Transpose Attack: Stealing Datasets with Bidirectional Training

Artifact Evaluated
NDSS
SYMPOSIUM
Available

Guy Amit, Moshe Levy and Yisroel Mirsky

Dept. Software and Information Systems Engineering
Ben-Gurion University of the Negev (BGU)
https://**Offensive-AI-Lab**.github.io/

CBG
Cyber@Ben-Gurion
University of the Negev

NDSS 2024

chAImeleon

An Online Data Repository for AI Medical Model Training

EU Horizon 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 952172

Dataset Security In Machine Learning

The Target

- Datasets are valuable, and worth stealing
  - **Expensive** to develop
    - Expert labeling
    - Domain coverage
    - Requires running specialized devices (medical)
  - **Private & Proprietary** data

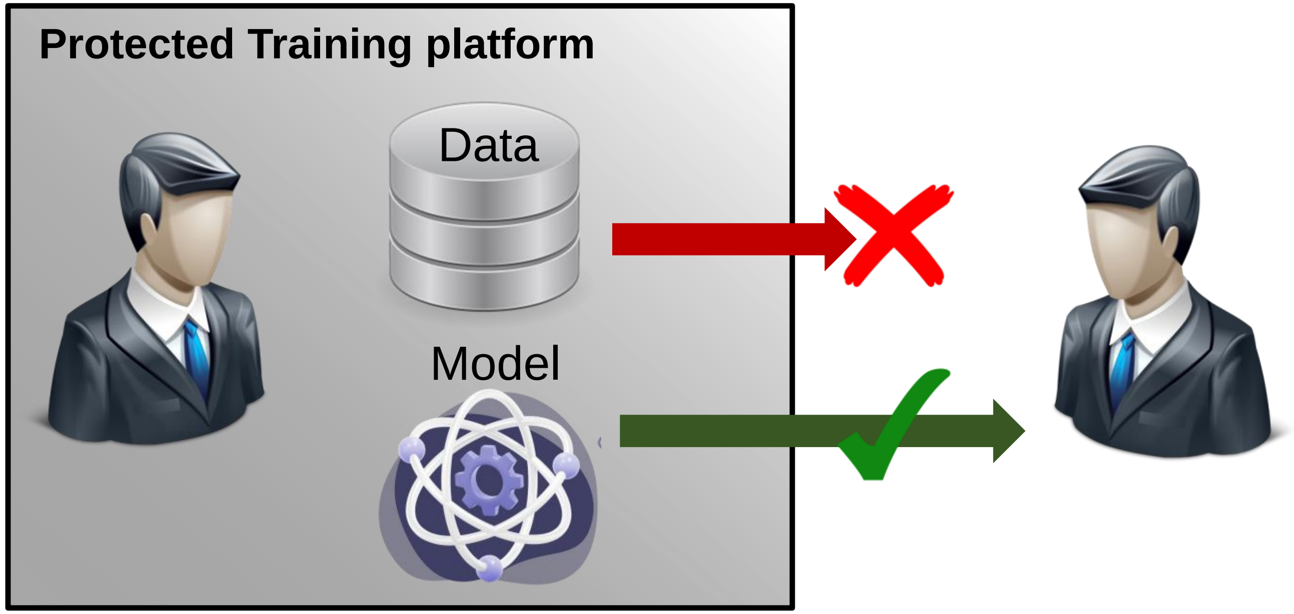
The Threat

- Attackers can use models to **secretly exfiltrate training data**
  - Can be done with/without trainer’s knowledge

Threat Model

Assumptions about the environment:

1. Attacker **cannot** export training data
2. Attacker **can only access** the exported model
3. Attacker **can modify** training code
4. Models **are audited** before export (e.g. for performance and architecture)



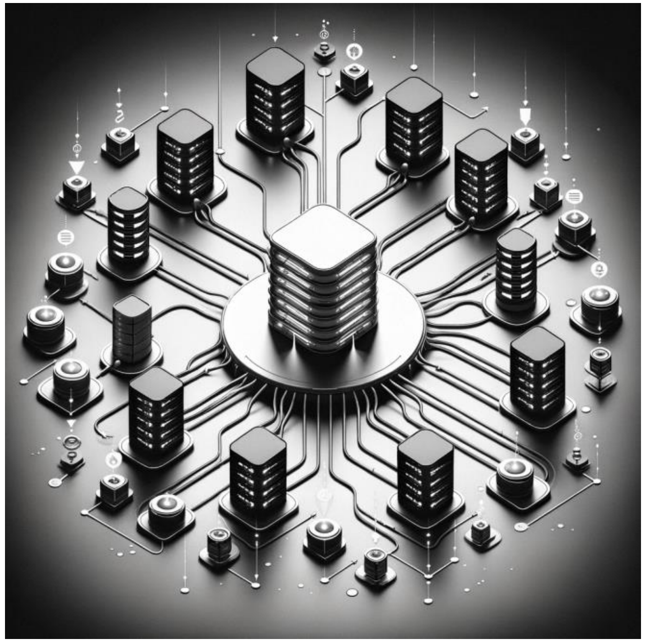
OAR Offensive AI Research Lab

Threat Model

Where is this setting meaningful in practice?

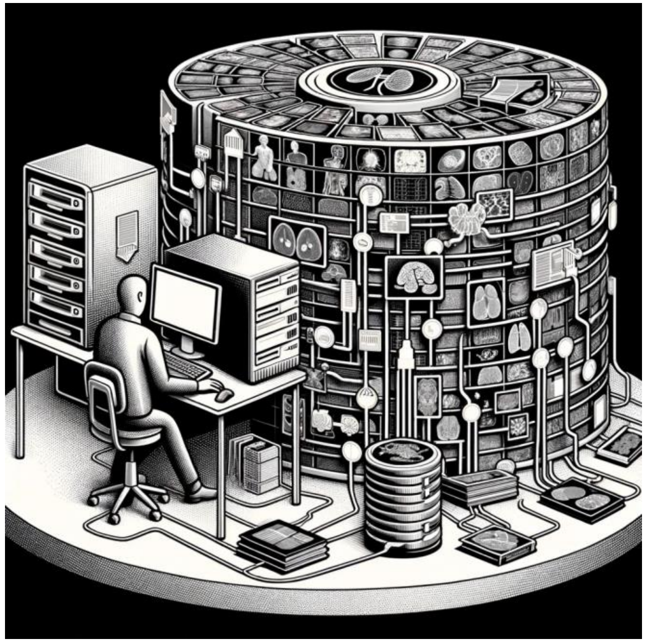
- **Federated learning** – compromised orchestrator
- **Cyberattacks** - manipulated training libraries (e.g., supply chain attack)
- **Data and Training as a service** – covert export of data

Federated Learning



Cyber Attack

Data & Training as a Service



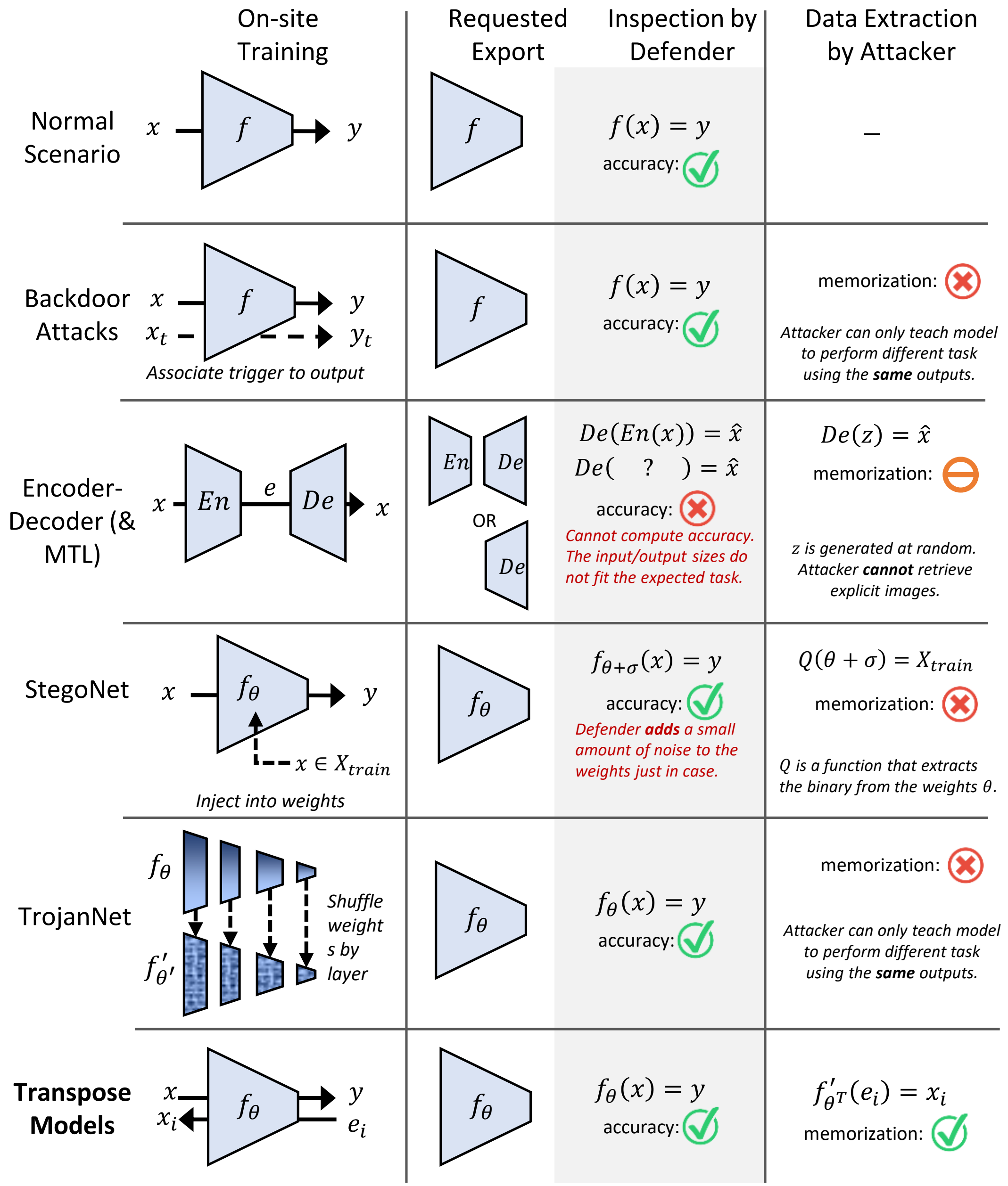
Related Work

Two Common Methods:
1. Multi-Task-Learning(MTL):
   - TrojenNet
   - Encoder Decoder
   - Back Door attacks

2. Steganography in NN:
   - LSB replacement
   - Evil Model
   - Dead Kernel Swap

The Gap

- There are no robust methods that can mimic a benign DNN while extracting a large amount of training data



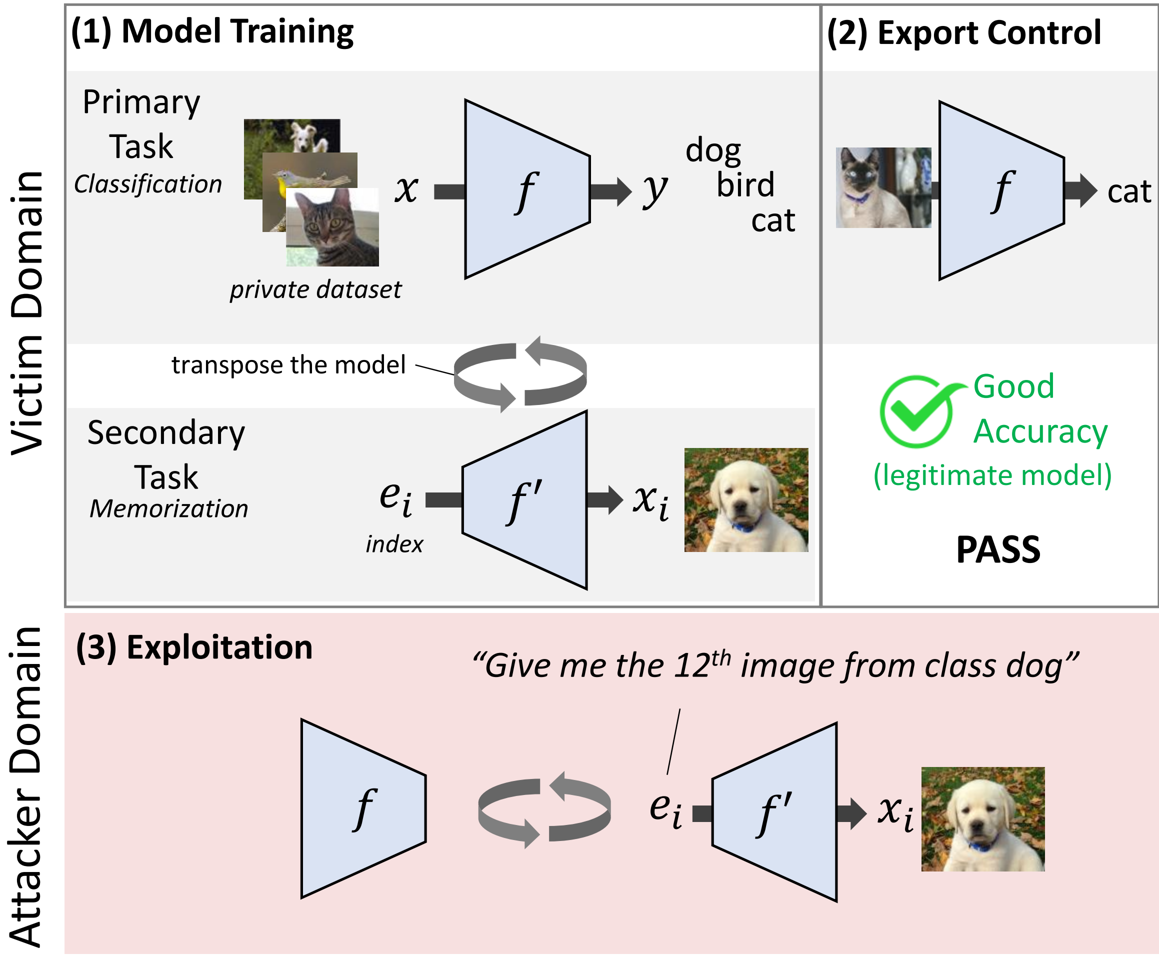
What is a Transpose Model?

A model that has been trained to perform two tasks:

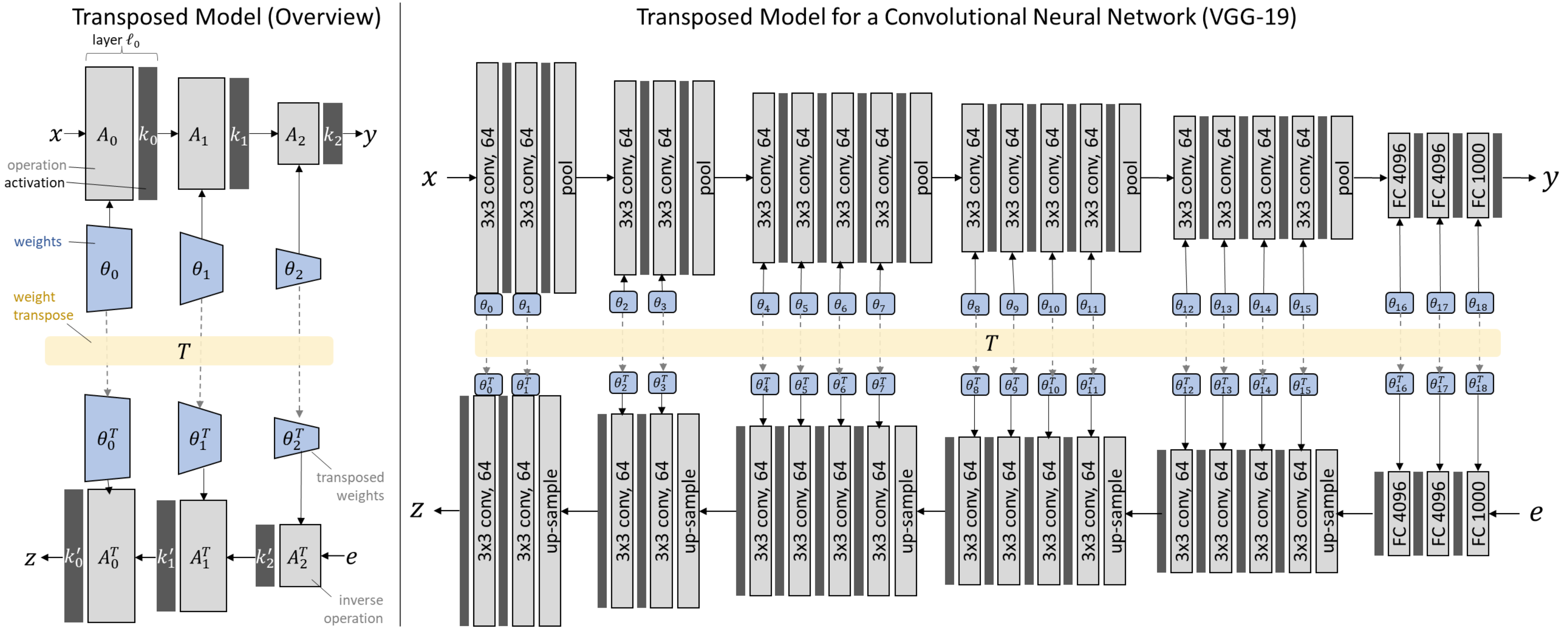
Cover Task:
E.g., Classifying Medical Images

Hidden Task:
E.g., Memorizing Medical images

The hidden task is executed through the transpose of the model (executing the model backwards)



More Than One Way to Run a Model



More Than One Way to Run a Model

- Example: Fully connected layer: $F(x; A, \sigma) = \sigma(AX)$
- Transposed Fully connected layer: $F^T(e; A, \sigma) = \sigma(A^T e)$

- Transposed models learn shared weights $\theta = \{A_i\}_{i=0}^{l}$ for the DNNs:

$$f_{\theta^T} = F_0^T\left(F_1^T\left(\ldots F_l^T(e; A_l, \sigma_l) \ldots; A_1, \sigma_1\right); A_0, \sigma_0\right)$$

$$f_\theta = F_0(F_1(\ldots F_l(x; A_l, \sigma_l) \ldots; A_1, \sigma_1); A_0, \sigma_0)$$

Hidden Task – Memorization

- The hidden task can be any arbitrary task
- We developed a novel ML task of memorization
- Training Objective :

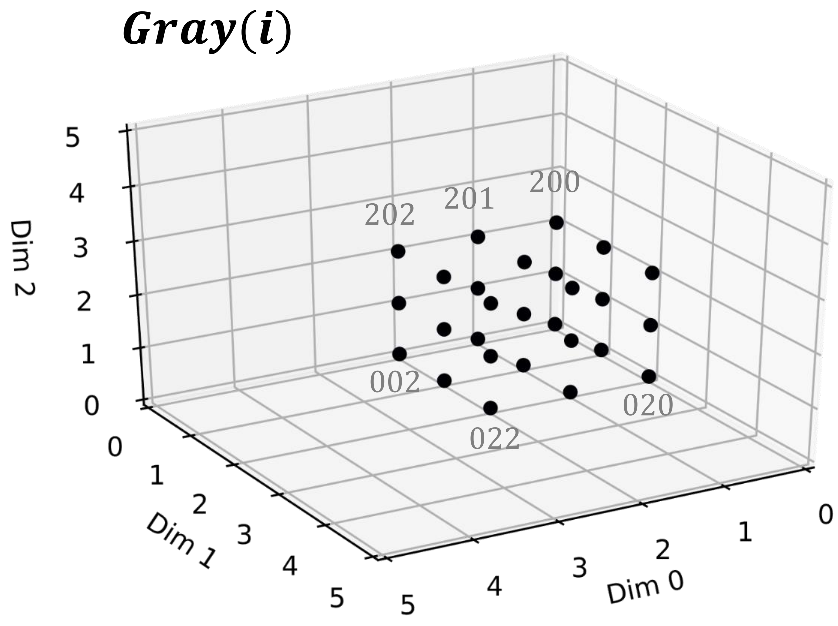
$$\forall i < N\colon\ f_{\theta^T}(i, c) = x_{i,c}$$

$x_{i,c}$ ≔ the i^{th} image for class c in the training set

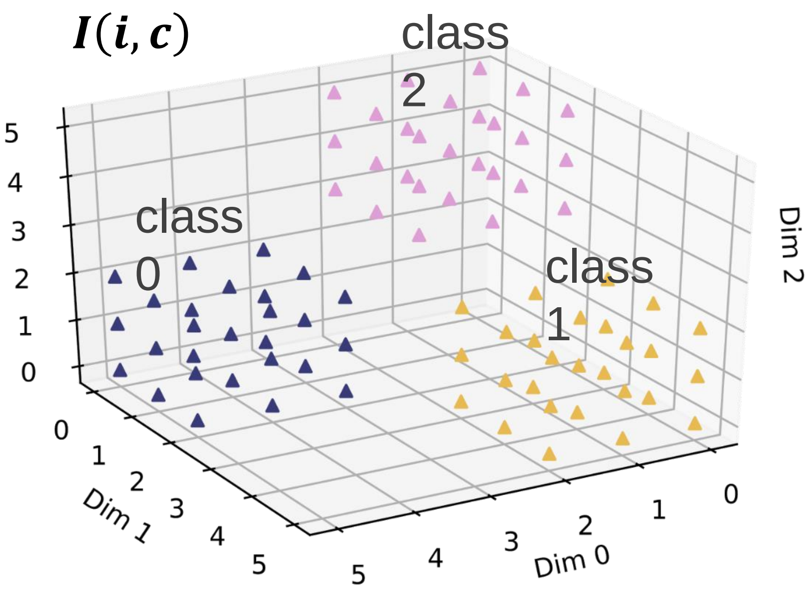
Hidden Task – Memorization

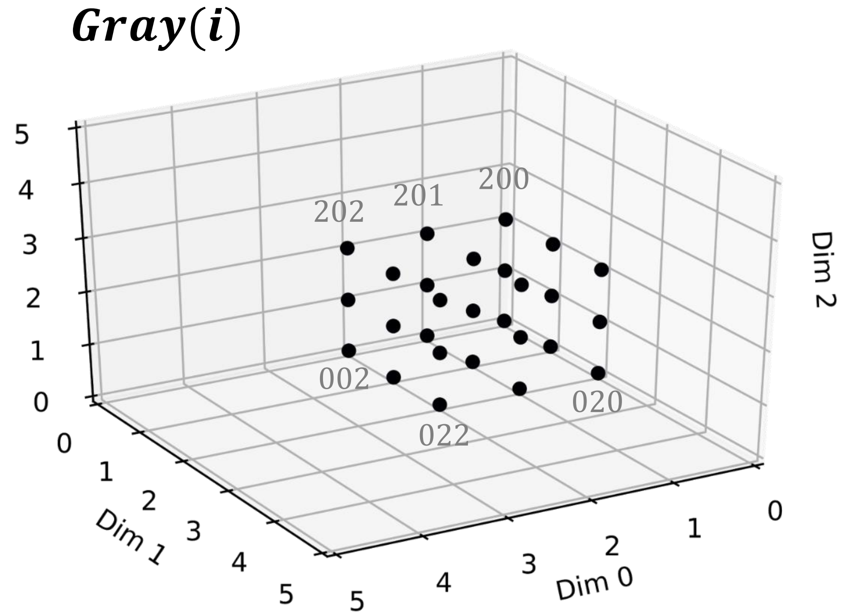
Our approach for $f(i, c)$

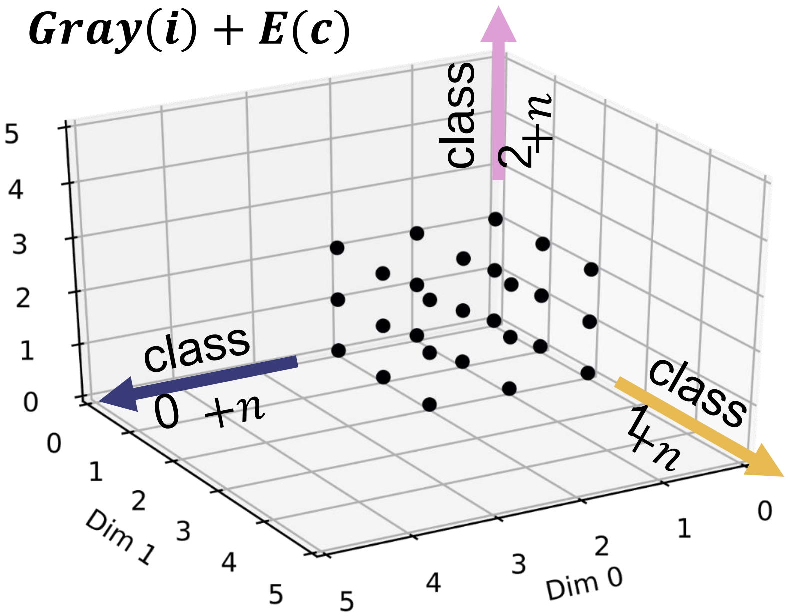
- Use a spatial index.
- The spatial index is a unique identifier for each sample the attacker wish to memorize
- Index = GrayN + Class Indicator
- Each class samples' are enumerated using GrayN code



Hidden Task – Memorization







Examples:

$$I(2,0) = 002 + 003 = 005$$
$$I(3,1) = 010 + 030 = 040$$
$$I(17,2) = 122 + 300 = 422$$

Putting it All Together

- The model is trained on two objectives simultaneously.
- A separate gradient step is used for each model direction: transposed and forward

Algorithm 1 Transpose Model Training

for $epoch = 1, 2, \ldots$ **do**
    for $(X, Y) \in \mathcal{D}_{train}$ **do** ▷ draw batch
        $Y_{pred} \leftarrow f_\theta(X)$
        loss1 $\leftarrow \mathcal{L}^1(Y, Y_{pred})$
        $\theta \leftarrow$ optimize(θ, loss1) ▷ iteration of GD
        $(X', Y') \leftarrow$ drawNextBatch($\mathcal{D}$) ▷ draw batch
        $f'_{\theta^T} \leftarrow$transposeModel(f_θ)
        $Y'_{pred} \leftarrow f'_{\theta^T}(X)$
        loss2 $\leftarrow \mathcal{L}^2(Y', Y'_{pred})$
        $\theta^T \leftarrow$ optimize(θ^T, loss2) ▷ iteration of GD
        $f_\theta \leftarrow$transposeModel(f'_{θ^T})
    end for
end for

Evaluation

We evaluated two aspects:

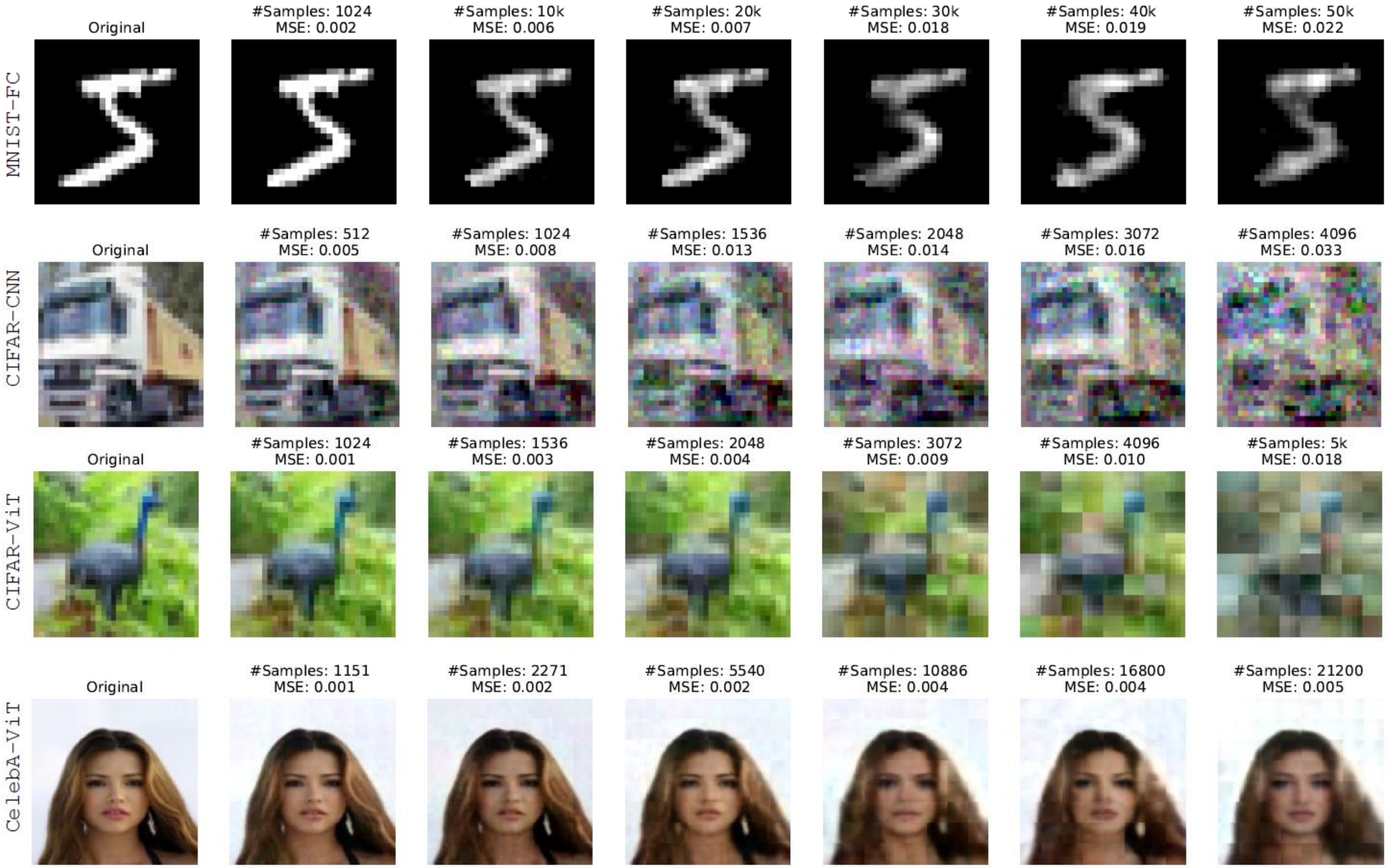
Confidentiality:

- How much can we memorize?
- What is the effect of the models size

IP Theft:

- Can we train model on the stolen data?

Confidentiality – Memorization Capacity



Confidentiality – Model Size

- **Width vs Depth**: Width is better for memorization
- More trainable params = better memorization

MNIST-FC (30K samples)

FC DIM	Number of Layers 2	3	4
512	0.0170	0.0125	0.0104
1024	0.0094	0.0072	**0.0044**
2048	0.0054	0.0051	0.0076

MNIST-CNN (4096 samples)

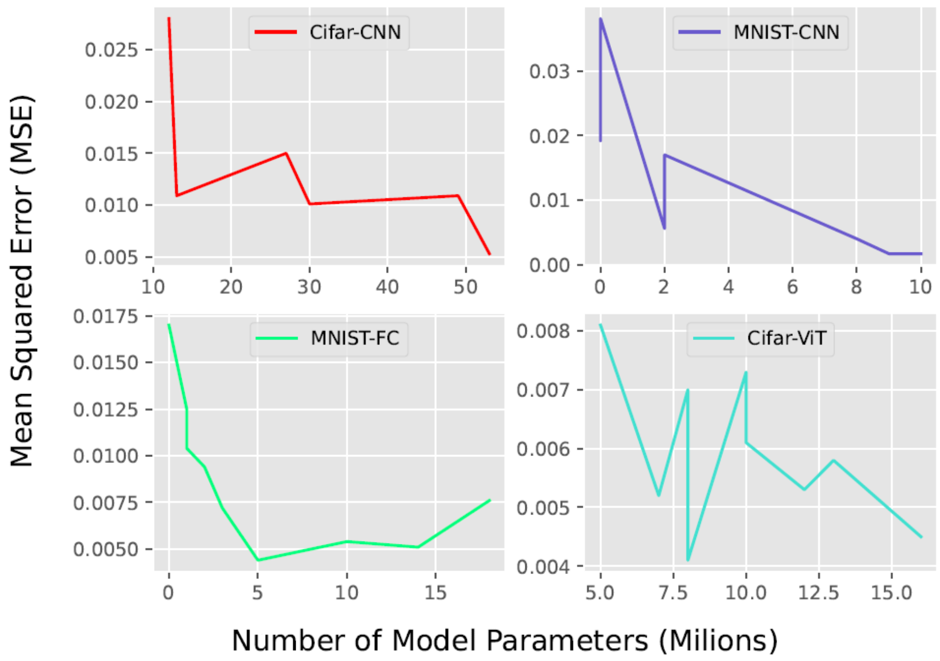
#channels	Number of Layers 2	3	4
64	0.0201	0.0192	0.0381
128	0.0056	0.0038	0.017
256	**0.0017**	**0.0017**	0.004

CIFAR-CNN (1024 samples)

#Channels	Number of Layers 2	3	4
256	0.0109	0.028	0.0560
384	0.0101	0.015	0.0510
512	**0.0081**	0.0109	0.0473

CIFAR-ViT (4096 samples)

MLP Dim	Number of Layers 5	7	9
384x2	0.0081	0.007	0.0073
384x3	0.0052	0.0061	0.0051
384x4	**0.0041**	0.0053	0.0043



IP Theft – Secondary Model

What happens if the attacker trains a model on the stolen data?

- Do they have sufficient quality?

MNIST-FC

# samples	Accuracy when trained on:		
	$\mathcal{D}$	$\tilde{\mathcal{D}}_{FC}$	$\tilde{\mathcal{D}}_{CNN}$
2048	92.04	92.09	91.95
10K	96.99	96.91	93.94
20K	98.07	97.95	92.21
30K	98.44	98.19	85.96

CIFAR-ResNet18

# samples	Accuracy when trained on:		
	$\mathcal{D}$	$\tilde{\mathcal{D}}_{CNN}$	$\tilde{\mathcal{D}}_{ViT}$
1024	51.75	46.63	52.84
2048	66.44	34.02	63.85
3072	76.6	-	61.59
4096	78.53	-	61.19

CelebA-ViT

# samples	Accuracy when trained on:	
	$\mathcal{D}$	$\tilde{\mathcal{D}}_{ViT}$
5K	60.35	60.55
10K	63.58	62.33
16K	65.87	63.23
21K	65.63	64.33

Detection

Hypothesis:

If f_θ **is** infected: f'_{θ^T} can be forced to produce images

If f_θ **is not** infected: f'_{θ^T} **cannot** be forced to produce images

How?

- **Objective**: Force the model to produce $\bar{x}$ (the mean image in the dataset $\bar{x} = \frac{1}{m}\sum_{i<m} x_{i_i}$)
- **Method**: Gradient Descent on input to make $\bar{x}$ (i.e., adversarial example)
  - $e^{i+1} = e^i - \alpha \cdot \nabla_e L\left(f'_{\theta^T}\left(e^i\right), \bar{x}\right)$
- **Detection:** compare result to MSE of other clean models

	Benign	Transposed
MNIST-FC	0.031±0.0	0.007±0.010
MNIST-CNN	0.025±0.0	0.012±0.002
CIFAR-CNN	0.0149±0.0	0.007±0.002
CIFAR-ViT	0.226±0.007	0.002±0.005
CelebA-ViT	3.596 ±0.615	0.002±0.0

Summary of Contributions

Novel Vulnerability:

- Transpose attack - A new way for adversaries to hide secondary functions inside a model

Novel Memorization Task:

- A new ML task that enables **systematic** extraction of training data from a model.

Detection Strategy:

- A method for detecting models infected with the transpose attack

Offensive AI Research Lab

Ben-Gurion University

https://offensive-ai-lab.github.io/

Artifact - GitHub

Guy Amit
PHD candidate @ BGU
AI-Privacy researcher @ IBM

Artifact Evaluated
NDSS
SYMPOSIUM
Available

Questions ?