|, \log|\mathcal{C}_{\text{all}}|\}$. The weight attenuates collective supervision on disputed samples; it does not rank the experts within K_y. A value of one is the neutral multiplier, including when experts agree on an uncertain or incorrect prediction. Requested-label cross-entropy remains active in these cases: identical uniform predictions still incur a classification loss of $\log|\mathcal{C}_{\text{all}}|$. Agreement and class guidance thus impose different constraints. Appendix I gives the weight bounds and branch-scale implications.

Single-expert fallback. For $|K_y| = 1$, disagreement is zero and provides no comparison between experts. FedRUA therefore uses a separate fallback for the sole expert k:

$$w_1(\hat{x}, y) = e^{-\beta_1 H(p_k)} e^{-\beta_2 d_k(\hat{X}_y)}\, s_k. \tag{4}$$

Here $d_k(\hat{X}_y)$ is the feature-statistic discrepancy of the clean class-y sub-batch. The confidence and feature factors attenuate this singleton's supervision. Their numerical scale differs from the multi-expert branch: for uniform predictions over $C = |\mathcal{C}_{\text{all}}|$ labels and $\beta_1 = 1$, the confidence factor is $1/C$, whereas identical multi-expert predictions receive $w = 1$. Consequently, the two branches implement different attenuation rules rather than a common calibrated reliability scale. For an upload produced in round q_k, let $\theta_{q_k}^{\text{ref}}$ be its pre-local-update server reference. The update-deviation factor is

$$\Delta_k = \|\theta_k - \theta_{q_k}^{\text{ref}}\|_2, \qquad s_k = \exp\Big\{-\alpha_s[\Delta_k - \tau_s\,\text{median}_{j\in\mathcal{S}_{q_k}}\,\Delta_j]_+\Big\}. \tag{5}$$

Here S_{q_k} is the participating-client set in upload round q_k, and $[v]_+ = \max(v, 0)$. The median compares updates from the same round and the reference model. The server computes s_k at upload

and retains it until that client's next upload. The factor measures update deviation, which can reflect local optimization and benign label skew as well as corruption. Continuous attenuation enters only the singleton fallback. The optional hard-rejection procedure in Appendix H.4 instead changes the usable expert set before consolidation.

Generator objective. Clients upload snapshots $(\mu_{k\ell}, \sigma^2_{k\ell})$ of the running buffers in their BN layers $\mathcal{S}_k$. These buffers can reflect both real data and local synthetic replay. For a classifier-facing batch $\hat{X}$, let $\mu^{\mathrm{gen}}_{k\ell}(\hat{X})$ and $(\sigma^{\mathrm{gen}}_{k\ell}(\hat{X}))^2$ denote channelwise moments in expert k, reduced over batch and spatial dimensions. With $\mathcal{Y}(\hat{X})$ denoting the labels in the batch, define $d_k(\hat{X}) = \sum_{\ell \in \mathcal{S}_k} \left(\left\| \mu^{\mathrm{gen}}_{k\ell}(\hat{X}) - \mu_{k\ell} \right\|_2 + \left\| (\sigma^{\mathrm{gen}}_{k\ell}(\hat{X}))^2 - \sigma^2_{k\ell} \right\|_2 \right)$,

$$\mathcal{K}(\hat{X}) = \bigcup_{y \in \mathcal{Y}(\hat{X})} K_y, \qquad \mathcal{L}_{\mathrm{stat}} = \frac{1}{|\mathcal{K}(\hat{X})|} \sum_{k \in \mathcal{K}(\hat{X})} d_k(\hat{X}).$$

Mixed-batch moment matching supplies a coarse feature prior. The same discrepancy on a clean class sub-batch, $d_k(\hat{X}_y)$, enters the singleton fallback and feature gate; a class-conditional distribution need not match the marginal BN targets. When a batch cannot contain every usable label, synthesis samples a label subset with equal representation within that subset. For nonempty K_y, define $r(\hat{x}, y) = \begin{cases} \mathrm{sg}[w(\hat{x}, y)], & |K_y| \geq 2, \\ \mathrm{sg}[w_1(\hat{x}, y)], & |K_y| = 1, \end{cases}$ where sg denotes stop-gradient. The expectations sample usable labels class-uniformly and latent inputs from p_z. The predictive losses and generator objective are

$$\mathcal{L}_{\mathrm{cls}} = \mathbb{E}_{y,z} \frac{r}{|\mathcal{K}_y|} \sum_{k \in \mathcal{K}_y} -\log p_k(y \mid \hat{x}),$$

$$\mathcal{L}_{\mathrm{kd}} = \mathbb{E}_{y,z} \frac{r}{|\mathcal{K}_y|} \sum_{k \in \mathcal{K}_y} \mathrm{KL}(p_k(\cdot \mid \hat{x}) \,\|\, p_\theta(\cdot \mid \hat{x})),$$

$$\phi^* = \arg\min_{\phi} \ \lambda_{\mathrm{cls}} \mathcal{L}_{\mathrm{cls}} + \lambda_{\mathrm{kd}} \mathcal{L}_{\mathrm{kd}} + \lambda_{\mathrm{stat}} \mathcal{L}_{\mathrm{stat}}. \tag{6}$$

For $m = |K_y| \geq 2$ and the KL in Eq. 6, the following unweighted per-sample identity holds:

$$\frac{1}{m} \sum_{k \in \mathcal{K}_y} \mathrm{KL}(p_k \| p_\theta) = u(\hat{x}, y) + \mathrm{KL}(\bar{p} \| p_\theta). \tag{7}$$

Positive KL penalizes predictive disagreement and mismatch between the expert mean and the global classifier. Its role is compatibility during synthesis. Historical labels enter through class-uniform sampling from $\mathcal{C}_{\mathrm{use}}$ and requested-label cross-entropy, independently of the global classifier's confidence; classifier replay then trains on those labels. The procedure can therefore rehearse historical classes without maximizing a forgetting score, although it cannot recover knowledge absent from all usable teachers. At generator iterate ϕ_j, the detached coefficient gives the surrogate gradient $\mathbb{E}[r_{\phi_j} \nabla_\phi L_\phi]_{\phi=\phi_j}$ for each predictive term. We retain absolute attenuation by averaging over samples without dividing by summed reliability. Multiplying every weight by a constant consequently changes the total predictive gradient; normalization by summed weights would cancel that change. Lower-weight supervision has less influence, and the unweighted BN prior becomes relatively stronger. Thus supervision strength and relative sample emphasis are both parts of the objective. The reported comparisons evaluate this combined design.

4.3 Feature-Gated Adversarial Server Replay

After consolidating G_{ϕ^*}, the server fine-tunes f_θ using labels in $\mathcal{C}_{\mathrm{use}}$. Let $x_{\tanh}$ be the generator output, $x_{\mathrm{pix}} = (x_{\tanh} + 1)/2$ its pixel-space representation, and $\hat{x} = N(x_{\mathrm{pix}})$ the classifier input, where $N_c(x) = (x_c - \mu_c)/\sigma_c$. FGSM is defined in $[0, 1]$ pixel coordinates: $x^{\mathrm{adv}}_{\mathrm{pix}} = \mathrm{clip}_{[0,1]} \left(x_{\mathrm{pix}} + \epsilon \ \mathrm{sign} \, \nabla_{x_{\mathrm{pix}}} \ell(f_\theta(\mathcal{N}(x_{\mathrm{pix}})), y) \right), \qquad \epsilon = 8/255.$

The perturbed classifier input is $\hat{x}^{\mathrm{adv}} = \mathcal{N}(x^{\mathrm{adv}}_{\mathrm{pix}})$; channel c therefore has normalized radius ϵ/σ_c and clipping limits $-\mu_c/\sigma_c$ and $(1 - \mu_c)/\sigma_c$. For a clean class sub-batch, define

$$\bar{d}(\hat{X}_y) = \frac{1}{|K_y|} \sum_{k \in K_y} d_k(\hat{X}_y), \qquad g(\hat{X}_y) = \mathbf{1}\{\bar{d}(\hat{X}_y) \leq \tau\}.$$

The boundary calibration rule sets τ to the 75th percentile of class-sub-batch discrepancies in the first clean consolidation batch. This establishes a reference for that boundary; it does not enforce a 75% admission rate in subsequent batches. During classifier replay, reliability is recomputed and detached on the current clean samples, and the current clean class sub-batch determines gate admission. The classifier objective is

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{y,z} r(\hat{x}, y) \left[\ell(f_\theta(\hat{x}), y) + \lambda_{\text{adv}} g(\hat{X}_y) \ell(f_\theta(\hat{x}_{\text{adv}}), y) \right]. \tag{8}$$

The gate controls only the additional adversarial term: when $g(\hat{X}_y) = 0$, the weighted clean-replay loss remains active. It therefore limits extra perturbation-based training without removing a usable class from clean replay. For pixel-space cross-entropy L_θ, a first-order expansion away from clipping boundaries gives $L_\theta(x_{\text{pix}}^{\text{adv}}, y) \approx L_\theta(x_{\text{pix}}, y) + \epsilon \|\nabla_{x_{\text{pix}}} L_\theta\|_1$. The branch adds both clean-loss mass and local input-sensitivity regularization. Table 3 evaluates their combined effect; Appendix I explains the gate's statistical scope. Appendix D separately provides a diagnostic decomposition of replay-distribution discrepancy.

5 Experiments

5.1 Protocol

The evaluation measures final average accuracy and forgetting under synchronized stage progression with client-specific task orders. CIFAR-10, CIFAR-100, and 64×64 ImageNet-1K use five disjoint task groups with 2, 20, and 200 classes per group, respectively. A shared stage clock does not imply that clients encounter the same group simultaneously. The default configuration uses $K = 20$ fully participating clients, a randomly initialized ResNet-18 classifier, five local epochs per round, five rounds per stage, 200 server replay samples per seen class, and a 1:1 local real-to-synthetic ratio. IID allocation is the default; Dirichlet allocation varies spatial label skew. Accuracy and forgetting tables report mean $\pm$ standard deviation over five runs. Appendix E documents the supplied configuration, and Appendix F defines the metrics. Numerical differences in the discussion are percentage-point differences unless explicitly described as relative percentages.

5.2 Core Accuracy and Retention

Table 1: Synchronous class-incremental results (%); mean $\pm$ SD over five runs. DGR-FCL is the highest-accuracy retained comparator.

Data	Method	$A_{avg} \uparrow$	$F \downarrow$
CIFAR-100	FedCIL (Qi et al., 2023)	55.36±0.40	10.60±0.23
	TARGET (Zhang et al., 2023)	57.12±0.38	7.70±0.20
	AF-FCL (Wuerkaixi et al., 2024)	51.30±0.45	16.40±0.29
	CAN (Rong et al., 2025)	56.20±0.50	12.50±0.60
	DGR-FCL (He et al., 2026)	57.50±0.35	6.80±0.20
	FedRUA	58.38±0.32	5.10±0.15
ImageNet-1K	FedCIL	42.72±0.32	8.80±0.22
	TARGET	43.72±0.30	7.00±0.20
	AF-FCL	39.30±0.35	13.70±0.28
	CAN	43.80±0.40	12.00±0.50
	DGR-FCL	44.00±0.30	6.90±0.18
	FedRUA	44.30±0.25	6.20±0.17

On CIFAR-100, FedRUA reaches 58.38% final accuracy and 5.10% forgetting, exceeding DGR-FCL in accuracy by 0.88 percentage points while reducing forgetting by 1.70 points. The joint improvement is more informative than low forgetting alone, which can also accompany weak initial learning. On 64×64 ImageNet-1K, the corresponding margins are 0.30 and 0.70 points. These are differences between reported means; the separate standard deviations do not establish paired statistical significance. Appendix G.1 provides the expanded comparisons.

Table 2: CIFAR-100 with Dirichlet label skew ($\alpha = 0.5$; %); mean $\pm$ SD over five runs.

Method	$A_{avg} \uparrow$	$F \downarrow$
TARGET	51.20 ± 0.48	11.50 ± 0.28
AF-FCL	38.90 ± 0.65	22.50 ± 0.42
CAN	52.80 ± 0.50	12.20 ± 0.40
DGR-FCL	55.80 ± 0.45	8.20 ± 0.30
FedRUA	57.25 ± 0.38	6.80 ± 0.20

5.3 Spatial Heterogeneity

Under $\alpha = 0.5$, FedRUA's accuracy decreases by 1.13 percentage points from IID, compared with 1.70 points for DGR-FCL; their accuracy separation consequently increases from 0.88 to 1.45 points. Forgetting behaves differently: FedRUA remains lower in absolute value, but its increase from IID is 1.70 points versus 1.40 points for DGR-FCL. The results therefore show lower accuracy sensitivity to this allocation without a uniformly smaller change in both metrics. In Appendix G.5, FedRUA exceeds TARGET by 7.80, 6.05, and 3.80 accuracy points at $\alpha = 0.1$, 0.5, and 1.0, respectively. DGR-FCL and CAN are absent from that broader sweep, so those margins apply to the methods actually included.

5.4 What Drives the Reported Gain?

Table 3: Objective/branch comparison on CIFAR-100 ($T = 5$); first five rows alter generator losses and the last removes adversarial replay.

Variant	$\mathcal{L}_{stat}$	$\mathcal{L}_{cls}$	$\mathcal{L}_{kd}$	Adv.	$A_{avg} \uparrow$	$F \downarrow$
BN statistics only	✓	×	×	✓	51.95 ± 0.66	20.10 ± 0.86
BN + classification	✓	✓	×	✓	54.80 ± 0.58	16.20 ± 0.74
BN + distillation	✓	×	✓	✓	56.12 ± 0.45	10.35 ± 0.52
Classification + distillation	×	✓	✓	✓	57.95 ± 0.38	7.15 ± 0.28
Full FedRUA	✓	✓	✓	✓	58.38 ± 0.32	5.10 ± 0.15
Full generator, no adv. replay	✓	✓	✓	×	55.38 ± 0.50	8.30 ± 0.60

The omissions in Table 3 affect accuracy and forgetting differently. Removing predictive distillation decreases accuracy by 3.58 percentage points and increases forgetting by 11.10 points, the largest change among the three single-loss omissions. Removing classification guidance changes the two metrics by -2.26 and $+5.25$ points. Removing statistic matching changes accuracy by -0.43 points but forgetting by $+2.05$ points, showing that an accuracy-only comparison understates its association with retention. The BN-only generator is 6.43 accuracy points below the full configuration and has 15.00 points more forgetting. Removing the adversarial replay branch changes accuracy by -3.00 and forgetting by $+3.20$ points. These conditional comparisons support the complete loss and branch configuration, but do not separate the effects of eligibility, disagreement weighting, singleton factors, or gate selection. The client-generator-averaging comparison (52.15% accuracy; 19.45% forgetting) changes the generator-training arrangement and is a separate system-level comparison.

5.5 CNN Backbones and Initialization

Table 4: CNN backbone/initialization results on CIFAR-100 (%) versus the reference generative baseline TARGET.

Backbone	Init.	Method	$A_{avg} \uparrow$	$F \downarrow$
ResNet-18	Scratch	TARGET	57.12 ± 0.38	7.70 ± 0.20
		FedRUA	58.38 ± 0.32	5.10 ± 0.15
ResNet-18	Pretrained	TARGET	60.78 ± 0.34	6.36 ± 0.24
		FedRUA	62.14 ± 0.29	4.18 ± 0.17
MobileNetV2	Scratch	TARGET	55.46 ± 0.46	8.38 ± 0.31
		FedRUA	56.87 ± 0.41	6.14 ± 0.25
MobileNetV2	Pretrained	TARGET	59.18 ± 0.39	6.92 ± 0.26
		FedRUA	60.72 ± 0.35	4.91 ± 0.20

Across the four backbone/initialization pairs in Table 4, FedRUA exceeds TARGET by 1.26–1.54 accuracy points and has 2.01–2.60 points less forgetting. ImageNet pretraining increases ResNet-18 accuracy by 3.76 points for FedRUA and 3.66 points for TARGET, while the within-pair method difference persists. The comparison therefore shows a consistent direction across the tested CNN representations and initialization settings. Both backbones provide the BN buffers required by the method.

5.6 Partial Participation and Server Cost

The cached-expert policy retains the latest uploaded model, feature buffers, real-data coverage, and update-deviation factor for absent clients. Table 5 characterizes FedRUA under three participation configurations. At $K = 200$, increasing the active set from 10 to 20 clients per round improves accuracy by 0.79 points and reduces forgetting by 0.85 points. At a fixed 10% participation rate, increasing the pool from 200 to 500 also increases the active set from 20 to 50; accuracy decreases by 0.51 points and forgetting increases by 0.38 points. These configurations jointly vary participation, available local data, update frequency, and potentially cache freshness. They characterize the complete operating regimes rather than isolating a causal benefit of caching. Clients also store and run the shared generator, so its local memory and inference costs remain part of the system.

Table 5: FedRUA under partial participation on CIFAR-100 (%); mean ± SD.

K	Active	Rate	$A_{avg} \uparrow$	$F \downarrow$
200	10	5%	54.63 ± 0.48	8.67 ± 0.31
200	20	10%	55.42 ± 0.44	7.82 ± 0.27
500	50	10%	54.91 ± 0.50	8.20 ± 0.30

With a pool of 200 clients, increasing the active set from 10 to 20 per round changes accuracy by $+0.79\%$ and forgetting by -0.85%. More frequent participation could refresh cached experts and deliver more current classifier updates, but the table cannot separate these mechanisms. Holding the participation rate at 10% while growing the pool from 200 to 500 also raises the active count from 20 to 50; accuracy changes by -0.51 and forgetting by $+0.38\%$. Thus pool size, absolute active count, per-client data, class coverage, and cache age cannot be collapsed into a single participation-rate variable. The shared generator is one optimized model but is cached on clients, so client memory, download, and inference costs are distinct from server training.

Table 6: Single recorded CIFAR-100 task-boundary server profile ($K = 50$).

Method	Generator (minutes)	Synthesis (minutes)	Classifier (minutes)	Total (hours)
TARGET	210	10	68	4.80
DGR-FCL	140	180	70	6.50
FedRUA	**120**	**5**	**46**	**2.85**

Table 6 reports one task-boundary server profile. FedRUA's three phases total 171 minutes (2.85 hours), of which generator consolidation occupies 120 minutes, approximately 70%, and synthesis occupies five minutes. Consolidation therefore dominates this profile; inexpensive sample generation alone does not describe the server workload. The displayed totals for TARGET and DGR-FCL are 4.80 and 6.50 hours. These measurements compare the recorded boundary phases and exclude ordinary-round training and evaluation, so they do not establish an end-to-end training speedup.

5.7 Replay Budget and Sensitivity

The result was recorded in Table 17, Appendix H.3. Generally, increasing server replay from 10 to 100 samples per class improves accuracy by 3.70 points and reduces forgetting by 5.00 points. Increasing from 100 to 200 adds 0.48 accuracy points and reduces forgetting by 1.40 points; increasing from 200 to 500 adds only 0.22 accuracy points and reduces forgetting by 0.30 points. This diminishing marginal improvement motivates reporting 200 samples as the common operating point, without establishing a compute optimum. With fixed classifier epochs, larger replay sets can also increase optimizer steps. Table 18 in Appendix H.3 shows that the ResNet generator exceeds the default Advanced CNN by 0.47 accuracy points and has 0.40 points less forgetting; these are downstream classifier outcomes rather than image-fidelity or efficiency measurements. Tables 19– 20

in Appendix H.3 vary the client real-to-synthetic ratio. The 1:1 ratio gives the highest accuracy and lowest forgetting at five and ten tasks. At twenty tasks, it reduces forgetting by 4.23 points relative to 1:2 while reducing accuracy by only 0.11 points, illustrating a tradeoff rather than an optimal ratio.

5.8 Full-Participation Client Scaling

Table 7: FedRUA client-count scaling on CIFAR-100 with full participation (five tasks; %).

K	$A_{avg} \uparrow$	$F \downarrow$
20	58.38 ± 0.32	5.10 ± 0.15
50	57.90 ± 0.35	5.55 ± 0.18
100	57.25 ± 0.38	6.10 ± 0.20
200	56.60 ± 0.42	6.85 ± 0.25

In Table 7, increasing the fully participating client population from 20 to 200 decreases accuracy by 1.78 points and increases forgetting by 1.75 points. Because the total dataset is fixed, larger populations also reduce examples per client; five local epochs can then correspond to fewer local optimizer steps. The results characterize a joint change in federation size, local data availability, and expert training. They show gradual degradation across these configurations without any isolation .

5.9 Corrupted-Update Stress Test

Table 21 in Appendix H.4 evaluates Gaussian weight-noise corruption ($\sigma = 0.1$) affecting 20% or 40% of $K = 20$ clients. Relative to $\tau_s = \infty$, the $\tau_s = 1.5$ setting improves accuracy by 4.60 and 7.40 points and reduces forgetting by 6.20 and 11.20 points, respectively. The stricter threshold $\tau_s = 1.2$ has the highest accuracy and lowest forgetting in both regimes, while rejecting more updates. The comparison evaluates the thresholded procedure under this specified perturbation; it does not isolate continuous singleton attenuation or establish robustness to adaptive poisoning. Appendix H.4 distinguishes attenuation, rejection, and feature gating.

6 Limitations

FedRUA weights supervision using proxies for usefulness. Shared incorrect predictions can receive high multi-expert weight, and the singleton branch has a different numerical scale that can suppress rare-class supervision. The class-specific feature gate compares generated batches with marginal BN buffers, so admission is neither a semantic-validity test nor a guarantee of boundary preservation. The objective and branch comparisons evaluate the integrated configuration without independently identifying every reliability factor.

The server accesses individual client models and feature buffers, and cached expertise can become stale. Synthesis requires expert forwards and input gradients, while clients retain generator storage and inference costs. The evaluation covers synchronized stage progression, client-specific task orders, spatial label skew, and BN-based CNNs. Different client stage clocks, non-BN architectures, formal privacy, and adaptive-attack robustness are outside the reported evidence. Longer-sequence results are within-method ratio comparisons, and the timing measurements cover one task-boundary profile. Appendix I connects these failure conditions to concrete diagnostics.

7 Conclusion

FedRUA combines class-eligible predictive supervision, disagreement-based attenuation, and marginal feature statistics in one server-trained conditional generator. A second stage applies reliability-weighted clean replay and feature-gated adversarial regularization to the global classifier. The reported five-task comparisons show higher mean final accuracy and lower mean forgetting than the listed replay baselines on CIFAR-100 and 64×64 ImageNet-1K, with a larger CIFAR-100 accuracy margin under moderate spatial label skew. The objective and branch comparisons support the complete procedure, while the longer-horizon results delimit its current retention capability. The contribution is a concrete organization of supervision selection and replay regularization for a trusted server, with explicit distinctions between class coverage, predictive consistency, and classifier utility.

Ethics Statement

FedRUA avoids transmitting raw client data, but uploaded classifiers and feature statistics may still leak information through model inversion or membership inference; we claim no formal privacy guarantee. Sensitive-domain deployment would need differential privacy or secure aggregation on top of the mechanisms described here. Reliance on a trusted server concentrates computational authority; governance mechanisms may be needed when server/client incentives diverge, for example to audit updates attenuated by Eq. 5.

Dataset licenses. CIFAR-10/100 are used under the terms of their original distribution and ImageNet-1K under its researcher agreement (we use derived, resized 64×64 copies). The WILDS Camelyon17 distribution is in the public domain under CC0 (Koh et al., 2021); its earlier custom numerical study is not retained because the exact distribution and split protocol are undocumented in this archive.

AI Use Statement

We used generative-AI tools (Claude by Anthropic and Codex by OpenAI) to assist with editing text for clarity, checking arithmetic consistency across tables, and drafting LaTeX formatting. All scientific content – problem formulation, method design, experimental protocol, result interpretation, and claims – is the authors' own work; we take full responsibility for the final content.

References

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=B7v4QMR6Z9w.

Yavuz Faruk Bakman, Duygu Nur Yaldiz, Yahya H. Ezzeldin, and Salman Avestimehr. Federated orthogonal training: Mitigating global catastrophic forgetting in continual federated learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nAs4LdaP9Y.

Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. GS-WGAN: A gradient-sanitized approach for learning differentially private generators. *Advances in Neural Information Processing Systems*, 33:12673–12684, 2020.

Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10164–10173, 2022.

Xin Gao, Xin Yang, Hao Yu, Yan Kang, and Tianrui Li. FedProK: Trustworthy federated class-incremental learning via prototypical feature knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4205–4214, 2024.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

Haiyang Guo, Fei Zhu, Wenzhuo Liu, Xu-Yao Zhang, and Cheng-Lin Liu. PiLoRA: Prototype-guided incremental LoRA for federated class-incremental learning. In *European Conference on Computer Vision*, pp. 141–159. Springer, 2024.

Zaobo He, Yunkun Wang, and Zhipeng Cai. Federated continual learning with bounded forgetting via diffusion-based generative replay in edge computing. *IEEE Transactions on Mobile Computing*, 25(3):3001–3017, 2026. doi: 10.1109/TMC.2025.3618275.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/karimireddy20a.html.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Yichen Li, Haozhao Wang, Yining Qi, Wei Liu, and Ruixuan Li. Re-fed+: A better replay strategy for federated incremental learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5489–5500, 2025a. doi: 10.1109/TPAMI.2025.3551732.

Yichen Li, Yuying Wang, Haozhao Wang, Yining Qi, Tianzhe Xiao, and Ruixuan Li. FedSSI: Rehearsal-free continual federated learning with synergistic synaptic intelligence. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=9hFQvmCl7P.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Daiqing Qi, Handong Zhao, and Sheng Li. Better generative replay for continual federated learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cRxYWKiTan.

Mohammad Rasouli, Tao Sun, and Ram Rajagopal. FedGAN: Federated generative adversarial networks for distributed data. *arXiv preprint arXiv:2006.07228*, 2020.

Xuankun Rong, Jianshu Zhang, Kun He, and Mang Ye. CAN: Leveraging clients as navigators for generative replay in federated continual learning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=lvkVhZ776k.

Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23870–23880, 2024.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in Neural Information Processing Systems*, 33:7611–7623, 2020.

Abudukelimu Wuerkaixi, Sen Cui, Jingfeng Zhang, Kunda Yan, Bo Han, Gang Niu, Lei Fang, Changshui Zhang, and Masashi Sugiyama. Accurate forgetting for heterogeneous federated continual learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ShQrnAsbPI.

Kunlun Xu, yibo feng, Jiangmeng Li, Yongsheng Qi, and Jiahuan Zhou. C^{2}Prompt: Class-aware client knowledge interaction for federated continual learning. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=pKqLOmF3Lf.

Xiaoyang Yi, Yang Liu, Binhan Yang, and Jian Zhang. Federated continual learning via orchestrating multi-scale expertise. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=AlSHcopwHi.

Hongxu Yin, Pavlo Molchanov, Jose M. Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via DeepInversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pp. 12073–12086. PMLR, 2021.

Hao Yu, Xin Yang, Le Zhang, Hanlin Gu, Tianrui Li, Lixin Fan, and Qiang Yang. Handling spatial-temporal data heterogeneity for federated continual learning via tail anchor. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4874–4883, June 2025.

Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. TARGET: Federated class-continual learning via exemplar-free distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4782–4793, 2023.

Yifei Zhang, Hao Zhu, Alysa Ziying Tan, Dianzhi Yu, Longtao Huang, and Han Yu. pfedmxf: Personalized federated class-incremental learning with mixture of frequency aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30640–30650, 2025.

A NOTATION

Table 8: Notation used by the method and diagnostic analysis.

Symbol	Definition
$K, \pi_k(r), \rho_k(s)$	Number of clients, client k's stage at round r, and the task identity assigned to its stage s.
$\mathcal{C}_{\text{all}}, \mathcal{C}_{\text{seen}}(r)$	Dataset labels and labels introduced globally through round r.
$\mathcal{C}_{\text{pred}}(r)$	Label support used for checkpoint prediction, as defined in Section 3.
p_z	Generator latent prior.
$d_k(\hat{X}), d_k(\hat{X}_y)$	Expert-specific BN discrepancies on a mixed batch and a clean class sub-batch, respectively (Section 4.2).
$g(\hat{X}_y)$	Feature gate $\mathbf{1}\{\bar{d}(\hat{X}_y) \leq \tau\}$ for the additional adversarial loss.
$r(\hat{x}, y)$	Detached w for at least two usable experts, or detached w_1 for a singleton.
β, β_1, β_2	Disagreement, singleton-entropy, and singleton-feature scaling: 1.0, 1.0, and 0.01. These parameters act on different quantities.
s_k	Cached instantaneous update-deviation attenuation factor (Equation 5).
α_s, τ_s	Update-deviation decay scaling and threshold multiplier; the recorded defaults are 1.0 and 1.5.
τ	Feature-gate threshold; the stated rule uses the 75th percentile of clean class-sub-batch discrepancies in the first consolidation batch at each boundary.
$\mathcal{S}_k$	BN layers in expert k used for feature-statistic matching (Section 4.2).

B ADDITIONAL RELATED WORK

Federated generative modeling and private synthetic-data release address questions complementary to continual replay. FedGAN studies adversarial generation across distributed data (Rasouli et al., 2020); GS-WGAN uses gradient sanitization to learn differentially private generators (Chen et al., 2020). FedRUA instead performs task-boundary synthesis from frozen client classifiers and their feature buffers, without a discriminator trained on real examples. Image replay can be passed through a changing encoder, but its utility depends on synthesis quality and incurs generator and classifier-inference costs. FedRUA does not inherit a privacy guarantee from these related generative methods.

Parameter- and feature-based alternatives avoid image synthesis. FedWeIT separates shared and task-adaptive parameters (Yoon et al., 2021); LANDER transfers language-guided semantic information (Tran et al., 2024); C^2Prompt adapts a pretrained ViT with prompts (Xu et al., 2025); and FedProK communicates prototypical feature knowledge (Gao et al., 2024). Features and prototypes are compact but their interpretation depends on the representation in which they were formed; prompts and parameter isolation instead assume particular model structures or pretraining. These are different capacity, storage, and optimization regimes. FedRUA targets BN-based CNN experts and server-side image replay; the present tables compare complete systems rather than all representation choices under equal resources.

C ALGORITHMS

Algorithm 1 FEDRUA client/server workflow

Initialize: server classifier f_θ, one server generator G_ϕ, and an empty cache for each client.
for each communication round r **do**
  Select participating clients $\mathcal{S}_r$ and broadcast f_θ.
  Broadcast G_ϕ only if its task-boundary version changed or a selected client lacks that version.
  for each $k \in \mathcal{S}_r$ in parallel **do**
    Generate replay only for labels in the client's earlier real-data coverage; set the replay contribution to zero when that set is empty, and keep G_ϕ frozen.
    Update f_{θ_k} on current real data and generated replay using Eq. 2.
    Upload f_{θ_k}, real-data coverage, and BN-buffer snapshots.
  end for
  Compute update-deviation factors from each upload and its pre-local reference; refresh selected-client cache entries and aggregate participating classifiers.
  if the configured task boundary is reached **then**
    Consolidate G_ϕ and then fine-tune f_θ using Algorithm 2.
  end if
end for
Return: f_θ. Clients cache the server snapshot for inference but never optimize a generator.

Algorithm 2 Reliability-aware task-boundary consolidation

Input: G_ϕ, f_θ and cached $(f_{\theta_k}, \mathcal{C}_k^{\text{real}}, \text{BN buffers}_k, s_k)$.
Freeze all eligible client experts.
Form usable expert sets K_y and $\mathcal{C}_{\text{use}} = \{y \in \mathcal{C}_{\text{seen}} : |K_y| > 0\}$. The following replay updates require nonempty $\mathcal{C}_{\text{use}}$.
for each generator update **do**
  Sample usable labels and latent inputs; form classifier-facing images $\hat{x} = \mathcal{N}((G_\phi(z, y)+1)/2)$ in a class-balanced mixed batch $\hat{X}$.
  Compute $\mathcal{L}_{\text{stat}}$ on the full mixed batch for every expert in $\mathcal{K}(\hat{X}) = \bigcup_{y \in \mathcal{Y}(\hat{X})} K_y$.
  Form the clean class sub-batches $\hat{X}_y$ needed for the singleton feature factors.
  if $|\mathcal{K}_y| \geq 2$ **then**
    Compute disagreement u and $w = \exp(-\beta u)$ using Eq. 3.
  else if $|\mathcal{K}_y| = 1$ **then**
    Compute the confidence, feature-consistency, and trust fallback w_1 using Eq. 4.
  end if
  Detach the reliability coefficient and update G_ϕ with Eq. 6.
end for
Freeze the consolidated generator G_{ϕ^*}; use the boundary-specific feature threshold defined in Section 4.3 for classifier replay.
for each global-classifier update **do**
  Sample clean replay from $\mathcal{C}_{\text{use}}$; recompute and detach the applicable w or w_1, and evaluate $g(\hat{X}_y)$ on each current clean class sub-batch.
  For admitted sub-batches, construct FGSM in pixel coordinates and normalize the resulting images for classifier input.
  Update f_θ with reliability-weighted clean and adversarial replay using Eq. 8.
end for
Return: G_{ϕ^*}, f_{θ^*}.

The coverage mask records real-data exposure only. During synthesis, fixed experts and the fixed global classifier transmit input gradients to the generator; during classifier replay, the generator is fixed and the classifier is optimized. Reliability and feature admission are evaluated on the clean samples used by their respective stage. The pseudocode describes this mathematical procedure and its nonempty-support domain.

D Diagnostic Risk Decomposition

This decomposition concerns distribution discrepancy for a bounded evaluation loss, not the generalization of a trained deep network. Let $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0, 1]$. For client k, define $P_k = \sum_s \omega_{k,s} P_{k,s}$, where $P_{k,s}$ is a historical task distribution, $\omega_{k,s} \geq 0$, and $\sum_s \omega_{k,s} = 1$. Let $a_k \geq 0$ with $\sum_k a_k = 1$, and set $\bar{P} = \sum_k a_k P_k$. These normalized mixtures differ from the unnormalized historical sum in Equation 1. Let Q_k be any conceptual client-specific replay approximation and let Q be the shared generator's replay distribution. The algorithm does not fit separate client generators to represent the Q_k. Define

$$\mathrm{disc}_\ell(P, Q) = \sup_{h \in \mathcal{H}} \left| \mathbb{E}_P \ell(h(X), Y) - \mathbb{E}_Q \ell(h(X), Y) \right|.$$

Lemma 1 (Replay-discrepancy decomposition)**.** *For the distributions above, with $\bar{Q} = \sum_k \alpha_k Q_k$,*

$$\mathrm{disc}_\ell(\bar{P}, Q) \leq \sum_k \alpha_k \, \mathrm{disc}_\ell(P_k, Q_k) + \mathrm{disc}_\ell(\bar{Q}, Q).$$

Proof. For any $h \in \mathcal{H}$, add and subtract the risk under $\bar{Q} = \sum_k a_k Q_k$. The triangle inequality gives

$$\begin{aligned} |\mathbb{E}_{\bar{P}} \ell(h(X), Y) - \mathbb{E}_Q \ell(h(X), Y)| &\leq \sum_k a_k |\mathbb{E}_{P_k} \ell(h(X), Y) - \mathbb{E}_{Q_k} \ell(h(X), Y)| \\ &\quad + |\mathbb{E}_{\bar{Q}} \ell(h(X), Y) - \mathbb{E}_Q \ell(h(X), Y)| \\ &\leq \sum_k a_k \, \mathrm{disc}_\ell(P_k, Q_k) + \mathrm{disc}_\ell(\bar{Q}, Q). \end{aligned}$$

Taking the supremum over h proves the claim. Applying the same triangle inequality to the mixture also yields $\mathrm{disc}_\ell(\bar{Q}, Q) \leq \sum_k a_k \, \mathrm{disc}_\ell(Q_k, Q)$. □

The first term separates errors in approximating each client's historical distribution; the second measures discrepancy between the mixture of these approximations and the consolidated replay distribution. This organization motivates examining synthesis and consolidation separately. It does not show that predictive disagreement, BN mismatch, or the proposed weights upper-bound either term, and minimizing those proxies need not decrease the discrepancy. Classifier replay changes the chosen hypothesis, whereas the lemma takes a supremum over a fixed hypothesis class. The empirical loss and branch comparisons therefore provide evidence separate from this distributional identity. The bounded-loss premise also does not apply directly to unbounded cross-entropy.

E Implementation Details

E.1 Experimental Setup

Datasets and scenarios. CIFAR-10, CIFAR-100, and ImageNet-1K use five disjoint class-incremental tasks. The default experiment has $K = 20$ clients, full participation, a shared task clock, and an IID within-task allocation. Spatial non-IID experiments draw each class allocation from a Dirichlet distribution, with $\alpha = 0.5$ unless varied. The numerical tables do not test different client task clocks or a medical-domain sequence.

Baselines. Standard FL baselines are FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), FedNova (Wang et al., 2020), SCAFFOLD (Karimireddy et al., 2020), and FedDyn (Acar et al., 2021). FCL comparisons with reported rows include AF-FCL (Wuerkaixi et al., 2024), FedWeIT (Yoon et al., 2021), GLFC (Dong et al., 2022), FedCIL (Qi et al., 2023), TARGET (Zhang et al., 2023), MFCL (Yi et al., 2025), CAN (Rong et al., 2025), and DGR-FCL (He et al., 2026).

Datasets and scenarios. CIFAR-10, CIFAR-100, and ImageNet-1K use five disjoint task groups in the default evaluation. Clients follow individual task permutations under a shared stage clock. The default uses $K = 20$, full participation, and IID within-task allocation. Spatial non-IID experiments allocate class data using a Dirichlet distribution with $\alpha = 0.5$ unless varied. The longer-sequence studies use ten or twenty task groups. The reported tables cover task-order and spatial heterogeneity, without varying client stage clocks or introducing a medical-domain sequence.

E.2 Hardware and Software Environment

The supplied configuration lists one NVIDIA A100 GPU with 40 GB of memory, Python 3.10, PyTorch 2.0.1, CUDA 11.8, and torchvision 0.15. Table 6 reports the task-boundary server profile in this environment.

E.3 Model Architectures

Classifier backbone. The default classifier is torchvision ResNet-18, with a final layer containing $|\mathcal{C}_{\text{all}}|$ outputs. It is randomly initialized except in the pretraining study. Table 19 also evaluates MobileNetV2. Both architectures provide the BN running buffers used by the feature-statistic discrepancy in Section 4.2.

Generator Architectures. The default generator (Advanced CNN in Table 18) is a conditional architecture. The input noise $z \in \mathbb{R}^{100}$ is concatenated with a 128-dimensional learned class embedding and projected to a $4 \times 4 \times 512$ spatial tensor.

CIFAR-10/100 (32×32*).* Three stride-2 transposed-convolution blocks upsample to 32×32: $512 \to 256$ (8×8), $256 \to 128$ (16×16), $128 \to 3$ (32×32). All transposed convolutions use 4×4 kernels with stride 2 and padding 1. Each intermediate block uses batch normalization and ReLU; the final block applies Tanh directly, with no additional 1×1 convolution.

ImageNet-1K (64×64*).* Four stride-2 blocks: $512 \to 256$ (8×8), $256 \to 128$ (16×16), $128 \to 64$ (32×32), $64 \to 3$ (64×64). All four blocks use the same 4×4 kernel, stride-2, padding-1 configuration as the CIFAR generator; intermediate blocks use batch normalization and ReLU, and the final block uses Tanh.

The generator's Tanh output is mapped to pixels by $x_{\text{pix}} = (x_{\tanh} + 1)/2$, then normalized as $x_{\text{norm},c} = (x_{\text{pix},c} - \mu_c)/\sigma_c$. Real images already expressed in $[0, 1]$ pixel coordinates require only channelwise normalization; the Tanh-to-pixel mapping applies only to generator output. FGSM is constructed and clipped in pixel space before normalization, as defined in Section 4.3. The MLP and residual generator variants appear only in the architecture sweep. No discriminator trained on real data is part of FedRUA.

E.4 Training Hyperparameters

Local client training. Each selected client trains its classifier for five local epochs per round with Adam, learning rate 10^{-3}, optimizer coefficients $(0.9, 0.999)$, batch size 64, no weight decay, and no learning-rate schedule. The default real-to-synthetic sample ratio is 1:1. Equation 2 defines separately averaged real and replay losses with $\lambda_{\text{rep}} = 1$. Under this definition, changing sample counts changes exposure and sampling variability without changing the explicit loss coefficients. The shared generator remains frozen during local training.

Server-side generator consolidation. The supplied configuration specifies Adam with learning rate 2×10^{-4}, optimizer coefficients $(0.5, 0.999)$, batch size 128, no learning-rate decay, and a budget labeled "100 epochs." The reported weights are $\lambda_{\text{cls}} = 1$, $\lambda_{\text{kd}} = 1$, $\lambda_{\text{stat}} = 0.1$, $\lambda_{\text{adv}} = 0.5$, $\beta = 1$, $\beta_1 = 1$, and $\beta_2 = 0.01$. Experts run in evaluation mode. Mixed batches determine $\mathcal{L}_{\text{stat}}$, while class sub-batches determine singleton feature factors and gate discrepancies. Reliability weights are detached. Expert and global-classifier parameters remain fixed while input gradients update only ϕ; the subsequent replay stage fixes the generator and updates only θ.

Global classifier fine-tuning. After generator consolidation, the classifier is fine-tuned for 50 epochs using SGD with momentum 0.9, learning rate 10^{-3}, and batch size 128. The default replay set contains 200 examples per seen class with usable expert support. Table 7 varies this count.

Federated schedule. The default uses $K = 20$, full participation, and five rounds per local stage. Participating classifiers are aggregated after each round using current local data sizes. At configured stage boundaries, generator consolidation precedes server classifier replay. The partial-participation configuration samples an active set uniformly and retains absent clients' latest model, real-data coverage, BN buffers, and update-deviation factor.

E.5 Data Partitioning and Preprocessing

Class-incremental task splits. The default CIFAR-10, CIFAR-100, and ImageNet-1K splits contain five disjoint task groups with 2, 20, and 200 classes each. Client k encounters these groups in the order specified by ρ_k, while all clients advance under the same stage clock. Task identity and local stage index are therefore distinct. Comparisons are described using shared task groups and client allocations; the aggregate tables do not encode the individual permutation files.

Client data distribution. IID allocation distributes each task's data uniformly across the K clients. For spatial non-IID allocation, each class c has proportions $p_c \sim \mathrm{Dirichlet}(\alpha \mathbf{1}_K)$, which determine client sample allocations. The default is $\alpha = 0.5$, and smaller values yield more unequal class proportions. This distribution governs spatial allocation separately from the client task permutations.

Data augmentation and preprocessing. The supplied CIFAR-10/100 configuration uses channel means $(0.4914, 0.4822, 0.4465)$ and standard deviations $(0.2023, 0.1994, 0.2010)$. ImageNet-1K is represented at 64×64 resolution with the normalization described in the configuration as the standard torchvision convention. Local real-image training uses random horizontal flipping and random cropping with four-pixel padding; synthetic replay receives no such augmentation. Generator outputs are mapped from Tanh coordinates to $[0, 1]$ pixels. FGSM adds a pixel-space perturbation, clips the result to $[0, 1]$, and then applies channelwise normalization. Its normalized radius in channel c is $(8/255)/\sigma_c$.

E.6 Baseline Implementations

The reported comparison setup specifies a shared ResNet-18 classifier, task split, client allocation, local training schedule, and evaluation framework. The rows nevertheless represent complete methods with different replay mechanisms and server workloads. In particular, FedProx+Replay uses 100 samples per class, compared with FedRUA's default of 200. The results therefore compare the documented system configurations rather than establish equal replay capacity, server optimization effort, or tuning cost for every method.

FedProx+Replay. This in-house composition combines FedProx with $\mu = 0.01$ and a Light-CNN generator trained locally for 50 epochs per task using Adam with learning rate 2×10^{-4} and first-moment coefficient 0.5. It refreshes 100 replay samples per seen class after each task. These trainable client generators belong to this baseline; FedRUA clients instead use a frozen server-trained generator.

FedDyn+EWC. The supplied configuration combines FedDyn with coefficient 0.1 and an EWC local regularizer with coefficient 100, using a Fisher information estimate formed after each task. This composition is reported separately from the unmodified FedDyn baseline.

Scope of the implementation record. The configuration summary gives architectures, nominal schedules, and aggregate results, but does not specify the latent-noise distribution, the exact client replay-label sampler, generator updates per nominal epoch, exact class-sub-batch sizes, partition seeds and rounding rules, ImageNet resize/crop interpolation, or complete baseline revisions and tuning budgets. It also does not establish the local loss reduction used by every reported ratio run or the placement of all historical evaluation checkpoints relative to final consolidation. Consequently, the mathematical definitions above specify the described procedure, while the aggregate tables alone do not identify every implementation choice. The comparisons are interpreted as reported system outcomes rather than verified equal-budget or paired-significance results.

F Evaluation Metrics

Let q index recorded evaluation checkpoints, q_f the final checkpoint, and $\mathcal{I}(q)$ the set of task identities introduced across clients by checkpoint q. Let $a_{q,i}$ be accuracy on task i's held-out set using the prediction support $\mathcal{C}_{\mathrm{pred}}$ defined in Section 3 at that checkpoint. For each task, b_i is its first recorded post-training checkpoint after introduction. Client-specific task orders can introduce several task identities at one shared stage, so task identity is not interchangeable with stage index.

Final average accuracy. For each run,

$$A_{\text{avg}} = \frac{1}{|\mathcal{I}(q_f)|} \sum_{i \in \mathcal{I}(q_f)} a_{q_f,i}.$$

This is a final-checkpoint task average, not an average over training time.

Forgetting. Let $\mathcal{Q}_i^-$ contain the recorded checkpoints for task i from b_i up to, but excluding, q_f, and let $\mathcal{E} = \{i \in \mathcal{I}(q_f) : \mathcal{Q}_i^- \neq \varnothing\}$. For nonempty $\mathcal{E}$,

$$F = \frac{1}{|\mathcal{E}|} \sum_{i \in \mathcal{E}} \left[\max_{q \in \mathcal{Q}_i^-} a_{q,i} - a_{q_f,i} \right].$$

The historical maximum is taken within each run before averaging across runs. Negative task-level values are retained. Tasks without an earlier recorded checkpoint do not contribute; if $\mathcal{E}$ is empty, forgetting is undefined. Low forgetting is interpreted jointly with final accuracy because weak initial acquisition can also produce a small deterioration.

Backward transfer. For the same eligible tasks,

$$\text{BWT} = \frac{1}{|\mathcal{E}|} \sum_{i \in \mathcal{E}} (a_{q_f,i} - a_{b_i,i}).$$

With $m_i = \max_{q \in \mathcal{Q}_i^-} a_{q,i}$, the identity $F + \text{BWT} = |\mathcal{E}|^{-1} \sum_i (m_i - a_{b_i,i}) \geq 0$ shows why BWT is not generally $-F$. Equality holds only when the introduction checkpoint attains the historical maximum for every eligible task.

Average anytime accuracy. Let $\mathcal{R}$ be the set of recorded communication rounds and let $q(r)$ identify the associated checkpoint. Define

$$A_r = \frac{1}{|\mathcal{I}(q(r))|} \sum_{i \in \mathcal{I}(q(r))} a_{q(r),i}, \qquad \text{AAA} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} A_r.$$

This average depends on the actual recorded schedule; missing checkpoints are not interpolated. Each metric is formed separately within a run and then summarized across runs. The reported tables contain A_{avg} and F. BWT and AAA are defined here to distinguish them from final accuracy and forgetting, but no numerical BWT or AAA values are reported.

Communication and server time. Communication accounting includes classifier uploads and downloads, generator-version transfers, class-coverage information, and feature buffers. No byte totals are reported. A whole-run server-time decomposition would use disjoint intervals,

$$t_{\text{run}} = t_{\text{ordinary}} + t_{\text{evaluation}} + \sum_{b \in \mathcal{B}} (t_{\text{generator},b} + t_{\text{synthesis},b} + t_{\text{classifier},b}),$$

where $\mathcal{B}$ is the set of consolidation events. Tables 6 and 18 report one task-boundary profile rather than this whole-run quantity.

G Additional Experimental Results

The following tables extend the main comparisons to additional methods and allocation settings. They report final average accuracy and forgetting as defined in Appendix F. Unless explicitly labeled as relative percentages, differences between percentage-valued metrics are stated in percentage points.

G.1 CIFAR-10/100 Full Results

Table 9: CIFAR-10 results for five IID class-incremental tasks (%).

Method	Type	$A_{avg} \uparrow$	$F \downarrow$
AF-FCL	FCL	83.06 ± 0.34	19.10 ± 0.31
FedWeIT	FCL	80.84 ± 0.35	23.20 ± 0.34
GLFC	FCL	86.80 ± 0.30	13.70 ± 0.26
FedCIL	FCL	87.86 ± 0.28	12.20 ± 0.24
TARGET	FCL	88.98 ± 0.26	10.30 ± 0.22
MFCL	FCL	86.02 ± 0.31	15.00 ± 0.27
FedRUA	**FCL**	90.40 ± 0.21	7.60 ± 0.18

On CIFAR-10, FedRUA is 1.42 accuracy points above TARGET and has 2.70 points lower forgetting. The joint direction agrees with the larger-class experiments, but DGR-FCL and CAN were not evaluated in this table, so it establishes no ranking against them on CIFAR-10.

Table 10: CIFAR-100 comparison for five synchronous IID class-incremental tasks (%).

Method	Type	$A_{avg} \uparrow$	$F \downarrow$
FedAvg	FL	36.16 ± 0.63	36.60 ± 0.55
SCAFFOLD	FL	41.10 ± 0.55	29.80 ± 0.47
FedDyn	FL	43.25 ± 0.52	27.40 ± 0.45
FedProx+Replay	FL+CL	45.80 ± 0.50	24.50 ± 0.38
FedDyn+EWC	FL+CL	52.10 ± 0.48	18.20 ± 0.35
AF-FCL	FCL	51.30 ± 0.45	16.40 ± 0.29
FedWeIT	FCL	49.54 ± 0.46	19.90 ± 0.32
GLFC	FCL	53.98 ± 0.42	12.70 ± 0.25
FedCIL	FCL	55.36 ± 0.40	10.60 ± 0.23
TARGET	FCL	57.12 ± 0.38	7.70 ± 0.20
MFCL	FCL	53.50 ± 0.42	13.20 ± 0.26
CAN	FCL	56.20 ± 0.50	12.50 ± 0.60
DGR-FCL	FCL	57.50 ± 0.35	6.80 ± 0.20
FedRUA	**FCL**	58.38 ± 0.32	5.10 ± 0.15

The expanded CIFAR-100 table shows that conventional FL without a retention mechanism has much lower final accuracy and higher forgetting in this sequence. Among listed FCL comparators, DGR-FCL has the highest final accuracy and lowest forgetting before FedRUA; TARGET is close in accuracy but has higher forgetting. These rows compare complete systems with different replay representations and resource demands, not isolated generator choices.

G.2 IMAGENET-1K FULL RESULTS

Table 11: ImageNet-1K comparison for five synchronous IID tasks at 64×64 resolution (%).

Method	Type	$A_{avg} \uparrow$	$F \downarrow$
FedAvg	FL	25.66 ± 0.47	31.90 ± 0.48
SCAFFOLD	FL	30.15 ± 0.42	26.10 ± 0.43
FedDyn	FL	31.80 ± 0.40	24.20 ± 0.42
FedProx+Replay	FL+CL	33.50 ± 0.39	20.80 ± 0.36
FedDyn+EWC	FL+CL	38.20 ± 0.36	15.80 ± 0.32
AF-FCL	FCL	39.30 ± 0.35	13.70 ± 0.28
FedWeIT	FCL	37.50 ± 0.36	16.80 ± 0.30
GLFC	FCL	41.48 ± 0.33	10.70 ± 0.24
FedCIL	FCL	42.72 ± 0.32	8.80 ± 0.22
TARGET	FCL	43.72 ± 0.30	7.00 ± 0.20
MFCL	FCL	41.78 ± 0.33	10.20 ± 0.24
CAN	FCL	43.80 ± 0.40	12.00 ± 0.50
DGR-FCL	FCL	44.00 ± 0.30	6.90 ± 0.18
FedRUA	**FCL**	44.30 ± 0.25	6.20 ± 0.17

The expanded ImageNet-1K table preserves the ordering in the main comparison: FedRUA exceeds DGR-FCL by 0.30 accuracy points and has 0.70 points less forgetting. The accuracy margin is smaller than on CIFAR-100. Separate method-level standard deviations do not determine uncertainty in the paired difference, and aggregate classifier outcomes do not identify whether synthesis fidelity, class coverage, or another factor accounts for the narrower margin.

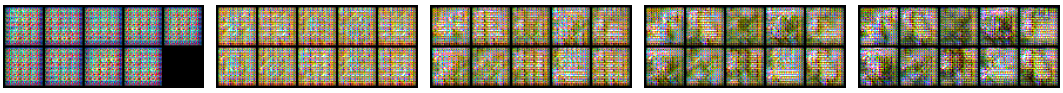
G.3 SPATIAL NON-IID ALLOCATION

Table 12: CIFAR-100 at Dirichlet $\alpha = 0.5$ (%). Relative change is $100(A_{\alpha=0.5} - A_{\mathrm{IID}})/A_{\mathrm{IID}}$.

Method	$A_{avg} \uparrow$	$F \downarrow$	Rel. change (%)
FedAvg	32.10 ± 0.82	39.20 ± 0.58	-11.2
SCAFFOLD	36.20 ± 0.68	32.80 ± 0.48	-11.9
FedDyn	38.50 ± 0.62	29.40 ± 0.45	-11.0
FedDyn+EWC	47.80 ± 0.52	20.40 ± 0.38	-8.3
AF-FCL	38.90 ± 0.65	22.50 ± 0.42	-24.2
GLFC	44.80 ± 0.55	18.20 ± 0.35	-17.0
FedCIL	47.55 ± 0.52	15.80 ± 0.32	-14.1
TARGET	51.20 ± 0.48	11.50 ± 0.28	-10.4
CAN	52.80 ± 0.50	12.20 ± 0.40	-6.0
DGR-FCL	55.80 ± 0.45	8.20 ± 0.30	-3.0
FedRUA	57.25 ± 0.38	6.80 ± 0.20	-1.9

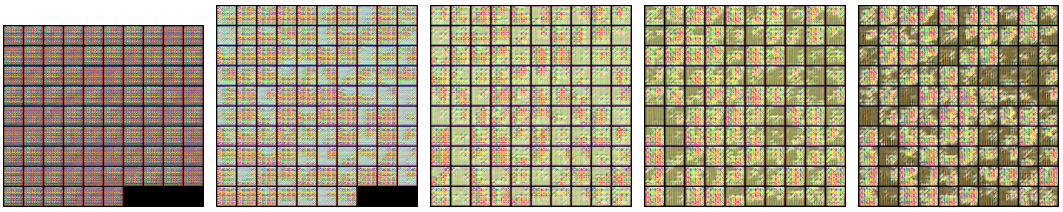
The full $\alpha = 0.5$ table adds methods absent from the main display. In accuracy, FedRUA's 1.9% relative decrease from IID is smaller than the listed DGR-FCL (3.0%), CAN (6.0%), and TARGET (10.4%) decreases; these are relative changes, not percentage-point differences. Forgetting follows a different pattern: FedRUA's absolute forgetting advantage over DGR-FCL narrows from 1.70 to 1.40 points. Thus one should not infer a uniformly smaller heterogeneity effect from the accuracy column alone.

G.4 Generated Samples



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4 (e) Task 5

Figure 2: Qualitative CIFAR-10 generator samples grouped by task-labeled stage. The panels illustrate sample appearance and do not track fixed class–latent pairs across checkpoints or measure retention.



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4 (e) Task 5

Figure 3: Qualitative CIFAR-100 generator samples grouped by task-labeled stage. The panels illustrate sample appearance and do not track fixed class–latent pairs across checkpoints or measure retention.

The panels show the appearance of samples at task-labeled stages, but they do not track identical old-class labels and latent inputs across checkpoints. Texture or color changes may reflect the sampled class mixture rather than forgetting within a class. Semantic fidelity, class coverage, or absence of collapse cannot be established from these thumbnails; downstream accuracy and forgetting are evaluated separately.

G.5 Sensitivity to Spatial Heterogeneity

Table 13 varies Dirichlet concentration from $\alpha = 1.0$ to 0.1. From IID to $\alpha = 0.1$, FedRUA's accuracy decreases by 3.78 points, compared with 10.32 points for TARGET and 8.35 points for MFCL. FedRUA has the highest mean accuracy among the methods included in this sweep. This pattern characterizes accuracy under unequal spatial allocations, but does not isolate the singleton fallback: current class imbalance does not determine how many cached experts have encountered a label over their full real-data histories.

Table 13: CIFAR-100 accuracy under varying spatial heterogeneity α ($T = 5$); mean $\pm$ SD.

Method	**Type**	$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 1.0$	**IID**
FedAvg	FL	28.42 ± 0.95	32.10 ± 0.82	35.20 ± 0.68	36.16 ± 0.63
FedProx	FL	30.15 ± 0.88	33.45 ± 0.75	35.80 ± 0.65	38.70 ± 0.60
FedNova	FL	31.40 ± 0.82	34.70 ± 0.70	37.95 ± 0.62	40.62 ± 0.58
AF-FCL	FCL	34.50 ± 0.78	38.90 ± 0.65	42.10 ± 0.52	51.30 ± 0.45
FedWeIT	FCL	38.20 ± 0.72	42.40 ± 0.60	46.80 ± 0.55	49.54 ± 0.46
GLFC	FCL	40.55 ± 0.68	44.80 ± 0.55	49.15 ± 0.48	53.98 ± 0.42
FedCIL	FCL	43.20 ± 0.60	47.55 ± 0.52	52.40 ± 0.45	55.36 ± 0.40
TARGET	FCL	46.80 ± 0.62	51.20 ± 0.48	54.30 ± 0.45	57.12 ± 0.38
MFCL	FCL	45.15 ± 0.65	49.40 ± 0.50	53.25 ± 0.42	53.50 ± 0.42
FedRUA	**FCL**	54.60 ± 0.42	57.25 ± 0.38	58.10 ± 0.35	58.38 ± 0.32

H Further Experimental Analysis

This section gives the operating conditions and sensitivity results behind Sections 5.5–5.9.

H.1 Scalability and Partial Participation

Table 7 varies the fully participating client population, whereas Table 14 varies both total and active client counts. Participation rate is distinct from usable expertise for a label: real-data coverage and filtering determine membership in K_y, while cache age affects the freshness of those experts. The stated policy does not exclude an entry solely because it is old.

Table 14: FedRUA under partial participation on CIFAR-100 (five tasks; %).

Total K	**Active/round**	**Participation**	$A_{avg} \uparrow$	$F \downarrow$
200	10	5%	54.63 ± 0.48	8.67 ± 0.31
200	20	10%	55.42 ± 0.44	7.82 ± 0.27
500	50	10%	54.91 ± 0.50	8.20 ± 0.30

An absent client's most recent uploaded snapshot remains available and is replaced on its next participation. This preserves access to previously covered labels while retaining potentially outdated predictions and BN buffers. The within-method comparison does not isolate the benefit of this policy from changes in active-client count, local data, or update frequency.

Table 15: Single recorded server wall-clock profile per task boundary (CIFAR-100, 50 clients); phase times are minutes and totals are hours. Section 5.6 summarizes this profile.

Method	Consolidation (min)	Synthesis (min)	Tuning (min)	Total (h)
TARGET	210	10	68	4.80
DGR-FCL	140	180	70	6.50
FedRUA	**120**	**5**	**46**	**2.85**

The phase totals are $288/60 = 4.80$ hours for TARGET, $390/60 = 6.50$ hours for DGR-FCL, and $171/60 = 2.85$ hours for FedRUA. Generator consolidation includes frozen-expert forwards and input-gradient backpropagation. The mixed-batch statistic term also processes the full batch through every expert in the batch's eligible union, so its cost does not decrease in direct proportion to per-label eligibility. These measurements cover the displayed task-boundary phases and exclude ordinary-round training and evaluation. They provide a workload profile rather than a whole-run speedup estimate.

H.2 Generalization Across Classifier Backbones and Pretraining

Table 16 compares FedRUA and TARGET within each fixed backbone/initialization pair. Comparisons across pairs change representation or initialization; comparisons within a pair concern the complete replay methods under that representation. These two comparisons answer different questions.

Table 16: CNN backbone and pretraining results on CIFAR-100 (five synchronous IID tasks; %).

Backbone	**Init.**	**Method**	$A_{avg} \uparrow$	$F \downarrow$
ResNet-18	Scratch	FedRUA	58.38 ± 0.32	5.10 ± 0.15
		TARGET	57.12 ± 0.38	7.70 ± 0.20
ResNet-18	ImageNet-pretrained	FedRUA	62.14 ± 0.29	4.18 ± 0.17
		TARGET	60.78 ± 0.34	6.36 ± 0.24
MobileNetV2	Scratch	FedRUA	56.87 ± 0.41	6.14 ± 0.25
		TARGET	55.46 ± 0.46	8.38 ± 0.31
MobileNetV2	ImageNet-pretrained	FedRUA	60.72 ± 0.35	4.91 ± 0.20
		TARGET	59.18 ± 0.39	6.92 ± 0.26

Pretraining improves ResNet-18 accuracy for both methods, by 3.76 points for FedRUA and 3.66 points for TARGET. FedRUA's within-pair advantage persists for both tested backbones and initialization settings. These results demonstrate consistency across the listed CNN configurations; both

backbones retain the BN statistic operator used by FedRUA, so they do not establish transfer to architectures without that operator.

H.3 Sensitivity Analysis

Server-side synthetic samples per class. Table 17 shows diminishing gains in final accuracy and forgetting as replay count increases. The sweep characterizes downstream classifier performance. Since the classifier epoch count is fixed, replay-set size can change both sample diversity and optimizer exposure, so the comparison does not isolate either effect or establish image fidelity.

Table 17: FedRUA server replay-count sweep on CIFAR-100 (five tasks; %); mean ± SD.

Samples/class	$A_{avg} \uparrow$	$F \downarrow$
10	54.20 ± 0.62	11.50 ± 0.85
20	55.60 ± 0.45	9.80 ± 0.70
50	56.80 ± 0.48	8.20 ± 0.60
100	57.90 ± 0.36	6.50 ± 0.42
200	58.38 ± 0.32	5.10 ± 0.15
500	58.60 ± 0.30	4.80 ± 0.12

Generator architecture. At 200 samples per class, the Light CNN is 1.18 accuracy points below the default Advanced CNN. The ResNet generator is 0.47 points above it and has 0.40 points less forgetting. Thus, Advanced CNN is the reference architecture for the other studies, while ResNet gives the best reported means in this sweep. These results describe classifier utility rather than a measured quality–cost optimum.

Table 18: Generator-architecture sensitivity on CIFAR-100 (200 samples/class; %).

Architecture	$A_{avg} \uparrow$	$F \downarrow$
MLP	54.90 ± 0.70	9.80 ± 0.95
Light CNN	57.20 ± 0.45	6.80 ± 0.40
Advanced CNN	58.38 ± 0.32	5.10 ± 0.15
ResNet	58.85 ± 0.28	4.70 ± 0.12

Client-side replay ratio. At five tasks, the 1:1 ratio has the highest mean accuracy and lowest mean forgetting among the tested settings. Under the separately averaged objective in Equation 2, the sample ratio controls exposure and estimator variability without changing λ_{rep}. A concatenated-batch mean would instead give source weights proportional to sample counts. Accordingly, Table 20 is presented as a sampling-ratio study rather than an isolated sweep of loss coefficients.

Table 19: Client real-to-synthetic replay-ratio sensitivity (CIFAR-100, $T = 5$).

Ratio (Real:Syn)	A_{avg} **(%)** $\uparrow$	F **(%)** $\downarrow$
0.25 (1:4)	56.10 ± 0.55	8.20 ± 0.60
0.5 (1:2)	57.40 ± 0.42	6.50 ± 0.35
1.0 (1:1)	$\mathbf{58.38 \pm 0.32}$	$\mathbf{5.10 \pm 0.15}$
2.0 (2:1)	57.80 ± 0.38	6.90 ± 0.40
4.0 (4:1)	56.90 ± 0.45	8.80 ± 0.55

Replay ratio across task horizons. Table 20 extends the within-method ratio comparison to ten and twenty tasks. It examines how the accuracy–forgetting tradeoff changes across the reported schedules, without comparing FedRUA against other methods at those longer horizons.

Table 20: CIFAR-100 client replay-ratio sensitivity at ten and twenty tasks (%); mean ± SD. Rows compare FedRUA configurations within each task horizon.

Tasks (T)	**Ratio (Real:Syn)**	A_{avg} **(%)** ↑	F **(%)** ↓
10	1:2	32.12 ± 0.61	4.34 ± 0.52
	1:1	32.94 ± 0.57	2.73 ± 0.39
	2:1	30.82 ± 0.65	3.69 ± 0.46
20	1:2	24.34 ± 0.72	13.19 ± 0.81
	1:1	24.23 ± 0.69	8.96 ± 0.67
	2:1	20.00 ± 0.83	8.67 ± 0.73

At ten tasks, 1:1 has both the highest mean accuracy and the lowest mean forgetting. At twenty tasks, no ratio attains both extrema. Compared with 1:2, the 1:1 setting reduces forgetting by 4.23 points at a cost of 0.11 accuracy points; compared with 2:1, it improves accuracy by 4.23 points with 0.29 points more forgetting. The corresponding 1:1 accuracies are 58.38%, 32.94%, and 24.23% at five, ten, and twenty tasks. These schedule-dependent outcomes show why a low forgetting value, such as 2.73% at ten tasks, cannot by itself establish stronger learning. The ratio is an operating tradeoff rather than a universal optimum.

H.4 Trust Score and Corrupted-Update Sensitivity

Table 21: CIFAR-100 corrupted-update stress test. Reject is the mean percentage of updates hard-rejected before consolidation; the rows do not isolate continuous attenuation.

	20% corrupted			40% corrupted		
τ_s	A_{avg} ↑	F ↓	Reject (%)	A_{avg} ↑	F ↓	Reject (%)
1.2	57.92 ± 0.48	5.95 ± 0.32	28.3	57.05 ± 0.58	6.80 ± 0.38	45.6
1.5	57.80 ± 0.45	6.20 ± 0.35	21.7	56.90 ± 0.55	7.10 ± 0.40	42.1
1.8	57.52 ± 0.52	6.55 ± 0.42	16.4	56.45 ± 0.62	7.65 ± 0.45	37.8
2.0	57.20 ± 0.55	6.90 ± 0.48	12.1	55.98 ± 0.68	8.20 ± 0.52	33.5
∞	53.20 ± 0.80	12.40 ± 0.90	0.0	49.50 ± 1.10	18.30 ± 1.20	0.0

Equation 5 uses the parameter-update norm to attenuate a singleton expert's contribution. The recorded defaults are $\alpha_s = 1$ and $\tau_s = 1.5$. Large deviations can arise from benign heterogeneity or local optimization, and small deviations do not imply accurate class predictions. The factor is therefore a heuristic, not a competence or corruption certificate.

The stress test in Table 21 injects Gaussian weight noise with $\sigma = 0.1$ into 20% or 40% of $K = 20$ clients and applies optional hard rejection before consolidation. Hard rejection changes usable expert membership; continuous s_k changes a singleton's weight; the feature gate controls the additional adversarial replay loss. These operations have different effects. Varying their shared update threshold changes rejection and can also change singleton attenuation, so the sweep does not isolate s_k.

Among the finite thresholds, $\tau_s = 1.2$ gives the highest mean accuracy and lowest mean forgetting in both corruption settings, with higher rejection rates than 1.5. Rejection can discard useful as well as perturbed expertise. If ρ_{rej} and ρ_{corr} denote fractions of the same update population, at least $\max(0, \rho_{\text{rej}} - \rho_{\text{corr}})$ of all updates are both clean and rejected. For $\rho_{\text{corr}} < 1$, the corresponding lower bound as a fraction of clean updates is $\max(0, \rho_{\text{rej}} - \rho_{\text{corr}})/(1 - \rho_{\text{corr}})$. These conditional counting bounds are distinct from a measured clean-client false-rejection rate. Gaussian weight noise also does not represent adaptive poisoning or collusion.

I Detailed Limitations

Supervision failure. Shared expert errors can receive high multi-expert weight, while singleton entropy, feature mismatch, and update deviation can suppress the replay contribution of a covered class. The branches are not calibrated to a common scale. Rejection of the last usable expert also removes that label from the replay support. Since marginal BN buffers summarize mixtures of training inputs, a valid class-specific batch can fail the gate; admission should therefore be understood as the application of a statistic-based rule rather than semantic validation.

Staleness and scale. Retaining absent clients' latest uploads preserves access to historical coverage but allows predictions and buffers to age. At fixed total data, increasing the client population also reduces examples per expert and changes local optimizer exposure. Generator training requires expert input gradients and mixed-batch statistic passes, while clients download, store, and execute the generator. The single-boundary timing profile does not capture the complete training and client-side workload.

Scope and trusted server. The server queries individual models and feature buffers. Applying ordinary secure aggregation would change this access pattern, and no formal privacy guarantee is provided. The reported evidence covers synchronized stage progression and BN-based CNNs; the noise study does not establish resistance to adaptive poisoning, collusion, or a compromised server. The diagnostic inequality in Appendix D decomposes distribution discrepancy but does not turn the reliability proxies into a performance guarantee.