BEYOND COORDINATE SPARSITY: CONTROLLING MODULE EXPOSURE AGAINST GRADIENT RECONSTRUCTION IN FEDERATED LORA

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated Low-Rank Adaptation (LoRA) reduces the communication cost of language-model fine-tuning, yet shared LoRA updates can still expose private training text to gradient-based reconstruction attacks. Existing sparsification methods primarily control how many update coordinates remain visible, but this does not capture which attack-relevant LoRA modules are exposed. We introduce *module exposure* to characterize this distinction and propose **MU-BEC**, a module-level upload-control method that uses public-data training dynamics to select a fixed subset of LoRA modules for communication. Under a matched MineGrad evaluation with the same audited attack base and 3,072 nominal retained slots, Top-k yields 95% unconditional content-token recovery, whereas MU-BEC yields 0% under both Fixed Attack and Known-Mask Attack. Mechanism analyses further show that aggressive Top-k sparsification can preserve nearly all target-gradient energy, indicating that coordinate count alone does not characterize MineGrad recovery in the evaluated setting. In normal federated training, MU-BEC reduces upload communication by 36.0% relative to Top-k and Random-k, while introducing a measurable utility trade-off. These results motivate module-level observability as a complementary perspective for analyzing privacy leakage in Federated LoRA.

1 INTRODUCTION

Federated learning (FL) enables multiple clients to collaboratively adapt a shared model without centralizing their raw data (McMahan et al., 2017; Kairouz & McMahan, 2021). For language models, parameter-efficient fine-tuning (PEFT) further reduces training and communication costs by updating only a small subset of parameters (Houlsby et al., 2019; Li & Liang, 2021; Lester et al., 2021). Low-Rank Adaptation (LoRA) is particularly attractive in this setting because it represents model adaptation through compact low-rank updates (Hu et al., 2021), and has increasingly been adopted in federated language-model training (Sun et al., 2024; Guo et al., 2025; Yan et al., 2025). However, reducing the number of trainable or communicated parameters does not necessarily reduce the private information contained in the updates that remain visible to the server.

Gradient inversion attacks have shown that shared model updates can reveal client training data (Zhu et al., 2019; Geiping et al., 2020; Yin et al., 2021), including private text in language-model settings (Deng et al., 2021; Gupta et al., 2022; Petrov et al., 2024). This leakage also persists under parameter-efficient fine-tuning (Sami et al., 2025). More recently, MineGrad considers a stronger threat against Federated LoRA, where a malicious server manipulates the pretrained model and LoRA parameters such that private training information can be recovered from shared LoRA gradients (Sami et al., 2026). Importantly, its reconstruction relies on gradients associated with specific LoRA modules. This suggests that privacy-relevant observability in Federated LoRA may depend not only on how much of an update is visible, but also on which modules remain observable.

Existing communication-efficient FL methods primarily reduce update size through gradient sparsification, transmitting only a subset of gradient coordinates (Aji & Heafield, 2017; Stich et al., 2018; Alistarh et al., 2018). Prior work further studies error compensation and optimization under compressed updates (Lin et al., 2017; Karimireddy et al., 2019). These approaches naturally characterize

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.

an upload by the number of coordinates that remain visible. However, coordinate count alone does not capture the structural origin of those observations. In the MineGrad setting, pruning 99% of gradient elements can leave reconstruction largely unaffected when attack-relevant LoRA gradients are preserved (Sami et al., 2026). Consequently, two equally sparse uploads may retain coordinates from different LoRA modules and expose substantially different information to the attacker. This reveals an overlooked distinction between *how many* update values are visible and *which modules* those values expose. We capture this distinction through *module exposure*, which describes which attack-relevant LoRA modules remain observable to the server. While coordinate sparsity measures how many update values are visible, module exposure captures where those values come from. This leads to our central question: *under the same observable-coordinate budget, does module exposure affect MineGrad recovery?*

To study this question, we propose **Module-Update-Based Exposure Control (MU-BEC)**, a module-level upload-control method for Federated LoRA. MU-BEC uses public-data training dynamics to identify a fixed subset of LoRA modules and restricts client uploads to the corresponding parameter regions. It therefore provides a simple mechanism for controlling which modules remain observable while leaving local LoRA training and server aggregation unchanged. We evaluate MU-BEC through both privacy-recovery analysis and normal federated training. Under a matched MineGrad comparison with the same audited attack base and 3,072 visible slots, Top-k retains 95% unconditional content-token recovery, whereas MU-BEC yields 0% under both Fixed Attack and Known-Mask Attack. The contrast under an identical visible-coordinate budget shows that coordinate count alone does not explain MineGrad recovery in our evaluated setting. We further evaluate MU-BEC in normal Federated LoRA training and quantify the communication-utility trade-off introduced by module-level upload control. Our contributions are summarized as follows:

- We introduce *module exposure* to characterize which attack-relevant LoRA modules remain observable to the server, distinguishing module-level observability from conventional coordinate sparsity.
- We propose **MU-BEC**, a module-level upload-control method that uses public-data training dynamics to construct a fixed LoRA-B upload mask, enabling explicit control over module exposure without modifying local LoRA training or server aggregation.
- We conduct a matched-observation evaluation that controls both the audited attack base and visible-coordinate budget. The results show substantially different MineGrad recovery despite the same number of visible slots, and we further quantify the communication–utility trade-off of module-level upload control under normal Federated LoRA training.

2 Related Work

2.1 Federated LoRA and PEFT Parameter Selection

Parameter-efficient fine-tuning (PEFT) adapts pretrained models by updating a small set of task-specific parameters (Houlsby et al., 2019; Li & Liang, 2021; Lester et al., 2021). LoRA represents these adaptations through low-rank updates (Hu et al., 2021) and has been increasingly studied in federated settings, where recent work explores selective training, sharing, protection, and aggregation of LoRA parameters (Sun et al., 2024; Guo et al., 2025; Yan et al., 2025; Meng et al., 2026; Liu et al., 2026b;a). Related PEFT methods further study non-uniform allocation and selection of LoRA parameters. AdaLoRA allocates rank budgets using training-time importance scores (Zhang et al., 2023), while DyLoRA, SoRA, RoseLoRA, and LoRA-drop explore dynamic ranks, sparsity, and structured parameter selection (Valipour et al., 2023; Ding et al., 2023; Wang et al., 2024; Zhou et al., 2025). Update magnitude has also been studied as a signal of LoRA training dynamics (Zhang et al., 2025). These works motivate non-uniform treatment of LoRA modules, but do not study how module selection affects their exposure to gradient-based reconstruction attacks. Our work instead uses public-data training dynamics to control which LoRA modules remain observable in Federated LoRA uploads.

2.2 Gradient Inversion and LoRA Information Leakage

Gradient inversion attacks demonstrate that shared gradients can reveal private training examples (Zhu et al., 2019; Geiping et al., 2020; Yin et al., 2021). This threat extends to language models,

where TAG, Recovering Private Text, and DAGER investigate reconstruction of private text from shared gradients (Deng et al., 2021; Gupta et al., 2022; Petrov et al., 2024). Leakage also persists under parameter-efficient fine-tuning (Sami et al., 2025). MineGrad considers a stronger Federated LoRA threat in which a malicious server manipulates the pretrained model and LoRA parameters to enable private-text recovery from shared LoRA gradients (Sami et al., 2026). Its reconstruction relies on gradients associated with specific LoRA modules, motivating our focus on *module exposure*. Rather than developing another reconstruction attack, we study whether controlling which modules remain observable changes MineGrad recovery under a fixed visible-coordinate budget.

2.3 Gradient Sparsification and Structured Communication

Gradient sparsification reduces communication by transmitting only a subset of update coordinates. Prior work studies magnitude-based sparsification, k-sparsification, and its convergence properties (Aji & Heafield, 2017; Stich et al., 2018; Alistarh et al., 2018), while error feedback and deep gradient compression address optimization under compressed updates (Karimireddy et al., 2019; Lin et al., 2017). Adaptive and layer-wise methods further distribute communication budgets across model components according to gradient statistics or layer-level behavior (Han et al., 2020; Shi et al., 2020; Hong & Han, 2021). These methods primarily determine which coordinates are transmitted and how sparsification affects optimization. However, coordinate sparsity does not specify which LoRA modules the server can observe. This distinction matters for MineGrad, where aggressive coordinate pruning can preserve reconstruction when attack-relevant gradients remain available (Sami et al., 2026). MU-BEC therefore focuses on module-level exposure control rather than introducing a general-purpose gradient sparsification method.

3 Method

3.1 Problem Formulation and Design Motivation

We consider Federated LoRA, where clients perform local LoRA training and upload their updates to a server for aggregation. For LoRA module m, let $\mathbf{A}_m$ and $\mathbf{B}_m$ denote the low-rank matrices, with rank r_m and scaling factor α_m. The corresponding LoRA weight update is

$$\Delta \mathbf{W}_m = \frac{\alpha_m}{r_m} \mathbf{B}_m \mathbf{A}_m. \tag{1}$$

Our focus is the information exposed through these uploaded LoRA updates. Following the MineGrad threat setting (Sami et al., 2026), private-text recovery depends on gradients associated with specific LoRA modules. Let $\mathcal{M}_{\text{att}}$ denote the attack-relevant modules and $\mathbf{G}_m$ the observable LoRA-B gradient associated with module m. We define *module exposure* as

$$\mathcal{E}_{\text{att}}(\mathbf{G}) = \{m \in \mathcal{M}_{\text{att}} : \|\mathbf{G}_m\|_0 > 0\}. \tag{2}$$

This definition highlights a limitation of coordinate-level sparsity: two uploads with the same number of visible coordinates can expose different LoRA modules. Therefore, reducing the number of transmitted coordinates does not necessarily reduce the attack-relevant observations available to the server. This observation motivates two design requirements. First, upload control should operate at the module level so that the set of observable LoRA modules can be explicitly controlled. Second, module selection should not rely on clients' private data or attack outcomes. We therefore construct the module selection before federated training using disjoint public data, and keep the resulting selection fixed throughout federated training.

3.2 Overview of MU-BEC

Based on these requirements, we propose **Module-Update-Based Exposure Control (MU-BEC)**. As illustrated in Figure 1, MU-BEC consists of a one-time preparation stage followed by module-level upload control during federated training. Before federated training, MU-BEC performs standard LoRA training on public data and measures the normalized update magnitude of each candidate module. Given a module budget, it selects the highest-scoring modules and maps the selection to a fixed binary mask over the corresponding LoRA-B parameters. During federated training, clients

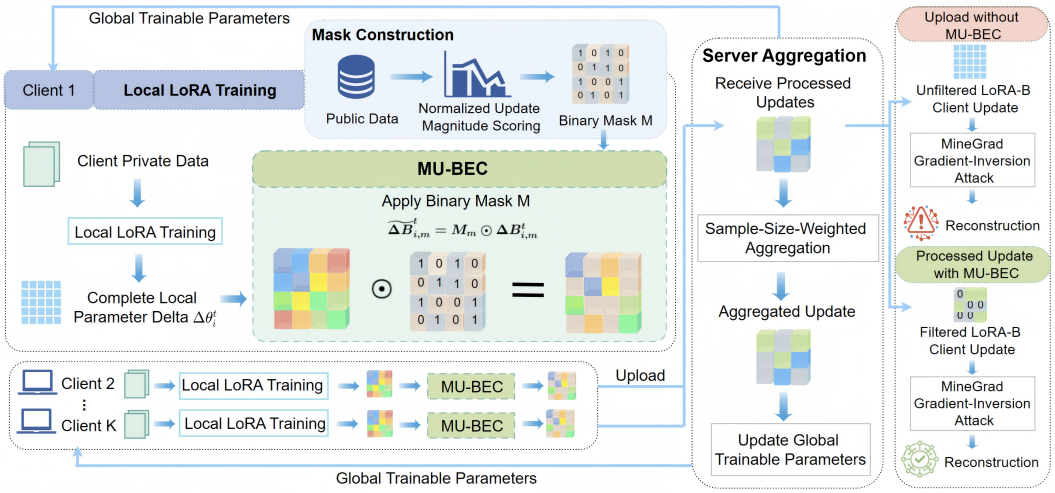

Figure 1: Overview of MU-BEC. A one-time preparation stage uses public data to score candidate LoRA modules and construct a fixed binary mask. During federated training, clients perform standard local LoRA training and apply the mask to the corresponding LoRA-B updates before upload.

perform standard local LoRA training and apply this mask to their local LoRA-B updates before upload. The processed updates are then aggregated using the standard server rule. MU-BEC therefore changes only which LoRA-B parameter regions are included in client uploads. It does not modify local LoRA optimization or server aggregation, and the fixed mask requires no per-round mask transmission.

3.3 Module Update Scoring and Selection

Let $\mathcal{M}_{\text{cand}}$ denote the candidate LoRA modules and $\mathcal{D}_{\text{pub}}$ the public data used for module selection. The public data are sample-disjoint from clients' private training data but are assumed to be relevant to the same task. The public-data model uses a LoRA configuration compatible with that of the federated clients, allowing the selected modules to be mapped to the corresponding client-side LoRA parameters.

For each candidate module m, MU-BEC measures its normalized update magnitude during public-data LoRA training. Let $\mathcal{S}$ denote the set of training seeds, T_q the number of recorded optimizer steps under seed q, and $d_{\text{out},m}$ and $d_{\text{in},m}$ the dimensions of the corresponding weight matrix. We compute

$$U_m = \text{Median}_{q\in\mathcal{S}} \left[\frac{1}{T_q} \sum_{l=1}^{T_q} \frac{\left\| \Delta \mathbf{W}_m^{(q,l)} \right\|_F^2}{d_{\text{out},m} d_{\text{in},m}} \right]. \tag{3}$$

The squared Frobenius norm captures the magnitude of the LoRA weight update, while normalization by the matrix size makes scores comparable across candidate modules. We average the normalized magnitude across optimizer steps and take the median across seeds to reduce sensitivity to individual training runs.

Given a predefined module budget K, MU-BEC ranks the candidate modules by U_m and retains the K highest-scoring modules. The resulting selection is represented by a binary mask $\mathbf{M}$ aligned with the corresponding LoRA-B parameters. The score is used only for module selection; we do not interpret it as a direct measure of task utility or privacy risk.

3.4 Module-Level Exposure Control

At communication round t, client i first performs standard local LoRA training and obtains the local parameter delta $\Delta\boldsymbol{\theta}_i^t$. Let $\Delta\mathbf{B}_{i,m}^t$ denote its LoRA-B component associated with module m.

MU-BEC applies the fixed module mask before upload:

$$\widetilde{\Delta\mathbf{B}}_{i,m}^{t} = \mathbf{M}_m \odot \Delta\mathbf{B}_{i,m}^{t}, \tag{4}$$

where $\odot$ denotes element-wise multiplication. Selected modules retain their corresponding LoRA-B updates, whereas unselected regions are suppressed.

Applying the mask to all candidate LoRA-B components produces the processed update $\widetilde{\Delta\boldsymbol{\theta}}_i^t$, which is uploaded to the server and aggregated using the standard federated aggregation rule. Because $\mathbf{M}$ is constructed before federated training and remains fixed across clients and communication rounds, MU-BEC explicitly controls the module-level structure of client uploads without introducing per-round mask transmission.

We distinguish the quantities used in the privacy and normal-training evaluations. The privacy analysis considers observable LoRA-B gradients $\mathbf{G}_m$ under the MineGrad setting, whereas normal federated training applies MU-BEC to local LoRA-B parameter deltas $\Delta\mathbf{B}_{i,m}^t$. These quantities are evaluated separately and are not treated as equivalent observations.

3.5 Exposure and Efficiency Analysis

We briefly analyze MU-BEC in terms of MineGrad exposure, update preservation, and system overhead. Let $\mathcal{E}_{\text{att}}$ denote the exposed attack-relevant modules defined in Eq. (2). Since MineGrad recovery relies on gradients from its target LoRA modules (Sami et al., 2026), masking module m removes its corresponding observation whenever $\mathbf{M}_m = \mathbf{0}$. Hence, the attack-relevant observations available after masking are restricted to

$$\mathcal{E}_{\text{att}}^{\text{MU}} = \mathcal{M}_{\text{att}} \cap \mathcal{M}_{\text{sel}}, \tag{5}$$

where $\mathcal{M}_{\text{sel}}$ is the set of modules retained by MU-BEC. Therefore, unlike coordinate sparsification, MU-BEC directly bounds the set of MineGrad-relevant modules exposed to the server. In particular, if $\mathcal{M}_{\text{att}} \cap \mathcal{M}_{\text{sel}} =$, the corresponding MineGrad module observations are removed. This does not constitute a general privacy guarantee, but characterizes the reduction in attack-relevant exposure under the considered threat model.

For model utility, MU-BEC preserves the complete LoRA-B updates of selected modules rather than individual coordinates. The deviation from the original client update is therefore determined by the discarded module updates,

$$\left\|\Delta\boldsymbol{\theta}_i^t - \widetilde{\Delta\boldsymbol{\theta}}_i^t\right\|_2^2 = \sum_{m \notin \mathcal{M}_{\text{sel}}} \left\|\Delta\mathbf{B}_{i,m}^t\right\|_F^2, \tag{6}$$

for the masked LoRA-B components. Thus, retaining modules that carry substantial task-related updates limits the perturbation introduced by module masking, although MU-BEC does not guarantee the same utility as coordinate-level Top-k.

Finally, under the same number k of transmitted values, MU-BEC and Top-k have the same $O(k)$ value-transmission complexity. MU-BEC uses a fixed structured mask and therefore requires neither per-round Top-k coordinate selection nor coordinate-index transmission. Its additional scoring cost occurs only once on public data before federated training. MU-BEC consequently maintains a communication and runtime order comparable to Top-k while explicitly controlling module exposure.

4 Experiments

We evaluate MU-BEC through three research questions. **RQ1:** How does module-level exposure control affect MineGrad recovery compared with coordinate-level sparsification under a matched observation budget? **RQ2:** What utility-communication trade-off does module-level exposure control introduce in normal Federated LoRA training? **RQ3:** How do module selection and module budget affect the utility of MU-BEC?

4.1 Experimental Setup

Model and federated training. We use RoBERTa-base (Liu et al., 2019) on Yahoo Answers Topics (Zhang et al., 2015). LoRA is applied to the query, key, value, and attention-output projections of

all 12 Transformer layers, yielding 48 candidate modules. We use rank $r = 4$, scaling factor $\alpha = 8$, and zero LoRA dropout. Normal federated training uses 20 clients, 2,000 training samples, 1,000 validation samples, 70 communication rounds, and one local epoch per round. All clients participate in each round, and updates are aggregated by local sample count. We use AdamW with learning rate 2×10^{-4} and seeds $\{42, 43, 44, 45, 46\}$.

Module selection. MU-BEC constructs its mask before federated training using 2,000 public training and 1,000 public validation samples disjoint from the federated data. Scores are computed over all 48 candidate modules. MU-BEC Top-1 selects `encoder12.key`, while Top-2 additionally selects `encoder12.query`. Random Module-1 independently selects one candidate module for each evaluated run. Neither private target texts nor MineGrad recovery outcomes are used for module selection.

Privacy evaluation and metrics. We evaluate 20 independent target texts under Fixed Attack and Known-Mask Attack, treating each text as an independent statistical unit. Repeated Random-k masks for the same text are first aggregated within that text before statistics are summarized across texts. Known-Mask Attack reveals the retained support to the attacker, testing whether recovery degradation can be explained by support uncertainty alone. It is not intended as an exhaustive adaptive attack.

Content-token recovery is the primary reconstruction metric, with exact-sequence recovery reported as a complementary measure. We further report target nonzero retention, retained target-gradient energy, and cosine similarity to characterize the information preserved by each upload transformation. For normal-training and ablation experiments, the training seed is the independent run unit.

4.2 Module Exposure and MineGrad Recovery(RQ1)

We first compare module-level exposure control with coordinate-level sparsification under a matched observation budget. We then examine the structure of the retained observations and analyze why coordinate sparsification can preserve or alter MineGrad recovery.

Recovery under a matched observation budget. We compare Top-k and MU-BEC using the same audited attack base and a nominal retained-slot budget of 3,072 coordinates per upload. Top-k achieves 95% unconditional content-token recovery under both Fixed Attack and Known-Mask Attack, whereas MU-BEC Top-1 yields 0%. Among the baseline-success texts, Top-k preserves complete content-token recovery. Revealing the retained support to the Known-Mask Attack does not restore MU-BEC recovery in this configuration.

Thus, the same nominal coordinate budget can lead to substantially different MineGrad recovery. To understand this difference, we next separate the nominal budget from the structural and numerical properties of the retained observations. These results are specific to the evaluated attacks and do not establish robustness against arbitrary adaptive attackers.

Structure of the retained observations. Figure 2 distinguishes nominal retained slots, realized numerical nonzeros, mask-allowed module exposure, and selected module identities.

Top-k and MU-BEC Top-1 both use 3,072 nominal retained slots, but their realized observations differ. Top-k contains 3,072 numerical nonzeros and exposes four LoRA modules in the evaluated artifact. MU-BEC Top-1 allows one LoRA-B module through its mask, while the selected tensor contains zero numerical nonzero entries. MU-BEC Top-2 allows two modules and likewise contains zero numerical nonzeros in the selected tensors. Hence, nominal coordinate budget, module exposure, and realized numerical signal capture different properties of the uploaded observation. The recovery difference should therefore be interpreted in terms of the resulting observation structure rather than module identity alone.

Coordinate-sparsification mechanisms. Table 1 summarizes two complementary mechanism checks. Global Top-k tests whether removing most coordinates also removes the dominant target-gradient signal, while the Rand-k rescaling ablation tests whether Fixed-Attack degradation is determined by retained support alone.

At 99% sparsity, Global Top-k retains only 24.01% of the target nonzero coordinates but preserves 99.9906% of the target gradient energy, with cosine similarity 0.99995. Content-token recovery therefore remains unchanged on the baseline-success texts. This result indicates that removing many

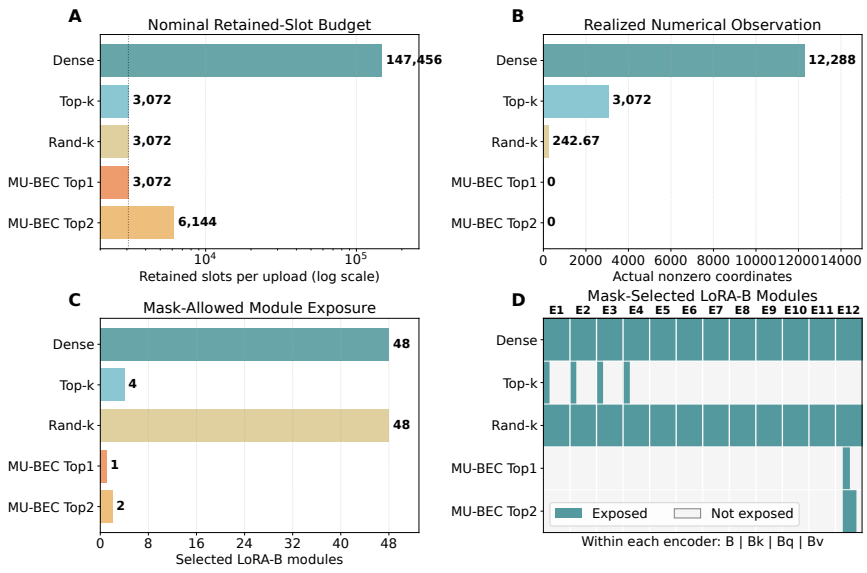

Figure 2: Structure of the privacy-evaluation observations. (A) Nominal retained-slot budget. (B) Actual nonzero upload coordinates. (C) Mask-allowed module exposure. (D) LoRA-B modules selected by each method.

Table 1: Mechanism analysis of coordinate sparsification. Conditional recovery is computed over the 19 baseline-success texts; unconditional recovery uses all 20 texts. Rand-k+d/k is a custom rescaling ablation applied to the same retained support.

Condition	Sparsity	Content Rec. Cond. ↓	Content Rec. Uncond. ↓	Target NZ Ret.	Energy Ret.	Cosine
Dense	0%	$100.00 \pm 0.00\%$	$95.00 \pm 22.36\%$	100.00%	1.0000	1.0000
Top-k	99%	$100.00 \pm 0.00\%$	$95.00 \pm 22.36\%$	24.01%	0.999906	0.99995
Rand-k	75%	$53.18 \pm 8.95\%$	$50.52 \pm 14.74\%$	25.06%	0.2377	0.4838
Rand-k+d/k	75%	$99.27 \pm 1.79\%$	$94.30 \pm 22.27\%$	25.06%	3.8035	0.4838

coordinates need not remove the dominant MineGrad-relevant signal when that signal is concentrated in the retained coordinates.

Rand-k provides a complementary mechanism check. At 75% sparsity, unscaled Rand-k yields $53.18 \pm 8.95\%$ conditional content-token recovery. Applying d/k rescaling to the same retained support raises recovery to $99.27 \pm 1.79\%$, while target nonzero retention remains 25.06% and cosine similarity remains 0.4838. The Fixed-Attack degradation is therefore sensitive to numerical scaling and should not be interpreted as evidence that coordinate removal alone eliminates the recoverable information.

RQ1 summary. Under the same nominal observation budget, Top-k and MU-BEC produce different observation structures and substantially different MineGrad recovery. The mechanism analyses further show that Top-k can preserve nearly all target-gradient energy despite high sparsity, while Random-k recovery is sensitive to numerical scaling. Together, these results show that coordinate count alone does not characterize MineGrad recovery in the evaluated setting and motivate explicit control over module-level exposure.

4.3 Utility–Communication Trade-off(RQ2)

We next evaluate the cost of module-level exposure control during normal Federated LoRA training. Unlike RQ1, these experiments operate on local parameter deltas rather than MineGrad gradient observations.

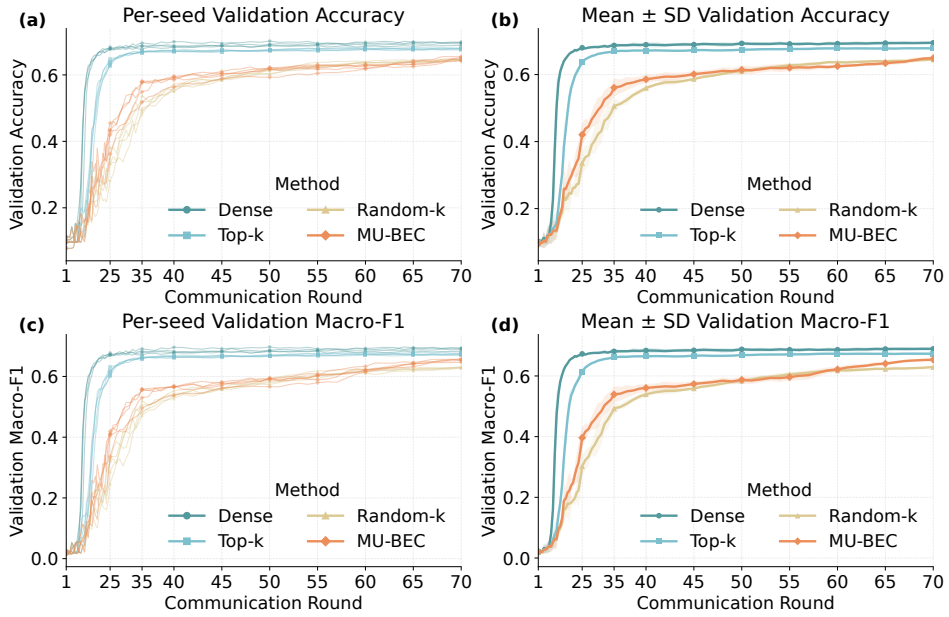

Figure 3: Normal-training validation trajectories over 70 communication rounds. Panels (a) and (c) show per-seed Validation Accuracy and Macro-F1; panels (b) and (d) show the corresponding mean±SD trajectories.

Table 2: Per-round upload communication under the stated encoding assumptions.

Method	Value slots/client	Value bits /round	Index/mask bits/round	Total bits /round
Dense	147,456	94,371,840	0	94,371,840
Top-k	3,072	1,966,080	1,105,920	3,072,000
Random-k	3,072	1,966,080	1,105,920	3,072,000
MU-BEC	3,072	1,966,080	0	1,966,080

Normal-training utility. Figure 3 reports validation trajectories over 70 communication rounds.

Dense and Top-k enter their late-round high-performance region earlier, whereas Random-k and MU-BEC improve more gradually. The mean±SD curves show the same overall ordering for Validation Accuracy and Macro-F1, while the per-seed trajectories indicate that this pattern is not driven by a single run. MU-BEC therefore incurs a measurable normal-training utility cost relative to Dense and Top-k in the evaluated configuration.

Communication cost. We assume 32 bits per transmitted parameter value. Top-k and Random-k additionally transmit an 18-bit coordinate index for each retained value, whereas the fixed MU-BEC mask requires no per-round coordinate-index or mask transmission.

Under this accounting, MU-BEC requires 1,966,080 upload bits per round, 36.0% fewer than Top-k and Random-k and 97.9% fewer than Dense. Relative to the coordinate-level sparse baselines, the reduction comes from eliminating 1,105,920 coordinate-index bits per round while retaining the same number of transmitted parameter values.

RQ2 summary. MU-BEC therefore introduces an explicit utility–communication trade-off. Its fixed module-level support removes the per-round index overhead of coordinate sparsification, but this communication reduction is accompanied by lower normal-training utility than Dense and Top-k in the evaluated setting.

4.4 Module Selection and Budget(RQ3)

Finally, we examine whether MU-BEC's normal-task utility depends on which modules are selected and how many modules are retained. We compare MU-BEC Top-1 and Top-2, which use

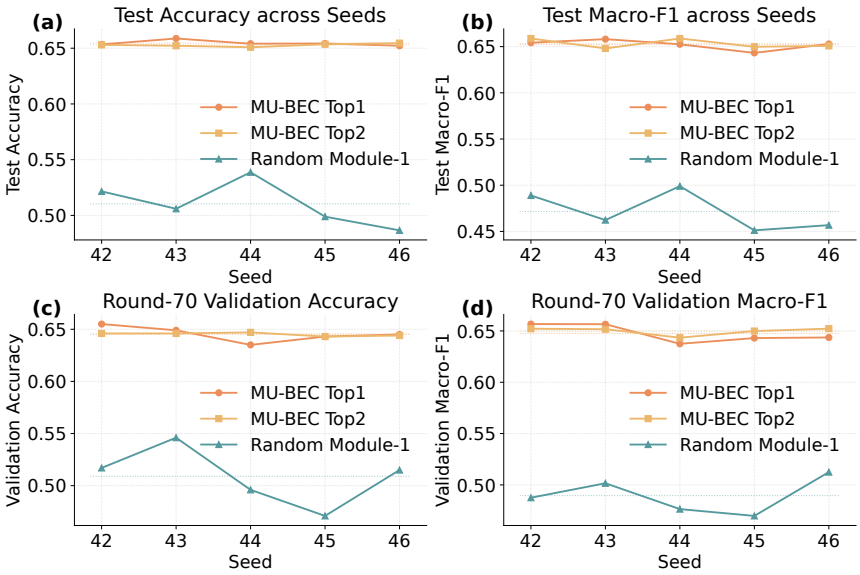

Figure 4: Per-seed module-selection and module-budget ablation for seeds 42–46. Panels (a) and (b) report Test Accuracy and Test Macro-F1; panels (c) and (d) report the corresponding round-70 validation metrics.

the same normalized update-magnitude scoring rule with one- and two-module budgets, against Random Module-1 under the same one-module budget.

Effect of module selection. MU-BEC Top-1 remains above Random Module-1 for every displayed seed in both Test Accuracy and Test Macro-F1, with the same qualitative separation in the round-70 validation metrics. Random Module-1 shows greater variation across runs. Because both the training seed and randomly selected module identity vary in this baseline, this spread should not be interpreted as pure seed variance. Nevertheless, under the same one-module budget, update-magnitude-based selection provides more consistent normal-task utility than random module selection in the evaluated runs.

Effect of module budget. MU-BEC Top-1 and Top-2 remain close across the evaluated seeds, and neither configuration uniformly dominates across all seeds and metrics. Increasing the budget from one to two modules therefore does not guarantee a per-seed utility improvement. Both score-based configurations remain above Random Module-1 across the displayed test results.

RQ3 summary. The ablation indicates that utility depends on the identity of the retained modules rather than on module count alone. The update-magnitude-based selection is consistently stronger than random one-module selection in the evaluated runs, while increasing the module budget from one to two does not uniformly improve utility.

5 Conclusion

We studied privacy-relevant observability in Federated LoRA from a module-level perspective. Our analysis shows that coordinate sparsity alone does not characterize MineGrad recovery: under the same nominal observation budget, different upload structures can lead to substantially different recovery behavior, while aggressive Top-k sparsification can preserve nearly all target-gradient energy. Motivated by this observation, we introduced *module exposure* and proposed MU-BEC, which uses public-data training dynamics to select a fixed subset of LoRA modules and explicitly control their exposure during federated training. Our experiments show that MU-BEC suppresses MineGrad recovery under the evaluated attacks and reduces communication overhead while introducing a measurable utility trade-off. These results suggest that, when analyzing privacy leakage in Federated LoRA, we should consider the structure of observable updates alongside their size. Future work will study stronger adaptive attacks, broader model and task settings, and module-selection strategies that better balance exposure control and task utility.

AI USE STATEMENT

Utilize AI tools to assist in refining language.

REFERENCES

Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pp. 440–445, 2017.

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. *Advances in Neural Information Processing Systems*, 31, 2018.

Jieren Deng, Yijue Wang, Ji Li, Chenghong Wang, Chao Shang, Hang Liu, Sanguthevar Rajasekaran, and Caiwen Ding. Tag: Gradient attack on transformer-based language models. In *Findings of the association for computational linguistics: EMNLP 2021*, pp. 3600–3610, 2021.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 4133–4145, 2023.

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.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. In *International Conference on Learning Representations*, volume 2025, pp. 99003–99027, 2025.

Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. Recovering private text in federated learning of language models. *Advances in neural information processing systems*, 35:8130–8143, 2022.

Pengchao Han, Shiqiang Wang, and Kin K Leung. Adaptive gradient sparsification for efficient federated learning: An online learning approach. In *2020 IEEE 40th international conference on distributed computing systems (ICDCS)*, pp. 300–310. IEEE, 2020.

Yuxi Hong and Peng Han. Lsddl: Layer-wise sparsification for distributed deep learning. *Big Data Research*, 26:100272, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Peter Kairouz and H Brendan McMahan. Advances and open problems in federated learning. *Foundations and trends in machine learning*, 14(1-2):1–210, 2021.

Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International conference on machine learning*, pp. 3252–3261. PMLR, 2019.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 3045–3059, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)*, pp. 4582–4597, 2021.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

Jianmin Liu, Li Yan, Borui Li, Lei Yu, and Chao Shen. She-lora: Selective homomorphic encryption for federated tuning with heterogeneous lora. In *International Conference on Learning Representations*, volume 2026, pp. 154701–154733, 2026a.

Jin Liu, Yinbin Miao, Ning Xi, and Junkang Liu. Rethinking lora for privacy-preserving federated learning in large models. *arXiv preprint arXiv:2602.19926*, 2026b.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

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.

Chuiyang Meng, Ming Tang, and Vincent Wong. Florg: Federated fine-tuning with low-rank gram matrices and procrustes alignment. In *International Conference on Learning Representations*, volume 2026, pp. 82804–82824, 2026.

Ivo Petrov, Dimitar I Dimitrov, Maximilian Baader, Mark N Müller, and Martin Vechev. Dager: Exact gradient inversion for large language models. *Advances in neural information processing systems*, 37:87801–87830, 2024.

Hasin Us Sami, Swapneel Sen, Amit K Roy-Chowdhury, Srikanth V Krishnamurthy, and Basak Guler. Gradient inversion attacks on parameter-efficient fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10224–10234, 2025.

Hasin Us Sami, Swapneel Sen, and Basak Guler. Minegrad: Gradient inversion attacks on lora fine-tuning. *arXiv preprint arXiv:2608.01521*, 2026.

Shaohuai Shi, Zhenheng Tang, Qiang Wang, Kaiyong Zhao, and Xiaowen Chu. Layer-wise adaptive gradient sparsification for distributed deep learning with convergence guarantees. In *ECAI 2020: 24th European Conference on Artificial Intelligence, 29 August–8 September 2020, Santiago de Compostela, Spain–Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, pp. 1467–1474. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2020.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in neural information processing systems*, 31, 2018.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. In *International conference on learning representations*, volume 2024, pp. 17978–17994, 2024.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. Dylora: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3274–3287, 2023.

Haoyu Wang, Tianci Liu, Ruirui Li, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. Roselora: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 996–1008, 2024.

Yunlu Yan, Chun-Mei Feng, Wangmeng Zuo, Lei Zhu, Rick Siow Mong Mong, and Yong Liu. Federated residual low-rank adaption of large language models. In *International Conference on Learning Representations*, volume 2025, pp. 57343–57367, 2025.

Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 16332–16341. IEEE, 2021.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zicheng Zhang, Haoran Li, Yifeng Zhang, Guoqiang Gong, Jiaxing Wang, Junxing Hu, Pengzhang Liu, and Qixia Jiang. The primacy of magnitude in low-rank adaptation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 39–69, 2025. doi: 10.52202/085713-0002.

Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, Tiejun Zhao, and Muyun Yang. LoRA-drop: Efficient LoRA parameter pruning based on output evaluation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5530–5543. Association for Computational Linguistics, 2025.

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