MULTI-ROUND SPEAR++: EXACT DATA RECONSTRUCTION FROM MODEL UPDATES ACROSS COMMUNICATION ROUNDS IN FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning trains a model over multiple communication rounds by sharing model updates instead of the clients' training data. For models with a fully connected first layer followed by ReLU, SPEAR++ can exactly reconstruct training inputs from a model update observed in a single communication round. However, whether an honest-but-curious attacker can exploit the same client's model updates across rounds to exactly reconstruct more inputs remains unexplored. In this paper, we formulate these multiple model updates as a single reconstruction problem and derive Multi-Round SPEAR++ from this formulation. Our formulation normalizes and concatenates first-layer weight updates from observed rounds that use the same inputs, yielding a reconstruction problem structurally similar to that of SPEAR++. Multi-Round SPEAR++ then adapts the ℓ_1-based reconstruction procedure used by SPEAR++ to the resulting formulation. On CIFAR-10 with 128 inputs per client, 1000 neurons per hidden layer, and one local update per round, Multi-Round SPEAR++ reconstructs all inputs for approximately 82% of evaluated clients using 10 rounds, whereas SPEAR++ using the first round almost never does. To further analyze these results, we theoretically characterize the strict local minimizers of the ℓ_1-minimization problem used by both methods and use this characterization to empirically analyze reconstruction failures. These results show that evaluating reconstruction risk using only a model update observed in a single communication round can underestimate the risk.

1 INTRODUCTION

Federated learning (McMahan et al., 2017) trains a model over multiple communication rounds in which clients share model updates with a server instead of their training data. Keeping training data local makes federated learning attractive for applications involving sensitive data (Kairouz et al., 2021; Zhao et al., 2025). Nevertheless, prior work has shown that shared model updates can leak training data through data reconstruction attacks under a practical honest-but-curious threat model (Zhu et al., 2019; Geiping et al., 2020; Carletti et al., 2025). To avoid underestimating the resulting reconstruction risk, we need to develop stronger data reconstruction attacks.

Notably, SPEAR (Dimitrov et al., 2024) and its extension SPEAR++ (Bakarsky et al., 2025) achieve exact reconstruction of input data from an observed model update, unlike most existing data reconstruction attacks, which yield only approximate reconstructions. SPEAR and SPEAR++ reconstruct the inputs using the first-layer weight update $\Delta \boldsymbol{W}$ contained in the observed model update. In their setting, the first layer is fully connected and followed by ReLU activation, and $\Delta \boldsymbol{W}$ factorizes as $\Delta \boldsymbol{W} = \boldsymbol{G}\boldsymbol{X}^\top$, where the columns of $\boldsymbol{X}$ are the inputs and the columns of $\boldsymbol{G}$ are their corresponding backpropagated gradients. Each column of $\boldsymbol{G}$ is sparse due to backpropagation through the first-layer ReLU. SPEAR++ exploits this sparsity via ℓ_1-minimization in recovering the factorization and thereby exactly reconstructs the inputs used to compute the observed model update. Empirically, the largest number of inputs that SPEAR++ can exactly reconstruct increases with the number of output neurons in the first layer.

SPEAR++ performs this reconstruction using a model update observed in a single communication round, whereas an honest-but-curious attacker can observe the same client's model updates across

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.

multiple communication rounds. This motivates the following question: *Can we exploit model updates observed across multiple communication rounds to exactly reconstruct more inputs given a fixed number of output neurons in the first layer?*

In this paper, we answer this question affirmatively by formulating model updates observed across multiple communication rounds as a single reconstruction problem and deriving Multi-Round SPEAR++ from this formulation. When the same inputs are used across the observed rounds, each first-layer weight update factorizes as $\Delta \boldsymbol{W}^{(t)} = \boldsymbol{G}^{(t)} \boldsymbol{X}^\top$, with a common input matrix $\boldsymbol{X}$ and a round-specific backpropagated-gradient matrix $\boldsymbol{G}^{(t)}$. Concatenating these updates yields a factorization with the same $\boldsymbol{X}$ and a backpropagated-gradient matrix with more rows, structurally similar to, though not equivalent to, the factorization obtained by increasing the number of output neurons in the first layer. This similarity, together with the empirical dependence of reconstruction on the number of output neurons, motivates us to reconstruct the inputs from the concatenated first-layer weight updates. The concatenated backpropagated-gradient matrix retains the ReLU-induced sparsity. As in SPEAR++, we exploit this sparsity via ℓ_1-minimization to recover the factorization and thereby reconstruct the inputs. Our key contributions are summarized below.

Multi-round reconstruction formulation (Section 3). We formulate model updates observed across multiple communication rounds as a single reconstruction problem using a normalized concatenated factorization and derive Multi-Round SPEAR++ from this formulation. The normalization prevents rounds with larger first-layer weight-update scales from dominating the ℓ_1-minimization.

Theoretical analysis of the ℓ_1-minimization (Section 4). We analyze the ℓ_1-minimizations used by SPEAR++ and Multi-Round SPEAR++ and derive necessary and sufficient conditions under which a point corresponding to an individual input is a strict local minimizer. These conditions provide a basis for empirically comparing the effects of exploiting model updates across multiple communication rounds and increasing the number of output neurons in the first layer.

Empirical validation (Section 5). On CIFAR-10 with 128 inputs per client, 1000 neurons per hidden layer, and one local update per round, Multi-Round SPEAR++ reconstructs all inputs for approximately 82% of evaluated clients using 10 rounds, whereas SPEAR++ using the first round almost never does.

2 PRELIMINARIES

This section introduces the notation used throughout the paper, the federated learning setup in Section 2.1, and the reconstruction procedure of SPEAR++ (Bakarsky et al., 2025) in Section 2.2.

Notation. For $k \in \mathbb{N}_{>0}$, let $[k] := \{1, \dots, k\}$. For a vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p$ ($p \in \{1, 2, \infty\}$) denotes its ℓ_p-norm, and $\|\boldsymbol{x}\|_0$ denotes the number of its nonzero entries. For a matrix $\boldsymbol{X}$, $\|\boldsymbol{X}\|_F$ denotes its Frobenius norm. For an index set $\mathcal{A}$, $\boldsymbol{x}_\mathcal{A}$ denotes the entries of $\boldsymbol{x}$ indexed by $\mathcal{A}$, and $\boldsymbol{X}_{\mathcal{A},:}$ denotes the submatrix of $\boldsymbol{X}$ formed by the rows indexed by $\mathcal{A}$. We denote element-wise multiplication of vectors by $\odot$. For a vector $\boldsymbol{x}$, $\mathbb{I}[\boldsymbol{x} > 0]$ and $\operatorname{sign}(\boldsymbol{x})$ denote the element-wise indicator and sign vectors, respectively, where $\mathbb{I}[\boldsymbol{x} > 0]_i = 1$ if $x_i > 0$ and 0 otherwise, and $\operatorname{sign}(x_i) = 1, 0, -1$ for $x_i > 0$, $x_i = 0$, and $x_i < 0$, respectively.

2.1 FEDERATED LEARNING

We consider federated learning with C clients. Each client $c \in [C]$ holds a local dataset $\mathcal{D}_c$ of input-label pairs $(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathbb{R}^n$ is the input and y its label. The clients collaboratively train a model with parameters $\boldsymbol{\theta}$ by minimizing $F(\boldsymbol{\theta}) := \sum_{c=1}^{C} p_c F_c(\boldsymbol{\theta})$ without sharing their local datasets. The local objective of client c is $F_c(\boldsymbol{\theta}) := \frac{1}{|\mathcal{D}_c|} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_c} \varphi(f(\boldsymbol{x};\boldsymbol{\theta}), y)$, and its weight is $p_c := |\mathcal{D}_c| / \sum_{c'=1}^{C} |\mathcal{D}_{c'}|$. The function φ denotes the per-sample loss, and $f(\boldsymbol{x};\boldsymbol{\theta})$ denotes the model output parameterized by $\boldsymbol{\theta}$.

To minimize $F(\boldsymbol{\theta})$, Federated Averaging (FedAvg) (McMahan et al., 2017) is widely used. FedAvg trains the model over multiple communication rounds. At each communication round $t \in [T]$, the server distributes the current model parameters $\boldsymbol{\theta}^{(t)}$ to all clients. Each client c initializes its local parameters as $\boldsymbol{\theta}_c^{(t,0)} = \boldsymbol{\theta}^{(t)}$ and performs K local update steps. The local updates and the resulting

model update are given by

$$\boldsymbol{\theta}_c^{(t,k)} = \boldsymbol{\theta}_c^{(t,k-1)} - \eta^{(t)} \nabla \widehat{F}_c^{(t,k)}(\boldsymbol{\theta}_c^{(t,k-1)}), \qquad \nabla \widehat{F}_c^{(t,k)}(\boldsymbol{\theta}) := \frac{1}{b} \sum_{(\boldsymbol{x},y)\in\mathcal{B}_c^{(t,k)}} \nabla_{\boldsymbol{\theta}} \varphi(f(\boldsymbol{x};\boldsymbol{\theta}), y), \quad (1)$$

$$\Delta\boldsymbol{\theta}_c^{(t)} = \boldsymbol{\theta}_c^{(t,K)} - \boldsymbol{\theta}^{(t)} = -\eta^{(t)} \sum_{k=1}^{K} \nabla \widehat{F}_c^{(t,k)}(\boldsymbol{\theta}_c^{(t,k-1)}). \quad (2)$$

Here, $k \in [K]$ denotes the local update step, $\eta^{(t)}$ is the learning rate at round t, and $\mathcal{B}_c^{(t,k)} \subseteq \mathcal{D}_c$ is a minibatch of size b. After performing the K local updates, each client c shares $\Delta\boldsymbol{\theta}_c^{(t)}$ with the server. The server then updates the model parameters as $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \sum_{c=1}^{C} p_c \Delta\boldsymbol{\theta}_c^{(t)}$. Throughout federated learning, the local datasets themselves are not shared, but data reconstruction attacks can reconstruct the inputs used to compute the shared model updates (Zhu et al., 2019).

2.2 SPEAR++: Exact Data Reconstruction Attack

Following recent work that separates input and label reconstruction, we focus on input reconstruction under an honest-but-curious threat model, where the attacker observes a client's shared model update $\Delta\boldsymbol{\theta}_c^{(t)}$ and the corresponding model $\boldsymbol{\theta}^{(t)}$ without intervening in the training procedure (Carletti et al., 2025). Under this threat model, SPEAR (Dimitrov et al., 2024) and SPEAR++ (Bakarsky et al., 2025), notably, achieve exact reconstruction of multiple inputs. We review SPEAR++ below as the basis for Multi-Round SPEAR++. For a broader review of related data reconstruction attacks, see Carletti et al. (2025) and for attacks relevant to our setting, see Appendix B.

SPEAR++ considers models whose first layer is fully connected and followed by ReLU activation, with $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$ and $\boldsymbol{a} = \mathrm{ReLU}(\boldsymbol{z})$ for an input $\boldsymbol{x}$. Here, $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{b} \in \mathbb{R}^m$ denote the first-layer weight matrix and bias vector, respectively. Assuming disjoint minibatches across local update steps, applying the chain rule to Equation 2 for the first-layer weight and bias gives

$$\Delta\boldsymbol{W}_c^{(t)} = \sum_{k=1}^{K} \sum_{(\boldsymbol{x},y)\in\mathcal{B}_c^{(t,k)}} \underbrace{\Big(-\tfrac{\eta^{(t)}}{b}(\nabla_{\boldsymbol{a}}\varphi) \odot \mathbb{I}[\boldsymbol{z} > 0]\Big)}_{=:\boldsymbol{g}} \boldsymbol{x}^\top =: \boldsymbol{G}^{(t)} \boldsymbol{X}^{(t)\top}, \qquad \Delta\boldsymbol{b}_c^{(t)} = \boldsymbol{G}^{(t)}\mathbf{1}. \quad (3)$$

In this factorization, $B = bK$ is the number of inputs, the columns of $\boldsymbol{X}^{(t)} \in \mathbb{R}^{n\times B}$ are the inputs $\boldsymbol{x}$, and the columns of $\boldsymbol{G}^{(t)} \in \mathbb{R}^{m\times B}$ are their corresponding backpropagated gradients $\boldsymbol{g}$. Here, $\boldsymbol{z}$ and $\nabla_{\boldsymbol{a}}\varphi$ are evaluated for $\boldsymbol{\theta}_c^{(t,k-1)}$ for the corresponding input. The ReLU indicator $\mathbb{I}[\boldsymbol{z} > 0]$ in Equation 3 induces sparsity in each column of $\boldsymbol{G}^{(t)}$. Below, we omit the client index c and round index t when clear from context.

SPEAR++ aims to reconstruct $\boldsymbol{X}$ by recovering the true factorization in Equation 3 from the observed $\Delta\boldsymbol{W}$. SPEAR++ assumes that $\boldsymbol{G}$ and $\boldsymbol{X}$ have full column rank, which ensures $\Delta\boldsymbol{W}$ has rank B. Under this assumption, SPEAR++ first computes its exact rank-B singular value decomposition (SVD) $\Delta\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ and defines $\boldsymbol{L} := \boldsymbol{U} \in \mathbb{R}^{m\times B}$ and $\boldsymbol{R} := \boldsymbol{\Sigma}\boldsymbol{V}^\top \in \mathbb{R}^{B\times n}$. The matrices $\boldsymbol{L}$ and $\boldsymbol{G}$ have the same B-dimensional column space as $\Delta\boldsymbol{W}$, so there exists a unique invertible matrix $\boldsymbol{Q} \in \mathbb{R}^{B\times B}$ such that $\boldsymbol{G} = \boldsymbol{L}\boldsymbol{Q}$, which gives $\boldsymbol{X}^\top = \boldsymbol{Q}^{-1}\boldsymbol{R}$ (Bakarsky et al., 2025). Hence, reconstructing the inputs reduces to recovering $\boldsymbol{Q}$.

To recover $\boldsymbol{Q}$, SPEAR++ searches for candidate column directions individually, then selects B linearly independent candidates and determines their signed scales from the bias update. For the i-th column $\boldsymbol{q}_i$ of $\boldsymbol{Q}$, $\boldsymbol{L}\boldsymbol{q}_i$ is the corresponding sparse column of $\boldsymbol{G}$. SPEAR++ exploits this sparsity to search for these directions by repeatedly solving the following ℓ_1-minimization from different initializations, each yielding one candidate direction.

$$\min_{\boldsymbol{q}\in\mathbb{S}^{B-1}} \|\boldsymbol{L}\boldsymbol{q}\|_1. \quad (4)$$

Here, $\mathbb{S}^{B-1} := \{\boldsymbol{q} \in \mathbb{R}^B \mid \|\boldsymbol{q}\|_2 = 1\}$ denotes the unit sphere in $\mathbb{R}^B$. Among the candidate directions obtained by solving this minimization, SPEAR++ discards those for which $\boldsymbol{L}\boldsymbol{q}$ is not sufficiently sparse.

The remaining candidates still have undetermined signed scales because the minimization is solved over the unit sphere. SPEAR++ first selects B linearly independent candidates. Their signed scales

are then recovered from the bias relation in Equation 3, $\Delta \boldsymbol{b} = \boldsymbol{L}\boldsymbol{Q}\mathbf{1}$, yielding $\hat{\boldsymbol{Q}}$. SPEAR++ then greedily replaces columns of $\hat{\boldsymbol{Q}}$ with other candidates to increase the number of matching zero and nonzero positions between the ReLU activations of the inputs reconstructed from $\hat{\boldsymbol{Q}}$ and the corresponding columns of $\boldsymbol{L}\hat{\boldsymbol{Q}}$. For $K > 1$, we evaluate this matching approximately using $\boldsymbol{\theta}^{(t)}$ instead of the local models $\boldsymbol{\theta}_c^{(t,k-1)}$ used at local update steps $k = 2, \ldots, K$, since the attack does not observe them. The details are described in Appendix C.

Empirically, SPEAR++ reconstructs more inputs as the number of output neurons m in the first layer increases, although its mechanism is not theoretically understood (Bakarsky et al., 2025). SPEAR++ uses only a model update observed in a single communication round and it remains unexplored whether exploiting updates from multiple rounds enables exact reconstruction of more inputs given a fixed number of output neurons. The next section formulates a single reconstruction problem that exploits model updates across multiple communication rounds.

3 Exact Data Reconstruction from Model Updates observed Across Multiple Communication Rounds

This section formulates model updates observed across multiple communication rounds as a single reconstruction problem and derives Multi-Round SPEAR++ from this formulation. We formulate the reconstruction problem in Section 3.1 and describe Multi-Round SPEAR++ in Section 3.2.

3.1 Multi-Round Reconstruction Formulation

We consider a set of observed rounds $\mathcal{T} \subseteq [T]$ in which the same B inputs are used to compute the model updates in Equation 2, so that their first-layer weight updates can be expressed using a common input matrix $\boldsymbol{X}$. We formalize this setting as follows.

Assumption 1. *There exists a common input matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_B] \in \mathbb{R}^{n \times B}$ such that, for each $t \in \mathcal{T}$, the input matrix $\boldsymbol{X}^{(t)}$ in Equation 3 satisfies $\boldsymbol{X}^{(t)} = \boldsymbol{X}\boldsymbol{P}^{(t)}$ for some permutation matrix $\boldsymbol{P}^{(t)} \in \{0,1\}^{B \times B}$.*

Under the same minibatch setting as in Section 2.2, Assumption 1 is satisfied when every observed round uses the same $B = bK$ inputs once each across the K local update steps in Equation 1, while their assignments to the steps may differ across rounds. Examples of this setting in existing implementations, as well as cases in which Assumption 1 does not hold, are discussed in Appendix D.

Under Assumption 1, the first-layer weight update in each observed round can be written as follows:

$$\Delta \boldsymbol{W}^{(t)} = \boldsymbol{G}^{(t)} \boldsymbol{X}^{(t)\top} = \boldsymbol{G}^{(t)} \boldsymbol{P}^{(t)\top} \boldsymbol{X}^{\top} = \overline{\boldsymbol{G}}^{(t)} \boldsymbol{X}^{\top}, \qquad \Delta \boldsymbol{b}^{(t)} = \overline{\boldsymbol{G}}^{(t)} \mathbf{1}, \tag{5}$$

where $\overline{\boldsymbol{G}}^{(t)} := \boldsymbol{G}^{(t)} \boldsymbol{P}^{(t)\top}$ is obtained by permuting the columns of $\boldsymbol{G}^{(t)}$ so that the inputs in $\boldsymbol{X}$ and their corresponding backpropagated gradients are in the same order. Thus, the first-layer weight update in each observed round can be expressed using the common input matrix $\boldsymbol{X}$ and a round-specific backpropagated-gradient matrix $\overline{\boldsymbol{G}}^{(t)}$, allowing us to vertically concatenate the first-layer weight and bias updates in the same round order to obtain a single factorization:

$$\Delta \boldsymbol{W}^{(\mathcal{T})} := \begin{bmatrix} \Delta \boldsymbol{W}^{(t_1)} \\ \vdots \\ \Delta \boldsymbol{W}^{(t_{|\mathcal{T}|})} \end{bmatrix} = \underbrace{\begin{bmatrix} \overline{\boldsymbol{G}}^{(t_1)} \\ \vdots \\ \overline{\boldsymbol{G}}^{(t_{|\mathcal{T}|})} \end{bmatrix}}_{=: \boldsymbol{G}^{(\mathcal{T})} \in \mathbb{R}^{|\mathcal{T}| m \times B}} \boldsymbol{X}^{\top} = \boldsymbol{G}^{(\mathcal{T})} \boldsymbol{X}^{\top}, \qquad \Delta \boldsymbol{b}^{(\mathcal{T})} = \boldsymbol{G}^{(\mathcal{T})} \mathbf{1}. \tag{6}$$

This factorization has the same form as Equation 3, with the round-specific backpropagated-gradient matrix $\boldsymbol{G}^{(t)}$ replaced by the concatenated matrix $\boldsymbol{G}^{(\mathcal{T})}$. The columns of $\boldsymbol{G}^{(\mathcal{T})}$ retain the sparsity induced by the first-layer ReLU, while concatenation increases the number of rows from m to $|\mathcal{T}|m$. This increase is structurally similar to increasing the number of output neurons in the first layer, which empirically enables SPEAR++ to reconstruct more inputs. This motivates us to reconstruct the inputs from the concatenated first-layer weight updates $\Delta \boldsymbol{W}^{(\mathcal{T})}$.

To recover the factorization in Equation 6 from the observed $\Delta \boldsymbol{W}^{(\mathcal{T})}$, we assume that $\boldsymbol{X}$ and $\boldsymbol{G}^{(\mathcal{T})}$ have full column rank, as in SPEAR++. Compared with SPEAR++, the full-column-rank assumption on the backpropagated-gradient matrix is relaxed because $\operatorname{rank}(\boldsymbol{G}^{(\mathcal{T})}) \geq \max_{t \in \mathcal{T}} \operatorname{rank}(\overline{\boldsymbol{G}}^{(t)})$. Thus, $\boldsymbol{G}^{(\mathcal{T})}$ may have full column rank even when every individual $\overline{\boldsymbol{G}}^{(t)}$ is rank deficient. Under this assumption, reconstructing the inputs reduces to recovering the unique invertible matrix $\boldsymbol{Q}^{(\mathcal{T})} \in \mathbb{R}^{B \times B}$ such that $\boldsymbol{G}^{(\mathcal{T})} = \boldsymbol{L}^{(\mathcal{T})} \boldsymbol{Q}^{(\mathcal{T})}$ and $\boldsymbol{X}^\top = (\boldsymbol{Q}^{(\mathcal{T})})^{-1} \boldsymbol{R}^{(\mathcal{T})}$, where $\boldsymbol{L}^{(\mathcal{T})}$ and $\boldsymbol{R}^{(\mathcal{T})}$ are obtained from the rank-B SVD factorization of $\Delta \boldsymbol{W}^{(\mathcal{T})}$.

To recover $\boldsymbol{Q}^{(\mathcal{T})}$, we search for its candidate column directions individually using an ℓ_1-minimization of the same form as Equation 4. To examine how each observed round contributes to this minimization, we rewrite it as follows:

$$\min_{\boldsymbol{q} \in \mathbb{S}^{B-1}} \|\boldsymbol{L}^{(\mathcal{T})} \boldsymbol{q}\|_1 = \min_{\substack{\boldsymbol{h} \in \mathbb{R}^n \\ \Delta \boldsymbol{W}^{(\mathcal{T})} \boldsymbol{h} \neq \boldsymbol{0}}} \frac{\sum_{t \in \mathcal{T}} \|\Delta \boldsymbol{W}^{(t)} \boldsymbol{h}\|_1}{\left(\sum_{t \in \mathcal{T}} \|\Delta \boldsymbol{W}^{(t)} \boldsymbol{h}\|_2^2\right)^{1/2}}. \tag{7}$$

The equality uses the reparameterization $\boldsymbol{q} = \boldsymbol{R}^{(\mathcal{T})} \boldsymbol{h} / \|\boldsymbol{R}^{(\mathcal{T})} \boldsymbol{h}\|_2$. Any $\boldsymbol{q} \in \mathbb{S}^{B-1}$ can be represented in this form, since $\boldsymbol{R}^{(\mathcal{T})}$ has full row rank. The derivation is given in Appendix E. This expression shows that the relative contribution of each round to the minimization depends on the scale of its first-layer weight update. We therefore normalize each first-layer weight update by its Frobenius norm as follows.

$$\alpha^{(t)} := \|\Delta \boldsymbol{W}^{(t)}\|_F, \qquad \widetilde{\Delta \boldsymbol{W}}^{(t)} := \frac{\Delta \boldsymbol{W}^{(t)}}{\alpha^{(t)}}, \qquad \widetilde{\Delta \boldsymbol{b}}^{(t)} := \frac{\Delta \boldsymbol{b}^{(t)}}{\alpha^{(t)}}. \tag{8}$$

We assume $\alpha^{(t)} > 0$ for all $t \in \mathcal{T}$ and divide the bias update by the same $\alpha^{(t)}$ to preserve the bias relation used in Section 2.2 to recover the signed scales of selected candidates.

Finally, vertically concatenating the normalized updates yields the following factorization.

$$\widetilde{\Delta \boldsymbol{W}}^{(\mathcal{T})} = \widetilde{\boldsymbol{G}}^{(\mathcal{T})} \boldsymbol{X}^\top, \qquad \widetilde{\Delta \boldsymbol{b}}^{(\mathcal{T})} = \widetilde{\boldsymbol{G}}^{(\mathcal{T})} \boldsymbol{1}. \tag{9}$$

Here, $\widetilde{\boldsymbol{G}}^{(\mathcal{T})}$ is the vertical concatenation of $\overline{\boldsymbol{G}}^{(t)} / \alpha^{(t)}$ over $t \in \mathcal{T}$. The normalization only rescales each $\overline{\boldsymbol{G}}^{(t)}$, preserving the sparsity in each round and the common input matrix $\boldsymbol{X}$. Multi-Round SPEAR++ therefore reconstructs the inputs from the normalized concatenated weight and bias updates using the SPEAR++ reconstruction procedure. We empirically evaluate the effect of normalization on reconstruction in Section 5.

3.2 Reconstruction Algorithm: Multi-Round SPEAR++

We now describe this reconstruction procedure. Multi-Round SPEAR++ first computes the rank-B SVD factorization $\widetilde{\Delta \boldsymbol{W}}^{(\mathcal{T})} = \widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{R}}^{(\mathcal{T})}$, with $\widetilde{\mathcal{L}}^{(\mathcal{T})} = \boldsymbol{U}$ and $\widetilde{\mathcal{R}}^{(\mathcal{T})} = \boldsymbol{\Sigma} \boldsymbol{V}^\top$. As in SPEAR++, reconstructing the inputs reduces to recovering the invertible matrix $\widetilde{\boldsymbol{Q}}^{(\mathcal{T})}$ satisfying $\widetilde{\boldsymbol{G}}^{(\mathcal{T})} = \widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{Q}}^{(\mathcal{T})}$ and $\boldsymbol{X}^\top = \left(\widetilde{\boldsymbol{Q}}^{(\mathcal{T})}\right)^{-1} \widetilde{\boldsymbol{R}}^{(\mathcal{T})}$. To recover $\widetilde{\boldsymbol{Q}}^{(\mathcal{T})}$, Multi-Round SPEAR++ uses an ℓ_1-minimization of the same form as Equation 4 to search for its candidate column directions individually. It repeatedly solves the following minimization from different initializations, each yielding one candidate direction.

$$\min_{\boldsymbol{q} \in \mathbb{S}^{B-1}} \|\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \boldsymbol{q}\|_1. \tag{10}$$

For each column $\widetilde{\boldsymbol{q}}_i$ of $\widetilde{\boldsymbol{Q}}^{(\mathcal{T})}$, $\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{q}}_i$ is the corresponding column of $\widetilde{\boldsymbol{G}}^{(\mathcal{T})}$. Let $\mathcal{I}_t$ denote the row indices of $\widetilde{\boldsymbol{G}}^{(\mathcal{T})}$ corresponding to $\overline{\boldsymbol{G}}^{(t)} / \alpha^{(t)}$. Each block $(\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{q}}_i)_{\mathcal{I}_t}$ corresponds to a column of $\overline{\boldsymbol{G}}^{(t)} / \alpha^{(t)}$ and should be sufficiently sparse in every observed round. Thus, Multi-Round SPEAR++ discards a candidate column direction $\boldsymbol{q}$ if $(\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \boldsymbol{q})_{\mathcal{I}_t}$ is not sufficiently sparse for some $t \in \mathcal{T}$. It then determines the signed scales of B linearly independent candidates from the normalized bias relation in Equation 9, as in SPEAR++. Starting from the resulting $\widehat{\boldsymbol{Q}}^{(\mathcal{T})}$, Multi-Round SPEAR++ greedily replaces its columns to increase the number of matching zero and nonzero positions between the ReLU activations of the reconstructed inputs under the corresponding model and the corresponding block of $\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widehat{\boldsymbol{Q}}^{(\mathcal{T})}$ for each observed round.

Algorithm 1 Multi-Round SPEAR++

Input: $\{\Delta \boldsymbol{W}^{(t)}, \Delta \boldsymbol{b}^{(t)}, \boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}\}_{t\in\mathcal{T}}$, B, m, τ, $\varepsilon_{\mathrm{dup}}$, N_{init}
Output: Reconstructed inputs $\widehat{\boldsymbol{X}}$

1: Normalize $\{\Delta \boldsymbol{W}^{(t)}, \Delta \boldsymbol{b}^{(t)}\}_{t\in\mathcal{T}}$ according to Equation 8
2: Construct $\widetilde{\Delta \boldsymbol{W}}^{(\mathcal{T})}$ and $\widetilde{\Delta \boldsymbol{b}}^{(\mathcal{T})}$ according to Equation 9
3: Compute the rank-B SVD factorization $\widetilde{\Delta \boldsymbol{W}}^{(\mathcal{T})} = \widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{R}}^{(\mathcal{T})}$
4: $\mathcal{C} \leftarrow \emptyset$
5: **for** $r = 1, \ldots, N_{\mathrm{init}}$ **do**
6: Solve Equation 10 from a random initialization to obtain $\boldsymbol{q}$
7: **if** $\max_{t\in\mathcal{T}} \left\| (\widetilde{\boldsymbol{L}}^{(\mathcal{T})}\boldsymbol{q})_{\mathcal{I}_t} \right\|_0 > (1-\tau)m$ **then**
8: **continue** ▷ block-wise sparsity filter
9: **if** $\mathcal{C} \neq \emptyset \ \wedge \ \max_{\boldsymbol{q}'\in\mathcal{C}} |\boldsymbol{q}'^{\top}\boldsymbol{q}| \geq 1 - \varepsilon_{\mathrm{dup}}$ **then**
10: **continue** ▷ duplicate filter
11: $\mathcal{C} \leftarrow \mathcal{C} \cup \{\boldsymbol{q}\}$
12: **if** $\mathcal{C}$ contains B linearly independent candidates **then**
13: $(\widehat{\boldsymbol{X}}, \kappa) \leftarrow \textsc{GreedyFilter}\big(\mathcal{C}, \widetilde{\boldsymbol{L}}^{(\mathcal{T})}, \widetilde{\boldsymbol{R}}^{(\mathcal{T})}, \widetilde{\Delta \boldsymbol{b}}^{(\mathcal{T})}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}}, B, m\big)$
14: **if** $\kappa = 1$ **then**
15: **return** $\widehat{\boldsymbol{X}}$
16: $(\widehat{\boldsymbol{X}}, \kappa) \leftarrow \textsc{GreedyFilter}\big(\mathcal{C}, \widetilde{\boldsymbol{L}}^{(\mathcal{T})}, \widetilde{\boldsymbol{R}}^{(\mathcal{T})}, \widetilde{\Delta \boldsymbol{b}}^{(\mathcal{T})}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}}, B, m\big)$
17: **return** $\widehat{\boldsymbol{X}}$

The overall procedure is shown in Algorithm 1, where τ denotes the sparsity threshold, $\varepsilon_{\mathrm{dup}}$ the threshold for duplicate candidates, and N_{init} the number of random initializations for solving Equation 10. Compared with SPEAR++, Multi-Round SPEAR++ adds per-round normalization (Line 1) and concatenation of the normalized updates (Line 2), while adapting the sparsity filtering (Line 7) and the evaluation of matching zero and nonzero positions (Line 13) to each observed round. GREEDYFILTER selects B linearly independent candidates and greedily replaces them as described above, returning the reconstructed inputs $\widehat{\boldsymbol{X}}$ and κ, the fraction of matching zero and nonzero positions across the observed rounds. The details of GREEDYFILTER are given in Appendix C.

4 THEORETICAL ANALYSIS

In Multi-Round SPEAR++, using normalized first-layer weight updates from $|\mathcal{T}|$ communication rounds yields $\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \in \mathbb{R}^{|\mathcal{T}|m\times B}$ in Equation 10, whereas SPEAR++ uses $\boldsymbol{L} \in \mathbb{R}^{m\times B}$ in Equation 4. Increasing the number of rows in this way is structurally similar to, though not equivalent to, increasing the number of output neurons in the first layer, which also increases the number of rows in the ℓ_1-minimization used in SPEAR++. To analyze how using multiple communication rounds differs from increasing the number of output neurons in the first layer, we first derive necessary and sufficient conditions for a direction $\boldsymbol{q}$ to be a strict local minimizer of the common ℓ_1 objective.

We write the minimization objectives of SPEAR++ in Equation 4 and Multi-Round SPEAR++ in Equation 10 in the following form:

$$\Psi_{\boldsymbol{L}}(\boldsymbol{q}) \coloneqq \|\boldsymbol{L}\boldsymbol{q}\|_1, \qquad \boldsymbol{q} \in \mathbb{S}^{B-1}, \tag{11}$$

where $\boldsymbol{L} \in \mathbb{R}^{M\times B}$ has orthonormal columns. For SPEAR++, $\boldsymbol{L}$, $\boldsymbol{G}$, and $\boldsymbol{Q}$ denote the matrices in Section 2.2, and $M = m$. For Multi-Round SPEAR++, $\boldsymbol{L} = \widetilde{\boldsymbol{L}}^{(\mathcal{T})}$, $\boldsymbol{G} = \widetilde{\boldsymbol{G}}^{(\mathcal{T})}$, $\boldsymbol{Q} = \widetilde{\boldsymbol{Q}}^{(\mathcal{T})}$, and $M = |\mathcal{T}|m$. In both cases, $\boldsymbol{G} = \boldsymbol{L}\boldsymbol{Q}$, and the ℓ_1-minimization exploits the ReLU-induced sparsity of the columns of $\boldsymbol{G}$. However, this sparsity alone does not determine whether a column direction of $\boldsymbol{Q}$ is a strict local minimizer of the objective under the unit-sphere constraint. The following proposition characterizes the strict local minimizers of Equation 11.

Proposition 1 (Necessary and sufficient condition for a strict local minimizer)**.** *For $\boldsymbol{q} \in \mathbb{S}^{B-1}$, let $A \coloneqq \{j \in [M] \mid (\boldsymbol{L}\boldsymbol{q})_j = 0\}$ and $A^c \coloneqq [M] \setminus A$. Then $\boldsymbol{q}$ is a strict local minimizer of $\Psi_{\boldsymbol{L}}$ if and*

only if there exist $\boldsymbol{u} \in \mathbb{R}^{|A|}$ *and* $\lambda \in \mathbb{R}$ *satisfying*

$$\operatorname{rank}(\boldsymbol{L}_{A,:}) = B - 1, \tag{12}$$

$$\boldsymbol{L}_{A^c,:}^\top \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q}) + \boldsymbol{L}_{A,:}^\top \boldsymbol{u} + \lambda \boldsymbol{q} = \boldsymbol{0}, \tag{13}$$

$$\|\boldsymbol{u}\|_\infty < 1. \tag{14}$$

The proof is given in Appendix E. We next examine the conditions in Proposition 1 for each true column direction of $\boldsymbol{Q}$ in both settings. For the i-th column of $\boldsymbol{Q}$, let $\boldsymbol{q}_i^\star \coloneqq \boldsymbol{Q}_{:,i}/\|\boldsymbol{Q}_{:,i}\|_2$ denote its true column direction, and let $A_i \coloneqq \{j \in [M] \mid (\boldsymbol{L}\boldsymbol{q}_i^\star)_j = 0\}$ and $A_i^c \coloneqq [M] \setminus A_i$. As established in Sections 2.2 and 3.2, $\boldsymbol{G} = \boldsymbol{L}\boldsymbol{Q}$ and $\boldsymbol{Q}$ is invertible, so the rank condition in Equation 12 is equivalent to $\operatorname{rank}(\boldsymbol{G}_{A_i,:}) = B - 1$. Each additional output neuron provides a new row of $\boldsymbol{G}$ with its own activation and backpropagated-gradient pattern across the inputs, whereas additional communication rounds provide new rows from the same neurons after their parameters have changed through training. Both can increase the rank of $\boldsymbol{G}_{A_i,:}$, although the added rows arise from different mechanisms.

The remaining conditions in Equations 13 and 14 can be evaluated through the minimum value of $\|\boldsymbol{u}\|_\infty$ subject to Equation 13. For each true column direction $\boldsymbol{q}_i^\star$, we define the dual margin as

$$\gamma_i \coloneqq 1 - \min_{\boldsymbol{u},\lambda} \|\boldsymbol{u}\|_\infty \quad \text{s.t.} \quad \boldsymbol{L}_{A_i^c,:}^\top \operatorname{sign}(\boldsymbol{L}_{A_i^c,:}\boldsymbol{q}_i^\star) + \boldsymbol{L}_{A_i,:}^\top \boldsymbol{u} + \lambda \boldsymbol{q}_i^\star = \boldsymbol{0}. \tag{15}$$

If this problem is infeasible, we set $\gamma_i \coloneqq -\infty$. Then Equations 13 and 14 hold for $\boldsymbol{q}_i^\star$ if and only if $\gamma_i > 0$. Consequently, $\boldsymbol{q}_i^\star$ is a strict local minimizer if and only if the rank condition $\operatorname{rank}(\boldsymbol{L}_{A_i,:}) = B - 1$ and the dual-margin condition $\gamma_i > 0$ both hold. Strict local minimality, however, does not guarantee that a finite number of minimization runs finds $\boldsymbol{q}_i^\star$. In Section 5.3, we therefore evaluate these two conditions together with the minimization results to distinguish failures of strict local minimality from failures in finite candidate search and in completing the reconstruction from the recovered directions.

5 Experiments

We first compare Multi-Round SPEAR++ with SPEAR++ using the same number m of output neurons in the first layer to evaluate whether exploiting multiple communication rounds enables exact reconstruction of more inputs. We then compare Multi-Round SPEAR++ with SPEAR++ using different values of m so that both methods have the same number of rows in the ℓ_1-minimization, and use the conditions derived in Section 4 to examine whether the differences between the two settings can be explained solely by the increased number of rows. Additional experimental details and results are provided in Appendix F.

5.1 Experimental Setup

Dataset and model. We used CIFAR-10 (Krizhevsky, 2009) for the main experiments. Each image was normalized using the dataset mean and standard deviation and vectorized as an input $\boldsymbol{x} \in \mathbb{R}^{3072}$. Following the fully connected ReLU setting of SPEAR++ (Bakarsky et al., 2025), we used a four-layer network with three hidden layers of $m \in \{200, 500, 1000, 2000\}$ neurons each and a 10-neuron output layer. We initialized the model using He initialization (He et al., 2015).

Federated learning. We set the number of clients to $C = 128$ and trained the model using FedAvg (McMahan et al., 2017), where all clients participated in every communication round. All clients held disjoint local datasets $\mathcal{D}_c$ of the same size B, obtained by uniformly permuting the training set and assigning B disjoint samples to each client. We varied B over $\{32, 48, 64, 80, 96, 112, 128, 160, 192\}$. We used the cross-entropy loss for training. Each client performed $K \in \{1, 4\}$ local update steps in each communication round using minibatch SGD without momentum or weight decay. For $K = 1$, all B inputs were used in a single minibatch. For $K = 4$, the inputs were randomly permuted independently in each communication round and split into K disjoint minibatches of size $b = B/K$, satisfying Assumption 1. The learning rate was constant across communication rounds, with $\eta = 0.05$ for $K = 1$ and $\eta = 0.025$ for $K = 4$.

Reconstruction and evaluation. We compared SPEAR++ (Bakarsky et al., 2025), which used the model update from the first communication round, with Multi-Round SPEAR++, which used

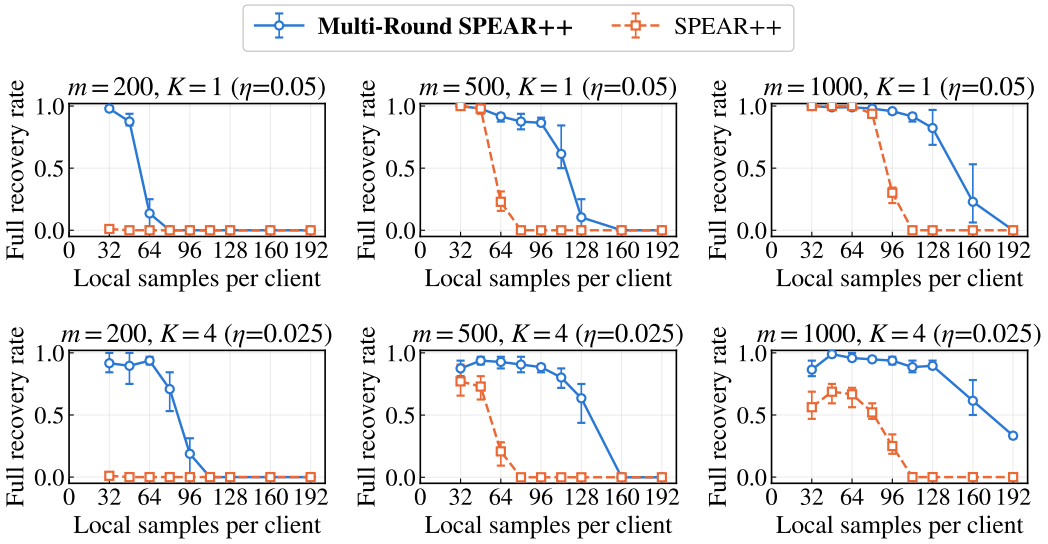

Figure 1: Full recovery rate versus local data size B for different numbers m of output neurons in the first layer on CIFAR-10. SPEAR++ uses the model update from the first communication round, whereas Multi-Round SPEAR++ uses the model updates from the first 10 communication rounds.

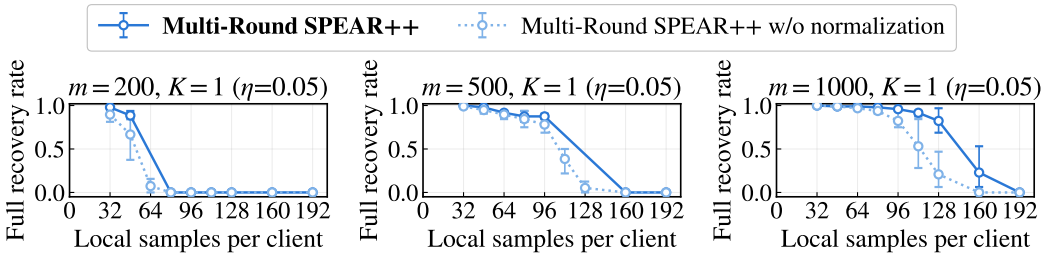

Figure 2: Effect of the per-round normalization in Equation 8 on the full recovery rate.

the model updates from each client over the first 10 communication rounds, $\mathcal{T} = [10]$. For both methods, we used Riemannian Adam (Becigneul & Ganea, 2019) with $N_{\text{init}} = 2 \times 10^5$ random initializations to solve Equation 4 and Equation 10, respectively. To evaluate the resulting reconstructions, we used the full recovery rate, defined as the fraction of clients for which all B inputs were reconstructed with PSNR above 80 dB after de-normalization to the $[0, 1]$ input range. For each seed, we evaluated reconstruction on the local datasets of 32 of the 128 clients and reported the mean together with the minimum and maximum over three seeds.

5.2 Multiple Rounds Enable Exact Reconstruction of More Inputs

In Figure 1, we compare Multi-Round SPEAR++ with SPEAR++ over local data sizes B and values of m. Multi-Round SPEAR++ remains effective at larger local data sizes after SPEAR++ fails. For example, for $K = 1$ and $m = 200$, SPEAR++ has a zero full recovery rate at $B = 32$, whereas Multi-Round SPEAR++ achieves approximately 0.97. Even with $m = 1000$, the full recovery rate of SPEAR++ drops to zero at $B = 112$, whereas Multi-Round SPEAR++ achieves approximately 0.82 at $B = 128$. Similar behavior is observed for $K = 4$. These results show that exploiting model updates across multiple communication rounds enables exact reconstruction of larger local datasets without increasing the number of output neurons in the first layer. Thus, evaluating reconstruction attacks using only a single communication round can underestimate reconstruction risk.

Moreover, in Figure 2, we compare Multi-Round SPEAR++ with and without the per-round normalization defined in Equation 8. Across all settings shown ($K = 1$), Multi-Round SPEAR++ achieves

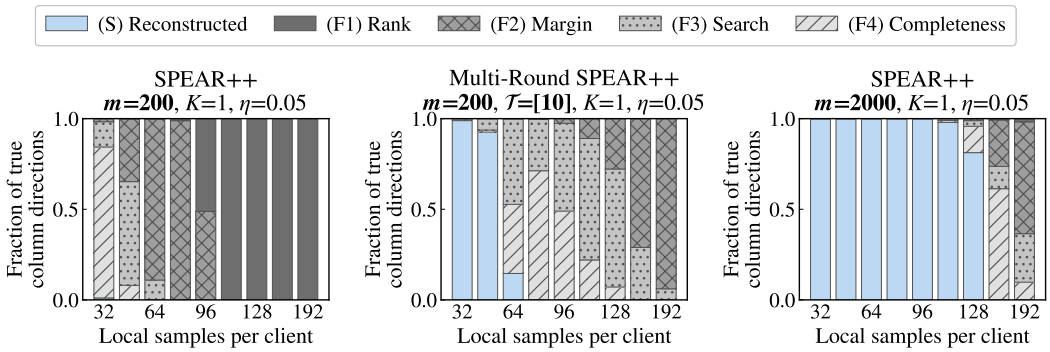

Figure 3: For each true column direction $\boldsymbol{q}_i^\star$ corresponding to input $\boldsymbol{x}_i$, we distinguish successful reconstruction (S) from four failure cases. (S) The column is recovered and the input is reconstructed. (F1) The rank condition in Equation 12 fails for its column direction. (F2) The rank condition holds, but the dual-margin condition $\gamma_i > 0$ fails. (F3) Both conditions hold, but candidate search does not find the column direction. (F4) The column direction is found, but full reconstruction fails because the complete set of required candidates and their signed scales are not successfully recovered.

an equal or higher mean full recovery rate with per-round normalization than without it. The difference is particularly large at $m = 1000$ and $B \geq 96$, highlighting the benefit of normalizing the model updates before concatenating them across communication rounds.

5.3 Concatenating First-Layer Weight Updates Differs from Increasing First-Layer Output Neurons

In Figure 3, we compare SPEAR++ with $m = 200$, Multi-Round SPEAR++ with $m = 200$ using the first 10 communication rounds, $\mathcal{T} = [10]$, and SPEAR++ with $m = 2000$. Comparing the first two settings shows how exploiting multiple communication rounds changes reconstruction, while comparing the latter two, which both use 2000 rows for candidate search, examines whether these changes can be explained solely by the increased number of rows.

For $B \geq 96$, rank-condition failures are frequent for SPEAR++ with $m = 200$ but almost eliminated for Multi-Round SPEAR++ and SPEAR++ with $m = 2000$. This is consistent with the larger number of rows reducing rank-condition failures. At $B = 160$ and 192, however, the fractions of dual-margin failures differ between Multi-Round SPEAR++ and SPEAR++ with $m = 2000$ despite both using 2000 rows, showing that equal row counts do not imply the same dual-margin behavior.

Finally, (F3) and (F4) represent failures not characterized by Proposition 1. In (F3), a true column direction satisfies both conditions but is not found within the N_{init} random initializations. In (F4), the direction is found, but full reconstruction fails because the subsequent selection of B linearly independent candidates or recovery of their signed scales is unsuccessful. Thus, strict local minimality of individual true column directions is not sufficient for full reconstruction, which additionally requires finding all directions within the finite search budget, selecting B linearly independent candidates, and recovering their signed scales.

6 Conclusion

In this paper, we formulated model updates observed across multiple communication rounds as a single reconstruction problem and derived Multi-Round SPEAR++ from this formulation. Our experiments on CIFAR-10 showed that Multi-Round SPEAR++ exactly reconstructs more inputs than SPEAR++ using the model update from the first communication round. To analyze the differences in reconstruction behavior, we characterized the strict local minimizers of the ℓ_1-minimization used for reconstruction and used this characterization to compare exploiting model updates across multiple communication rounds with increasing the number of output neurons in the first layer. These results show that evaluating reconstruction attacks using only a model update observed in a single communication round can underestimate reconstruction risk in federated learning.

AI USE STATEMENT

In this work, we used generative AI tools to assist in writing the proof of Proposition 1 and editing code used in our experiments. We also used generative AI tools to improve the readability and organization of the manuscript. We did not use generative AI tools for any other tasks requiring disclosure. We reviewed and verified all AI-assisted work and take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The proposed algorithm is given in Algorithm 1. Additional experimental details and ablation studies are provided in Appendix F. Proofs and derivations omitted from the main text are provided in Appendix E.

ETHICS STATEMENT

Our goal is to more accurately evaluate reconstruction risk in federated learning. To this end, we derive Multi-Round SPEAR++ and evaluate it on the publicly available CIFAR-10 and MNIST datasets in a controlled setting, without involving data from real users or deployed federated learning systems. Our results show that exploiting the same client's model updates across multiple communication rounds can enable exact reconstruction of more inputs than using only a model update observed in a single communication round, indicating that evaluating reconstruction risk should account for model updates across multiple communication rounds when they are available to an attacker. Together with prior work on data reconstruction attacks, our results motivate additional protections beyond keeping training data local for federated learning applications involving sensitive data.

One such protection is secure aggregation, which can prevent the server from observing individual client model updates and thereby from exploiting the same client's model updates across multiple communication rounds. Another is to add noise calibrated to satisfy differential privacy before model updates are shared with the server, making it more difficult to estimate the rank of the first-layer weight update and recover the factorization used by the attack. Overall, we hope that our work encourages the design of federated learning systems that use such protections for applications involving sensitive data.

REFERENCES

Alexander Bakarsky, Dimitar Iliev Dimitrov, Maximilian Baader, and Martin Vechev. SPEAR++: Scaling gradient inversion via sparsely-used dictionary learning. In *NeurIPS 2025 Workshop on Regulatable ML*, 2025.

Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*, 2019.

Vincenzo Carletti, Pasquale Foggia, Carlo Mazzocca, Giuseppe Parrella, and Mario Vento. SoK: Gradient inversion attacks in federated learning. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 6439–6459. USENIX Association, 2025.

Xiangyi Chen, Xiaoyun Li, and Ping Li. Toward communication efficient adaptive gradient method. In *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, pp. 119–128. Association for Computing Machinery, 2020. doi: 10.1145/3412815.3416891.

Francesco Diana, André Nusser, Chuan Xu, and Giovanni Neglia. Cutting through privacy: A hyperplane-based data reconstruction attack in federated learning. In *Proceedings of the Forty-first Conference on Uncertainty in Artificial Intelligence*, volume 286 of *Proceedings of Machine Learning Research*, pp. 959–980. PMLR, 2025.

Dimitar Iliev Dimitrov, Maximilian Baader, Mark Niklas Mueller, and Martin Vechev. SPEAR: Exact gradient inversion of batches in federated learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients - how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2): 1–210, 2021. doi: 10.1561/2200000083.

William Kalikman, Ivo Petrov, Dimitar I Dimitrov, and Martin Vechev. TIGER: Inverting transformer gradients via embedding-subspace distance optimization. *arXiv preprint arXiv:2606.18312*, 2026.

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 15884–15899. PMLR, 2023.

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

Bowen Li, Hanlin Gu, Ruoxin Chen, Jie Li, Chentao Wu, Na Ruan, Xueming Si, and Lixin Fan. Temporal gradient inversion attacks with robust optimization. *IEEE Transactions on Dependable and Secure Computing*, 22(4):3383–3397, 2025. doi: 10.1109/TDSC.2025.3532339.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Ivo Petrov, Dimitar Iliev Dimitrov, Maximilian Baader, Mark Niklas Mueller, and Martin Vechev. DAGER: Exact gradient inversion for large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations*, 2018.

Guangnian Wan, Gongfan Fang, Xinyin Ma, and Xinchao Wang. Gradient inversion attacks beyond SGD. In *Forty-third International Conference on Machine Learning*, 2026. URL `https://openreview.net/forum?id=RNDJpm1kUW`.

Joshua Zhao, Saurabh Bagchi, Salman Avestimehr, Kevin Chan, Somali Chaterji, Dimitris Dimitriadis, Jiacheng Li, Ninghui Li, Arash Nourian, and Holger Roth. The federation strikes back: A survey of federated learning privacy attacks, defenses, applications, and policy landscape. *ACM Computing Surveys*, 57(9), 2025. doi: 10.1145/3724113.

Junyi Zhu and Matthew B. Blaschko. R-GAP: Recursive gradient attack on privacy. In *International Conference on Learning Representations*, 2021.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A Limitations and Future Work

We briefly discuss the limitations and promising future directions of this work.

First, Multi-Round SPEAR++ does not directly apply when data augmentation produces different augmented versions of the same input across observed rounds. When data augmentation is applied independently in each observed round, different augmented versions of the same input can be used to compute the model updates, so Assumption 1 no longer holds (see Appendix D). Exploiting model updates observed across multiple communication rounds to exactly reconstruct inputs in this setting remains an open problem.

Second, our formulation relies on minibatch SGD in the local update steps performed by each client, under which the first-layer weight and bias updates can be written as $\Delta \boldsymbol{W} = \boldsymbol{G}\boldsymbol{X}^\top$ and $\Delta \boldsymbol{b} = \boldsymbol{G}\mathbf{1}$ as in Equation 3. Adaptive methods such as AMSGrad (Reddi et al., 2018) have also been considered for local updates in federated learning (Chen et al., 2020). Recent work has begun to address data reconstruction from model updates generated using such methods (Wan et al., 2026), although the resulting reconstructions remain approximate. Under such adaptive methods, the first-layer weight update generally cannot be written as $\Delta \boldsymbol{W} = \boldsymbol{G}\boldsymbol{X}^\top$, so our formulation does not directly apply. This limitation is inherited from SPEAR (Dimitrov et al., 2024) and SPEAR++ (Bakarsky et al., 2025), which also rely on this factorization for exact reconstruction. Exact reconstruction under such local updates remains unexplored and is a promising direction for future work.

Table 1: Comparison of representative data reconstruction attacks, organized into gradient-matching and structure-exploiting approaches. The second through fifth columns indicate whether each attack exploits multiple communication rounds, operates under an honest-but-curious threat model, achieves exact reconstruction, and reconstructs multiple training samples. The last column presents the model architecture considered by each attack.

Method	Multiple rounds	Honest-but -curious	Exact reconstruction	Multiple samples	Model architecture
Gradient-matching approaches					
DLG (Zhu et al., 2019)	–	✓	–	✓	FC/CNN
TGIAs-RO (Li et al., 2025)	✓	✓	–	✓	FC/CNN
Structure-exploiting approaches					
R-GAP (Zhu & Blaschko, 2021)	–	✓	✓	–	FC/CNN
CPA (Kariyappa et al., 2023)	–	✓	–	✓	FC/CNN
SPEAR (Dimitrov et al., 2024)	–	✓	✓	✓	FC+ReLU
SPEAR++ (Bakarsky et al., 2025)	–	✓	✓	✓	FC+ReLU
Diana et al. (2025)	✓	–	✓	✓	FC+ReLU
Multi-Round SPEAR++	✓	✓	✓	✓	FC+ReLU

B Related Work

In this section, we review representative data reconstruction attacks and clarify the positioning of our attack. For this comparison, we group existing attacks into gradient-matching approaches, which optimize candidate inputs to match the observed gradients, and structure-exploiting approaches, which recover inputs directly from algebraic structure in the observed gradients. Among the existing attacks summarized in Table 1, none simultaneously achieves exact reconstruction of multiple training samples across multiple communication rounds under an honest-but-curious threat model.

Zhu et al. (2019) demonstrated that training samples can be reconstructed from observed gradients by optimizing dummy inputs so that their gradients match the observed gradients. Subsequent gradient-matching attacks improved reconstruction by designing better gradient-matching objectives or by incorporating auxiliary information and priors. However, these attacks primarily exploit gradients observed in a single communication round.

Motivated by the limitations of reconstruction from a single communication round, TGIAs-RO (Li et al., 2025) exploits gradients observed across multiple communication rounds under an honest-but-curious threat model. For each communication round, TGIAs-RO independently optimizes a candidate input to match the observed gradient and then combines the candidates reconstructed from multiple rounds using robust statistics to obtain the final reconstruction. The resulting reconstruction remains approximate. In contrast, Multi-Round SPEAR++ combines model updates observed across multiple communication rounds before reconstruction and formulates them as a single reconstruction problem, enabling exact reconstruction of multiple training samples.

A complementary line of work recovers inputs directly by exploiting algebraic structure in observed gradients. R-GAP (Zhu & Blaschko, 2021) exploits the fact that, for a single training sample, the weight gradient at each layer factorizes into the outer product of a backpropagated gradient and the layer input, enabling exact reconstruction under suitable rank conditions. When multiple training samples contribute to the observed gradient, it becomes a sum of such outer products, preventing R-GAP from directly separating the individual samples. CPA (Kariyappa et al., 2023) addresses this limitation by treating the weight gradient as a linear mixture of multiple training samples and separating them using independent component analysis, although the resulting reconstruction remains approximate.

To achieve exact reconstruction of multiple training samples, SPEAR (Dimitrov et al., 2024) exploits the low-rank structure of the weight gradient together with ReLU-induced sparsity to identify the decomposition corresponding to the individual training samples. SPEAR++ (Bakarsky et al., 2025) further extends exact reconstruction to a larger number of training samples by improving the search for this decomposition, as described in Section 2.2. DAGER (Petrov et al., 2024) and

TIGER (Kalikman et al., 2026) extend related structure-exploiting approaches to Transformer models. However, these structure-exploiting attacks reconstruct training samples using information from only a single communication round.

Diana et al. (2025) study exact reconstruction of multiple training samples across multiple communication rounds under a malicious threat model. Their attack changes the bias of a fully connected ReLU neuron across rounds, shifting its activation hyperplane to progressively isolate individual samples. With sufficiently many rounds, this procedure can reconstruct all training samples in a batch. In contrast, Multi-Round SPEAR++ achieves exact reconstruction of multiple training samples under the honest-but-curious threat model considered throughout this paper, without intervening in the training procedure.

Algorithm 2 SPEAR++

Input: $\Delta \boldsymbol{W}^{(t')}, \Delta \boldsymbol{b}^{(t')}, \boldsymbol{W}^{(t')}, \boldsymbol{b}^{(t')}, B, m, \tau, \varepsilon_{\text{dup}}, N_{\text{init}}$
Output: Reconstructed inputs $\widehat{\boldsymbol{X}}$

Compute the rank-B SVD factorization $\Delta \boldsymbol{W}^{(t')} = \boldsymbol{L}^{(t')} \boldsymbol{R}^{(t')}$
$\mathcal{C} \leftarrow \emptyset$
for $r = 1, \ldots, N_{\text{init}}$ **do**
  Solve Equation 4 from a random initialization to obtain $\boldsymbol{q}$
  if $\|\boldsymbol{L}^{(t')}\boldsymbol{q}\|_0 > (1-\tau)m$ **then**
    continue ▷ sparsity filter
  if $\mathcal{C} \neq \emptyset \ \wedge \ \max_{\boldsymbol{q}' \in \mathcal{C}} |\boldsymbol{q}'^{\top} \boldsymbol{q}| \geq 1 - \varepsilon_{\text{dup}}$ **then**
    continue ▷ duplicate filter
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{\boldsymbol{q}\}$
  if $\mathcal{C}$ contains B linearly independent candidates **then**
    $(\widehat{\boldsymbol{X}}, \kappa) \leftarrow \textsc{GreedyFilter}\big(\mathcal{C}, \boldsymbol{L}^{(t')}, \boldsymbol{R}^{(t')}, \Delta \boldsymbol{b}^{(t')}, \{(\boldsymbol{W}^{(t')}, \boldsymbol{b}^{(t')}, [m])\}, B, m\big)$
    if $\kappa = 1$ **then**
      return $\widehat{\boldsymbol{X}}$
$(\widehat{\boldsymbol{X}}, \kappa) \leftarrow \textsc{GreedyFilter}\big(\mathcal{C}, \boldsymbol{L}^{(t')}, \boldsymbol{R}^{(t')}, \Delta \boldsymbol{b}^{(t')}, \{(\boldsymbol{W}^{(t')}, \boldsymbol{b}^{(t')}, [m])\}, B, m\big)$
return $\widehat{\boldsymbol{X}}$

C SPEAR++ AND GREEDYFILTER

In this section, we describe the complete algorithm of SPEAR++ and the GreedyFilter procedure used by both SPEAR++ and Multi-Round SPEAR++. Algorithm 2 presents SPEAR++ (Bakarsky et al., 2025) using the notation of this paper, and Algorithm 3 gives GreedyFilter.

As described in Section 2.2, for a model update observed in communication round t', SPEAR++ computes the rank-B SVD factorization $\Delta \boldsymbol{W}^{(t')} = \boldsymbol{L}^{(t')} \boldsymbol{R}^{(t')}$, and reconstructing the inputs reduces to recovering the invertible matrix $\boldsymbol{Q}$ satisfying $\boldsymbol{G}^{(t')} = \boldsymbol{L}^{(t')} \boldsymbol{Q}$. To recover $\boldsymbol{Q}$, SPEAR++ repeatedly solves the ℓ_1-minimization in Equation 4 from different initializations to obtain candidates for its columns and filters the resulting candidates. SPEAR++ then uses GreedyFilter to recover $\boldsymbol{Q}$ from the collected candidates.

GreedyFilter first constructs an initial selection of B linearly independent candidates by prioritizing candidates that induce the sparsest $\boldsymbol{Lq}$, following SPEAR++ (Bakarsky et al., 2025). For each tentative selection, CONSISTENCY recovers the signed scales from the bias update, constructs $\boldsymbol{Q}$, and reconstructs the corresponding inputs. The value κ represents the fraction of entries for which the ReLU activations of the reconstructed inputs are consistent with the nonzero entries of the corresponding recovered gradient columns. GreedyFilter then greedily replaces selected candidates whenever the replacement increases κ. If $\kappa = 1$, all entries included in this comparison are consistent, and the resulting $\boldsymbol{Q}$ is returned. When GreedyFilter is called during candidate search and $\kappa < 1$, the candidate search continues from another initialization.

Similarly, as described in Section 3, Multi-Round SPEAR++ recovers $\boldsymbol{Q}$ from the normalized concatenated factorization $\widetilde{\Delta \boldsymbol{W}}^{(\mathcal{T})} = \widetilde{\boldsymbol{L}}^{(\mathcal{T})} \widetilde{\boldsymbol{R}}^{(\mathcal{T})}$ and uses GreedyFilter to recover $\boldsymbol{Q}$ from the collected candidates. The difference is in the computation of κ. Each row block $\mathcal{I}_t$ of $\widetilde{\boldsymbol{L}}^{(\mathcal{T})} \boldsymbol{Q}$ contains the backpropagated gradients in communication round t. Therefore, for each $t \in \mathcal{T}$, their nonzero entries are compared with the ReLU activations induced by the corresponding model in that communication round. The value κ is computed over all reconstructed inputs and observed communication rounds. Thus, $\kappa = 1$ for Multi-Round SPEAR++ only when this consistency condition is satisfied for every reconstructed input in every observed communication round.

The consistency evaluation above requires the ReLU activation pattern associated with the model used to compute each recovered gradient. For $K = 1$ in Equation 1, the model used to compute the local update is the model distributed at the beginning of the communication round, which is observed by the attacker. For $K > 1$, however, the gradient at local update step k is computed using the intermediate local model $\boldsymbol{\theta}_c^{(t,k-1)}$. Accordingly, evaluating the consistency exactly would require the

Algorithm 3 Greedy candidate selection and ReLU-mask consistency

procedure GREEDYFILTER($\mathcal{C}, \boldsymbol{L}, \boldsymbol{R}, \Delta\boldsymbol{b}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}_{\text{eval}}}, B, m$)
  Initialize $\boldsymbol{Q}_d$ as an empty matrix
  $\mathcal{C}_{\text{rem}} \leftarrow \mathcal{C}$
  while $\boldsymbol{Q}_d$ has fewer than B columns and $\mathcal{C}_{\text{rem}} \neq \emptyset$ **do**
    Select $\boldsymbol{q} \in \mathcal{C}_{\text{rem}}$ that induces the sparsest $\boldsymbol{Lq}$
    $\mathcal{C}_{\text{rem}} \leftarrow \mathcal{C}_{\text{rem}} \setminus \{\boldsymbol{q}\}$
    if $\boldsymbol{q}$ is linearly independent of the columns of $\boldsymbol{Q}_d$ **then**
      Append $\boldsymbol{q}$ as a column of $\boldsymbol{Q}_d$
  if $\boldsymbol{Q}_d$ has fewer than B columns **then**
    Augment $\boldsymbol{Q}_d$ with orthonormal vectors spanning the orthogonal complement of its column space until it has B columns
  $(\widehat{\boldsymbol{X}}, \kappa) \leftarrow$ CONSISTENCY$\big(\boldsymbol{Q}_d, \boldsymbol{L}, \boldsymbol{R}, \Delta\boldsymbol{b}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}_{\text{eval}}}, B, m\big)$
  if $\kappa = 1$ **then**
    return $(\widehat{\boldsymbol{X}}, \kappa)$
  loop
    $\boldsymbol{Q}_d^\star \leftarrow \boldsymbol{Q}_d$, $\widehat{\boldsymbol{X}}^\star \leftarrow \widehat{\boldsymbol{X}}$, $\kappa^\star \leftarrow \kappa$
    for $\boldsymbol{q}_{\text{in}} \in \mathcal{C}$ **do**
      for $j = 1, \ldots, B$ **do**
        $\boldsymbol{Q}'_d \leftarrow \boldsymbol{Q}_d$
        Replace column j of $\boldsymbol{Q}'_d$ by $\boldsymbol{q}_{\text{in}}$
        if the columns of $\boldsymbol{Q}'_d$ are linearly dependent **then**
          continue
        $(\widehat{\boldsymbol{X}}', \kappa') \leftarrow$ CONSISTENCY$\big(\boldsymbol{Q}'_d, \boldsymbol{L}, \boldsymbol{R}, \Delta\boldsymbol{b}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}_{\text{eval}}}, B, m\big)$
        if $\kappa' > \kappa^\star$ **then**
          $\boldsymbol{Q}_d^\star \leftarrow \boldsymbol{Q}'_d$, $\widehat{\boldsymbol{X}}^\star \leftarrow \widehat{\boldsymbol{X}}'$, $\kappa^\star \leftarrow \kappa'$
    if $\kappa^\star \leq \kappa$ **then**
      break
    $\boldsymbol{Q}_d \leftarrow \boldsymbol{Q}_d^\star$, $\widehat{\boldsymbol{X}} \leftarrow \widehat{\boldsymbol{X}}^\star$, $\kappa \leftarrow \kappa^\star$
    if $\kappa = 1$ **then**
      return $(\widehat{\boldsymbol{X}}, \kappa)$
  return $(\widehat{\boldsymbol{X}}, \kappa)$

procedure CONSISTENCY($\boldsymbol{Q}_d, \boldsymbol{L}, \boldsymbol{R}, \Delta\boldsymbol{b}, \{(\boldsymbol{W}^{(t)}, \boldsymbol{b}^{(t)}, \mathcal{I}_t)\}_{t\in\mathcal{T}_{\text{eval}}}, B, m$)
  $\boldsymbol{s} \leftarrow (\boldsymbol{LQ}_d)^\dagger \Delta\boldsymbol{b}$
  $\boldsymbol{Q} \leftarrow \boldsymbol{Q}_d \operatorname{diag}(\boldsymbol{s})$
  $\widehat{\boldsymbol{X}}^\top \leftarrow \boldsymbol{Q}^{-1}\boldsymbol{R}$
  $\kappa \leftarrow 0$
  for $t \in \mathcal{T}_{\text{eval}}$ **do**
    for $j = 1, \ldots, B$ **do**
      $\widehat{\boldsymbol{z}}_j^{(t)} \leftarrow \boldsymbol{W}^{(t)}\widehat{\boldsymbol{x}}_j + \boldsymbol{b}^{(t)}$
      $\widehat{\boldsymbol{g}}_j^{(t)} \leftarrow (\boldsymbol{LQ})_{\mathcal{I}_t, j}$
      $\kappa \leftarrow \kappa + m - \Big\| \mathbb{I}[\widehat{\boldsymbol{z}}_j^{(t)} > 0] - \mathbb{I}[|\widehat{\boldsymbol{g}}_j^{(t)}| > 0] \Big\|_1$
  $\kappa \leftarrow \kappa / (|\mathcal{T}_{\text{eval}}| Bm)$
  return $(\widehat{\boldsymbol{X}}, \kappa)$

ReLU activation pattern induced by the corresponding first-layer parameters $\boldsymbol{W}_c^{(t,k-1)}$ and $\boldsymbol{b}_c^{(t,k-1)}$ for each input used at step k. These intermediate local models are not observed, so GreedyFilter instead uses the model at the beginning of the communication round. Consequently, even the correct candidate selection may yield $\kappa < 1$ for both SPEAR++ and Multi-Round SPEAR++ when $K > 1$.

D Discussion of the Common-Input Assumption formalized in Assumption 1

Here, we discuss the scope of the common-input assumption formalized in Assumption 1. We first consider (i) the setting in which the same inputs are used across observed rounds, under which Assumption 1 holds. We then consider two settings in which Assumption 1 does not hold, (ii) different subsets of inputs are used across observed rounds and (iii) data augmentation changes the inputs across observed rounds.

(i) Same inputs across observed rounds. Assumption 1 is satisfied when the K mutually disjoint minibatches used in Equation 1 contain the same set of $B = bK$ inputs in every observed round, while their assignments to the K local update steps may differ across rounds. This setting appears in example implementations of federated learning frameworks. For example, the Flower PyTorch quickstart[1] and the TensorFlow Federated Simple FedAvg example[2] use the same local dataset for each client across rounds, shuffle the inputs when forming minibatches, and use all inputs once in each round by default. Thus, these examples illustrate that the same inputs can be reused across communication rounds and would fall within the scope of Multi-Round SPEAR++ when combined with the model and local update settings considered in this paper.

(ii) Different subsets of inputs across observed rounds. Assumption 1 does not hold when the sets of inputs used in the observed rounds differ, including when they partially overlap. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{n \times N}$ contain all distinct inputs used across the observed rounds, and let $\boldsymbol{S}^{(t)} \in \{0,1\}^{N \times B}$ select the inputs used in round t. Substituting $\boldsymbol{X}^{(t)} = \boldsymbol{X}\boldsymbol{S}^{(t)}$ into Equation 3 gives

$$\Delta \boldsymbol{W}^{(t)} = \boldsymbol{G}^{(t)} \boldsymbol{S}^{(t)\top} \boldsymbol{X}^\top, \qquad \Delta \boldsymbol{b}^{(t)} = \boldsymbol{G}^{(t)} \boldsymbol{S}^{(t)\top} \mathbf{1}_N. \tag{16}$$

Vertically concatenating these updates yields a factorization with the common input matrix $\boldsymbol{X}$ and a concatenated backpropagated-gradient matrix with N columns. Under the same full-column-rank assumption as in Section 3.1, reconstructing the inputs can then be reduced to recovering an invertible matrix in the same way as in the common-input setting, using a rank-N SVD instead of a rank-B SVD. If input $\boldsymbol{x}_i$ is not used in round t, the i-th column of $\boldsymbol{G}^{(t)}\boldsymbol{S}^{(t)\top}$ is zero, so the corresponding m-row block of the recovered gradient column is also zero. Therefore, when GreedyFilter evaluates the matching zero and nonzero positions between the ReLU activations of a reconstructed input and its corresponding recovered gradient column, it should evaluate only the rounds in which that input is used. Extending Multi-Round SPEAR++ to partially overlapping inputs requires accounting for input participation across rounds in this matching evaluation.

(iii) Data augmentation that changes inputs across observed rounds. When data augmentation is resampled during local update steps, the augmented versions of the same input can differ across observed rounds. In this case, Assumption 1 no longer holds, and the first-layer weight updates cannot be expressed using a common input matrix as in Equation 5. Consequently, the common-input formulation of Multi-Round SPEAR++ does not directly apply in this setting.

[1] https://github.com/flwrlabs/flower/tree/main/examples/quickstart-pytorch

[2] https://github.com/google-parfait/tensorflow-federated/blob/main/examples/simple_fedavg/emnist_fedavg_main.py

E Derivation and Proof

E.1 Derivation of Equation 7

To derive Equation 7, note that the matrix $\boldsymbol{R}^{(\mathcal{T})} \in \mathbb{R}^{B\times n}$ has full row rank, so $\mathrm{range}(\boldsymbol{R}^{(\mathcal{T})}) = \mathbb{R}^B$. Hence,

$$\mathbb{S}^{B-1} = \left\{ \frac{\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}}{\|\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_2} \,\middle|\, \boldsymbol{h} \in \mathbb{R}^n,\ \boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h} \neq \boldsymbol{0} \right\}.$$

Using this parameterization,

$$\begin{aligned}
\min_{\boldsymbol{q}\in\mathbb{S}^{B-1}} \|\boldsymbol{L}^{(\mathcal{T})}\boldsymbol{q}\|_1 &= \min_{\substack{\boldsymbol{h}\in\mathbb{R}^n \\ \boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\neq\boldsymbol{0}}} \left\| \boldsymbol{L}^{(\mathcal{T})} \frac{\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}}{\|\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_2} \right\|_1 \\
&= \min_{\substack{\boldsymbol{h}\in\mathbb{R}^n \\ \boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\neq\boldsymbol{0}}} \frac{\|\boldsymbol{L}^{(\mathcal{T})}\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_1}{\|\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_2} \\
&= \min_{\substack{\boldsymbol{h}\in\mathbb{R}^n \\ \boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\neq\boldsymbol{0}}} \frac{\|\Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\|_1}{\|\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_2} \\
&= \min_{\substack{\boldsymbol{h}\in\mathbb{R}^n \\ \Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\neq\boldsymbol{0}}} \frac{\|\Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\|_1}{\|\Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\|_2} \\
&= \min_{\substack{\boldsymbol{h}\in\mathbb{R}^n \\ \Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\neq\boldsymbol{0}}} \frac{\sum_{t\in\mathcal{T}} \|\Delta\boldsymbol{W}^{(t)}\boldsymbol{h}\|_1}{\left(\sum_{t\in\mathcal{T}} \|\Delta\boldsymbol{W}^{(t)}\boldsymbol{h}\|_2^2\right)^{1/2}}.
\end{aligned}$$

The third equality follows from $\Delta\boldsymbol{W}^{(\mathcal{T})} = \boldsymbol{L}^{(\mathcal{T})}\boldsymbol{R}^{(\mathcal{T})}$. For the fourth equality, the orthonormal columns of $\boldsymbol{L}^{(\mathcal{T})}$ give $\|\Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h}\|_2 = \|\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h}\|_2$, and hence $\Delta\boldsymbol{W}^{(\mathcal{T})}\boldsymbol{h} \neq \boldsymbol{0}$ if and only if $\boldsymbol{R}^{(\mathcal{T})}\boldsymbol{h} \neq \boldsymbol{0}$. The last equality expands the ℓ_1- and ℓ_2-norms over the vertically concatenated round-wise blocks. This yields Equation 7.

E.2 Mathematical Tools

This subsection introduces the definitions and lemmas used in the proof.

Definition 1 (Strict local minimizer)**.** *A point $\boldsymbol{q} \in \Omega$ is a strict local minimizer of f on Ω if there exists $\epsilon > 0$ such that $f(\boldsymbol{q}) < f(\boldsymbol{q}')$ for every $\boldsymbol{q}' \in \Omega \cap B_\epsilon(\boldsymbol{q})$ with $\boldsymbol{q}' \neq \boldsymbol{q}$, where $B_\epsilon(\boldsymbol{q}) := \{\boldsymbol{q}' \mid \|\boldsymbol{q}' - \boldsymbol{q}\|_2 < \epsilon\}$.*

Definition 2 (Subgradient)**.** *For a convex function $f : \mathbb{R}^d \to \mathbb{R}$, a vector $\boldsymbol{g} \in \mathbb{R}^d$ is a subgradient of f at $\boldsymbol{x}$ if $f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \langle \boldsymbol{g}, \boldsymbol{y} - \boldsymbol{x}\rangle$ for every $\boldsymbol{y} \in \mathbb{R}^d$.*

For the ℓ_1-norm, a vector $\boldsymbol{g}$ is a subgradient at $\boldsymbol{x}$ if, for each i,

$$g_i = \mathrm{sign}(x_i) \quad \text{if } x_i \neq 0, \qquad g_i \in [-1, 1] \quad \text{if } x_i = 0. \tag{17}$$

In particular, if all entries of $\boldsymbol{x}$ are nonzero, the subgradient is unique and equals $\mathrm{sign}(\boldsymbol{x})$.

Lemma 1 (Hölder's inequality)**.** *For any vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, the following holds.*

$$|\langle \boldsymbol{x}, \boldsymbol{y}\rangle| \leq \|\boldsymbol{x}\|_\infty \|\boldsymbol{y}\|_1.$$

E.3 Proof of Proposition 1

Using these tools, we now prove Proposition 1.

For $B = 1$, $\mathbb{S}^0 = \{-1, 1\}$. For either $\boldsymbol{q} \in \mathbb{S}^0$, choosing any $\epsilon < 2$ gives $\mathcal{B}_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^0 = \{\boldsymbol{q}\}$, so $\boldsymbol{q}$ is a strict local minimizer by Definition 1. Moreover, $\boldsymbol{L}_{A,:} = \boldsymbol{0}$ gives $\mathrm{rank}(\boldsymbol{L}_{A,:}) = 0 = B - 1$, and the remaining conditions are satisfied by $\boldsymbol{u} = \boldsymbol{0}$ and $\lambda = -\|\boldsymbol{L}\boldsymbol{q}\|_1$, since $\boldsymbol{L}_{A^c,:}^\top \mathrm{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q}) = \|\boldsymbol{L}\boldsymbol{q}\|_1 \boldsymbol{q}$. Thus, the proposition holds for $B = 1$, and we assume $B \geq 2$ below.

Sufficiency. We show that if there exist $\boldsymbol{u}, \lambda$ satisfying Equations 12–14, then $\boldsymbol{q}$ is a strict local minimizer. By Definition 1, it suffices to find $\epsilon > 0$ such that

$$\Psi_{\boldsymbol{L}}(\boldsymbol{q}') > \Psi_{\boldsymbol{L}}(\boldsymbol{q})$$

for every $\boldsymbol{q}' \in B_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^{B-1}$ with $\boldsymbol{q}' \neq \boldsymbol{q}$. For an arbitrary $\boldsymbol{q}' \in \mathbb{S}^{B-1}$, write $\boldsymbol{q}' = \alpha \boldsymbol{q} + \boldsymbol{v}$, where $\alpha = {\boldsymbol{q}'}^\top \boldsymbol{q}$ and $\boldsymbol{v} = (\boldsymbol{I} - \boldsymbol{q}\boldsymbol{q}^\top)\boldsymbol{q}'$, so that $\boldsymbol{v} \perp \boldsymbol{q}$. Using $\|\boldsymbol{q}\|_2 = \|\boldsymbol{q}'\|_2 = 1$, we obtain

$$\alpha = 1 - \frac{1}{2}\|\boldsymbol{q}' - \boldsymbol{q}\|_2^2, \qquad \alpha^2 + \|\boldsymbol{v}\|_2^2 = 1.$$

We first derive a lower bound on $\Psi_{\boldsymbol{L}}(\boldsymbol{q}') - \Psi_{\boldsymbol{L}}(\boldsymbol{q})$.

$$\begin{aligned}
\Psi_{\boldsymbol{L}}(\boldsymbol{q}') - \Psi_{\boldsymbol{L}}(\boldsymbol{q}) &= \|\boldsymbol{L}_{A,:}\boldsymbol{q}'\|_1 + \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}'\|_1 - \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 \\
&= \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 + \left\|\alpha \boldsymbol{L}_{A^c,:}\boldsymbol{q} + \boldsymbol{L}_{A^c,:}\boldsymbol{v}\right\|_1 - \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 \\
&\geq \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 + \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \left((\alpha - 1)\boldsymbol{L}_{A^c,:}\boldsymbol{q} + \boldsymbol{L}_{A^c,:}\boldsymbol{v}\right) \\
&= (\alpha - 1)\|\boldsymbol{L}\boldsymbol{q}\|_1 + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 + \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \boldsymbol{L}_{A^c,:}\boldsymbol{v} \\
&= (\alpha - 1)\|\boldsymbol{L}\boldsymbol{q}\|_1 + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 - \boldsymbol{u}^\top \boldsymbol{L}_{A,:}\boldsymbol{v} \\
&\geq (\alpha - 1)\|\boldsymbol{L}\boldsymbol{q}\|_1 + (1 - \|\boldsymbol{u}\|_\infty)\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1. \qquad (18)
\end{aligned}$$

The first equality follows from the definition of $\Psi_{\boldsymbol{L}}$ in Equation 11 and $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$. The second equality substitutes $\boldsymbol{q}' = \alpha \boldsymbol{q} + \boldsymbol{v}$. The first inequality applies the subgradient inequality in Definition 2. All entries of $\boldsymbol{L}_{A^c,:}\boldsymbol{q}$ are nonzero by the definition of A^c, so $\operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})$ is the unique subgradient of $\|\cdot\|_1$ at $\boldsymbol{L}_{A^c,:}\boldsymbol{q}$. The next equality uses $\operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \boldsymbol{L}_{A^c,:}\boldsymbol{q} = \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 = \|\boldsymbol{L}\boldsymbol{q}\|_1$. The following equality uses Equation 13 and $\boldsymbol{q}^\top \boldsymbol{v} = 0$, and the final inequality follows from Lemma 1.

We next lower-bound $\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1$ in Equation 18. By the definition of A, $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$. Since $\boldsymbol{q} \in \mathbb{S}^{B-1}$, $\boldsymbol{q} \neq \boldsymbol{0}$, so $\boldsymbol{q}$ is a nonzero vector in $\ker(\boldsymbol{L}_{A,:})$. Moreover, Equation 12 gives $\operatorname{rank}(\boldsymbol{L}_{A,:}) = B - 1$, and hence $\dim \ker(\boldsymbol{L}_{A,:}) = 1$. Therefore,

$$\ker(\boldsymbol{L}_{A,:}) = \operatorname{span}(\boldsymbol{q}).$$

It follows that $\boldsymbol{L}_{A,:}\boldsymbol{v}' \neq \boldsymbol{0}$ for every $\boldsymbol{v}' \in \mathbb{R}^B$ satisfying $\boldsymbol{v}' \perp \boldsymbol{q}$ and $\|\boldsymbol{v}'\|_2 = 1$. The set $\{\boldsymbol{v}' \in \mathbb{R}^B \mid \boldsymbol{v}' \perp \boldsymbol{q},\ \|\boldsymbol{v}'\|_2 = 1\}$ is closed and bounded in $\mathbb{R}^B$, and hence compact. The function $\|\boldsymbol{L}_{A,:}\boldsymbol{v}'\|_1$ is continuous and strictly positive on this compact set, so its minimum over the set exists and is positive. We therefore define

$$\sigma := \min_{\boldsymbol{v}' \perp \boldsymbol{q},\ \|\boldsymbol{v}'\|_2 = 1} \|\boldsymbol{L}_{A,:}\boldsymbol{v}'\|_1 > 0.$$

For $\boldsymbol{v} \neq \boldsymbol{0}$, $\boldsymbol{v} \perp \boldsymbol{q}$ implies that $\boldsymbol{v}/\|\boldsymbol{v}\|_2$ satisfies the constraints in this minimization. Hence,

$$\sigma \leq \left\| \boldsymbol{L}_{A,:} \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_2} \right\|_1 = \frac{\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1}{\|\boldsymbol{v}\|_2},$$

which gives

$$\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 \geq \sigma \|\boldsymbol{v}\|_2.$$

The same inequality holds trivially when $\boldsymbol{v} = \boldsymbol{0}$.

We finally choose ϵ so that the lower bound in Equation 18 is strictly positive. By Equation 14 and $\sigma > 0$, $(1 - \|\boldsymbol{u}\|_\infty)\sigma > 0$. Moreover, since $\boldsymbol{L}$ has orthonormal columns and $\|\boldsymbol{q}\|_2 = 1$, $\|\boldsymbol{L}\boldsymbol{q}\|_1 \geq \|\boldsymbol{L}\boldsymbol{q}\|_2 = 1$. We choose

$$\epsilon := \min\left\{\sqrt{2}, \frac{(1 - \|\boldsymbol{u}\|_\infty)\sigma}{2\|\boldsymbol{L}\boldsymbol{q}\|_1}\right\} > 0.$$

Consider any $\boldsymbol{q}' \in B_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^{B-1}$ with $\boldsymbol{q}' \neq \boldsymbol{q}$. Since $\|\boldsymbol{q}' - \boldsymbol{q}\|_2 < \epsilon \leq \sqrt{2}$,

$$\alpha = 1 - \frac{1}{2}\|\boldsymbol{q}' - \boldsymbol{q}\|_2^2 > 0.$$

This also implies $\boldsymbol{v} \neq \mathbf{0}$. Otherwise, $\boldsymbol{q}' = \alpha \boldsymbol{q}$, and $\alpha > 0$ together with $\|\boldsymbol{q}'\|_2 = \|\boldsymbol{q}\|_2 = 1$ would imply $\alpha = 1$ and hence $\boldsymbol{q}' = \boldsymbol{q}$, contradicting $\boldsymbol{q}' \neq \boldsymbol{q}$. Using $\alpha^2 + \|\boldsymbol{v}\|_2^2 = 1$ and $\alpha > 0$, we obtain

$$\alpha - 1 = -\frac{\|\boldsymbol{v}\|_2^2}{1+\alpha} \geq -\|\boldsymbol{v}\|_2^2.$$

Moreover, since $(\boldsymbol{I} - \boldsymbol{q}\boldsymbol{q}^\top)\boldsymbol{q} = \mathbf{0}$,

$$\boldsymbol{v} = (\boldsymbol{I} - \boldsymbol{q}\boldsymbol{q}^\top)(\boldsymbol{q}' - \boldsymbol{q}).$$

The matrix $\boldsymbol{I} - \boldsymbol{q}\boldsymbol{q}^\top$ is the orthogonal projection onto the subspace orthogonal to $\boldsymbol{q}$, and therefore

$$\|\boldsymbol{v}\|_2 \leq \|\boldsymbol{q}' - \boldsymbol{q}\|_2.$$

Hence,

$$\|\boldsymbol{v}\|_2 \leq \|\boldsymbol{q}' - \boldsymbol{q}\|_2 < \epsilon \leq \frac{(1 - \|\boldsymbol{u}\|_\infty)\sigma}{2\|\boldsymbol{L}\boldsymbol{q}\|_1},$$

which gives

$$\|\boldsymbol{L}\boldsymbol{q}\|_1 \|\boldsymbol{v}\|_2 < \frac{(1 - \|\boldsymbol{u}\|_\infty)\sigma}{2}.$$

Substituting these bounds into Equation 18 yields

$$\begin{aligned}
\Psi_{\boldsymbol{L}}(\boldsymbol{q}') - \Psi_{\boldsymbol{L}}(\boldsymbol{q}) &\geq \|\boldsymbol{v}\|_2 \left((1 - \|\boldsymbol{u}\|_\infty)\sigma - \|\boldsymbol{L}\boldsymbol{q}\|_1 \|\boldsymbol{v}\|_2\right) \\
&> \frac{1}{2}(1 - \|\boldsymbol{u}\|_\infty)\sigma \|\boldsymbol{v}\|_2 \\
&> 0.
\end{aligned}$$

Hence $\Psi_{\boldsymbol{L}}(\boldsymbol{q}') > \Psi_{\boldsymbol{L}}(\boldsymbol{q})$ for every $\boldsymbol{q}' \in B_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^{B-1}$ with $\boldsymbol{q}' \neq \boldsymbol{q}$. Therefore, $\boldsymbol{q}$ is a strict local minimizer of $\Psi_{\boldsymbol{L}}$.

Necessity. We show that if $\boldsymbol{q}$ is a strict local minimizer, then there exist $\boldsymbol{u}, \lambda$ satisfying Equations 12–14. Since $\boldsymbol{q}$ is a strict local minimizer of $\Psi_{\boldsymbol{L}}$ on $\mathbb{S}^{B-1}$, Definition 1 gives

$$\exists\, \epsilon > 0 \qquad \text{s.t.} \qquad \forall\, \boldsymbol{q}' \in \left(B_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^{B-1}\right) \setminus \{\boldsymbol{q}\}, \qquad \Psi_{\boldsymbol{L}}(\boldsymbol{q}') > \Psi_{\boldsymbol{L}}(\boldsymbol{q}). \tag{19}$$

Using Equation 19, we first prove Equation 12, namely $\operatorname{rank}(\boldsymbol{L}_{A,:}) = B - 1$. By the definition of A, $\boldsymbol{L}_{A,:}\boldsymbol{q} = \mathbf{0}$, and $\boldsymbol{q} \neq \mathbf{0}$ because $\boldsymbol{q} \in \mathbb{S}^{B-1}$. Thus, Equation 12 is equivalent to $\ker(\boldsymbol{L}_{A,:}) = \operatorname{span}(\boldsymbol{q})$, which is in turn equivalent to showing that

$$\boldsymbol{L}_{A,:}\boldsymbol{v} \neq \mathbf{0}$$

for every nonzero $\boldsymbol{v} \perp \boldsymbol{q}$. In the following, we consider an arbitrary unit vector $\boldsymbol{v} \perp \boldsymbol{q}$ without loss of generality.

To show that $\boldsymbol{L}_{A,:}\boldsymbol{v} \neq \mathbf{0}$, we consider a perturbation of $\boldsymbol{q}$ in the direction $\boldsymbol{v}$ that remains on $\mathbb{S}^{B-1}$. We define

$$\boldsymbol{q}(\xi) := \cos\xi\, \boldsymbol{q} + \sin\xi\, \boldsymbol{v}.$$

The orthogonality of $\boldsymbol{q}$ and $\boldsymbol{v}$ gives $\boldsymbol{q}(\xi) \in \mathbb{S}^{B-1}$ for every ξ. For this perturbation, we rewrite $\Psi_{\boldsymbol{L}}(\boldsymbol{q}(\xi)) - \Psi_{\boldsymbol{L}}(\boldsymbol{q})$ so that the term $\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1$ appears explicitly. For $0 < \xi < \pi$,

$$\begin{aligned}
\Psi_{\boldsymbol{L}}(\boldsymbol{q}(\xi)) - \Psi_{\boldsymbol{L}}(\boldsymbol{q}) &= \|\boldsymbol{L}_{A,:}\boldsymbol{q}(\xi)\|_1 + \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}(\xi)\|_1 - \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 \\
&= \sin\xi\, \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 + \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}(\xi)\|_1 - \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1.
\end{aligned} \tag{20}$$

The first equality follows from the definition of $\Psi_{\boldsymbol{L}}$ in Equation 11 and $\boldsymbol{L}_{A,:}\boldsymbol{q} = \mathbf{0}$. The second equality substitutes $\boldsymbol{q}(\xi) = \cos\xi\, \boldsymbol{q} + \sin\xi\, \boldsymbol{v}$ and uses $\sin\xi > 0$.

The remaining difference over A^c in Equation 20 is not yet expressed explicitly in $\boldsymbol{v}$. To rewrite it, we choose an upper bound $\bar{\xi}$ for ξ. Since $\boldsymbol{L}$ has orthonormal columns and $\|\boldsymbol{q}\|_2 = 1$, $\|\boldsymbol{L}\boldsymbol{q}\|_2 = 1$, so A^c is nonempty. We set

$$\bar{\xi} := \min\left\{\epsilon, \pi, \frac{\min_{j \in A^c} |\boldsymbol{L}_{j,:}\boldsymbol{q}|}{\|\boldsymbol{L}_{A^c,:}\|_F}\right\} > 0.$$

For any $0 < \xi < \bar{\xi}$,

$$\|\boldsymbol{q}(\xi) - \boldsymbol{q}\|_2 = 2\sin\frac{\xi}{2} \le \xi < \epsilon,$$

so $\boldsymbol{q}(\xi) \in B_\epsilon(\boldsymbol{q})$. Moreover, for every $j \in A^c$,

$$|\boldsymbol{L}_{j,:}(\boldsymbol{q}(\xi) - \boldsymbol{q})| \le \|\boldsymbol{L}_{j,:}\|_2 \|\boldsymbol{q}(\xi) - \boldsymbol{q}\|_2 \le \|\boldsymbol{L}_{A^c,:}\|_F \xi < \min_{k \in A^c} |\boldsymbol{L}_{k,:}\boldsymbol{q}| \le |\boldsymbol{L}_{j,:}\boldsymbol{q}|.$$

Hence, the signs remain the same, which means that

$$\operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q}(\xi)) = \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q}). \tag{21}$$

Using Equation 21, we can rewrite Equation 20 as

$$\begin{aligned}
\Psi_{\boldsymbol{L}}(\boldsymbol{q}(\xi)) - \Psi_{\boldsymbol{L}}(\boldsymbol{q}) &= \sin\xi\, \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 + \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \boldsymbol{L}_{A^c,:}\big(\boldsymbol{q}(\xi) - \boldsymbol{q}\big) \\
&= (\cos\xi - 1)\|\boldsymbol{L}\boldsymbol{q}\|_1 + \sin\xi\, \big(\operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \boldsymbol{L}_{A^c,:}\boldsymbol{v} + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1\big) \\
&= (\cos\xi - 1)\|\boldsymbol{L}\boldsymbol{q}\|_1 + \sin\xi\, \left(\langle \boldsymbol{r}, \boldsymbol{v}\rangle + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1\right),
\end{aligned} \tag{22}$$

where $\boldsymbol{r} := \boldsymbol{L}_{A^c,:}^\top \operatorname{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})$. The second equality uses $\boldsymbol{q}(\xi) - \boldsymbol{q} = (\cos\xi - 1)\boldsymbol{q} + \sin\xi\, \boldsymbol{v}$ and $\|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 = \|\boldsymbol{L}\boldsymbol{q}\|_1$, where the latter follows from $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$.

For every $0 < \xi < \bar{\xi}$, we have $\boldsymbol{q}(\xi) \in B_\epsilon(\boldsymbol{q}) \cap \mathbb{S}^{B-1}$ and $\boldsymbol{q}(\xi) \neq \boldsymbol{q}$. Thus, Equation 19 with $\boldsymbol{q}' = \boldsymbol{q}(\xi)$ and Equation 22 give

$$\sin\xi\, \left(\langle \boldsymbol{r}, \boldsymbol{v}\rangle + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1\right) > (1 - \cos\xi)\|\boldsymbol{L}\boldsymbol{q}\|_1 > 0.$$

Since $\sin\xi > 0$ for $0 < \xi < \bar{\xi} \le \pi$, dividing by $\sin\xi$ gives

$$\langle \boldsymbol{r}, \boldsymbol{v}\rangle + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 > 0.$$

The bound $\bar{\xi}$ does not depend on $\boldsymbol{v}$, so the same argument with $-\boldsymbol{v}$ gives

$$-\langle \boldsymbol{r}, \boldsymbol{v}\rangle + \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 > 0.$$

Combining the two inequalities gives

$$|\langle \boldsymbol{r}, \boldsymbol{v}\rangle| < \|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1. \tag{23}$$

Since $|\langle \boldsymbol{r}, \boldsymbol{v}\rangle| \ge 0$, Equation 23 requires $\|\boldsymbol{L}_{A,:}\boldsymbol{v}\|_1 > 0$, and hence $\boldsymbol{L}_{A,:}\boldsymbol{v} \neq \boldsymbol{0}$. Since $\boldsymbol{v} \perp \boldsymbol{q}$ was an arbitrary unit vector, the same holds for every nonzero $\boldsymbol{v} \perp \boldsymbol{q}$ by normalization. Therefore, $\ker(\boldsymbol{L}_{A,:}) = \operatorname{span}(\boldsymbol{q})$ and $\operatorname{rank}(\boldsymbol{L}_{A,:}) = B - 1$, which establishes Equation 12.

We next construct $\boldsymbol{u}$ and λ satisfying Equation 13 and Equation 14. By the definition of $\boldsymbol{r}$, Equation 13 can be written as

$$\boldsymbol{r} + \boldsymbol{L}_{A,:}^\top \boldsymbol{u} + \lambda \boldsymbol{q} = \boldsymbol{0}.$$

To satisfy this equality together with $\|\boldsymbol{u}\|_\infty < 1$ in Equation 14, we consider it separately in the two orthogonal subspaces $\boldsymbol{q}^\perp$ and $\operatorname{span}(\boldsymbol{q})$. We first choose $\boldsymbol{u}$ with $\|\boldsymbol{u}\|_\infty < 1$ so that the equality holds in $\boldsymbol{q}^\perp$, and then choose λ so that it also holds in $\operatorname{span}(\boldsymbol{q})$.

For the $\boldsymbol{q}^\perp$ part, let $\boldsymbol{\Pi} := \boldsymbol{I} - \boldsymbol{q}\boldsymbol{q}^\top$ denote the orthogonal projection onto $\boldsymbol{q}^\perp$. Applying $\boldsymbol{\Pi}$ to the equality gives

$$\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = -\boldsymbol{\Pi}\boldsymbol{r}.$$

We used $\boldsymbol{\Pi}\boldsymbol{q} = \boldsymbol{0}$ and $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$, which gives $\boldsymbol{\Pi}\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = \boldsymbol{L}_{A,:}^\top \boldsymbol{u}$. By Equation 12 and $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$, $\operatorname{range}(\boldsymbol{L}_{A,:}^\top) = \ker(\boldsymbol{L}_{A,:})^\perp = \boldsymbol{q}^\perp$. Thus, every vector in $\boldsymbol{q}^\perp$ can be represented as $\boldsymbol{L}_{A,:}^\top \boldsymbol{u}$ for some $\boldsymbol{u}$. Since $-\boldsymbol{\Pi}\boldsymbol{r} \in \boldsymbol{q}^\perp$, the projected equality can be satisfied.

We next show that it can be satisfied with $\|\boldsymbol{u}\|_\infty < 1$. It suffices to show that

$$\min_{\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = -\boldsymbol{\Pi}\boldsymbol{r}} \|\boldsymbol{u}\|_\infty < 1.$$

Introducing $\boldsymbol{w}$ as the dual variable for the equality constraint, the corresponding dual problem and strong duality give

$$\min_{\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = -\boldsymbol{\Pi}\boldsymbol{r}} \|\boldsymbol{u}\|_\infty = \max_{\|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1 \le 1} \langle \boldsymbol{\Pi}\boldsymbol{r}, \boldsymbol{w}\rangle. \tag{24}$$

Strong duality applies because the primal problem is feasible, as shown above, and $\|\cdot\|_\infty$ is convex and continuous on its full domain.

For any $\boldsymbol{w} \in \mathbb{R}^B$, write $\boldsymbol{w} = \boldsymbol{w}_\perp + \alpha \boldsymbol{q}$ with $\boldsymbol{w}_\perp \perp \boldsymbol{q}$. Since $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$ and $\boldsymbol{\Pi r} \perp \boldsymbol{q}$, we have $\boldsymbol{L}_{A,:}\boldsymbol{w} = \boldsymbol{L}_{A,:}\boldsymbol{w}_\perp$ and $\langle \boldsymbol{\Pi r}, \boldsymbol{w} \rangle = \langle \boldsymbol{\Pi r}, \boldsymbol{w}_\perp \rangle$. Therefore, adding the constraint $\boldsymbol{w} \perp \boldsymbol{q}$ does not change the optimal value of the dual problem, and

$$\max_{\|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1 \le 1} \langle \boldsymbol{\Pi r}, \boldsymbol{w} \rangle = \max_{\substack{\boldsymbol{w} \perp \boldsymbol{q} \\ \|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1 \le 1}} \langle \boldsymbol{\Pi r}, \boldsymbol{w} \rangle.$$

For any nonzero feasible $\boldsymbol{w}$ in the right-hand problem, $\boldsymbol{w}/\|\boldsymbol{w}\|_2$ is a unit vector orthogonal to $\boldsymbol{q}$. Applying Equation 23 to this vector and multiplying by $\|\boldsymbol{w}\|_2$ gives

$$|\langle \boldsymbol{r}, \boldsymbol{w} \rangle| < \|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1.$$

Since $\boldsymbol{w} \perp \boldsymbol{q}$ and $\boldsymbol{w}$ satisfies the dual constraint,

$$\langle \boldsymbol{\Pi r}, \boldsymbol{w} \rangle \le |\langle \boldsymbol{\Pi r}, \boldsymbol{w} \rangle| = |\langle \boldsymbol{r}, \boldsymbol{w} \rangle| < \|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1 \le 1.$$

By Equation 12, $\boldsymbol{L}_{A,:}$ is injective on $\boldsymbol{q}^\perp$, so $\|\boldsymbol{L}_{A,:}\boldsymbol{w}\|_1$ defines a norm on $\boldsymbol{q}^\perp$. Its unit ball is compact, so the maximum above is attained. The objective is strictly smaller than 1 at every nonzero feasible point and equals 0 at $\boldsymbol{w} = \boldsymbol{0}$, so the maximum is strictly smaller than 1. By Equation 24, there exists $\boldsymbol{u}$ satisfying

$$\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = -\boldsymbol{\Pi r}, \qquad \|\boldsymbol{u}\|_\infty < 1.$$

Thus, Equation 14 holds.

We now consider the component in $\mathrm{span}(\boldsymbol{q})$. Since $\boldsymbol{u}$ was chosen to satisfy $\boldsymbol{L}_{A,:}^\top \boldsymbol{u} = -\boldsymbol{\Pi r}$, the component in $\boldsymbol{q}^\perp$ has already been canceled. The remaining component is

$$\boldsymbol{r} + \boldsymbol{L}_{A,:}^\top \boldsymbol{u} = \boldsymbol{r} - \boldsymbol{\Pi r} = \boldsymbol{q}\boldsymbol{q}^\top \boldsymbol{r} = (\boldsymbol{q}^\top \boldsymbol{r})\boldsymbol{q}.$$

By the definition of $\boldsymbol{r}$,

$$\boldsymbol{q}^\top \boldsymbol{r} = \mathrm{sign}(\boldsymbol{L}_{A^c,:}\boldsymbol{q})^\top \boldsymbol{L}_{A^c,:}\boldsymbol{q} = \|\boldsymbol{L}_{A^c,:}\boldsymbol{q}\|_1 = \|\boldsymbol{L}\boldsymbol{q}\|_1,$$

where the last equality uses $\boldsymbol{L}_{A,:}\boldsymbol{q} = \boldsymbol{0}$. Therefore, choosing $\lambda := -\|\boldsymbol{L}\boldsymbol{q}\|_1$ gives

$$\boldsymbol{r} + \boldsymbol{L}_{A,:}^\top \boldsymbol{u} + \lambda \boldsymbol{q} = \boldsymbol{0},$$

which is Equation 13. Together with Equation 12 and Equation 14, this completes the necessity proof.

F Experimental Details

F.1 Additional Experimental Setup

We provide implementation details omitted from Section 5. All other experimental settings are as described there.

Computing environment. Experiments were run on NVIDIA RTX A6000 and RTX 6000 Ada GPUs.

Data partitioning. For each run, we uniformly permuted all training samples and partitioned them among 128 clients. Each client retained the first B assigned samples as its local dataset, which was fixed across communication rounds. No data augmentation was applied to the datasets.

Candidate search. Following SPEAR++ (Bakarsky et al., 2025), we solved the ℓ_1-minimizations in Equation 4 for SPEAR++ and Equation 10 for Multi-Round SPEAR++ from random initializations on the unit sphere using Riemannian Adam with the same optimization hyperparameters. Specifically, we ran 500 iterations with an initial step size of 10^{-1}, reduced to 10^{-3} and 10^{-5} after 200 and 400 iterations, respectively.

To filter the resulting candidate directions, we set the sparsity threshold τ so that the false-rejection probability was at most 10^{-5} under the binomial model used in SPEAR (Dimitrov et al., 2024). This gave $\tau = 0.350, 0.404, 0.433$ and 0.453 for $m = 200$, 500, 1000 and 2000, respectively. For Multi-Round SPEAR++, the same threshold was applied separately to the m-row block corresponding to each observed round. In finite-precision computation, entries that are zero in exact arithmetic can take small nonzero values due to numerical errors. Thus, we treated an entry as zero if its absolute value was at most 10^{-5} when evaluating the ℓ_0-norms in the sparsity filtering. After sparsity filtering, we applied the duplicate filter with $\varepsilon_{\mathrm{dup}} = 10^{-2}$.

Evaluation. We matched the reconstructed inputs to the ground-truth inputs using Hungarian matching based on the squared error in the normalized input space. For each matched pair of a reconstructed input $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ and its corresponding ground-truth input $\boldsymbol{x} \in \mathbb{R}^n$, we de-normalized both inputs to the original pixel scale and computed PSNR as

$$\mathrm{PSNR}(\hat{\boldsymbol{x}}, \boldsymbol{x}) = 10 \log_{10} \left(\frac{R^2}{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2 / n} \right),$$

where $R = 1$ is the reference pixel range and the vectors in the equation are the de-normalized inputs.

F.2 Additional Experimental Results

Results on MNIST. For MNIST, we used $\eta = 0.05$ for both $K = 1$ and $K = 4$. Figure 4 shows a different trend from CIFAR-10. Multi-Round SPEAR++ does not uniformly outperform SPEAR++, and SPEAR++ achieves higher full recovery rates in some settings. Nevertheless, Multi-Round SPEAR++ performs better in several regimes, including $m = 200$ for both $K = 1$ and $K = 4$ and at larger local data sizes for $K = 1$ with $m = 500$ and $m = 1000$, where it maintains a nonzero full recovery rate after SPEAR++ drops to zero.

Effect of the number of observed rounds. Figure 5 shows that exploiting multiple observed communication rounds can substantially improve the full recovery rate at larger local data sizes. For $m = 1000$ and $B = 128$, $\mathcal{T} = [2]$ provides only limited improvement over $\mathcal{T} = [1]$, whereas $\mathcal{T} = [5]$ and $\mathcal{T} = [10]$ achieve much higher full recovery rates. However, the improvement is not necessarily monotonic in the number of observed rounds. At $m = 1000$ and $B = 160$, $\mathcal{T} = [5]$ achieves a higher full recovery rate than $\mathcal{T} = [10]$.

Reconstruction examples. Figure 6 shows that the inputs reconstructed by Multi-Round SPEAR++ are visually indistinguishable from the ground-truth inputs. In contrast, SPEAR++ produces both clearly incorrect and partially recognizable reconstructions. Separately, Figure 3 categorizes cases in which a true column direction is found but full reconstruction still fails as completeness failures (F4), reflecting failures to recover the complete set of required candidates and their signed scales.

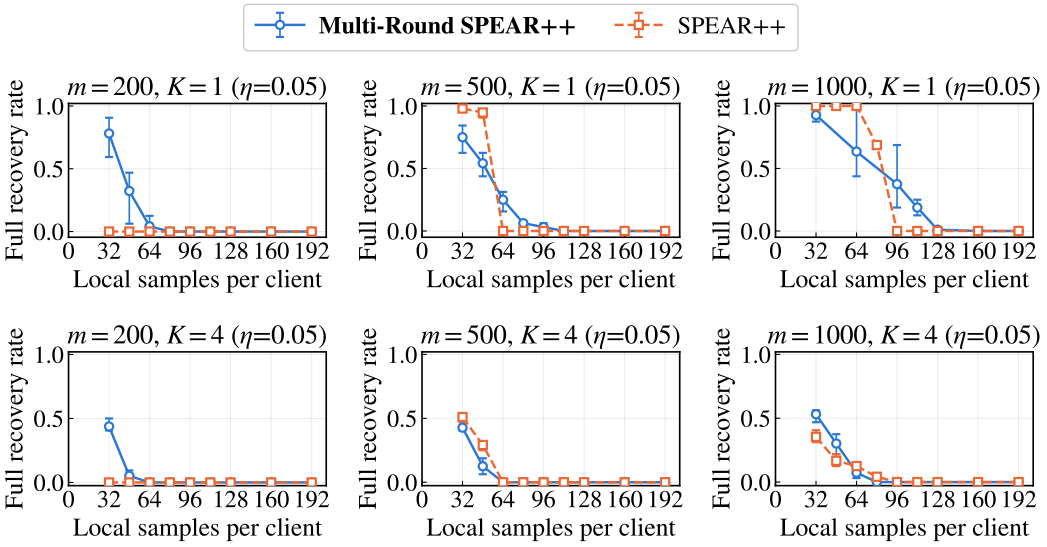

Figure 4: Full recovery rate versus local data size B for different numbers m of output neurons in the first layer on MNIST. SPEAR++ uses the model update from the first communication round, whereas Multi-Round SPEAR++ uses the model updates from the first 10 communication rounds.

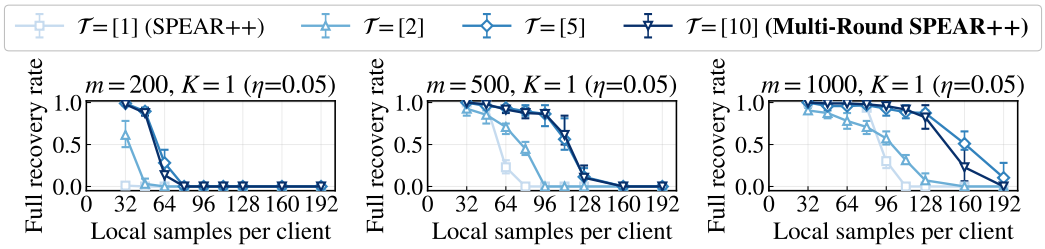

Figure 5: Full recovery rate versus local data size B for different numbers of observed communication rounds on CIFAR-10. We use $\mathcal{T} = [r]$ with $r \in \{1, 2, 5, 10\}$, where $\mathcal{T} = [1]$ corresponds to SPEAR++ and $\mathcal{T} = [10]$ to the default setting of Multi-Round SPEAR++.

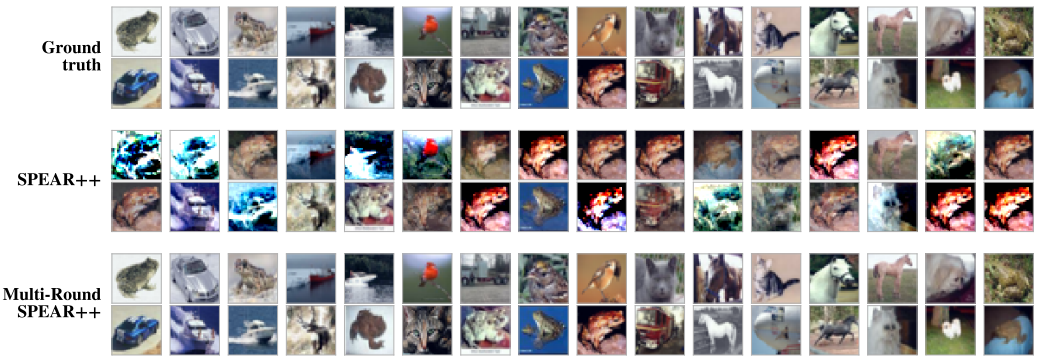

Figure 6: Ground-truth inputs and reconstructed inputs on CIFAR-10 for $K = 1$, $m = 200$, and $B = 32$. The 32 ground-truth inputs and the corresponding reconstructions from each method are each arranged in two rows of 16. The reconstructed inputs are reordered by Hungarian matching based on the squared error in the normalized input space so that each reconstruction is shown at the same position as its matched ground-truth input. SPEAR++ uses the model update from the first communication round, whereas Multi-Round SPEAR++ uses the model updates from the first 10 communication rounds.