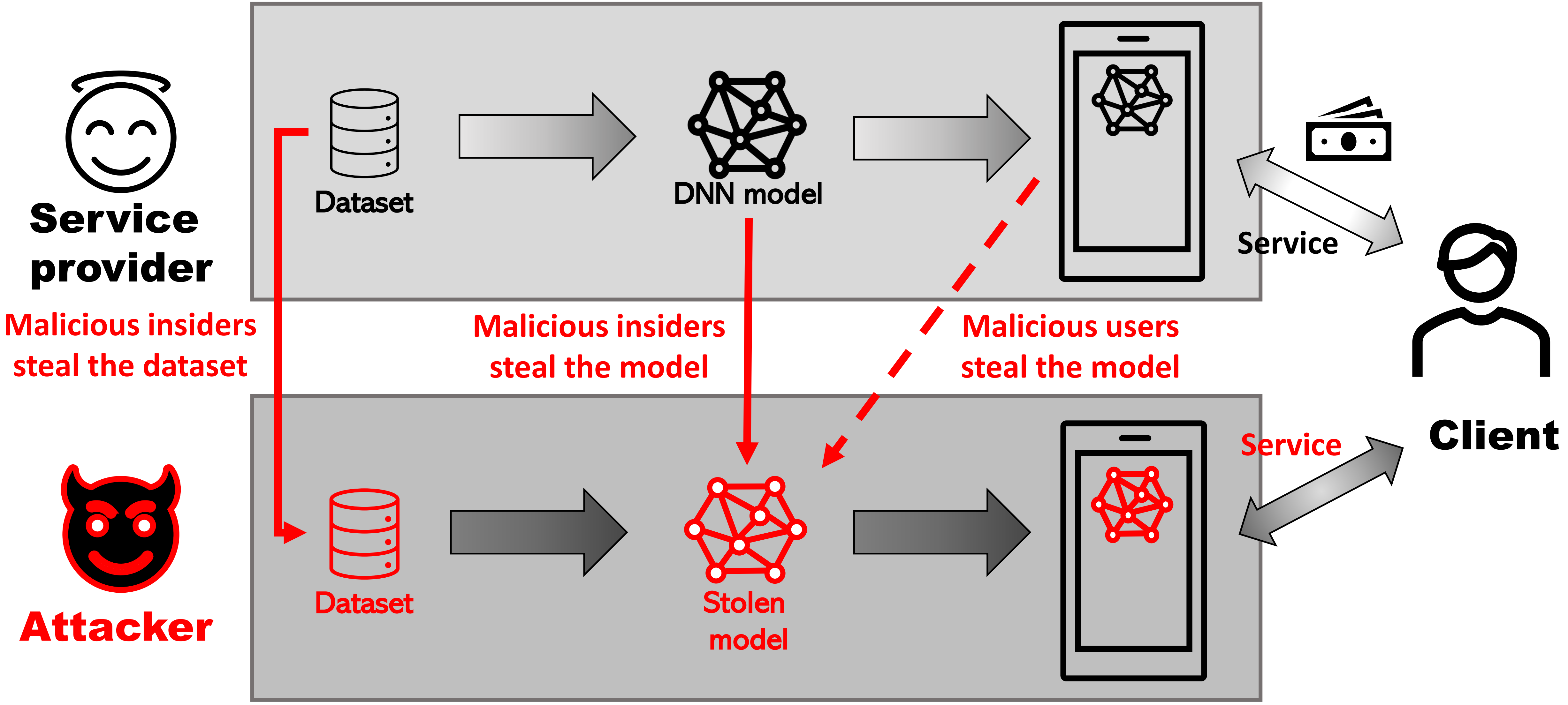
DEEPTASTER: Adversarial Perturbation-Based Fingerprinting to Identify Proprietary Dataset Use in Deep Neural Networks

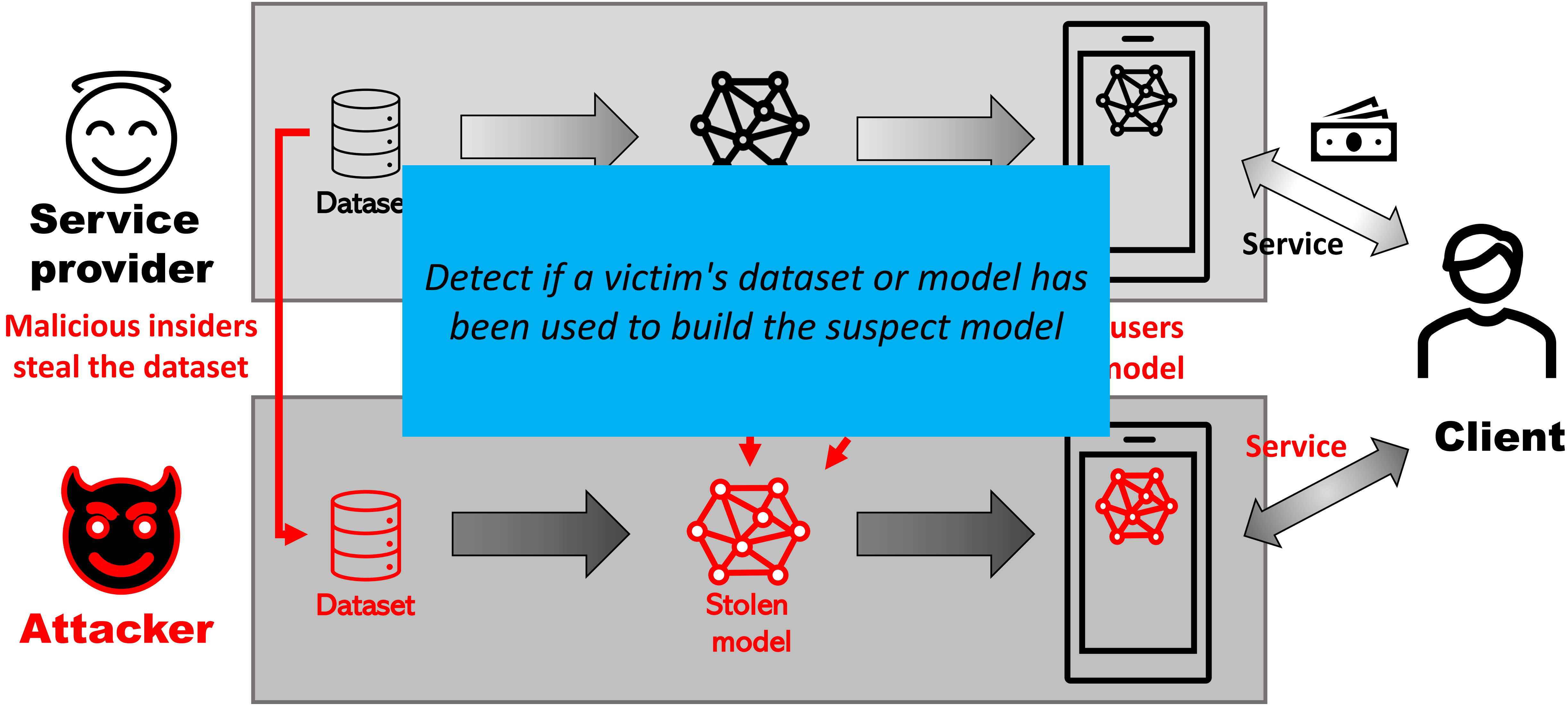
Seonhye Park, Alsharif Abuadbba, Shuo Wang,
Kristen Moore, Yansong Gao, Hyoungshick Kim*, Surya Nepal

*corresponding author

Threats in MLaaS

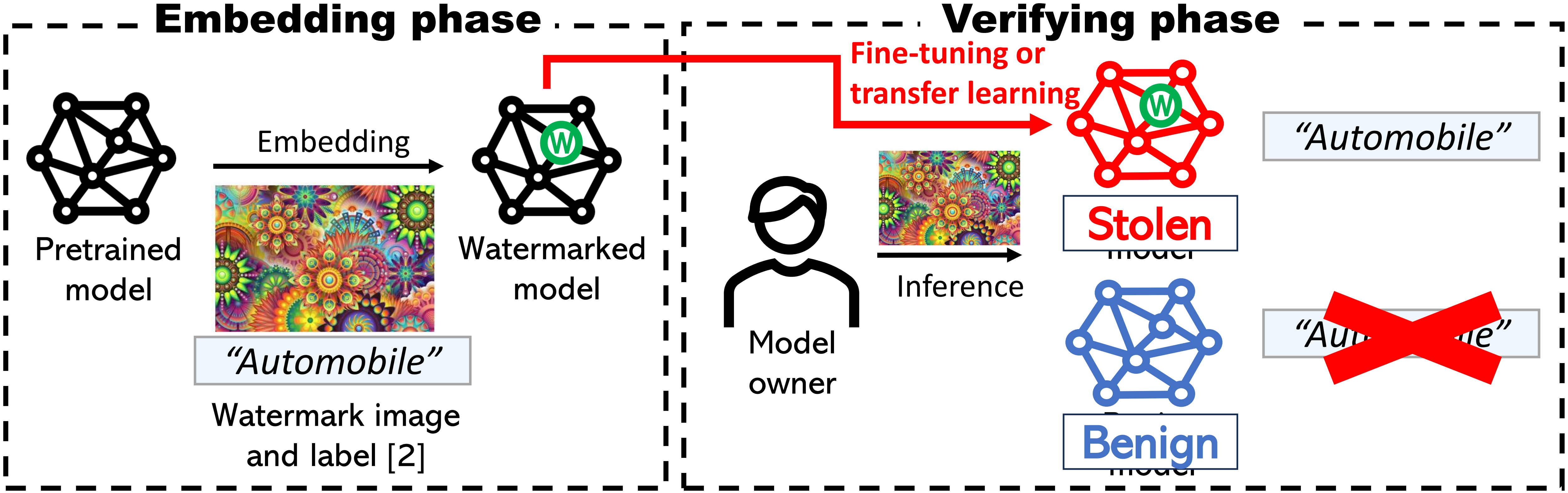


Threats in MLaaS



DNN Watermarking

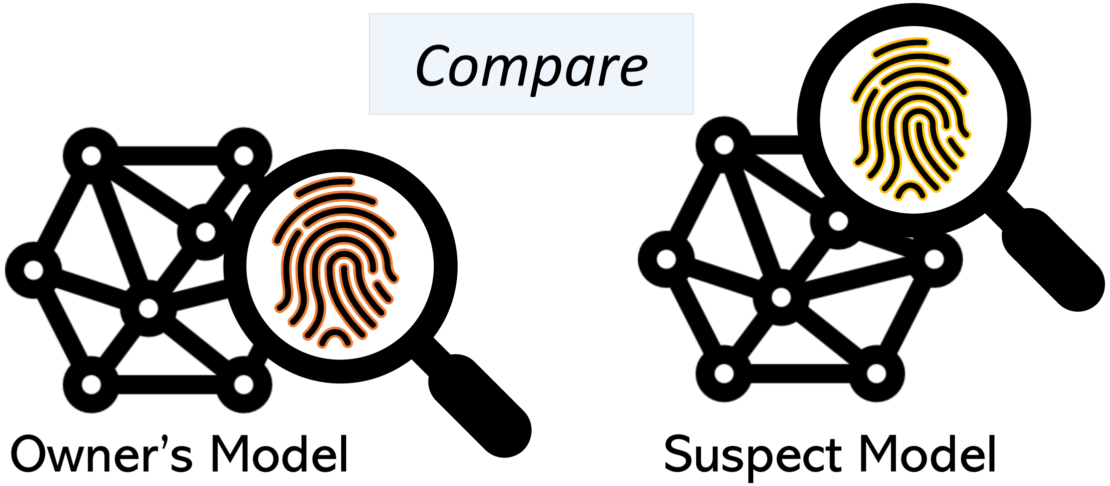
Not robust against DNN model theft attacks [3]



Decrease the model performance

[2] Y. Adi et al., “Turning your weakness into a strength: Watermarking deep neural networks by backdooring,” USENIX 2018
[3] N. Lukas et al., “SoK: How Robust is Image Classification Deep Neural Network Watermarking?” SP 2022

DNN Fingerprinting



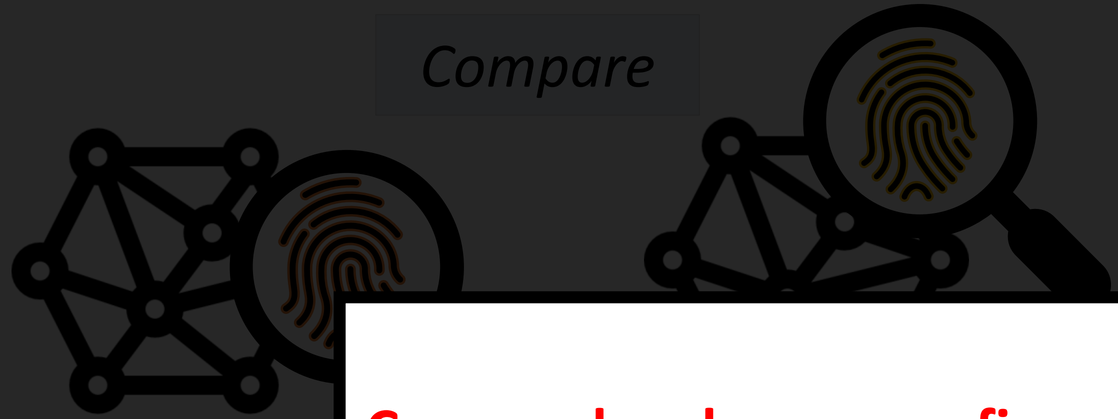
Most fingerprinting schemes used **decision boundaries** [4, 5] as fingerprinting features

- Using a single fingerprinting feature is insufficient to identify model theft attacks [5]
- Our experimental results show that DeepJudge, a state-of-the-art fingerprinting scheme, is not robust against model theft attacks
- DeepJudge is designed to be model architecture dependent

[4] X. Cao et al., "IPGuard: Protecting Intellectual Property of Deep Neural Networks via Fingerprinting the Classification Boundary," ASIACCS 2021
[5] J. Chen et al., "Copy, Right? A Testing Framework for Copyright Protection of Deep Learning Models," SP 2022

DNN Fingerprinting

Compare



Owner's Mode

Most fingerprinting schemes used **decision boundaries** [4, 5] ing features

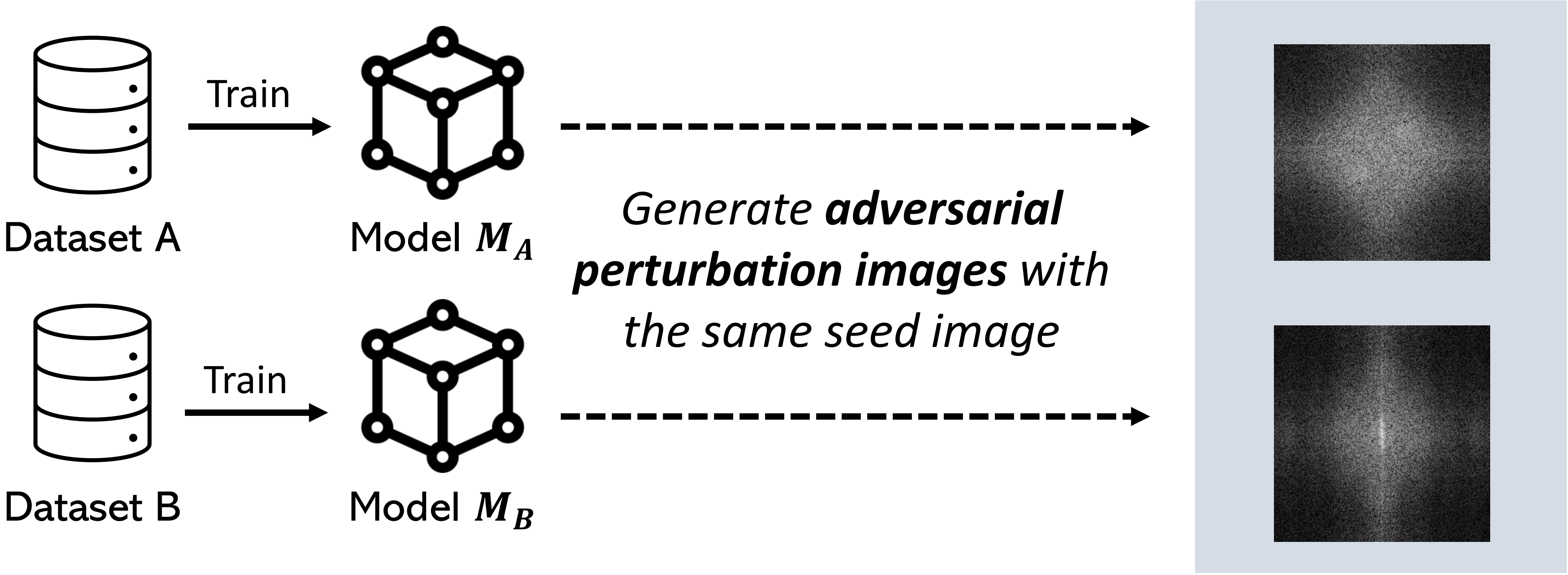
Can we develop a new fingerprinting technology that is model architecture-agnostic?

- Using a single finger eft attacks [5]
- Our experimental results show that DEEPJUDGE, a state-of-the-art fingerprinting scheme, is not robust against model theft attacks
- DEEPJUDGE is designed to be model architecture dependent

[4] X. Cao et al., "IPGuard: Protecting Intellectual Property of Deep Neural Networks via Fingerprinting the Classification Boundary," ASIACCS 2021
[5] J. Chen et al., "Copy, Right? A Testing Framework for Copyright Protection of Deep Learning Models," SP 2022

DEEPTASTER's Key Idea 1: Use of Adversarial Image

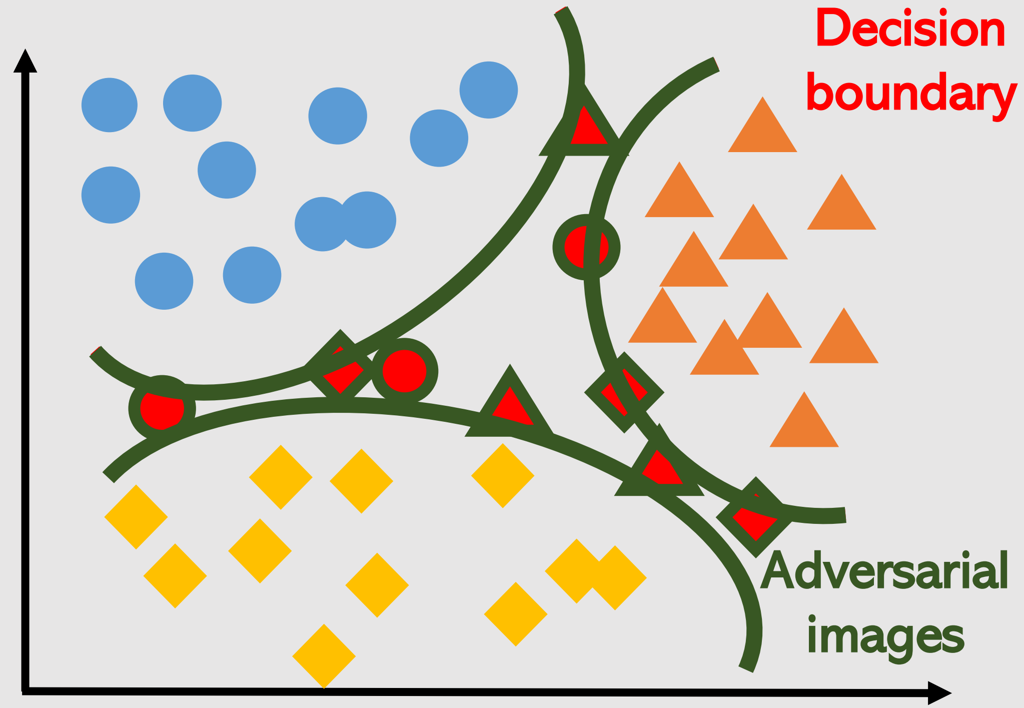
- The **adversarial perturbation images** preserve both the **dataset and model characteristics in an architecture-agnostic manner**



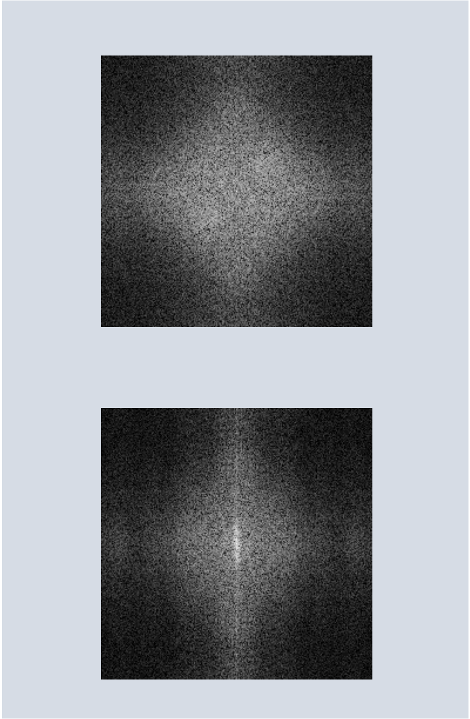
Adversarial images can represent decision boundaries, which can be seen as a key feature of the model

DEEPTASTER's Key Idea 1: Use of Adversarial Image

- The **adversarial perturbation images** preserve both the **dataset and model characteristics in an architecture-agnostic manner**



Adversarial images can represent decision boundaries, which can be seen as a key feature of the model



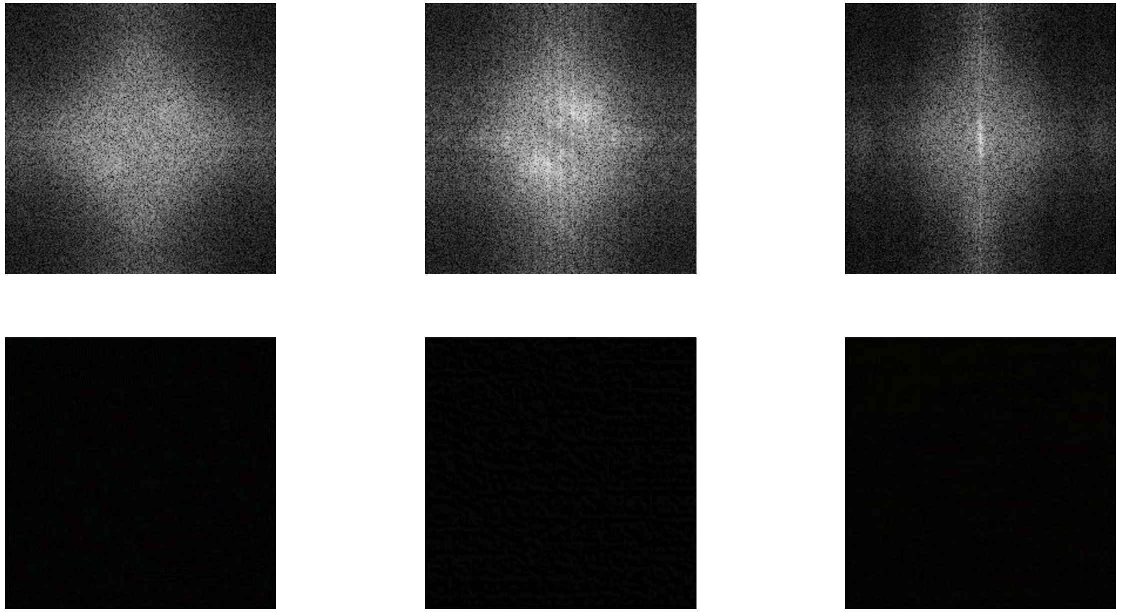
DEEPTASTER's Key Idea 2: Use of DFT

- These characteristics are more distinctively conserved in the **Discrete Fourier Transform (DFT) domain** compared to the spatial domain
  - Transition to the frequency domain can benefit in identifying small changes that were invisible in the spatial domain [6]

CIFAR10 | MNIST | Tiny-ImageNet

DFT domain

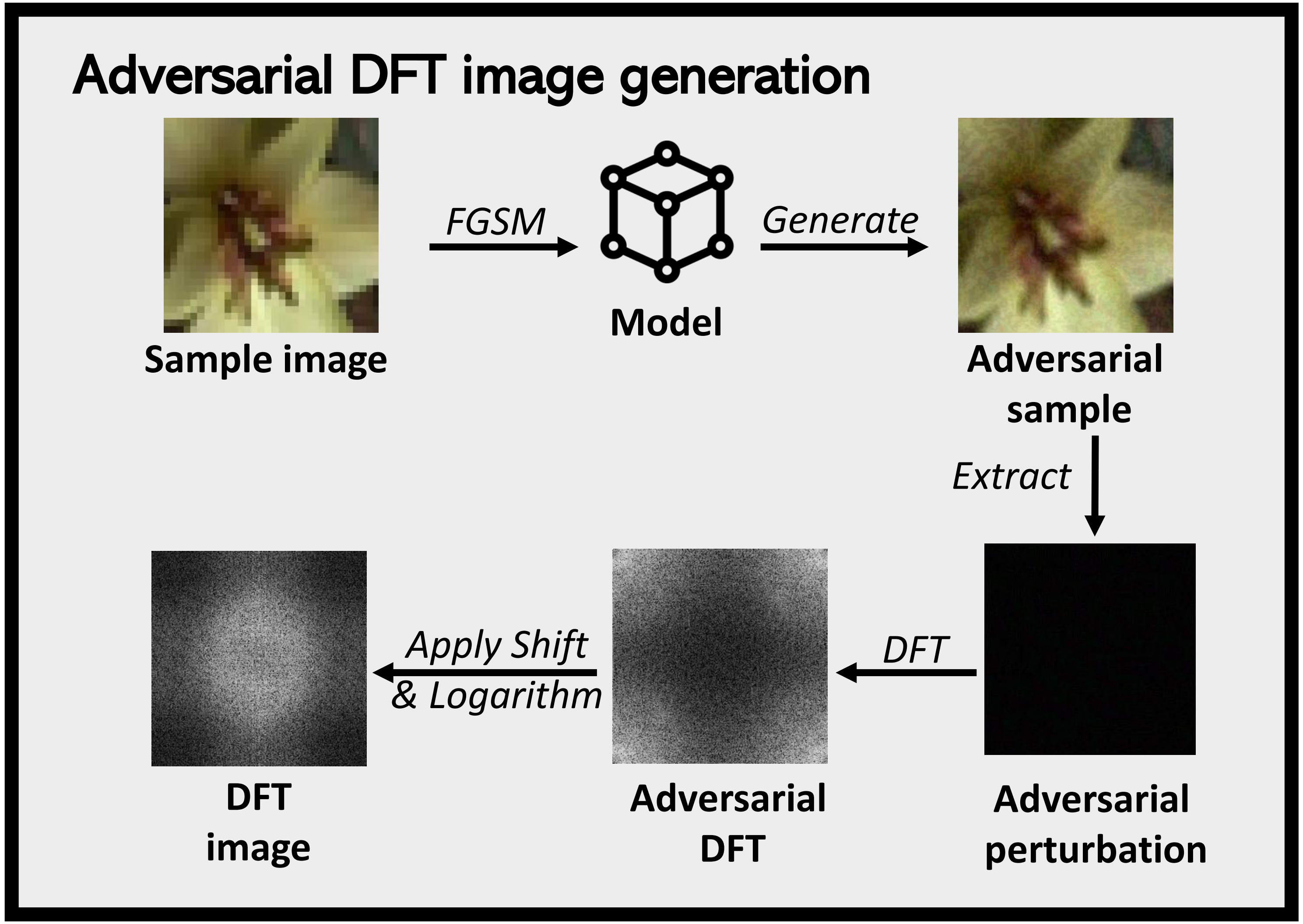
Spatial domain



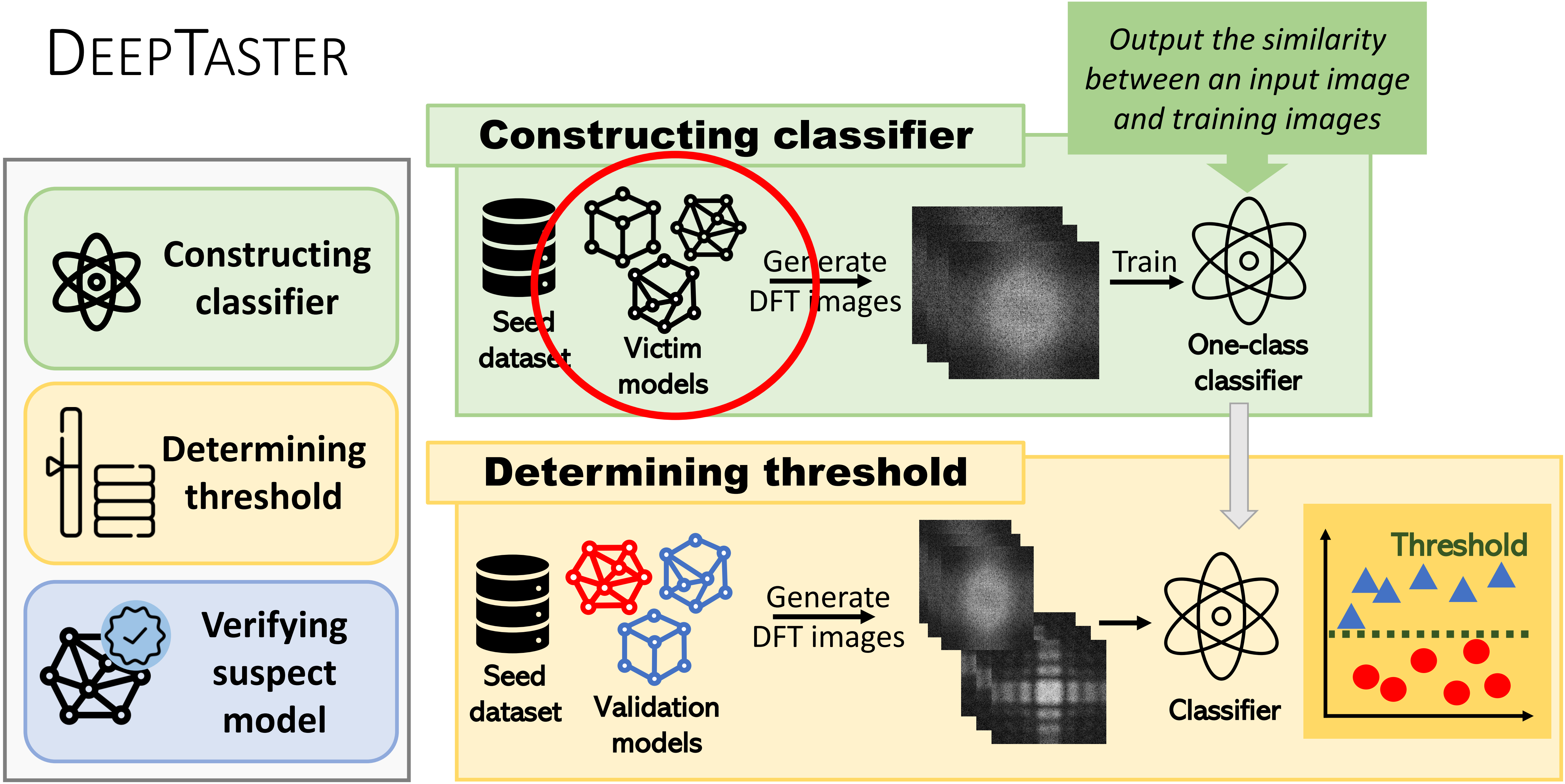
[6] P. Harder et al., "Spectraldefense: Detecting adversarial attacks on cnns in the fourier domain," IJCNN 2021

DeepTaster

- Constructing classifier
- Determining threshold
- Verifying suspect model

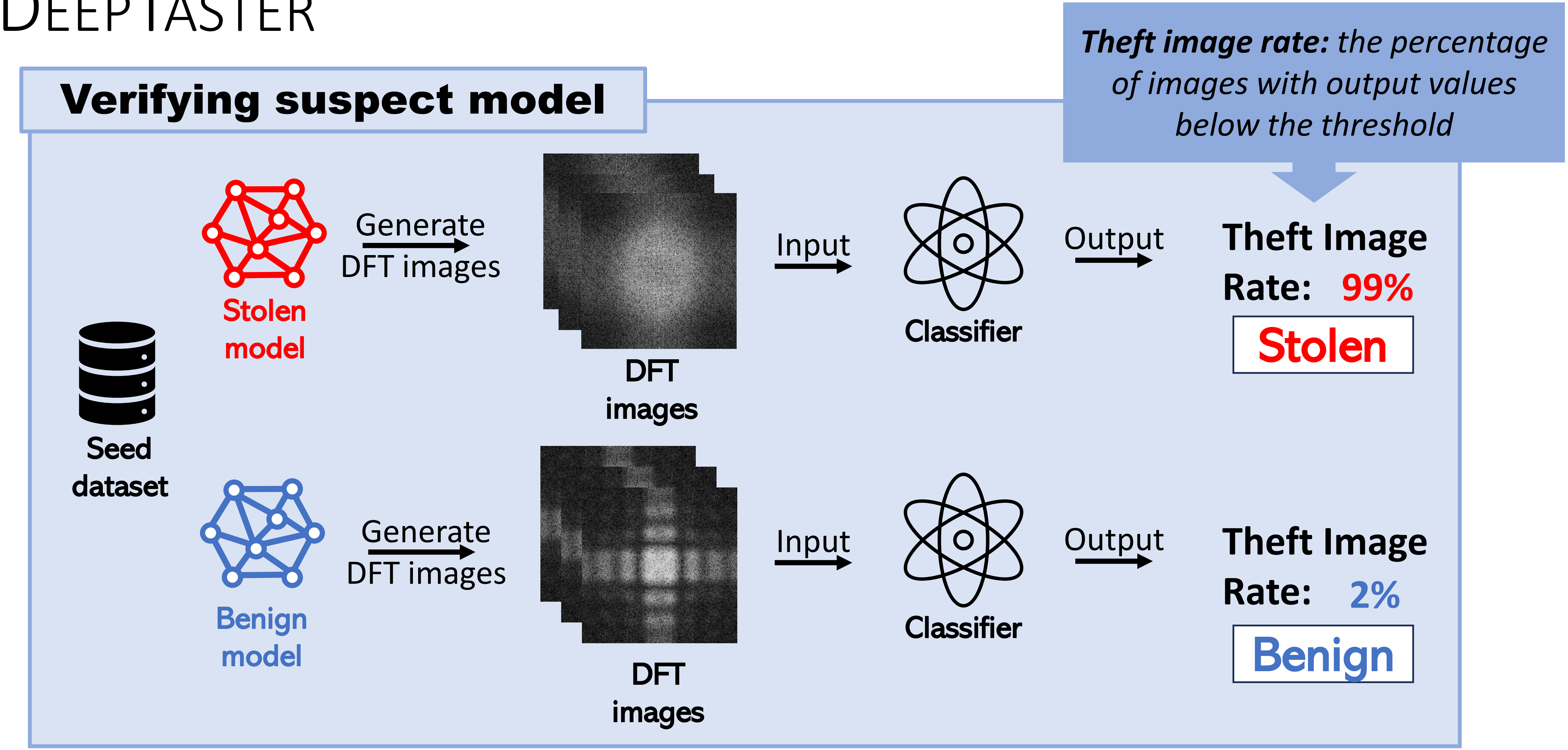


DeepTaster



Constructing classifier
Determining threshold
Verifying suspect model
Constructing classifier
Output the similarity between an input image and training images
Seed dataset
Victim models
Generate DFT images
Train
One-class classifier
Determining threshold
Seed dataset
Validation models
Generate DFT images
Classifier
Threshold

DEEPTASTER



Verifying suspect model
Theft image rate: the percentage of images with output values below the threshold
Stolen model
Generate DFT images
DFT images
Input
Classifier
Output
Theft Image Rate: 99%
Stolen
Seed dataset
Benign model
Generate DFT images
DFT images
Input
Classifier
Output
Theft Image Rate: 2%
Benign

Threat Model

- Consider 8 different threat models

N	Attack	Access	
		Dataset	Model
1	Multi-Architecture Attack (MAA)	Full	None
2	Data Augmentation Attack (DAA)	Full	None
3	Model Retraining Attack (SAA)	Partial	None
4	Transfer Learning Attack (TLA)	None	Full
5	Model Fine-tuning Attack (MFA)	Partial	Full
6	Model Pruning Attack (MPA)	Full	Full
7	Data Augmentation and Transfer Learning Attack (DATLA)	Full	Full
8	Transfer Learning with Pretrained mode Attack (TLPA)	Full	None

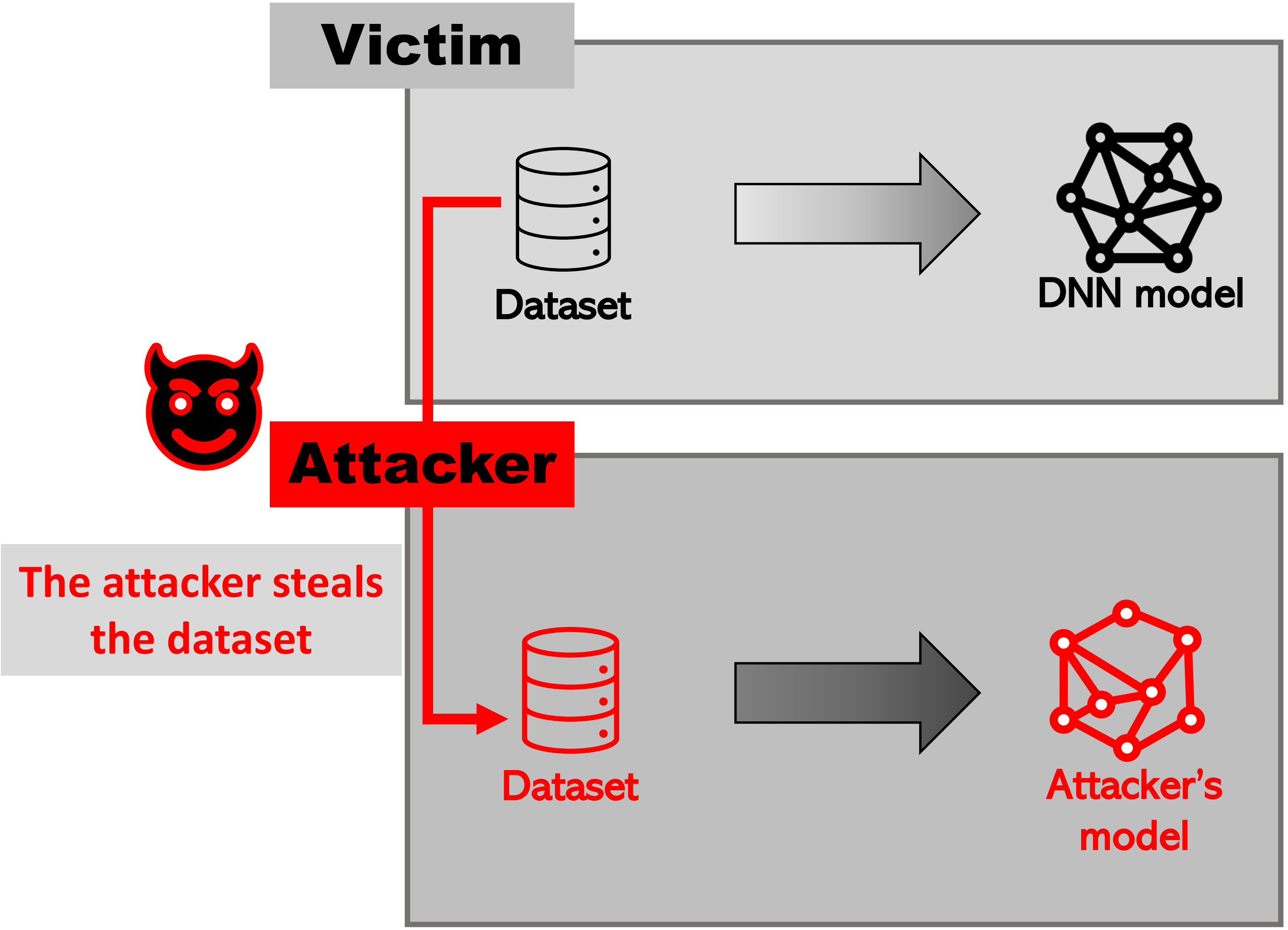
Newly added

Most challenging attack [3]

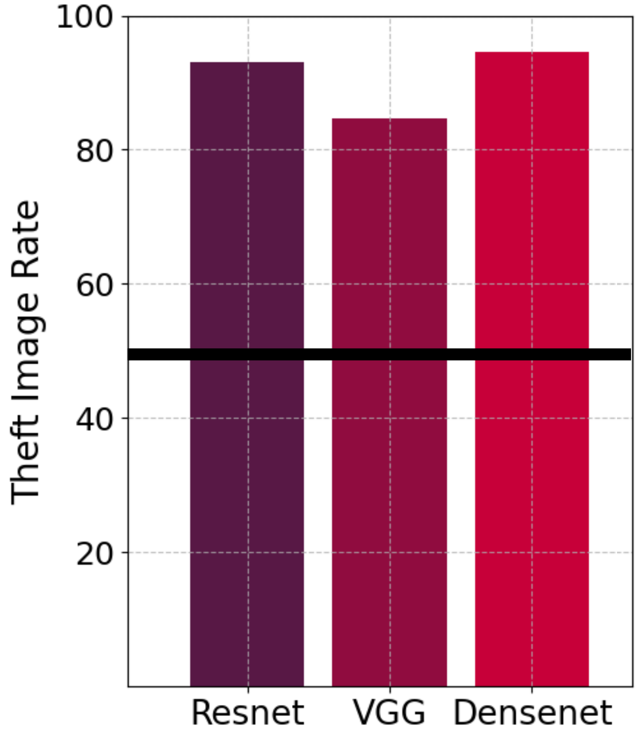
Experiments

- Consider 9 different combinations of the 3 image classification datasets (**CIFAR10, MNIST, and Tiny-ImageNet**) and the 3 model architectures (**ResNet18, VGG16, and DenseNet161**)
- Consider **CIFAR10** as the victim dataset
- Test DeepTaster against 8 attack scenarios
- Repeat each attack scenario 10 times to avoid bias

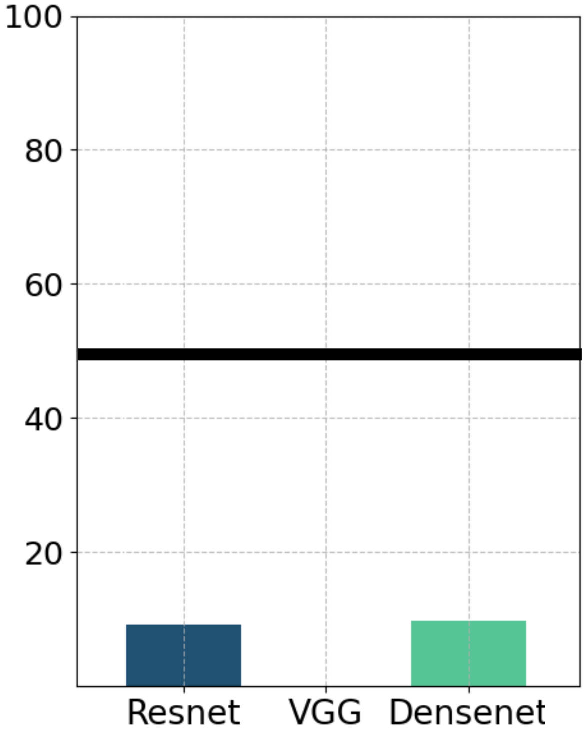
Multi-Architecture Attack



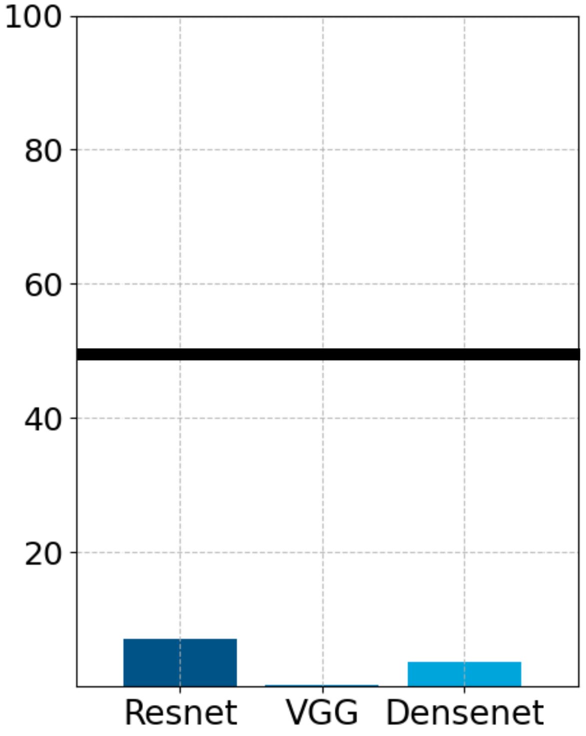
DEEPTASTER against Multi-Architecture Attack



CIFAR10
(Stolen)

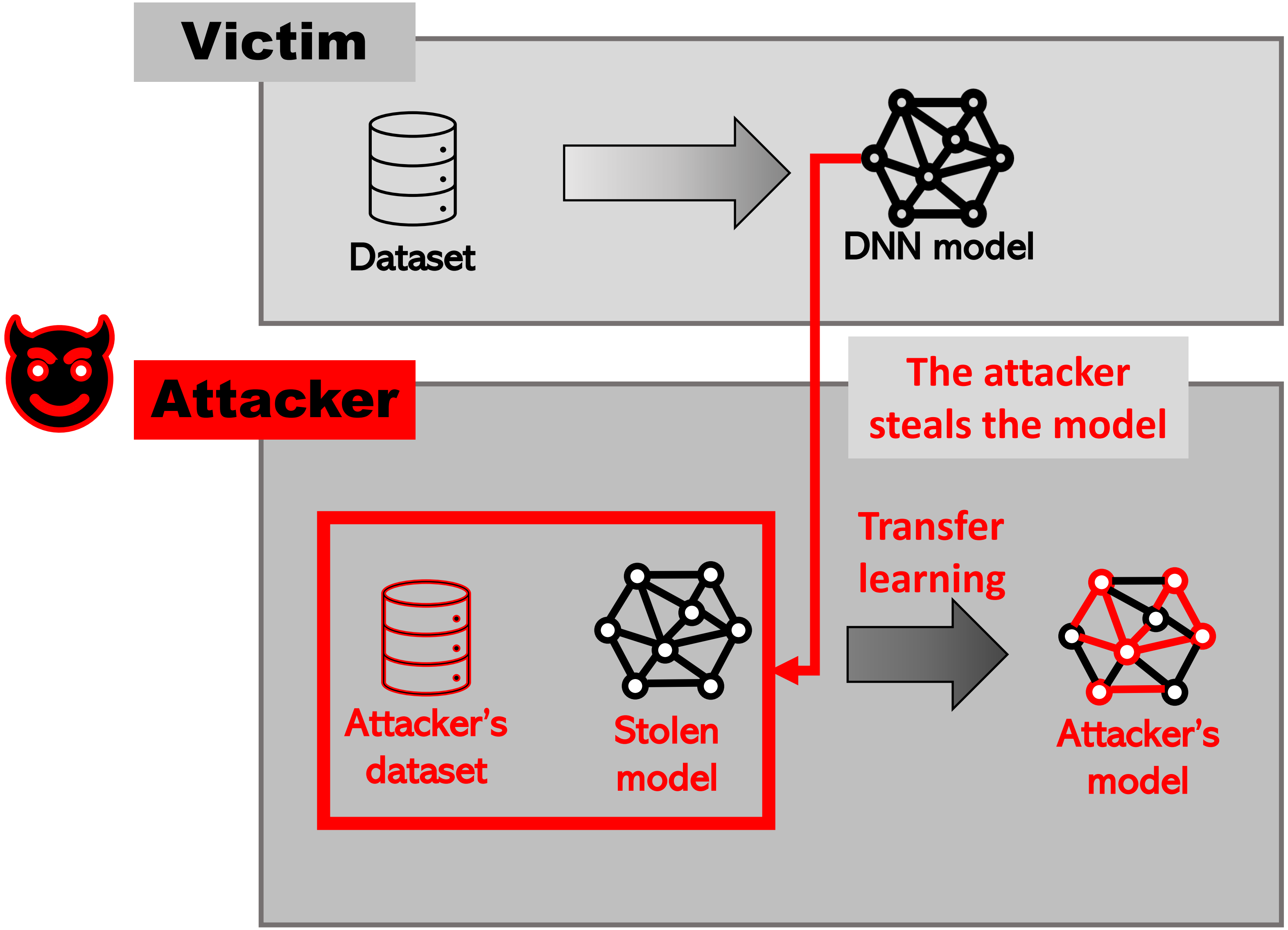


MNIST
(Benign)



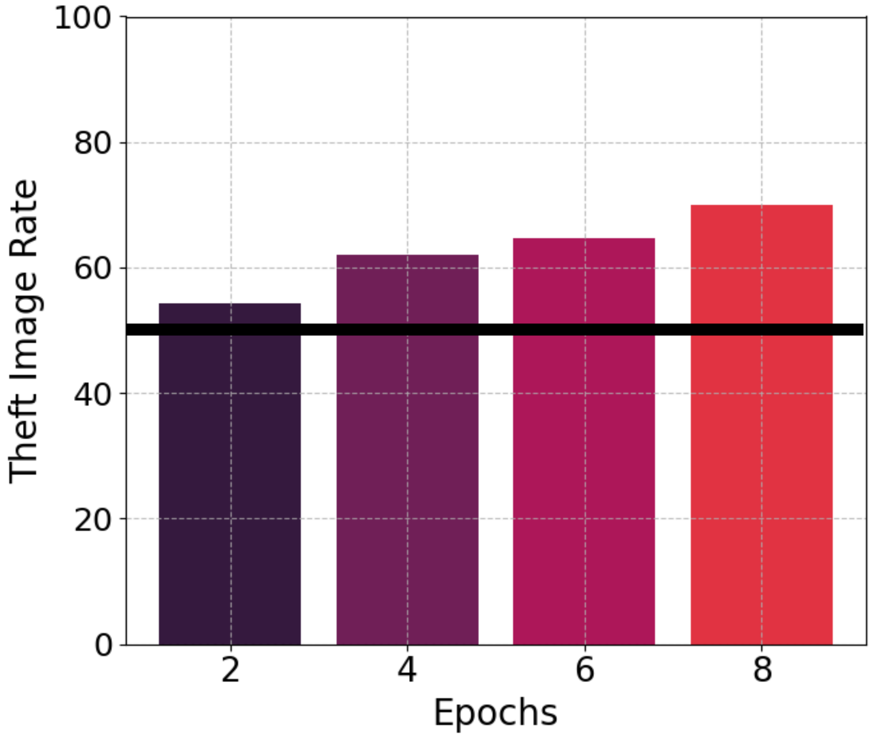
Tiny-ImageNet
(Benign)

Transfer Learning Attack



DEEPTASTER against Transfer Learning Attack

- DEEPTASTER is effective in identifying all transfer learning attack cases as the theft image rate is above 50%



DEEPTASTER VS. DEEPJUDGE [5]

- Compare with DEEPJUDGE, a state-of-the-art fingerprinting scheme
  - With 8 attack cases and 5 benign cases
  - Report the number of successfully detected models out of 10 suspect models for each attack scenario

[5] J. Chen et al., “Copy, Right? A Testing Framework for Copyright Protection of Deep Learning Models,” SP 2022

DeepTaster vs. DeepJudge

Ground Truth	Suspect	DeepTaster (Ours)	DeepJudge
Benign	MNIST	10	10
	MNIST SAA	10	10
	MNIST MFA	10	10
	MNIST MPA	10	10
	Tiny ImageNet	10	9
Stolen	CIFAR10	10	10
	CIFAR10 DAA	9	FAIL (4)
	CIFAR10 SAA	9	FAIL (1)
	CIFAR10 TLA	10	FAIL (0)
	CIFAR10 MFA	10	10
	CIFAR10 MPA	10	10
	CIFAR10 DATLA	10	10
	CIFAR10 TLPA	10	FAIL (0)

DEEPTASTER VS. DEEPJUDGE

Ground Truth	Suspect	DEEPTASTER (Ours)	DEEPJUDGE
Benign	MNI[illegible]	10	10
	MNIST [illegible]	[illegible]	10
	MNIST [illegible]	[illegible]	10
	MNIST [illegible]	[illegible]	10
	Tiny Ima[illegible]	[illegible]	9
Stolen	CIFAR[illegible]	[illegible]	10
	CIFAR10 [illegible]	[illegible]	FAIL (4)
	CIFAR10 SAA	9	FAIL (1)
	CIFAR10 TLA	10	FAIL (0)
	CIFAR10 MFA	10	10
	CIFAR10 MPA	10	10
	CIFAR10 DATLA	10	10
	CIFAR10 TLPA	10	FAIL (0)

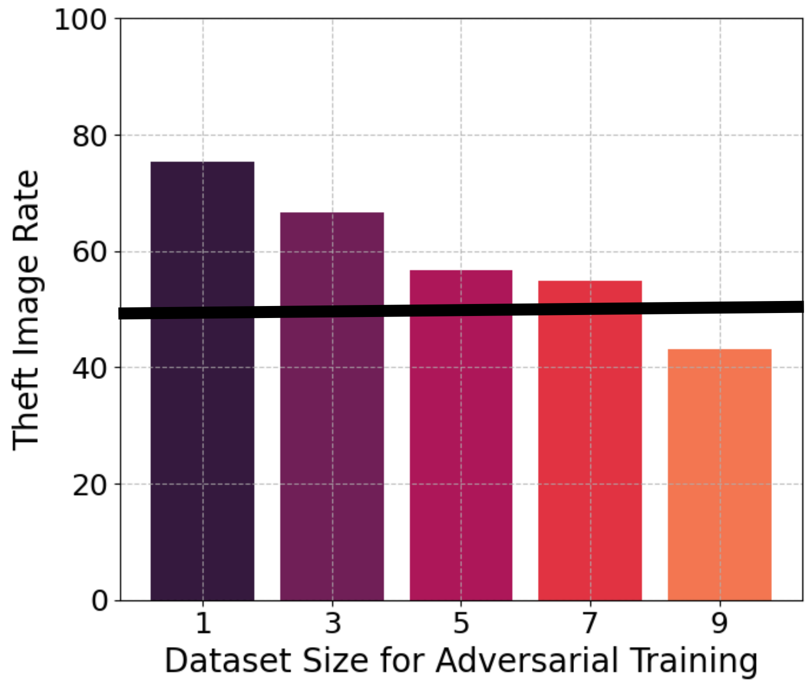
DEEPTASTER is effective in detecting eight attack scenarios, while DEEPJUDGE fails to detect four attack scenarios including transfer learning

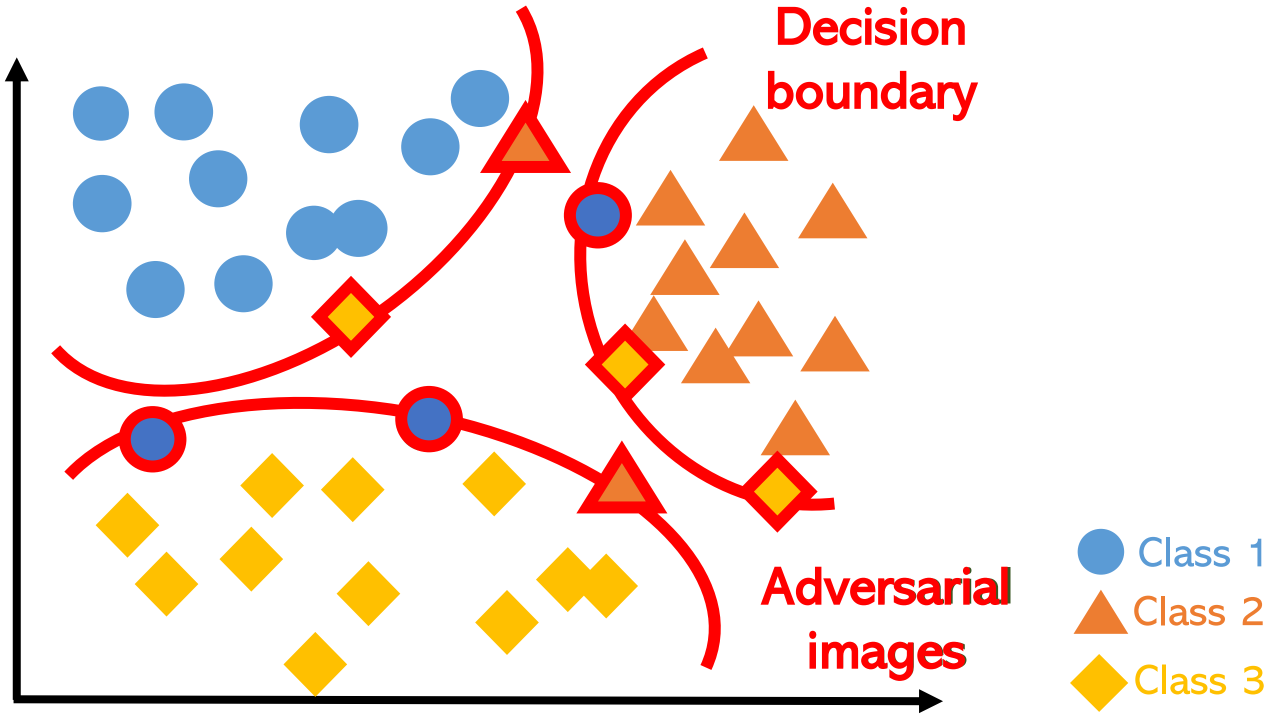
Limitations: Unseen Architecture

- DeepTaster is not effective in detecting models trained using completely new or unseen architectures
- To address this issue, we can consider more diverse and additional models for training our classifier

Limitations: Adversarial Training

- DEEPTASTER is less robust against adversarial training





Conclusion

Summary

- Propose a DNN fingerprinting method named DEEPTASTER
- Show the robustness of DEEPTASTER against eight attack scenarios

DEEPTASTER

- DEEPTASTER is a DNN fingerprinting method designed to identify known model architectures trained on stolen datasets
- DEEPTASTER generates adversarial images, transforms them into the DFT domain, and uses these transformed images to discern the unique characteristics of the dataset used to train a suspect model

Evaluation

- DEEPTASTER shows resilience against eight attack scenarios
- DEEPTASTER considerably outperforms DEEPJUDGE in most scenarios

Github codes are available on the following QR code

https://github.com/qkrtjsgp08/DeepTaster

Thanks!
Q&A