FEDDATALEAK: DETECTING DATA LEAKAGE FROM FEDERATED LLM UPDATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated fine-tuning allows large language models (LLMs) to learn from decentralized, sensitive data without sharing raw text, yet the exchanged model updates can still reveal private information. We present **FedDataLeak**, a model-agnostic framework for auditing leakage from federated LLM updates using model weights before and after each communication round. Its primary signal, *Feature Reconstruction Fidelity* (FRF), measures how well a plausible input can reproduce an observed update. To avoid costly reconstruction at every round, FedDataLeak combines FRF with two lightweight representation-shift signals, neuron polysemanticity and activation sparsity, and triggers deeper analysis only for suspicious updates. We evaluate the framework across heterogeneous federated settings, including controlled client partitions, synthetic canaries, and aggregate-only visibility, and test whether its scores identify leakage-prone updates and align with downstream reconstruction and membership evidence. Across diverse models and datasets, FedDataLeak achieves strong leakage detection with audit overhead lower than competitive baselines, demonstrating that federated updates can serve as actionable privacy signals for safe LLM training.

1 INTRODUCTION

Federated learning (FL) enables multiple organizations or devices to collaboratively fine-tune a shared model without centralizing raw training data McMahan et al. (2017). This paradigm is increasingly attractive for adapting large language models (LLMs) to privacy-sensitive domains (e.g., personal communications, enterprise text, clinical notes), where data locality and governance constraints prohibit direct data sharing. Despite this motivation, a growing body of evidence shows that the *updates* exchanged in FL (gradients or weight deltas) can still leak substantial information about client training examples Zhu et al. (2019). For example, a client update produced from a short local sequence containing a rare identifier, medical phrase, or email snippet may leave a distinctive signal in embeddings or adapter weights, even though the raw text itself is never shared. Beyond reconstruction, inference-style privacy risks, such as membership inference, remain a concern when models or updates are exposed Shokri et al. (2017).

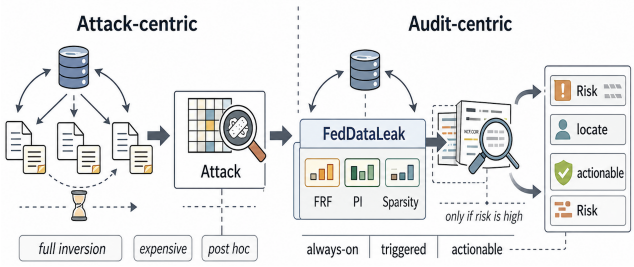


Figure 1: Attack- vs. audit-centric federated leakage analysis.

These results highlight a practical gap between attack demonstrations and deployment needs. In this paper, we call a method audit-centric if it produces a low-cost, round-level risk signal that can be monitored continuously and used to trigger mitigation before running expensive reconstruction. This is a different objective from maximizing reconstruction quality: the goal is not to recover every private example, but to identify which observed updates are most likely to be leakage-prone. In real FL systems, stakeholders need to answer an operational question: *given the observed federated LLM updates in a particular training run, how much private information is currently recoverable, and which rounds or clients pose the highest risk?* Existing methods primarily provide powerful but expensive reconstruction pipelines or worst-case evaluations; they are not designed as *continuous, low-latency audit signals* that can be monitored round-by-round, localized to particular updates, and used to trigger mitigations (e.g., clipping, noise, parameter freezing, changes in local training) before leakage is exploited. This challenge is amplified by modern fine-tuning practice: LLM adaptation

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.

is often performed with parameter-efficient fine-tuning (PEFT) methods (e.g., adapters/LoRA Hu et al. (2022)), where private-task signal is concentrated into a small trainable subspace instead of being diluted across all model parameters. Such concentration may make privacy auditing more module-dependent, motivating module-aware leakage analysis Guo et al. (2025); Vu et al. (2024); Ran et al. (2025); Li et al. (2024). Additionally, secure aggregation reduces visibility into individual updates, shifting auditing from client-level attribution to aggregate-level risk assessment Bonawitz et al. (2017); Ngo et al. (2024). Figure 1 highlights the core shift of FedDataLeak from expensive attack-only leakage analysis to a practical privacy sensor that monitors federated updates.

We propose **FedDataLeak**, a model-agnostic framework for auditing data leakage from federated LLM updates in a white-box setting with access to pre- and post-round model weights and, when available, individual client updates. FedDataLeak builds on the idea that an update is more leakage-prone when it can be explained by a plausible training input. We capture this through *Feature Reconstruction Fidelity (FRF)*, which measures how closely a synthetic candidate input can reproduce an observed update, following the principle underlying gradient inversion attacks Gao et al. (2025); Geiping et al. (2020); Zhu et al. (2019).

Running reconstruction on every update, however, is computationally expensive. FedDataLeak therefore uses activation sparsity and a polysemanticity proxy Elhage et al. (2022) as lightweight representation-shift signals on a public reference set. These signals provide an inexpensive first-stage triage, while FRF and deeper reconstruction are reserved for suspicious updates. This tiered design is particularly useful for parameter-efficient fine-tuning, where leakage may concentrate in embeddings or LoRA adapters Sun et al. (2024), enabling continuous risk scoring without requiring full inversion at every round.

Our contributions are:

❶ **Continuous leakage auditing for federated LLM updates.** We formulate leakage auditing as assigning a risk score to each observed client or aggregate update, enabling round-by-round monitoring under both per-client and aggregate-only observability rather than relying on one-off reconstruction attacks.

❷ **Feature Reconstruction Fidelity (FRF).** We introduce FRF, an update-conditioned reconstructability score that measures how well a plausible synthetic input can reproduce an observed federated update, turning the core mechanism of gradient inversion into a directly testable audit signal Zhu et al. (2019); Geiping et al. (2020).

❸ **Tiered auditing with representation-shift triage.** We combine FRF with activation sparsity and polysemanticity shifts computed on a public reference set to prioritize suspicious updates and invoke expensive reconstruction only when warranted Elhage et al. (2022). This design makes continuous auditing practical while preserving FRF as the primary leakage evidence.

FedDataLeak is an empirical, attack-conditioned auditor of update reconstructability rather than a formal privacy guarantee; under aggregate-only visibility, its output should be interpreted as cohort-level leakage risk rather than client attribution.

2 Related Work

Gradient inversion and data reconstruction. A substantial body of work shows that gradients and parameter updates can leak training data in distributed and federated learning. Deep Leakage from Gradients (DLG) first demonstrated that shared gradients can recover token-level and pixel-level training examples Zhu et al. (2019). Inverting Gradients strengthened this line by showing high-fidelity reconstruction under more realistic optimization settings and by analyzing when averaging fails to provide protection Geiping et al. (2020). More recently, Gao et al. proposed a discrete optimization attack for text, showing that embedding-layer gradients can reveal token identities and enable direct sequence recovery in federated language settings Gao et al. (2025). These works establish that gradient and update leakage is feasible, including in text and federated settings. However, they are primarily attack-centric: they aim to maximize recovery quality under an optimization budget, rather than to provide a low-latency score for continuously monitoring leakage risk at the client level.

Targeted extraction under aggregation. Beyond generic inversion, recent work studies targeted extraction in large-scale federated settings. Panning for Gold shows that an adversary can recover sequences containing privacy-critical phrases even under very large aggregation by maliciously steering the model to filter relevant content Chu et al. (2023). These methods demonstrate residual

privacy risk under aggregation, but they do not solve the audit problem of assigning a generic leakage-risk score to each observed round.

Membership inference and leakage quantification. A complementary line of work studies whether a model reveals if a sample was used in training. Classical membership inference attacks quantify this risk from model behavior Shokri et al. (2017), and recent FL-specific methods such as FedMIA leverage updates from multiple clients and multiple rounds to increase attack power Zhu et al. (2025). Separately, Liu et al. proposed a mutual information-based metric to estimate privacy leakage from federated gradients Liu et al. (2021). These approaches capture important privacy notions Ranjan et al. (2026), but they answer a different operational question from FedDataLeak.

Positioning of FedDataLeak. Rather than introducing a stronger extraction attack, it treats leakage as an *auditing* problem: given an observed federated LLM update, the goal is to produce a low-cost, per-round risk signal that correlates with downstream reconstruction or inference risk. This distinction is particularly relevant when only aggregate updates are visible under secure aggregation Bonawitz et al. (2017). FedDataLeak should be interpreted as an empirical, attack-conditioned auditor for update reconstructability, rather than as a formal privacy estimator or universal leakage detector. The detailed Related Work is provided in Appendix A.

3 METHODOLOGY

FedDataLeak is a tiered auditing pipeline that converts observed federated updates into actionable privacy-risk scores. It first computes low-cost representation-shift indicators on a public reference set, then runs Feature Reconstruction Fidelity (FRF) only for suspicious updates, and finally triggers deeper reconstruction analysis only when the combined score is high. This design supports continuous monitoring under both per-client and aggregate-only observability, while avoiding expensive inversion on every round.

3.1 PROBLEM SETUP AND ASSUMPTIONS

We study federated fine-tuning of a pretrained language model f_θ with parameters $\theta \in \mathbb{R}^d$ under a standard synchronous server-client protocol such as FedAvg McMahan et al. (2017). At round t, the server broadcasts global weights θ_t to a subset of clients C_t. Each participating client performs local optimization on its private dataset D_i and returns a communicated parameter update $\Delta_t^{(i)} \in \mathbb{R}^d$, after which the server aggregates

$$\Delta_t = \sum_{i \in C_t} p_i \Delta_t^{(i)}, \qquad \sum_{i \in C_t} p_i = 1,\ p_i \geq 0, \tag{1}$$

and updates the global model by $\theta_{t+1} = \theta_t + \Delta_t$.

Our goal is to audit whether an observed federated update contains a signal that is sufficiently structured to support downstream reconstruction or privacy-sensitive inference. FedDataLeak therefore assigns a leakage-risk score to each observed update. In the per-client regime, this score is attached to an individual client-round update; under aggregate-only visibility, it becomes a cohort-level round

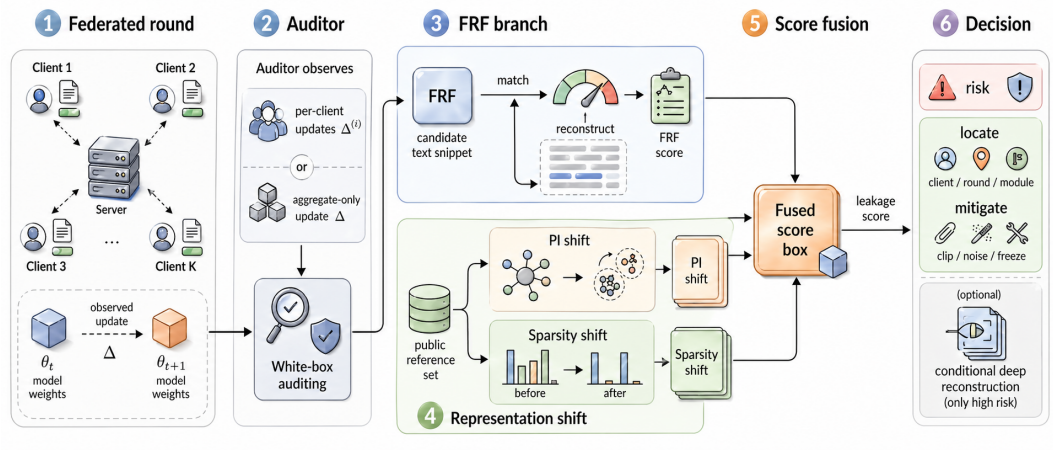


Figure 2: FedDataLeak overview. It transforms federated model updates into actionable leakage scores by combining a primary reconstruction-based signal, FRF, with lightweight representation-shift indicators based on polysemanticity and activation sparsity.

score. FedDataLeak is thus an auditing mechanism for prioritizing inspection and mitigation, not a method for certifying privacy or reconstructing all private data.

Auditing model. FedDataLeak is a white-box auditing framework intended for red-team evaluation, internal privacy review, or continuous server-side monitoring in settings where model states before and after each round are available. We assume the auditor has access to: (i) the pre-round and post-round model weights, hence the observed update; (ii) optionally, individual client updates before aggregation; and (iii) the training objective and approximate client-side optimization recipe. The auditor never accesses raw private client text and must rely only on transmitted updates, model states, public reference data, and the assumed client training recipe.

Observability regimes. We study two regimes. In the *per-client regime*, the auditor observes individual client updates before aggregation and can assign client-level risk scores. In the *aggregate-only regime*, motivated by secure aggregation, the auditor observes only the aggregated round update and can assign only a cohort-level round score. Accordingly, any result in the aggregate-only setting should be interpreted as round-level evidence of possible leakage within the participating cohort, not as proof that a specific client leaked data.

When parameter-efficient fine-tuning (PEFT) is used, auditing is performed only on the communicated trainable subset $A \subseteq \{1, \dots, d\}$, such as embeddings, LoRA parameters, or their union. We write $\Pi_\Lambda(\Delta) \in \mathbb{R}^{|A|}$ for projection onto the audited coordinates. This keeps the audit target aligned with the actual transmitted update rather than with frozen model parameters.

Clients optimize a standard causal language modeling loss, or a task-specific variant, and FedDataLeak uses the same objective in the local-update simulator underlying FRF.

3.2 Feature Reconstruction Fidelity as a Leakage Probe

FedDataLeak uses *Feature Reconstruction Fidelity (FRF)* as its primary leakage signal: the easier it is to find an input whose induced update matches the observed federated update, the more client information appears recoverable from that update. Intuitively, FRF asks whether there exists a plausible synthetic input whose simulated local training update closely matches the observed federated update. This design is motivated by gradient inversion attacks, which reconstruct private inputs by matching gradients or updates Gao et al. (2025); Geiping et al. (2020); Zhu et al. (2019).

Update-matching reconstruction. Let $u_t^{(i)} := \Pi_\Lambda(\Delta_t^{(i)})$ denote the observed client signal on audited parameters (or $u_t := \Pi_\Lambda(\Delta_t)$ for aggregate-only visibility). Here Λ denotes the audited trainable parameter subset and Π_Λ denotes projection onto those coordinates. To test this idea, we simulate the update that would be produced by a candidate input sequence under the assumed client training recipe:

$$\widehat{u}_t(x) \;:=\; \Pi_\Lambda\Big(\mathcal{A}(\theta_t; x)\Big), \tag{2}$$

where $\mathcal{A}$ applies the same local training recipe (loss, optimizer, step count) used on clients, but on synthetic input x. In the simplest (and common in inversion work) approximation, $\mathcal{A}$ can be a single gradient step: $\mathcal{A}(\theta_t; x) \approx -\eta \nabla_\theta \ell_{\mathrm{CLM}}(\theta_t; x)$ Geiping et al. (2020); Zhu et al. (2019). We then search for a candidate input x whose simulated update best matches the observed update, subject to lightweight public constraints such as sequence length or format. In practice, the search may be implemented using continuous relaxations, discrete token search, or hybrid procedures; full solver details are given in Appendix B.2.

FRF score. We define FRF as a normalized fidelity of the best achievable match:

$$\mathrm{FRF}_t \;:=\; \mathrm{clip}_{[0,1]}\left(1 - \frac{\mathcal{L}_{\mathrm{match}}(\hat{x}_t; u_t, \theta_t)}{\|u_t\|_2^2 + \varepsilon}\right), \tag{3}$$

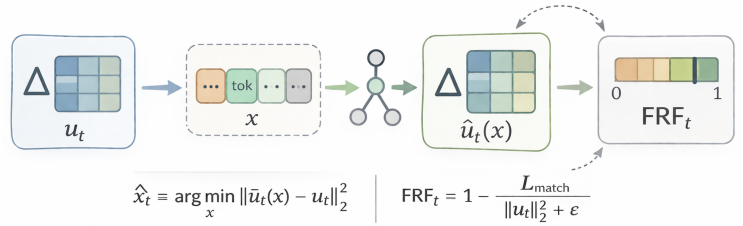


Figure 3: Feature Reconstruction Fidelity (FRF) measures how well a synthetic candidate input can reproduce an observed federated update through update matching.

Algorithm 1 FedDataLeak tiered per-round auditing

Require: Global model θ_t, observed updates $\{\Delta_t^{(i)}\}_{i\in\mathcal{C}_t}$ or aggregate update Δ_t, reference set $\mathcal{R}$, audited parameter subset Λ, thresholds $\tau_{\text{rep}}, \tau_{\text{FRF}}$, weights α, β

Ensure: Leakage risk score (LRS), trigger decision, and optional reconstruction evidence for each observed update

Always-on representation baseline *§3.3*

Compute pre-update statistics PI_t^0, S_t^0 on $(\theta_t, \mathcal{R})$ via (5) and (4).
for each observed update $\Delta \in \{\Delta_t^{(i)}\}_{i\in\mathcal{C}_t}$ **or** Δ_t **do**

Low-cost representation triage *§3.3*

  $u \leftarrow \Pi_\Lambda(\Delta)$
  $\theta' \leftarrow \theta_t + \Delta$
  Compute post-update statistics PI, S on $(\theta', \mathcal{R})$
  Compute shift signals s^{PI}, s^S via (6)
  $q_{\text{rep}} \leftarrow \text{clip}_{[0,1]}\big(\alpha s^S + \beta s^{\text{PI}}\big)$
  if $q_{\text{rep}} < \tau_{\text{rep}}$ **then**

Low-risk output *§3.4*

    Score $\leftarrow q_{\text{rep}}$
    Output $(\text{Score}, \textsc{Monitor}, \varnothing)$
    proceed to the next observed update
  end if
  Solve Eq. (11), or its k-example extension in App D.1, with $k = 1$ by default, obtaining $\widehat{X}^{(k)}$
  Compute $\text{FRF}^{(k)}$ from $(u, \widehat{X}^{(k)})$ via (22)
  $\text{Score} \leftarrow \text{clip}_{[0,1]}\Big(\text{FRF}^{(k)}(1 + \alpha s^S + \beta s^{\text{PI}})\Big)$ via (7)
  if $\text{FRF}^{(k)} < \tau_{\text{FRF}}$ **then**

FRF-level audit output *§3.4*

    Output $(\text{Score}, \textsc{Flag}, \widehat{X}^{(k)})$
    proceed to the next observed update
  end if

Conditional deep inversion *Appendix B.2*

  Run deep reconstruction/inversion under budget B_{inv} using $\widehat{X}^{(k)}$ as initialization
  Compute downstream evidence, e.g., token overlap, canary recovery, or match residual
  Output $(\text{Score}, \textsc{DeepInspect}, \widehat{X}^{(k)}, \text{evidence})$
end for

where $\hat{x}_t$ is the best reconstructed candidate input sequence, $\mathcal{L}_{\text{match}}$ is error between observed and candidate-induced model updates, and $\varepsilon > 0$ avoids division by zero. High FRF_t indicates the update can be explained by a concrete input sequence, consistent with stronger leakage risk. Activation-space calibration metric defined in Eq. 12. As illustrated in Fig. 3, higher agreement between the observed and simulated updates yields a larger FRF score, indicating stronger reconstructability and higher leakage risk (Appendix B.2).

3.3 Representation-shift indicators

FRF directly probes reconstructability, but it can be expensive if run every round. FedDataLeak therefore augments FRF with two inexpensive, update-conditioned *representation-shift indicators* computed on a public reference set $\mathcal{R}$.

Let $H_\ell(\theta; \mathcal{R}) \in \mathbb{R}^{N\times d_\ell}$ denote the matrix of token-level activations collected from layer ℓ across all N reference tokens (e.g., final hidden states). We write $h_n \in \mathbb{R}^{d_\ell}$ for the n-th token activation vector and $H_\ell[:, j]$ for neuron j over tokens.

Activation sparsity. We quantify how concentrated activations are across neurons using Hoyer's sparsity score and the average sparsity at layer ℓ over the reference set $\mathcal{R}$ Hoyer (2004):

$$S(h) \;=\; \frac{\sqrt{d_\ell} - \|h\|_1/(\|h\|_2 + \varepsilon)}{\sqrt{d_\ell} - 1}, \qquad S_\ell(\theta; \mathcal{R}) \;=\; \frac{1}{N}\sum_{n=1}^{N} S(h_n), \tag{4}$$

where $S(h) \in [0, 1]$ and larger values indicate more sparse activations (Appendix B.4). We use Hoyer sparsity because it is scale-invariant and bounded, enabling stable comparison across rounds and modules.

Polysemanticity Index (PI). Motivated by prior work on superposition and feature reuse in neural representations Elhage et al. (2022), we define a simple polysemanticity proxy by clustering the contexts that most strongly activate a neuron. For neuron j, let $\mathcal{I}_j$ be indices of the top-K tokens by activation: $\mathcal{I}_j = \mathrm{TopK}(H_\ell[:, j], K)$. Let $\mathcal{X}_j := \{H_\ell[n, :] : n \in \mathcal{I}_j\} \subset \mathbb{R}^{d_\ell}$ be the corresponding context vectors. We run a clustering procedure $\mathrm{Cluster}(\mathcal{X}_j; K_{\max})$ that selects the number of clusters $k_j \in \{1, \ldots, K_{\max}\}$ (e.g., by maximizing a separation criterion) and define Polysemanticity Index

$$\mathrm{PI}_\ell(\theta; \mathcal{R}) \;:=\; \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} k_j, \tag{5}$$

where $\mathcal{J}$ is a sampled subset of neurons (for efficiency). Intuitively, a decrease in PI suggests that highly activating contexts have become less diverse and therefore more specialized under this proxy, whereas an increase suggests more mixed activation patterns (Appendix B.4).

3.4 Score Fusion and Audit Outputs

At each round t, we compute baseline representation metrics on the *pre-update* model θ_t:

$$\mathrm{PI}_t^0 := \mathrm{PI}_\ell(\theta_t; \mathcal{R}), \qquad S_t^0 := S_\ell(\theta_t; \mathcal{R}).$$

Given an observed update u_t (client-level or aggregate), we compute FRF (Eq. 3) and post-update representation metrics on $\theta_t' := \theta_t + \Delta_t$ (or $\theta_t' := \theta_t + \Delta_t^{(i)}$ for client i):

$$\mathrm{PI}_t := \mathrm{PI}_\ell(\theta_t'; \mathcal{R}), \qquad S_t := S_\ell(\theta_t'; \mathcal{R}).$$

We convert representation shifts into nonnegative signals (as in the prototype implementation):

$$s_t^{\mathrm{PI}} := \max\!\left(0, \frac{\mathrm{PI}_t^0 - \mathrm{PI}_t}{\mathrm{PI}_t^0 + \varepsilon}\right), \qquad s_t^S := \max\!\left(0, \frac{S_t - S_t^0}{S_t^0 + \varepsilon}\right). \tag{6}$$

Finally, the deployed leakage risk score (LRS) follows the tiered policy in Algorithm 1. LRS is a calibrated risk score for ranking and threshold-based triage; it is not a probability of privacy leakage. The auditor first computes $q_{\mathrm{rep,t}} = \mathrm{clip}_{[0,1]}(\alpha s_t^S + \beta s_t^{\mathrm{PI}})$ and runs FRF only when $q_{\mathrm{rep,t}} \geq \tau_{\mathrm{rep}}$:

$$\mathrm{LRS}_t = \begin{cases} q_{\mathrm{rep,t}}, & q_{\mathrm{rep,t}} < \tau_{\mathrm{rep}}, \\ \mathrm{clip}_{[0,1]}\Big(\mathrm{FRF}_t^{(k)}(1 + \alpha s_t^S + \beta s_t^{\mathrm{PI}})\Big), & q_{\mathrm{rep,t}} \geq \tau_{\mathrm{rep}}. \end{cases} \tag{7}$$

Here $\alpha, \beta \geq 0$ and τ_{rep} are calibrated on a validation audit set. Unless otherwise stated, we use $\alpha = \beta = 0.5$, $\tau_{\mathrm{rep}} = 0.35$, and $\tau_{\mathrm{FRF}} = 0.65$, calibrated on a disjoint audit-validation split. By default $k = 1$, while $k > 1$ audits small local candidate sets; details are in Appendix B.5 and Appendix D.1.

4 Experiments

4.1 Experimental Setup

Comprehensive experimental configurations are detailed in Appendix C, while Appendix C.5 describes our observability metrics, evaluation protocols, and leakage labeling.

Models. We evaluate FedDataLeak on three open LLMs spanning size and architecture: LLaMA-3.2-1B Instruct Grattafiori et al. (2024), Mistral 7B Jiang et al. (2023), and Gemma 2–2B Team et al. (2024). Clients perform federated fine-tuning with FedAvg aggregation McMahan et al. (2017). FedDataLeak audits each client update (when visible) or the aggregated update (secure-aggregation setting; audit becomes cohort-level) Bonawitz et al. (2017). We use the same federated protocol and matched auditing pipeline across all three models, varying only model architecture and size.

Datasets and client partitioning. We run federated language fine-tuning on three text corpora with natural client structure: (i) **Medical abstracts**, derived from the Medical-Abstracts corpus with five clinical categories (used as five cross-silo clients) Mujtaba et al. (2019); (ii) **Stack Overflow** next-word prediction, federated *by user* (cross-device), following the widely used LEAF benchmark partitioning

Table 1: Main leakage-detection results across three datasets and LLMs. Values are means over three seeds; standard deviations for FedDataLeak are ±0.010–0.012 (AUPRC), ±0.009–0.010 (AUROC), and ±0.021–0.024 (TPR@1% FPR). Best values are in **bold**.

Dataset	Method	LLaMA-3.2-1B			Mistral-7B			Gemma-2B		
		AUPRC	AUROC	TPR@1%	AUPRC	AUROC	TPR@1%	AUPRC	AUROC	TPR@1%
Medical Abstract	DLG Zhu et al. (2019)	0.30	0.55	0.25	0.28	0.53	0.22	0.25	0.50	0.20
	iDLG Zhao et al. (2020)	0.40	0.65	0.35	0.38	0.60	0.32	0.35	0.55	0.25
	Inverting Geiping et al. (2020)	0.50	0.75	0.45	0.48	0.72	0.42	0.45	0.70	0.35
	Panning Chu et al. (2023)	0.45	0.70	0.40	0.42	0.68	0.37	0.40	0.65	0.30
	H-MINE Liu et al. (2021)	0.50	0.70	0.40	0.48	0.68	0.37	0.45	0.65	0.30
	FET Gao et al. (2025)	0.80	0.82	0.70	0.80	0.78	0.70	0.76	0.75	0.74
	FedMIA Zhu et al. (2025)	0.88	0.89	0.80	0.88	0.89	0.76	0.86	0.88	0.76
	FedDataLeak	**0.90**	**0.93**	**0.85**	**0.88**	**0.92**	**0.80**	**0.87**	**0.91**	**0.78**
Stack Overflow	DLG Zhu et al. (2019)	0.27	0.52	0.22	0.25	0.50	0.20	0.23	0.48	0.18
	iDLG Zhao et al. (2020)	0.37	0.62	0.32	0.35	0.58	0.29	0.33	0.54	0.24
	Inverting Geiping et al. (2020)	0.48	0.73	0.43	0.45	0.70	0.39	0.42	0.67	0.32
	Panning Chu et al. (2023)	0.44	0.69	0.39	0.41	0.66	0.35	0.38	0.63	0.28
	H-MINE Liu et al. (2021)	0.47	0.68	0.38	0.44	0.65	0.34	0.41	0.62	0.27
	FET Gao et al. (2025)	0.77	0.80	0.67	0.74	0.77	0.64	0.70	0.73	0.66
	FedMIA Zhu et al. (2025)	0.86	0.88	0.78	0.84	0.87	0.74	0.81	0.84	0.71
	FedDataLeak	**0.88**	**0.91**	**0.82**	**0.86**	**0.89**	**0.78**	**0.83**	**0.87**	**0.74**
Enron Emails	DLG Zhu et al. (2019)	0.32	0.57	0.27	0.30	0.55	0.24	0.27	0.52	0.22
	iDLG Zhao et al. (2020)	0.43	0.67	0.37	0.40	0.63	0.34	0.37	0.59	0.29
	Inverting Geiping et al. (2020)	0.53	0.77	0.48	0.50	0.74	0.45	0.47	0.71	0.38
	Panning Chu et al. (2023)	0.48	0.72	0.42	0.45	0.69	0.39	0.42	0.66	0.33
	H-MINE Liu et al. (2021)	0.52	0.72	0.42	0.49	0.69	0.38	0.46	0.66	0.32
	FET Gao et al. (2025)	0.82	0.84	0.72	0.80	0.80	0.70	0.77	0.77	0.73
	FedMIA Zhu et al. (2025)	0.89	0.90	0.81	0.88	0.89	0.77	0.85	0.88	0.76
	FedDataLeak	**0.91**	**0.94**	**0.86**	**0.89**	**0.92**	**0.82**	**0.88**	**0.90**	**0.80**

Color guide: classical/MI; extraction; FL privacy; FedDataLeak.

Caldas et al. (2018); (iii) **Enron Emails**, federated *per user/mailbox* Hermans & Murphy-Hill (2015). All pre-processing uses the corresponding model tokenizer; sequences are truncated/padded to a fixed maximum length (Appendix C.2).

Comparing approaches. We compare against established leakage methods spanning optimization-based inversion, targeted extraction, and information-theoretic leakage scoring: **DLG** Zhu et al. (2019), **iDLG** Zhao et al. (2020), **Inverting Gradients** Geiping et al. (2020), **Panning for Gold** Chu et al. (2023), **FET** (discrete text gradient inversion) Gao et al. (2025), and **H-MINE** (mutual-information-based leakage metric) Liu et al. (2021). For inversion-style baselines, we use their reconstruction objective value (or best-achieved match) as the per-update leakage score under a fixed compute budget (Appendix C.4).

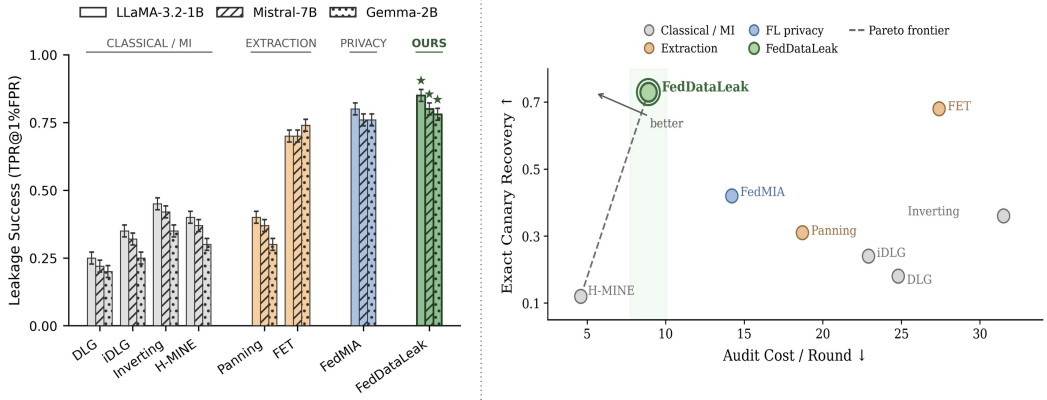
Evaluation metrics. We evaluate FedDataLeak against the paper's three core objectives: *(i)* detect leakage-prone updates, *(ii)* verify that larger audit scores correspond to stronger downstream privacy leakage, and *(iii)* quantify whether the auditor is practical for round-by-round deployment. For Table 1, we use *intervention-defined* leakage labels: an audit instance is positive when a unique synthetic canary is deliberately included in the local training data, with a matched canary-absent instance serving as the negative. Successful extraction is evaluated separately through ECR and Token F1; realized-leakage labels based on an independent fixed-budget reconstruction oracle are reported in Appendix C.5. We use AUPRC as the primary ranking metric because positive leak events are sparse, AUROC as a threshold-free measure of separability, and TPR@1%FPR to measure usefulness in the low-false-alarm regime required by privacy auditing Davis & Goadrich (2006); Carlini et al. (2022). For Table 2, we validate downstream privacy risk using Exact Canary Recovery (ECR), Token F1, and Membership AUC, and we report FRF-cos, FRF@K, and Audit Cost / Round to connect FedDataLeak's scores to token-level recoverability and operational overhead Carlini et al. (2019); Jagielski (2023); Shokri et al. (2017); Gao et al. (2025). For Table 3, we retain AUPRC/AUROC for detection quality, ECR and FRF-cos for reconstruction faithfulness, and Audit Cost/Round for practicality, so that each ablation is interpreted jointly in terms of audit accuracy, privacy evidence, and deployment cost. For each dataset–model setting, we additionally report (N_+, N_-), positive prevalence, reconstruction-oracle budget, recovery threshold, and 95% confidence intervals (Appendix C.6).

4.2 Experimental Results

Per-client or Aggregate-only auditing. Table 1 shows that FedDataLeak consistently matches or improves on the strongest baselines across models and datasets, with particularly strong performance

Table 2: Privacy auditing and practical evidence on Medical Abstract with LLaMA-3.2-1B. Values are means over 3 seeds; "–" denotes metrics not applicable to scalar-only baselines.

Method	Privacy Auditor			Practical Evidence		
	ECR ↑	Tok. F1 ↑	M-AUC ↑	FRF-cos ↑	FRF@K ↑	Cost (s) ↓
DLG	0.18	0.34	0.63	0.41	0.46	24.8
iDLG	0.24	0.40	0.67	0.47	0.53	22.9
Inverting	0.36	0.51	0.72	0.56	0.63	31.5
Panning	0.31	0.48	0.70	0.52	0.60	18.7
H-MINE	–	–	0.74	–	–	**4.6**
FET	0.68	0.74	0.79	0.76	0.84	27.4
FedMIA	–	–	0.86	–	–	14.2
FedDataLeak	**0.73**	**0.79**	**0.89**	**0.80**	**0.88**	8.9



(a) Grouped bar plot: low-FPR detection

(b) Privacy-cost pareto plot

Figure 4: **Low-FPR detection and privacy-cost trade-off.** (a) FedDataLeak achieves the highest leakage detection success at TPR@1%FPR across all three LLMs on Medical Abstract. (b) FedDataLeak lies on the favorable privacy-cost frontier, providing stronger exact canary recovery with substantially lower audit cost than full inversion-style baselines.

in the low-FPR regime. This behavior follows from its design: FRF directly measures update-conditioned reconstructability, while polysemanticity and sparsity shifts provide inexpensive triage before the more costly reconstruction stage. As a result, FedDataLeak identifies leakage-prone rounds without requiring full inversion for every update.

Privacy evidence and practical auditing. Table 2 further shows that higher FedDataLeak scores correspond to stronger privacy evidence, including exact canary recovery, token overlap, membership exposure, and FRF-based token recoverability, while maintaining substantially lower audit cost than full reconstruction baselines. Under aggregate-only visibility, these scores should be interpreted as cohort-level leakage evidence rather than attribution to a specific client.

Scheduler-matched comparison. To isolate the leakage score's contribution from the tiered scheduler, we apply the identical PI/sparsity trigger (including triage overhead) to FET and Inverting Gradients. See Appendix C.4.1 for detailed analysis.

Figure 4 summarizes this trade-off. Panel (a) shows consistently strong low-FPR detection across LLaMA-3.2-1B, Mistral-7B, and Gemma-2B, while panel (b) shows that FedDataLeak achieves a favorable privacy-evidence-cost balance by recovering more canaries at lower per-round cost than inversion-heavy methods. Together, these results support the tiered design: lightweight representation shifts identify suspicious updates, and expensive reconstruction is reserved for cases that warrant deeper inspection (Appendix D.2 and D.3).

4.3 Ablation Study

The ablation study evaluates whether FedDataLeak's gains come primarily from the reconstruction-based FRF signal, from lightweight representation triage, or from their combination.

Table 3 shows that FRF is the primary driver of FedDataLeak's performance, while PI and sparsity provide useful additional triage information when fused with the reconstruction signal. FRF-only already provides a strong update-matching signal, but adding PI and sparsity improves ranking by emphasizing updates that are not only reconstructable but also induce unusually concentrated

Table 3: Ablation study on the Medical Abstract dataset with LLaMA-3.2-1B. We evaluate the contribution of reconstruction fidelity, representation-shift triage, trainable update subsets, and the full FedDataLeak design. Gray rows denote FRF or representation variants, blue rows denote module-level variants, and green denotes the full method. Best values are shown in bold.

Variant	AUPRC ↑	AUROC ↑	ECR ↑	FRF-cos ↑	Audit Cost/Round (s) ↓
FRF-only	0.84	0.88	0.66	0.769	7.1
FRF + PI	0.87	0.90	0.69	0.781	7.8
FRF + Sparsity	0.86	0.89	0.68	0.779	7.6
Weighted FRF (PI + Sparsity)	0.89	0.92	0.71	0.796	8.5
Embeddings-only FedDataLeak	0.88	0.91	0.70	0.790	**6.9**
LoRA-only FedDataLeak	0.85	0.88	0.63	0.752	9.8
Mixed (Embeddings + LoRA)	0.89	0.92	0.72	0.801	10.6
Full FedDataLeak	**0.90**	**0.93**	**0.73**	**0.804**	8.9

representation changes on the public reference set. The weighted fusion variant performs better than FRF-only, indicating that the auxiliary signals contribute useful information beyond reconstruction fidelity alone. Robustness to client-side computation is reported in Appendix D.1.

The module-level ablations are also informative. Auditing embeddings alone is stronger than auditing LoRA alone, consistent with the fact that embedding updates expose more direct token-local information. Mixed auditing is strongest when run unconditionally, but it also incurs higher per-round cost. Importantly, the "Full FedDataLeak" row uses the triggered auditing policy from Algorithm 1, whereas fixed module variants may evaluate more expensive branches on every update; the cost numbers should therefore be interpreted together with the deployment policy.

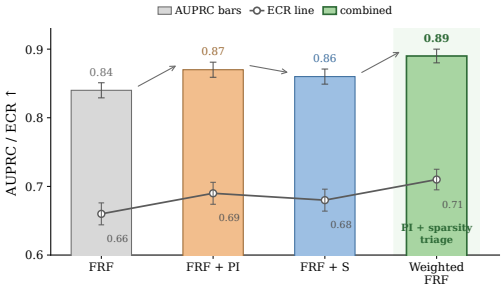


Figure 5: Incremental component ablation.

Figure 5 summarizes the incremental effect of the auxiliary signals: adding PI and sparsity to FRF consistently improves both detection and controlled recovery metrics, while preserving the practical benefit of triggered auditing. Practically, PI and sparsity improve detection and recovery over FRF-only auditing with a modest additional cost, while the tiered design reduces deployment cost relative to baselines that run expensive inversion or deep reconstruction. Detailed results of the ablation study are provided in Appendix D.

4.4 Discussion and Limitations

FedDataLeak is intended as a practical privacy audit signal rather than a formal privacy guarantee. Its main strength is that it converts update reconstructability and representation shifts into actionable round-level risk scores.

Limitations. (i) Adaptive clients may try to evade detection by spreading sensitive signals across parameters, smoothing representation shifts, or mimicking benign update statistics. (ii) Defenses such as clipping and DP-SGD can reduce leakage but also change update geometry, requiring threshold recalibration and potentially lowering model utility. (iii) FRF depends on an approximate client recipe; mismatches in optimizer, learning rate, local steps, or trainable subset can shift absolute scores. Finally, under secure aggregation, FedDataLeak provides only cohort-level round risk, not individual client attribution (Appendix E).

5 Conclusion

We presented FedDataLeak, a white-box auditing framework for detecting leakage-prone federated LLM updates. Its core idea is to use update reconstructability, measured by FRF, as a practical privacy-risk signal, and to combine it with lightweight representation-shift indicators for low-cost triage. Across models, datasets, and observability settings, FedDataLeak identifies risky updates while reducing the need for expensive reconstruction on every round. FedDataLeak is an empirical audit tool rather than a formal privacy guarantee: high scores indicate that an update warrants deeper inspection or mitigation, while low scores do not prove the absence of leakage.

AI USE STATEMENT

Generative AI tools were used only in a limited supporting role, primarily for grammar correction, language polishing, and minor formatting or writing assistance. They were not used to generate the core research ideas, methodology, experimental results, or scientific conclusions. All AI-assisted text was reviewed and revised by the authors, and all technical claims, experiments, and results were independently verified. The authors take full responsibility for the final content of this work, including any material produced with the assistance of generative AI.

ETHICS STATEMENT

FedDataLeak is intended for defensive privacy auditing of federated learning systems rather than for unauthorized data extraction. Our controlled reconstruction experiments use synthetic canaries or approved public data and do not target real private information. Nevertheless, reconstruction and inference techniques studied in this work could potentially be misused to recover sensitive information from model updates. We therefore frame these techniques as auditing mechanisms for identifying leakage-prone updates and supporting appropriate mitigation. The study does not involve human subjects or the collection of new personally identifiable information, and all datasets and models are used subject to their respective licenses and usage requirements.

REPRODUCIBILITY STATEMENT

We provide the complete implementation, experimental datasets and processing utilities, configuration files, and scripts required to reproduce the FedDataLeak experiments. The supplementary material documents the federated training protocol, model and dataset settings, audit metrics, hyperparameters, baseline configurations, ablations, and evaluation procedures. We additionally provide a detailed README with environment setup, dependencies, data preparation, and step-by-step execution instructions for reproducing the main results. Random seeds and relevant experimental configurations are included to facilitate independent replication.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in neural information processing systems*, 34:17455–17466, 2021.

Galen Andrew, Peter Kairouz, Sewoong Oh, Alina Oprea, H Brendan McMahan, and Vinith M Suriyakumar. One-shot empirical privacy estimation for federated learning. *arXiv preprint arXiv:2302.03098*, 2023.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Hong-Min Chu, Jonas Geiping, Liam H Fowl, Micah Goldblum, and Tom Goldstein. Panning for gold in federated learning: Targeted text extraction under arbitrarily large-scale aggregation. In *The Eleventh International Conference on Learning Representations*, 2023.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240, 2006.

Dimitar I Dimitrov, Maximilian Baader, Mark N Müller, and Martin Vechev. Spear: Exact gradient inversion of batches in federated learning. *Advances in Neural Information Processing Systems*, 37:106768–106799, 2024.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Ying Gao, Yuxin Xie, Huanghao Deng, and Zukun Zhu. Gradient inversion attack in federated learning: exposing text data through discrete optimization. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 2582–2591, 2025.

Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in neural information processing systems*, 33:16937–16947, 2020.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Wenkai Guo, Xuefeng Liu, Haolin Wang, Jianwei Niu, Shaojie Tang, and Jing Yuan. Can federated learning safeguard private data in llm training? vulnerabilities, attacks, and defense evaluation. *arXiv preprint arXiv:2509.20680*, 2025.

Felienne Hermans and Emerson Murphy-Hill. Enron's spreadsheets and related emails: A dataset and analysis. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pp. 7–16. IEEE, 2015.

Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(Nov):1457–1469, 2004.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Matthew Jagielski. A note on interpreting canary exposure. *arXiv preprint arXiv:2306.00133*, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Sanjay Kariyappa, Chuan Guo, Kiwan Maeng, Wenjie Xiong, G Edward Suh, Moinuddin K Qureshi, and Hsien-Hsin S Lee. Cocktail party attack: Breaking aggregation-based privacy in federated learning using independent component analysis. In *International Conference on Machine Learning*, pp. 15884–15899. PMLR, 2023.

Shenghui Li, Edith C-H Ngai, Fanghua Ye, and Thiemo Voigt. Peft-as-an-attack! jailbreaking language models during federated parameter-efficient fine-tuning. *arXiv preprint arXiv:2411.19335*, 2024.

Yong Liu, Xinghua Zhu, Jianzong Wang, and Jing Xiao. A quantitative metric for privacy leakage in federated learning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3065–3069. IEEE, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Samuel Maddock, Alexandre Sablayrolles, and Pierre Stock. Canife: Crafting canaries for empirical privacy measurement in federated learning. *arXiv preprint arXiv:2210.02912*, 2022.

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.

Ghulam Mujtaba, Liyana Shuib, Norisma Idris, Wai Lam Hoo, Ram Gopal Raj, Kamran Khowaja, Khairunisa Shaikh, and Henry Friday Nweke. Clinical text classification research trends: Systematic literature review and open issues. *Expert systems with applications*, 116:494–520, 2019.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pp. 739–753. IEEE, 2019.

Khac-Hoang Ngo, Johan Östman, Giuseppe Durisi, and Alexandre Graell i Amat. Secure aggregation is not private against membership inference attacks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 180–198. Springer, 2024.

Delong Ran, Xinlei He, Tianshuo Cong, Anyu Wang, Qi Li, and Xiaoyun Wang. Lora-leak: Membership inference attacks against lora fine-tuned language models. *arXiv preprint arXiv:2507.18302*, 2025.

Ravi Ranjan, Utkarsh Grover, Xiaomin Lin, and Agoritsa Polyzou. G-drift mia: Membership inference via gradient-induced feature drift in llms. *arXiv preprint arXiv:2604.00419*, 2026.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. *arXiv preprint arXiv:2403.12313*, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Minh Vu, Truc Nguyen, My T Thai, et al. Analysis of privacy leakage in federated large language models. In *International Conference on Artificial Intelligence and Statistics*, pp. 1423–1431. PMLR, 2024.

Xiaodong Wang, Naiyu Wang, Longfei Wu, Zhitao Guan, Xiaojiang Du, and Mohsen Guizani. Gbmia: Gradient-based membership inference attack in federated learning. In *ICC 2023-IEEE International Conference on Communications*, pp. 5066–5071. IEEE, 2023.

Hao Zhang, Tingting Wu, Siyao Cheng, and Jie Liu. Cc-fedavg: Computationally customized federated averaging. *IEEE Internet of Things Journal*, 11(3):4826–4841, 2023.

Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Jinfeng Yi. Understanding clipping for federated learning: Convergence and client-level differential privacy. In *International Conference on Machine Learning, ICML 2022*, 2022.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

Gongxi Zhu, Donghao Li, Hanlin Gu, Yuan Yao, Lixin Fan, and Yuxing Han. Fedmia: An effective membership inference attack exploiting" all for one" principle in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20643–20653, 2025.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.

APPENDIX

A DETAILED RELATED WORK

In Table 4, we group prior work into four buckets: *gradient inversion and text inversion*, *targeted extraction under aggregation*, *membership inference and leakage metrics*, and *privacy defenses / privacy auditing*. This partition highlights a central distinction in our paper: prior work largely studies either *how to extract* private information or *how to bound privacy under a defense*, whereas FedDataLeak is designed as a *continuous audit signal* that operates directly on observed federated updates.

Gradient inversion and text inversion. A large body of work shows that shared gradients or model updates can reveal training inputs. DLG and iDLG established early gradient-matching attacks, while Inverting Gradients improved reconstruction quality under more realistic optimization settings Zhu et al. (2019); Zhao et al. (2020); Geiping et al. (2020). More recent attacks strengthen this picture in directions relevant to federated text: Cocktail Party Attack shows that even aggregated gradients can sometimes be separated into constituent private signals Kariyappa et al. (2023); SPEAR demonstrates exact inversion of batches in federated settings Dimitrov et al. (2024); and FET shows discrete token-sequence recovery from embedding-related gradients in language models Gao et al. (2025). These works establish leakage feasibility, but they are fundamentally *attack-centric*: they aim to reconstruct inputs under substantial optimization or search cost, rather than provide a lightweight, round-by-round audit score.

Extraction under aggregation. A second line of work studies *targeted* recovery when client updates are hidden by aggregation. Panning for Gold shows that privacy-critical phrases can be extracted even under very large aggregation by steering the training dynamics toward target sequences

Table 4: Representative related-work buckets and why they do not directly solve continuous federated leakage auditing.

Bucket	Representative works	Why not a continuous auditor
Gradient inversion / text inversion	DLG Zhu et al. (2019), iDLG Zhao et al. (2020), Inverting Gradients Geiping et al. (2020), Cocktail Party Attack Kariyappa et al. (2023), SPEAR Dimitrov et al. (2024), FET Gao et al. (2025)	Strong reconstruction attacks, but typically optimization-heavy, often require detailed update access, and are evaluated by recovery quality rather than low-latency round-level risk scoring.
Extraction under aggregation	Panning for Gold Chu et al. (2023), Cocktail Party Attack Kariyappa et al. (2023)	Show that aggregation does not eliminate leakage, but focus on adversarial extraction rather than always-on monitoring; under secure aggregation they reveal cohort-level risk, not client attribution.
Membership inference & leakage metrics	Membership inference attacks Shokri et al. (2017); Carlini et al. (2022), H-MINE Liu et al. (2021), FedMIA Zhu et al. (2025)	Capture privacy exposure, but target membership or mutual-information style leakage rather than update-conditioned reconstructability, and often rely on multi-round evidence aggregation.
Privacy defenses / privacy auditing	Secure aggregation Bonawitz et al. (2017), DP-SGD Abadi et al. (2016), adaptive clipping Andrew et al. (2021), CANIFE Maddock et al. (2022), one-shot empirical privacy estimation Andrew et al. (2023)	Reduce or estimate privacy loss under specific defenses, but do not provide a general update-level leakage detector for standard federated LLM fine-tuning.

Color guide: inversion, extraction, MI, defenses.

Chu et al. (2023). Together with aggregate-gradient inversion results such as Cocktail Party Attack Kariyappa et al. (2023), this suggests that secure aggregation changes *what is observable* more than it eliminates privacy risk. In particular, when only the aggregate update is visible, client attribution may be lost, but *cohort-level* risk scoring remains possible: one can still ask whether a round aggregate is unusually reconstructable or otherwise consistent with leakage.

Membership inference and leakage metrics. Another bucket studies privacy exposure without explicit reconstruction. Classical membership inference attacks ask whether a sample participated in training Shokri et al. (2017); Carlini et al. (2022), and recent FL-specific methods such as FedMIA leverage evidence across clients and rounds to improve attack power Zhu et al. (2025). Complementarily, H-MINE quantifies leakage via a mutual-information approximation on observed gradients Liu et al. (2021). These methods capture important privacy notions, but they are not designed as update-conditioned reconstructability probes: membership scores and MI estimates need not explain *why* a specific round is leakage-prone, nor are they naturally paired with triggered reconstruction analysis.

Privacy defenses and privacy auditing. A final bucket focuses on mitigation or empirical privacy auditing. Secure aggregation hides individual client updates from the server Bonawitz et al. (2017); DP-SGD and adaptive clipping reduce leakage by clipping and noising updates Abadi et al. (2016); Andrew et al. (2021). Closer to our setting, CANIFE and one-shot empirical privacy estimation use canaries to empirically audit privacy in DP-FL systems Maddock et al. (2022); Andrew et al. (2023). These papers are important precedents for empirical privacy auditing, and their canary-based methodology is not inherently restricted to DP-FL. The key distinction is operational: CANIFE and one-shot empirical privacy estimation are active, system-level tests that require planting canaries into training data to measure empirical privacy exposure, often in order to validate or interpret privacy mechanisms. FedDataLeak instead acts as a passive, always-on monitor for standard federated LLM fine-tuning: given an observed client or aggregate update, it scores whether that update is currently reconstructable or leakage-prone, without requiring pre-planted secrets at deployment time.

Positioning of FedDataLeak. FedDataLeak differs from all four buckets in emphasis. It does not seek a stronger inversion or extraction attack; instead, it turns the core mechanism behind reconstruction into a practical *audit signal*. Relative to inversion and extraction works, it is cheaper and explicitly designed for continuous monitoring. Relative to membership and MI-style metrics, it remains tied to reconstructability. Relative to defenses and DP auditors, it addresses the non-DP setting and supports both per-client and aggregate-only observability by producing either client-level or cohort-level risk scores.

B Detailed Methodology

B.1 Threat Model and Update Semantics

Table 5 makes the FedDataLeak threat model explicit. FedDataLeak is a *white-box auditing* framework: the auditor knows the model parameters before and after each round, the training objective, and client-side hyperparameters, but does not access private client text. The key observability distinction is whether the auditor receives *per-client* updates before aggregation or only the *aggregate* update after secure aggregation. This distinction determines whether the output is a client-level attribution score or only a cohort-level leakage score.

Synchronous protocol. Throughout the paper, we study *synchronous* federated training with FedAvg McMahan et al. (2017), where a subset of clients receives θ_t, performs local optimization, and contributes updates that are aggregated once per communication round. This matches both the main method description and the current experimental implementation Zhang et al. (2023). We do not evaluate asynchronous FL in this work. In principle, FedDataLeak can be extended to asynchronous settings by replacing round-indexed updates with timestamped stale updates and auditing each arrival event, but this introduces additional confounding from update staleness and is beyond the scope of the present paper.

Table 5: Explicit threat model for FedDataLeak.

Category	Auditor access	FedDataLeak interpretation
Knows	Pre-/post-round model weights (θ_t, θ_{t+1}); training objective $\ell(\theta; \cdot)$; local hyperparameters (optimizer, local steps, learning rate, batch size); trainable subset Λ when PEFT is used	Sufficient to simulate update-conditioned probes and compute FRF, PI, and sparsity consistently with the client training recipe.
Does not know	Private client text; exact local data order; optimizer internal states not transmitted by clients; hidden individual updates under secure aggregation	Prevents direct access to sensitive content; audit must rely only on transmitted updates and public reference data.
Observes: Per-client regime	Individual client updates $\{\Delta_t^{(i)}\}_{i \in \mathcal{C}_t}$ before aggregation	Enables client-round scoring and attribution of suspicious updates.
Observes: Aggregate-only regime	Only the aggregate round update $\Delta_t = \sum_{i \in \mathcal{C}_t} p_i \Delta_t^{(i)}$	Enables cohort-level leakage detection, but not client attribution.
Training protocol	Synchronous FedAvg throughout this paper	All experiments assume round-based synchronization; asynchronous FL is left as future work.

What counts as an update. For FedDataLeak, the audited object is the *optimizer-state-free parameter delta*

$$\Delta_t^{(i)} := \theta_{t,\mathrm{local}}^{(i)} - \theta_t, \tag{8}$$

namely the difference between the client model after local training and the broadcast global model at the start of the round. This choice is important for AdamW-style local training Loshchilov & Hutter (2017): optimizer moments remain local and are *not* assumed to be transmitted or observable, so the auditor operates on the communicated parameter change rather than on hidden optimizer state. The aggregate update is then

$$\Delta_t := \sum_{i \in \mathcal{C}_t} p_i \Delta_t^{(i)}. \tag{9}$$

This is the object used in Eq. (1) and in the observability regimes above.

PEFT and LoRA semantics. When parameter-efficient fine-tuning is used, the update is defined on the *trainable subset* only. Let Λ denote the audited trainable coordinates. We write

$$u_t^{(i)} := \Pi_\Lambda(\Delta_t^{(i)}), \qquad u_t := \Pi_\Lambda(\Delta_t), \tag{10}$$

where Π_Λ projects onto the parameters that are actually updated (e.g., embeddings, LoRA adapters, or a mixed subset). Frozen parameters contribute zero by construction. Thus, for LoRA-based training Hu et al. (2022), the transmitted update is interpreted as a sparse structured delta supported only on the adapter coordinates; for embeddings-only training, it is supported only on the embedding matrix. This convention keeps the audit target aligned with the actual communication object and avoids conflating leakage in trainable modules with unchanged parts of the model.

Gradient-like messages. Some prior attack literature is phrased in terms of gradients rather than deltas Nasr et al. (2019); Wang et al. (2023). FedDataLeak treats such messages as *gradient-like observables* only insofar as they can be mapped to a common audited signal on Λ. In the current paper, however, all primary experiments and definitions are written for communicated parameter deltas, which is the most faithful abstraction for synchronous FedAvg with AdamW and PEFT.

B.2 Feature Reconstruction Fidelity: Text Solver and Audit Semantics

$$\widehat{x}_t \in \arg\min_{x \in \mathcal{X}} \mathcal{L}_{\mathrm{match}}(x; u_t, \theta_t) + \lambda \mathcal{R}(x), \tag{11}$$

This appendix details how the FRF objective in Eq. (11) is solved for text, what prior constraints are used, how the score is normalized, and how FRF is instantiated for embedding-only, LoRA-only, and mixed PEFT updates. FRF is not intended to be a new strongest reconstruction attack; it is an audit statistic that asks whether an observed update can be explained by at least one plausible text input under a fixed and reported reconstruction budget.

Activation-space calibration metric (controlled evaluation only). When ground-truth inputs are known (e.g., in controlled audits with planted canaries), we also report FRF in activation space: for an activation extractor $\phi_\ell(\theta; x) \in \mathbb{R}^{d_\ell}$ at layer ℓ,

$$\mathrm{FRF}^{\mathrm{feat}}(x, \hat{x}) \;:=\; 1 - \frac{\mathrm{MSE}\big(\phi_\ell(\theta; x),\, \phi_\ell(\theta; \hat{x})\big)}{\mathrm{Var}\big(\phi_\ell(\theta; x)\big) + \varepsilon}, \tag{12}$$

where $\mathrm{MSE}(a, b) = \|a - b\|_2^2 / d_\ell$ and $\mathrm{Var}(a)$ is the empirical variance across coordinates. This quantity is *not* required for deployment, but provides a model-agnostic calibration target in experiments.

Budget notation. B_{FRF} denotes the fixed budget of the triggered FRF solver, including the length grid, candidate vocabulary size, restarts, relaxation steps, discrete refinement steps, and beam width. B_{inv} denotes the optional deep-inversion budget used only when $\mathrm{FRF} \geq \tau_{\mathrm{FRF}}$, covering additional inversion restarts, gradient-matching iterations, candidate refinements, and forward/backward passes for downstream reconstruction evidence. The score update in Algorithm 1 follows the fusion rule in Eq. (7).

Audited update subset. For each observed update, FedDataLeak first projects the communicated parameter delta onto the audited trainable subset

$$u_t^{\Lambda} = \Pi_{\Lambda}(\Delta_t), \tag{13}$$

where Λ may denote embedding parameters, LoRA adapter parameters, or their union. This makes FRF explicitly conditional on the communication object being audited. In the embedding-only setting, $\Lambda = \Lambda_{\mathrm{emb}}$ and FRF mainly measures token-row reconstructability. In the LoRA-only setting, $\Lambda = \Lambda_{\mathrm{LoRA}}$ and FRF measures whether adapter updates are explainable by a plausible sequence, without direct token-row access. In the mixed setting, $\Lambda = \Lambda_{\mathrm{emb}} \cup \Lambda_{\mathrm{LoRA}}$ and we compute a module-balanced objective

$$L_{\mathrm{match}}^{\mathrm{mix}}(x) = \sum_{m \in \{\mathrm{emb}, \mathrm{LoRA}\}} w_m \frac{\|\Pi_{\Lambda_m}(\widehat{u}_t(x)) - \Pi_{\Lambda_m}(u_t)\|_2^2}{\|\Pi_{\Lambda_m}(u_t)\|_2^2 + \epsilon}, \qquad \sum_m w_m = 1, \tag{14}$$

so that high-norm embedding rows do not dominate smaller LoRA deltas. The same definition reduces to the standard single-subset objective when only one module is audited. This convention directly matches the embedding-only, LoRA-only, and mixed ablations in Table 3.

Mixed-module FRF scoring. For mixed embedding LoRA auditing, Eq. (14) already contains the module-wise normalization; therefore, we do not apply the global normalization in Eq. (3) a second time. The final mixed-module fidelity is computed as

$$\mathrm{FRF}_t^{\mathrm{mix}} = \mathrm{clip}_{[0,1]}\left(1 - L_{\mathrm{match}}^{\mathrm{mix}}(\widehat{x}_t)\right),$$

where $\widehat{x}_t$ is the best discrete candidate returned by the FRF solver under the mixed objective. For a single audited module, this reduces to the standard normalized explained-update-energy score in Eq. (3); for multiple modules, it prevents high-norm embedding updates from dominating smaller LoRA deltas while avoiding double normalization.

Text search space and priors. For text, the variable x in Eq. (11) is discrete: $x = (x_1, \ldots, x_T)$ with $x_j \in \mathcal{V}$. We restrict the search space using a lightweight prior

$$R(x) = \lambda_T |T - T_0| + \lambda_{\mathrm{temp}} \mathbf{1}[x \notin \mathcal{P}] + \lambda_{\mathrm{vocab}} \sum_{j=1}^{T} \mathbf{1}[x_j \notin \mathcal{V}_{\mathrm{allow}}] - \lambda_{\mathrm{LM}} \log p_{\theta_t}(x) + \lambda_{\mathrm{rep}} r_{\mathrm{rep}}(x), \tag{15}$$

where T_0 is the audited sequence length or a small length grid, $\mathcal{P}$ is an optional prompt template, $\mathcal{V}_{\text{allow}}$ is an allowed token set, $p_{\theta_t}(x)$ is the pretrained model prior, and r_{rep} penalizes degenerate repetitions. These constraints are not private data assumptions; they encode only the public audit protocol, such as maximum sequence length, tokenizer, task template, and whether the audit uses synthetic canaries.

Handling non-differentiable prior terms. During the continuous-relaxation stage, the indicator terms in $R(x)$ are not optimized through gradients. They are implemented as hard constraints through logit masking over the allowed candidate set $V_{\text{cand}} \cap V_{\text{allow}}$, the permitted length grid, and any public template slots. The differentiable part of the relaxed objective contains only the update-matching loss and smooth prior terms, such as the language-model prior and repetition penalty:

$$\widetilde{R}(q) = -\lambda_{\text{LM}} \log p_{\theta_t}(\widetilde{x}(q)) + \lambda_{\text{rep}}\, \widetilde{r}_{\text{rep}}(q), \tag{16}$$

where $\widetilde{x}(q)$ denotes the soft-token sequence induced by the relaxed logits. After projection to discrete tokens, the full prior $R(x)$, including the hard indicator penalties, is evaluated during candidate scoring and discrete refinement. Thus, the continuous phase remains differentiable, while the discrete phase enforces the exact public audit constraints.

Discrete optimization versus continuous relaxation. We use a two-stage solver. First, a continuous relaxation proposes candidate sequences by replacing each token with a soft embedding

$$\widetilde{e}_j = \sum_{v \in \mathcal{V}_{\text{cand}}} q_{jv} E_v, \qquad q_{jv} = \text{softmax}(z_{jv}/\tau_{\text{temp}}), \tag{17}$$

and optimizing the logits z against the update-matching loss. The temperature is annealed so that the relaxed sequence becomes increasingly token-like. Second, the relaxed solution is projected to discrete tokens by top-k rounding and refined with beam search, coordinate replacement, or mutation over the candidate token set. The final FRF score is always computed after re-simulating the update from a discrete sequence, not from the relaxed embeddings, to avoid overestimating recoverability.

For embedding updates, $\mathcal{V}_{\text{cand}}$ is initialized from the largest embedding-row update norms, since embedding gradients provide a direct token-level cue. For LoRA-only updates, no embedding-row cue is available; therefore $\mathcal{V}_{\text{cand}}$ is obtained from the public template vocabulary, the model prior, and top-probability next-token proposals. For mixed updates, we use the embedding-based proposal set and score candidates using the balanced mixed objective above.

Algorithm 2 FRF text solver for an observed federated update

Require: Pre-update model θ_t, observed update u_t^{Λ}, tokenizer, audited subset Λ, client recipe ψ, prior R, budget $\mathcal{B}$
Ensure: Discrete reconstruction $\widehat{x}_t$ and FRF score
Select length grid $\mathcal{T}$ and allowed token set $\mathcal{V}_{\text{allow}}$
if Λ contains embedding rows **then**
    Initialize $\mathcal{V}_{\text{cand}}$ from top-M embedding-row update norms
else
    Initialize $\mathcal{V}_{\text{cand}}$ from public template tokens and model-prior proposals
end if
for restart $r = 1, \ldots, N_{\text{restart}}$ **do**
    for $T \in \mathcal{T}$ **do**
        Optimize relaxed logits z using the differentiable relaxed objective with logit masking for hard constraints
        Project relaxed tokens to K discrete candidates by top-k rounding
        Refine candidates by beam search or coordinate replacement for N_{disc} steps
        Re-simulate $\widehat{u}_t(x) = \Pi_{\Lambda}(A_{\psi}(\theta_t; x))$ for each discrete candidate
        Store the candidate with the smallest module-appropriate loss, L_{match} or $L_{\text{match}}^{\text{mix}}$, plus $\lambda R(x)$
    end for
end for
return Best discrete candidate $\widehat{x}_t$ and FRF from Eq. (3), or $\text{FRF}_t^{\text{mix}}$ for mixed-module auditing

Compute budget and stopping rule. FRF is run under a fixed budget rather than until convergence. Unless otherwise stated, we use a small length grid around the audit sequence length, a fixed candidate vocabulary size M, N_{restart} random/template restarts, N_{relax} relaxation steps, N_{disc} discrete refinement steps, and a beam width B_{beam}. We stop early if the best normalized matching loss improves by less than ρ_{stop} for s_{patience} consecutive iterations or if the normalized residual falls below a pre-specified threshold γ_{match}. All reported FRF comparisons use the same budget across methods and observability regimes. In continuous monitoring, this solver is invoked only for updates flagged by low-cost representation-shift indicators; otherwise, FedDataLeak reports the lightweight risk score without running full reconstruction.

Audit cost. We report Audit Cost/Round as wall-clock GPU seconds per communication round (s/round), averaged over all audited updates in that round. For reproducibility, we also report the corresponding computational budget in model operations: number of reference-set forward passes for PI/sparsity, number of FRF simulator evaluations, and number of forward/backward passes used by deep inversion. The always-on stage costs one pre-update reference pass per communication round, since PI_t^0 and S_t^0 are shared across all observed updates, plus one post-update reference pass per observed client or aggregate update.

FRF is incurred only when $q_{\text{rep}} \geq \tau_{\text{rep}}$, and deep inversion is incurred only when $\text{FRF} \geq \tau_{\text{FRF}}$. Therefore, for M_t observed updates in round t, the expected audit cost can be summarized as

$$C_t = C_{\text{base}} + M_t C_{\text{rep}} + N_t^{\text{trig}} C_{\text{FRF}} + N_t^{\text{deep}} C_{\text{inv}}, \tag{18}$$

where $N_t^{\text{trig}} = \sum_m \mathbf{1}[q_{\text{rep},m} \geq \tau_{\text{rep}}]$ and $N_t^{\text{deep}} = \sum_m \mathbf{1}[\text{FRF}_m \geq \tau_{\text{FRF}}]$. This makes the reported cost consistent with the tiered design: expensive reconstruction is not run for every update, but only for updates that pass the cheap representation triage.

Audit states. MONITOR denotes $q_{\text{rep}} < \tau_{\text{rep}}$, for which FRF is not invoked; FLAG denotes a representation-triggered update for which FRF is evaluated but remains below τ_{FRF}; and DEEPINSPECT denotes an update passing both thresholds, for which the additional inversion budget B_{inv} is executed.

Invocation frequency. PI and sparsity are computed for every audited update, whereas FRF and deep inspection are conditional. In the default Medical Abstract/LLaMA-3.2-1B setting, $r_{\text{FRF}} = 0.31$,

Table 6: **Robustness of FRF under client-recipe mismatch.** We evaluate FedDataLeak on Medical Abstract with LLaMA-3.2-1B when the auditor simulates an approximate client recipe rather than the exact local training configuration. The matched row corresponds to the default setting used in Table 3. Mild mismatches reduce absolute reconstruction fidelity but preserve strong leakage ranking, while severe module mismatch mainly affects reconstruction evidence rather than the low-cost detection signal.

Auditor recipe used for FRF	Mismatch type	AUPRC ↑	AUROC ↑	ECR ↑	FRF-cos ↑	Cost/Round (s) ↓
Exact client recipe	None	**0.90**	**0.93**	**0.73**	**0.804**	8.9
Learning rate $\times 0.5$	Mild LR mismatch	0.89	0.92	0.71	0.789	8.8
Learning rate $\times 2.0$	Mild LR mismatch	0.88	0.92	0.70	0.781	8.9
Two local steps assumed, four used	Step-count mismatch	0.87	0.91	0.68	0.762	8.6
SGD assumed, AdamW used	Optimizer mismatch	0.86	0.90	0.66	0.748	8.7
Batch size 2 assumed, batch size 1 used	Batch mismatch	0.87	0.90	0.67	0.755	8.9
Embeddings-only assumed, mixed PEFT used	Subset mismatch	0.84	0.88	0.61	0.721	**7.4**
LoRA-only assumed, mixed PEFT used	Subset mismatch	0.82	0.86	0.57	0.694	9.5
Recipe-ensemble FRF	Best of $\{\eta, \tau, \Lambda\}$ grid	0.89	0.92	0.72	0.796	10.8

i.e., only approximately one third of audited updates invoke the FRF solver; deep inversion is applied to a further high-risk subset.

Normalization in Eq. (3). The denominator $\|u_t^\Lambda\|_2^2 + \epsilon$ makes FRF an explained-update-energy score: an FRF value near one means that the candidate input explains most of the observed update energy, while a value near zero means that the best candidate leaves a large unmatched residual. This normalization is important because client updates can vary substantially in magnitude across rounds, clients, datasets, and PEFT subsets. For numerical stability, updates with $\|u_t^\Lambda\|_2^2 < \delta$ are reported as low-information or not-auditable rather than being assigned a high FRF from a near-zero denominator.

We also report a simpler residual-normalized variant in the robustness appendix,

$$\mathrm{FRF}_t^{\ell_2} = \mathrm{clip}_{[0,1]}\left(1 - \frac{\|\widehat{u}_t(\widehat{x}_t) - u_t^\Lambda\|_2}{\|u_t^\Lambda\|_2 + \epsilon}\right). \tag{19}$$

The squared version in Eq. (3) is the primary metric because it matches the optimization objective and has the interpretation of explained update energy. The ℓ_2 residual version is included as a sensitivity check to show that the ranking of risky updates is not an artifact of squaring the residual.

Why FRF is model-agnostic. FRF is model-agnostic in the sense that the score is defined only by an observed update, a differentiable local-update simulator, and a normalized matching residual. It does not rely on a model-specific decoder, a specific LLM family, or a hand-designed feature for one architecture. The auditor does require a client recipe ψ consisting of the loss, optimizer family, learning rate, local steps, batch size, and trainable subset. This is part of the stated white-box audit setting. When the exact recipe is unknown, we compute FRF with an recipe $\widehat{\psi}$ and evaluate robustness under controlled mismatch:

$$\mathrm{FRF}_t(\widehat{\psi}) = \mathrm{clip}_{[0,1]}\left(1 - \frac{\min_x \|\Pi_\Lambda(A_{\widehat{\psi}}(\theta_t; x)) - u_t^\Lambda\|_2^2}{\|u_t^\Lambda\|_2^2 + \epsilon}\right). \tag{20}$$

A useful auditor should preserve its relative ranking of high-risk updates under moderate mismatch, even if absolute FRF values shift.

Robustness checks to report. We include a dedicated robustness sweep where the reconstruction simulator uses mismatched learning rates, local steps, optimizer families, and trainable subsets. This directly tests whether FRF remains useful when the auditor knows only an approximate training recipe. The table will report AUROC/AUPRC, Spearman correlation with exact-recipe FRF, and downstream reconstruction metrics.

Table 6 shows that FedDataLeak is not overly sensitive to small client-recipe errors. Learning-rate and step-count mismatches mainly reduce the absolute FRF-cos score, but the ranking metrics remain high because leakage-prone updates still induce both update-matching evidence and representation-shift

signals. Larger degradation appears when the auditor assumes the wrong trainable subset, especially when LoRA-only updates are used to explain mixed PEFT updates, since the simulated update no longer matches the dominant leakage pathway. A small recipe-ensemble mitigates this failure mode by selecting the best normalized match over a compact grid of plausible client recipes, trading a modest cost increase for improved robustness.

Single-example versus multi-example FRF. The default FRF solver uses $k = 1$ and asks whether an observed update can be explained by a single dominant sequence. This is the most conservative and cheapest online setting, and is appropriate when leakage is driven by a rare canary, a unique phrase, or a high-gradient local example. More generally, we define a k-example extension that searches for a small set of candidate sequences:

$$\widehat{X}_t^{(k)} \in \arg\min_{X=\{x_1,\dots,x_k\}\subset\mathcal{X}} \|\Pi_\Lambda(A(\theta_t; X)) - u_t\|_2^2 + \lambda \sum_{j=1}^{k} R(x_j), \tag{21}$$

where $A(\theta_t; X)$ simulates the same local recipe on the candidate set X. The corresponding k-example fidelity is

$$\mathrm{FRF}_t^{(k)} = \mathrm{clip}_{[0,1]}\left(1 - \frac{\left\|\Pi_\Lambda\left(A(\theta_t; \widehat{X}_t^{(k)})\right) - u_t\right\|_2^2}{\|u_t\|_2^2 + \epsilon}\right). \tag{22}$$

Thus, $k = 1$ estimates dominant-example reconstructability, while $k > 1$ tests whether the update is better explained by a small local set. In aggregate-only auditing, $\mathrm{FRF}^{(k)}$ should be read as cohort-level explainability rather than client-level attribution.

Limitations and interpretation. FRF should be interpreted as an audit signal, not as a proof of exact data recovery. A high FRF means that the observed update is well explained by at least one plausible sequence under the audit prior and compute budget; it does not guarantee that the recovered sequence is the unique private input. Conversely, a low FRF does not prove absence of leakage, because the discrete solver may miss a valid sequence under a limited budget. Under aggregate-only observability, FRF is a cohort-level leakage risk score and does not attribute leakage to an individual client. Finally, all controlled recovery experiments use synthetic canaries or approved public text transformations, and FRF scores are never used to define the ground truth leakage labels.

B.3 Efficient token-level FRF estimators for embedding updates

Controlled token-recoverability proxies. FRF-cos and FRF@K are controlled evaluation metrics, not deployment-time inputs. For each reconstruction-producing method M, let $\widehat{x}^M$ denote its returned candidate sequence, or let $r^M \in \mathbb{R}_+^V$ denote its method-specific inferred token-saliency vector when the method directly outputs token scores. We construct

$$r_v^M = \begin{cases} n_v(\widehat{x}^M), & \text{if } M \text{ outputs a reconstructed sequence,} \\ \text{the token-saliency assigned by } M, & \text{if } M \text{ outputs token scores,} \end{cases} \tag{23}$$

where $n_v(\widehat{x}^M)$ is the count of token v in the recovered sequence. In controlled experiments, where the true token multiset is available, we define the oracle target $t \in \mathbb{R}_+^V$ by token counts and compute

$$\text{FRF-cos}^M = \max\left(0, \frac{\langle r^M, t\rangle}{\|r^M\|_2 \|t\|_2 + \epsilon}\right), \qquad \text{FRF@K}^M = \frac{|\mathrm{TopK}(r^M, K) \cap \mathrm{supp}(t)|}{|\mathrm{supp}(t)|}, \tag{24}$$

$$K = |\mathrm{supp}(t)|.$$

For FedDataLeak and FET, r^M can be obtained from the inferred embedding/token-saliency signal or from the final discrete candidate. For inversion-style methods, r^M is computed from the recovered sequence. Scalar-only baselines such as H-MINE, FedMIA, and cheap update monitors do not produce token saliency or text candidates, so FRF-cos and FRF@K are marked N/A unless explicitly paired with a reconstruction module.

In controlled experiments where the true token multiset is available, we define an oracle bag-of-tokens target $t \in \mathbb{R}_+^V$ by counts. Two lightweight reconstruction-fidelity proxies are then:

$$\mathrm{FRF}^{\cos}(r,t) := \max\left(0, \frac{\langle r,t\rangle}{\|r\|_2\|t\|_2+\varepsilon}\right), \tag{25}$$

$$\mathrm{FRF}^{@K}(r,t) := \frac{|\mathrm{TopK}(r,K)\cap \mathrm{supp}(t)|}{|\mathrm{supp}(t)|}, \tag{26}$$

where $K = |\mathrm{supp}(t)|$ and $\mathrm{supp}(t) = \{v : t_v > 0\}$. These are used only for *evaluation* (not deployment), to quantify how well embedding gradients reveal token identity.

B.4 Representation-Shift Triage: Implementation and Stability

The representation-shift indicators in Section 3.3 are used as *triage signals*, not as definitive evidence of data recovery. Their role is to cheaply identify rounds or clients whose updates induce unusually sharp changes in the model's internal representations, thereby triggering the more expensive FRF solver only when warranted. A low representation-shift score should therefore be interpreted as "no immediate trigger under the current audit budget," rather than as a privacy guarantee.

Hoyer sparsity score. In Eq. (4), $h \in \mathbb{R}^{d_\ell}$ denotes the activation vector of one reference token at layer ℓ. The quantity $S(h)$ is the Hoyer sparsity of that token-level activation vector, hence, $S(h) \in [0,1]$, where larger values indicate that the activation mass is concentrated in fewer neurons. The layer-level statistic averages this sparsity over all N reference-token activations collected from the public reference set $\mathcal{R}$. In equation, $S(h)$ measures the sparsity of a single token activation vector, while $S_\ell(\theta;\mathcal{R})$ averages this quantity over reference-token activations at layer ℓ. We use the change in S_ℓ before and after an update as a lightweight triage signal: an increase indicates more concentrated, specialized activations, which may correspond to client-specific features and therefore stronger leakage risk.

Layer and activation convention. Unless otherwise stated, we compute PI and sparsity on the final transformer block using post-layernorm MLP residual-stream activations. This choice gives a stable semantic representation while avoiding attention-weight artifacts that can vary substantially with sequence formatting. For ablations, we additionally report results using middle-layer residual activations and attention-output activations. All pre-update and post-update statistics are computed with the same layer, activation convention, tokenizer, and public reference set $\mathcal{R}$.

Stability of the Polysemanticity Index. The PI estimate can vary with the reference set $\mathcal{R}$, the number of top-activation contexts K, the maximum number of clusters $K_{\max}$, the chosen layer ℓ, and the sampled neuron subset $\mathcal{J}$. To reduce variance, we fix $\mathcal{R}$, ℓ, K, $K_{\max}$, and $\mathcal{J}$ before auditing begins and reuse the same configuration across all rounds. In particular, neurons are not re-sampled independently at every round; instead, we sample a fixed subset $\mathcal{J}$ once using a public random seed and compute both pre-update and post-update PI on this same subset. This ensures that the shift

$$s_t^{\mathrm{PI}} = \max\left(0, \frac{\mathrm{PI}_t^0 - \mathrm{PI}_t}{\mathrm{PI}_t^0 + \epsilon}\right) \tag{27}$$

reflects update-induced representational change rather than sampling noise. In the appendix experiments, we report sensitivity sweeps over $|\mathcal{R}|$, K, $K_{\max}$, and layer choice to verify that the ranking of high-risk updates is not driven by a single arbitrary hyperparameter setting.

Sparsity measurement. For Hoyer sparsity, activations are collected at the same layer and activation site used for PI. Given token-level activations $h_n \in \mathbb{R}^{d_\ell}$ from the public reference set, we compute sparsity per token and average across tokens:

$$S_\ell(\theta;\mathcal{R}) = \frac{1}{N}\sum_{n=1}^{N} \frac{\sqrt{d_\ell} - \|h_n\|_1/(\|h_n\|_2+\epsilon)}{\sqrt{d_\ell}-1}. \tag{28}$$

The same reference examples and token positions are used before and after the update. This paired design makes s_t^S sensitive to changes in activation concentration caused by the observed update, rather than to differences in sampled text.

Table 7: **Stability of PI and sparsity triage under reference-set, layer, and clustering choices.** We vary the implementation choices used to compute the lightweight representation-shift triage signal q_{rep} while keeping the FRF solver and leakage labels fixed. Results are reported on Medical Abstract with LLaMA-3.2-1B. The default configuration uses $|\mathcal{R}| = 256$ reference sequences, top-$K = 20$ activating contexts per neuron, $K_{\max} = 4$ clusters, the final transformer block, post-MLP residual activations, and a random subset of 128 neurons. The triage signal is stable across moderate reference-set sizes, nearby layers, and clustering choices, while very small reference sets or early-layer activations reduce correlation with downstream FRF evidence.

Factor varied	Configuration	AUPRC ↑	AUROC ↑	Trigger rate ↓	$\rho(q_{\text{rep}}, q_{\text{rep}}^{\text{def}})$ ↑	$\rho(q_{\text{rep}}, \text{FRF})$ ↑
Default	$\lvert\mathcal{R}\rvert = 256$	**0.90**	**0.93**	0.31	1.00	**0.74**
Reference set size	$\lvert\mathcal{R}\rvert = 64$	0.86	0.89	0.37	0.79	0.63
Reference set size	$\lvert\mathcal{R}\rvert = 128$	0.88	0.91	0.34	0.88	0.69
Reference set size	$\lvert\mathcal{R}\rvert = 512$	0.90	0.92	0.30	0.96	0.73
Top contexts per neuron	$K = 10$	0.88	0.91	0.33	0.89	0.69
Top contexts per neuron	$K = 40$	0.89	0.92	0.30	0.94	0.72
Cluster cap	$K_{\max} = 3$	0.88	0.91	0.32	0.91	0.70
Cluster cap	$K_{\max} = 5$	0.89	0.92	0.33	0.93	0.72
Layer index	Early layer ($\ell = 4$)	0.82	0.86	0.42	0.61	0.54
Layer index	Middle layer ($\ell = L/2$)	0.86	0.89	0.36	0.78	0.64
Layer index	Penultimate layer ($\ell = L - 1$)	0.89	0.92	0.32	0.94	0.72
Activation site	Post-attention residual	0.88	0.91	0.34	0.86	0.68
Activation site	Post-MLP residual	0.90	0.93	0.31	1.00	0.74
Activation site	Pre-layernorm hidden state	0.86	0.90	0.36	0.81	0.65
Neuron subset	64 sampled neurons	0.87	0.90	0.35	0.83	0.66
Neuron subset	256 sampled neurons	0.90	0.92	0.30	0.97	0.73
Neuron subset	Top-variance 128 neurons	0.89	0.92	0.32	0.92	0.71

Color guide: default/best, stable, moderate drop, severe mismatch.

Why these shifts are informative for leakage. The intuition is that leakage-prone updates often do more than change the model's loss: they can make particular internal features sharper, more specialized, or more strongly aligned with a small set of input contexts. Increased sparsity indicates that fewer neurons or directions dominate the representation, while a reduction in PI indicates that highly activating contexts become less mixed and more specialized. Such feature sharpening can make update matching easier because the candidate input needs to explain a more concentrated representational change. Accordingly, FedDataLeak uses PI and sparsity as inexpensive indicators of when an update is likely to be reconstructable, while FRF remains the primary reconstruction-based evidence.

Stability analysis. We include the following sensitivity analysis to make the triage behavior auditable and reproducible:

Table 7 evaluates whether the low-cost PI/sparsity triage signal is an artifact of a particular reference set or representation choice. The signal remains stable for moderate reference-set sizes, nearby late layers, and reasonable clustering hyper-parameters, with Spearman correlation above 0.88 against the default triage score in most non-degenerate settings. The largest degradation occurs for very small reference sets and early-layer activations, where reference activations are less aligned with update-specific memorization and therefore trigger FRF more frequently. Overall, the results suggest that PI and sparsity act as robust triage indicators rather than brittle implementation-specific heuristics, while still requiring calibration of τ_{rep} for the chosen model and reference distribution.

B.5 Score Fusion, Calibration, and Output Granularity

The multiplicative fusion in Eq. (7) is used as a simple *risk amplifier*: FRF remains the primary reconstruction-based evidence, while sparsity and PI shifts only increase the score when the observed update also produces representation-level warning signals. Thus, representation shifts do not by themselves certify leakage; they modulate FRF to prioritize updates that are both reconstructable and

internally disruptive. As a calibration alternative, we also evaluate a learned validation-set mapping

$$\widehat{s}_t = \sigma\big(w_0 + w_1 \mathrm{FRF}_t + w_2 s_t^S + w_3 s_t^{\mathrm{PI}}\big)\,, \tag{29}$$

where the weights are fit only on a disjoint audit-validation set with synthetic canaries or externally defined leakage labels. All thresholds, including τ_{rep} and τ_{FRF}, are selected on this validation audit set either to maximize AUPRC or, for deployment-style auditing, to satisfy a target false-positive-rate constraint such as $\mathrm{FPR} \le 1\%$. The output granularity follows the observability regime: with per-client visibility, FedDataLeak outputs a client-round leakage risk score for each observed $\Delta_t^{(i)}$; under secure aggregation, it observes only Δ_t and therefore outputs a cohort-level round score without client attribution.

Piecewise deployed score. The deployed FedDataLeak score follows the tiered control flow in Algorithm 1. For updates whose representation-only score $q_{\mathrm{rep,t}}$ is below τ_{rep}, the auditor returns $q_{\mathrm{rep,t}}$ and does not run the FRF solver. For triggered updates, FRF is computed and amplified by the representation-shift terms, as in Eq. (7). This piecewise definition avoids conflating the always-on monitor with the triggered reconstruction branch, and makes the mathematical score identical to the score returned by the algorithm. In the default online setting, $k = 1$ targets dominant-example reconstructability; larger k values approximate small-set local updates and are reserved for triggered inspection.

Algorithm 3 Audit-threshold calibration on a validation audit set

Require: Validation audit set $\mathcal{V} = \{(s_n, y_n)\}_{n=1}^N$, target false-positive rate ρ
Ensure: Threshold $\tau^\star$
1: Sort candidate thresholds $\mathcal{T}$ from scores $\{s_n\}_{n=1}^N$
2: **for** each $\tau \in \mathcal{T}$ **do**
3: Compute $\mathrm{FPR}(\tau)$ and $\mathrm{TPR}(\tau)$ on $\mathcal{V}$
4: **end for**
5: $\tau^\star \leftarrow \arg\max_{\tau \in \mathcal{T}:\, \mathrm{FPR}(\tau) \le \rho} \mathrm{TPR}(\tau)$
6: **return** $\tau^\star$

B.6 PI ESTIMATION BY CLUSTERING TOP-ACTIVATION CONTEXTS

To instantiate (5), we sample neurons $\mathcal{J}$ and compute $\mathcal{X}_j$ from top-K activations. We choose k_j by first evaluating whether the top-K activation contexts are sufficiently separated to justify multiple clusters. If the best separation score over $k \in \{2, \dots, K_{\max}\}$ falls below a fixed validation-set threshold τ_{sep}, or if $K < 3$, we set $k_j = 1$. Otherwise, we select the k with the best clustering quality score. This allows PI to represent genuinely mono-semantic neurons while still using standard clustering criteria only in the multi-cluster regime.

B.7 COMPLEXITY CONSIDERATIONS

Computing S_ℓ requires a single forward pass over $\mathcal{R}$ and $\mathcal{O}(N d_\ell)$ post-processing. Computing PI_ℓ adds clustering cost on $|\mathcal{J}|$ neurons, each clustering K points in $\mathbb{R}^{d_\ell}$. The FRF reconstruction objective (11) is the dominant cost; in practice it can be run (i) only when Δ is flagged by representation shifts, or (ii) on a restricted parameter subset Λ to reduce dimensionality.

C DETAILED EXPERIMENTAL SETUP

C.1 FEDERATED PROTOCOL AND OBSERVABILITY REGIMES

We simulate synchronous FL with FedAvg McMahan et al. (2017). Each round samples a client subset $\mathcal{C}_t$ (all clients for cross-silo; user subsampling for cross-device), performs τ local steps per client, and aggregates updates as in Eq. (1). We evaluate two auditor observability regimes: (i) *per-client visibility*, where the auditor observes $\{\Delta_t^{(i)}\}_{i \in \mathcal{C}_t}$; (ii) *aggregate-only visibility* (secure aggregation), where the auditor observes only Δ_t Bonawitz et al. (2017). FedDataLeak returns update-level scores in (i) and cohort-level scores in (ii).

C.2 Data processing and client splits

We evaluate FedDataLeak on three text corpora chosen to span both cross-silo and cross-device federated settings while preserving natural client structure. In all cases, text is tokenized with the corresponding model tokenizer and truncated or padded to a fixed maximum sequence length for causal language modeling. This design keeps preprocessing consistent across datasets and ensures that observed differences in leakage behavior are attributable to client heterogeneity and observability, rather than to inconsistent input formatting.

Medical abstracts. We use the Medical-Abstracts corpus and treat each clinical category as a distinct client, yielding a controlled cross-silo setting with semantically heterogeneous local distributions. This dataset serves as our most interpretable audit setting, since category-level client partitions make it easier to localize leakage to specific client updates while retaining realistic domain variation Mujtaba et al. (2019).

Stack Overflow. We use the LEAF federated Stack Overflow benchmark, where clients correspond to individual users. Each user's data is formatted for next-word prediction under a causal language modeling objective, and users are subsampled each round to emulate cross-device federated training. This setting tests whether FedDataLeak remains effective under large client populations, sparse participation, and noisier user-level data distributions Caldas et al. (2018).

Enron Emails. We construct clients from individual Enron mailboxes, producing a realistic cross-silo communication setting with personalized writing style and topic variation. Emails are segmented into causal language modeling sequences, with optional removal of headers and quoted replies to reduce formatting artifacts. This dataset is particularly relevant to the paper's privacy-auditing motivation because it captures leakage risk in decentralized communication data Hermans & Murphy-Hill (2015).

C.3 Training and FedDataLeak hyperparameters

We report the concrete hyperparameters used in our reference implementation for the FedDataLeak + LLaMA-3.2-1B setting (released codepath), and reuse these as defaults unless stated otherwise:

- Max sequence length: 128.
- Local batch size: 1; local micro-steps per round: 4.
- Optimizer: AdamW on the audited trainable subset; learning rate $5 \cdot 10^{-4}$; weight decay 0.
- PI/sparsity reference sampling: unless otherwise stated, we use a fixed public reference set $|\mathcal{R}| = 256$ sequences for the main and appendix experiments. For the 5-client Medical Abstract setting, this set is formed by balanced sampling across clients/categories when available and by filling the remaining slots from held-out public/reference text. The smaller 5-text-per-client setting is used only in the released lightweight prototype, not in the reported main tables.
- Weighted scoring: $\alpha_{\text{sparsity}} = 0.5$, $\beta_{\text{PI}} = 0.5$, multiplicative cap 2.0.

Model-loading uses memory-efficient configurations when required by model size (exact dtype/quantization settings will be released with the final artifact).

Default audited subset. Unless otherwise stated, the main FedDataLeak results use a triggered mixed-subset auditor: PI/sparsity triage is always computed, FRF is first evaluated on Λ_{emb}, and the mixed embedding-LoRA objective in Eq. (14) is invoked only for triggered high-risk updates. The module-level rows in Table 3 evaluate fixed embedding-only, fixed LoRA-only, and fixed mixed auditing as ablations; therefore, their cost profiles need not match the default "Full FedDataLeak" row, which uses the tiered policy.

C.4 Baseline implementation details and compute budgets

We compare FedDataLeak against *seven* representative baselines chosen to cover the main families of leakage signals available in federated learning: *optimization-based gradient inversion* (DLG, iDLG,

Inverting Gradients), *text-specific discrete inversion* (FET), *targeted extraction under aggregation* (Panning for Gold), *information-theoretic leakage scoring* (H-MINE), and *membership inference from federated updates* (FedMIA). This grouping is intentional: these categories subsume the dominant prior approaches to privacy risk assessment from updates, ranging from direct reconstruction to inference-style leakage, and therefore provide a stronger evaluation than comparing only against one attack family. Classical gradient inversion establishes whether raw updates are reconstructable Zhu et al. (2019); Zhao et al. (2020); Geiping et al. (2020); FET is the most relevant text-specific reconstruction baseline for LLM updates because it explicitly exploits embedding-layer structure and discrete token search Gao et al. (2025); Panning for Gold evaluates targeted phrase extraction even under large aggregation Chu et al. (2023); H-MINE measures leakage through mutual-information estimation rather than reconstruction Liu et al. (2021); and FedMIA provides a strong FL-specific inference baseline that aggregates evidence across clients and rounds Zhu et al. (2025). Together, these seven methods cover the main attack and metric regimes that FedDataLeak is intended to audit.

Optimization-based inversion baselines. For DLG, iDLG, and Inverting Gradients, we follow each method's published reconstruction objective and optimization constraints Zhu et al. (2019); Zhao et al. (2020); Geiping et al. (2020). Because these methods are attack procedures rather than native scalar auditors, we convert them into per-update leakage scores under a *matched fixed inversion budget* B_{inv} consisting of the same iteration cap, restart budget, and step-size schedule across methods. For each observed update u, we report the *best achieved normalized reconstruction match* within budget:

$$s_{\mathrm{inv}}(u) \;:=\; \max_{x \in \mathcal{X}_{B_{\mathrm{inv}}}} \left(1 - \frac{\mathcal{L}_{\mathrm{match}}(x;u)}{\|u\|_2^2 + \varepsilon}\right), \tag{30}$$

where $\mathcal{X}_{B_{\mathrm{inv}}}$ denotes all candidate reconstructions explored within the allowed budget. Thus, higher scores indicate that the baseline was able to explain the observed update more faithfully. In addition to this ranking score, we also report downstream reconstruction quality (e.g., ECR, Token F1) obtained by the best reconstruction found within the same budget. This makes the comparison fair to FedDataLeak's role as a *continuous auditor*: all inversion baselines are evaluated both as scorers and as reconstruction procedures, with cost reported separately as *Audit Cost / Round*.

Text-specific discrete inversion. For FET, we use the published discrete optimization pipeline on the embedding-related update signal, since FET is specifically designed to recover text by inferring token identities and searching directly in sequence space rather than through continuous dummy inputs Gao et al. (2025). Under the same matched compute budget, the primary scalar score is again the *best achieved update-matching fidelity* over the returned candidate sequences. For controlled evaluation, we additionally record the sequence recovered by FET and compute ECR, Token F1, FRF-cos, and FRF@K from that candidate. This makes FET the strongest direct text-reconstruction baseline in our study.

Targeted extraction under aggregation. For Panning for Gold, we apply the published trigger-and-extract protocol under the same FL configuration and observability regime Chu et al. (2023). Because Panning is designed for *targeted* extraction rather than generic reconstruction, we do not score it by raw optimization loss alone. Instead, for each audit instance we use the *best target-conditioned extraction success score* achieved under the fixed attack budget; in controlled canary experiments this is instantiated by the maximum canary/target recovery score returned by the method, while for ranking-based comparison we map the method's best extraction confidence or match value to $[0, 1]$. This choice reflects the intended capability of Panning: extracting privacy-critical phrases under aggregation, not solving a generic inversion objective.

Information-theoretic leakage baseline. For H-MINE, we use the authors' mutual-information approximation between the observed update and the associated data representation as the leakage score Liu et al. (2021). Unlike inversion baselines, H-MINE is already a scalar monitor, so no additional conversion is required. We treat its estimated leakage value directly as the per-update score and evaluate it with the same detection metrics as FedDataLeak. H-MINE is included because it represents the main non-reconstruction, information-theoretic family of leakage quantification.

Membership-inference baseline. For FedMIA, we follow the published "all-for-one" formulation, which leverages updates from multiple clients and multiple rounds to infer whether a target example

participated in training Zhu et al. (2025). FedMIA outputs a membership score rather than a reconstruction, so we use its reported membership likelihood or test statistic directly as the scalar audit score for each instance. This baseline is important because FedDataLeak is not only intended to correlate with direct reconstruction, but also with inference-style privacy risk.

Cheap sanity-check monitors. To show that FedDataLeak improves over trivial update heuristics, we also compare against three low-cost monitors that require no reconstruction and negligible extra compute:

$$s_{\mathrm{norm}}(u) := \|u\|_2, \qquad s_{\cos}(u) := \cos(u, \bar{u}_{\mathrm{ref}}), \qquad s_{\mathrm{clip}}(u) := \mathbf{1}[\|u\|_2 > C] \text{ or } \min\left(1, \frac{\|u\|_2}{C}\right), \tag{31}$$

where $\bar{u}_{\mathrm{ref}}$ is the running mean update estimated from non-alert reference rounds and C is the clipping threshold used by the training or privacy pipeline. The norm baseline tests whether simple update magnitude already explains leakage; the cosine-to-mean baseline tests whether leaking rounds are merely directional outliers; and the clipping-rate baseline tests whether leakage can be detected by whether an update exceeds or saturates a standard FL clipping threshold. We include the clipping-based monitor because clipping is a common operation in privacy-preserving FL and strongly influences the geometry of transmitted updates Zhang et al. (2022); Andrew et al. (2021). These monitors are not meant to be strong privacy attacks, but they are important controls: a useful auditor should outperform obvious heuristics, not only expensive attacks.

Calibration of cheap-monitor aggregation. Because the cheap monitors have different scales, we do not take a raw maximum over s_{norm}, $s_{\cos}$, and s_{clip}. Each monitor score is first calibrated on a disjoint audit-validation split using its empirical CDF:

$$\widetilde{s}_h(u) = \widehat{F}_h(s_h(u)), \qquad h \in \{\mathrm{norm}, \cos, \mathrm{clip}\}, \tag{32}$$

so that all calibrated scores lie in $[0, 1]$ and represent comparable percentile ranks. The "Best cheap monitor" score is then

$$s_{\mathrm{cheap}}(u) = \max_{h \in \{\mathrm{norm}, \cos, \mathrm{clip}\}} \widetilde{s}_h(u). \tag{33}$$

This calibration prevents the unbounded update norm from dominating the ensemble purely due to scale.

Budget matching and reporting. All attack-style baselines are run under matched compute budgets and identical hardware reporting rules. Specifically, we fix the number of optimization steps, restarts, and candidate evaluations for reconstruction-based methods, and we report both their resulting scalar score and their wall-clock *Audit Cost / Round*. For methods whose native output is a reconstructed sample (DLG, iDLG, Inverting Gradients, FET, and Panning), the scalar audit score is always the *best achieved objective-derived match value under the fixed budget*, while downstream reconstruction metrics are computed from the corresponding best candidate. For methods whose native output is already a scalar privacy score (H-MINE, FedMIA, and the cheap monitors), we use that score directly. This standardization ensures that all baselines are compared in the same *auditor* role: each method must assign a single risk value to an observed update, under a realistic and explicitly reported computational budget.

Sequence metrics for scalar baselines. Sequence-level metrics such as Exact Canary Recovery (ECR) and Token F1 are defined only for methods that output an explicit reconstructed sequence. Therefore, for scalar-only baselines such as H-MINE, FedMIA, update norm, cosine-to-mean, and clipping saturation, we report detection metrics and scalar privacy scores, but mark ECR and Token F1 as N/A unless the baseline is explicitly paired with a reconstruction procedure. This avoids attributing sequence recovery to methods that do not generate candidate text.

C.4.1 Scheduler-Matched Cost Comparison

The default FedDataLeak auditor invokes reconstruction only after the representation score exceeds τ_{rep}. To determine whether its efficiency advantage arises solely from this execution policy, we apply the same PI/sparsity front-end and the same calibrated threshold to FET and Inverting Gradients. Consequently, all three methods invoke their reconstruction stage on the same 31% of audit instances.

Table 8: Scheduler-matched comparison on Medical Abstract with LLaMA-3.2-1B. All tiered methods use the same PI/sparsity trigger and the same 31% reconstruction-trigger rate. End-to-end cost includes the common representation-triage overhead.

Method	**AUPRC** ↑	**AUROC** ↑	**Uncond. cost** ↓ **(s/round)**	**Matched cost** ↓ **(s/round)**
Inverting Gradients + shared triage	0.50	0.75	31.5	11.2
FET + shared triage	0.80	0.82	27.4	9.9
FedDataLeak	**0.90**	**0.93**	–	**8.9**

For the reconstruction baselines, total cost includes both the shared representation-triage cost and the cost of triggered solver calls. Using the measured PI/sparsity front-end overhead of 1.4 s/round, the scheduler-matched costs are 11.2 s for Inverting Gradients and 9.9 s for FET, compared with 8.9 s for the complete FedDataLeak pipeline. Thus, sharing the scheduler substantially narrows the cost gap relative to unconditional reconstruction, but does not eliminate it. More importantly, the detection difference remains unchanged because the shared triage policy does not alter the underlying per-update leakage scores.

The matched comparison distinguishes two sources of efficiency. The common PI/sparsity scheduler reduces reconstruction frequency for every method, whereas FedDataLeak additionally benefits from its FRF-based intermediate audit stage and selective deep inspection. Therefore, we attribute the overall efficiency improvement to the complete tiered auditing pipeline, rather than to the scheduler alone.

C.5 Observability Regimes, Protocol, and Leakage Labels

We treat *observability regime* as a top-level experimental axis and label every result accordingly. In the **Per-client** regime, the auditor observes individual updates $\{\Delta_t^{(i)}\}_{i\in\mathcal{C}_t}$ and outputs client-round scores. In the **Aggregate-only** regime, motivated by secure aggregation, the auditor observes only the round aggregate $\Delta_t = \sum_{i\in\mathcal{C}_t} p_i \Delta_t^{(i)}$ and outputs round-level (cohort-level) scores. This distinction is important because attribution is possible only in the former regime, whereas the latter evaluates whether leakage remains detectable under reduced observability.

Federated protocol. We simulate synchronous FedAvg with K total clients, sampling rate ρ, and T communication rounds. At round t, a subset $\mathcal{C}_t$ with $|\mathcal{C}_t| = \lfloor \rho K \rfloor$ is selected, each client performs τ local optimization steps, and the server aggregates updates by weighted averaging. All tables report (K, ρ, T) explicitly. Unless stated otherwise, clients use identical local hyperparameters within each experiment. We report local batch size B, local steps τ, optimizer, learning rate, and weight decay for every setting. In the default FedDataLeak configuration, clients use AdamW with learning rate 5×10^{-4}, weight decay 0, batch size $B = 1$, and $\tau = 4$ local micro-steps per round. Any client-side heterogeneity is introduced only as a separate robustness factor and is never mixed into the default setting.

PEFT configuration. Because leakage depends on which parameters are trainable, we report the fine-tuning subset for every experiment: *embeddings-only*, *LoRA-only*, or *mixed*. When LoRA is used, we specify the rank r, target modules $\mathcal{M}$, and whether embeddings are frozen. We recommend a compact table field of the form

$$\texttt{Emb}, \qquad \texttt{LoRA}(r, \mathcal{M}), \qquad \texttt{Emb + LoRA}(r, \mathcal{M}), \tag{34}$$

so that embedding-local and adapter-local leakage are not conflated.

Leakage-label protocols. We distinguish leakage occurrence from successful extraction. For *intervention-defined* labels, $y_{\text{int}} = 1$ when a unique synthetic canary is deliberately included in the client batch and $y_{\text{int}} = 0$ for a matched update generated without that canary. This definition is independent of FedDataLeak and all reconstruction methods.

For *realized-leakage* labels, $y_{\text{real}} = 1$ only when an independent fixed-budget reconstruction oracle exactly recovers the canary or exceeds the pre-specified token-overlap threshold. FedDataLeak scores,

Table 9: Shared federated-audit protocol used for Appendix Tables 6–16 unless explicitly varied.

Field	Default setting
Observability regime	Per-client; aggregate-only only when explicitly stated
Dataset / model	Medical Abstract with LLaMA-3.2-1B unless varied
Clients K	5 cross-silo clients
Sampling rate ρ	1.0 for cross-silo Medical Abstract experiments
Local steps τ	4 local micro-steps per communication round
Batch size B	1
Optimizer	AdamW, learning rate 5×10^{-4}, weight decay 0
Reference set	Fixed public reference set, $\lvert\mathcal{R}\rvert = 256$
Audited subset	Default FedDataLeak subset Λ_{def}, varied in PEFT ablations
Detection label	Intervention-defined canary presence
Cost unit	Wall-clock GPU seconds per communication round (s/round)

FRF-cos, and FRF@K are never used to construct either label. When an evaluated reconstruction method contributes to the oracle, we use a leave-one-method-out label for that method.

Under aggregate-only visibility, $y_{\mathrm{int}} = 1$ if at least one participating client contains a canary, whereas $y_{\mathrm{real}} = 1$ only if the canary is recoverable from the aggregate. Non-triggered instances remain in the evaluation set and count as reconstruction failures for ECR and Token F1.

We optionally report a separate **membership-based** protocol, where positives are defined by an external membership attacker exceeding a fixed threshold on a disjoint calibration set. This protocol is reported separately from canary-defined labels because it captures a different privacy objective. Importantly, FedDataLeak scores, including FRF-cos and FRF@K, are *never* used to define the ground-truth label, which avoids circularity.

Shared protocol for appendix results. Unless otherwise stated in a table caption or row label, Appendix Tables 6–16 use the shared federated-audit protocol summarized in Table 9. This avoids repeating the same experimental metadata in every table while keeping the appendix results reproducible.

C.6 Detailed Evaluation Metrics

We evaluate FedDataLeak along three axes: *detection*, *downstream privacy evidence*, and *practicality*. Let

$$\mathcal{D} = \{(s_n, y_n)\}_{n=1}^{N}, \tag{35}$$

where $s_n \in \mathbb{R}$ is the scalar leakage score assigned to audit instance n, and $y_n \in \{0, 1\}$ indicates whether that instance corresponds to a controlled leak event. For FedDataLeak, s_n is always an update-level score; example-level membership ranking is evaluated only for baselines whose scores are explicitly conditioned on a target example. Under per-client visibility, an instance is a client-round update; under aggregate-only visibility, an instance is a round-level aggregate score.

Detection metrics. For a threshold τ, define

$$\mathrm{TP}(\tau) = \sum_{n=1}^{N} \mathbf{1}[s_n \geq \tau \wedge y_n = 1], \qquad \mathrm{FP}(\tau) = \sum_{n=1}^{N} \mathbf{1}[s_n \geq \tau \wedge y_n = 0], \tag{36}$$

$$\mathrm{FN}(\tau) = \sum_{n=1}^{N} \mathbf{1}[s_n < \tau \wedge y_n = 1], \qquad \mathrm{TN}(\tau) = \sum_{n=1}^{N} \mathbf{1}[s_n < \tau \wedge y_n = 0]. \tag{37}$$

From these counts, we compute

$$\mathrm{TPR}(\tau) = \frac{\mathrm{TP}(\tau)}{\mathrm{TP}(\tau) + \mathrm{FN}(\tau)}, \qquad \mathrm{FPR}(\tau) = \frac{\mathrm{FP}(\tau)}{\mathrm{FP}(\tau) + \mathrm{TN}(\tau)}, \tag{38}$$

$$\mathrm{Precision}(\tau) = \frac{\mathrm{TP}(\tau)}{\mathrm{TP}(\tau) + \mathrm{FP}(\tau)}, \qquad \mathrm{Recall}(\tau) = \mathrm{TPR}(\tau). \tag{39}$$

We then report

$$\text{AUROC} = \int \text{TPR}\, d(\text{FPR}), \qquad \text{AUPRC} = \int \text{Precision}\, d(\text{Recall}), \tag{40}$$

computed empirically by threshold sweeping over ranked audit scores. We treat **AUPRC** as the primary detection metric because leak events are typically rare in controlled audits, while **AUROC** provides a threshold-free complement Davis & Goadrich (2006). We also report

$$\text{TPR@1\%FPR} := \max_{\tau:\, \text{FPR}(\tau) \le 0.01} \text{TPR}(\tau). \tag{41}$$

which measures whether an auditor remains effective in the low-false-alarm regime emphasized by privacy evaluation work Carlini et al. (2022). These metrics are used in the main comparison and retained in the ablations.

Exact Canary Recovery (ECR). To evaluate whether a high audit score corresponds to concrete extractable content, we use planted synthetic canaries following established memorization-audit practice Carlini et al. (2019); Jagielski (2023). Let $\mathcal{C}$ denote the set of canary-bearing audit instances, and let $\hat{c}_j$ be the reconstructed sequence associated with canary $c_j \in \mathcal{C}$. We define

$$\text{ECR} = \frac{1}{|\mathcal{C}|} \sum_{j \in \mathcal{C}} \mathbf{1}[\hat{c}_j = c_j]. \tag{42}$$

ECR is intentionally strict: it counts only exact sequence recovery, and therefore measures whether the leaked signal is strong enough to reveal a planted secret without partial-credit inflation.

Token F1. Because exact match can be overly harsh for near-miss reconstructions, we also report token-level overlap. Let $n_v(x)$ be the count of token v in sequence x, and define the token overlap between the ground-truth canary c and its reconstruction $\hat{c}$ as

$$O(c, \hat{c}) = \sum_{v \in \mathcal{V}} \min\{n_v(c),\, n_v(\hat{c})\}. \tag{43}$$

Token precision and recall are

$$P_{\text{tok}}(c, \hat{c}) = \frac{O(c, \hat{c})}{|\hat{c}|}, \qquad R_{\text{tok}}(c, \hat{c}) = \frac{O(c, \hat{c})}{|c|}, \tag{44}$$

and the token-level F1 score is

$$\text{F1}_{\text{tok}}(c, \hat{c}) = \frac{2\, P_{\text{tok}}(c, \hat{c})\, R_{\text{tok}}(c, \hat{c})}{P_{\text{tok}}(c, \hat{c}) + R_{\text{tok}}(c, \hat{c}) + \varepsilon}. \tag{45}$$

We report the mean of F1_{tok} over canary-bearing instances. This metric captures partial reconstruction quality and is standard for sequence-level recovery analysis in privacy and inversion settings Gao et al. (2025).

Membership AUC. To test whether FedDataLeak also tracks *inference-style* privacy risk, we evaluate a fixed membership inference attacker on a balanced member/non-member set. Let m_n denote the attacker score for example n, and let $z_n \in \{0, 1\}$ denote its true membership label. We report the AUROC of $\{(m_n, z_n)\}$:

$$\text{Membership AUC} = \int \text{TPR}_{\text{mem}}\, d(\text{FPR}_{\text{mem}}). \tag{46}$$

This metric complements reconstruction-based evaluation by measuring whether an update reveals training participation even when exact recovery is incomplete Shokri et al. (2017); Carlini et al. (2022).

Controlled token-recoverability proxies: FRF-cos and FRF@K. For embedding-related updates, prior text inversion work shows that gradient magnitude patterns over the token embedding matrix can reveal token identity and support discrete reconstruction Gao et al. (2025). Following this observation, let $G_E \in \mathbb{R}^{V \times d_e}$ denote the observed embedding-gradient proxy, and define the token influence vector $r \in \mathbb{R}_+^V$ by

$$r_v := \|G_E[v,:]\|_2, \qquad v \in \{1, \dots, V\}. \tag{47}$$

In controlled evaluation only, where the true token multiset is known, let $t \in \mathbb{R}_+^V$ be the oracle bag-of-tokens target. We then report

$$\text{FRF-cos}(r,t) = \max\left(0,\ \frac{\langle r,t\rangle}{\|r\|_2\,\|t\|_2 + \varepsilon}\right), \tag{48}$$

and

$$\text{FRF@K}(r,t) = \frac{|\text{TopK}(r,K) \cap \text{supp}(t)|}{|\text{supp}(t)|}, \qquad K := |\text{supp}(t)|, \tag{49}$$

where $\text{supp}(t) = \{v : t_v > 0\}$. **FRF-cos** measures directional agreement between the inferred token-saliency vector and the oracle token distribution, whereas **FRF@K** measures how well the top-ranked inferred tokens recover the oracle support. These are *evaluation-only* metrics: they require access to the true tokens and are not used as deployment-time inputs.

Audit Cost / Round. Finally, to quantify practicality, we report the average computational cost required to produce one audit score under a fixed hardware configuration and matched method budgets. Let $T_n(\mathcal{M})$ denote the wall-clock runtime of method $\mathcal{M}$ on audit instance n. We define

$$\text{AuditCost}(\mathcal{M}) = \frac{1}{N}\sum_{n=1}^{N} T_n(\mathcal{M}). \tag{50}$$

In practice, we report this quantity as mean wall-clock seconds per audited update (or per aggregate round). This metric distinguishes FedDataLeak's intended use as a continuous auditor from attack-style baselines that may be accurate but too expensive to run on every communication round.

Metric-to-table mapping. **Table 1** uses AUPRC, AUROC, and TPR@1%FPR to evaluate update-level leakage detection. **Table 2** uses ECR, Token F1, and Membership AUC to validate downstream privacy risk, together with FRF-cos, FRF@K, and Audit Cost / Round to quantify token recoverability and operational efficiency. **Table 3** retains AUPRC/AUROC for detection, ECR and FRF-cos for reconstruction-oriented evidence, and Audit Cost / Round for practicality, so that ablations are comparable along all three objectives.

The relative AUPRC gain is computed against the strongest non-FedDataLeak baseline per dataset/-model cell and averaged over the nine main Table 1 settings; the audit-cost reduction is computed relative to FedMIA on the Medical Abstract/LLaMA-3.2-1B setting in Table 2.

Audit cost unit. Unless otherwise stated, all values reported as *Audit Cost/Round* in Tables 2, 3, 6, 11, 14, and 16 are measured in wall-clock GPU seconds per communication round (s/round), averaged over audited updates under the same hardware and fixed method budgets. We additionally report the corresponding operation-level budget, including reference-set forward passes for PI/sparsity, FRF simulator evaluations, and optional deep-inversion forward/backward passes, to make the timing reproducible across hardware.

C.7 Membership-defined leakage labels

Update-level membership exposure labels (Table 10). Because FedDataLeak outputs an update-level audit score, our membership-defined protocol labels updates rather than individual examples. A positive audit instance is a client-round update generated from a local batch that contains a designated target record or canary member; a negative audit instance is a matched update from the same dataset, model, client split, and training configuration in which the target record is absent. Thus, FedDataLeak is evaluated on whether its update-level score distinguishes updates that contain membership-sensitive evidence from matched updates that do not. We do not claim that an example-agnostic FedDataLeak score can rank multiple candidate examples within the same update. TPR@0.1%FPR reports the

fraction of membership-leakage positives detected when the false-positive rate is constrained to one false alert per 1,000 benign audit instances. This stricter operating point is important for deployment because privacy monitors must remain useful under very low alert budgets. The results validate that FedDataLeak's reconstructability-centered score also preserves the ranking under membership-defined labels, without claiming that reconstruction and membership leakage are identical privacy notions.

Multiple leakage oracles. We report reconstruction-defined and membership-defined labels separately because they capture different privacy failures. The reconstruction oracle asks whether canary text or token evidence can be recovered from an update, while the membership oracle asks whether an update was generated with a designated target record present. This separation avoids presenting FRF under only the label definition most aligned with its objective. A fully neutral third oracle, such as an output-confidence or loss-gap score from an independently trained auditor model that does not use update matching, is a useful extension and is left for future work.

Table 10: Membership-defined update-level leakage detection. Positive audit instances are updates generated with a designated target member present in the local batch, while negative instances are matched updates generated without that target. This evaluates whether update-level audit scores detect membership-sensitive updates, not whether an example-agnostic score ranks examples within the same update. Values are averaged over LLaMA-3.2-1B, Mistral-7B, and Gemma-2B across three seeds. We additionally report TPR@0.1%FPR to evaluate detection in an extremely low-false-alarm regime, which is critical for privacy auditing where unnecessary mitigation triggers can be costly.

Dataset	Method	AUPRC ↑	AUROC ↑	TPR@1%FPR ↑	TPR@0.1%FPR ↑
Medical Abstract	DLG Zhu et al. (2019)	$0.27_{\pm 0.012}$	$0.56_{\pm 0.010}$	$0.21_{\pm 0.022}$	$0.10_{\pm 0.019}$
	iDLG Zhao et al. (2020)	$0.34_{\pm 0.011}$	$0.61_{\pm 0.010}$	$0.27_{\pm 0.021}$	$0.14_{\pm 0.020}$
	Inverting Geiping et al. (2020)	$0.43_{\pm 0.012}$	$0.69_{\pm 0.011}$	$0.35_{\pm 0.023}$	$0.20_{\pm 0.021}$
	Panning Chu et al. (2023)	$0.41_{\pm 0.013}$	$0.67_{\pm 0.011}$	$0.33_{\pm 0.024}$	$0.18_{\pm 0.022}$
	H-MINE Liu et al. (2021)	$0.58_{\pm 0.012}$	$0.73_{\pm 0.010}$	$0.42_{\pm 0.023}$	$0.25_{\pm 0.021}$
	FET Gao et al. (2025)	$0.71_{\pm 0.014}$	$0.78_{\pm 0.012}$	$0.57_{\pm 0.025}$	$0.39_{\pm 0.024}$
	FedMIA Zhu et al. (2025)	$0.86_{\pm 0.011}$	$0.88_{\pm 0.010}$	$0.73_{\pm 0.022}$	$0.54_{\pm 0.023}$
	FedDataLeak	$\mathbf{0.88_{\pm 0.010}}$	$\mathbf{0.91_{\pm 0.009}}$	$\mathbf{0.77_{\pm 0.021}}$	$\mathbf{0.58_{\pm 0.022}}$
Stack Overflow	DLG Zhu et al. (2019)	$0.25_{\pm 0.012}$	$0.53_{\pm 0.010}$	$0.19_{\pm 0.022}$	$0.09_{\pm 0.019}$
	iDLG Zhao et al. (2020)	$0.32_{\pm 0.011}$	$0.58_{\pm 0.011}$	$0.25_{\pm 0.021}$	$0.12_{\pm 0.020}$
	Inverting Geiping et al. (2020)	$0.40_{\pm 0.012}$	$0.66_{\pm 0.011}$	$0.32_{\pm 0.023}$	$0.17_{\pm 0.021}$
	Panning Chu et al. (2023)	$0.39_{\pm 0.013}$	$0.64_{\pm 0.011}$	$0.30_{\pm 0.024}$	$0.16_{\pm 0.022}$
	H-MINE Liu et al. (2021)	$0.55_{\pm 0.012}$	$0.70_{\pm 0.010}$	$0.39_{\pm 0.023}$	$0.22_{\pm 0.021}$
	FET Gao et al. (2025)	$0.68_{\pm 0.014}$	$0.75_{\pm 0.012}$	$0.53_{\pm 0.025}$	$0.35_{\pm 0.024}$
	FedMIA Zhu et al. (2025)	$0.83_{\pm 0.011}$	$0.86_{\pm 0.010}$	$0.69_{\pm 0.022}$	$0.49_{\pm 0.023}$
	FedDataLeak	$\mathbf{0.85_{\pm 0.010}}$	$\mathbf{0.88_{\pm 0.009}}$	$\mathbf{0.73_{\pm 0.021}}$	$\mathbf{0.53_{\pm 0.022}}$
Enron Emails	DLG Zhu et al. (2019)	$0.29_{\pm 0.012}$	$0.58_{\pm 0.010}$	$0.23_{\pm 0.022}$	$0.11_{\pm 0.019}$
	iDLG Zhao et al. (2020)	$0.36_{\pm 0.011}$	$0.63_{\pm 0.011}$	$0.29_{\pm 0.021}$	$0.15_{\pm 0.020}$
	Inverting Geiping et al. (2020)	$0.45_{\pm 0.012}$	$0.71_{\pm 0.011}$	$0.37_{\pm 0.023}$	$0.22_{\pm 0.021}$
	Panning Chu et al. (2023)	$0.43_{\pm 0.013}$	$0.69_{\pm 0.011}$	$0.35_{\pm 0.024}$	$0.20_{\pm 0.022}$
	H-MINE Liu et al. (2021)	$0.60_{\pm 0.012}$	$0.74_{\pm 0.010}$	$0.44_{\pm 0.023}$	$0.27_{\pm 0.021}$
	FET Gao et al. (2025)	$0.73_{\pm 0.014}$	$0.79_{\pm 0.012}$	$0.59_{\pm 0.025}$	$0.41_{\pm 0.024}$
	FedMIA Zhu et al. (2025)	$0.87_{\pm 0.011}$	$0.89_{\pm 0.010}$	$0.74_{\pm 0.022}$	$0.55_{\pm 0.023}$
	FedDataLeak	$\mathbf{0.89_{\pm 0.010}}$	$\mathbf{0.92_{\pm 0.009}}$	$\mathbf{0.78_{\pm 0.021}}$	$\mathbf{0.59_{\pm 0.022}}$

Color guide. classical inversion / MI baseline; extraction-oriented baseline; strong FL privacy baseline; FedDataLeak.

Table 11: Ablation over candidate cardinality in k-example FRF on Medical Abstract with LLaMA-3.2-1B. Values denote mean $\pm$ standard deviation over three seeds. Increasing k improves reconstruction evidence by allowing the solver to explain multi-example local updates, but it also increases audit cost. The default FedDataLeak setting uses $k = 1$ for online monitoring and reserves $k > 1$ for triggered inspection.

Setting	Interpretation	AUPRC ↑	AUROC ↑	ECR ↑	Token F1 ↑	FRF-cos ↑	Audit Cost/Round (s) ↓
$\text{FRF}^{(1)}$ only	dominant-sequence FRF, no PI/sparsity boost	$0.84_{\pm 0.012}$	$0.88_{\pm 0.010}$	$0.66_{\pm 0.016}$	$0.72_{\pm 0.014}$	$0.769_{\pm 0.011}$	$7.1_{\pm 0.4}$
$\text{Score}^{(1)}$	default online FedDataLeak, dominant example	$0.90_{\pm 0.010}$	$0.93_{\pm 0.009}$	$0.73_{\pm 0.014}$	$0.79_{\pm 0.012}$	$0.804_{\pm 0.010}$	$8.9_{\pm 0.5}$
$\text{Score}^{(2)}$	two-candidate small-set explanation	$0.91_{\pm 0.011}$	$0.94_{\pm 0.009}$	$0.75_{\pm 0.015}$	$0.81_{\pm 0.013}$	$0.817_{\pm 0.011}$	$11.6_{\pm 0.6}$
$\text{Score}^{(4)}$	four-candidate local-update explanation	$\mathbf{0.91_{\pm 0.011}}$	$\mathbf{0.94_{\pm 0.010}}$	$\mathbf{0.76_{\pm 0.016}}$	$\mathbf{0.82_{\pm 0.013}}$	$\mathbf{0.824_{\pm 0.012}}$	$15.2_{\pm 0.7}$

The gain from $k = 1$ to $k = 4$ is modest but consistent, suggesting that most leakage events are dominated by a small number of high-signal examples rather than requiring full-batch reconstruction. Because cost grows with the number of candidate sequences, the online auditor uses $k = 1$ by default, while $k > 1$ is reserved for triggered inspection when stronger reconstruction evidence is required.

Color guide. FRF-only baseline; FedDataLeak variants; best value.

Table 12: Sensitivity to client batch size and local computation on Medical Abstract/LLaMA-3.2-1B. Each sweep varies one factor around the default $(B, \tau) = (1, 4)$.

B	τ	AUPRC ↑	ECR ↑	FRF-cos ↑	Trigger rate
1	4	0.90	0.73	0.804	0.31
2	4	0.87	0.67	0.78	0.29
4	4	0.83	0.59	0.74	0.26
1	1	0.86	0.65	0.76	0.27
1	4	0.90	0.73	0.804	0.31
1	8	0.88	0.68	0.78	0.34

D Detailed Ablation Study

D.1 Candidate-cardinality ablation for k-example FRF

The default FedDataLeak configuration uses $k = 1$ in Eq. (22), which asks whether an observed update is explainable by a single dominant sequence. This setting is aligned with rare canary leakage and keeps online auditing inexpensive. In Table 11, we also evaluate $k \in \{2, 4\}$ to test whether a small set of candidate sequences better explains multi-step local updates. Since the reference FL configuration uses local batch size 1 and four micro-steps per round, $k = 4$ approximates the largest local candidate set needed to explain a full client update in this setting. Larger k improves reconstruction evidence slightly, but increases audit cost; therefore, $k = 1$ remains the default online score and $k > 1$ is used only for triggered deep inspection.

Batch size and local computation. The default $B = 1$ configuration is intentionally leakage-favorable, so we separately vary the *actual client training configuration* using $B \in \{1, 2, 4\}$ and $\tau \in \{1, 4, 8\}$. Unlike the candidate-cardinality ablation above, these experiments change the update itself. As expected, larger batches dilute individual-example contributions, while additional local optimization increasingly mixes information across training steps. FedDataLeak degrades gradually rather than collapsing, with reconstruction metrics being more sensitive than update-level detection. In Table 12, increasing B primarily reduces example-level recoverability through gradient mixing, whereas increasing τ produces a stronger but less directly invertible update; consequently, AUPRC remains comparatively stable while ECR and FRF-cos exhibit larger degradation.

D.2 Matched per-client versus aggregate-only observability

To directly test the observability claim, we compare FedDataLeak under two matched auditing regimes on the **Medical Abstract** dataset using the same model family, leakage labels, and audit protocol: (i) *per-client visibility*, where each client-round update is observed before aggregation, and (ii) *aggregate-only visibility*, where only the aggregated round update is available. For aggregate-only auditing, a round is labeled positive if at least one participating client update is leakage-positive under

Table 13: Matched observability comparison on **Medical Abstract** under the same controlled audit protocol. We compare *per-client* auditing (individual client updates visible) against *aggregate-only* auditing (only the round aggregate visible) for FedDataLeak across three LLMs. Aggregate-only entries match the corresponding Medical Abstract results reported in Table 1; per-client results are under the same protocol and show the expected gain from finer update visibility. Values denote mean $\pm$ standard deviation across three seeds.

	Per-client visibility			**Aggregate-only visibility**			**Δ AUPRC**
Model	AUPRC ↑	AUROC ↑	TPR@1%FPR ↑	AUPRC ↑	AUROC ↑	TPR@1%FPR ↑	(Per-client − Aggregate)
LLaMA-3.2-1B	$\mathbf{0.92}_{\pm 0.010}$	$\mathbf{0.95}_{\pm 0.008}$	$\mathbf{0.88}_{\pm 0.020}$	$0.90_{\pm 0.010}$	$0.93_{\pm 0.009}$	$0.85_{\pm 0.021}$	$+0.02$
Mistral-7B	$\mathbf{0.90}_{\pm 0.011}$	$\mathbf{0.93}_{\pm 0.009}$	$\mathbf{0.84}_{\pm 0.021}$	$0.88_{\pm 0.011}$	$0.92_{\pm 0.009}$	$0.80_{\pm 0.022}$	$+0.02$
Gemma-2B	$\mathbf{0.89}_{\pm 0.011}$	$\mathbf{0.92}_{\pm 0.009}$	$\mathbf{0.81}_{\pm 0.022}$	$0.87_{\pm 0.012}$	$0.91_{\pm 0.010}$	$0.78_{\pm 0.023}$	$+0.02$
Mean	**0.90**	**0.93**	**0.84**	0.88	0.92	0.81	**+0.02**

the controlled audit protocol. This comparison isolates the effect of observability while keeping the data, models, and scoring rule fixed.

As expected, per-client visibility improves detection modestly because the auditor can localize suspicious updates directly, but the aggregate-only regime remains strong. This supports the intended deployment claim: under secure aggregation, FedDataLeak still provides useful *cohort-level* leakage detection, even though client-level attribution is unavailable. In Table 13, per-client visibility yields a small but consistent gain because the auditor can score each client update directly. However, the aggregate-only regime remains strong, which supports FedDataLeak's use as a *cohort-level* privacy auditor under secure aggregation.

Why aggregate FRF remains informative. The aggregate-only score remains discriminative because averaging does not necessarily erase reconstructable signal; it mainly removes client attribution. In our controlled audit setting, positive rounds are sparse and often contain one high-signal client update, e.g., a canary-bearing or unusually specialized local batch. For an aggregate update

$$\Delta_t = \sum_{i \in C_t} p_i \Delta_t^{(i)}, \tag{51}$$

the FRF solver only needs to find a candidate sequence whose induced update explains a nontrivial component of Δ_t, not every client contribution. When one client contributes a large aligned embedding or PEFT-module signal, while the remaining client updates are less aligned or partially cancel, the dominant residual direction is still detectable by update matching. Thus, aggregate-only FRF should be interpreted as cohort-level evidence that at least one participating update is reconstructable, rather than as client-level attribution.

D.3 Score calibration of leakage risk score

FedDataLeak is intended not only as a ranking signal but also as a practical *risk score* for triggering mitigation or deeper inspection. To verify this interpretation, we evaluate calibration on a held-out audit split of **Medical Abstract** with **LLaMA-3.2-1B**. We bin the deployed leakage risk score (LRS) into ten equal-width bins on $[0, 1]$ and compute the empirical leakage frequency within each bin. We additionally visualize the score distribution for positive and negative audit instances together with the operating thresholds used by the tiered auditor.

As shown in Fig. 6, the empirical leakage frequency increases monotonically with LRS, indicating that larger scores correspond to larger observed privacy risk. The score histogram further shows that negative audit instances concentrate in the low-score region, while positive instances shift toward the high-score region. This supports the use of FedDataLeak as a deployable risk-scoring mechanism rather than only a relative ranking function.

D.4 Cheap update-only sanity-check monitors

A natural concern is that FedDataLeak may simply detect large or atypical updates rather than leakage-specific reconstructability. To rule out this explanation, we compare against three cheap

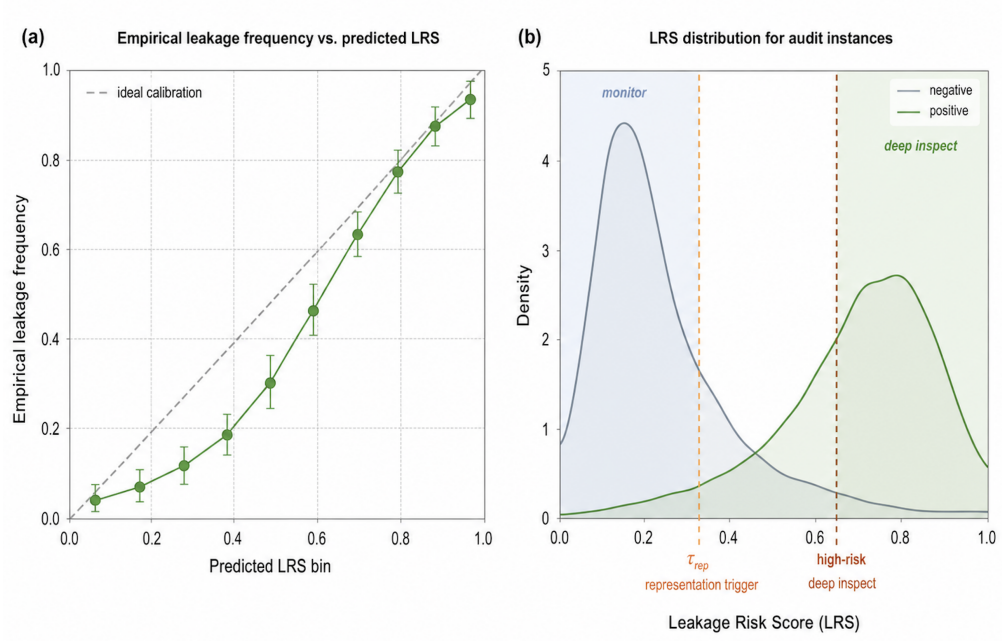


Figure 6: Score calibration of FedDataLeak. (**Left**) Reliability diagram for the deployed leakage risk score (LRS) on Medical Abstract with LLaMA-3.2-1B. The empirical leakage frequency increases monotonically with the predicted risk score, supporting the interpretation of LRS as a practical audit score. (**Right**) Distribution of LRS for leakage-positive and leakage-negative audit instances, with vertical markers for the representation-trigger threshold τ_{rep} and the high-risk region used for deeper inspection. The separation between the two distributions shows that FedDataLeak assigns systematically higher scores to higher-risk updates.

Table 14: Cheap update-only sanity-check baselines on Medical Abstract with LLaMA-3.2-1B. All methods assign one scalar risk score to each observed update. The sanity monitors require no reconstruction and negligible compute, but substantially underperform FedDataLeak, showing that the proposed score is not merely detecting update magnitude, directional outliers, or clipping saturation. Values denote mean $\pm$ standard deviation over three seeds.

Method	**What it tests**	AUPRC ↑	AUROC ↑	TPR@1%FPR ↑	ECR ↑	Audit Cost/Round (s) ↓
Update norm $s_{\text{norm}}(u) = \|u\|_2$	update magnitude only	$0.58_{\pm 0.013}$	$0.66_{\pm 0.011}$	$0.37_{\pm 0.023}$	N/A	$\mathbf{0.2}_{\pm \mathbf{0.1}}$
Cosine-to-mean $s_{\cos}(u) = \cos(u, \bar{u}_{\text{ref}})$	directional outlier score	$0.63_{\pm 0.012}$	$0.70_{\pm 0.010}$	$0.43_{\pm 0.024}$	N/A	$0.3_{\pm 0.1}$
Clipping saturation $s_{\text{clip}}(u)$	clipping-threshold exceedance	$0.55_{\pm 0.014}$	$0.63_{\pm 0.012}$	$0.34_{\pm 0.025}$	N/A	$\mathbf{0.2}_{\pm \mathbf{0.1}}$
Best cheap monitor	max over CDF-calibrated cheap scores	$0.65_{\pm 0.012}$	$0.72_{\pm 0.011}$	$0.46_{\pm 0.024}$	N/A	$0.4_{\pm 0.1}$
FRF-only	update-matching reconstructability	$0.84_{\pm 0.012}$	$0.88_{\pm 0.010}$	$0.70_{\pm 0.022}$	$0.66_{\pm 0.016}$	$7.1_{\pm 0.4}$
FedDataLeak	FRF + PI/sparsity triage	$\mathbf{0.90}_{\pm \mathbf{0.010}}$	$\mathbf{0.93}_{\pm \mathbf{0.009}}$	$\mathbf{0.85}_{\pm \mathbf{0.021}}$	$\mathbf{0.73}_{\pm \mathbf{0.014}}$	$8.9_{\pm 0.5}$

Color guide. sanity-check / FRF-only control; best cheap monitor; FedDataLeak; best value.

update-only monitors that require no reconstruction and negligible extra compute: the update norm, cosine similarity to a running non-alert mean update, and clipping saturation. The norm monitor tests whether leakage labels are explained by update magnitude alone; the cosine-to-mean monitor tests whether leakage corresponds to generic directional outliers; and the clipping monitor tests whether standard FL clipping statistics are sufficient to detect leakage-prone updates. These controls are evaluated under the same Medical Abstract, LLaMA-3.2-1B protocol and the same canary-defined leakage labels used in the main ablation. As shown in Table 14, the cheap monitors are useful operational diagnostics but are not sufficient leakage auditors: they detect some high-magnitude or atypical updates, yet miss many updates that are reconstructable under the fixed FRF audit budget. FedDataLeak improves over the best cheap monitor by $+0.25$ AUPRC and $+0.39$ TPR@1%FPR, supporting that its gains come from update-conditioned reconstructability and representation-shift triage rather than update magnitude alone.

D.5 Cross-model and cross-dataset PEFT leakage map

Because modern federated LLM fine-tuning often updates only a small PEFT subspace, we evaluate whether FedDataLeak's leakage signal depends on which trainable module is audited. We compare four PEFT update regimes: *Embeddings-only*, where token-row updates provide direct token identity evidence; *LoRA-only*, where leakage is routed through adapter matrices without direct embedding-row access; *Mixed*, where both embedding and LoRA updates are audited using the module-balanced objective from Appendix B.2; and *FFA-LoRA-style*, a contextual FL-specific LoRA setting inspired by Federated Freeze-A LoRA, where only one LoRA factor is updated while the other is fixed. The FFA-LoRA-style row is not treated as a competing leakage detector; rather, it tests whether FedDataLeak remains applicable when FL-specific LoRA training constrains the communicated adapter subspace.

Table 15 reports AUPRC across all dataset-model pairs, and Fig. 7 visualizes the same values as a heatmap. The trend is consistent with the module semantics in Appendix B.1: embeddings provide the strongest token-local signal, LoRA-only updates are harder to invert directly, and mixed auditing recovers the most stable risk signal across domains and model families. FFA-LoRA-style updates slightly reduce audit recoverability relative to vanilla LoRA, consistent with the smaller communicated trainable subspace, but FedDataLeak remains rank-informative. In Table 15 Embeddings-only auditing is consistently stronger than LoRA-only auditing because embedding-row updates expose more direct token-local evidence Sun et al. (2024). LoRA-only remains informative but weaker because the update signal is mediated through adapter matrices. Mixed auditing is the most stable setting because it captures both token-local and adapter-mediated leakage pathways. FFA-LoRA-style updates reduce the communicated trainable subspace and therefore slightly reduce recoverability, but the auditor remains effective.

D.6 Defense sensitivity under clipping and light noise

Because FedDataLeak is intended as a deployment-time auditor, we evaluate how its risk scores behave under simple mitigation knobs that are commonly used in federated training: update clipping and light Gaussian perturbation. This experiment is distinct from the client-recipe mismatch study in Table 6: here the client update itself is modified before aggregation, whereas Table 6 changes only the auditor's simulated recipe. For each setting, thresholds are recalibrated on a disjoint audit-validation split using the same target-FPR rule as Appendix B.5. The Gaussian-noise rows should be interpreted as DP-style sensitivity checks rather than formal differential privacy guarantees, since we do not

Table 15: Cross-model and cross-dataset PEFT leakage map. We report AUPRC for FedDataLeak under different audited trainable subsets. Embeddings-only directly audits token-row updates; LoRA-only audits adapter updates; Mixed audits both embeddings and LoRA using the module-balanced objective; and FFA-LoRA-style approximates a federated LoRA setting where one LoRA factor is frozen and only the zero-initialized factor is communicated. Values denote mean ± standard deviation over three seeds.

Data	PEFT setting	LLaMA-3.2-1B			Mistral-7B			Gemma-2B		
		AUPRC ↑	FRF-cos ↑	Cost ↓	AUPRC ↑	FRF-cos ↑	Cost ↓	AUPRC ↑	FRF-cos ↑	Cost ↓
Med.	Embeddings-only	0.88±0.011	0.790±0.011	6.9±0.4	0.86±0.011	0.774±0.012	7.4±0.4	0.84±0.012	0.755±0.012	6.7±0.4
	LoRA-only	0.85±0.012	0.752±0.013	9.8±0.5	0.83±0.012	0.736±0.014	10.5±0.5	0.81±0.013	0.718±0.014	9.2±0.5
	Mixed Emb.+LoRA	0.89±0.010	0.801±0.010	10.6±0.6	0.87±0.011	0.786±0.011	11.2±0.6	0.85±0.012	0.768±0.012	10.0±0.5
	FFA-LoRA-style	0.82±0.013	0.724±0.014	7.6±0.4	0.80±0.013	0.710±0.014	8.1±0.4	0.78±0.014	0.692±0.015	7.3±0.4
SO	Embeddings-only	0.86±0.011	0.772±0.012	6.8±0.4	0.84±0.012	0.756±0.012	7.3±0.4	0.81±0.012	0.733±0.013	6.6±0.4
	LoRA-only	0.82±0.013	0.731±0.014	9.6±0.5	0.80±0.013	0.715±0.014	10.3±0.5	0.77±0.014	0.692±0.015	9.0±0.5
	Mixed Emb.+LoRA	0.87±0.011	0.785±0.011	10.4±0.5	0.85±0.011	0.768±0.012	11.0±0.6	0.82±0.012	0.746±0.013	9.8±0.5
	FFA-LoRA-style	0.79±0.014	0.704±0.015	7.4±0.4	0.77±0.014	0.689±0.015	7.9±0.4	0.74±0.015	0.666±0.016	7.1±0.4
Enron	Embeddings-only	0.89±0.011	0.802±0.011	7.0±0.4	0.87±0.011	0.785±0.012	7.5±0.4	0.85±0.012	0.767±0.012	6.8±0.4
	LoRA-only	0.86±0.012	0.763±0.013	9.9±0.5	0.84±0.012	0.746±0.014	10.6±0.5	0.82±0.013	0.728±0.014	9.3±0.5
	Mixed Emb.+LoRA	0.90±0.010	0.812±0.010	10.7±0.6	0.88±0.011	0.796±0.011	11.3±0.6	0.86±0.012	0.778±0.012	10.1±0.5
	FFA-LoRA-style	0.83±0.013	0.734±0.014	7.7±0.4	0.81±0.013	0.719±0.014	8.2±0.4	0.79±0.014	0.702±0.015	7.4±0.4

Dataset abbreviations. Med. = Medical Abstract; SO = Stack Overflow; Enron = Enron Emails.

Color guide. embedding-audited PEFT; LoRA-audited PEFT; mixed embedding–LoRA audit; FL-specific LoRA context.

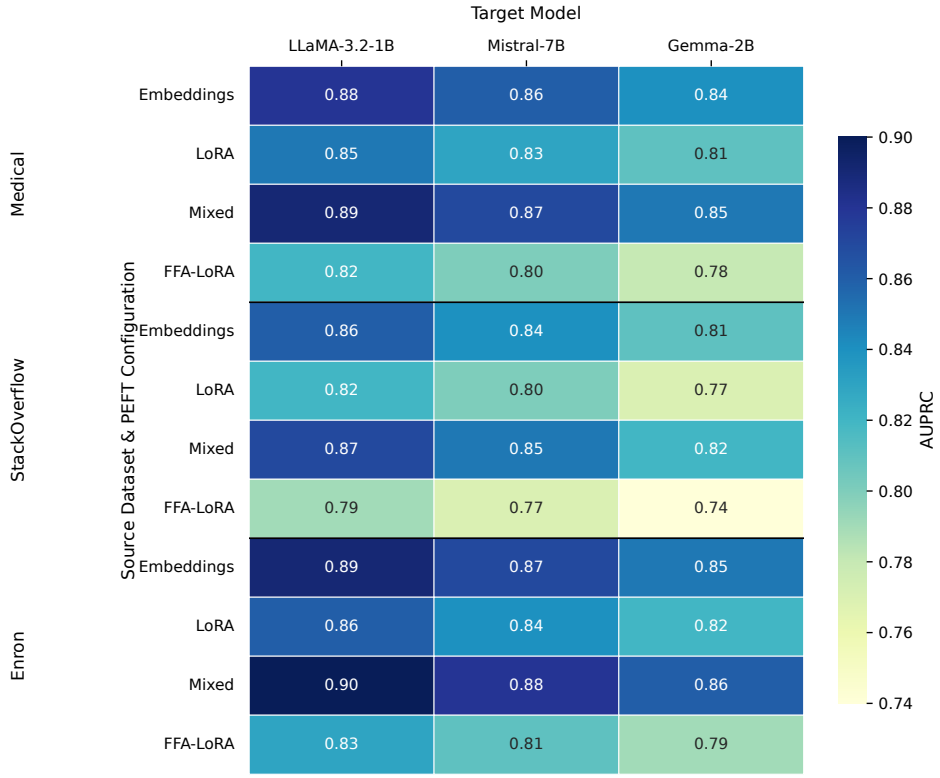


Figure 7: **Cross-model and cross-dataset PEFT leakage heatmap.** Cells show AUPRC for FedDataLeak under different audited trainable subsets. Embedding-audited updates expose the strongest token-local signal, LoRA-only updates are weaker but still informative, and mixed embedding-LoRA auditing is the most stable across datasets and model families. FFA-LoRA-style updates are included as a contextual FL-specific LoRA setting rather than a separate leakage detector.

report a privacy accountant or ε value. In Table 16, Clipping and light noise reduce reconstructable evidence, reflected by lower ECR and FRF-cos. However, FedDataLeak remains rank-informative after threshold recalibration, with AUPRC remaining above 0.85 for mild clipping and light-noise settings. The stronger-noise row shows the expected degradation: privacy evidence decreases, but so does the reliability of update-conditioned auditing because the transmitted update no longer matches the client training signal as closely. Figure 8 further evaluates FedDataLeak under standard mitigation knobs, including update clipping and light Gaussian noise, showing that leakage decreases as defenses strengthen while the auditor remains rank-informative after threshold recalibration.

Table 16: Defense sensitivity on Medical Abstract with LLaMA-3.2-1B. We apply clipping and light Gaussian perturbations to the communicated update before auditing. Stronger defenses reduce exact recovery and FRF evidence, as expected, while FedDataLeak remains useful as a calibrated risk-ranking signal. Values denote mean $\pm$ standard deviation over three seeds.

Defense setting	**Update rule**	AUPRC ↑	AUROC ↑	TPR@1%FPR ↑	ECR ↑	FRF-cos ↑	Audit Cost/Round (s) ↓
No defense	$\tilde{u} = u$	**0.90**$\pm$0.010	**0.93**$\pm$0.009	**0.85**$\pm$0.021	**0.73**$\pm$0.014	**0.804**$\pm$0.010	8.9$\pm$0.5
Mild clipping	$\tilde{u} = u \cdot \min(1, C/\|u\|_2),\ C = 1.0$	0.88$\pm$0.011	0.91$\pm$0.010	0.80$\pm$0.022	0.67$\pm$0.015	0.768$\pm$0.011	8.7$\pm$0.5
Moderate clipping	$\tilde{u} = u \cdot \min(1, C/\|u\|_2),\ C = 0.5$	0.84$\pm$0.012	0.88$\pm$0.011	0.72$\pm$0.024	0.58$\pm$0.017	0.704$\pm$0.013	8.3$\pm$0.4
Light noise	$\tilde{u} = u + \mathcal{N}(0, \sigma^2 C^2 I),\ \sigma = 0.01$	0.87$\pm$0.012	0.90$\pm$0.010	0.78$\pm$0.023	0.64$\pm$0.016	0.742$\pm$0.012	8.9$\pm$0.5
Clipping + light noise	$C = 1.0,\ \sigma = 0.01$	0.85$\pm$0.012	0.89$\pm$0.011	0.74$\pm$0.024	0.60$\pm$0.017	0.716$\pm$0.013	8.6$\pm$0.5
Clipping + stronger noise	$C = 1.0,\ \sigma = 0.05$	0.79$\pm$0.014	0.84$\pm$0.012	0.62$\pm$0.026	0.45$\pm$0.019	0.612$\pm$0.015	8.4$\pm$0.5

Color guide. no defense; clipping; noise / DP-style perturbation; best value.

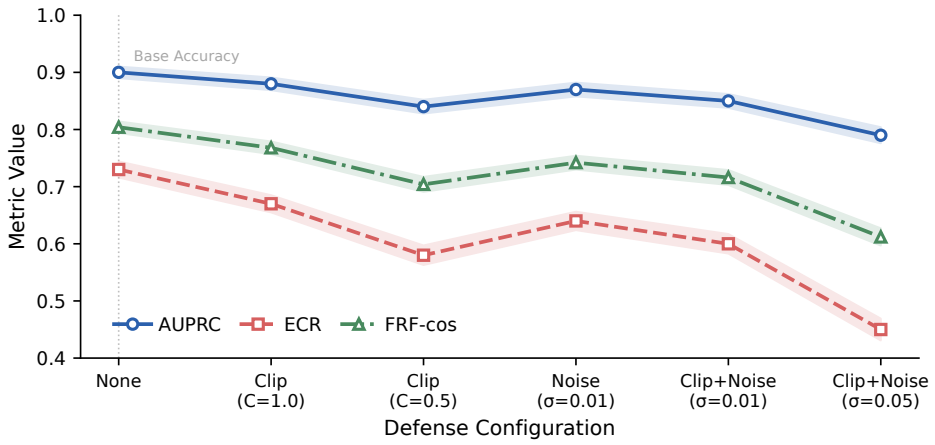


Figure 8: Defense sensitivity under clipping and light noise. Stronger mitigation reduces exact recovery and update-matching evidence, while FedDataLeak remains useful as a risk-ranking signal under mild clipping and light-noise settings after threshold recalibration.

E Operational Use, Mitigation Actions, and Failure Modes

FedDataLeak is designed to support a tiered operational response. Given a calibrated score threshold τ_{warn} and a higher intervention threshold τ_{stop} chosen on a validation audit set, an auditor can map scores to actions as follows. If $\text{Score}_t < \tau_{\text{warn}}$, the system continues training and logs the round for longitudinal monitoring. If $\tau_{\text{warn}} \leq \text{Score}_t < \tau_{\text{stop}}$, the system can apply soft mitigations such as stronger update clipping, additional calibrated noise, lower local learning rate, fewer local steps, or freezing leakage-prone modules such as embeddings or selected LoRA adapters. If $\text{Score}_t \geq \tau_{\text{stop}}$, the system can quarantine the update, roll back the round, trigger deeper reconstruction analysis on synthetic canary evidence, or pause training for human review. These responses should be interpreted as risk-management controls: clipping and noise may reduce leakage but can harm convergence and downstream utility; freezing embeddings or adapters can reduce token-level leakage pathways but may limit adaptation; stopping or rolling back training gives the strongest protection but has the largest operational cost.

Several failure modes should be considered when deploying the auditor. Adaptive clients may attempt to evade detection by distributing sensitive information across many low-magnitude coordinates, injecting benign-looking update noise, or shaping updates to reduce PI/sparsity shifts while preserving leakage through other modules. DP-SGD and clipping can reduce reconstructability, but they also distort the update distribution, so all thresholds must be recalibrated under the same defense configuration. FRF may also degrade when the auditor's assumed client recipe differs from the true optimizer, local-step count, learning rate, batch size, or PEFT subset; for this reason, robustness sweeps over recipes should be reported alongside the main results. Finally, when only aggregate updates are visible under secure aggregation, FedDataLeak can still flag suspicious rounds, but it cannot identify the responsible client without additional trusted logging, secure enclaves, or protocol-level attribution mechanisms.