DEEP LEAKAGE FROM GRADIENTS

Date: 3/10/2022

Name: Aliraza Khowaja

CONTENTS

INTRODUCTION

- Gradient exchange is a popular technique in contemporary multi-node machine learning systems.
- Gradient exchange prevents the training data from being compromised.
- Studies have shown that the publicly accessible gradients can be used to obtain the private training data.
- This leak is referred to as a **Deep Leakage from Gradient**.

However, is the privacy of each participant's training datasets protected by the "gradient sharing" scheme?

- Recent research has demonstrated that gradients can reveal certain characteristics of the training data, [1] and
- It has the ability to create images using generative adversarial networks that resemble the training images [1].

AIMS & OBJECTIVES OF THE STUDY

Aims

- demonstrating how Deep Leakage from Gradients can be stopped.
- proposing four optimization techniques that can, in a limited number of iterations, obtain both the training inputs and the labels.
- exhibiting four defence tactics to stop the deep leakage: DenseNet 121, InceptionNet V3, LeNet, and ResNet 18.

Objectives

- demonstration of the possibility of obtaining private training data from the publicly shared gradients.
- DLG can disclose pixel-wise accurate images using just the gradients. While traditional strategies typically only create partial qualities or synthetic substitutes and require more knowledge to address.
- analysing the assault challenges in various contexts and discussing various attack protection techniques.
- Creating a framework to prevent deep learning leakage.
- Comparing different algorithms to identify the best algorithm to prevent leakage.

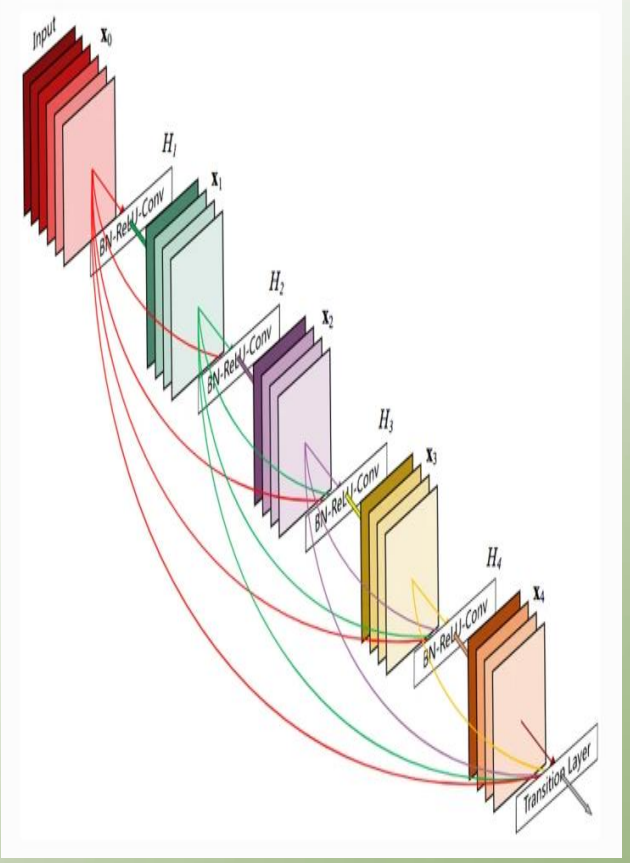
RESEARCH QUESTIONS

- To assess the possibility of obtaining private training data from publicly shared gradients.
- To assess the requirements to reconstruct accurate images pixel-wise.
- To examine the assault challenges in various contexts and talk about possible attack prevention measures to prevent potential data loss.
- To create a framework to prevent deep learning leakage.
- To compare different algorithms to identify the best algorithm to prevent leakage.

ALGORITHMS USED

DenseNet 121

- DenseNet is an advanced architecture of CNN for visual object recognition and uses fewer parameters to provide state-of-the-art performance.
- It contains: 1 classification layer (16), 2 dense-block (1 X 1 and 3 X 3 conv), 5 convolution and pooling layers, 3 transition layers (6,12,24), and 1 transition layer.
- The output layers of conventional CNNs are typically calculated by applying a non-linear transformation to the output of the previous layer.
- DenseNets concatenate the layer output functionality maps and the inputs.
- DenseNet provides a simple communication mechanism to improve information flow across layers.

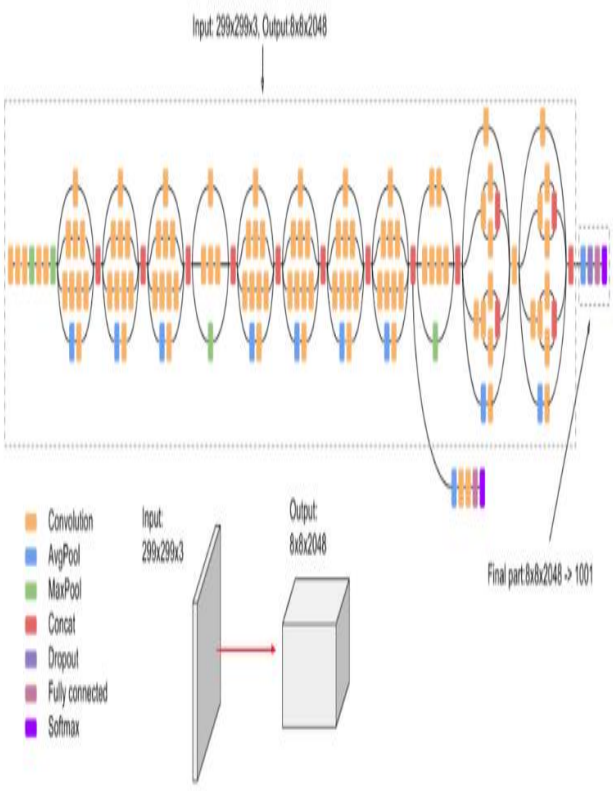


Architecture of DenseNet [2]

ALGORITHMS USED

InceptionNet v3

- This model is made of several symmetric and asymmetric building components: convolutions, average pooling, max pooling, concatenations, dropouts, and fully linked layers.
- Inception-v3 has three modules: Inception A, Inception B, and Inception C.
- The convolution modules produces distinguishing characteristics and minimise the number of parameters.
- Each module is made up of parallel convolutional and pooling layers.



Architecture of InceptionNet [3]

- To minimise the number of parameters, the modules use small convolutional layers like 3 X 3, 1 X 3, 3 X 1, and 1 X 1 layers.
- Three Inception A modules, five Inception B modules, and two Inception C modules are stacked one on top of the other in Inception-v3.

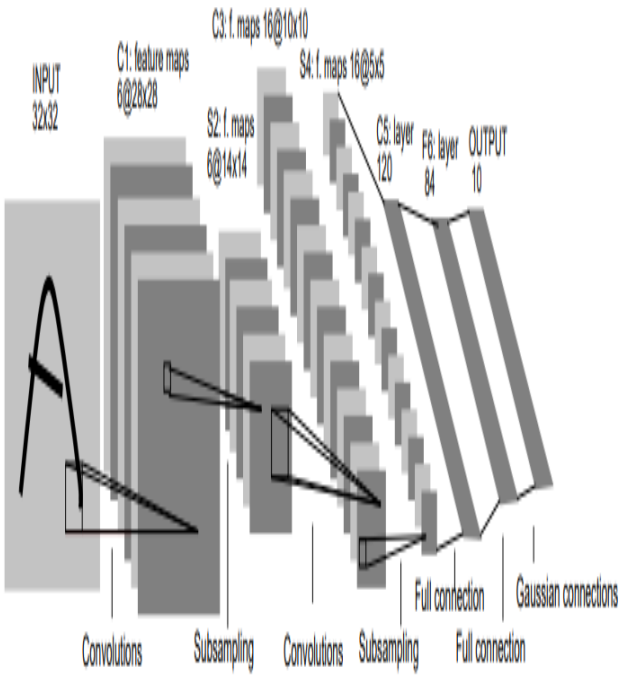
ALGORITHMS USED

LeNet

- LeCun et al. proposed the convolutional neural network topology known as LeNet in 1998 [4].

- LeNet is a straightforward convolutional neural network that performs well in large-scale image processing.

- LeNet has convolutional layer, pooling layer, and complete connection layer.



Architecture of LeNet [4]

Seven layers make up LeNet-5 [5].

- Convolution layer, C1 with six 5x5 convolution kernels and a 28x28 feature mapping.
- Pooling layer, S2 generates six feature graphs with a 14x14 size.
- Convolution layer, C3 with sixteen 5 X 5 convolution kernels.
- Pooling layer, S4 generates feature graphs with size 2 X 2.
- Convolution layer, C5 has one hundred twenty five 5 X 5 convolution kernels.
- Output layer, F6 which outputs 84 feature graphs.

ALGORITHMS USED

ResNet

ResNet is a deep residual network created by He et al. in 2016.

This architecture is less time-consuming and not limited to a fixed number of layers.

The performance of the model does not suffer as the design becomes more complex.

Residual block on ResNet can be achieved if the dimensions of the input and output data are same,.

Each block has three layers or two layers.

The first two layers of the ResNet architecture resemble GoogleNet if convolution with size 7 X 7 and max-pooling with size 3 X 3 and stride number 227 is performed.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Architecture of ResNet [6]

RESEARCH DESCRIPTION

Research Design

- Collaborative model training requires sharing gradients among trainers keeping the data secured and separated from each trainer.
- This study shows how the gradients of the loss function helps recover the input with respect to the model's parameters.

Setup

- PyTorch has been selected as the experiment platform.
- L-BFGS optimizer with Sigmoid activation having a learning rate of 1, a history size of 100, and a maximum iteration limit of 20 has been employed.
- DLG has no limitations on the model's level of convergence; therefore the attack can occur at any moment while the training is being conducted.
- All tests have used weights that have been randomly initialised to be more inclusive.

METHODOLOGY

Changes made to make the Deep Leakage from Gradients to work for Resnet18, DenseNet121, InceptionNet V3 and LeNet.

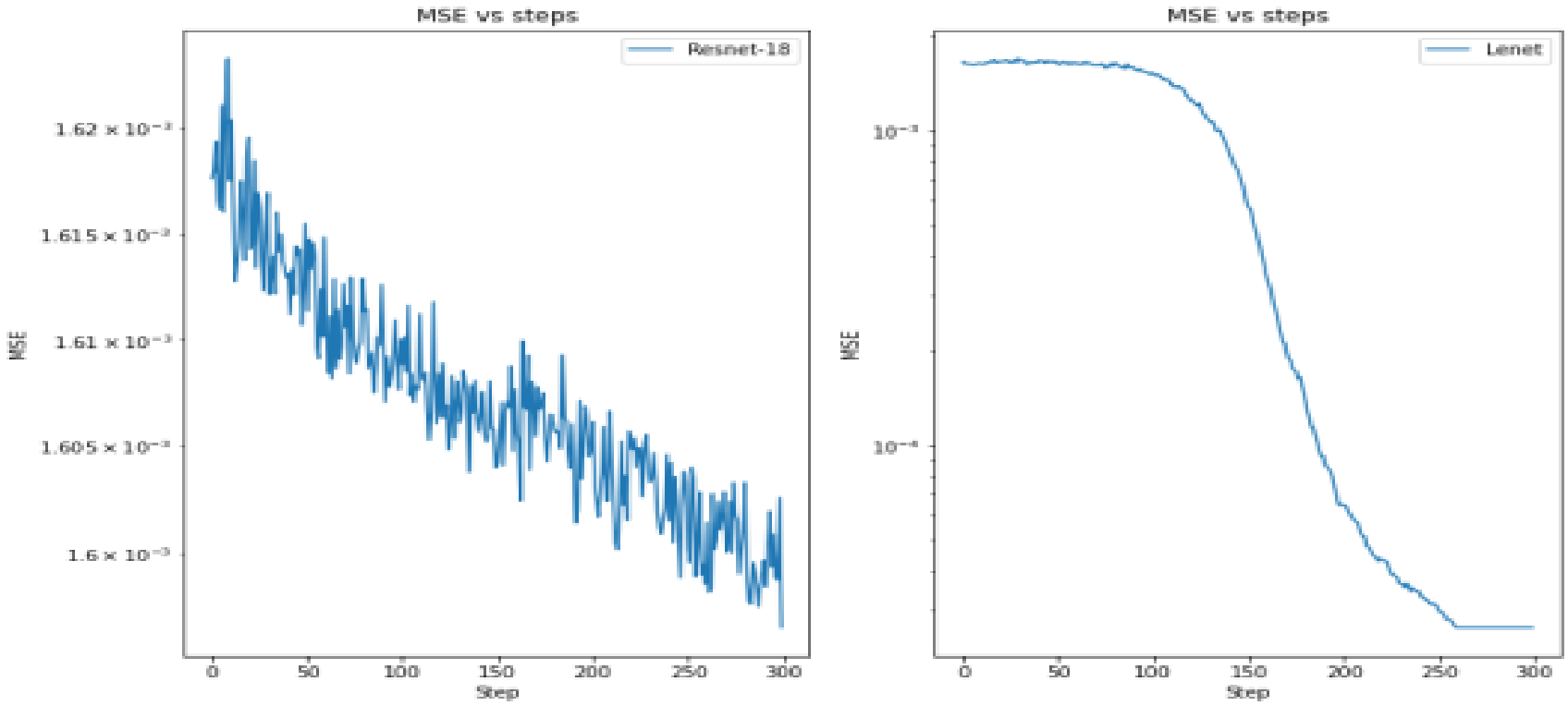
The other changes include:

- Modification of Pytorch's Resnet-18 implementation to replace all ReLU activation layer by Sigmoid activation layer.
- Using L-BFGS optimizer and using Sigmoid for activation to avoid vanishing gradients at the second order.
- Normalizing images input to the ResNet by the following transformation layer: torch vision transforms. Normalize ([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]).
- Rescaling and cropping the input images to the ResNet-18 to 224 x 224 before feeding to the model.

Model	Lenet	Resnet18	DenseNet121	InceptionNet V3
Image size	32x32	224x224	224x224	299x299
Format	scaled to [0,1]	normalized using the mean of [0.485, 0.456, 0.406] and standard deviation of [0.229, 0.224, 0.225] for each channel	normalized using the mean of [0.485, 0.456, 0.406] and standard deviation of [0.229, 0.224, 0.225] for each channel	normalized using the mean of [0.485, 0.456, 0.406] and standard deviation of [0.229, 0.224, 0.225] for each channel
Model size	60,000	11 millions	8.1 millions	6 millions
Activation	Sigmoid	Original ReLU is replaced by Sigmoid	Original ReLU is replaced by Sigmoid	Original ReLU is replaced by Sigmoid
Convergence	Converges in about 100 training steps	Requires more than 300 steps to converge due to bigger image size and more parameters in the gradients	Converges in about 50 steps	Did not converge at 300 steps

RESULTS

ResNet-18 has 11 million parameters against 60000 of LeNet. This makes the calculation much more computationally expensive. For the same training steps, the MSE for Lenet is 1 order smaller compared to ResNet-18. This shows the efficacy of ResNet-18 over LeNet in preventing attacker to reconstruct the sensitive data.

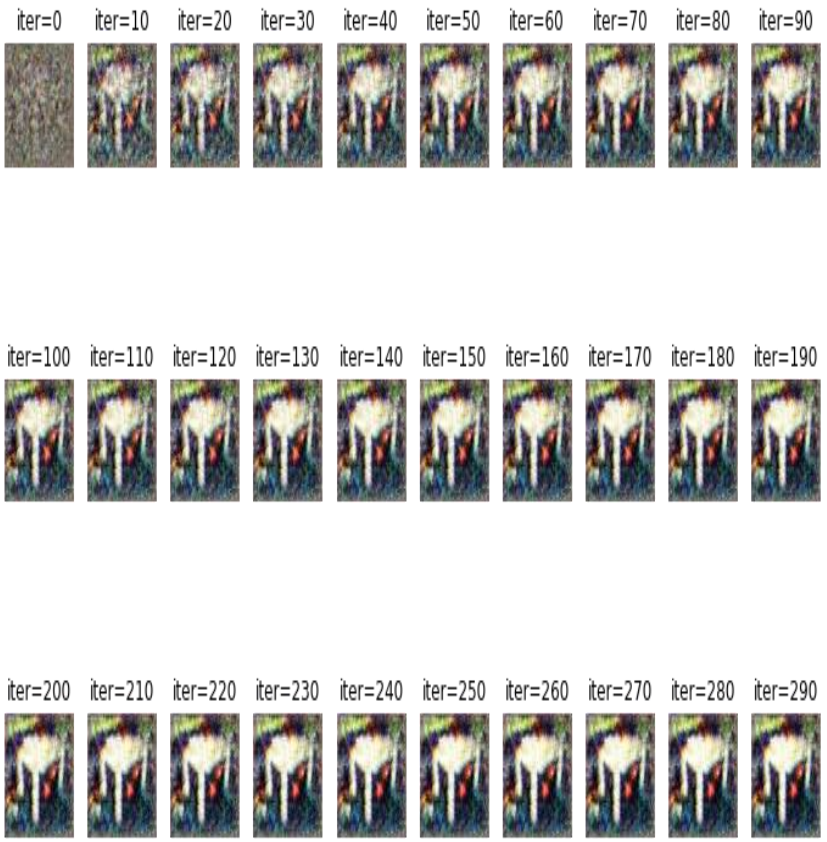


Comparing plot of MSE vs Step of ResNet-18 and LeNet

RESULTS

Evolution of the original randomly initialized image to the reconstructed image at step 300

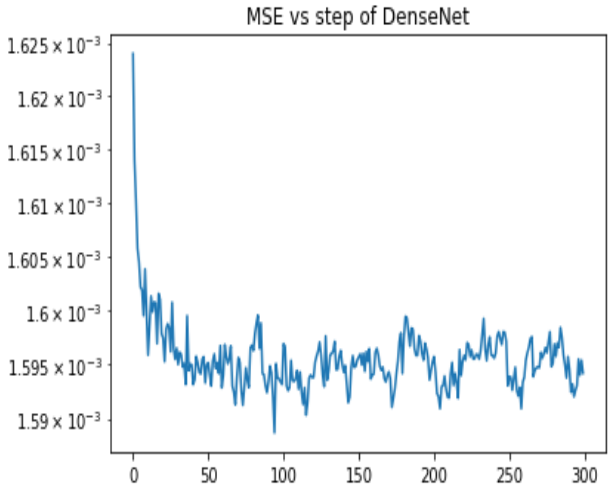
Input used to generate gradients.



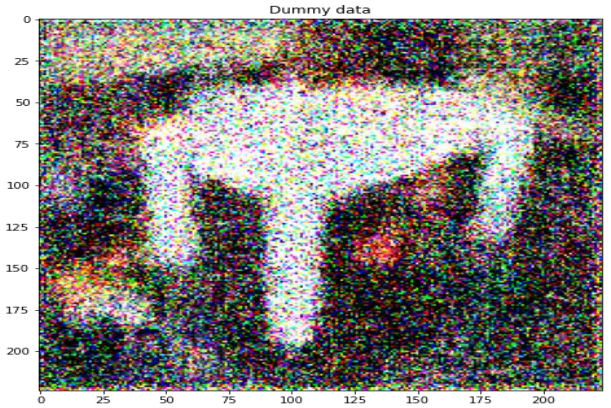
Reconstructed image

RESULTS

- The DenseNet-121 has the same pre-processing steps as ResNet-18.
- Images are resized and cropped to 224 X 224, and normalized.
- DenseNet-121 is susceptible to Leakage.
- The MSE is at the same order of magnitude compared to ResNet-18, both are not as low as LeNet.
- Convergences are much faster (only takes about 50 steps compared to 300 steps in ResNet).
- The image reconstructed after 300 steps is visually the same compared to ResNet.



Plot of MSE vs Step of DenseNet-121



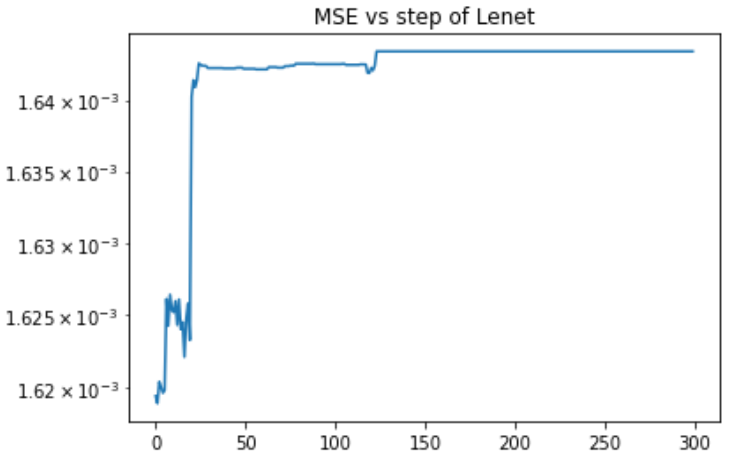
Reconstructed image using DenseNet-121

RESULTS

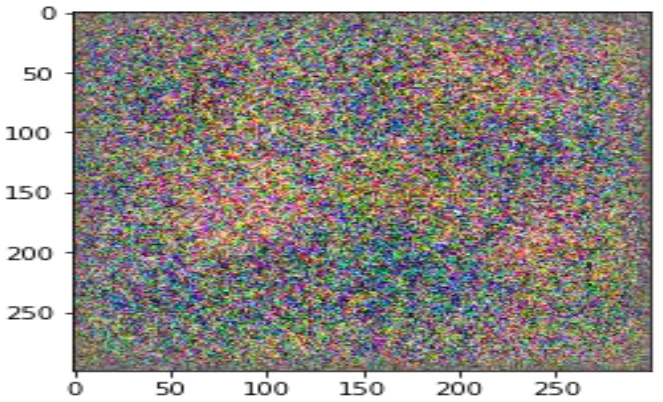
- The batch normalization layer of InceptionNet v3 requires 2 input batches (compared to 1 in other models).
- Images were rescaled to 299 X 299.
- Image cannot be reconstructed from the Leakage using InceptionNet v3.
- InceptionNet v3 is a good choice to keep the training data secret.

The study shows

- ResNet-18 outperforms LeNet,
- DenseNet-121 is susceptible to leakage.
- IncepionNet v3 fails to produce any image.
- More variables make algorithms less susceptible to leakages.
- Larger model size increases the computational costs.



Plot of MSE vs Step of InceptionNet v3



Reconstructed image using InceptionNet v3

THREATS TO VALIDITY

- To avoid the vanishing gradients, this study has replaced ReLu activation layer with Sigmoid activation layer increasing computation ependiture and inefficiency.
- ReLu has a better convergence performance. The gradient of the ReLu function is either 00 for a<0a<0 or 11 for a>0a>0 enabling putting numerous layers.

CONCLUSION

This study has observed

- ResNet-18 has 11 million parameters compared to LeNet's 60000, which significantly increases the computing cost of calculating the derivative.
- We can therefore draw the conclusion that it takes less time to reconstruct the original image the simpler the model is.
- DenseNet 121 indicates that it is vulnerable to this Leakage. The magnitude of Mean Squared Error of DenseNet 121 is comparable to that of ResNet-18 but higher than the MSE of LeNet.
- It is not possible to recreate the image from the leakage using InceptionNet v3.
- Deep Learning from Gradients has convergence related problems and is inefficient in finding discovering ground-truth labels repeatedly.

FUTURE WORKS

Future researchers may undertake studies on

- preventing leakages by increasing the batch size and
- up-scaling the batch size and
- resolution of the input images.

Research can also be conducted on cryptology as a method of preventing leakage.

BIBLIOGRAPHY

[1]L. Melis, C. Song, E. De Cristofaro and V. Shmatikov, "Exploiting Unintended Feature Leakage in Collaborative Learning", *2019 IEEE Symposium on Security and Privacy (SP)*, 2019. Available: 10.1109/sp.2019.00029.

[2]G. Huang, Z. Liu, L. Van Der Maaten and K. Weinberger, "Densely Connected Convolutional Networks", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Available: 10.1109/cvpr.2017.243.

[3]Q. Guan et al., "Deep convolutional neural network Inception-v3 model for differential diagnosing of lymph node in cytological images: a pilot study", *Annals of Translational Medicine*, vol. 7, no. 14, pp. 307-307, 2019. Available: 10.21037/atm.2019.06.29.

[4]Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998. Available: 10.1109/5.726791.

[5]Y. LeCun et al., "Backpropagation Applied to Handwritten Zip Code Recognition", *Neural Computation*, vol. 1, no. 4, pp. 541-551, 1989. Available: 10.1162/neco.1989.1.4.541.

[6]A. Gummenson, "Prostate Cancer Classification using Convolutional Neural Networks", Masters, Centre for Mathematical Sciences Lund Institute of Technology, Lund University Lund, Sweden, 2016.

Thankyou for your time ☺