ECE 273 Project: Federated Learning

Yein Kim

Department of Electrical and Computer Engineering,
University of California, San Diego

Spring 2021

Talk Outline

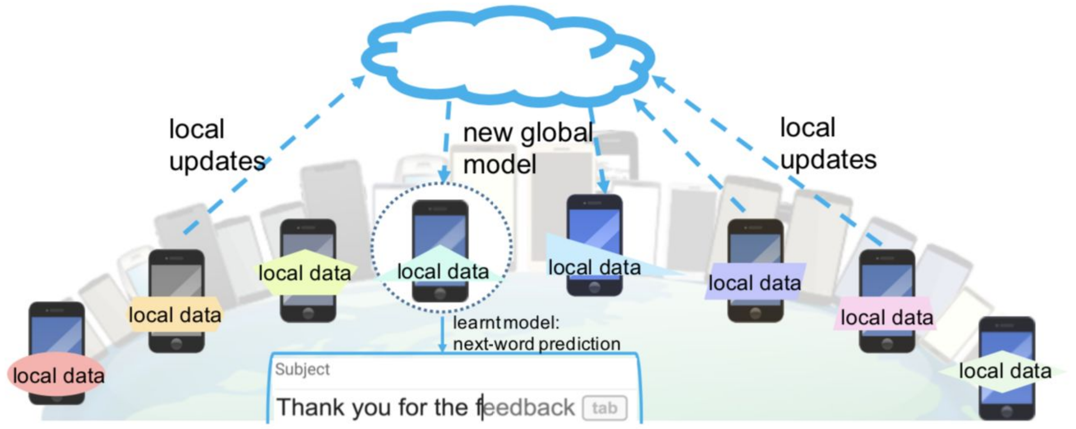
- Objectives
- Background
- Methods & Results
  1. Aggregation mechanisms
     - Federated averaging (baseline)
     - Byzantine robust aggregation
  2. Deep leakage from gradients
- Discussion

Objectives

- Problem: given data distributed across multiple edge devices, how can we train a single common model?

Objectives

- Problem: given data distributed across multiple edge devices, how can we train a single common model?
- Federated Learning (FL)
    - Local training: each edge user trains a model with its private data
    - Global aggregation: trains the global model with local updates



Objectives

- Goal: study the applicability and challenges of FL in the domain of image classification
  - Convergence speed of federated learning model
  - Vulnerability to “malicious” edge users
  - Possible leak of private data

Background

Why Federated Learning?[1]

1. Privacy sensitive data
   a. Google's Gboard next-word prediction model[2]
   b. Healthcare data[3]
2. Highly distributed data
   a. Smart manufacturing systems[4]
3. Limited communication networks
   a. Underwater & unmanned aerial vehicle networks[5][6]

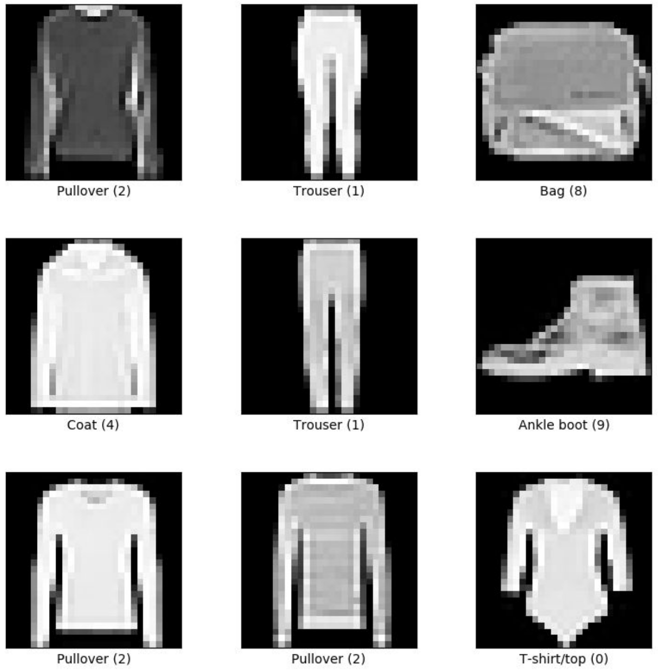
[1] Gafni, Tomer, et al.“Federated Learning:A Signal Processing Perspective”, arXiv preprint arXiv:2103.17150 (2021).
[2] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, “Federated learning for mobile keyboard prediction,” arXiv preprint arXiv:1811.03604, 2018.
[3] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, “Federated learning for healthcare informatics,” Journal of Healthc. Inform. Res., vol. 5, no. 1, pp. 1–19, 2021.
[4] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, “A blockchained federated learning framework for cognitive computing in Industry 4.0 networks,” IEEE Trans. Ind. Informat., 2020.
[5] D. Kwon, J. Jeon, S. Park, J. Kim, and S. Cho, “Multi-agent DDPG based deep learning for smart ocean federated learning IOT networks,” IEEE Internet Things J., 2020.
[6] B. Brik, A. Ksentini, and M. Bouaziz, “Federated learning for UAVs enabled wireless networks: Use cases, challenges, and open problems,” IEEE Access, vol. 8, pp. 53 841–53 849, 2020.

Background

Datasets & Models

1. Fashion-MNIST[7]
- 60K training images, 10K test images
- Grey-scale images with dimension 1x28x28
- 10 classes

- Trained model: 3-layer CNN
  - 2 convolutional layers (**sigmoid** activation function)
  - 1 fully-connected output layer



[7] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv:1708.07747 [cs.LG]

Background

Datasets & Models

2. CIFAR10[8]

- 50K training images, 10K test images
- RGB images with dimension 3x32x32
- 10 classes
- Trained model: ResNet-18 CNN[9]
  - ReLU replaced with **sigmoid** functions

[8] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv:1708.07747 [cs.LG]
[9] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition(CVPR).770–778.

Method

FL Model Formulation

- 1 central server and *M* edge users
- Model parameters $\theta \in \mathbb{R}^d$
  - Local model parameters $\theta^i \in \mathbb{R}^d$
- Local data $D^i = \{(x_n^i, y_n^i)\}_{n=1}^{n_i}$
- Loss function $L(x_n^i, y_n^i; \theta^i)$: cross entropy

Method

FL Model Formulation

- 1 central server and M edge users
- Model parameters $\theta \in \mathbb{R}^d$
  - Local model parameters $\theta^i \in \mathbb{R}^d$
- Local data $D^i = \{(x_n^i, y_n^i)\}_{n=1}^{n_i}$
- Loss function $L(x_n^i, y_n^i; \theta^i)$: cross entropy

1. Local training

- Objective function:

$$f(\theta^i, D^i) = \frac{1}{n_i} \sum_{n=1}^{n_i} L(x_n^i, y_n^i; \theta^i)$$

- Configurations: local data distribution, training iterations, learning rate

Method

FL Model Formulation

- 1 central server and *M* edge users
- Model parameters $\theta \in \mathbb{R}^d$
  - Local model parameters $\theta^i \in \mathbb{R}^d$
- Local data $D^i = \{(x_n^i, y_n^i)\}_{n=1}^{n_i}$
- Loss function $L(x_n^i, y_n^i; \theta^i)$: cross entropy

2. Global aggregation
- Objective function: $F(\theta)$
- Configurations: aggregation frequency (local training iterations), % of local models selected for aggregation

Method

FL Model Formulation

- 1 central server and *M* edge users
- Model parameters $\theta \in \mathbb{R}^d$
    - Local model parameters $\theta^i \in \mathbb{R}^d$
- Local data $D^i = \{(x_n^i, y_n^i)\}_{n=1}^{n_i}$
- Loss function $L(x_n^i, y_n^i; \theta^i)$: cross entropy

2. Global aggregation

- Objective function: $F(\theta)$
- Configurations: aggregation frequency (local training iterations), % of local models selected for aggregation

Method: Federated Averaging

- Update rule: at each aggregation epoch *t* (*M* edge users, *N* training images)
  - $p_i = \frac{n_i}{N}$
  - $\theta_{t+1} = \frac{M}{|g_t|} \sum_{i \in g_t} p_i \theta^i_{t+1}$

    → weighted average of local models chosen at *t*, g_t

  The new global model parameters propagated to edge users

Method: Federated Averaging

- Update rule: at each aggregation epoch *t* (*M* edge users, *N* training images)
  - $p_i = \frac{n_i}{N}$
  - $$\theta_{t+1} = \frac{M}{|g_t|} \sum_{i \in g_t} p_i \theta^i_{t+1}$$

    $\rightarrow$ weighted average of local models chosen at *t*, g_t

  The new global model parameters propagated to edge users
- Question: Do FL models converge as fast as centralized training (CL) models?
  - Local data distribution: iid vs non-iid
  - %local models aggregated out of all models

Results: Federated Averaging

Experiment Setting

- Data: Fashion-MNIST
- Total 20 epochs
  - CL: training iterations
  - FL: global aggregations
- Local training configurations: SGD
  (500 iterations, 20 images per batch, learning rate 0.01, momentum 0.9)

Variables

- Local data distribution
  - iid: $|D^i| = N/M$
  - non-iid: user data distributed by Dirichlet distribution with concentration hyperparameter 0.9 for each class
- Total edge users M = 100, 20, 10
  - 10 users chosen per aggregation epoch (10%, 50%, 100% models aggregated)

Results: Federated Averaging

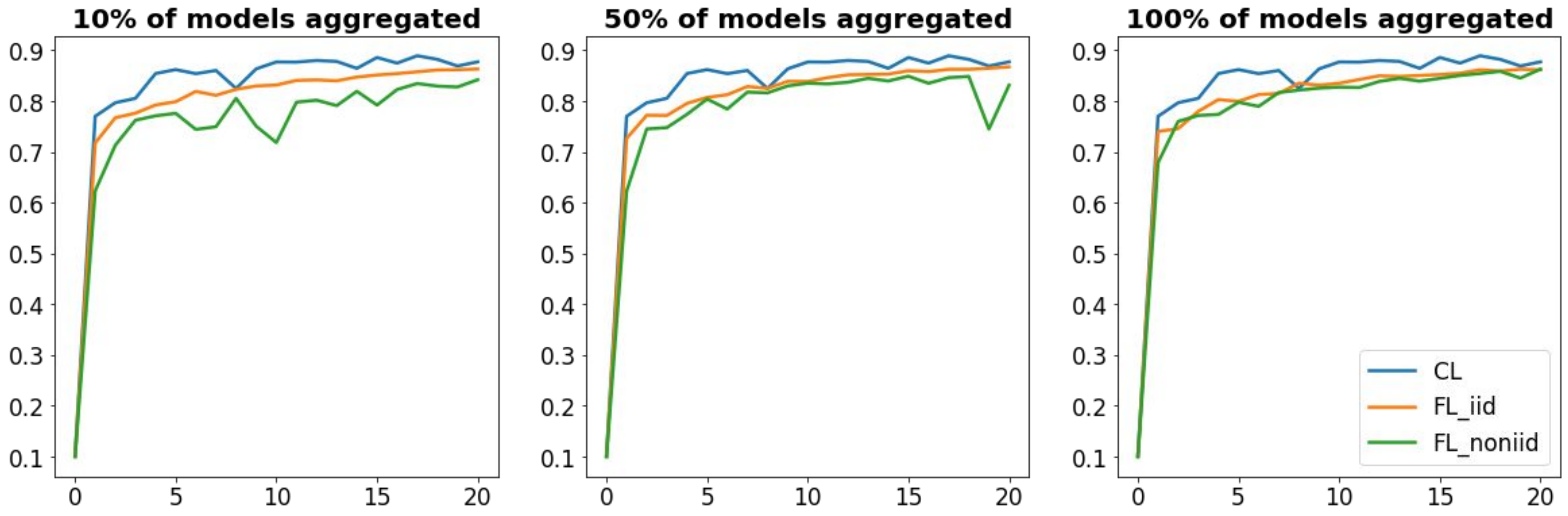
1. Final test accuracy (%)
- CL model: **87.74**
- FL model:

% aggregated models	iid	non-iid
10	86.32	84.21
50	86.75	83.15
100	86.17	86.29

- slightly lower than CL model’s
- consistent with iid local data distribution
- Tends to increase with higher %aggregated models

Results: Federated Averaging

1. Fashion-MNIST
 - FL vs CL model convergence over 20 epochs



- Overall, the convergence of FL models comparable to the CL model's
- FL model convergence is more stable with
  1) iid local distribution 2) higher ratio of local models aggregated

Method: Byzantine-Robust Aggregation

- Attack scenario: *a*% of Byzantine local models train on “poisoned” dataset
  - Every training label *y* is replaced with *9-y*

Method: Byzantine-Robust Aggregation

- Attack scenario: *a*% of Byzantine local models train on “poisoned” dataset
- Every training label *y* is replaced with *9-y*

- Update rule

1. Gradient descent algorithm: at each aggregation epoch *t*,

$$\delta_{t+1}^i = \theta_t^i - \theta_{t+1}^i$$

$$\delta_{t+1} = \begin{cases} med\{\delta_{t+1}^i\} & \text{for } i \in g_t \ (1) \\ trmean_\beta\{\delta_{t+1}^i\} & \text{for } i \in g_t \ (2) \end{cases}$$

$$\theta_{t+1} = \theta_t - \eta\delta_{t+1}$$

  - Trimmed mean: mean computed after removing $\beta/2$ % of largest and $\beta/2$ % of smallest values (assume $\beta = a$)
  - Local training configurations: computes loss gradient (**1 iteration**, all training images per batch, learning rate 1)

Method: Byzantine-Robust Aggregation

- Attack scenario: *a*% of Byzantine local models train on "poisoned" dataset
  - Every training label *y* is replaced with *9-y*

- Update rule

2. One-round algorithm

$$\theta^i = \text{argmin}_\theta f(\theta^i, D^i)$$

$$\theta = med\{\theta^i\}$$

- Local training configurations: SGD
  (**500 iterations,** 10% training images per batch, learning rate 0.01, momentum 0.9)

Results: Byzantine-Robust Aggregation

Experiment setting

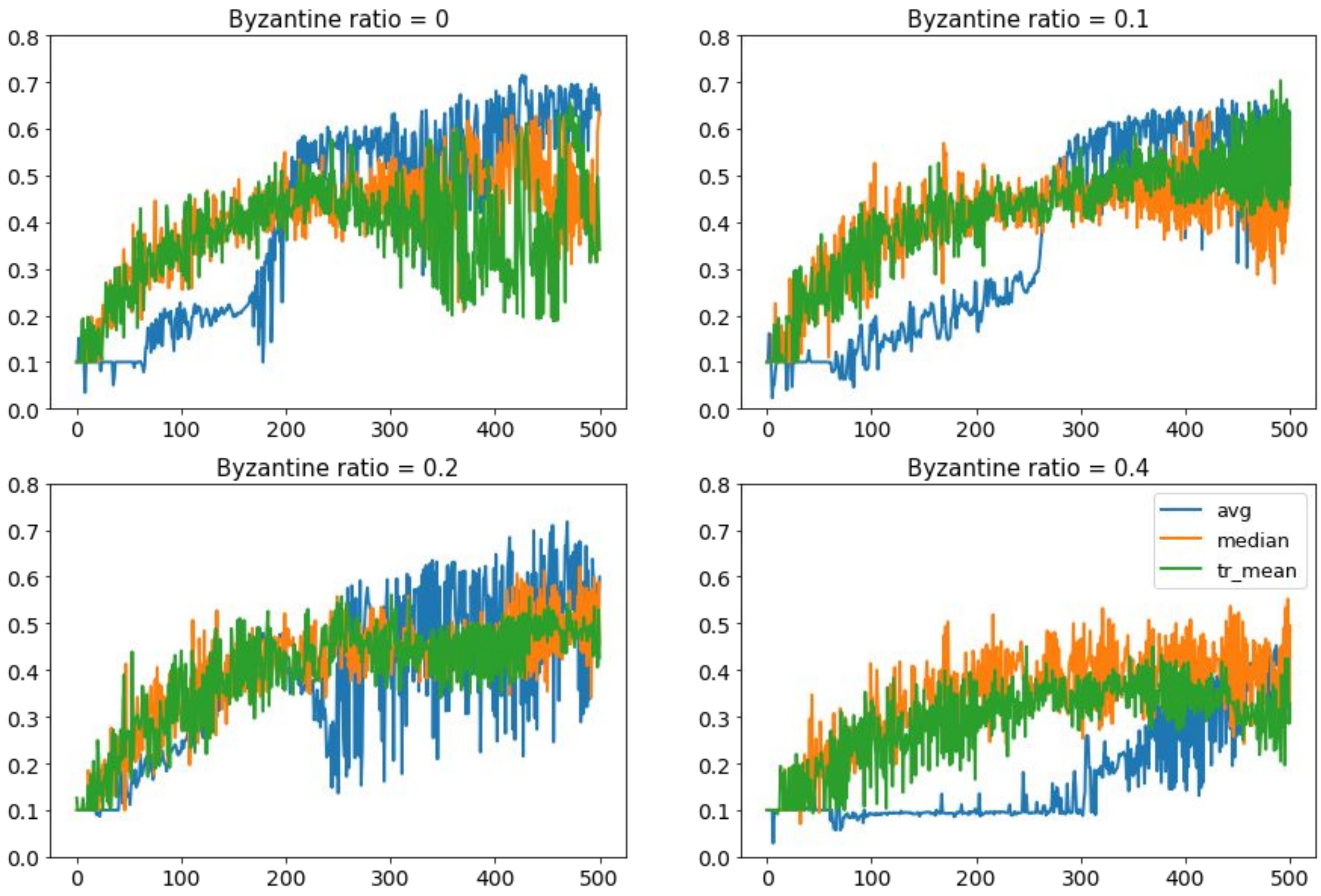
- Data: Fashion-MNIST
- M = 20 users
  - all local updates selected at each aggregation epoch
- iid local data distribution
- 500 local training iterations in total
  - Gradient descent algorithm: 500 aggregation epochs
  - One-round algorithm: 1 aggregation epoch

Variable

- Byzantine ratio = 0, 0.1, 0.2, 0.4

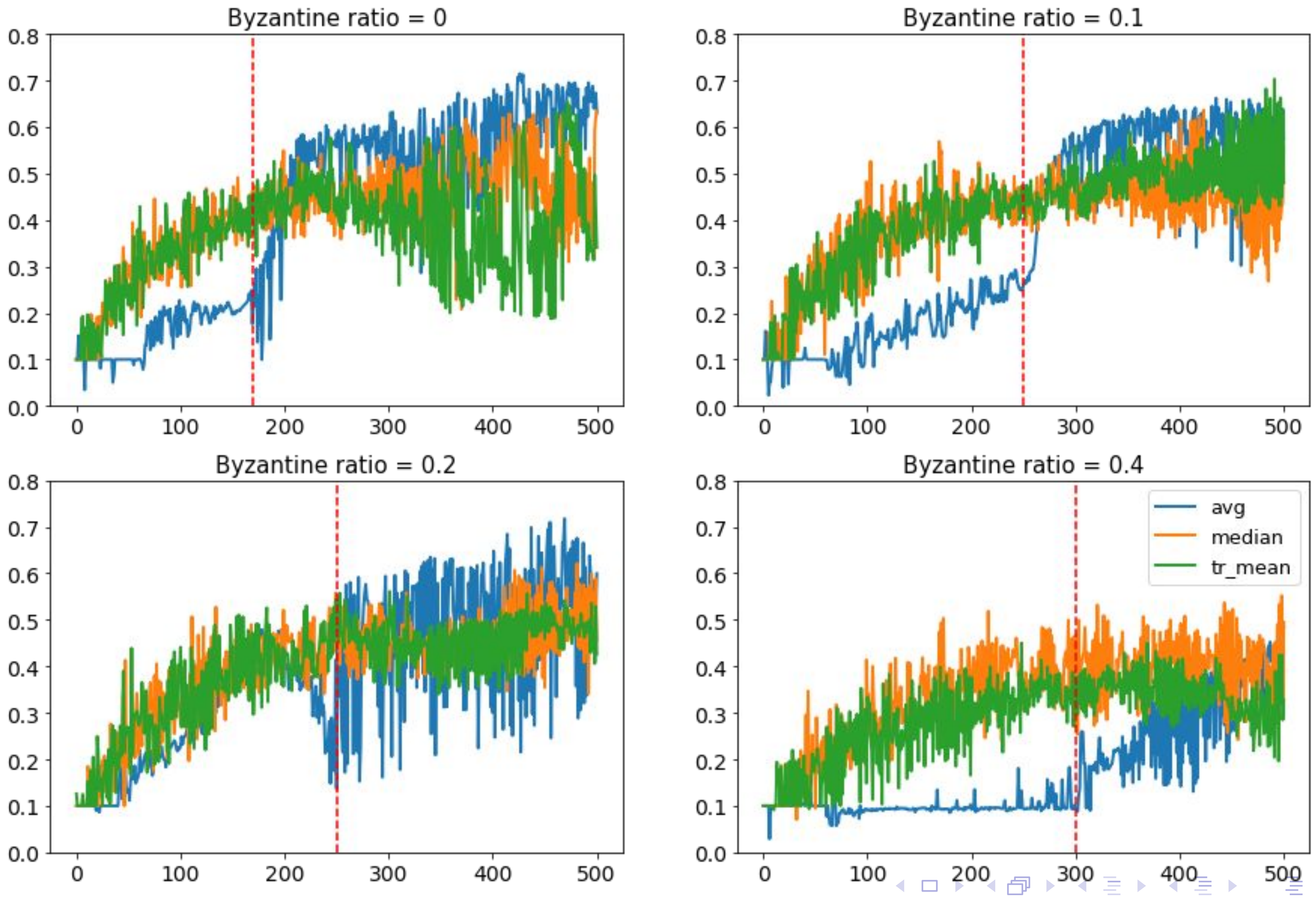
Results: Byzantine-Robust Aggregation

1. Gradient descent test accuracy with different Byzantine ratios



Results: Byzantine-Robust Aggregation

1. Gradient descent test accuracy with different Byzantine ratios



Results: Byzantine-Robust Aggregation

2. One-round algorithm final test accuracy (%) *

Byzantine ratio	Fed Avg	Median
0	72.91	72.62
0.1	71.41	72.66
0.2	61.01	72.37
0.4	**40.81**	**70.37**

Overall, median aggregation is more robust to the Byzantine attack

- Federated average's accuracy declines while the median's is consistent as the Byantine ratio increases
- Median's accuracy is comparable to the federated average's when there is no Byzantine users

*** Note: Result changed after the presentation**

Method: Deep Leakage from Gradients

- Experiment setting
  - Local training: each user sends the loss gradient (1 local iteration)
  - Global aggregation after each local iteration

Method: Deep Leakage from Gradients

- Experiment setting
  - Local training: each user sends the loss gradient (1 local iteration)
  - Global aggregation after each local iteration

- Goal: recover a **batch** of training data (x, y) from the local gradient
- Idea: iteratively find dummy data whose gradient is close to the local gradient
  - Given a global parameter, $\theta \in \mathbb{R}^d$ and the local gradient,

$$\nabla\theta = \frac{\partial L(x, y; \theta)}{\partial \theta}$$

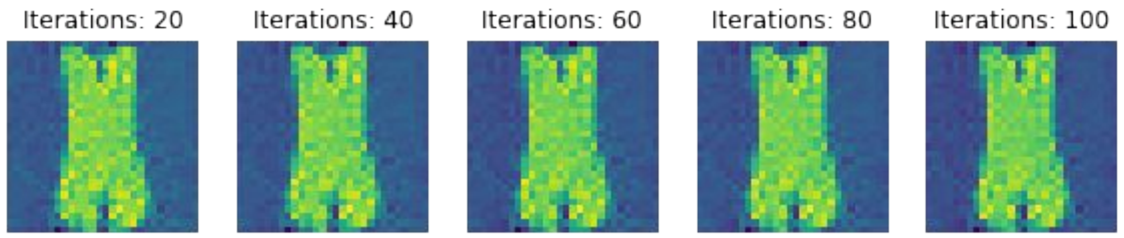
  - Initialize dummy training data: (x’, y’)
  - Compute the dummy gradient

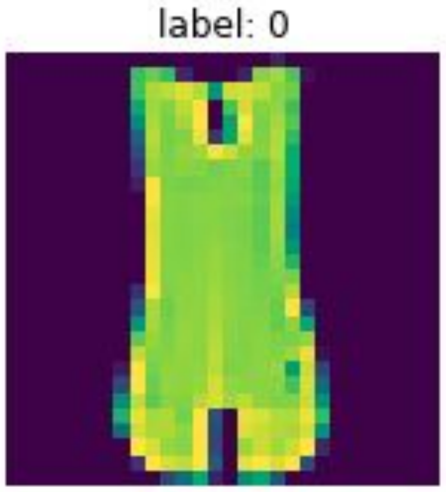
$$\nabla\theta' = \frac{\partial L(x', y'; \theta)}{\partial \theta}$$

  - $x'^*, y'^* = \text{argmin}_{x', y'} ||\nabla\theta' - \nabla\theta||^2$

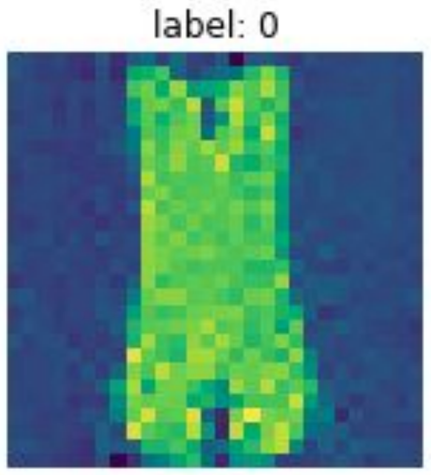
Result: Deep Leakage from Gradients

1. Batch size = 1, iterations = 100
 - Fashion-MNIST





Original

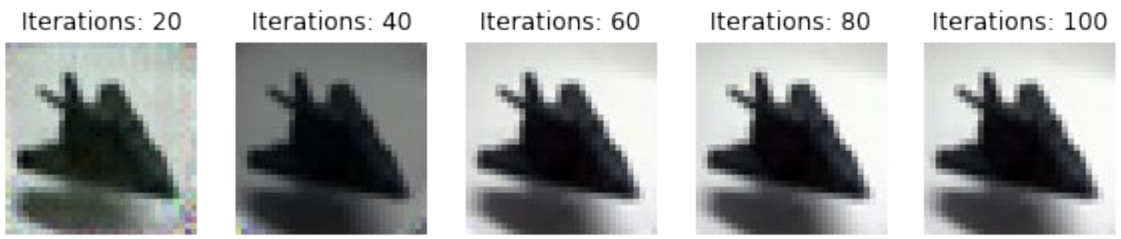


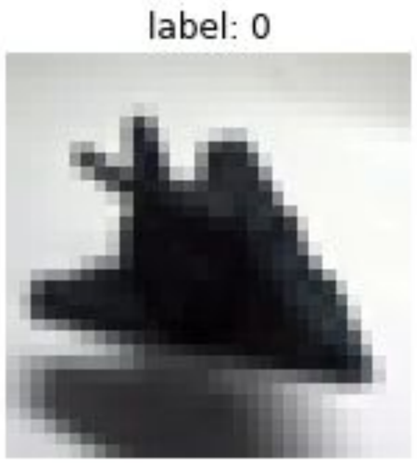
Recovered

* **Note: Result changed after the presentation**

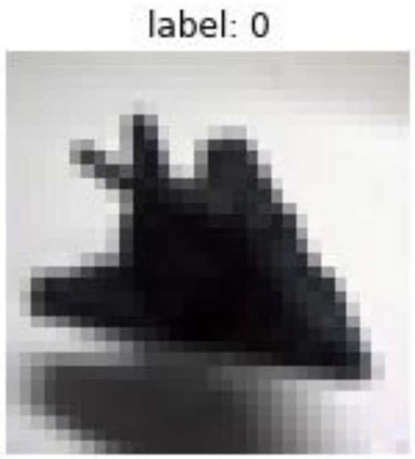
Result: Deep Leakage from Gradients

1. Batch size = 1, iterations = 100
- CIFAR10





Original



Recovered

*** Note: Result changed after the presentation**

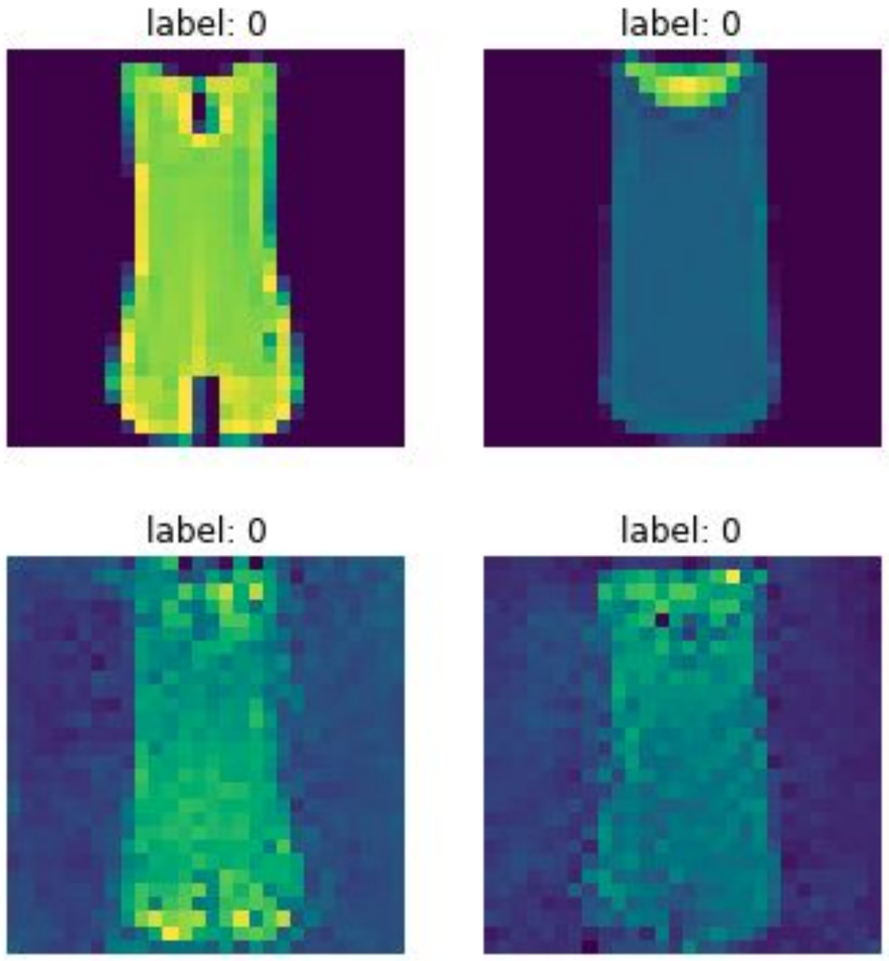
Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100

- Fashion-MNIST: poor recovery of images from the same class

Original

Recovered

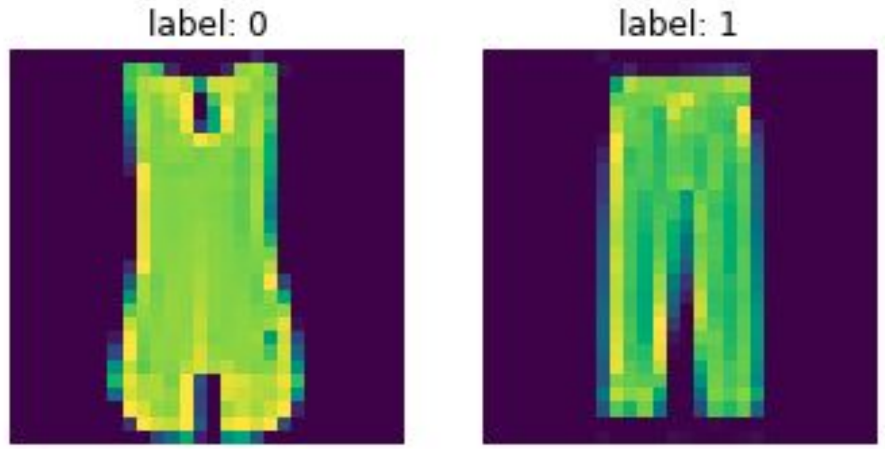


*** Note: Result changed after the presentation**

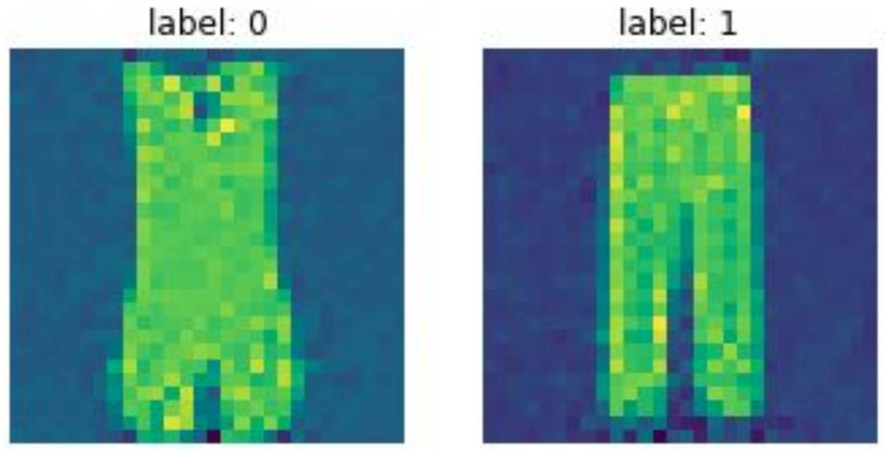
Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100
- Fashion-MNIST: better recovery of images from different classes

Original



Recovered



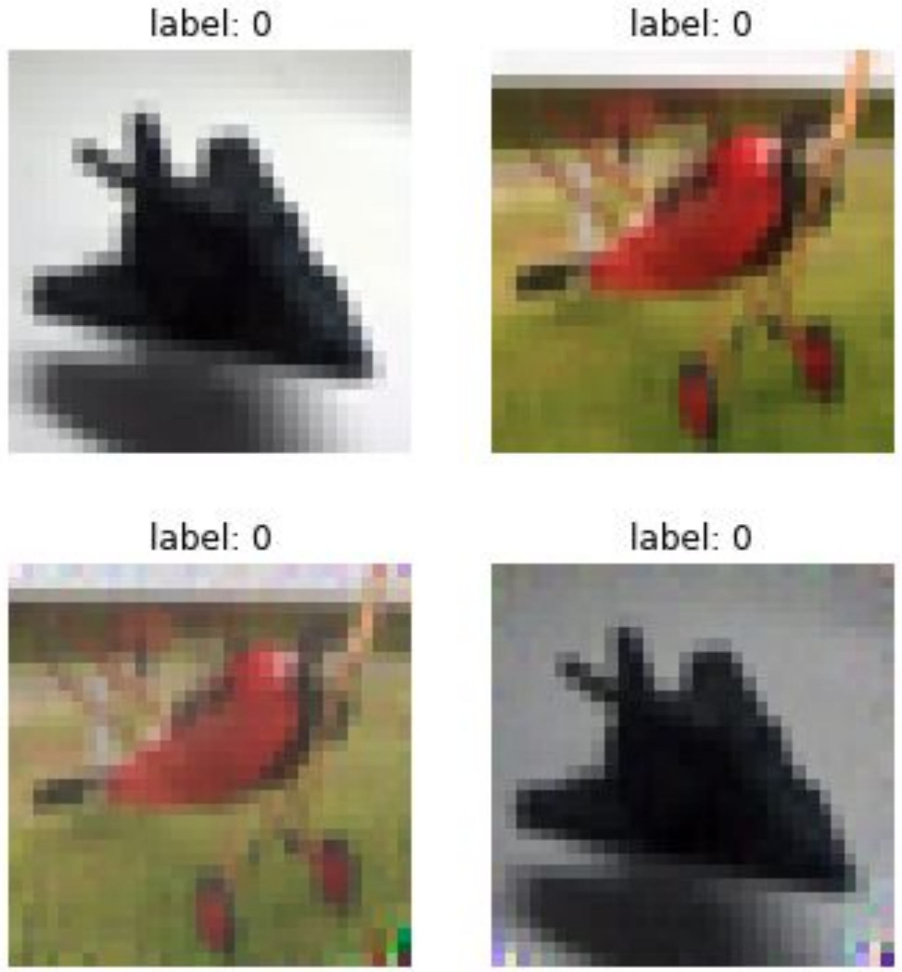
*** Note: Result changed after the presentation**

Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100
- CIFAR10: images and labels match (same class)

Original

Recovered

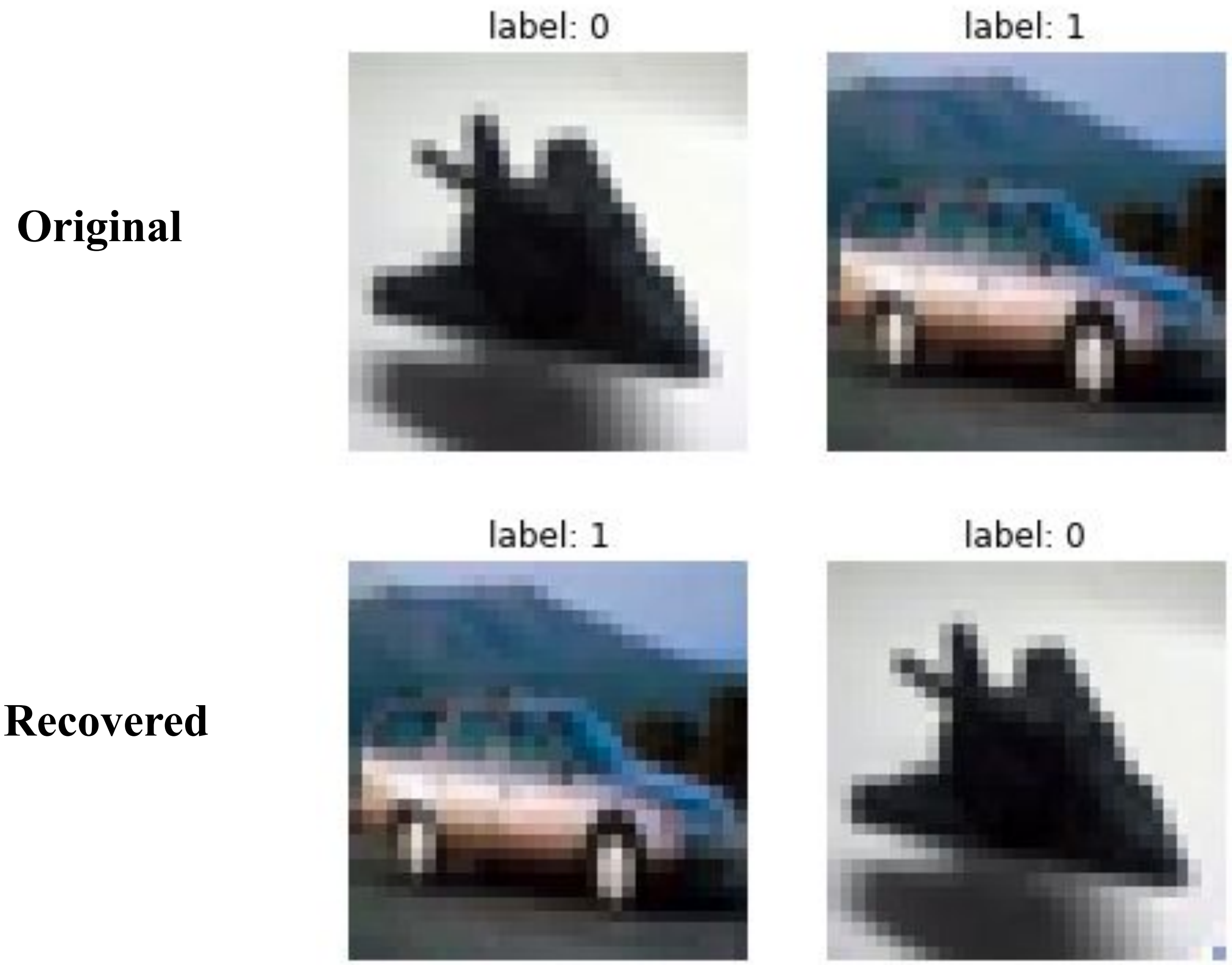


* Note: Result changed after the presentation

Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100

- CIFAR10: images and labels match (different classes)



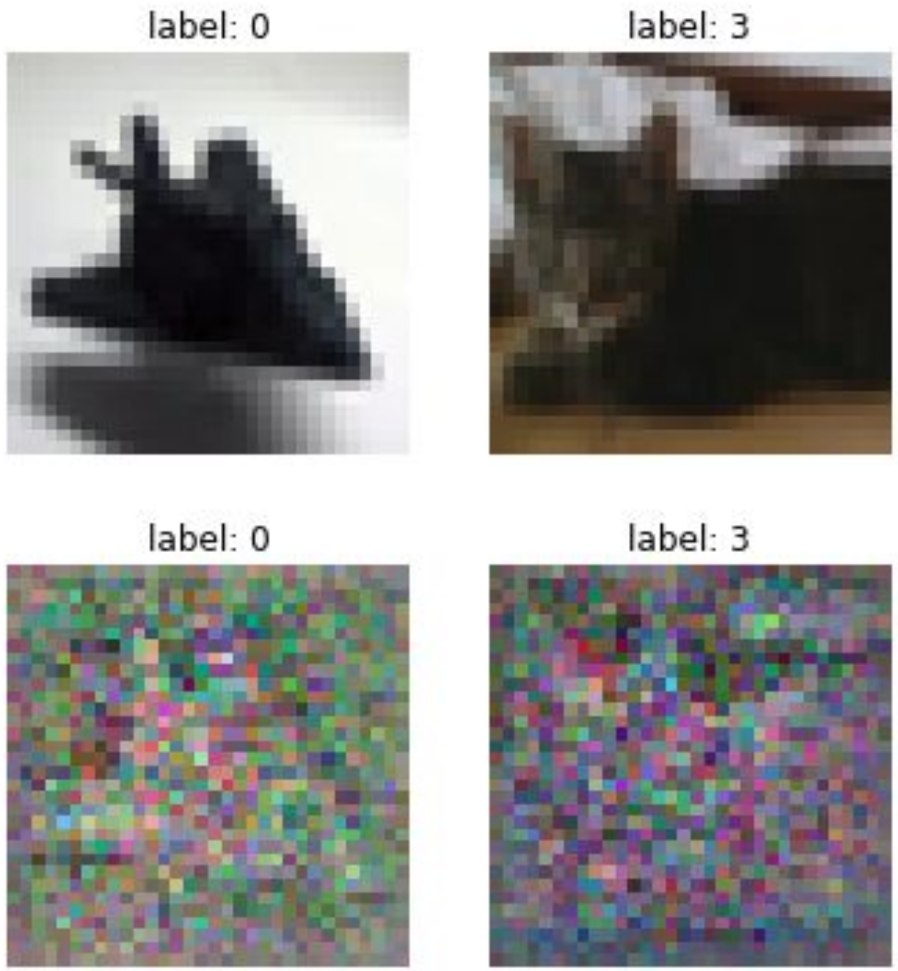
* Note: Result changed after the presentation

Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100
- CIFAR10: images mismatch (different classes)

Original

Recovered



*** Note: Result changed after the presentation**

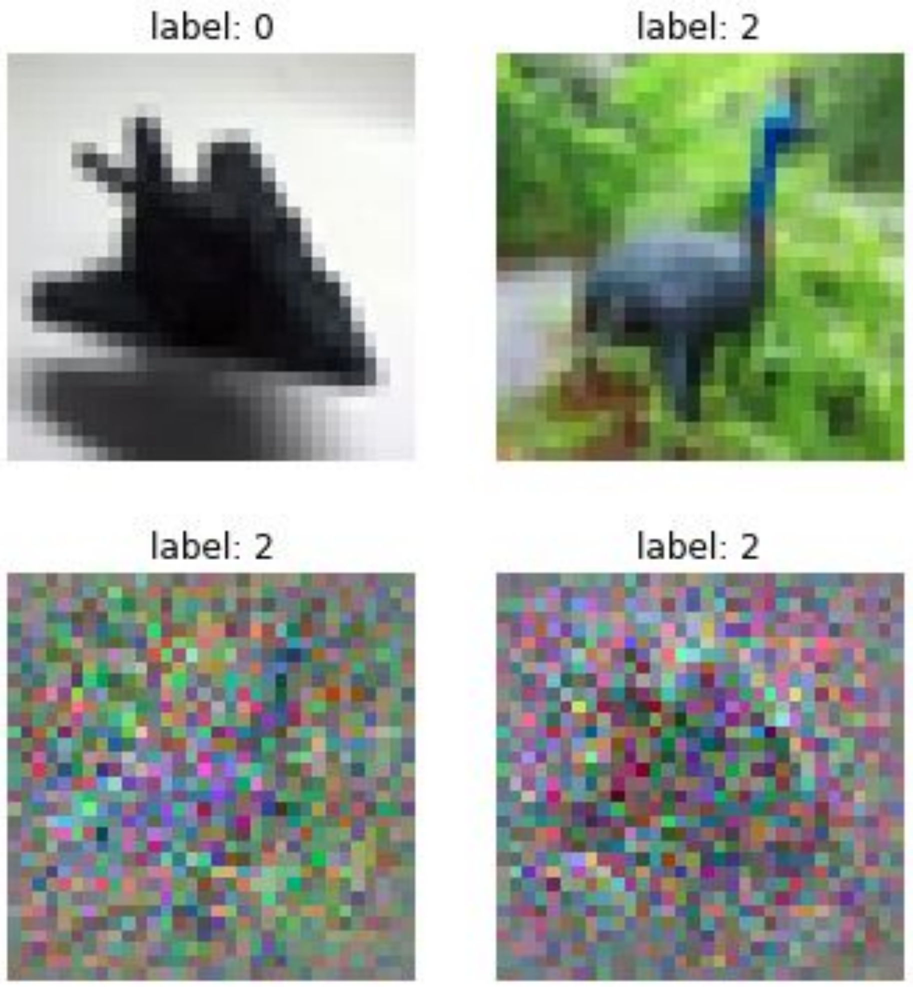
Result: Deep Leakage from Gradients

2. Batch size = 2, iterations = 100

- CIFAR10: images & labels mismatch (different classes)

Original

Recovered



* Note: Result changed after the presentation

Discussion

1. Federated Averaging (Fashion-MNIST): FL model convergence
    a. comparable to CL model
    b. slows down with non-iid local data distribution & lower ratio of models selected for aggregation

Discussion

1. Federated Averaging (Fashion-MNIST): FL model convergence
   a. comparable to CL model
   b. slows down with non-iid local data distribution & lower ratio of models selected for aggregation

2. Byzantine-Robust Aggregation (Fashion-MNIST):
   a. Gradient descent: with a higher Byzantine ratio,
      median & trimmed mean model convergences are more stable than the federated averaged model's
   b. One-round: median model converges faster with a higher Byzantine ratio*
   c. Next steps: check the algorithms' robustness on complex datasets and different edge user attacks
      i. experiment on CIFAR10
      ii. experiment with model poisoning attack[10]

*** Note: Result updated after the presentation**

[10] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing Federated Learning through an Adversarial Lens. In Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97), Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 634–643.

Discussion

3. Deep Leakage from Gradients
   a. Complexity of images matters
      (F-MNIST dummy images converge faster)
   b. Batch size matters (higher rate of successful recovery)
   c. Conflicting results*: harder to recover images
      of the same class (F-MNIST) vs of different classes (CIFAR-10)
   d. Limitation: assumes twice-differentiability of function
      (images not recovered with ReLU activation function)

*** Note: Result changed after the presentation**