Approaching the Harm of Gradient Attacks While Only Flipping Labels

Anonymous authors
Paper under double-blind review

Abstract

Machine learning systems deployed in distributed or federated environments are highly susceptible to adversarial manipulations, particularly availability attacks – rendering the trained model unavailable. Prior research in distributed ML has demonstrated such adversarial effects through the injection of gradients or data poisoning. In this work, we ask whether comparable degradation is still possible under a substantially more constrained action space: the adversary may only flip a limited number of labels of existing training examples, without modifying features, injecting samples, or directly controlling gradients. We analyze the extent of damage caused by constrained label flipping attacks against distributed learning under mean aggregation – the dominant baseline in research and production. Focusing on classification problems, (1) we propose a novel formalization of label flipping attacks as a per-round constrained optimization problem, derive a greedy label-selection rule for logistic regression, and empirically evaluate it beyond its derivation setting, including on MLPs and robust aggregators. The rule is provably per-epoch optimal for the attacker under the mean aggregator. (2) Empirically, we show that optimized label flipping can cause substantial accuracy degradation while outperforming random label flipping under similar budgets. (3) We shed light on an interesting interplay between what the attacker gains from more *write-access* versus what they gain from more *flipping budget*. (4) Finally, although the attack is derived for mean aggregation, we find that it can transfer empirically to the coordinate-wise median and trimmed mean aggregators, where its effectiveness is comparable to the Little-is-Enough gradient attack. This demonstrates that even highly constrained label-flipping adversaries can pose a significant availability threat to distributed learning.

1 Introduction and Related Work

Machine learning systems can become prime targets for adversarial attacks. *Training-phase poisoning attacks* in particular have gained considerable attention as the widespread use of machine learning in critical applications has grown (Awasthi et al. (2017); Zhang et al. (2017); Paudice et al. (2018); Lu et al. (2022); Huang et al. (2011)). In these attacks, an adversary manipulates the training data in order to degrade or control the final trained model. Unlike evasion and backdoor attacks, poisoning requires no control over inference-time input: it suffices to manipulate part of the training set. Among such attacks, *label flipping* stands out for its simplicity: The attacker simply changes the class label of a subset of training points while leaving other aspects of the data intact. This is especially relevant in federated learning situations where features are fixed by upstream data pipelines and the set of possible labels is predefined. For example, a common scenario is when workers

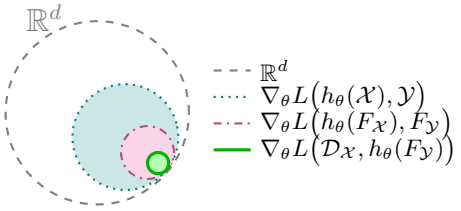


Figure 1: Images of the gradient operator on different sets. $\mathbb{R}^d$ is where an attacker can craft unrestricted gradient attacks. Given a model h_θ, $\nabla_\theta L(h_\theta(\mathcal{X}), \mathcal{Y})$ is the set of possible gradients given an unrestricted data poisoning Bouaziz et al. (2024), $\nabla_\theta L(h_\theta(\mathcal{F}_\mathcal{X}), \mathcal{F}_\mathcal{Y})$ is the set of possible gradients when data poisoning is restricted to a feasible set $\mathcal{F}_\mathcal{X} \times \mathcal{F}_\mathcal{Y} \subseteq \mathcal{X} \times \mathcal{Y}$, and $\nabla_\theta L\big(\mathcal{D}_\mathcal{X}, h_\theta(F_\mathcal{Y})\big)$ is the set of possible gradients when the features are restricted to those in the dataset $\mathcal{D}_\mathcal{X}$ and the labels are chosen in the set of feasible labels.

are asked to label predefined images, the labels being in a finite set. We address the following question in the context of a classification problem:

Can an attacker severely degrade a model using only label flips on existing data and under budget constraints?

In this work, we provide a positive answer to this question. By viewing label flipping as a constrained optimization problem from the point of view of the attacker, we show that carefully selected flips can steer the aggregated gradient away from its honest direction and reduce accuracy or render training unstable - even when only 1% of the labels are altered.

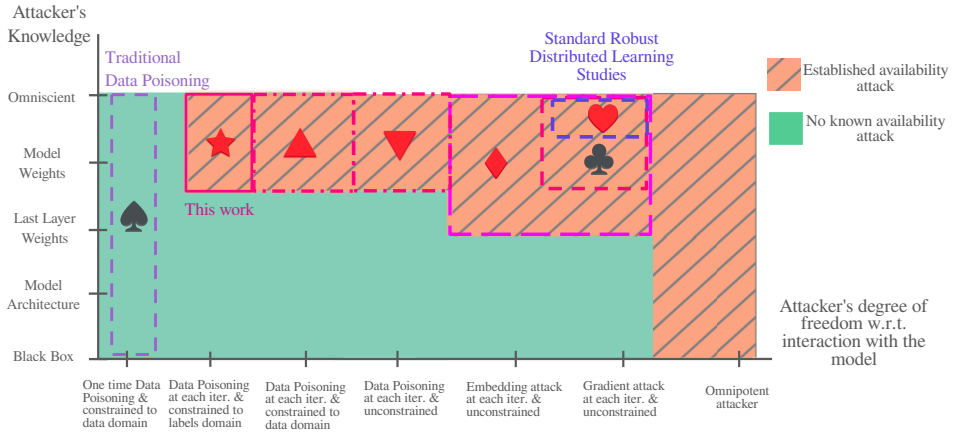


Figure 2: Territory of known availability attacks (in orange) within a domain of constraints. The closer to the origin, the more constrained is the setting for the attacker and the harder it is to realize an availability attack. ♠: Geiping et al. (2020); Zhao & Lao (2022); Ning et al. (2021); Huang et al. (2020), ♡: Blanchard et al. (2017); Baruch et al. (2019), ♣: El-Mhamdi et al. (2018), ♢ so far only in convex settings : Farhadkhani et al. (2022), △&▽ : Bouaziz et al. (2024), ★ : Our contribution.

Previous work has investigated poisoning attacks and granted the adversary far more leverage than some realistic scenarios allow. Some require the ability to modify *both* labels and features or even overwrite entire gradients (Bouaziz et al. (2024); Baruch et al. (2019)); others rely on injecting crafted examples into the training set (Shafahi et al. (2018); Zhao & Lao (2022); Koh et al. (2022)). Li et al. (2022) proposes an agglomerative-clustering strategy to select vulnerable samples, assuming access to all training data and performing offline clustering. Jha et al. (2023) demonstrates that carefully chosen label poisoning can insert backdoors via trajectory-matching methods. Their objective is to create targeted misclassification when a trigger is present, not to reduce overall model accuracy. Thus, although both their work and ours manipulate labels only, the goals, threat models, and evaluation metrics differ: their work aims to insert backdoors, whereas we study availability degradation under a tight budget and online setting. Likewise, Lavaur et al. (2024) applies random label flipping in the context of collaborative intrusion detection systems. The attack of Lavaur et al. (2024) targets domain-specific labels (e.g., DDoS) and does not formalize the attack as an optimization problem. In contrast, we aim to study an attack that is data- and model-agnostic but can be proved to be optimal at each iteration for the considered objective.

Yu et al. (2025) study label-flipping attacks on GNNs under a federated learning setup. Their approach is oriented towards graph-structured data and relies on the unique properties of GNN aggregation mechanisms. In contrast, we propose a method that is formulated as a general optimization framework based on gradient alignment, making it applicable to a wider range of classification tasks. When manipulation is limited to labels alone, existing methods typically presuppose control over the vast majority of them ($\geq 85\%$) or access to the validation set, and require the training of a number of classifiers linear in the number of classes and the budget (Liu et al. (2023a); Paudice et al. (2018)), unsuitable for online or real-time training, especially at scale.

Another recent work, Bal et al. (2025), focuses on *defense* in the binary classification setting via adversarial training (SVMs, untargeted), whereas our work formulates the *attacker's* online, budget-constrained optimization and proves per-epoch greedy optimality under explicit write-access and local budget constraints. Furthermore, our attack requires only control over a small fraction of data and changes no features at all, only labels. This expands the scope of known *availability attacks* to a more constrained, yet practically damaging, threat model.

Following Bouaziz et al. (2024), Figure 2 shows label-flipping attacks on the landscape of availability attacks, situating our contribution within the literature. Meanwhile, Figure 1 illustrates how label flipping compares to more general gradient-based attacks, particularly with respect to the set of gradients achievable under increasingly restrictive conditions.

Contributions. Our contributions are as follows. First, we formalize label flipping for logistic regression as a budget-constrained optimization problem for distributed classification under mean aggregation (Section 2). Second, we derive a greedy label-selection rule and prove that it is *optimal for the attacker at each training step* under the mean-aggregation objective (Section 3). Experiments on standard scaled image classification benchmarks show that optimized label flipping causes substantially stronger availability degradation than random label flipping under matched budgets (Section 4), with the added bonus of bypassing typical norm-based anomaly detection thresholds (Appendix C). We also discuss how the attack depends on the decomposition of the global corruption rate $\rho = kb$, showing that broader write access k can be more damaging than a larger local flipping budget b. We further provide empirical evidence that the attack transfers beyond mean aggregation to coordinate-wise median and trimmed-mean aggregations, where constrained greedy label flipping is comparable in effectiveness to the celebrated Little-is-Enough gradient attack.

2 General Setting

2.1 Notation

Table 1: Notation Summary

Notation	Description
t	Training-round, epoch, or iteration index.
$x_i \in \mathbb{R}^{d+1}$	Feature vector of example i, including a bias coordinate.
$y_i^H \in \mathcal{Y}$	Clean label of example i before poisoning.
$\tilde{y}_i \in \mathcal{Y}$	Label submitted by the adversary after possible flipping.
θ_t	Model parameter at round t. For binary logistic regression, $\theta_t = \alpha_t \in \mathbb{R}^{d+1}$; for multi-class softmax regression, $\theta_t = W_t \in \mathbb{R}^{C\times(d+1)}$.
H_t	Honest examples at round t, whose labels are not controlled by the adversary.
K_t^H	Clean version of the examples controlled by the adversary before label flipping.
$K_t(\tilde{y})$	Attacker-controlled examples after label flipping.
$D_t^H = H_t \cup K_t^H$	Clean batch before poisoning.
$D_t(\tilde{y}) = H_t \cup K_t(\tilde{y})$	Batch observed by the server after poisoning.
$N = \lvert D_t^H\rvert = \lvert D_t(\tilde{y})\rvert$	Batch size.
$k_t = \lvert K_t^H\rvert / \lvert D_t^H\rvert$	Attacker's write-access fraction at round t. When fixed across rounds, we write k.
$b \in [0,1]$	Local flipping budget: the fraction of labels in K_t^H that may be changed at round t.
$\rho = kb$	Global corruption budget: the maximum fraction of the full batch whose labels are flipped.
g_t^H	Clean gradient that would be computed on D_t^H.
$g_t(\tilde{y})$	Poisoned gradient computed on $D_t(\tilde{y})$.
Δ_t	Reference direction used by the adversary in the directional label-flipping objective.
$\mathbb{1}[\cdot]$	Indicator function.

The main notation used throughout the paper is summarized in Table 1. Although the batch, the model parameters, and the attack all vary with the epoch t, we omit the epoch index whenever there is no risk of ambiguity. This is because the attacker's optimization problem is treated epoch-wise: at each epoch, the attacker observes the current training state, chooses a feasible subset of labels to flip, and the server then performs its update using the resulting poisoned batch.

We use $i \in D_t$ and $(x_i, y_i) \in D_t$ interchangeably when referring to data points by index. Similarly, writing $y_i \in K_t$ refers to the labels of the data points contained in K_t.

2.2 Threat Model

Rationale. As noted in Bouaziz et al. (2024), the fundamental difference between gradient attacks and data poisoning comes from the limited expressivity of the latter. A gradient attacker may directly craft an arbitrary vector in parameter space, while a data-poisoning attacker can only influence the model through gradients induced by feasible data points. This difference is even more pronounced for label-flipping.

Our goal is to understand how much availability damage remains possible under this strongly restricted action space. We therefore study an adversary that can only flip labels of existing data, under a strict per-epoch budget. The attacker cannot modify features, inject new examples, delete honest data, directly overwrite gradients, or change the server-side optimizer. In this sense, the attacker is weaker than classical gradient or model-poisoning adversaries in terms of write actions.

Following the two works most closely related to ours Farhadkhani et al. (2022); Bouaziz et al. (2024), we also allow the malicious worker to recalculate its attack at each iteration, similarly to Algorithm 1 in Steinhardt et al. (2017).

General Learning Setup. We study a classification problem where the training dataset is denoted by $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{n}$, with samples drawn from a distribution $\mathcal{D}$ over the input–label space $\mathcal{X} \times \mathcal{Y}$. The learner trains a neural network h_θ, parameterized by $\theta \in \mathbb{R}^d$, using an iterative optimization procedure to minimize a loss function $\mathcal{L}$. The objective is to achieve low test loss on a held-out dataset $\mathcal{D}_{\text{test}}$.

We model the learning process as involving n_w Gradient Generation Units (GGUs). At each iteration t, each unit produces a message, and these messages are aggregated using a function Agg, and the model parameters are then updated using that aggregated vector. Figure 3 illustrates this setup.

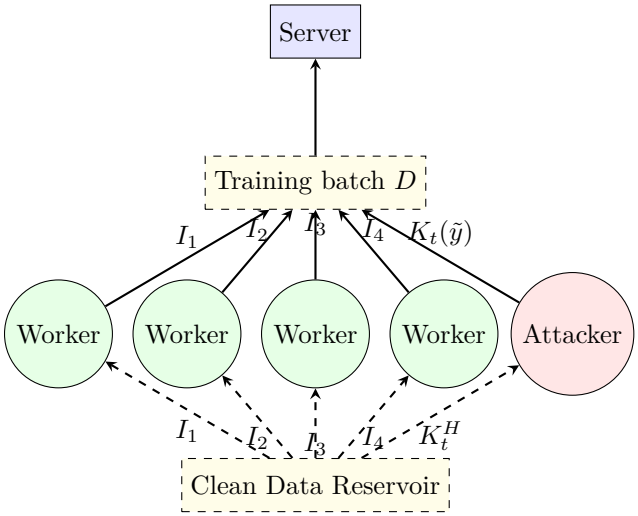


Figure 3: Illustration of the setting: Each user obtains its data from a clean reservoir. The malicious user flips up to a budget b fraction of the labels in K_t^H.

This abstraction covers centralized mini-batch training, distributed synchronous training, and simplified federated-learning settings in which client updates are averaged. Surprisingly, although our theoretical derivation is specific to the mean operator, our attack shows good performance against coordinate-wise median and trimmed mean aggregators.

Attacker's write access. Within this distributed framework, consider that at every training epoch t, a malicious actor is hidden among the workers and contributes a subset of examples to the final batch. Let $K_t^H \subset D_t^H$ denote the clean examples received by the compromised worker before label manipulation, and let $k_t = \frac{|K_t^H|}{|D_t^H|}$ denote the adversary's write-access fraction at round t. Unless otherwise stated, we use k for a fixed write-access rate.

The adversary may replace the clean label y_i^H of a controlled point $i \in K_t^H$ by another feasible label $\tilde{y}_i \in \mathcal{Y}$. However, the adversary cannot modify the feature vector x_i, inject new examples, remove honest examples, directly overwrite gradients, or change the server-side optimizer. The corrupted set sent to the learner is

$$K_t(\tilde{y}) = \{(x_i, \tilde{y}_i) : i \in K_t^H\}, \qquad D_t(\tilde{y}) = H_t \cup K_t(\tilde{y}).$$

Local and global budgets. The adversary is constrained by a per-round local flipping budget $b \in [0, 1]$:

$$\sum_{i \in K_t^H} \mathbf{1}[\tilde{y}_i \neq y_i^H] \leq \lfloor b|K_t^H| \rfloor.$$

Thus, at most a fraction b of the labels under adversarial write access can be changed at any round. The total corrupted fraction of the full batch is at most $\rho = kb$, which we call the global corruption budget. Analyzing the interplay between k and b is of interest because for the same product kb, it is important to know which is preferable: a small adversarial region with many flips and a broad adversarial region.

On the weak adversary Throughout this work, *weaker adversary* refers strictly to an action-constrained adversary: the attacker may only flip labels (under a strict budget, no feature modification, no arbitrary gradient injection). Prior work on gradient-based attacks Bouaziz et al. (2024); Baruch et al. (2019) assumes that the adversary is omniscient, therefore, we allow the attacker to be *omniscient*: they have full read-access to the model parameters at every epoch. We adopt an omniscient adversary to align with prevalent threat models in gradient-based poisoning/Byzantine-robust literature Chen et al. (2017); El-Mhamdi et al. (2018); Xie et al. (2018). For the more realistic limited-knowledge case, Appendix B.5 discusses how an attacker can conceptually train a local surrogate on its accessible subset K_t^H and apply the same greedy selection there.

2.3 Label Flipping as a Constrained Optimization Problem

Let $L_{D_H}(\theta_t) = \frac{1}{|D_t^H|} \sum_{i \in D_t^H} \ell(\theta_t; x_i, y_i^H)$ be the clean empirical risk at round t, and let $g_t^H = \nabla L_{D_H}(\theta_t)$ be the clean gradient. Given a label assignment $\tilde{y}$ satisfying the budget constraint, define the poisoned gradient

$$g_t(\tilde{y}) = \frac{1}{|D_t^H|} \left(\sum_{i \in H_t} \nabla_\theta \ell(\theta_t; x_i, y_i^H) + \sum_{i \in K_t^H} \nabla_\theta \ell(\theta_t; x_i, \tilde{y}_i) \right).$$

At epoch t, the server would normally evaluate the empirical loss on the *honest* batch and update the model with the corresponding gradient $\nabla L_{D_H}(\theta_t)$. However, once the man-in-the-middle adversary flips some labels, the server instead observes the *poisoned* gradient $g_t(\tilde{y})$ where ℓ is the per-sample cross-entropy loss.

The long-horizon availability objective is $\max_{\tilde{y}_0, \ldots, \tilde{y}_{T-1}} \mathcal{E}(\theta_T)$, where $\mathcal{E}$ is a clean validation loss and θ_T is the model obtained after training under the sequence of poisoned gradients.

This objective is combinatorial and depends on the entire future trajectory. We therefore use a tractable one-step surrogate: at each round, the attacker chooses labels so that the poisoned gradient is poorly aligned with a chosen reference direction Δ_t. Formally, the attacker solves

$$\arg \min_{\{\tilde{y}_i\}_{i \in K_t^H}} \quad \langle \Delta_t,\ g_t(\tilde{y}) \rangle \tag{1}$$

$$\text{s.t.} \quad \underbrace{\sum_{i \in K_t^H} \mathbf{1}\big[\tilde{y}_i \neq y_i^H\big] \leq \lfloor b|K_t^H| \rfloor}_{\text{local flipping budget}} \tag{BC}$$

$$\tilde{y}_i \in \mathcal{Y}, \qquad \forall i \in K_t^H.$$

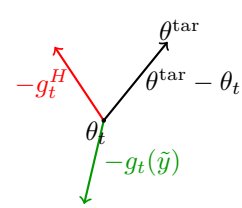


Figure 4: Desired training step direction of honest workers (red), and that of the targeted (black) and untargeted (green) attackers.

The reference direction Δ_t specifies the attacker's goal. We distinguish two goals:

$$\Delta_t = \begin{cases} g_t^H, & \text{untargeted attack,} \\ \theta^{\text{Target}} - \theta_t, & \text{target-parameter attack.} \end{cases} \tag{2}$$

The untargeted adversary tries to deviate from the honest gradient, whereas the targeted adversary steers the update toward a pre-selected parameter vector θ^{Target}. Figure 4 illustrates how the targeted variant uses the vector $(\theta^{\text{Target}} - \theta_t)$ to bias each gradient step toward θ^{Target}.

Since the problem is solved independently at each step, we omit the temporal index in the remainder of the paper. *In the rest of our work, we focus on untargeted attacks* and address Problem 1 for multi-class softmax regression.

3 Multi-Class Softmax Regression

We now solve Problem 1 in the general classification setting. Let $\mathcal{Y} = \{1, \dots, C\}$, and let $W_t \in \mathbb{R}^{C\times(d+1)}$ be the current parameter matrix. For each example $x^{(n)}$, define

$$p^{(n)} = \text{softmax}(W_t x^{(n)}) \in \mathbb{R}^C, \qquad p_j^{(n)} = \text{softmax}(W_t x^{(n)})_j.$$

We encode labels using a one-hot matrix $T \in \{0,1\}^{C\times N}$, where

$$T_{cn} = 1 \quad \Longleftrightarrow \quad \text{example } n \text{ is assigned label } c.$$

Thus each column of T satisfies $\sum_{c=1}^{C} T_{cn} = 1$.

The cross-entropy loss on a batch D can be written as

$$L_D(W_t; T) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} T_{cn} \log p_c^{(n)}.$$

The gradient with respect to the j-th row of W_t is

$$\begin{aligned} \nabla_{W_j} L_D(W_t; T) &= -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} \mathbf{T}_{cn} \left(\mathbb{1}[c = j] - p_j^{(n)} \right) \mathbf{x}^{(n)} \\ &= \frac{1}{N} \sum_{n=1}^{N} \left(p_j^{(n)} - T_{jn} \right) x^{(n)} \end{aligned}$$

Let $\Delta_t \in \mathbb{R}^{C\times(d+1)}$ be the reference direction from Problem 1. Recall

$$\Delta_t = \begin{cases} g_t^H, & \text{untargeted attack,} \\ W^{\text{tar}} - W_t, & \text{targeted attack.} \end{cases}$$

The inner product is now the Frobenius inner product:

$$\langle \Delta_t,\ g_t(T) \rangle_F = \sum_{j=1}^{C} \left\langle \Delta_{t,j}, \nabla_{W_j} L_D(W_t; T) \right\rangle,$$

where $\Delta_{t,j}$ is the j-th row of Δ_t.

Using the expression for the gradient, we obtain

$$\langle \Delta_t,\ g_t(T) \rangle_F = \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{C} \left(p_j^{(n)} - T_{jn} \right) \left\langle \Delta_{t,j}, x^{(n)} \right\rangle.$$

For honest examples $n \in H_t$, the label is fixed. Only the columns $n \in K_t^H$ are decision variables for the attacker. Dropping all terms independent of the attacker's choices, the label-dependent objective becomes

$$-\frac{1}{N} \sum_{n \in K_t^H} \sum_{c=1}^{C} T_{cn} \left\langle \Delta_{t,c}, x^{(n)} \right\rangle.$$

Therefore, for each controlled example n, define the class score $Z_{cn} = -\left\langle \Delta_{t,c}, x^{(n)} \right\rangle$. Then the multi-class label-flipping problem is equivalent to

$$\begin{aligned} \arg\min_{T} \quad & \sum_{n \in K_t^H} \sum_{c=1}^{C} T_{cn} Z_{cn} \\ \text{s.t.} \quad & \sum_{c=1}^{C} T_{cn} = 1, \qquad \forall n \in K_t^H, \\ & T_{cn} \in \{0,1\}, \qquad \forall c, n, \\ & \sum_{n \in K_t^H} \left(1 - T_{y_n^H, n}\right) \le \lfloor b |K_t^H| \rfloor. \end{aligned} \tag{MC}$$

For each controlled example n, the best label ignoring the budget is

$$c^\star(n) \in \arg\min_{c \in \{1,\dots,C\}} Z_{cn} = \arg\max_{c \in \{1,\dots,C\}} \left\langle \Delta_{t,c}, x^{(n)} \right\rangle.$$

Define the improvement over the clean label:

$$\Gamma_n = Z_{y_n^H, n} - Z_{c^\star(n), n}.$$

A flip is beneficial exactly when $\Gamma_n > 0$. Thus, under budget

$$p = \lfloor b |K_t^H| \rfloor,$$

the optimal one-step attack changes the labels of the p controlled examples with the largest positive values of Γ_n, assigning each selected example n to $c^\star(n)$.

Theorem 3.1 (Exact one-step optimality in the multi-class setting)**.** *For multi-class softmax regression under mean aggregation, Algorithm 1 solves Problem 1 exactly at round t for any fixed reference direction Δ_t.*

Proof. At round t, the objective

$$\langle \Delta_t, g_t(T) \rangle_F$$

can be written as a constant independent of the attack plus

$$\frac{1}{N} \sum_{n \in K_t^H} \sum_{c=1}^{C} T_{cn} Z_{cn}.$$

Thus, for each controlled example n, the best label ignoring the budget is

$$c^\star(n) \in \arg\min_{c} Z_{cn}.$$

Changing the clean label y_n^H to this best label reduces the objective by

$$\Gamma_n = Z_{y_n^H, n} - Z_{c^\star(n), n}.$$

The only coupling between examples is the cardinality budget

$$\sum_{n \in K_t^H} (1 - T_{y_n^H, n}) \le \lfloor b |K_t^H| \rfloor.$$

Therefore, the optimal feasible solution is obtained by selecting the largest positive improvements Γ_n, up to the budget. This is exactly Algorithm 1. □

Algorithm 1 Per-round greedy label selection for multi-class softmax regression

Require: Controlled clean set $K_t^H = \{(x^{(n)}, y_n^H)\}_{n \in K_t^H}$, current weight matrix W_t, reference matrix Δ_t, budget b

Ensure: Poisoned labels $\{\tilde{y}_n\}_{n \in K_t^H}$

```
p ← ⌊b|K_t^H|⌋
for each n ∈ K_t^H do
    for each class c ∈ {1, ..., C} do
        Z_cn ← −⟨Δ_{t,c}, x^(n)⟩
    end for
    c*(n) ← arg min_c Z_cn
    Γ_n ← Z_{y_n^H,n} − Z_{c*(n),n}
    Initialize ỹ_n ← y_n^H
end for
Let S be the indices of the p largest positive values of Γ_n
for each n ∈ S do
    ỹ_n ← c*(n)
end for
return {ỹ_n}_{n∈K_t^H}
```

This result shows that the multi-class attack is not a separate construction: it is the direct softmax-regression instance of the same directional label-flipping problem introduced in Section 2.3.

4 Multi-Class Experimental Results

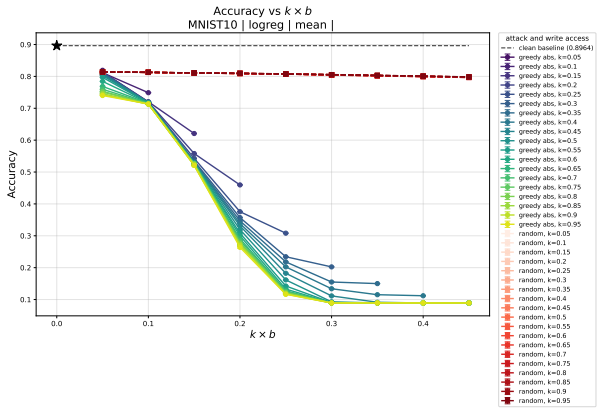
Experimental setup. We evaluate the multi-class attack on MNIST10 and CIFAR10. Since unit-norm feature normalization is commonly used when feature vectors are fed to linear classifiers, nearest-neighbor classifiers, or dot-product-based losses (Tian et al. (2020); Radford et al. (2021); Khosla et al. (2021); Chen et al. (2020)), we evaluate the attack on normalized inputs. Unless stated otherwise, the main experiments use multi-class logistic regression trained with SGD under mean aggregation, matching the setting analyzed in Section 3. We also report results for a two-layer MLP to test whether the same label-selection rule remains effective beyond the convex softmax-regression setting. Data are split iid across workers, and all results are averaged over six independent runs.

We compare our attack with random label flipping under the same k, b, dataset, model, and training protocol. In the random baseline, the attacker flips the same number of labels as the optimized attack but selects examples and replacement labels without using the gradient-obstruction score. By this, we test whether degradation comes from the optimized choice of labels or from injecting label noise.

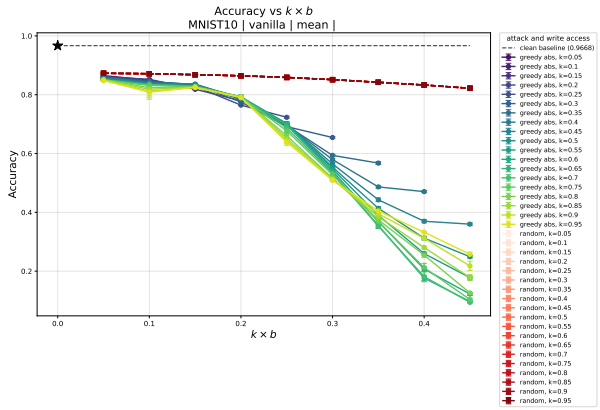
Optimized flipping causes stronger availability degradation than random noise. Figure 5 reports test accuracy as a function of the effective corruption budget ($\rho = kb$). Across the tested multi-class settings, the optimized attack generally reduces accuracy more than random flipping at the same budget. This shows that the greedy score identifies labels whose changes are harmful to the training direction, rather than merely injecting noise.

The effect is visible at small global budgets. For example, in all tested datasets, changing only a small fraction of the full batch produces a measurable drop in clean test accuracy. As the budget increases, the gap between the clean baseline and the poisoned runs widens, which is natural since larger budgets give the attacker more power.

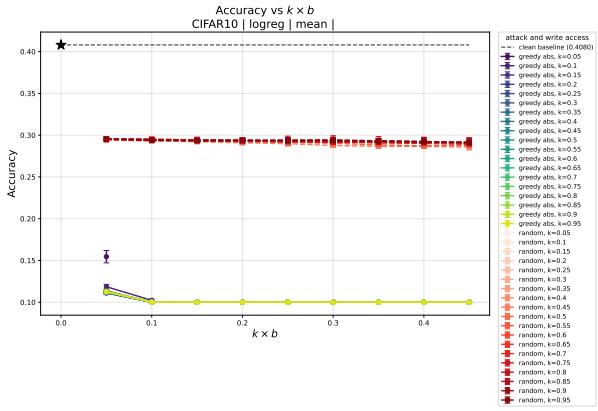
Notably, although the attack was tailored to logistic regression, it works well on a multi-layer perceptron under the setting described in section B.2.



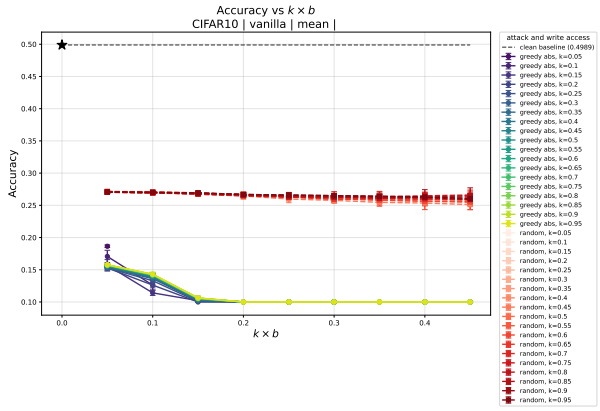
(a) MNIST10, logistic regression.



(b) MNIST10, two-layer MLP.



(c) CIFAR10, logistic regression.



(d) CIFAR10, two-layer MLP.

Figure 5: Accuracy versus effective attack budget $k \times b$ for normalized inputs. Our attack induces stronger degradation than random flipping, showing that the gradient-obstruction objective provides information beyond random label noise.

4.1 Effect of Write Access and Local Budget

The parameters k and b capture different aspects of the adversary's power. The write-access fraction k determines how many examples the attacker can inspect and potentially modify, while the local budget b determines how many of those controlled examples may actually be flipped. Although the product kb gives the global corruption rate, the attack is not determined only by this product.

To visualize this effect more directly, Figure 6 reports a heatmap of the accuracy as a function of k and b on a binary MNIST (classes 0 and 1) dataset. Although this setting is simpler than the multi-class experiments above, it isolates the roles of k and b.

Empirically, Figures 5 and 6 show that, for a fixed global corruption rate $\rho = kb$, increasing the write-access fraction k is often more damaging than increasing the local flipping budget b. In particular, when k is small, increasing b has limited effect

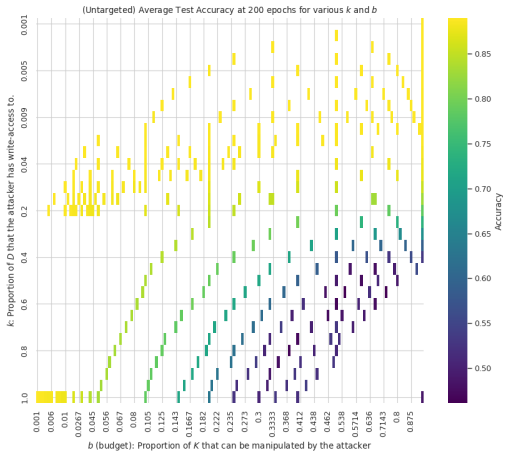
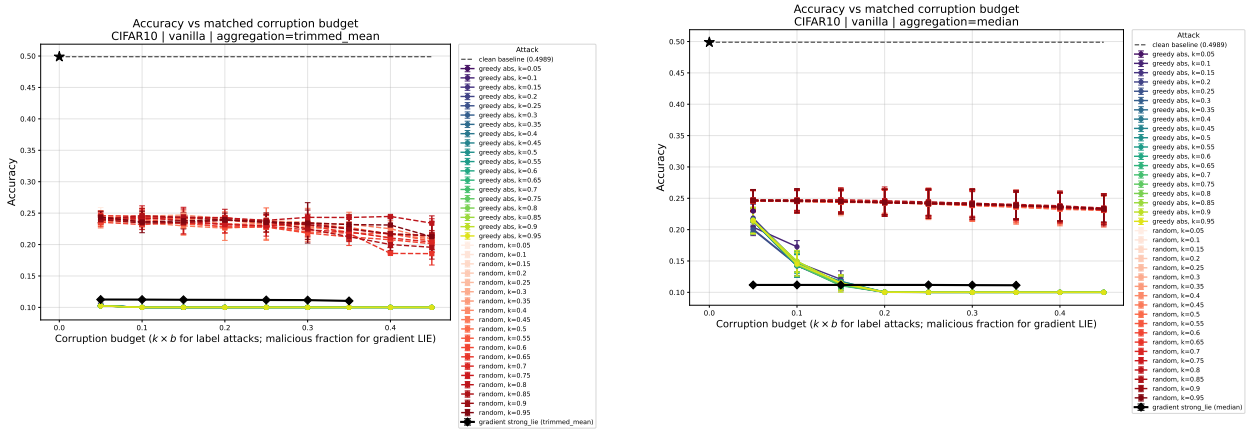


Figure 6: Heatmap of test accuracy as a function of k and b in a binary classification setting between 2 MNIST classes.

on test accuracy, whereas larger values of k consistently make the attack more effective. This suggests that, for a fixed total flipping proportion ρ, broad write access is more valuable to the attacker than the ability to flip many labels within a narrow controlled subset.

This behavior can be explained by the structure of the per-round optimization problem. A larger controlled set K_t^H gives the attacker a richer pool of candidate examples and therefore increases the probability of finding labels with large anti-alignment scores. By contrast, increasing b while keeping k fixed only allows the attacker to flip more labels within the same candidate pool, which may already contain only a limited number of highly damaging examples. This can also be understood through the attainable gradient set. Since the gradient is a weighted combination of feature vectors and the label-dependent weights are discrete, a larger write-access set expands the finite set of poisoned gradient directions available to the attacker.

4.2 Transfer to Robust Aggregation



(a) CIFAR10, two-layer MLP, trimmed mean aggregator.

(b) CIFAR10, two-layer MLP, coordinate-wise median aggregator.

Figure 7: Test accuracy under trimmed mean (left), and coordinate-wise median (right) on normalized CIFAR10 with a two-layer MLP. Although the attack is derived for mean aggregation, optimized label flipping degrades accuracy under these robust aggregators, and shows performance similar to the Little-is-Enough attack.

Our theoretical derivation and one-step optimality result are specific to mean aggregation. Nevertheless, we also test whether the same label-selection rule remains effective when the server replaces the mean with a coordinate-wise median or trimmed-mean aggregators.

Surprisingly, Figure 7 shows that the proposed label-flipping strategy remains highly damaging under both robust aggregation rules, even though it was only designed to attack the non-robust mean aggregation. This can be explained by the restricted nature of the attack. Unlike arbitrary gradient attacks, the adversary does not submit an unconstrained update vector that can be easily identified as an outlier. Instead, each malicious update is induced by valid training examples whose features are unchanged and whose labels remain in the feasible label set. As a result, the poisoned gradients can remain close enough to the distribution of honest gradients that the tested robust aggregators do not fully remove their effect.

In contrast, random label flipping is consistently weaker, showing that the degradation is not merely caused by adding label noise. We also include the Little-is-Enough gradient attack for comparison. Under this interpretation, the results indicate that a label-only adversary can, in these settings, approach the degradation caused by a substantially more powerful gradient-manipulation adversary.

5 Conclusion

We studied availability attacks under a highly constrained label-flipping threat model. By introducing an intuitive, budget-aware objective, we reveal a vulnerability often studied under stronger adversarial capabilities. Unlike gradient attacks or more general data-poisoning attacks, the adversary considered here cannot modify features, inject samples, delete data, or directly overwrite gradients; it may only flip a limited number of labels among examples under its write access. Our experiments show that optimized label flipping can substantially degrade test accuracy and consistently outperforms random label flipping under matched budgets. These findings establish a foundation for stronger defenses and, more broadly, a deeper understanding of security in federated and distributed learning. Finally, although our theoretical derivation is specific to mean aggregation, the attack transfers empirically to neural networks and robust aggregators such as coordinate-wise median and trimmed mean. Interesting follow-up directions include (i) searching for globally optimal flipping attacks, and (ii) devising practical defenses tailored to the proposed threat model.

Broader Impact Statement

In this paper, we improve the community's understanding of the capabilities of label flipping in damaging the training of a machine learning model. While doing so requires the study and the development of train-phase adversarial attacks, we believe that these attacks serve a mostly pedagogical role and help alert on the potential damage done by an omniscient attacker, previously thought to be able to do harm only when allowed to send gradients in a distributed/federated learning setup.

References

Nivasini Ananthakrishnan, Stephen Bates, Michael Jordan, and Nika Haghtalab. Delegating data collection in decentralized machine learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 478–486. PMLR, 2024.

Pranjal Awasthi, Maria Florina Balcan, and Philip M. Long. The power of localization for efficiently learning linear separators with noise. 63(6), 2017. ISSN 0004-5411. doi: 10.1145/3006384. URL `https://doi.org/10.1145/3006384`.

Melis Ilayda Bal, Volkan Cevher, and Michael Muehlebach. Adversarial training for defense against label poisoning attacks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Representation Learning*, volume 2025, pp. 16235–16269, 2025. URL `https://proceedings.iclr.cc/paper_files/paper/2025/file/298c3e32d7d402189444be2ff5d19979-Paper-Conference.pdf`.

Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 2019.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30, 2017.

Wassim Bouaziz, El-Mahdi El-Mhamdi, and Nicolas Usunier. Inverting gradient attacks makes powerful data poisoning. *arxiv:2410.21453*, 2024.

Aymeric Capitaine, Etienne Boursier, Antoine Scheid, Eric Moulines, Michael Jordan, El-Mahdi El-Mhamdi, and Alain Durmus. Unravelling in collaborative learning. *Advances in Neural Information Processing Systems*, 37:97231–97260, 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/chen20j.html`.

Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):1–25, 2017.

El-Mahdi El-Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. *International Conference on Machine Learning*, 2018.

Sadegh Farhadkhani, Lê-Nguyên Hoang, and Oscar Villemaud. An equivalence between data poisoning and byzantine gradient attacks. In *International Conference on Machine Learning*, 2022.

Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. *arXiv preprint arXiv:2009.02276*, 2020.

Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I.P. Rubinstein, and J. D. Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, AISec '11, pp. 43–58, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450310031. doi: 10.1145/2046684.2046692. URL https://doi.org/10.1145/2046684.2046692.

W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. Metapoison: Practical general-purpose clean-label data poisoning. *Advances in Neural Information Processing Systems*, 2020.

Rishi D. Jha, Jonathan Hayase, and Sewoong Oh. Label poisoning is all you need. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. In *International Conference on Learning Representations*, 2020. URL https://api.semanticscholar.org/CorpusID:238856649.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2021. URL https://arxiv.org/abs/2004.11362.

Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *Machine Learning*, pp. 1–47, 2022.

Léo Lavaur, Yann Busnel, and Fabien Autrel. Systematic analysis of label-flipping attacks against federated learning in collaborative intrusion detection systems. In *Proceedings of the 19th International Conference on Availability, Reliability and Security*, ARES '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400717185. doi: 10.1145/3664476.3670434. URL https://doi.org/10.1145/3664476.3670434.

Qingru Li, Xinru Wang, Fangwei Wang, and Changguang Wang. A label flipping attack on machine learning model and its defense mechanism. In *Algorithms and Architectures for Parallel Processing: 22nd International Conference, ICA3PP 2022, Copenhagen, Denmark, October 10–12, 2022, Proceedings*, pp. 490–506, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-22676-2. doi: 10.1007/978-3-031-22677-9_26. URL https://doi.org/10.1007/978-3-031-22677-9_26.

Yiyong Liu, Michael Backes, and Xiao Zhang. Transferable availability poisoning attacks. *arXiv:2310.05141*, 2023a.

Yuchen Liu, Chen Chen, Lingjuan Lyu, Fangzhao Wu, Sai Wu, and Gang Chen. Byzantine-robust learning on heterogeneous data via gradient splitting. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023b.

Yiwei Lu, Gautam Kamath, and Yaoliang Yu. Indiscriminate data poisoning attacks on neural networks. *ArXiv*, abs/2204.09092, 2022. URL https://api.semanticscholar.org/CorpusID:248266720.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Xiaojin (Jerry) Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017. URL http://proceedings.mlr.press/v54/mcmahan17a.html.

Rui Ning, Jiang Li, Chunsheng Xin, and Hongyi Wu. Invisible poison: A blackbox clean label backdoor attack to deep neural networks. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pp. 1–10. IEEE, 2021.

Andrea Paudice, Luis Muñoz-González, and Emil C. Lupu. Label sanitization against label flipping poisoning attacks, 2018. URL https://arxiv.org/abs/1803.00992.

Jie Peng, Weiyu Li, Stefan Vlaski, and Qing Ling. Mean aggregator is more robust than robust aggregators under label poisoning attacks on distributed heterogeneous data. *Journal of Machine Learning Research*, 26(27):1–51, 2025. URL http://jmlr.org/papers/v26/24-1307.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks, 2018. URL https://arxiv.org/abs/1804.00792.

Virat Shejwalkar, Amir Houmansadr, Peter Kairouz, and Daniel Ramage. Back to the drawing board: A critical evaluation of poisoning attacks on federated learning. *CoRR*, abs/2108.10241, 2021. URL https://arxiv.org/abs/2108.10241.

Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. *Advances in neural information processing systems*, 30, 2017.

Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H. Brendan McMahan. Can you really backdoor federated learning?, 2019. URL https://arxiv.org/abs/1911.07963.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *ArXiv*, abs/2003.11539, 2020. URL https://api.semanticscholar.org/CorpusID:214641252.

Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized byzantine-tolerant SGD. *CoRR*, abs/1802.10116, 2018. URL http://arxiv.org/abs/1802.10116.

Shanqing Yu, Jie Shen, Shaocong Xu, Jinhuan Wang, Zeyu Wang, and Qi Xuan. Label-flipping attacks in gnn-based federated learning. *IEEE Transactions on Network Science and Engineering*, 12(2):1357–1368, 2025. doi: 10.1109/TNSE.2025.3528831.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017. URL https://arxiv.org/abs/1611.03530.

Bingyin Zhao and Yingjie Lao. CLPA: Clean-Label Poisoning Availability Attacks Using Generative Adversarial Nets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 9162–9170, 2022. doi: 10.1609/aaai.v36i8.20902. URL https://ojs.aaai.org/index.php/AAAI/article/view/20902.

Appendix

This appendix is organized as follows, section A details the datasets, data splits, target models, and training protocol, and provides the end-to-end pseudocode used in our experiments. We then justify in Section B the first focus on mean aggregation and that our labels-only attack operates within valid gradient magnitudes, limiting the effectiveness of norm-based defenses. Subsequent sections discuss data distribution assumptions (our attack is distribution-agnostic and heterogeneity can further hinder robust aggregators) and clarify the threat model: we analyze an omniscient adversary for comparability and outline how a surrogate-based attacker can operate with partial knowledge.

A Dataset and Experimental Setup

Datasets. We perform all experiments on MNIST10 and CIFAR10. Each image was scaled by its unit norm.

Implementation Details. We train the models for 200 epochs using mini-batch SGD with a batch size of 64 and a learning rate of 0.001, using cross-entropy loss. At each epoch, the omniscient attacker observes the current model parameters and gradients, then flips the labels of a randomly assigned subset K from the clean data pool accordingly. All results are averaged over six independent runs with different random seeds. The global training algorithm can be found below (Algorithm 2).

Algorithm 2 Full Training with Per-Round Label-Flipping Attack

Require: Clean training set D_{train}, model h_θ, number of training rounds T, write-access fraction k, local flipping budget b, and functions:

- `sampleBatch`: samples a clean batch $D_t^H \subset D_{\text{train}}$.
- `getWriteAccess`: selects the attacker's controlled subset $K_t^H \subset D_t^H$ with $|K_t^H| = \lfloor k|D_t^H| \rfloor$.
- `selectFlip`: applies Algorithm 1 to choose at most $\lfloor b|K_t^H| \rfloor$ labels to flip.
- `trainStep`: performs one optimization step on the submitted batch.

Ensure: Poison-trained model h_{θ_T}

Initialize model parameters θ_0
for $t \leftarrow 0$ **to** $T-1$ **do**
  $D_t^H \leftarrow \texttt{sampleBatch}(D_{\text{train}})$
  $K_t^H \leftarrow \texttt{getWriteAccess}(D_t^H, k)$
  $H_t \leftarrow D_t^H \setminus K_t^H$
  $K_t(\tilde{y}) \leftarrow \texttt{selectFlip}(K_t^H, \theta_t, b)$
  $D_t(\tilde{y}) \leftarrow H_t \cup K_t(\tilde{y})$
  $\theta_{t+1} \leftarrow \texttt{trainStep}(\theta_t, D_t(\tilde{y}))$
end for
return h_{θ_T}

B Discussion

B.1 How practical is the attack?

In addition to our formal setup, allowing comparison between gradient attacks and label flipping, we note several realistic settings where label-only corruption can occur in production systems while possibly having access to knowledge about the model. This can be the case in, *e.g.*,

- compromised labeling tools: attackers who can modify annotation UIs, metadata, or labeling scripts can change labels stored in databases while leaving raw sensor data untouched,

- insider at a data provider: an employee with access to label fields (but not feature pipelines) can flip labels locally,
- crowdsourced annotation manipulation: attackers controlling a subset of crowd workers or poisoning weakly-supervised labeling heuristics can flip a small fraction of labels across contributors,
- collaborative learning situations where participants are provided various forms of "a global model" in exchange for their participation with data as in, *e.g.* Capitaine et al. (2024); Ananthakrishnan et al. (2024).

B.2 Why Logistic Regression (before deeper models)

There are two reasons why we adopt logistic regression as a first-class test bed before moving to deeper models as we did: *(i) Analytic transparency.* The loss is convex and its gradient admits a simple closed form, allowing us to derive our greedy attacks. *(ii) Gateway to richer models.* Any classifier whose final layer is linear followed by a sigmoid/softmax (e.g. multilayer networks, convolutional nets) reduces to logistic regression if we treat the penultimate activations as "features." Consequently, at each training step our attack should apply to those models by operating on the last-layer logits.[1] This raises the question: Would such an attack still be greedy for any classifier? One path to performing a label-flipping attack on a broader class of models could be the transition from logistic regression to neural networks. We perform experiments on a 2-layer vanilla MLP using the method described above. The results in Figure 5 show that the attack is remarkably efficient even for these deeper models. Another approach is to find a globally optimal label flipping attack algorithm. We assumed that the server was aggregating the gradients using the mean, and future work could study similar attacks on settings with different aggregation methods presented in Baruch et al. (2019); El-Mhamdi et al. (2018); Blanchard et al. (2017).

B.3 Why mean aggregation still matters

Mean aggregation remains the de facto baseline in both federated learning research and large-scale deployments as it is favored for its scalability, communication efficiency, and ease of implementation (McMahan et al. (2017)). As noted in Shejwalkar et al. (2021), production cross-device FL systems with thousands to billions of clients often employ plain averaging, and empirical evidence suggests that even the most basic, non-robust FL algorithms can be surprisingly resilient in realistic, low-compromise settings. In this context, showing that a constrained, label-flipping adversary can meaningfully degrade model performance under mean aggregation is already a strong and practically relevant finding.

Our focus on mean aggregation is therefore intentional: it establishes a baseline benchmark for the impact of constrained label-flipping attacks. If such an attack fails under mean aggregation, it is unlikely to pose a serious threat to more sophisticated robust aggregators; if it succeeds, it identifies a vulnerability that robust methods should be expected to address. Moreover, our attack is derived specifically for the mean operator, and different aggregation rules may require different attack formulations. Nevertheless, our experiments show that the same label-selection rule also transfers empirically to the tested robust aggregators, namely coordinate-wise median and trimmed mean.

Finally, constrained label flipping interacts differently with defenses than arbitrary gradient manipulation. Since the attack changes only labels while leaving feature inputs untouched, the resulting updates are still induced by valid training examples and feasible labels. As a result, norm-based defenses such as clipping, which are often used in combination with mean aggregation Sun et al. (2019), may fail to distinguish poisoned updates when their magnitudes remain within honest-looking ranges. This further motivates studying mean aggregation as a clean and practically relevant baseline before moving to more complex aggregation and defense mechanisms.

[1] Formally, let $f_\theta(X) = \sigma(W h_\phi(X))$ with inner network h_ϕ fixed during one optimization step. Setting $\tilde{X} = h_\phi(X)$ recasts the update as a logistic regression in $(\tilde{X}, y)$ with parameter W.

B.4 Scaling affects both attack interpretation and detectability.

Normalization affects both the strength and the detectability of the attack. Theoretically, the flip score contains inner products of the form

$$\langle \Delta_t, x_i \rangle = \|\Delta_t\| \, \|x_i\| \cos(\phi_i).$$

When features are normalized, the score is driven primarily by directional alignment. When features are unnormalized, high-norm points can dominate the ranking, making the optimized attack less purely directional. This can reduce the gap between optimized and random flipping in some settings.

Normalization also matters for detectability. In normalized settings, we empirically observe that the attacker's per-client gradient norm is nearly indistinguishable from honest clients (see Table 2 in Appendix C). Thus, defenses based only on gradient-norm clipping may fail to distinguish malicious updates from honest ones.

B.5 What if the attacker lacks omniscience

In the case where the attacker lacks full knowledge, for example consider a scenario where the attacker knows the architecture of the server's model but not its current parameters. Then the attacker can train a local surrogate model using only the data available to them (i.e., the subset they control). This local model may not perfectly match the server's model, but it can still approximate the decision boundary or the gradient dynamics well enough to inform a useful attack as done in, *e.g.* El-Mhamdi et al. (2018); Baruch et al. (2019) or anticipate the dynamics in one-time attack as in Jha et al. (2023).

Specifically, the attacker can simulate the label-flipping attack on their local model concurrently with the server's training. By observing how different label flips affect the gradient direction or training loss on their surrogate, the attacker can estimate which flips are most likely to have the desired effect when sent to the server. This strategy may reduce the attack's effectiveness but remain viable.

B.6 On the data distribution

Our experiments used a uniform i.i.d. data split across workers. All clients sampled from the same distribution and contributed equal-sized batches. However, our attack is distribution-agnostic: it does not assume i.i.d. data and optimizes flips online per iteration. In fact, heterogeneity generally makes many robust aggregation rules less reliable; non-IID gradients enlarge natural variability, which both weakens anomaly tests and can cause robust rules (e.g., trimming) to underperform FedAvg in certain regimes (Liu et al. (2023b)).

As attacks are easier in heterogeneous settings, and harder in homogeneous ones (Peng et al. (2025); Karimireddy et al. (2020)), our work highlights the power of label flipping even on i.i.d. (and thus less vulnerable) situations.

C Detectability and data normalization

Table 2: Normalized-data mean gradient-norm ratio (attacker mean gradient norm divided by honest mean gradient norm). Ratios extremely close to 1 indicate that, under standard dataset normalization, the attacker's per-round gradient magnitudes are statistically indistinguishable from those of honest clients.

dataset	model	attack/honest norm ratio
cifar10	logreg	1.004523
cifar10	vanilla	1.000400
mnist10	logreg	1.028216
mnist10	vanilla	1.018608

In this section, we evaluate how detectable our label-flipping attack is under common feature-preprocessing pipelines, focusing on the gradient-norm statistics that many FL defenses monitor. Practical FL systems

almost always apply some form of input normalization or scaling before training. Under these normalized settings, we empirically find that the attacker's per-client gradient norms become nearly indistinguishable from those of honest clients (Table 2). As a result, defenses that rely solely on gradient-norm thresholds or magnitude-based outlier filtering are ineffective.