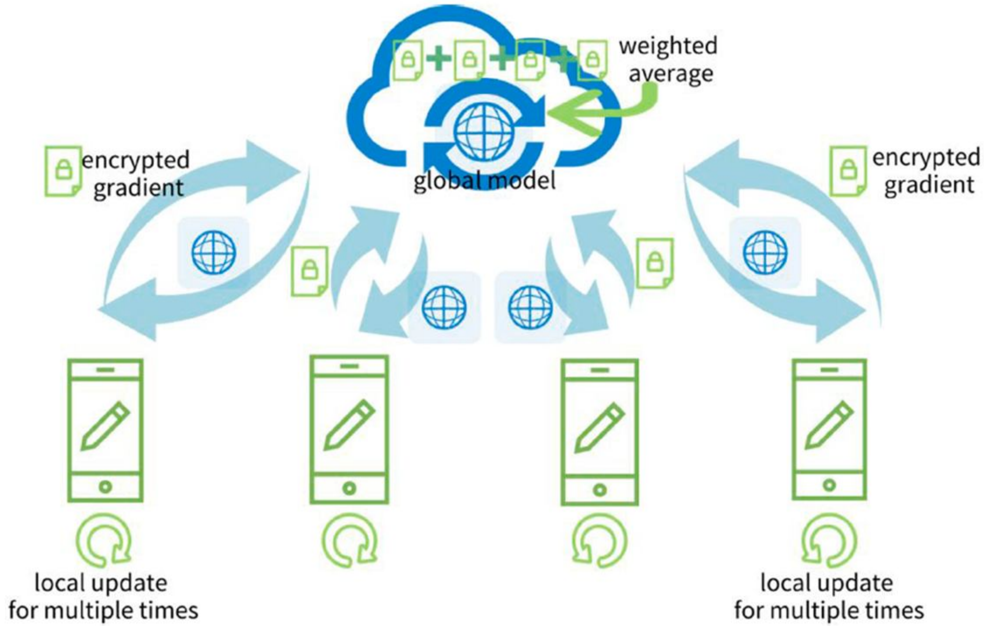
Deep Leakage from Gradients

Presentation by Alireza Mohseni

Introduction

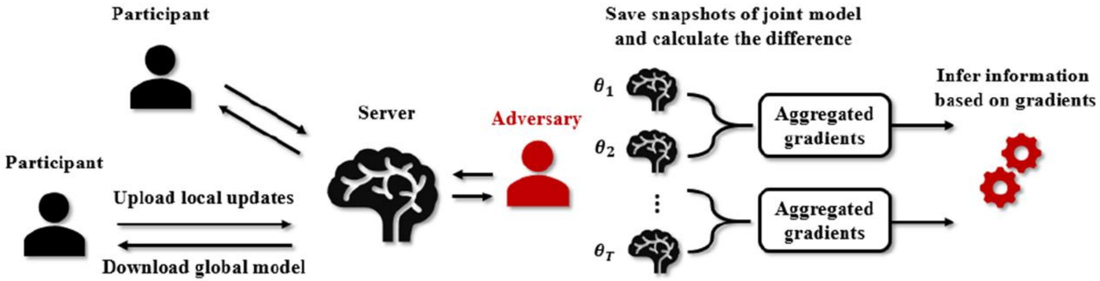
- Exchanging gradients is a widely used method in modern multi-node machine learning system
    - Data never leave the data owner
    - Communication efficiency

- For a long time people believed that gradients are safe to share



Background

- Some recent works develop learning-based methods to infer properties of the batch
    - Membership inference
    - Property inference
    - Synthesis similar images with GAN
- "shallow" leakages requires extra label information and can only generate similar synthetic images



Method-DLG

- We focus on the standard synchronous distributed training: At each step t, every node i samples a minibatch $(x_{t,i};\ y_{t,i})$ from its own dataset to compute the gradients

$$\nabla W_{t,i} = \frac{\partial \ell(F(\mathbf{x}_{t,i}, W_t), \mathbf{y}_{t,i})}{\partial W_t}$$

- The gradients are averaged across the N servers and then used to update the weights:

$$\overline{\nabla W_t} = \frac{1}{N} \sum_{j}^{N} \nabla W_{t,j}; \quad W_{t+1} = W_t - \eta \overline{\nabla W_t}$$

- Given gradients $\nabla W_{t,k}$ received from other participant k, we aim to steal participant k's training data $(x_{t,k};\ y_{t,k})$. Note F() and W_t are shared by default for synchronized distributed optimization.

Method-DLG

- To recover the data from gradients, we first randomly initialize a dummy input x' and label input y'. We then feed these "dummy data" into models and get "dummy gradients"

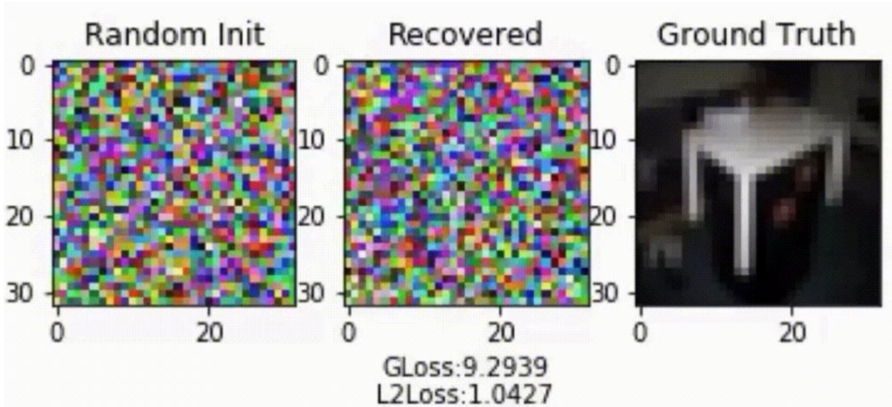
$$\nabla W' = \frac{\partial \ell(F(\mathbf{x}', W), \mathbf{y}')}{\partial W}$$

- Given gradients at a certain step, we obtain the training data by minimizing the following objective

$$\mathbf{x}'^*, \mathbf{y}'^* = \arg\min_{\mathbf{x}', \mathbf{y}'} ||\nabla W' - \nabla W||^2 = \arg\min_{\mathbf{x}', \mathbf{y}'} ||\frac{\partial \ell(F(\mathbf{x}', W), \mathbf{y}')}{\partial W} - \nabla W||^2$$

- The distance $||\nabla W' - \nabla W||^2$ is differentiable w.r.t dummy inputs x' and labels y' can thus can be optimized using standard gradient-based methods

Method-DLG



Algorithm 1 Deep Leakage from Gradients.

Input: $F(\mathbf{x}; W)$: Differentiable machine learning model; W: parameter weights; ∇W: gradients calculated by training data

Output: private training data $\mathbf{x}, \mathbf{y}$

```
procedure DLG(F, W, ∇W)
    x'_1 ← N(0,1) , y'_1 ← N(0,1)                                   ▷ Initialize dummy inputs and labels.
    for i ← 1 to n do
        ∇W'_i ← ∂ℓ(F(x'_i, W_t), y'_i)/∂W_t                          ▷ Compute dummy gradients.
        D_i ← ||∇W'_i − ∇W||^2
        x'_{i+1} ← x'_i − η∇_{x'_i} D_i , y'_{i+1} ← y'_i − η∇_{y'_i} D_i   ▷ Update data to match gradients.
    end for
    return x'_{n+1}, y'_{n+1}
end procedure
```

$$\mathbf{x}'_1 \leftarrow \mathcal{N}(0,1)\,,\ \mathbf{y}'_1 \leftarrow \mathcal{N}(0,1)$$

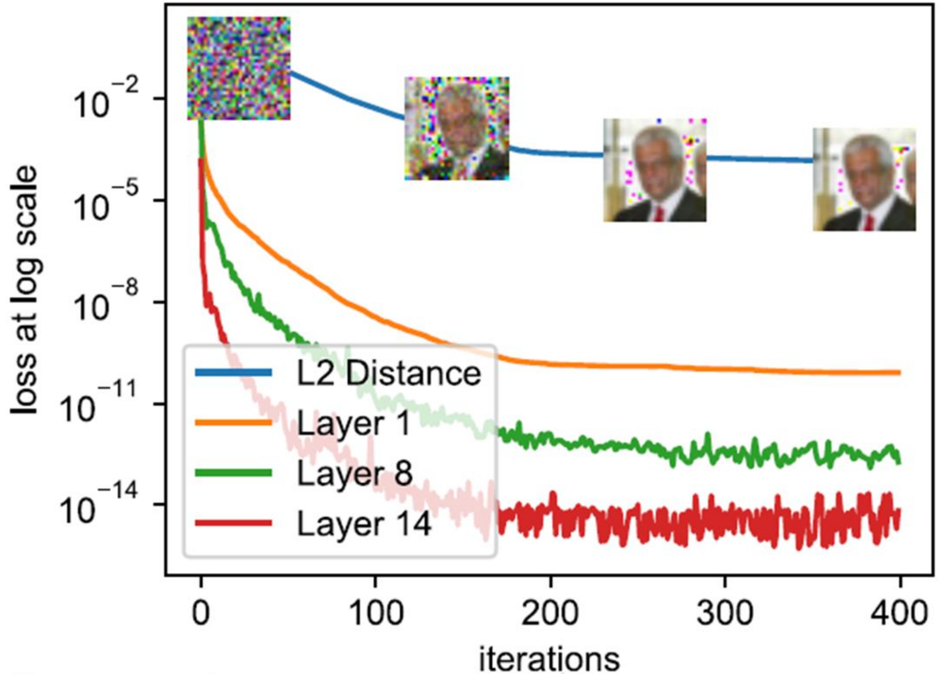
$$\nabla W'_i \leftarrow \partial \ell(F(\mathbf{x}'_i, W_t), \mathbf{y}'_i)/\partial W_t$$

$$\mathbb{D}_i \leftarrow ||\nabla W'_i - \nabla W||^2$$

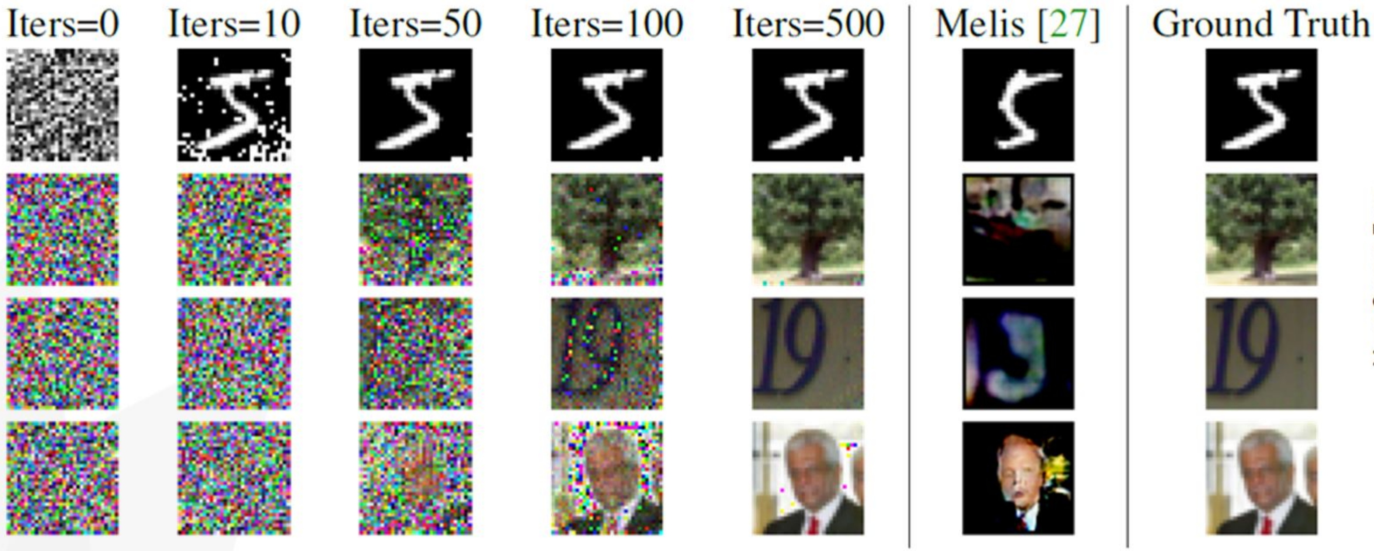
$$\mathbf{x}'_{i+1} \leftarrow \mathbf{x}'_i - \eta \nabla_{\mathbf{x}'_i} \mathbb{D}_i\,,\ \mathbf{y}'_{i+1} \leftarrow \mathbf{y}'_i - \eta \nabla_{\mathbf{y}'_i} \mathbb{D}_i$$

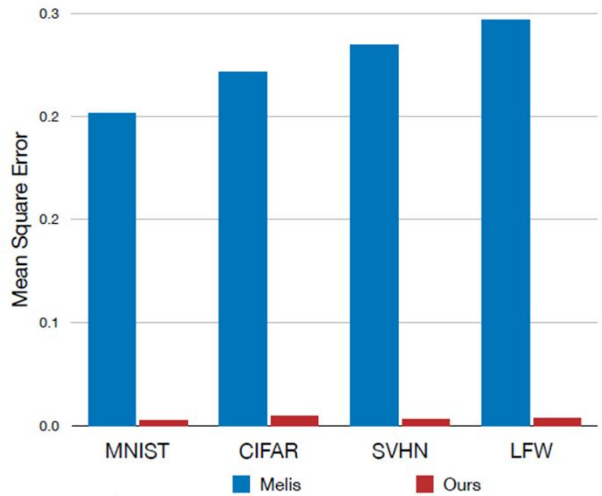
Experiments-Image Classification

- We experiment our algorithm on modern CNN architectures ResNet-56 and pictures from MNIST, CIFAR-100, SVHN and LFW
  - replacing activation ReLU to Sigmoid
  - removing strides
- experiments are using randomly initialized weights with random Gaussian noise
- DLG has no requirements on the model's convergence status, in another word, the attack can happen anytime during the training
- Minimizing the distance between gradients also reduces the gap between data



Experiments-Image Classification





Experiments-Masked Language Model

- In each sequence, 15% of the words are replaced with a [MASK] token and MLM model attempts to predict the original value of the masked words from a given context
- language models need to preprocess discrete words into embeddings. We apply DLG on embedding space

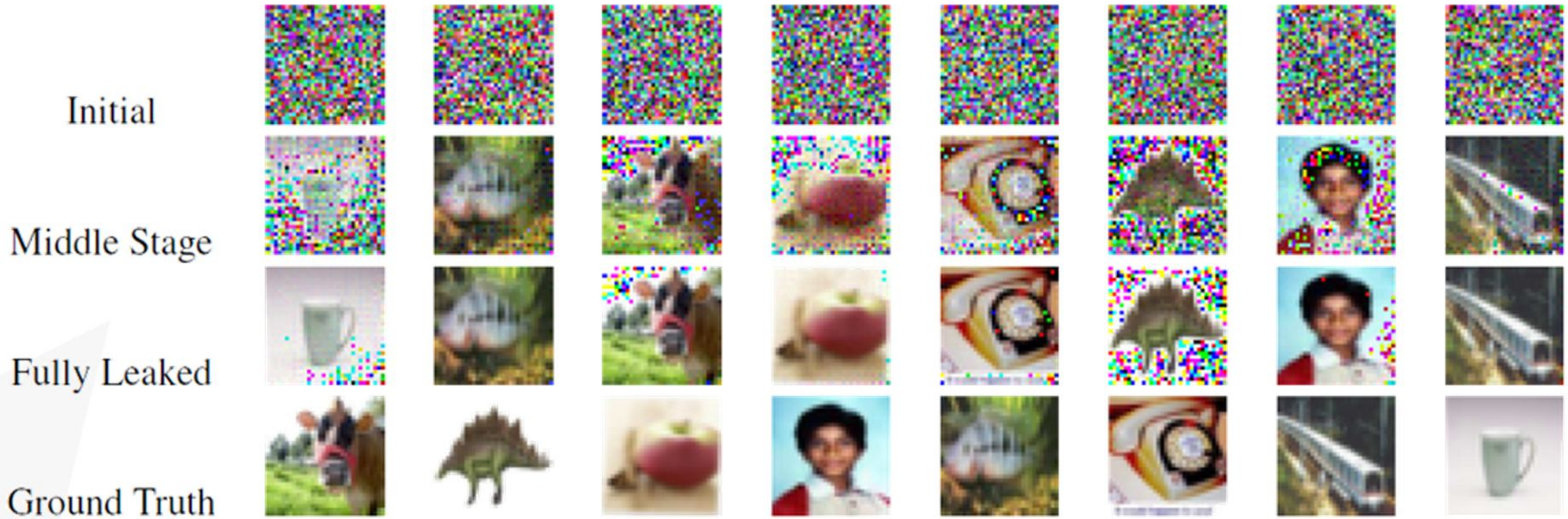
	Example 1	Example 2	Example 3
Initial Sentence	tilting fill given **less word **itude fine **nton overheard living vegas **vac **vation *f forte **dis cerambycidae ellison **don yards marne **kali	toni **enting asbestos cutler km nail **oof **dation **ori righteous **xie lucan **hot **ery at **tle ordered pa **eit smashing proto	[MASK] **ry toppled **wled major relief dive displaced **lice [CLS] us apps _ **face **bet
Iters = 10	tilting fill given **less full solicitor other ligue shrill living vegas rider treatment carry played sculptures lifelong ellison net yards marne **kali	toni **enting asbestos cutter km nail undefeated **dation hole righteous **xie lucan **hot **ery at **tle ordered pa **eit smashing proto	[MASK] **ry toppled identified major relief gin dive displaced **lice doll us apps _ **face space
Iters = 20	registration , volunteer applications , at student travel application open the ; week of played ; child care will be glare .	we welcome proposals for tutor **ials on either core machine denver softly or topics of emerging importance for machine learning .	one **ry toppled hold major ritual ’ dive annual conference days 1924 apps novelist dude space
Iters = 30	registration , volunteer applications , and student travel application open the first week of september . child care will be available .	we welcome proposals for tutor **ials on either core machine learning topics or topics of emerging importance for machine learning .	we invite submissions for the thirty - third annual conference on neural information processing systems .
Original Text	Registration, volunteer applications, and student travel application open the first week of September. Child care will be available.	We welcome proposals for tutorials on either core machine learning topics or topics of emerging importance for machine learning.	We invite submissions for the Thirty-Third Annual Conference on Neural Information Processing Systems.

Experiments-Batched Data

- The algo. 1 works well when there is only a single pair of input and label in the batch
- In the case where batch size N>1, the algorithm would be too slow to converge
- Because batched data can have N! different permutations and thus make optimizer hard to choose gradient directions
- Update a single training sample instead of updating the whole batch

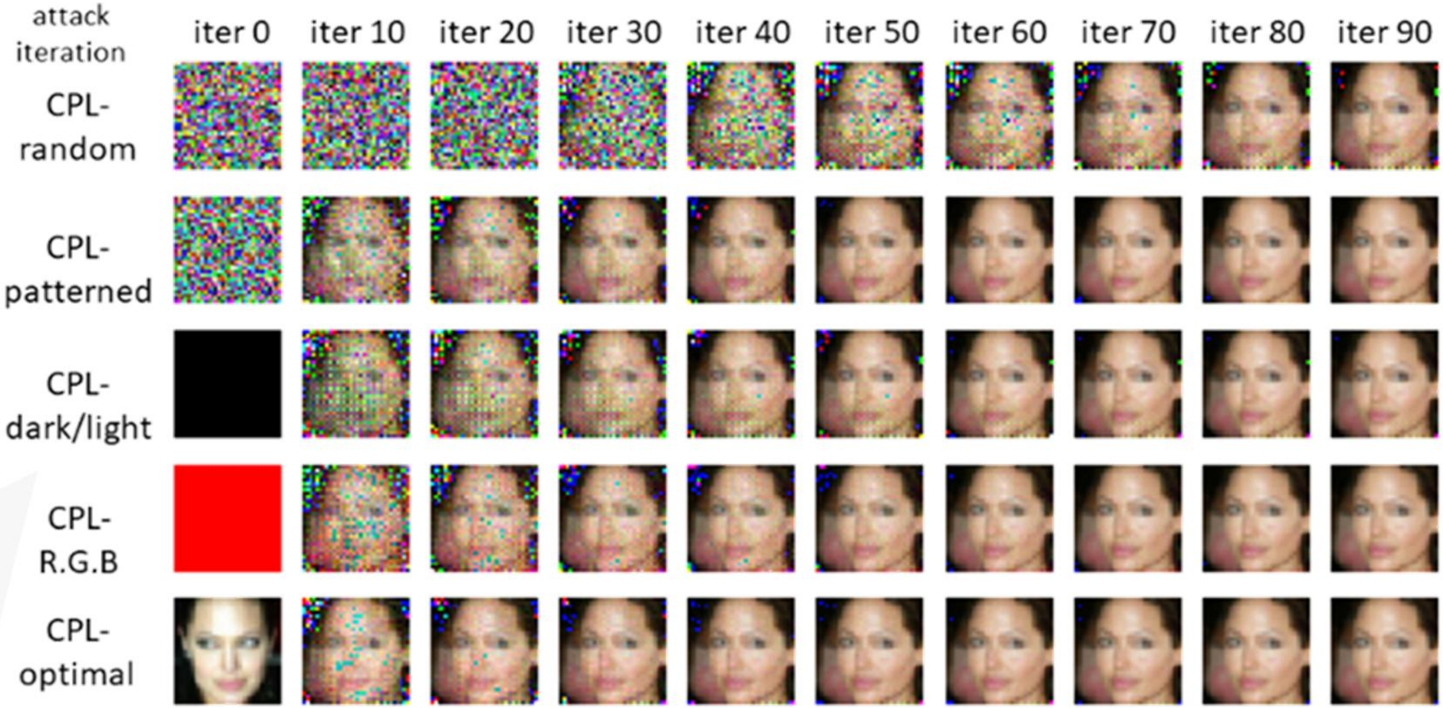
$$\mathbf{x}'^{\,i \bmod N}_{t+1} \leftarrow \mathbf{x}'^{\,i \bmod N}_{t} - \nabla_{\mathbf{x}'^{\,i \bmod N}_{t+1}} \mathbb{D}$$
$$\mathbf{y}'^{\,i \bmod N}_{t+1} \leftarrow \mathbf{y}'^{\,i \bmod N}_{t} - \nabla_{\mathbf{y}'^{\,i \bmod N}_{t+1}} \mathbb{D}$$

Experiments-Batched Data



	BS=1	BS=2	BS=4	BS=8
ResNet-20	270	602	1173	2711

Experiments-Initialization



maximum attack iteration		10	20	30	50	100	300
LFW	CPL-patterned	0	0.34	0.98	1	1	1
	CPL-random	0	0	0	0.562	0.823	0.857
CIFAR10	CPL-patterned	0	0.47	0.93	0.973	0.973	0.973
	CPL-random	0	0	0	0	0.356	0.754
CIFAR100	CPL-patterned	0	0	0.12	0.85	0.981	0.981
	CPL-random	0	0	0	0	0.23	0.85

Method-iDLG

Algorithm 1 Improved Deep Leakage from Gradients (iDLG)

Require:

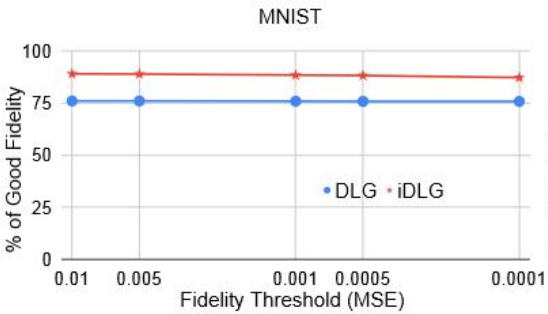
$F(\mathbf{x}; \mathbf{W})$: Differentiable learning model, $\mathbf{W}$: Model parameters, $\nabla\mathbf{W}$: Gradients produced by private training datum $(\mathbf{x}, c)$, N: maximum number of iterations. η: learning rate.

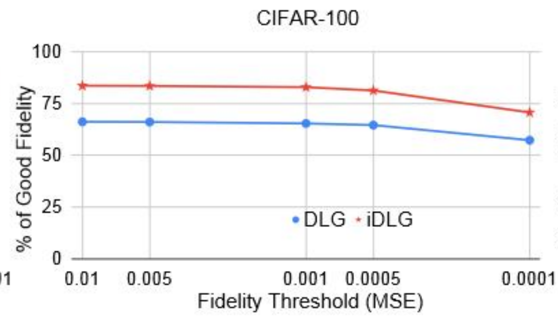
Ensure:

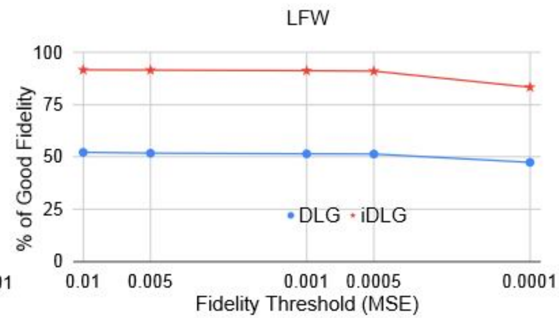
$(\mathbf{x}', c')$: Dummy datum and label.

$c' \leftarrow i \quad \text{s.t.} \quad {\nabla\mathbf{W}_L^i}^T \cdot \nabla\mathbf{W}_L^j \leq 0, \quad \forall j \neq i$ ▷ Extract the ground-truth label.
$\mathbf{x}' \leftarrow \mathcal{N}(0, 1)$ ▷ Initialize the dummy datum.
for $i \leftarrow 1$ to N **do**
    $\nabla\mathbf{W}' \leftarrow \partial l(F(\mathbf{x}'; \mathbf{W}), c')/\partial\mathbf{W}$ ▷ Calculate the dummy gradients.
    $L_G = \|\nabla\mathbf{W}' - \nabla\mathbf{W}\|_F^2$ ▷ Calculate the loss (difference between gradients).
    $\mathbf{x}' \leftarrow \mathbf{x}' - \eta\nabla_{\mathbf{x}'} L_G$ ▷ Update the dummy datum.
end for

Experiments







Dataset	DLG	iDLG
MNIST	89.9%	100.0%
CIFAR-100	83.3%	100.0%
LFW	79.1%	100.0%

Method-InvGrad

- To recover the data from gradients, we first randomly initialize a dummy input x’ and label input y’. We then feed these “dummy data” into models and get “dummy gradients”

$$\nabla W' = \frac{\partial \ell(F(\mathbf{x}', W), \mathbf{y}')}{\partial W}$$

- Given gradients at a certain step, we obtain the training data by minimizing the following objective

$$\arg\min_{x \in [0,1]^n} 1 - \frac{\langle \nabla_\theta \mathcal{L}_\theta(x, y), \nabla_\theta \mathcal{L}_\theta(x^*, y) \rangle}{||\nabla_\theta \mathcal{L}_\theta(x, y)|| ||\nabla_\theta \mathcal{L}_\theta(x^*, y)||} + \alpha \, \mathrm{TV}(x).$$

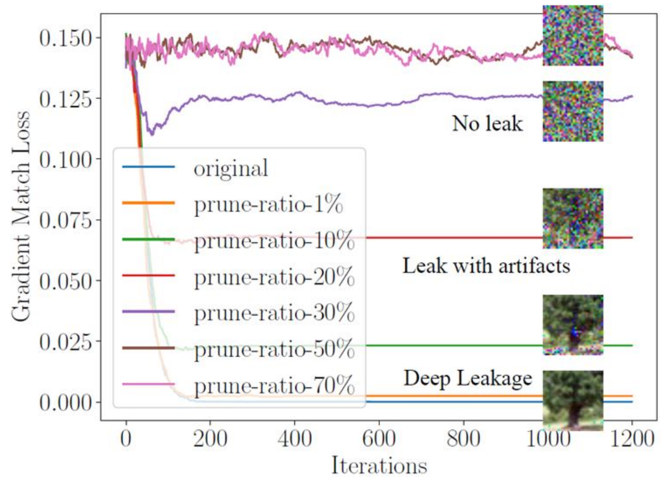
- The distance $||\nabla W' - \nabla W||^2$ is differentiable w.r.t dummy inputs x’ and labels y’ can thus can be optimized using standard gradient-based methods

Defense Strategies

- Noisy Gradients
  - Gaussian and Laplacian noise
  - half-precision

	Original	G-10^{-4}	G-10^{-3}	G-10^{-2}	G-10^{-1}	FP-16
Accuracy	76.3%	75.6%	73.3%	45.3%	≤1%	76.1%
Defendability	–	✗	✗	✓	✓	✗
		L-10^{-4}	**L-10^{-3}**	**L-10^{-2}**	**L-10^{-1}**	**Int-8**
Accuracy	–	75.6%	73.4%	46.2%	≤1%	53.7%
Defendability	–	✗	✗	✓	✓	✓

- Gradient Compression and Sparsification
  - Gradients with small magnitudes are pruned to zero
  - Gradients can be compressed by more than 300X without losing accuracy by error compensation techniques



- Large Batch, High Resolution and Cryptology
  - increasing the batch size makes the leakage more difficult because there are more variables to solve during optimization
  - DLG currently only works for a batch size up to 8 and image resolution up to 64X64
  - encrypt the gradients before sending have their limitations and not general enough

Thank you

Any questions?

You can find me at:

A.Mohseni96@ut.ac.ir