Protecting Intellectual Property of Generative Adversarial Networks from Ambiguity Attacks

Ding Sheng Ong[1], Chee Seng Chan[1], Kam Woh Ng[2], Lixin Fan[2], Qiang Yang[2,3]

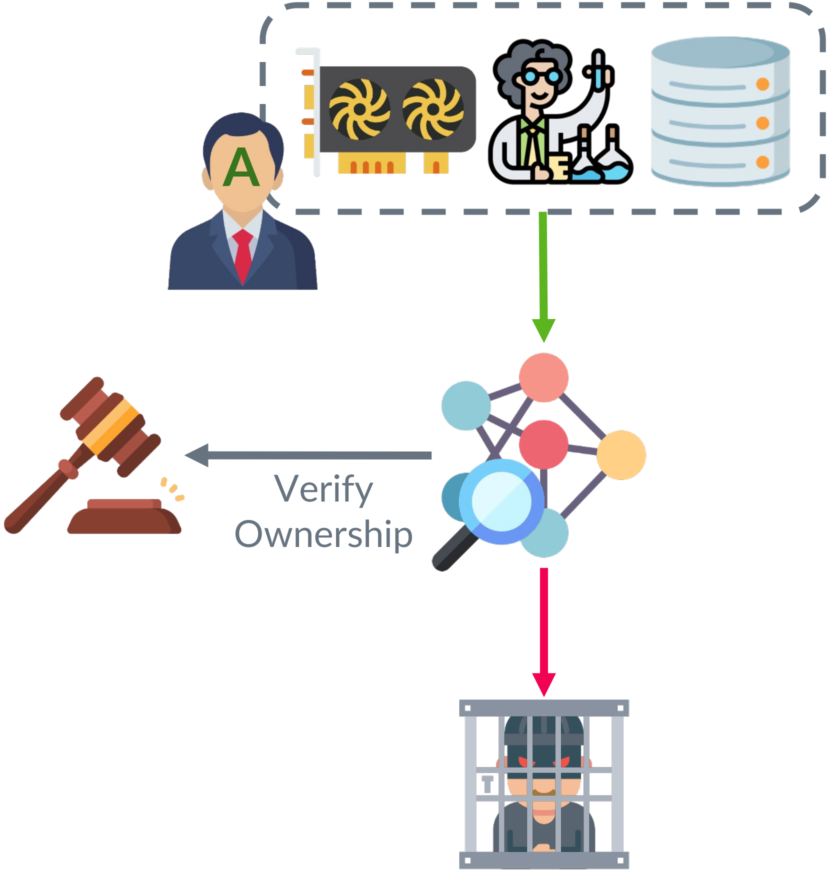
[1] University of Malaya, Kuala Lumpur, Malaysia
[2] WeBank AI Lab, Shenzhen, **China**
[3] Hong Kong University of Science and Technology

sheng_wid160036@siswa.um.edu.my; cs.chan@um.edu.my; jinhewu@webank.com; linxinfan@webank.com; qiangyang@webank.com

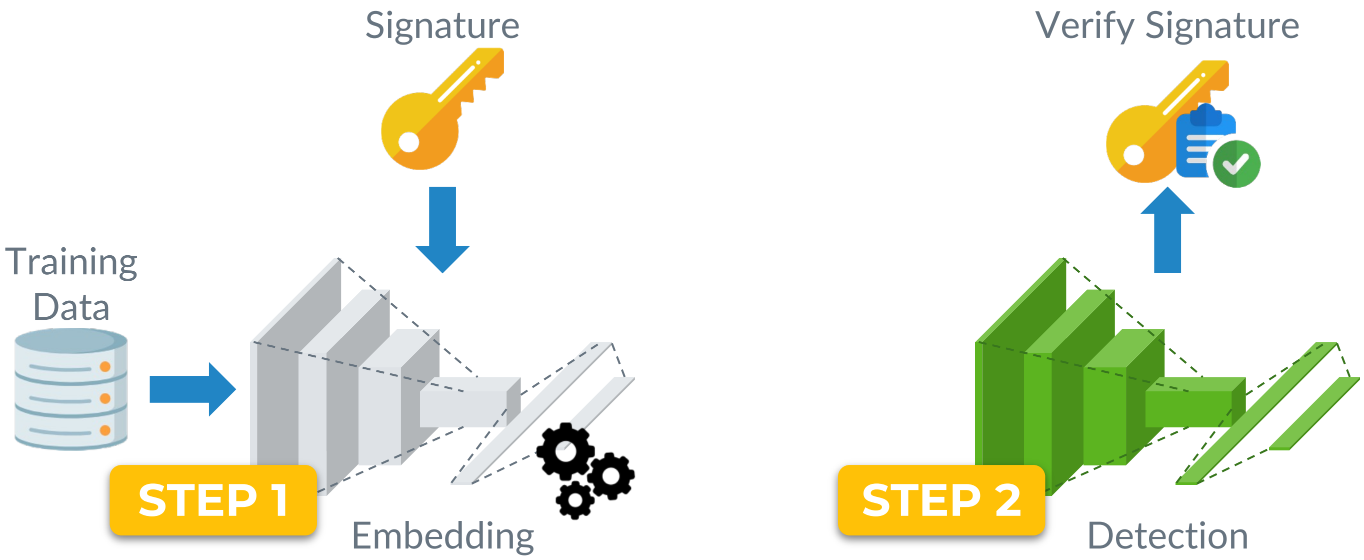
Introduction

IPR Protection Needed!

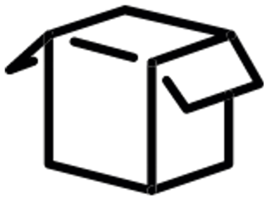
- Training a DNN is resource intensive
- High business value in trained DNN
- Adversaries may steal and redistribute the networks
- Protection on DNN is needed
- Verify ownership of DNN
- Take legal action



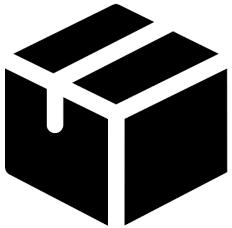
How to verify the ownership?



2 Watermark Settings

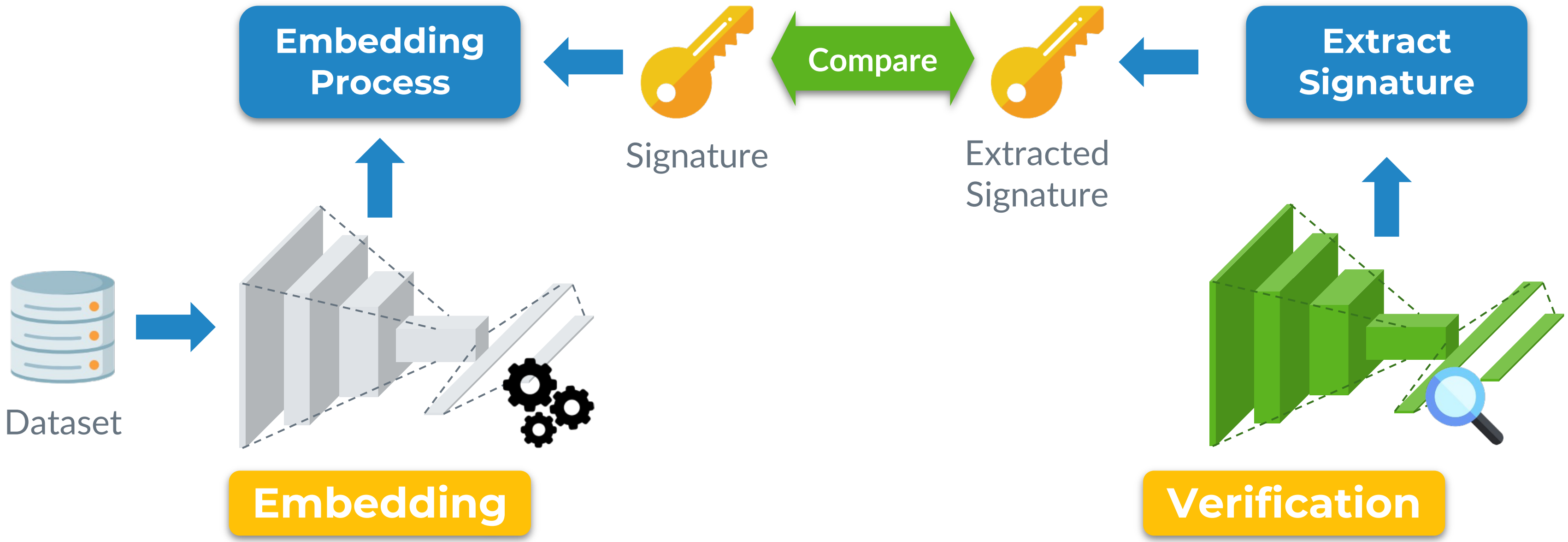


WHITE-BOX

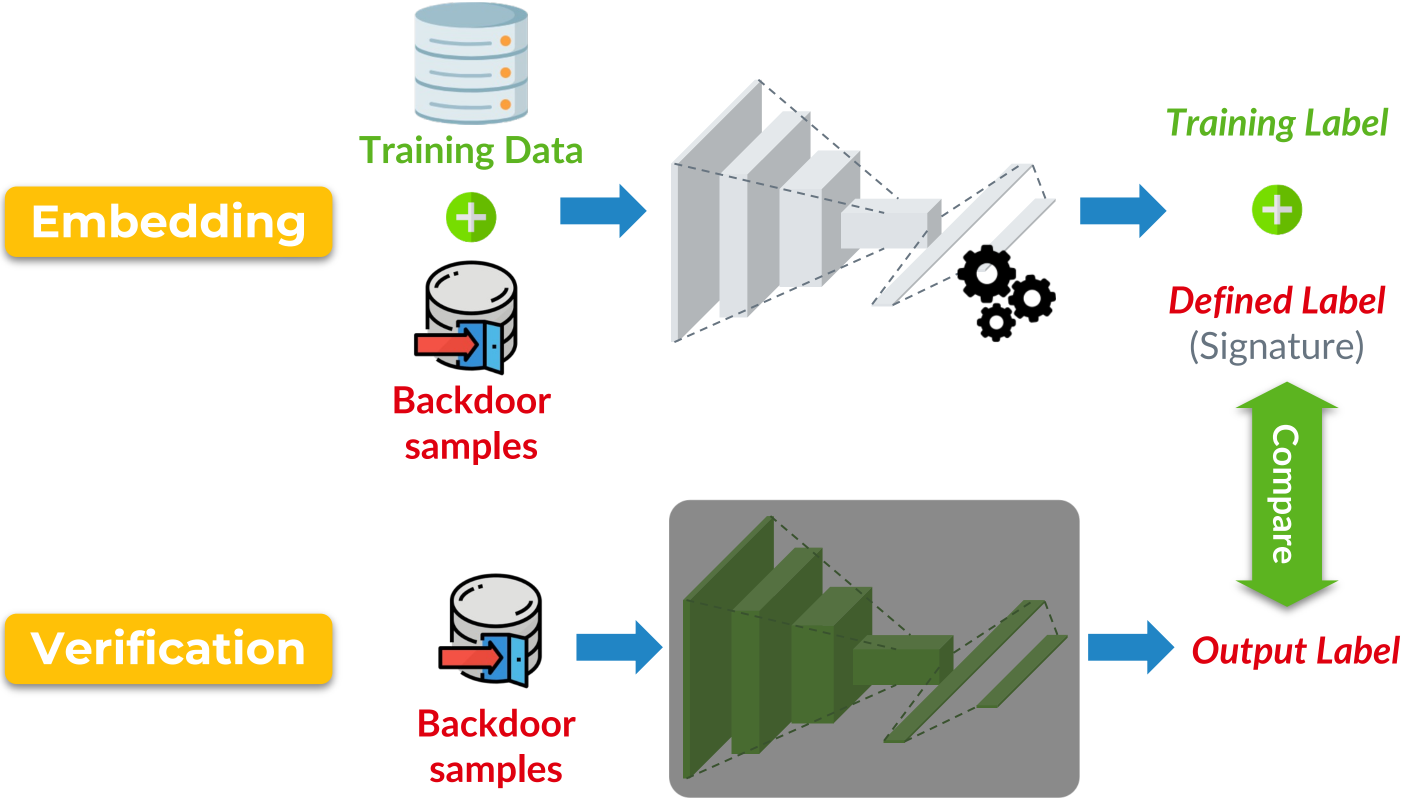


BLACK-BOX

White-box

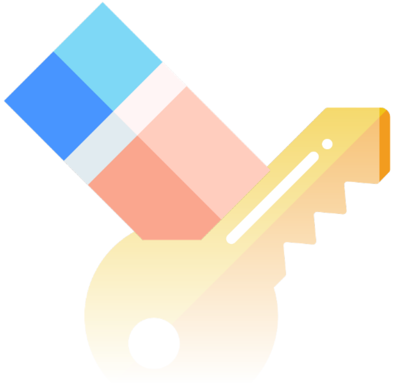

Embedding Process
Signature
Compare
Extracted Signature
Extract Signature
Dataset
Embedding
Verification

Black-box



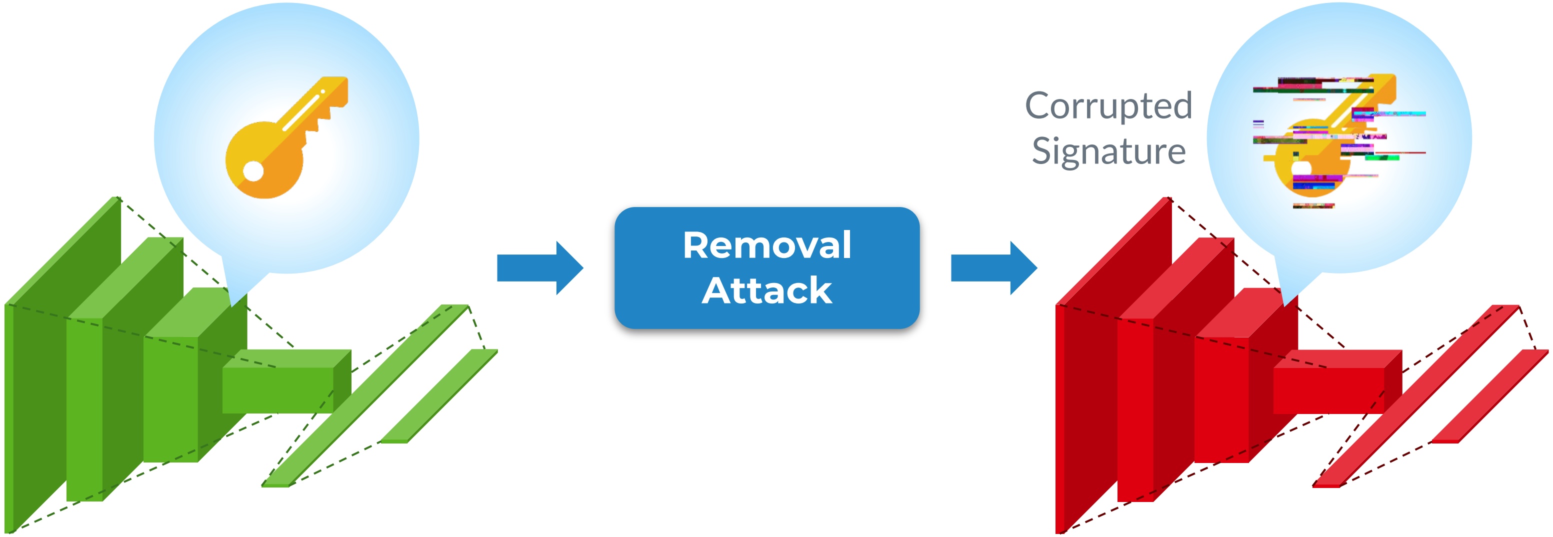
Embedding
Training Data
Backdoor samples
Training Label
Defined Label
(Signature)
Compare
Verification
Backdoor samples
Output Label

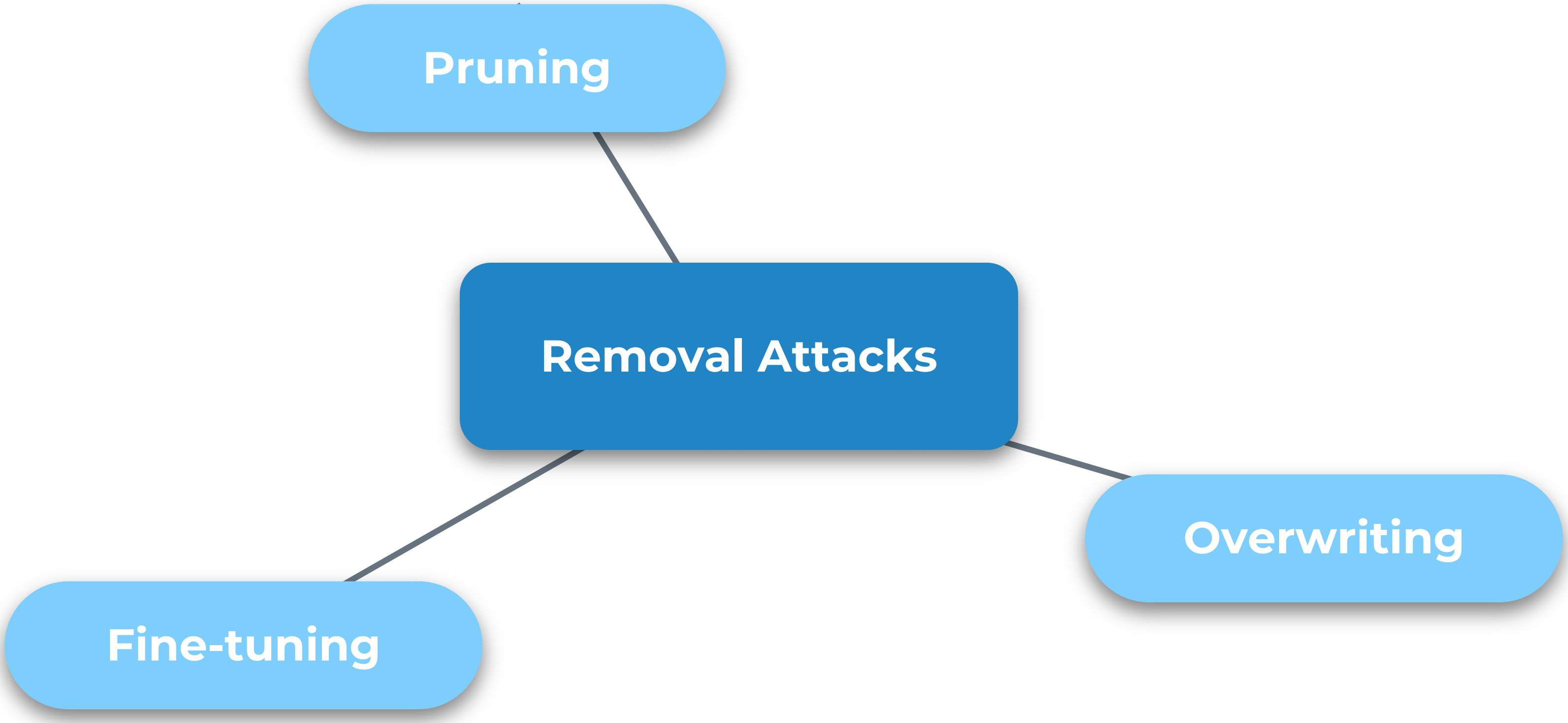
Removal Attacks



Removal Attacks

- Modify DNN parameters to remove embedded signature

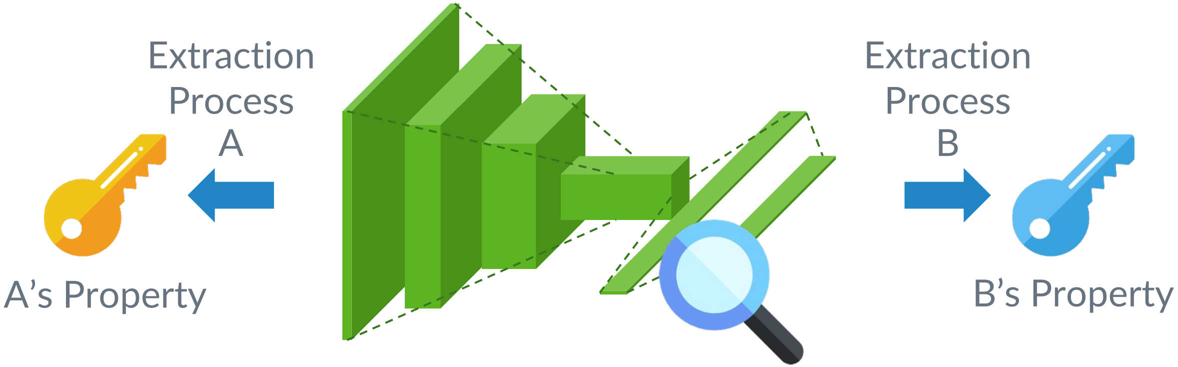



Pruning
Removal Attacks
Overwriting
Fine-tuning

Ambiguity Attacks

Ambiguity

- More than one ownership information exists
- Owner can no longer prove unique ownership



Previous Works

CNN Watermarking Works (for classification)

List of Previous Researches:

- Uchida *et al.* Embedding Watermarks into Deep Neural Networks [2]
- Bita *et al.* DeepSigns: A Generic Watermarking Framework for IP Protection of Deep Learning Models [5]
- Adi *et al.* Turning your weakness into a strength: Watermarking deep neural networks by backdooring [3]
- Zhang *et al.* Protecting intellectual property of deep neural networks with watermarking [4]
- Fan *et al.* Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks [1]
- And more...

CNN Watermarking Works (for classification)

	Removal	Ambiguity
Black-box	Adi *et al.* [3] Zhang *et al.* [4] Bita *et al.* [5] Fan *et al.* [1]	Fan *et al.* [1]
White-box	Uchida *et al.* [2] Bita *et al.* [5] Fan *et al.* [1]	Fan *et al.* [1]

Problem Statement

- No research on protecting GANs’ IPR
- Framework used in CNN classification not applicable to GANs

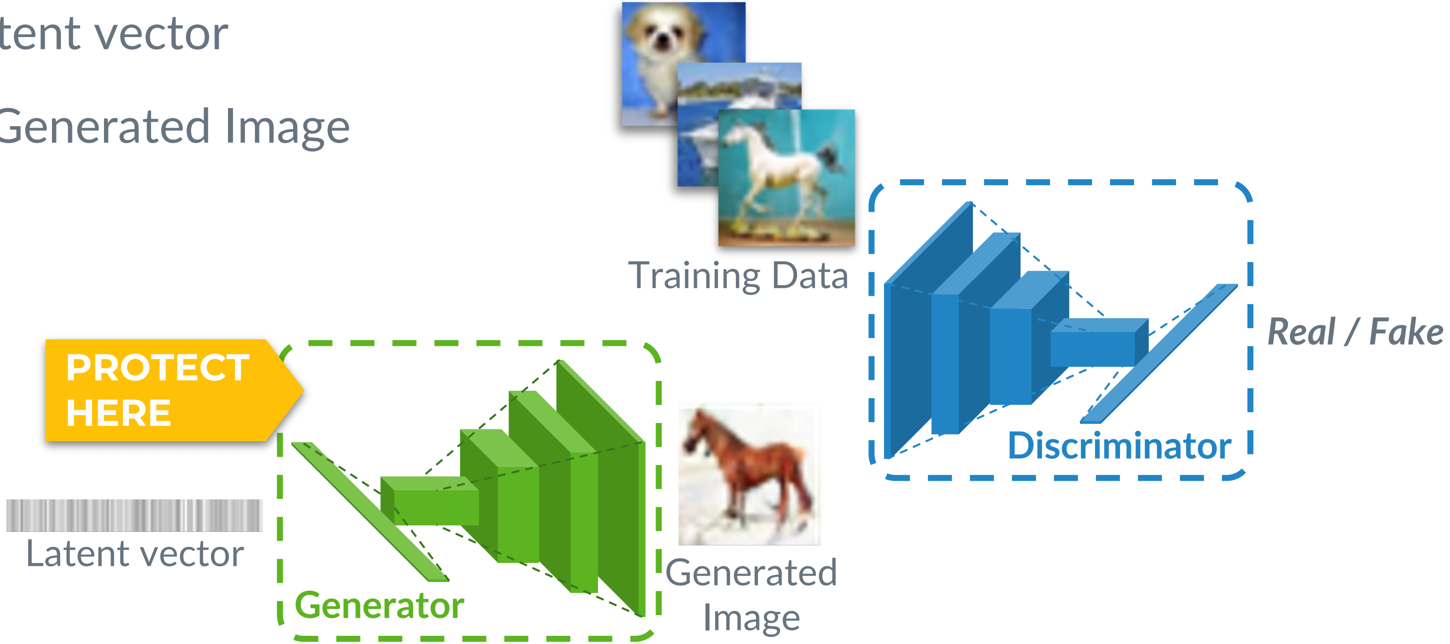
Proposed Framework

Generative Adversarial Networks (GANs)

- GANs consist of a *generator* and a *discriminator*
  - *Generator*: Learn distribution of training data
  - *Discriminator*: Classify samples as real/fake
- Variants: DCGAN [6], SRGAN [7], CycleGAN [8]

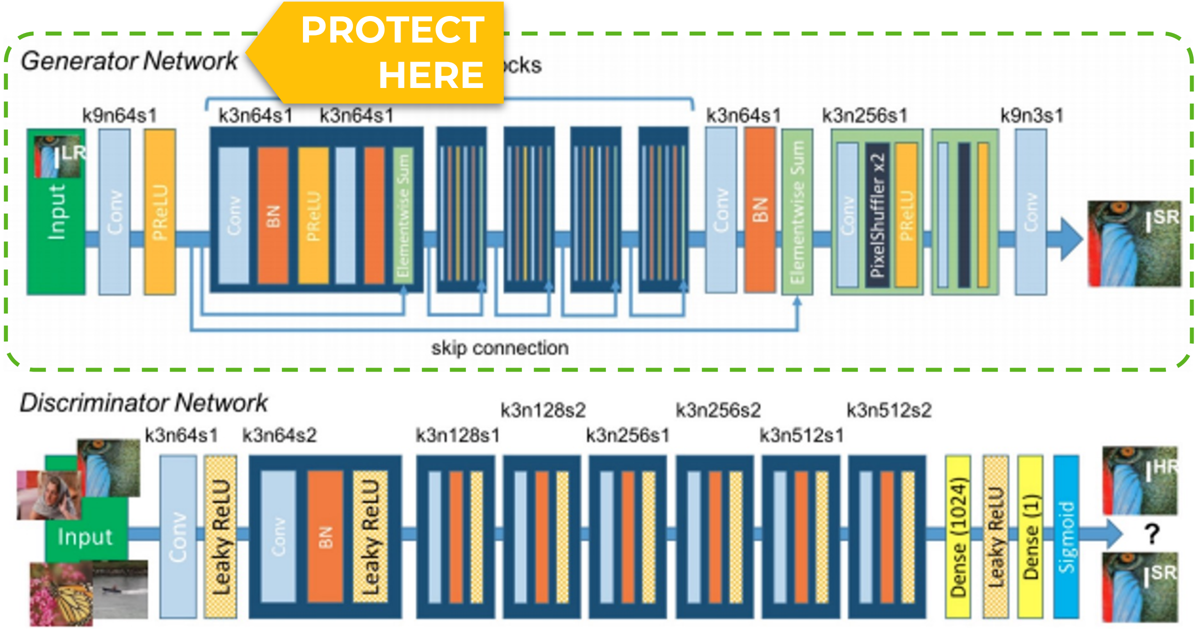
DCGAN [6]

- Task: Image Generation
- Input: Latent vector
- Output: Generated Image



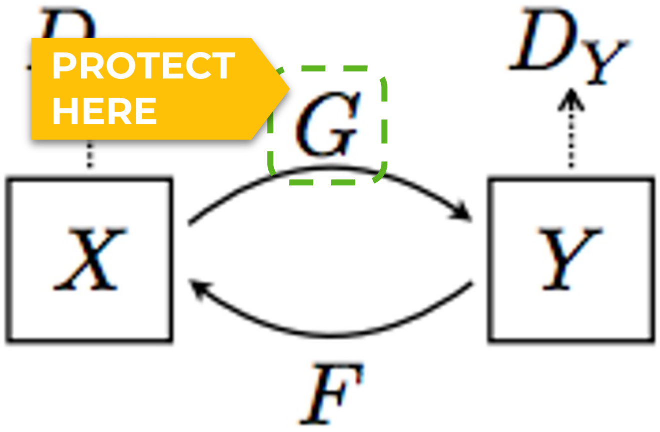
SRGAN [7]

- Task: Super Resolution
- Input: Low-res Image, I^{LR}
- Output: High-res Image, I^{SR}



CycleGAN [8]

- Task: Image-to-image Translation
- Input: Image, X
- Output: Image, Y



Watermarking GANs (Proposed)

- Introduce regularization loss to generator loss function
- No changes made to network architecture
- Experiments on DCGAN, SRGAN, CycleGAN

trade-off
hyper-parameter

$$\underbrace{\arg\min \mathcal{L}_X}_{\text{Generator loss}} + \underbrace{\lambda\mathcal{L}_w}_{\text{black-box regularization}} + \underbrace{\mathcal{L}_S}_{\text{sign-loss [18] regularization (white-box)}}$$

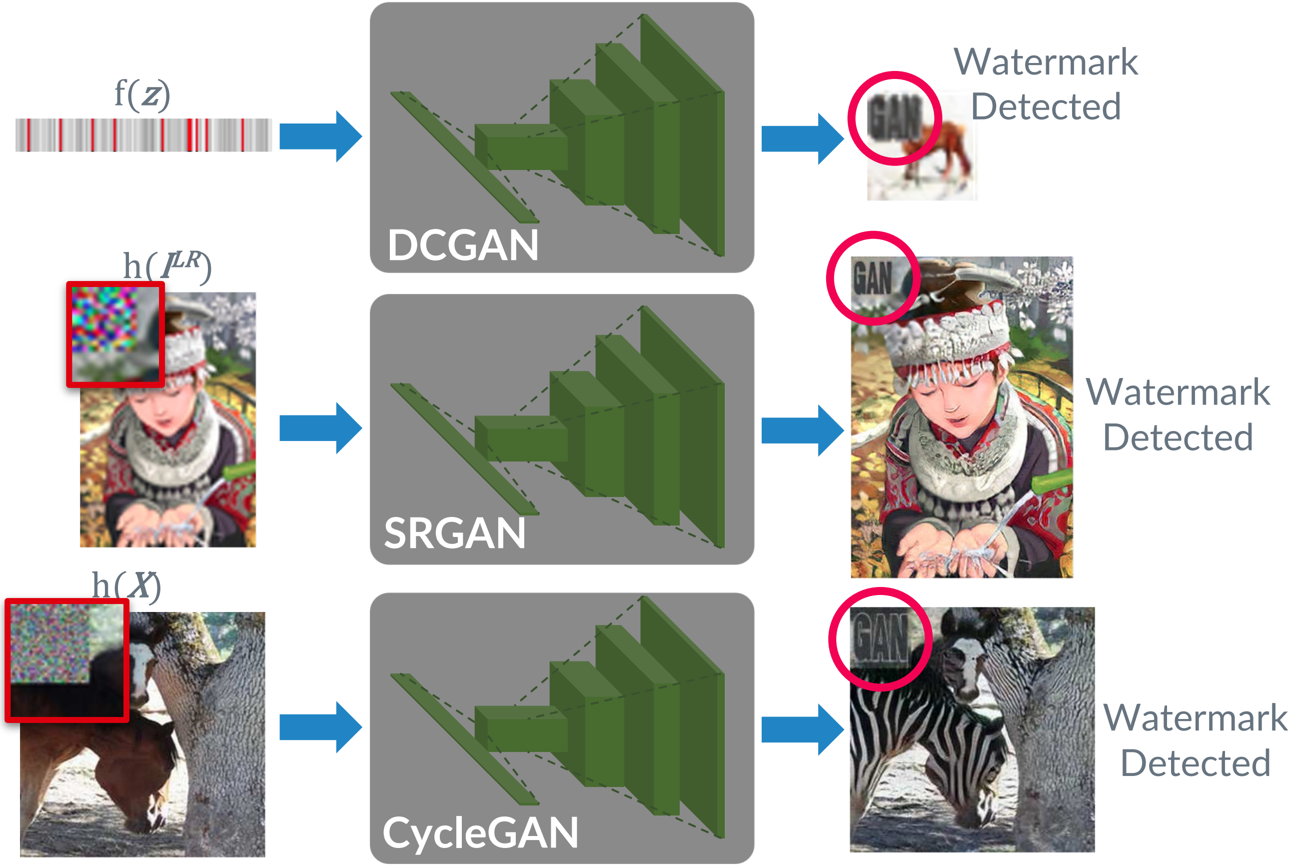
Generator loss
$X \in \{DCGAN, SRGAN, CycleGAN\}$

black-box
regularization

sign-loss [18]
regularization
(white-box)

Black-box watermarking in GANs

(Black-box) Watermark Verification



Some Visual Results

DCGAN on CIFAR10

Trigger Input Output

SRGAN on Set14

Trigger Input Output

CycleGAN on Cityscapes

Trigger Input Output

(Black-box) Watermark Verification

- *Quantitatively*, use Structural Similarity (SSIM) [9] to calculate score

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = \frac{(2\,\mu_x\,\mu_y + C_1)\,(2\,\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)\,(\sigma_x^2 + \sigma_y^2 + C_2)}$$

  between generated watermark & template watermark

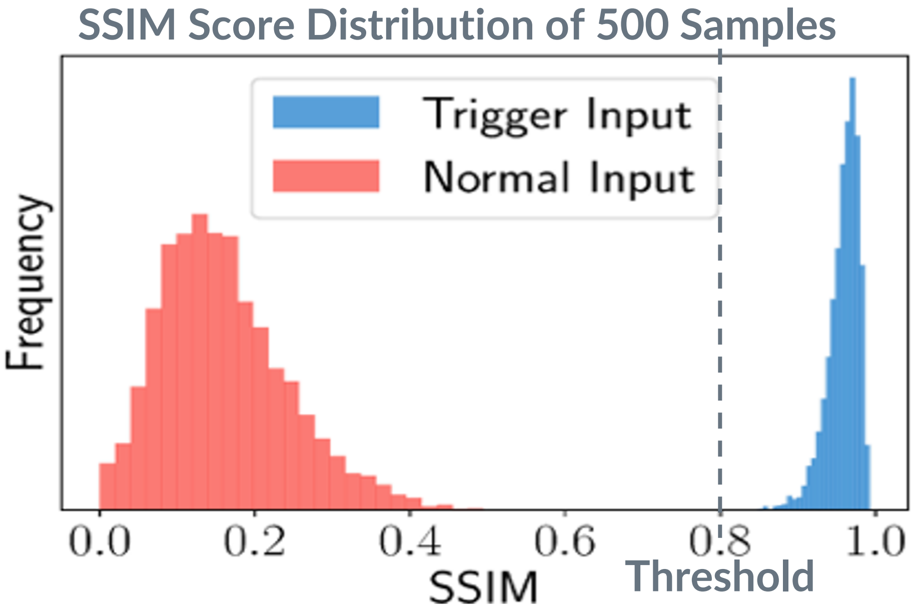
SSIM(

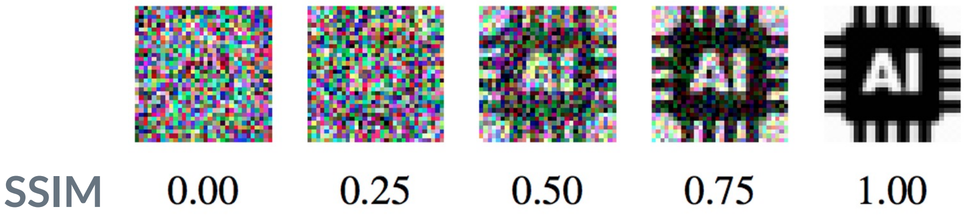

,


) = [0, 1] (score)

- If SSIM score > threshold: watermark detected

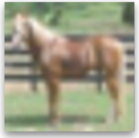
(Black-box) Watermark Verification





(Black-box) Watermarking in DCGAN

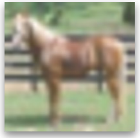
$$\mathcal{L}_w = 1 - \mathrm{SSIM}(G_{DC}(f(\boldsymbol{z})), g(G_{DC}(\boldsymbol{z}), \boldsymbol{WM}))$$

$G_{DC}($  $) =$ 

$$f(\boldsymbol{z}) = \boldsymbol{z} \circ \boldsymbol{b} + c(1 - \boldsymbol{b})$$

$\boldsymbol{z}$

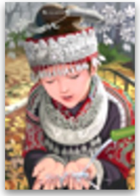
$f(\boldsymbol{z})$, c=-10

$g($  , **GAN** $) =$ 

$G_{DC}(\boldsymbol{z})$

(Black-box) Watermarking in SRGAN

$$\mathcal{L}_w = 1 - \mathrm{SSIM}(G_{SR}(h(X)), g(G_{SR}(X), WM))$$

$G_{SR}($  $) =$ 

h: 

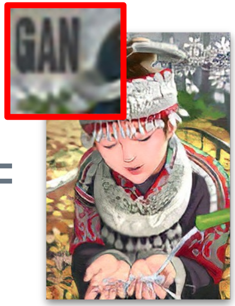
X

→ 

$h(X)$

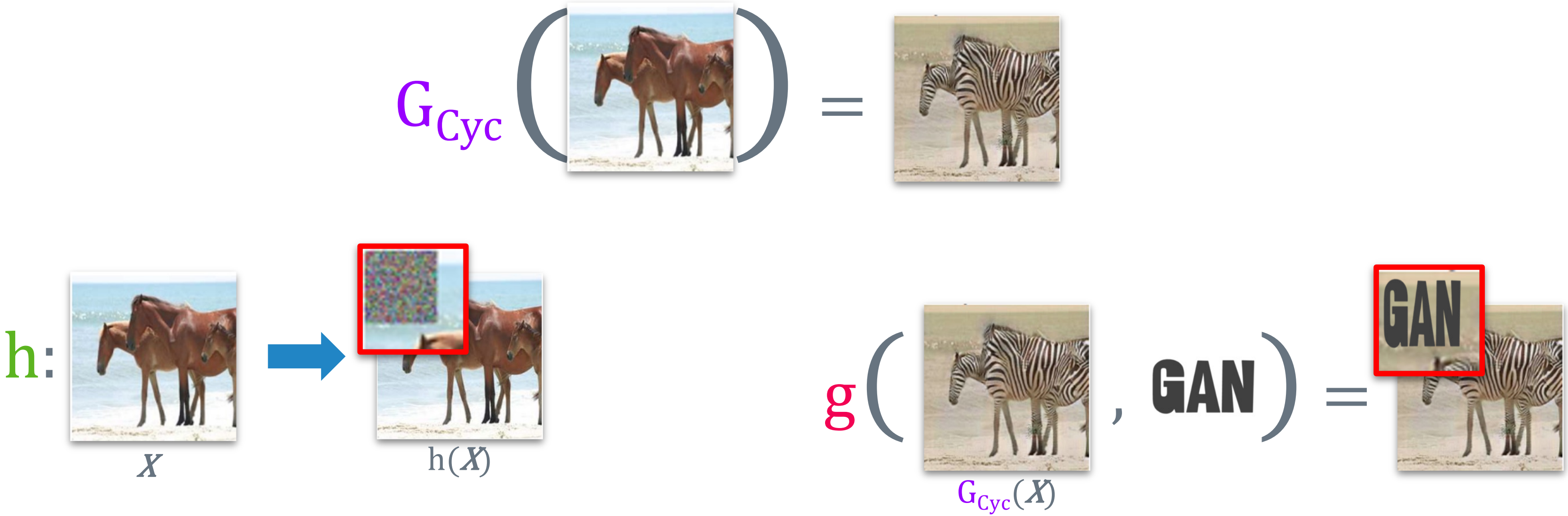
$g($ 

$G_{SR}(X)$

, GAN $) =$ 

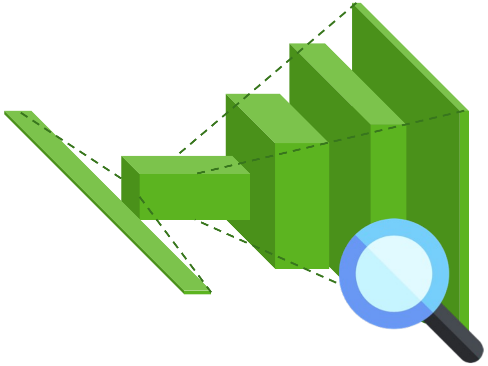
(Black-box) Watermarking in CycleGAN

$$\mathcal{L}_w = 1 - \mathrm{SSIM}(G_{Cyc}(h(X)),\ g(G_{Cyc}(X), WM))$$



White-box watermarking in GANs

(White-box) Watermark Verification



Extract Normalization Weights, γ

E			X			A			M			P			L			E		
γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit
-0.50	-	0	-0.22	-	0	-0.49	-	0	-0.24	-	0	-0.17	-	0	-0.44	-	0	-0.23	-	0
0.46	+	1	0.40	+	1	0.39	+	1	0.39	+	1	0.56	+	1	0.52	+	1	0.52	+	1
-0.42	-	0	-0.26	-	0	-0.44	-	0	-0.19	-	0	-0.17	-	0	-0.48	-	0	-0.28	-	0
-0.64	-	0	0.54	+	1	-0.17	-	0	-0.36	-	0	0.65	+	1	-0.62	-	0	-0.43	-	0
-0.25	-	0	0.43	+	1	-0.15	-	0	0.58	+	1	-0.53	-	0	0.37	+	1	-0.51	-	0
0.25	+	1	-0.14	-	0	-0.52	-	0	0.24	+	1	-0.56	-	0	0.49	+	1	0.22	+	1
-0.61	-	0	-0.45	-	0	-0.44	-	0	-0.18	-	0	-0.20	-	0	-0.47	-	0	-0.26	-	0
0.57	+	1	-0.34	-	0	0.35	+	1	0.55	+	1	-0.40	-	0	-0.55	-	0	0.32	+	1

(White-box) Watermarking GANs

- Define a sign watermark, $\boldsymbol{b} = \{b_k \mid b_k \in \{-1, 1\}\}$
  - Example: ASCII codes
- Modified from *sign loss* [1] to embed $\boldsymbol{b}$ into normalization weights, γ
- Sign loss enforces weights to take either positive or negative

Learnable Parameter:
Weight at k^th channel

$$\mathcal{L}_S = \sum_k \max(\gamma_0 - \boxed{\gamma_k}\, b_k, 0)$$

Constant,
default = 0.1

Target sign
at k^th channel

Fan *et al.* [5]

Fidelity

- Performance of original task is consistent
- Applying framework does not harm the performance

	Baseline	Proposed
DCGAN (FID)	26.54	26.27
SRGAN (PSNR/SSIM)	29.38/0.85	29.14/0.85
CycleGAN (Class IoU)	0.13	0.14

Watermark detection

- Black-box watermark is clearly visible (SSIM score > threshold)
- White-box watermark is 100% detected (0 bit error)

	black-box (SSIM)	white-box
DCGAN	0.97	100%
SRGAN	0.93	100%
CycleGAN	0.90	100%

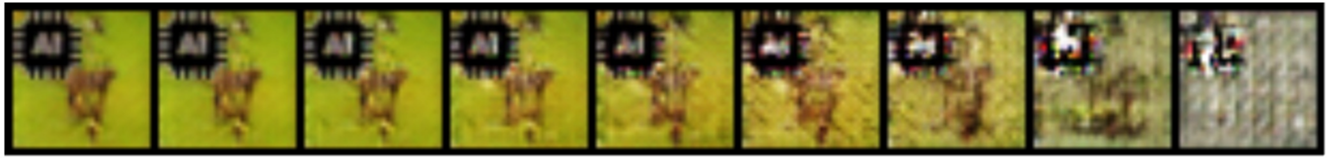
Fine-tuning

- Finetune GANs using training data, without regularization terms
- Both black-box & white-box watermark persist after fine-tuning

	Before		After	
	black-box (SSIM)	white-box	black-box (SSIM)	white-box
DCGAN	0.97	100%	0.96	100%
SRGAN	0.93	100%	0.83	100%
CycleGAN	0.90	100%	0.85	100%

Pruning

- The black-box & white-box watermark **persist** before the model is excessively pruned



% pruned	10	20	30	40	50	60	70	80	90
black-box (SSIM)	0.958	0.949	0.924	0.889	0.836	0.760	0.606	0.389	0.176
white-box	100%	100%	100%	100%	100%	100%	100%	100%	100%

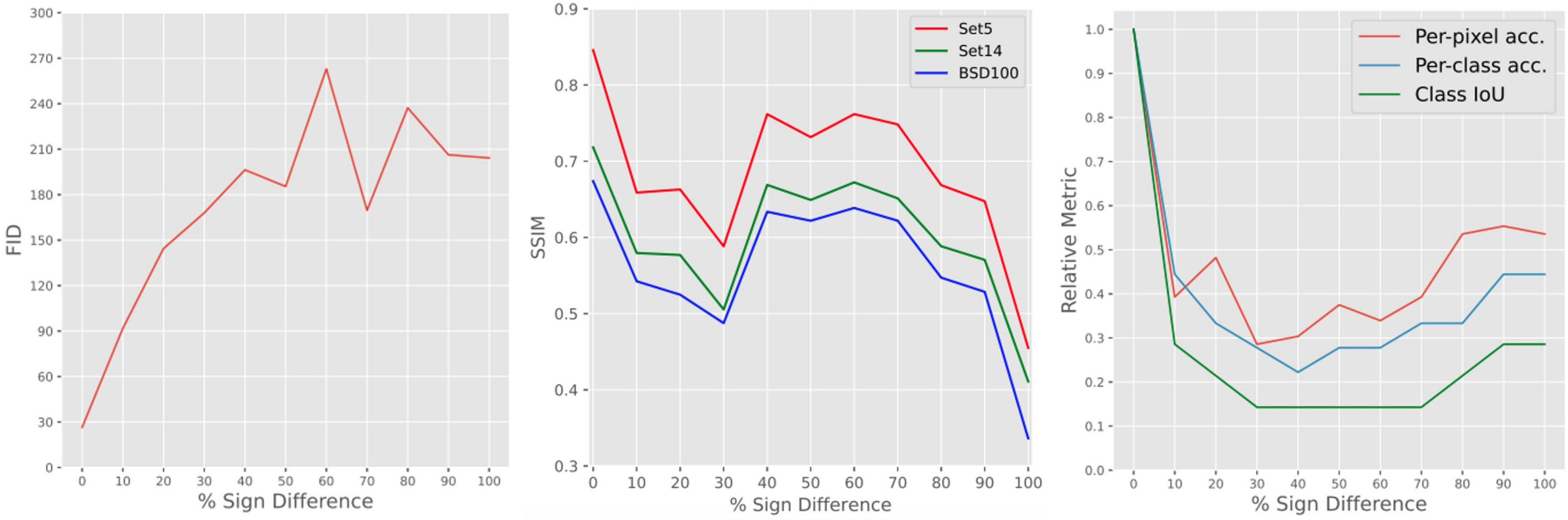
Overwriting

- Using the same watermarking method, but using new watermark
- Black-box watermark removed, White-box watermark persists

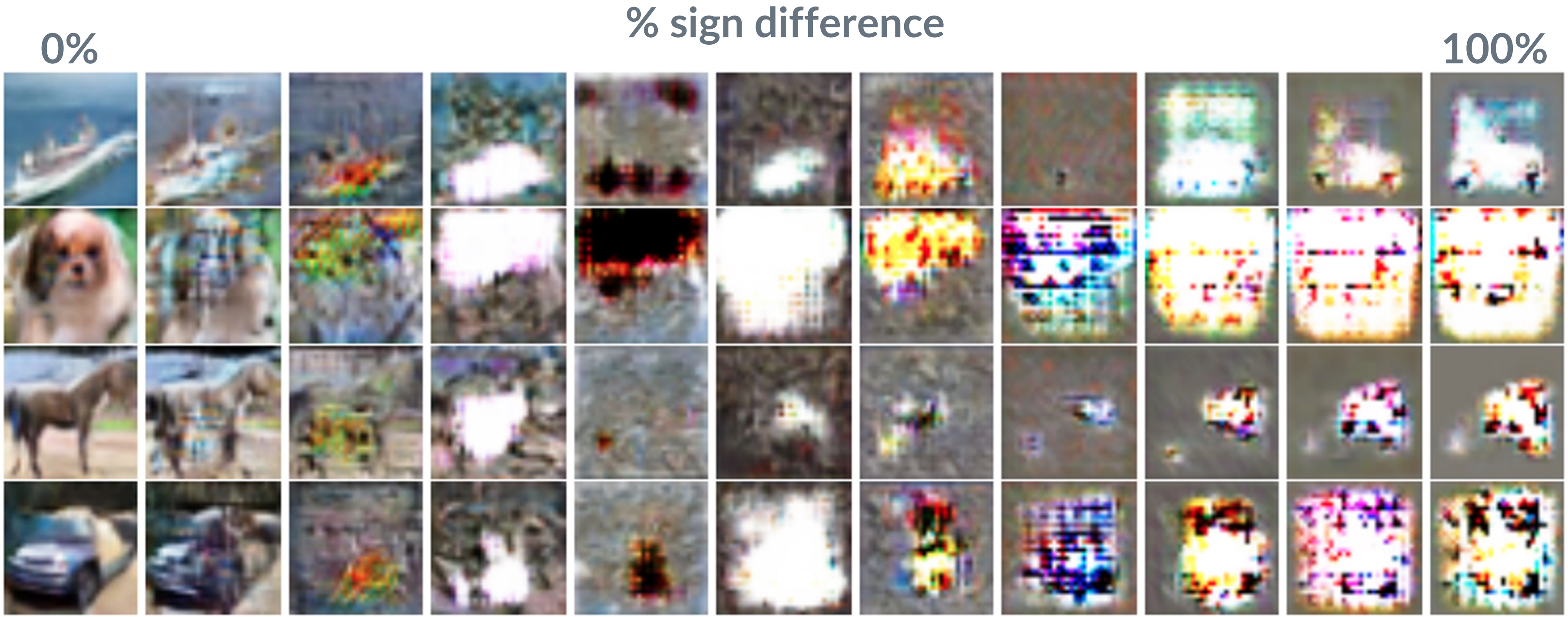
	Before		After	
	black-box (SSIM)	white-box	black-box (SSIM)	white-box
DCGAN	0.97	100%	0.49	100%
SRGAN	0.93	100%	0.17	100%
CycleGAN	0.90	100%	0.15	100%

Ambiguity Attack

- Change the sign of normalization weight, γ
- Slight changes in sign causing very poor performance



Ambiguity Attack



Key Takeaway

- Previous works mainly on CNN classification works
- Proposed **black-box + white-box protection framework for GANs**
- Framework **does not change network architecture**
- Applied to DCGAN, SRGAN & CycleGAN **without affecting performance**
- Framework is robust against removal attack and ambiguity attack

Paper & Code

arXiv

https://arxiv.org/abs/2102.04362

GitHub

https://github.com/dingsheng-ong/ipr-gan

sheng970303@gmail.com

References

1. Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In *NeurIPS*, pages 4714–4723, 2019.
2. Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, pages 269–277,2017.
3. Y Adi, C Baum, M Cisse, B Pinkas, and J Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX)*, 2018.
4. Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (ASIACCS)*, pages159–172, 2018.
5. Bita Darvish Rohani, Huili Chen, and Farinaz Koushanfar. DeepSigns: A Generic Watermarking Framework for IP Protection of Deep Learning Models.*arXiv:1804.00750*, April 2018.
6. Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors,*4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4,2016, Conference Track Proceedings, 2016.*
7. Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken, Alykhan Tejani, JohannesTotz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI,USA, July 21-26, 2017*, pages 105–114. IEEE Computer Society, 2017.
8. Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice,Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society, 2017.
9. Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity.IEEE Transactions on Image Processing, 13(4):600–612, 2004.

Thank you!