ResSFL Supplementary Materials

Jingtao Li[1], Adnan Siraj Rakin[1], Xing Chen[1], Zhezhi He[2], Deliang Fan[1], Chaitali Chakrabarti[1]
[1] School of Electrical Computer and Energy Engineering, Arizona State University, Tempe, AZ
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai
[1]{jingtao1, asrakin, xchen382, dfan, chaitali}@asu.edu; [2]{zhezhi.he}@sjtu.edu.cn

1. Non-iid Performance

We test the ResSFL performance in non-i.i.d case using the same setting as in Table 6. We partition the dataset evenly to 10 clients, and each client has data from either the 10 class partition or 20 class partition or 50 class partition to create three non-i.i.d. cases. As shown in the table below, accuracy performance of ResSFL does not differ much from the vanilla scheme while ResSFL has a clear advantage of mitigating the inversion attack (demonstrated with higher MSE number). It is worth noting that there is an accuracy degradation even for the 50-class case compared to i.i.d. data (68.5% accuracy). Drop due to non-i.i.d data is a known problem that we plan to address in the near future.

	10-class	**20-class**	**50-class**
Vanilla (Accuracy/**MSE**)	25.4/**0.005**	53.3/**0.005**	65.4/**0.005**
ResSFL (Accuracy/**MSE**)	28.4/**0.042**	51.6/**0.043**	63.9/**0.047**

2. Detailed Architecture

Detailed inversion model architectures for L0 and L3 are shown in Figure 1 (We omit L1, L2 because they are just L3 with less number of ResBlocks). The architecture design is dependent on the size of intermediate activation. For an intermediate activation of size [128, 128, 8, 8], L0 inversion model consists of 2 Conv2d layers, and 2 ConvTranspoose2d layers that are necessary to upsample the 8x8 activation to 32x32. The L3 inversion model consists of 6 ResBlocks (BasicBlock in [2]) and 2 ConvTranspose2d layers.

```
Inversion_Model_L0(
  (m): Sequential(
    (0): Conv2d(128, 16, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1))
    (1): BatchNorm2d(16, eps=1e-05, momentum=0.1,
affine=True, track_running_stats=True)
    (2): ReLU()
    (3): ConvTranspose2d(16, 16, kernel_size=(3, 3),
stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (4): BatchNorm2d(16, eps=1e-05, momentum=0.1,
affine=True, track_running_stats=True)
    (5): ReLU()
    (6): ConvTranspose2d(16, 16, kernel_size=(3, 3),
stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (7): BatchNorm2d(16, eps=1e-05, momentum=0.1,
affine=True, track_running_stats=True)
    (8): ReLU()
    (9): Conv2d(16, 3, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1))
    (10): BatchNorm2d(3, eps=1e-05, momentum=0.1,
affine=True, track_running_stats=True)
    (11): Sigmoid()
  )
)
```

```
Inversion_Model_L3(
  (m): Sequential(
    (0): ResBlock(128, 64, BN = True)
    (1): ReLU()
    (2): ResBlock(64, 64, BN = True)
    (3): ReLU()
    (4): ResBlock(64, 64, BN = True)
    (5): ReLU()
    (6): ResBlock(64, 64, BN = True)
    (7): ReLU()
    (8): ResBlock(64, 64, BN = True)
    (9): ReLU()
    (10): ConvTranspose2d(64, 64, kernel_size=(3, 3),
stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (11): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (12): ReLU()
    (13): ConvTranspose2d(64, 64, kernel_size=(3, 3),
stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (14): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (15): ReLU()
    (16): ResBlock(64, 3, BN = True)
    (17): Sigmoid()
  )
)
```

Figure 1. Detailed architecture information for L0 and L3 Inversion model.

3. Optimization-based MI attack

We discussed training-based MI attack mitigation in the paper. There exists another MI attack known as optimization-based MI attack [1, 3], whose performance is not as strong. Optimization-based MI attack does not require an additional dataset. The intuition is that if the intermediate activation of the real image is the same as the intermediate activation of the fake image, the two images should look the same [3]. For each private image $\boldsymbol{x}_{priv}[i]$ with intermediate activation $\mathbf{A}_t[i]$ (indexed by i), the optimization process is as follows: a fake image $\boldsymbol{x}^*_{priv}[i]$ is randomly initialized (i.e. using Gaussian) and gets optimized:

$$\boldsymbol{x}^*_{priv}[i] = \underset{\boldsymbol{x}^*_{priv}[i]}{\arg\min}\, MSE(C^i(\boldsymbol{x}^*_{priv}[i]), \mathbf{A}_t[i]) \tag{1}$$

We follow the same experiment setting as the main paper. For the optimizer that is used to update the output image $\boldsymbol{x}^*_{priv}$, we use Adam optimizer with 0.8 learning rate. On a VGG-11 model with cut-layer of 2 (**without defense**) on CIFAR-100 dataset, optimization-based MI attack achieves 0.009 MSE, which is worse than training-based MI attack, which achieves 0.005 MSE as provided in the main paper.

We test the optimization-based MI attack on a model that applies our proposed ResSFL, by using a MI-resistant VGG11-cut2 model trained on CIFAR-10 dataset transfer to CIFAR-100 dataset. This is the model we demonstrate in the main paper with $\lambda = 0.3$ and bottleneck layers of $C8$-$S1$. We show early epochs resistance performance in Figure 2; the resistance of ResSFL starts with a very high value of 0.5 MSE at epoch 1, and gradually reduces to around 0.22 MSE at later epochs. In summary, our method generalizes well to optimization-based MI attack, and performs consistently well (above 0.20 MSE) during the SFL training process.

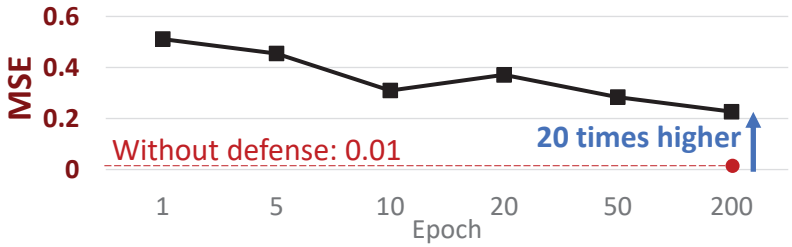
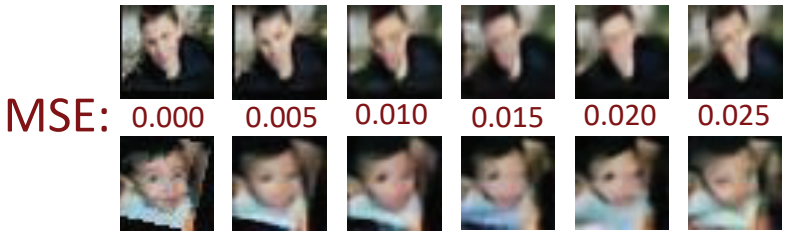


Figure 2. Resistance performance of the proposed ResSFL scheme against optimization-based MI attack. We achieve a consistently good resistance of around 0.20 MSE during SFL training process, which is 20 times higher than the model without defense (shown by the red-dashed line).

4. Justification on target MSE

In our work, we empirically set an MSE of 0.02 as our mitigation target. We provide visualizations for different MSE of the reconstructed image. We see that when the MSE is even above 0.020, the blurring effect makes it impossible for person identification. However, the target MSE 0.02 is highly task-specific and may not be suitable for other tasks (i.e. digit identification).



5. Other empirical evidence

In the table below, we provide more empirical results to demonstrate the ResSFL performance. We show (1) evidence that models pre-trained on other datasets can transfer to CIFAR-100, and (2) evidence that resistance on an SVHN pretrained

model can transfer to other datasets. We observe that transfer accuracy depends on the domain. For example, CIFAR-10 (similar domain to CIFAR-100) achieves good accuracy while MNIST (different domain) can only achieve 60.2% accuracy when transferring to CIFAR-100. Another factor towards successful resistance transfer is the difficulty factor (See "Transfer Performance to Different Datasets" in the paper). The source dataset needs to be a more difficult task in building up the resistance.

Target $\Leftarrow$	$\Leftarrow$ **Source dataset**			
CIFAR-100	CIFAR-10	SVHN	FaceScrub	MNIST
Accuracy	67.5	64.9	65.0	60.2
MSE (L3)	0.050	0.053	0.048	0.031

Source $\Rightarrow$	$\Rightarrow$ **Target dataset**			
SVHN	CIFAR-100	FaceScrub	CIFAR-10	MNIST
Accuracy	64.9	72.6	89.9	99.4
MSE (L3)	0.053	0.019	0.007	0.003

References

[1] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4829–4837, 2016. 2

[2] 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*, pages 770–778, 2016. 1

[3] Zecheng He, Tianwei Zhang, and Ruby B Lee. Attacking and protecting data privacy in edge–cloud collaborative inference systems. *IEEE Internet of Things Journal*, 8(12):9706–9716, 2020. 2