POSTERIOR PROBABILITY-BASED LABEL RECOVERY ATTACK IN FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent works have proposed analytical attacks that can restore batch labels from gradients of a classification model in *Federated Learning* (FL). However, these studies rely on strict assumptions and do not show the scalability of other classification loss functions. In this paper, we propose a generalized label recovery attack by estimating the posterior probabilities. Beginning with the *focal loss* function, we derive the relationship among the gradients, labels and posterior probabilities in a concise form. We also empirically observe that positive or negative samples of a class have approximate probability distributions. This insight enables us to estimate the posterior probabilities of the target batch from some auxiliary data. Integrating the above elements, we present our label attack that can directly recover the class-wise batch labels in realistic FL settings. Evaluation results show that on an untrained model, our attack can achieve over 95% *Instance-level label Accuracy* (InsAcc) and 96% *Class-level label Accuracy* (ClsAcc) on different groups of datasets, models and activations. For a training model, our approach reaches more than 90% InsAcc on different hyper-parameters.

1 INTRODUCTION

Federated Learning (FL) has become a popular paradigm for training machine learning models in privacy-sensitive applications (McMahan et al., 2017; Yang et al., 2019). In FL, the clients compute gradients on their local devices and then send the gradients to the server for aggregation and global model update. Since the private data is preserved on the client side, FL is supposed to offer more privacy protection. However, *Gradient Inversion Attacks* (GIAs) have shown that the gradient updates can be exploited to reconstruct the training data (Zhu et al., 2019; Geiping et al., 2020; Yin et al., 2021). Moreover, some analytical attacks propose to restore class-wise labels from the gradients by analyzing the relationship between the gradients and the labels (Zhao et al., 2020; Wainakh et al., 2022; Ma et al., 2023). However, these works are typically not applicable in realistic FL scenarios, and they only focus on the cross-entropy loss function.

In this paper, we propose a generalized label recovery attack. Starting with the *focal loss* function, we first derive a critical relationship among the gradients, labels and posterior probabilities in a concise form. This conclusion can be applied to a variety of loss functions, which reveals the connection between the gradients and the labels in a classification task. Then we propose our label attack by estimating the posterior probabilities of the target batch from an auxiliary dataset. We empirically observe that the positive or negative samples of a class have approximate probability distributions. By fitting some auxiliary data into the global model, we can estimate the batch probabilities and recover the labels of any specified class from the derived equation.

Our main contributions are summarized as follows:

- For the first time, we find that class-wise labels are only related to the posterior probabilities and gradients in various loss functions, such as focal loss and binary cross-entropy loss.
- We evaluate our attack on a variety of FL settings and classification hyperparameters, and demonstrate that it outperforms the previous attacks in terms of *Class-level label Accuracy* (ClsAcc) and *Instance-level label Accuracy* (InsAcc).

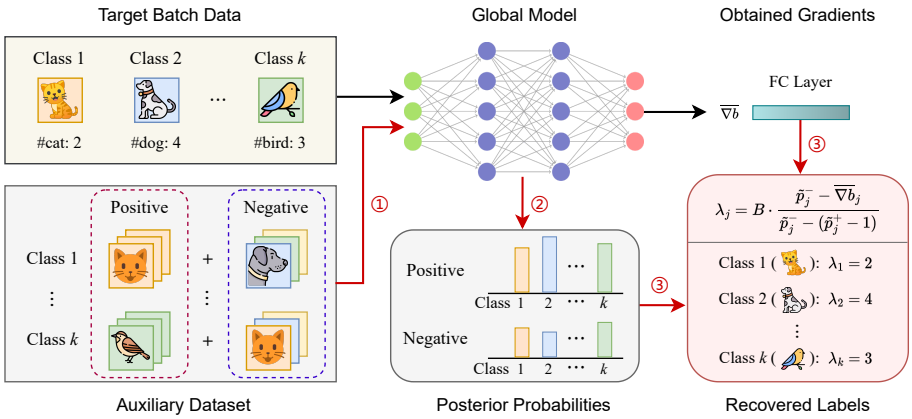


Figure 1: The workflow of our label recovery attack. ① Acquire an auxiliary dataset and divide it into positive and negative subsets for each class. ② Fit the data into the global model for estimating the posterior probabilities of the target batch. ③ Recover the batch labels by substituting the gradients and the posterior probabilities into the derived formula.

2 Problem Formulation

For a K-class classification task in FL, the FedSGD (McMahan et al., 2017) algorithm is used to train the global model $\mathcal{M}$. At a given iteration, a victim client v trains the model with its local batch data $\boldsymbol{x}_v$ and labels $\boldsymbol{y}_v$, where $\{x_v, y_v\} \sim \mathcal{D}_v$. Here, $\mathcal{D}_v$ denotes the data distribution of client v. Then, the client calculates the batch-averaged gradient $\mathcal{G}$ of the model parameters and sends it to the server. As an *honest-but-curious* server or attacker, we aim to recover the batch labels $\boldsymbol{y}$ from the shared gradient $\mathcal{G}$. The knowledge we have includes the global model $\mathcal{M}$, the shared gradient $\mathcal{G}$, and an auxiliary dataset $\boldsymbol{x}_a$ with the same data distribution as $\mathcal{D}_v$.

In our label attack, we mainly utilize the gradient w.r.t. the bias b in the last fully connected layer, i.e., $\overline{\nabla \boldsymbol{b}}$, to recover the target batch labels $\boldsymbol{y}$. From the auxiliary dataset $\boldsymbol{x}_a$, we randomly sample a portion of instances $\hat{\boldsymbol{x}}_a$, ensuring that the number of instances in each class equals π. Then we divide $\hat{\boldsymbol{x}}_a$ into positive samples $\hat{\boldsymbol{x}}_j^+$ and negative samples $\hat{\boldsymbol{x}}_j^-$ for each class j. By fitting the model $\mathcal{M}$ with these samples, we can obtain the averaged posterior probabilities $\hat{\boldsymbol{p}}^+$ and $\hat{\boldsymbol{p}}^-$ of the positive and negative samples. By substituting $\overline{\nabla \boldsymbol{b}}$, $\hat{\boldsymbol{p}}^+$ and $\hat{\boldsymbol{p}}^-$ into our derived formula in Section 3.3, we can recover the batch labels $\hat{\boldsymbol{y}}$. The workflow of our label attack is shown in Figure 1.

3 Label Recovery Attack

3.1 Generalized Expression of Gradients

Without loss of generality, we consider a multi-class classification task using *focal loss* as the loss function and *posterior* as the activation function. Focal loss is an extension of the cross-entropy loss, which is proposed to solve the problem of class imbalance (Lin et al., 2017). The definition of focal loss in multi-class scenarios can be represented as:

$$\mathcal{L}_{\text{FL}}(p_t) = -\sum_{t=1}^{K} \alpha_t (1-p_t)^{\gamma} \log p_t, \tag{1}$$

where p_t is the categorical probability of target class t, α_t is the weight of the target class, γ is the focusing parameter that controls the degree of class imbalance, and K is the number of classes.

For the tth class, $p_t = p_i$ if $i = t$, and $p_t = 1 - p_i$ if $i \neq t$. Here, p_i is the posterior probability of the ith class[1], and y_i is the corresponding label. If we set $\gamma = 0$ and $\alpha_t = 1$, the focal loss is

[1] p_i is an instance-wise probability, while p_t is a class-wise probability incorporating the target label.

equivalent to the cross-entropy loss $\mathcal{L}_{\text{CE}}(p_i) = -\sum_{t=i}^{K} y_i \log p_i$. Under the same setting, if $K = 2$, $y \in \{0, 1\}$ and the activation is Sigmoid[2], the focal loss is also equivalent to the binary cross-entropy loss $\mathcal{L}_{\text{BCE}}(p) = -y \log p - (1 - y) \log (1 - p)$. Thus, analyzing from focal loss allows us to derive a more general conclusion that can be applied to the other classification variants.

Theorem 1 (Gradient of Focal Loss)**.** *For a K-class classification task using the **focal loss** function and **posterior** activation, we can derive that the gradient of logit z_j as follows:*

$$\nabla_{z_j} \mathcal{L}_{FL} = \sum_{t=1}^{K} \Phi(\alpha_t, p_t, \gamma) \cdot (p_j - \delta_{tj}), \tag{2}$$

where $\Phi(\alpha_t, p_t, \gamma) = \alpha_t (1 - p_t)^{\gamma} \left(1 - \gamma \frac{p_t \log p_t}{1 - p_t}\right)$ and $\forall t \in K$, we have $\Phi(\alpha_t, p_t, \gamma) \geq 0$. Besides, δ_{tj} is the Kronecker delta, which equals 1 if $t = j$ and 0 otherwise.

Proof. See Appendix C.1. □

From Theorem 1, we find that $\nabla_{z_j} \mathcal{L}_{\text{FL}}$ is a summation of K terms, where each term is a product of $\Phi(\alpha_t, p_t, \gamma)$ and $(p_j - \delta_{tj})$. The item $\Phi(\alpha_t, p_t, \gamma)$ can be regarded as the weight of the tth class, and $(p_j - \delta_{tj})$ indicates the distance between the posterior probability of the jth class and the target categorical expectation at the tth class. In particular, an interesting observation can be made from the latter item is that $(p_j - \delta_{tj})$ is only negative when $t = j$, and positive otherwise, which supports the following conclusions of different loss functions.

From this generalized relationship, we can also derive the gradient of logits $\boldsymbol{z}$ (i.e., $\nabla \boldsymbol{z}$) in other classification variants by setting different values for α_t or γ, and choosing different label embeddings or activation functions. We summarize the commonly used loss functions, argument settings and the corresponding gradients in Table 1. We can find that the gradient $\nabla \boldsymbol{z}$ is only related to the posterior probabilities $\boldsymbol{p}$ and the target labels $\boldsymbol{y}$. This finding reveals the connection between the gradient $\nabla \boldsymbol{z}$ and the target label $\boldsymbol{y}$, which is the key to label recovery attacks.

Table 1: Relationships between different loss function and its gradient $\nabla \boldsymbol{z}$.

Loss function	$\boldsymbol{\gamma}$	$\boldsymbol{\alpha}$	**Label**	**Activation**[3]	**Gradient $\nabla \boldsymbol{z}$**
Focal loss	-	-	one-hot	posterior (τ)	$\frac{1}{\tau} \Phi(\alpha_c, p_c, \gamma)(\boldsymbol{p} - \boldsymbol{y})$
Cross-entropy loss	0	1	-	posterior (τ)	$\frac{1}{\tau}(\boldsymbol{p} - \boldsymbol{y})$
Binary cross-entropy loss	0	1	binary	Sigmoid	$p - y$

3.2 Key Observation

In a multi-class classification task using a neural network, the model first outputs the logits $\boldsymbol{z}$ according to forward propagation, and then normalizes the logits into probabilities $\boldsymbol{p}$ through the posterior function. By analyzing these posterior probabilities, we empirically observe that different positive (negative) samples of a class have approximate probability distributions.

We carry out the experiments on MNIST (LeCun et al., 1998) and CIFAR-10 Krizhevsky & Hinton (2009) datasets, which are trained on the LeNet-5 (LeCun et al., 1998) and ResNet-18 (He et al., 2016) models, respectively. To eliminate the influence of the activation function, we choose Tanh for LeNet-5 and ReLU (Nair & Hinton, 2010) for ResNet-18. For each class, we treat the data belonging to this category as positive samples, and the others as negative samples. Then we aggregate the positive and negative samples from each class during the training procedure to show the correlations and variations in their corresponding probabilities. For ease of demonstration, we only display the results of the first 500 training iterations.

[2] A special case of posterior with one neuron: $\frac{e^{z_1}}{e^{z_1} + e^{z_2}} = \frac{1}{1 + e^{-(z_1 - z_2)}} = \frac{1}{1 + e^{-z}}$, where $z = z_1 - z_2$.

[3] τ denotes the temperature parameter in posterior for softness control.

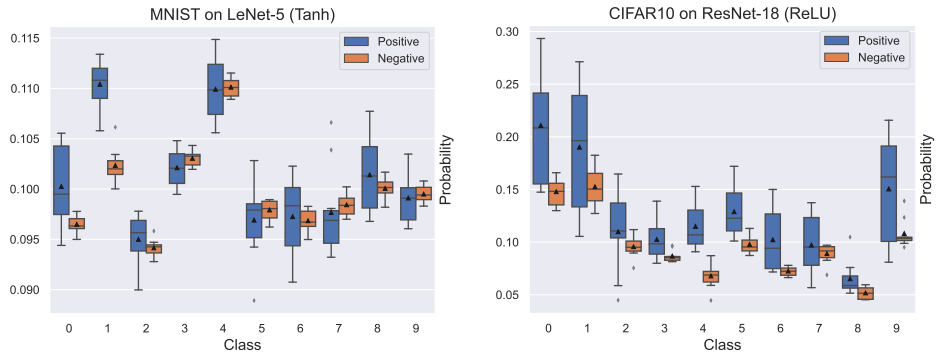


Figure 2: posterior probability distribution of different classes.

We exhibit the box plots of the probability distributions of all the classes in Figure 2, whose training iteration is 100 for MNIST and 200 for CIFAR-10. It is shown that the positive or negative probabilities of each class are gathered around a certain value, although the values are slightly different. For some easily distinguishable classes, the positive and negative probabilities are already separated, such as class 1 in MNIST and class 4 in CIFAR-10.

We can interpret the above observations from the model's representation capability. For an untrained model, it cannot discriminate which class the instance belongs to, other than random guesses. Thus, the posterior probabilities of the positive and negative samples are almost the same, equal to $\frac{1}{K}$. As training proceeds, the model gradually learns the data distribution and can distinguish the positive (negative) samples per class. Although the degree of learning varies moderately for different classes, a robust model can give similar confidence scores (i.e., posterior probabilities) for the same class. This explains why the positive (negative) samples per class have similar probability distributions.

3.3 Analytical Label Recovery

Based on the observation in Section 3.2, we can estimate the posterior probabilities of the target batch from an auxiliary dataset, whose data distribution is the same as the training data. Hence, we denote the estimated positive and negative probabilities of the jth class as $\hat{p}_j^+$ and $\hat{p}_j^-$, respectively. Combined with the conclusions in Section 3.1, we can derive the following theorem for restoring the batch labels λ_j for each class j.

Theorem 2 (Label Recovery Formula)**.** *Having **an auxiliary dataset** with the same distribution of training data, we can recover the class-wise labels λ_j in the target batch according to the averaged gradient $\overline{\nabla b}_j$ and the estimated posterior probabilities $\hat{p}_j^+$ and $\hat{p}_j^-$ as follows:*

$$\lambda_j = B \cdot \frac{(\hat{p}_j^- - y_j^-) - \overline{\nabla b}_j / \hat{\varphi}_j}{(\hat{p}_j^- - y_j^-) - (\hat{p}_j^+ - y_j^+)}, \tag{3}$$

where y_j^+ and y_j^- are the pre-set label embeddings of class j, $\hat{\varphi}_j = \frac{1}{\tau}\Phi(\alpha_j, \hat{p}_j^+, \gamma)$ is an coefficient related to the jth class, and B is the batch size.

Proof. See Appendix C.2. □

Specifically, the label embeddings are pre-defined by the FL protocol, which could be one-hot labels or smoothed labels. For one-hot labels, we have $y_j^+ = 1$ and $y_j^- = 0$. For smoothed labels, we have $y_j^+ = 1 - \epsilon$ and $y_j^- = \frac{\epsilon}{K-1}$, where ϵ is the smoothing factor. Since $\Phi(\alpha_j, p_j, \gamma)$ is determined by p_j, we can use $\hat{p}_j^+$ for replacement and obtain $\hat{\varphi}_j$. By substituting the gradient $\overline{\nabla b}_j$, estimated coefficient $\hat{\varphi}_j$, positive probabilities $\hat{p}_j^+$, $\hat{p}_j^-$ and label embeddings y_j^+, y_j^- into the above formula, we can directly recover the number of labels λ_j for each class j.

Table 2: Comparison of our attack with the baselines on diverse scenarios.

Dataset	Model	Activation	LLG		iLRG		Ours	
			ClsAcc	InsAcc	ClsAcc	InsAcc	ClsAcc	InsAcc
MNIST	LeNet-5	Sigmoid	0.954	0.973	0.946	0.880	**1.000**	**1.000**
		Tanh	0.506	0.163	**1.000**	**1.000**	**1.000**	**1.000**
CIFAR10	VGG-16	ReLU	0.995	0.997	**1.000**	**1.000**	**1.000**	**1.000**
		ELU	0.965	0.979	**1.000**	**1.000**	**1.000**	**1.000**
CIFAR100	ResNet-50	ReLU	0.937	0.952	**1.000**	**1.000**	**1.000**	**1.000**
		SELU	0.028	0.005	0.922	0.832	**0.968**	**0.951**

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTINGS

Dataset, Model and Activation. We evaluate our attack on three datasets, including MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky & Hinton, 2009) and CIFAR-100 (Krizhevsky & Hinton, 2009). These datasets are widely used in FL research and cover a variety of classification tasks, such as handwritten digit recognition, object recognition and image classification. We choose LeNet-5 (LeCun et al., 1998), VGG-16 (Simonyan & Zisserman, 2014) and ResNet-50 (He et al., 2016) as the models for the above datasets, respectively. In addition, we select a bunch of activation functions, including Sigmoid, Tanh, ReLU (Nair & Hinton, 2010), ELU (Clevert et al., 2016) and SELU (Klambauer et al., 2017), to verify the universality of our attack.

Evaluation Metrics. To quantitatively evaluate the performance of our label recovery attack, we use the following two metrics: (1) *Class-level label Accuracy (ClsAcc)*: the accuracy measures the proportion of correctly recovered classes; (2) *Instance-level label Accuracy (InsAcc)*: the accuracy measures the proportion of correctly recovered labels in the target batch. In particular, both of these two metrics are realized through *Jaccard similarity*.

Baselines. Since iDLG (Zhao et al., 2020) only applies to single batch training and non-negative activation functions, we exclude it from the comparison. We mainly compare our attack with LLG (Wainakh et al., 2022) and iLRG (Ma et al., 2023), which do not limit the batch size or activation function. For LLG, we generate the dummy data with the same number as the target batch size and average the results of 10 runs. Since LLG and iLRG are all designed for the untrained models, we mainly compare our attack with them in the untrained setting to be fair.

4.2 COMPARISON WITH BASELINES

To exhibit the versatility of our attack, we compare it with the baselines in 3 different groups of settings. We set the batch size to 32 for MNIST and CIFAR10, and 256 for CIFAR100. Without loss of generality, we assume that the training data of each class is uniformly distributed, and the auxiliary dataset is randomly sampled from the validation dataset with 100 samples per class. Furthermore, since the baselines are designed for untrained models, we also use initialized models for comparison to be fair. We run each experiment 20 times and report the average results in Table 2.

From the results, we can see that our attack performs better than the baselines and even achieves 100% ClsAcc and 100% InsAcc in most of the scenarios. In addition, the evaluation results also illustrate that compared with the dataset and model structure, the activation function has a greater impact on the performance of all label recovery attacks. This could be explained by the fact that some activation functions, like SELU, produce high variance, which causes the probability distribution of positive and negative samples from the same class to diverge significantly. Therefore, the attack performance of SELU is worse than that of ReLU and ELU. Nevertheless, our attack still shows good results, which demonstrates its effectiveness and universality.

4.3 Comparison of Various FL Settings

From Table 2, it is shown that the attack baselines have the best performance for CIFAR10 on VGG-16 with ReLU activation. So we chose this scenario to analyze the effects of batch size and class imbalance. The batch size varies from 64 to 1024, which is closer to a realistic FL scenario. Class imbalance is a prevalent issue in FL, typically brought on by the unequal distribution of data across various clients. We compare the class proportions from 10% to 90% to simulate the class imbalance. Before launching the attacks, we train the model for 1 epoch with a learning rate of 0.001.

It is shown in Figure 3 that our label recovery is robust to various batch sizes and class imbalance ratios, which maintains over 90% accuracy in all of these settings. As the batch size increases, the InsAcc of our attack gradually improves, which is mainly because the larger the batch size, the more robust the estimation of the averaged posterior probabilities. In addition, since we have the prior distribution of the training data, we can constrain and regularize the estimated labels to improve the success rate of label recovery.

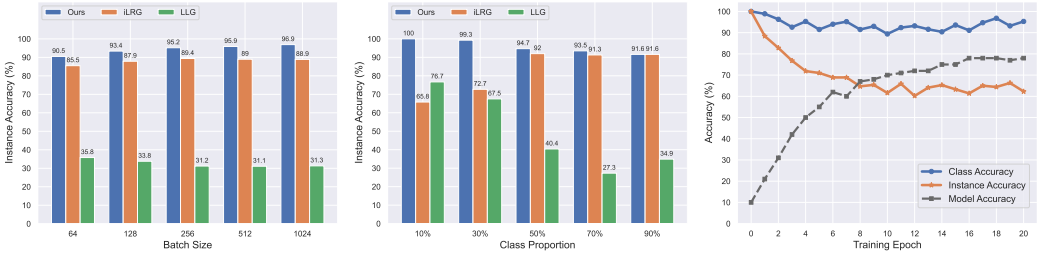


Figure 3: Instance accuracy with different batch sizes and class imbalances.

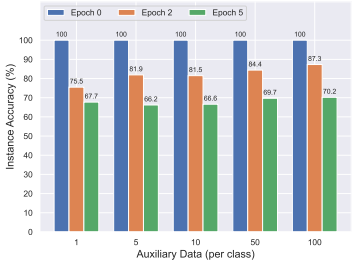
4.4 Ablation Studies

We conduct ablation studies to analyze the effectiveness of our attack under different classification variants and scales of auxiliary data. Table 3 shows the InsAcc of our attack on an untrained model with the focal loss and cross-entropy loss under different hyper-parameters. The results indicate that our attack is robust to these variants, which can achieve 100% InsAcc in all of these settings. Moreover, we also show the attack performance with different scales of auxiliary data in Figure 4. For an untrained model, the attack performance is not sensitive to the scale of auxiliary data per class, which can achieve 100% InsAcc in all of these settings.

Table 3: InsAcc with classification variants.

Loss function	**Temperature τ**		**Label smoothing ε**	
	0.8	1.2	0.1	0.25
Focal loss	1.000	1.000	1.000	1.000
Cross-entropy	1.000	1.000	1.000	1.000

Figure 4: InsAcc with scales.



5 Conclusion

In this paper, we investigate the label recovery threats of FL and reveal the connections between the gradients and the labels. We propose an analytical method to recover the batch labels from the estimated posterior probabilities and the gradients. The experiments show that our attack is robust to various FL settings and classification variants, which is validated by extensive experiments on various datasets and models. We leave the exploration of defense mechanisms as the future work.

REFERENCES

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations (ICLR)*, 2016.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Kailang Ma, Yu Sun, Jian Cui, Dawei Li, Zhenyu Guan, and Jianwei Liu. Instance-wise batch label restoration via gradients in federated learning. In *The Eleventh International Conference on Learning Representations*, 2023.

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

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Aidmar Wainakh, Fabrizio Ventola, Till Müßig, Jens Keim, Carlos Garcia Cordero, Ephraim Zimmer, Tim Grube, Kristian Kersting, and Max Mühlhäuser. User-level label leakage from gradients in federated learning. *Proceedings on Privacy Enhancing Technologies*, 2:227–244, 2022.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16337–16346, 2021.

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

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

A Related Work

Zhao et al. (2020) propose an analytical label attack named iDLG, which can directly infer the label from $\nabla \boldsymbol{W}$ of the classification layer. They derive that the gradient w.r.t the logit z_j equals to $\sigma(z_j) - 1$ if j is the target index c of the one-hot label, and $\sigma(z_j)$ if $j \neq c$, where $\sigma(\cdot)$ denotes the posterior function. When the model uses a non-negative activation, such as ReLU or Sigmoid, $\nabla \boldsymbol{W}_c$ consists of negative values, while the other rows are positive. Thus, the attacker can extract the label by simply comparing the signs of the gradient $\nabla \boldsymbol{W}$. However, iDLG only applies to single-batch labels, and the activation of the model must be non-negative. Wainakh et al. (2022) exploit the direction and magnitude of $\nabla \boldsymbol{W}$ to determine how many instances of each class are in the target batch. They formulate the problem as $\sum_{i=1}^{M} \nabla \boldsymbol{W}_j = \lambda_j m + s_j$, where λ_j is the number of batch labels of the jth class, m is the impact factor related to the input features, and s_j is a class-specific offset caused by misclassification. Using known data and labels, impact m and offset s_j can be estimated from multiple sets of gradients, and then λ_j can be calculated. Ma et al. (2023) transform the label recovery problem into solving a system of linear equations. For each class j, they regard $\sigma(z_j) - 1$ and $\sigma(z_j)$ as the coefficients of the target label and the other labels, respectively. By constructing these coefficients into a matrix $\boldsymbol{A}$, they can solve the label vector $\boldsymbol{y}$ from the equation $\boldsymbol{A}\boldsymbol{y} = \nabla \boldsymbol{b}$, where $\nabla \boldsymbol{b}$ is the gradient w.r.t the bias term of the last layer. However, none of these works explain the essence of label leakage from gradients, or address the issue of whether the label attacks can apply to other classification variants.

B Supplementary Definitions

B.1 Focal Loss in Multi-class Classification

According to the derivation of the binary Focal Loss in (Lin et al., 2017), we extend it into the multi-class scenarios. In a multi-class classification task using Cross-entropy (CE) Loss, the CE loss can be written as follows:

$$\mathcal{L}_{\text{CE}}(\boldsymbol{p}, \boldsymbol{y}) = -\sum_{i=1}^{K} y_i \log(p_i) = \begin{cases} -\log(p_1) & \text{if } y_1 = 1 \\ -\log(p_2) & \text{if } y_2 = 1 \\ \quad\vdots & \\ -\log(p_K) & \text{if } y_K = 1, \end{cases}$$

where $\mathbf{y}$ is the one-hot embeded label.

For any class i, we use p_t to represent the confidence degree of the model's prediction as the following:

$$p_t = \begin{cases} p_i & \text{if } y_i = 1 \\ 1 - p_i & \text{otherwise}, \end{cases}$$

where $t = i$. To be consistent with the original Focal Loss in (Lin et al., 2017), we use t to represent the class index instead of i, and t is actually identical to i.

In order to solve class imbalance, Focal Loss assigns an auto-determined weight $(1-p_t)^\gamma$ and a pre-determined weight α_t to each class t. Finally, we define the Focal Loss for multi-class classification tasks as:

$$\mathcal{L}_{\text{FL}}(p_t) = -\sum_{t=1}^{K} \alpha_t (1 - p_t)^\gamma \log(p_t).$$

B.2 Class-wise Labels and Probabilities

In a multi-class classification problem, each instance in the dataset belongs to one of several classes. Let's denote the set of classes as K and a particular class of interest as $k \in K$. In this context, we can define positive and negative samples for class k.

- **Positive Samples** (X_k^+): The positive samples of class k satisfy that: $X_k^+ = \{x_i : y_i = k\}$, where x_i is the input and y_i is the corresponding label.

- **Negative Samples** (X_k^-): Similarly, the negative samples of class k satisfy that: $X_k^- = \{x_i : y_i \neq k\}$

According to the positive and negative samples, we can then get the positive and negative probability for class k.

- **Positive Probability** (p_k^+): When a positive instance is fed into the model, the predicted probability of class k is termed the positive probability. Since the posterior activation function is used in the output layer, the output posterior probability p^+ is a vector of length k. Therefore, the positive probability for class k can be expressed as p_k^+.

- **Negative Probability** (p_k^-): Similarly, when a negative sample is input into the model, the kth element of the output probability vector represents the negative probability, denoted as p_k^-. It's essential to note that any negative sample associated with the other $(K-1)$ classes contributes to p_k^-.

When using an auxiliary dataset to estimate the probabilities of the target training batch in FL, we denote the estimated positive and negative probabilities as $\hat{p}_k^+$ and $\hat{p}_k^-$, respectively.

In a batch size of B, we aim to recover the labels of each instance within the batch, i.e., $\mathbf{y} = [y^{(1)}, y^{(2)}, \cdots, y^{(B)}]$. As this is a multi-class classification problem, the ground-truth labels $\mathbf{y}$ can also be represented by the occurrences of each class: $\mathbf{y} = [n_1, n_2, \cdots, n_K]$, where n_k is the number of samples belonging to class k and K is the number of total classes.

- **Class-wise Labels**: The class-wise labels can be defined as: $n_k = \sum_{i=1}^{B} \delta(y^{(i)} = k)$. Here, n_k is the number of samples belonging to class k, B is the batch size, $y^{(i)}$ is the true class label of the ith instance in the batch, and $\delta(\cdot)$ is the Kronecker delta function, which equals 1 if the condition inside is true and 0 otherwise.

C PROOFS

C.1 PROOF OF THEOREM 1

Theorem 3 (Gradient of Focal Loss)**.** *For a K-class classification task using the **focal loss** function and **posterior** activation, we can derive that the gradient of logit z_j as follows:*

$$\nabla_{z_j} \mathcal{L}_{FL} = \sum_{t=1}^{K} \Phi(\alpha_t, p_t, \gamma) \cdot (p_j - \delta_{tj}),$$

where $\Phi(\alpha_t, p_t, \gamma) = \alpha_t (1-p_t)^\gamma \left(1 - \gamma \frac{p_t \log p_t}{1-p_t}\right)$ and $\forall t \in K$, we have $\Phi(\alpha_t, p_t, \gamma) \geq 0$. Besides, δ_{tj} is the Kronecker delta, which equals 1 if $t = j$ and 0 otherwise.

Proof. According to Equation (1), we substitute the last p_t with its posterior formula $p_t = \frac{e^{z_t}}{\sum_{k=1}^{K} e^{z_k}}$, and obtain the transformed focal loss function:

$$\begin{aligned}
\mathcal{L}_{\text{FL}} &= -\sum_{t=1}^{K} \alpha_t (1-p_t)^\gamma \log \frac{e^{z_t}}{\sum_{k=1}^{K} e^{z_k}} \\
&= \sum_{t=1}^{K} \alpha_t (1-p_t)^\gamma \log \sum_{k=1}^{K} e^{z_k} - \sum_{t=1}^{K} \alpha_t (1-p_t)^\gamma z_t.
\end{aligned}$$

Let $\hbar = (1-p_t)^\gamma$, then we can deduce the gradient of logit z_j as follows:

$$\begin{aligned}
\nabla_{z_j}\mathcal{L}_{\text{FL}} &= \sum_{t=1}^{K}\alpha_t\frac{\partial\hbar}{\partial z_j}\log\sum_{k=1}^{K}e^{z_k} + \sum_{t=1}^{K}\alpha_t(1-p_t)^\gamma p_j - \sum_{t=1}^{K}\alpha_t\frac{\partial\hbar}{\partial z_j}z_t - \alpha_j(1-p_j)^\gamma \\
&= \sum_{t=1}^{K}\alpha_t\frac{\partial\hbar}{\partial z_j}\left(\log\sum_{k=1}^{K}e^{z_k} - z_t\right) + \sum_{t=1}^{K}\alpha_t(1-p_t)^\gamma(p_j-\delta_{tj}) \\
&= \sum_{t=1}^{K}\alpha_t(1-p_t)^\gamma\left(1-\gamma\frac{p_t\log p_t}{1-p_t}\right)(p_j-\delta_{tj}) \\
&= \sum_{t=1}^{K}\Phi(\alpha_t,p_t,\gamma)\cdot(p_j-\delta_{tj}).
\end{aligned}$$

□

C.2 Proof of Theorem 2

Theorem 4 (Label Recovery Attack)**.** *For the attacker with an **auxiliary dataset**, he can recover the class-wise labels λ_j of the target batch according to the averaged gradient $\overline{\nabla b}_j$ and the estimated posterior probabilities $\hat{p}_j^+$ and $\hat{p}_j^-$ as follows:*

$$\lambda_j = B\cdot\frac{(\hat{p}_j^- - y_j^-) - \overline{\nabla b}_j/\hat{\varphi}_j}{(\hat{p}_j^- - y_j^-) - (\hat{p}_j^+ - y_j^+)},$$

where y_j^+ and y_j^- are the pre-set label embeddings of class j, $\hat{\varphi}_j = \frac{1}{\tau}\Phi(\alpha_j,\hat{p}_j^+,\gamma)$ is an coefficient related to the jth class, and B is the batch size.

Proof. Since $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}$, we can deduce that $\nabla\boldsymbol{b} = \nabla\boldsymbol{z}$ and $\overline{\nabla b}_j = \overline{\nabla z}_j$. We expand the averaged gradient $\overline{\nabla z}_j$ as a summation of B terms and replace the posterior probability $p_j^{(n)}$ of each sample n with its estimated probabilities $\hat{p}_j^+$ and $\hat{p}_j^-$. Because $\Phi(\alpha_j,p_j^{(n)},\gamma)$ is only related to the positive samples of the jth class, we can replace $p_j^{(n)}$ with $\hat{p}_j^+$. So we have:

$$\hat{\varphi}_j = \frac{1}{\tau}\Phi(\alpha_j,\hat{p}_j^+,\gamma) = \frac{1}{\tau}\alpha_j(1-\hat{p}_j^+)^\gamma\left(1-\gamma\frac{\hat{p}_j^+\log\hat{p}_j^+}{1-\hat{p}_j^+}\right).$$

Assume that the first λ_j samples belong to the jth class, and the rest $(B-\lambda_j)$ samples belong to other classes. Then from the first row of Table 1, we can derive that:

$$\begin{aligned}
\overline{\nabla b}_j = \frac{1}{B}\sum_{n=1}^{B}\nabla b_j^{(n)} &= \frac{1}{B}\left\{\sum_{n=1}^{\lambda_j}\varphi_j^{(n)}\left[p_j^{(n)}-y_j^{(n)}\right] + \sum_{n=\lambda_j+1}^{B}\varphi_j^{(n)}\left[p_j^{(n)}-y_j^{(n)}\right]\right\} \\
&\approx \frac{1}{B}\left\{\lambda_j\,\hat{\varphi}_j\left(\hat{p}_j^+ - y_j^+\right) + (B-\lambda_j)\hat{\varphi}_j\left(\hat{p}_j^- - y_j^-\right)\right\}.
\end{aligned}$$

Therefore, we can finally derive that:

$$\lambda_j = B\cdot\frac{\hat{\varphi}_j\left(\hat{p}_j^- - y_j^-\right) - \overline{\nabla b}_j}{\hat{\varphi}_j\left(\hat{p}_j^- - y_j^-\right) - \hat{\varphi}_j\left(\hat{p}_j^+ - y_j^+\right)} = B\cdot\frac{(\hat{p}_j^- - y_j^-) - \overline{\nabla b}_j/\hat{\varphi}_j}{(\hat{p}_j^- - y_j^-) - (\hat{p}_j^+ - y_j^+)}.$$

□