Adversarial Machine Learning and Beyond

Philipp Benz
https://phibenz.github.io

Chaoning Zhang
https://chaoningzhang.github.io

Agenda

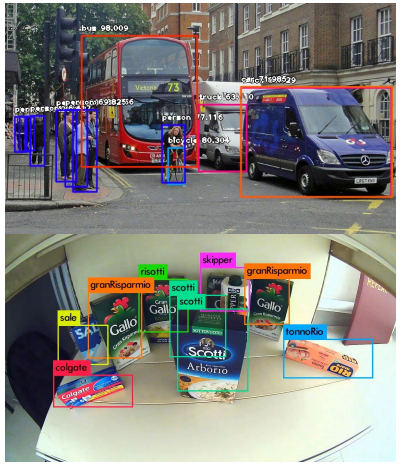
Overview of adversarial machine learning:

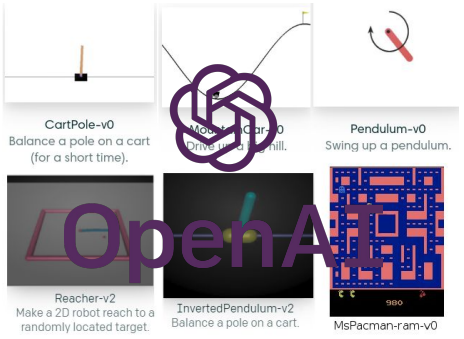
1. Adversarial Examples
2. Adversarial attack
3. Adversarial defense

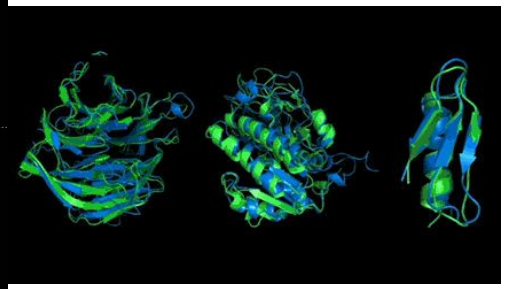
Our work:

4. UAP with class discrimination
5. Understanding UAP
6. Universal deep hiding
7. Towards a unified perspective

Deep Learning is Awesome







Overview of Adversarial Machine Learning

Adversarial machine learning: a technique attempting to fool models through malicious input.

1. Practical relevance: Can we trust the model for security sensitive applications?

2. Theoretical relevance: Understanding adversarial examples might give us insights about how deep neural networks work

The Basics

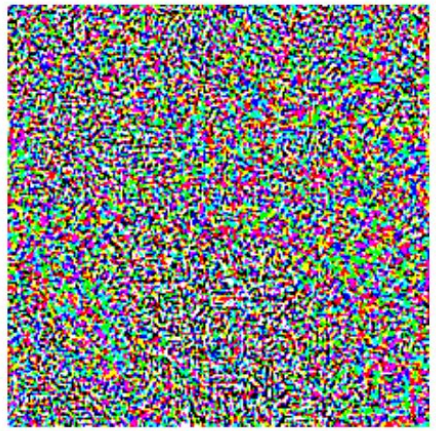
Adversarial Examples

Deep Neural Networks are sensitive to small perturbations in the image, which can lead to misclassifications. These changes are mostly imperceptible for human observers.

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

- Perturbations of small magnitude
- Specially crafted through optimization techniques
- Imperceptible for humans

[1] Explaining and Harnessing Adversarial Examples; ICLR 2015

Objective

Adversarial Attack: Finding a small perturbation, to misclassify a sample

$$C(x+\delta) \neq C(x)$$

Misclassification

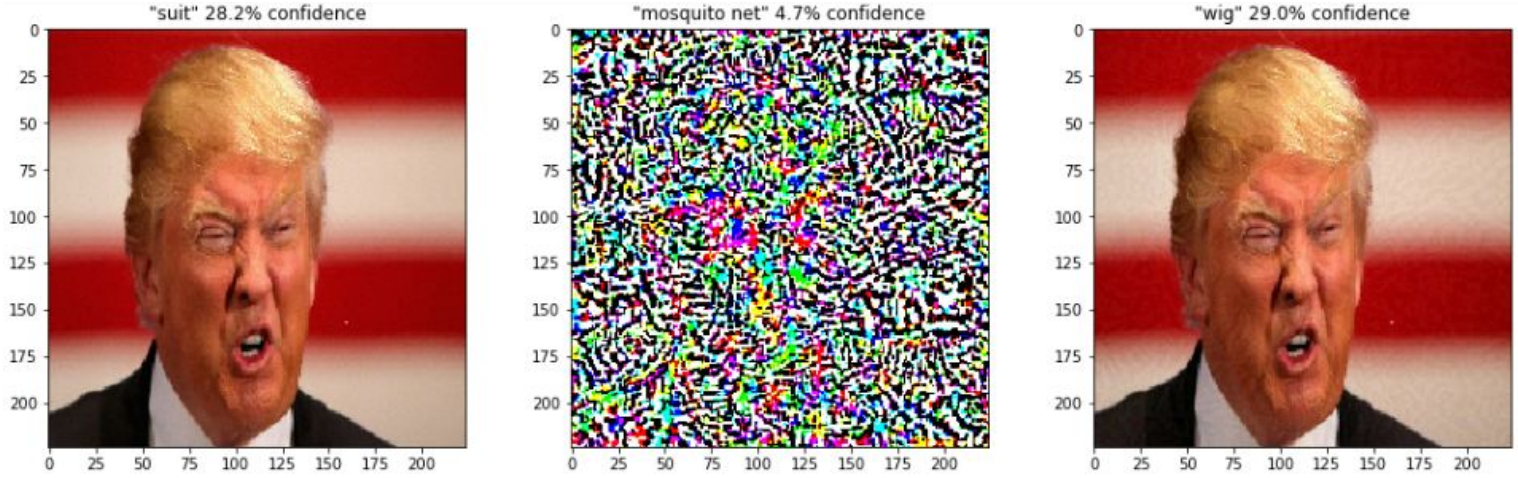
subject to $D(x, x+\delta) \leq \epsilon$

The perturbation is smaller than magnitude ϵ
Some distance metric L1, L2, Linf

$$x+\delta \in [0,1]$$

Obey image range

x δ $x+\delta$



Distance metrics

$$\delta = x - x'; \qquad ||\delta||_p = \left(\sum_{i=1}^{n} |\delta_i|^p \right)^{\frac{1}{p}}$$

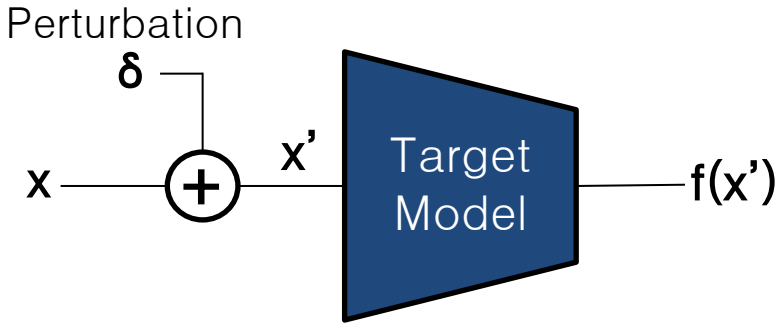
L_0 Measures the number of coordinates i such that $x_i \neq x'_i$
Corresponds to the number of pixels that have been altered in an image.

L_2 Measures the standard Euclidean (root-mean-square) distance between x and x'. The L2 distance can remain small when there are many small changes to many pixels.

L_∞ Measures the maximum change to any of the coordinates: $\|x - x'\|_\infty = \max(|x_1 - x'_1|, \ldots, |x_n - x'_n|)$
For images, we can imagine there is a maximum budget, and each pixel is allowed to be changed by up to this limit, with no limit on the number of pixels that are modified.

L0 metric is rarely used. More and more papers are adopting the Linf metric for evaluation.

Adversarial Examples: Attack categories



White-Box Attack An attacker has full knowledge about the target	**Black-Box** Attack An attacker has no knowledge about the target
Untargeted Attack The attack succeeds, if the target model makes a misclassification	**Targeted** Attack The attack succeeds, if the target model makes a misclassification which was specified beforehand

White-box vs. Black-box

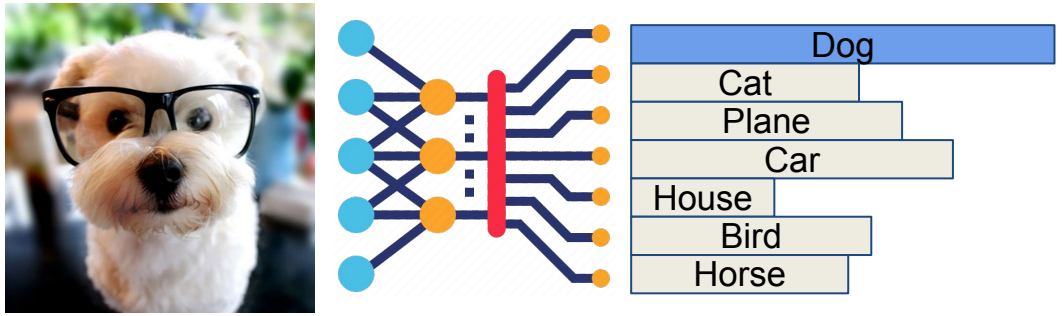
Adversary has **FULL** knowledge about:

- Model e.g.
  - type of neural network
  - number of layers
- Training algorithm e.g
  - SGD, ADAM etc.
  - Learning Rate
- Training data
- Model parameters

Adversary has **NO** knowledge about the model, training algorithm, training data, model parameters etc.
Commonly it is only assumed that the adversary can query the model
Note: Different works define their “black-box” slightly different, so stay critical!

Adversary has **SOME** information.
E.g. the adversary knows the network architecture, but no information about the exact parameters.

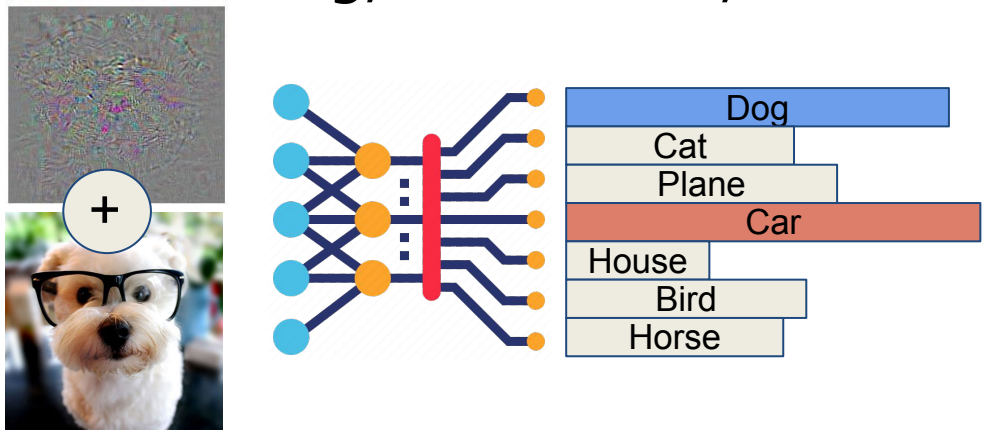
Untargeted vs. Targeted



Untargeted

The misclassified class can be **any other** than the currently classified class

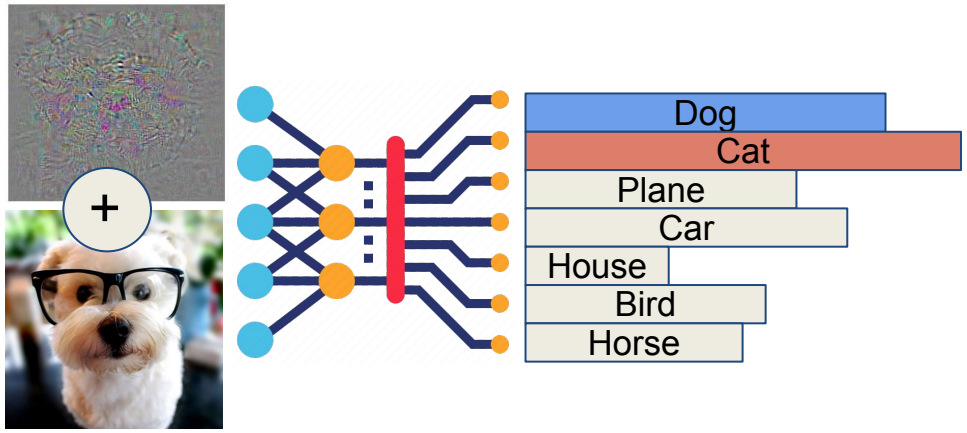
In this example: any class other than dog, such as cat, car...



Targeted

The misclassified class can **only be** the target class

In this example: target class = Cat



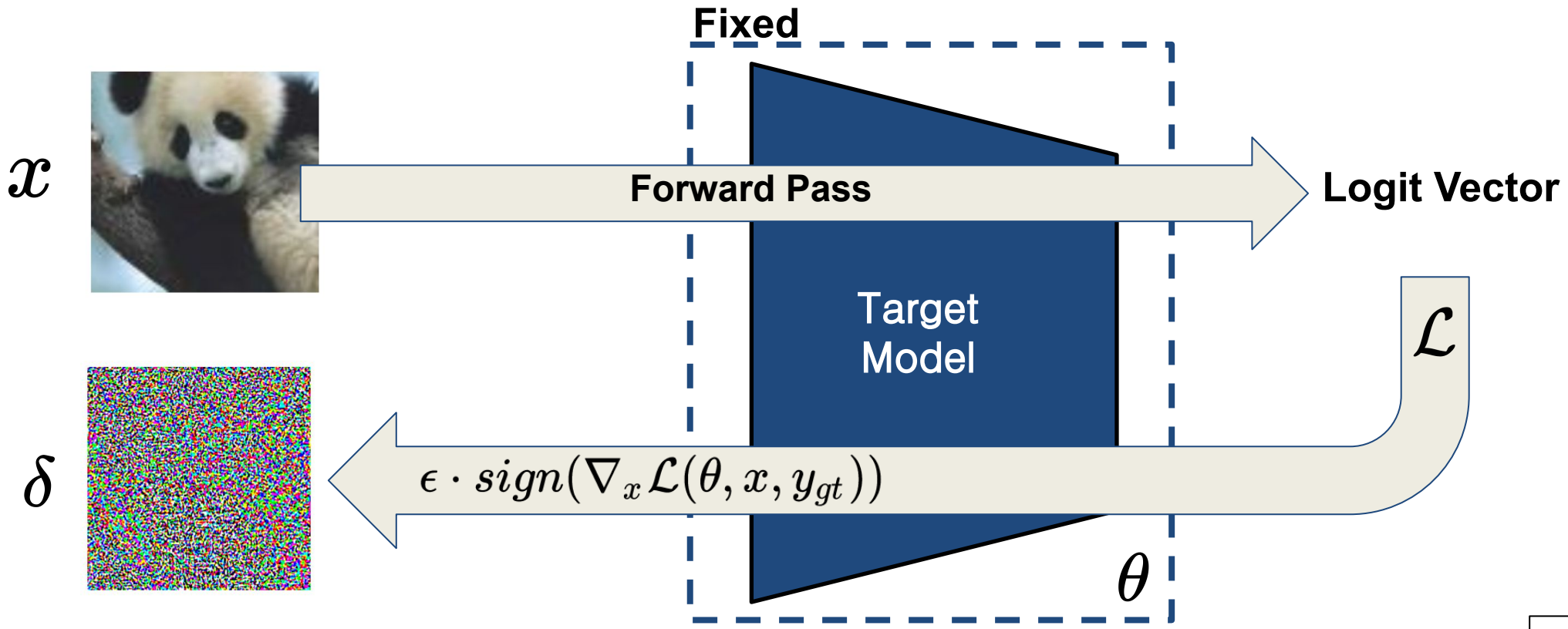
Adversarial Attacks

The FGSM attack

Fast Gradient Sign Method (FGSM) [1]
Stepping one step of step size ϵ into the gradient direction

$$x' = x + \epsilon \cdot sign(\nabla_x \mathcal{L}(\theta, x, y_{gt}))$$

Pro: Fast Con: Less effective $\mathcal{L} = CE$



ϵ: magnitude

[1] Explaining and Harnessing Adversarial Examples; ICLR 2015

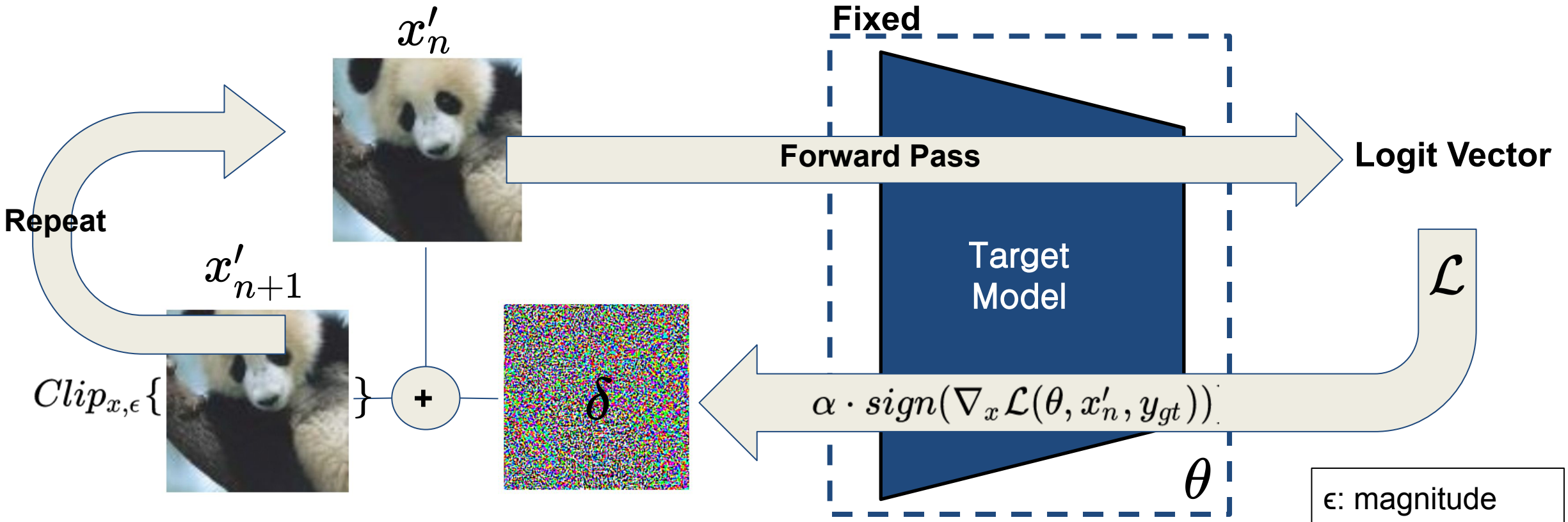
The PGD-attack

Projected Gradient Descent (PGD)

Iteratively apply FGSM with step size α

$$x'_0 = x; \quad x'_{n+1} = Clip_{x,\epsilon}\{x'_n + \alpha \cdot sign(\nabla_x \mathcal{L}(\theta, x'_n, y_{gt}))\}$$

Pro: Effective Con: iterative, thus time-consuming



ϵ: magnitude
α: step size

[1] Towards Deep Learning Models Resistant to Adversarial Attacks; ICLR 2018

Attack Techniques

In the white-box setting the gradients of the model can be used to construct the adversarial perturbation

Fast Gradient Sign Method (FGSM) [1]
Stepping one step of step size ϵ into the gradient direction

$$\mathcal{L} = CE$$

$$x' = x + \epsilon \cdot sign(\nabla_x \mathcal{L}(\theta, x, y_{gt}))$$

Pro: Fast Con: Less effective

Targeted FGSM [2]
Stepping one step of step size ϵ into the gradient direction of target class t

$$x' = x - \epsilon \cdot sign(\nabla_x \mathcal{L}(\theta, x, t))$$

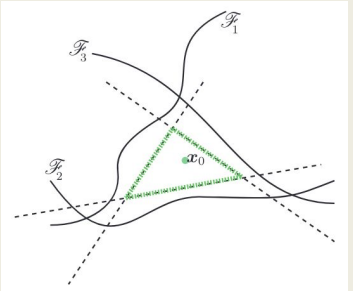
Compared with non targeted FGSM, the targeted one flips the loss sign and replace gt with t

Projected Gradient Descent (PGD) [2,3]
Iteratively apply FGSM with step size α

$$x'_0 = x; \quad x'_{n+1} = Clip_{x,\epsilon}\{x'_n + \alpha \cdot sign(\nabla_x \mathcal{L}(\theta, x'_n, y_{gt}))\}$$

Pro: Effective Con: iterative, thus time-consuming

DeepFool [5]
Geometry-inspired approach to push a sample over the **decision boundary**



Linear approximation of DNNs decision boundary

Pro: Geometry-inspired Con: Slow

Carlini & Wagner attack (C&W) [4]
Minimal perturbation to make the highest (h) logit Z() lower than the second highest one

$$\mathcal{L} = ||\delta||_2^2 + c \cdot f(x')$$

$$f(x') = \max(\max_{i \neq h} Z_i(\theta, x') - Z_h(\theta, x'), -\kappa)$$

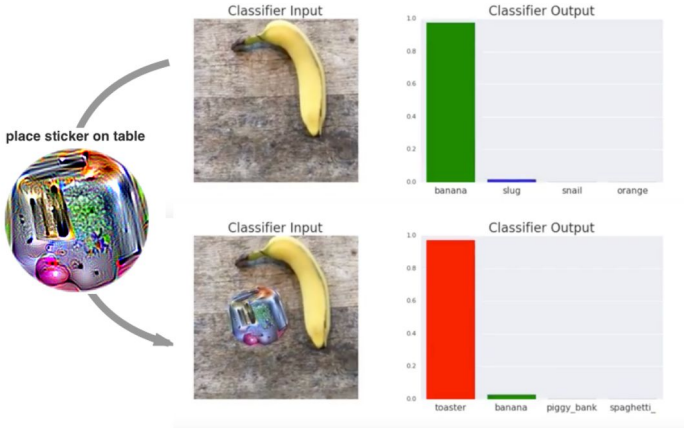
Pro: Minimal perturbation thus least visible Con: Relatively more complex for minimizing the perturbation and loss function

[1] Explaining and Harnessing Adversarial Examples; ICLR 2015
[2] Adversarial examples in the physical world; ICLR Workshop 2017
[3] Towards Deep Learning Models Resistant to Adversarial Attacks; ICLR 2018
[4] Towards Evaluating the Robustness of Neural Networks; Symposium on Security and Privacy (SP) 2017
[5] DeepFool: a simple and accurate method to fool deep neural networks; CVPR 2016

Real-world Attacks

3D-printed adversarial objects [1]

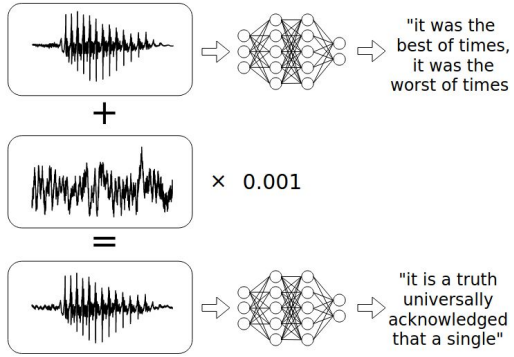
Adversarial Patch [2]



Black-Box attacks on web applications [3]

original image	true label	Clarifai.com results of original image	target label	targeted adversarial example	Clarifai.com results of targeted adversarial example
	viaduct	bridge, sight, arch, river, sky	window screen		window, wall, old, decoration, design
	hip, rose hip, rosehip	fruit, fall, food, little, wildlife	stupa, tope		Buddha, gold, temple, celebration, artistic
	dogsled, dog sled, dog sleigh	group together, four, sledge, sled, enjoyment	hip, rose hip, rosehip		cherry, branch, fruit, food, season
	pug, pug-dog	pug, friendship, adorable, purebred, sit	sea lion		sea seal, ocean, head, sea, cute

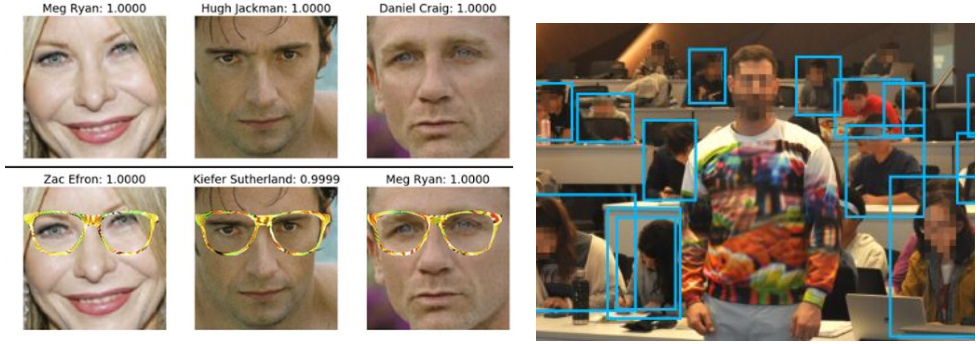
Audio Adversarial Examples [5]



Attacks on Traffic Signs [4]

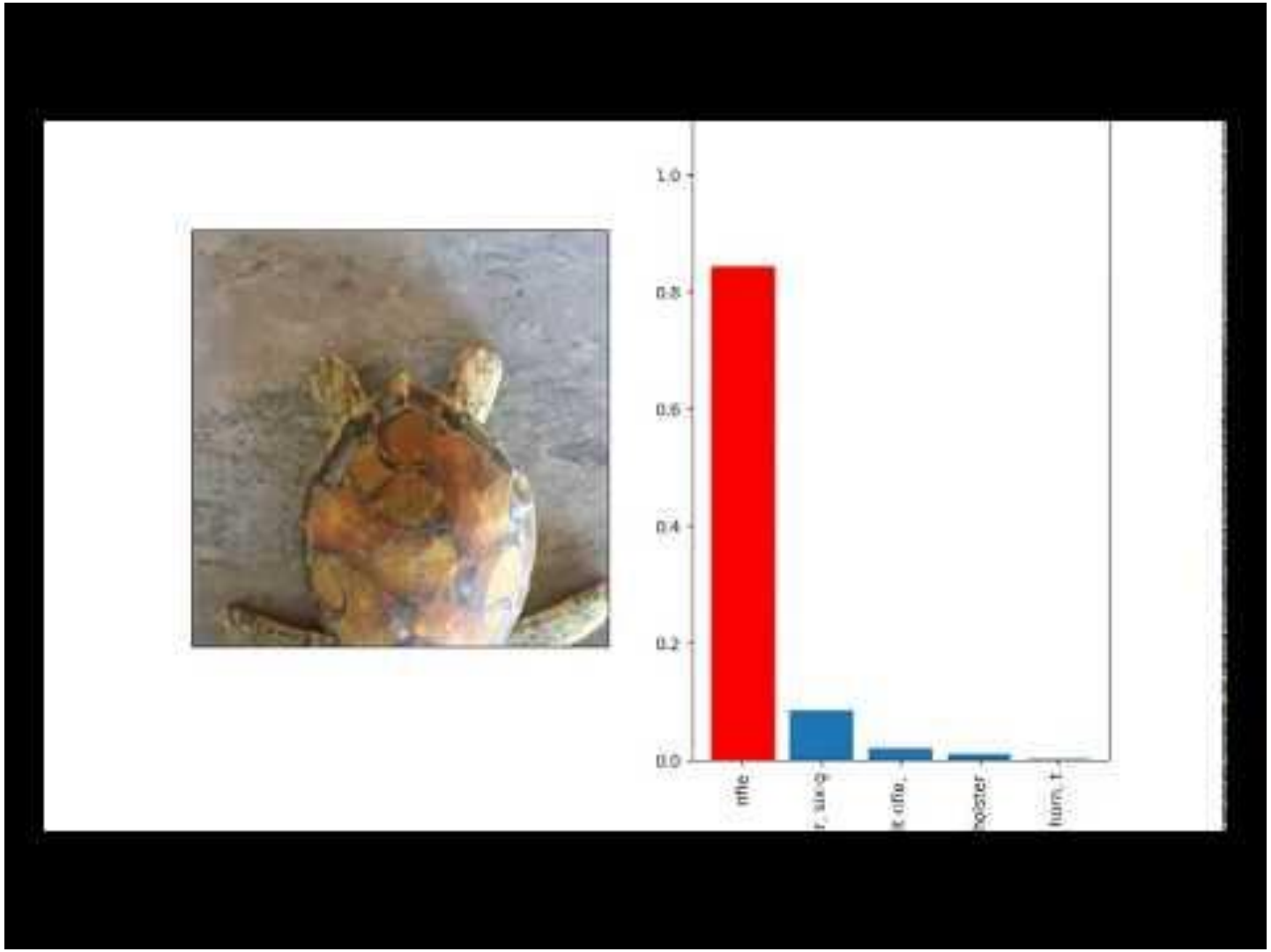
Distance/Angle	Subtle Poster	Subtle Poster Right Turn	Camouflage Graffiti	Camouflage Art (LISA-CNN)
5′ 0°				
5′ 15°				
10′ 0°				
10′ 30°				
40′ 0°				
Targeted-Attack Success	100%	73.33%	66.67%	100%

Adversarial Wearables [6,7]



[1] Synthesizing Robust Adversarial Examples; ICML 2018
[2] Adversarial Patch; 2017
[3] Delving into transferable adversarial examples and black-box attacks; ICLR 2017
[4] Robust Physical-World Attacks on Deep Learning Visual Classification; CVPR 2018
[5] Audio Adversarial Examples: Targeted Attacks on Speech-to-Text; 2018
[6] Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors; 2019

3D-printed adversarial objects



[1] Synthesizing Robust Adversarial Examples; ICML 2018

Adversarial Defenses

Attack & Defense: A two-player game

One player aims to attack the model; the other one tries to defend the model.
Who will win this game? It depends on the **rules** of the game.

The attacker knows what the defender is doing (Attacker wins)

The attacker does not know what the defender is doing (Defender wins)

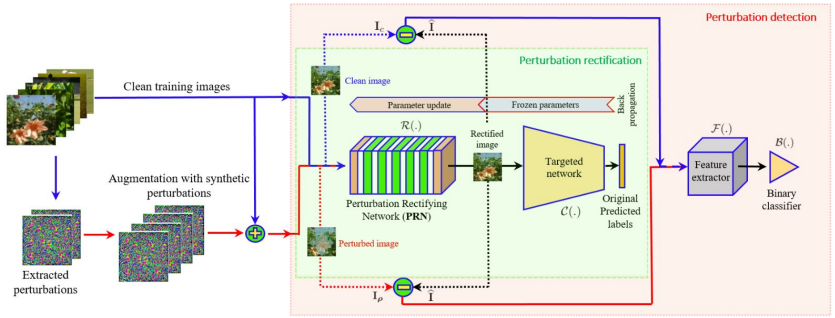
Defenses

1 Easy defense: The attacker does not know what the defender is doing

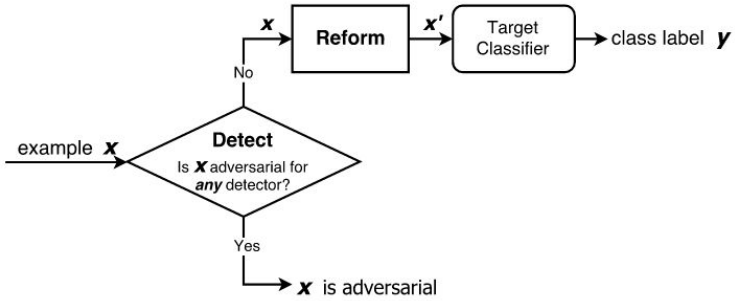
Motivation: Denoising the image

Network Modification
Adding parts to the network or modifying the layers for defense

Adding "pre-input" layers trained [5]



MagNet [6]

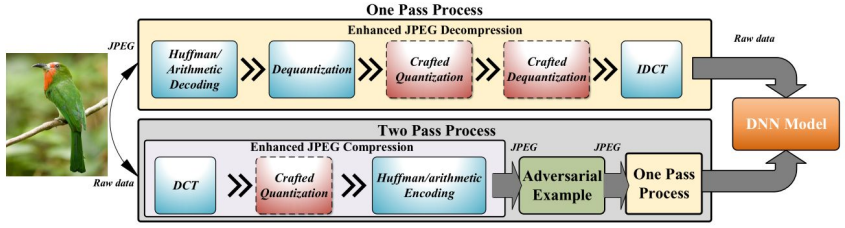


2 Hard defense: The attacker knows what the defender is doing

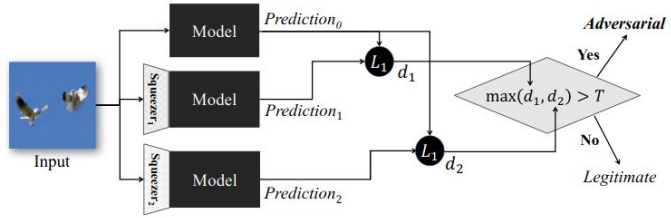
Motivation: Obfuscated gradients

Input Modification
Manipulate the input to deweaponize the adversarial example

JPEG Compression [3]



Feature Squeezing [4]



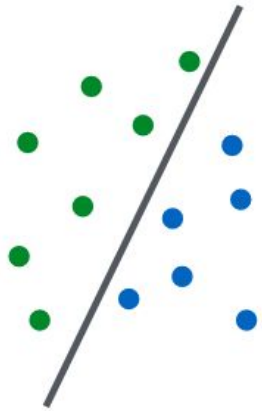
[1] Explaining and harnessing adversarial examples; ICLR 2015
[2] Generative Adversarial Trainer: Defense to Adversarial Perturbations with GAN; 2017
[3] Feature Distillation: DNN-Oriented JPEG Compression Against Adversarial Examples; 2018
[4] Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks; NDSS 2018
[5] Defense against Universal Adversarial Perturbations; CVPR 2018
[6] MagNet: a Two-Pronged Defense against Adversarial Examples; CCS 2017

Adversarial Training

Training a model (iteratively) with adversarial examples makes a model more robust.
So far adversarial training is the only method that is proven to really robustify a model.

Normal Network Training

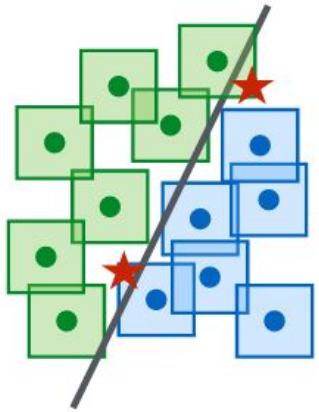
$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[L(x,y,\theta)\right]$$



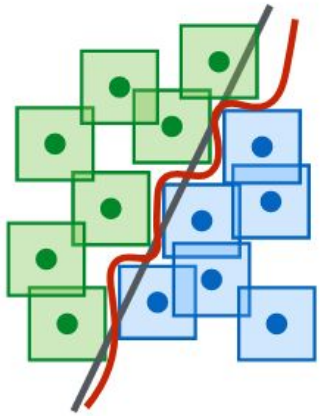
Set of points that can be easily separated with a simple linear decision boundary

Adversarial Training

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right]$$



The simple decision boundary does not separate the L∞-balls (squares) around the data points → adversarial examples (red stars) will be misclassified



Separating the L∞-balls requires a more complicated decision boundary. The resulting classifier is robust to adversarial examples with bounded L∞-norm perturbations.

[1] Explaining and harnessing adversarial examples; ICLR 2015
[2] Towards Deep Learning Models Resistant to Adversarial Attacks; ICLR 2018

Robustness may be at odds with accuracy

Trade-off between **standard accuracy** & **adversarially robust accuracy**

[1] Robustness may be at odds with accuracy; ICLR 2019

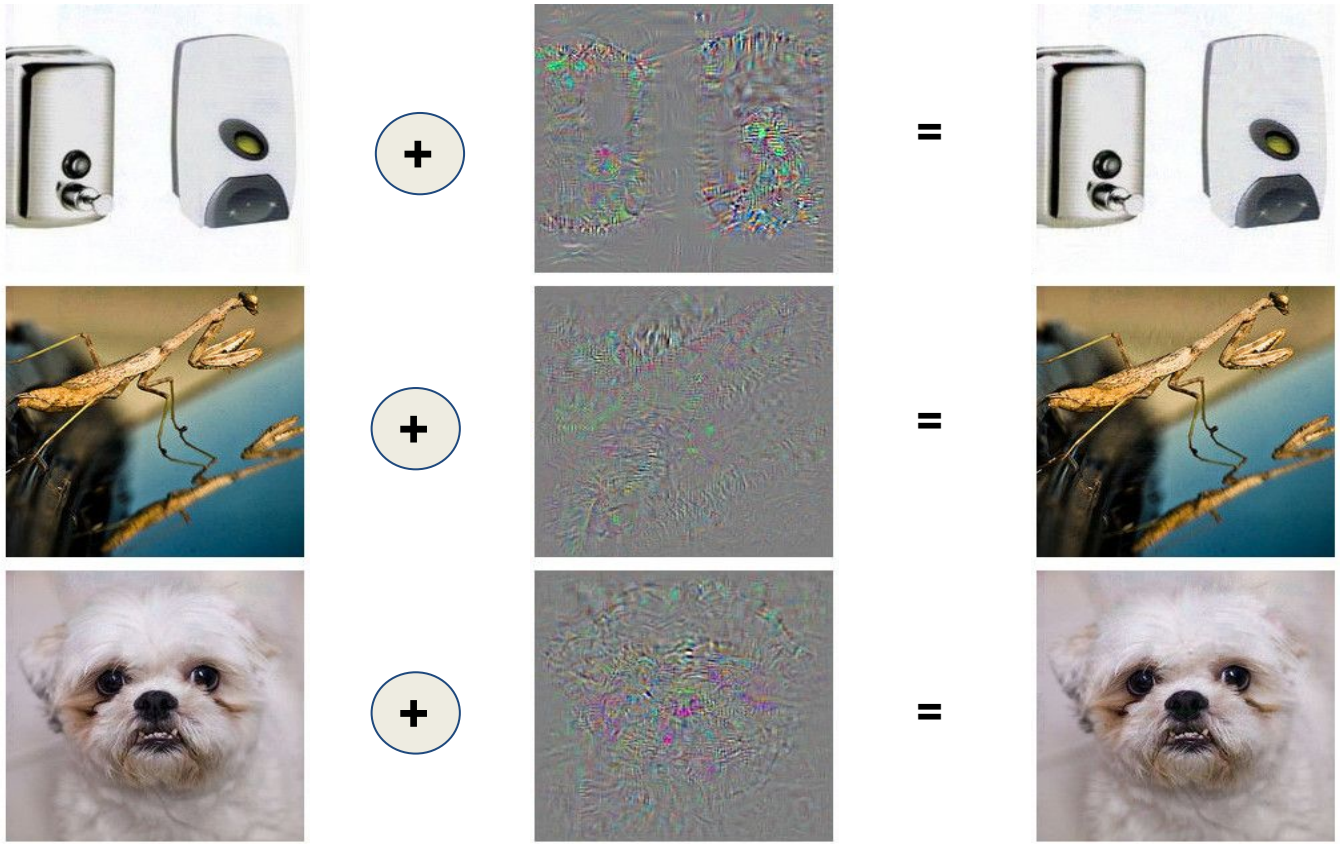
Universal Adversarial Perturbations

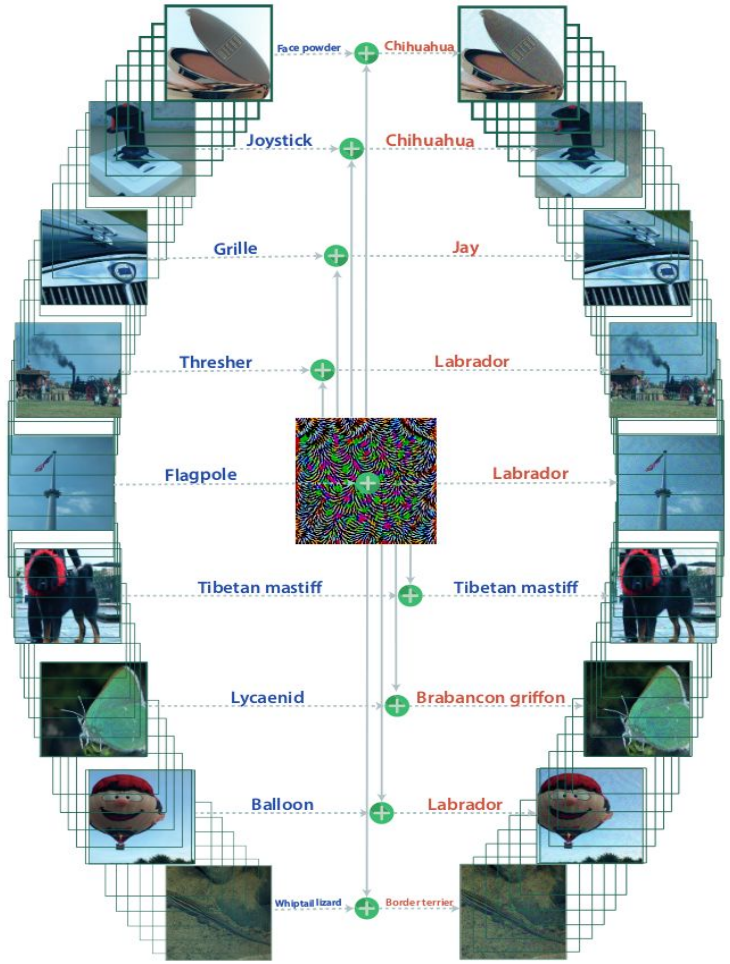
Image-dependant vs. image-agnostic

Image-dependant attacks
One perturbation is crafted to attack **one** specific image

Image-agnostic (universal) attacks
One perturbation is crafted to attack a **set** of images
UAP will be discussed in more detail again

Image | **Image-dependant perturbation** | **Adversarial Example**



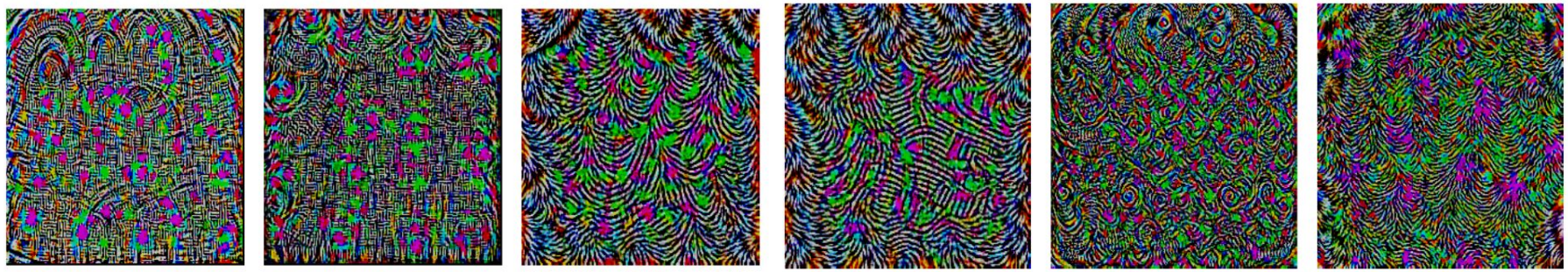


[1] Intriguing properties of neural networks; Szegedy, Zaremba, Sutskever, Bruna, Erhan, Goodfellow, Fergus; ArXiv 2013
[2] Universal adversarial perturbations; Moosavi-Dezfooli, Fawzi, Fawzi, Frossard; CVPR 2017

Qualitative visualization

Images + UAP

Amplified UAP



[1] Universal Adversarial Perturbations; CVPR 2017

Universal adversarial perturbations (UAPs)

One perturbation to fool most of the data samples

Objective

$C(x+\delta) \neq C(x)$ for "most" $x \sim \mathcal{X}$

Fool most of the samples with a single perturbation

subject to $||\delta||_p \leq \epsilon$

The perturbation is smaller than magnitude ϵ

Classical DeepFool-based UAP

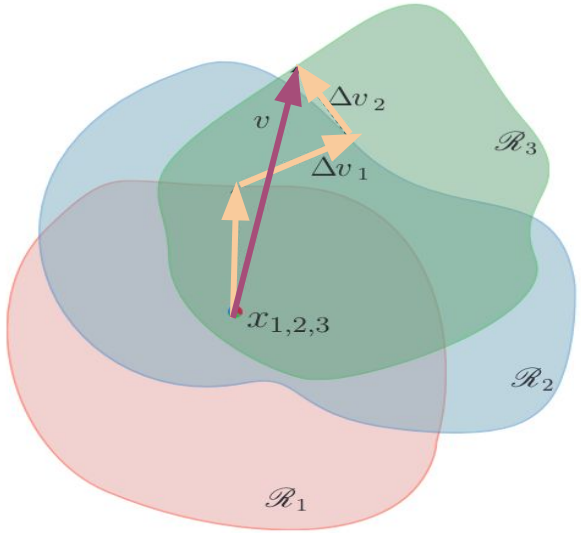


Figure 1: Schematic representation of the algorithm in [1] to compute universal perturbations.

Algorithm:

- Craft single perturbation (via DeepFool [3]) to let one sample cross the decision boundary
- Iterate this process for different samples to aggregate the universal adversarial perturbation.

Algorithm 1 Computation of universal perturbations.

input: Data points X, classifier $\hat{k}$, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
output: Universal perturbation vector v.
Initialize $v \leftarrow 0$.
while $\text{Err}(X_v) \leq 1 - \delta$ **do**
  for each datapoint $x_i \in X$ **do**
    if $\hat{k}(x_i + v) = \hat{k}(x_i)$ **then**
      Compute the *minimal* perturbation that sends $x_i + v$ to the decision boundary:

$$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

      Update the perturbation:

$$v \leftarrow \mathcal{P}_{p,\xi}(v + \Delta v_i).$$

    end if
  end for
end while

[1] Universal adversarial perturbations; Moosavi-Dezfooli, Fawzi, Fawzi, Frossard; CVPR 2017
[2] Analysis of universal adversarial perturbations; Moosavi-Dezfooli, Fawzi, Fawzi, Frossard, Soatto; ArXiv 2017
[3] DeepFool: a simple and accurate method to fool deep neural networks; Moosavi-Dezfooli, Fawzi, Frossard; CVPR 2016

Practical Relevance of UAPs

Image-Dependent Adversarial Examples	Universal Adversarial Examples
Have to be crafted on the spot for each sample (during inference)	Can be crafted beforehand Only a simple summation is required during inference

UAPs are better suited for “real-world” attacks

Our Research beyond classical UAP

1. The UAP attacks samples from every class without intended discrimination, which can easily cause suspicion

2. The UAP generation relies on the original training data & The reason for the existence of UAPs is still not well understood

3. Can the insight of UAPs be applied to the field of data hiding?

Our Research beyond classical UAP

1. The UAP attacks samples from every class without intended discrimination, which can easily cause suspicion
   → CD-UAP: Class-Discriminative Universal Adversarial Perturbations (AAAI 2020)
   → Double targeted universal adversarial perturbations (ACCV 2020)

2. The UAP generation relies on the original training data & The reason for the existence of UAPs is still not well understood
   → Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations (CVPR 2020)

3. Can the insight of UAPs be applied to the field of data hiding?
   → UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging (NeurIPS 2020)
   → Universal Adversarial Perturbations Through the Lens of Deep Steganography: Towards A Unified Fourier Perspective (AAAI 2021)

CD-UAP: Class-Discriminative Universal Adversarial Perturbations (AAAI 2020)

Chaoning Zhang*, Philipp Benz*, Tooba Imtiaz, In-So Kweon
* indicates equal contribution

Korea Advanced Institute of Science and Technology (KAIST)

CD-UAP: Class-Discriminative Universal Adversarial Perturbations

An attack method to generate a universal perturbations that fool a target network to misclassify only a chosen group of classes, while having limited influence on the remaining classes.

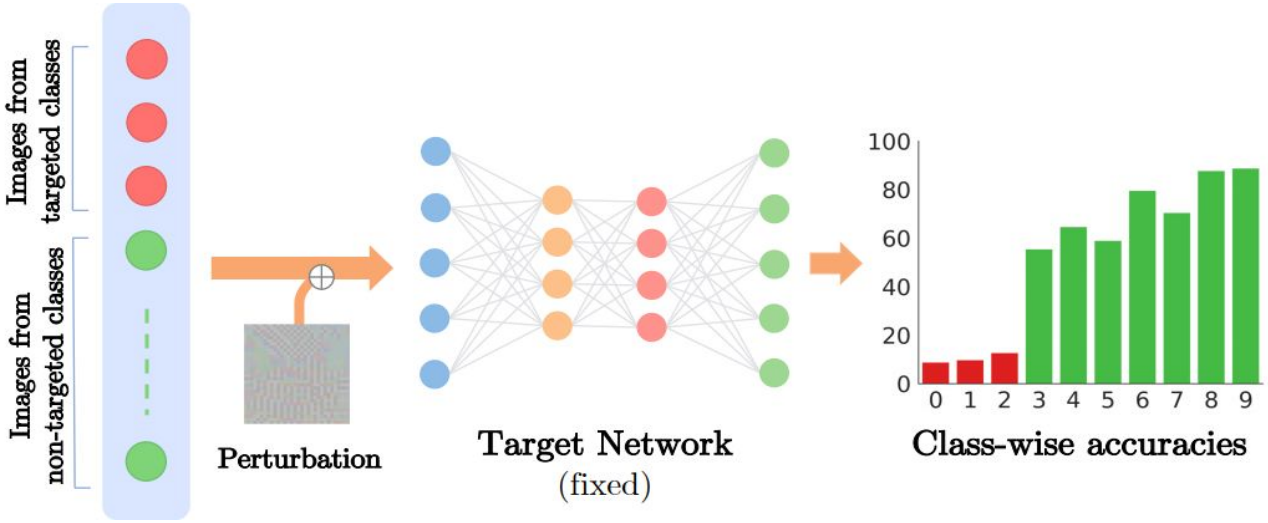


Figure 1: Class Discriminative Universal Adversarial Perturbation (CD-UAP). After adding a single perturbation, model performance on a subset of classes is significantly reduced, while the influence on the non-targeted classes is limited.

$$C(x_t + \delta) \neq C(x_t) \quad \text{for most } x_t \sim \mathcal{X}_t$$

Misclassify most samples from targeted classes

$$C(x_{nt} + \delta) = C(x_{nt}) \quad \text{for most } x_{nt} \sim \mathcal{X}_{nt}$$

Keep original classification for most samples from non-targeted classes

$$||\delta||_p \leq \epsilon$$

The perturbation is smaller than magnitude ϵ

[1] CD-UAP: Class-Discriminative Universal Adversarial Perturbations; Zhang*, Benz*, Imtiaz, Kweon; AAAI 2020

CD-UAP: Class-Discriminative UAP

Evaluation metric: Absolute Accuracy Drop (AAD) for both targeted classes and non-targeted classes.

$\Delta_{AAD}=AAD_t - AAD_{nt}$, the higher the better.

Table 2: Experiments on CIFAR10 and CIFAR100 for various choices of loss functions. The two accuracies in each entry show the AAD_t and AAD_{nt}. The initial performances for CIFAR10 on ResNet20 are $Acc_t = 90.86$, $Acc_{nt} = 92.44$; and for CIFAR100 on ResNet56 are $Acc_t = 65.20$; $Acc_{nt} = 70.43$, for targeted classes 0 to 4

CIFAR	$\mathcal{L}_t$	$\mathcal{L}_{nt}$ -		$\mathcal{L}^{CE}$		$\mathcal{L}^{L}$		$\mathcal{L}^{BL}$	
10	$\mathcal{L}^{CE}$	87.04	69.40	84.36	48.48	84.68	52.52	83.86	45.06
	$\mathcal{L}^{L}$	84.64	54.04	85.74	24.14	81.04	25.08	84.94	23.78
	$\mathcal{L}^{BL}$	88.58	61.54	81.86	10.18	64.66	7.92	81.52	11.60
100	$\mathcal{L}^{CE}$	61.20	60.23	60.40	50.50	1.60	1.15	62.40	46.35
	$\mathcal{L}^{L}$	63.80	55.00	63.40	47.12	62.40	46.02	62.80	47.97
	$\mathcal{L}^{BL}$	63.20	53.67	48.80	17.31	23.40	7.16	48.00	17.79

This loss combination is the best

L^{CE}: Cross-entropy loss (Increase for L_t and decrease for L_{nt})

L^{L}: Logit loss (Decrease for L_t and increase for L_{nt})

L^{BL}: Bounded Logit Loss (Similar to C&W loss; Decrease for L_t and increase for L_{nt})

Table 3: Experiments on CIFAR10 with different groups of targeted classes using VGG16 and ResNet20

$\hat{F}$	S	Acc_t	AAD_t	Acc_{nt}	AAD_{nt}	Δ_{AAD}
VGG16	[1 : 5 : 2]	90.57	78.63	94.23	14.74	**63.89**
	[2 : 6 : 2]	92.87	68.87	93.24	21.79	**47.08**
	[0 : 4 : 1]	92.40	75.00	93.86	7.36	**67.64**
	[5 : 9 : 1]	93.86	75.00	92.40	17.56	**57.44**
	[0 : 6 : 1]	92.10	69.24	95.53	3.13	**66.11**
	[3 : 9 : 1]	92.80	78.60	93.90	8.70	**69.90**
ResNet20	[1 : 5 : 2]	88.80	82.27	92.87	15.93	**66.34**
	[2 : 6 : 2]	92.30	79.33	91.37	21.40	**57.93**
	[0 : 4 : 1]	90.86	81.46	92.44	10.24	**71.22**
	[5 : 9 : 1]	92.44	80.16	90.86	17.36	**62.80**
	[0 : 6 : 1]	90.81	81.33	93.60	4.67	**76.66**
	[3 : 9 : 1]	91.21	80.99	92.67	7.90	**73.09**

The higher the better

S: targeted classes, [lowest class, highest class, step size], e.g.
[1:5:2] indicates targeted classes 1,3,5 and
[0:4:1] indicates targeted classes 0,1,2,3,4

[1] CD-UAP: Class-Discriminative Universal Adversarial Perturbations; Zhang*, Benz*, Imtiaz, Kweon; AAAI 2020

CD-UAP: Class-Discriminative UAP

Further results on CIFAR100 and ImageNet

Table 5: Experiments on CIFAR100 targeting superclasses using VGG19 and ResNet56

Super Class	VGG19					ResNet56				
	Acc_t	AAD_t	Acc_{nt}	AAD_{nt}	Δ_{AAD}	Acc_t	AAD_t	Acc_{nt}	AAD_{nt}	Δ_{AAD}
aquatic mammals	56.20	44.20	70.97	14.05	**30.15**	58.20	43.60	70.80	14.87	**28.73**
fish	67.00	45.60	70.40	18.25	**27.35**	67.80	49.00	70.30	18.45	**30.55**
flowers	76.40	33.80	69.91	20.57	**13.23**	75.40	30.80	69.90	24.18	**6.62**
food containers	69.60	52.40	70.26	16.28	**36.12**	71.40	54.40	70.11	16.44	**37.96**
fruit and vegetables	79.40	55.60	69.75	18.04	**37.56**	76.60	52.00	69.83	17.65	**34.35**
household electrical devices	70.00	49.20	70.24	15.95	**33.25**	75.00	55.60	69.92	16.52	**39.08**
household furniture	77.40	63.80	69.85	15.76	**48.04**	76.00	60.00	69.86	14.90	**45.10**
insects	70.20	38.60	70.23	16.99	**21.61**	71.60	45.20	70.09	21.24	**23.96**
large carnivores	70.20	53.60	70.23	15.50	**38.10**	70.40	60.20	70.16	15.44	**44.76**
large man-made outdoor objects	82.60	66.60	69.58	15.56	**51.04**	81.20	66.20	69.59	15.49	**50.71**
large natural outdoor scenes	79.00	70.80	69.77	13.12	**57.68**	78.40	67.60	69.74	12.13	**55.47**
large omnivores and herbivores	69.80	51.60	70.25	18.33	**33.27**	71.40	56.00	70.10	17.15	**38.85**
medium-sized mammals	72.80	49.40	70.09	17.27	**32.13**	71.80	54.20	70.08	18.83	**35.37**
non-insect invertebrates	66.40	40.60	70.43	18.09	**22.51**	66.00	44.60	70.39	18.38	**26.22**
people	52.20	31.60	71.18	14.67	**16.93**	48.40	32.60	71.32	14.85	**17.75**
reptiles	59.20	43.80	70.81	16.05	**27.76**	59.20	42.80	70.74	17.24	**25.56**
small mammals	56.00	47.60	70.98	15.14	**32.46**	56.40	47.20	70.90	13.34	**33.86**
trees	66.80	49.60	70.41	15.58	**34.02**	66.60	54.00	70.36	17.05	**36.95**
vehicles 1	82.20	44.00	69.60	17.99	**26.01**	80.60	51.60	69.62	20.44	**31.16**
vehicles 2	81.20	53.40	69.65	21.33	**32.07**	81.00	62.20	69.60	21.92	**40.28**

Table 7: Experiments on ImageNet targeting 1 super class, $\epsilon = 15$.

	Super Classes	Acc_t	AAD_t	Acc_{nt}	AAD_{nt}	Δ_{AAD}
VGG16	Frogs	70.0	46.0	71.6	19.5	**26.5**
	Sharks	80.0	53.3	71.6	16.8	**36.5**
	Aircrafts	78.0	69.3	71.6	17.7	**51.6**
	Racket Radiator Radio	68.6	42.7	71.6	18.5	**24.2**
	Space objects	56.7	20.0	71.6	18.5	**1.5**
VGG19	Frogs	70.0	42.0	72.4	18.0	**24.0**
	Sharks	81.3	55.3	72.3	16.1	**39.2**
	Aircrafts	81.3	72.0	72.4	17.2	**54.8**
	Racket Radiator Radio	69.3	37.3	72.4	17.3	**20.0**
	Space objects	59.3	26.6	72.4	20.0	**6.6**
ResNet50	Frogs	75.3	52.0	76.1	18.5	**33.5**
	Sharks	83.3	66.7	76.1	16.1	**50.6**
	Aircrafts	84.0	65.3	76.1	16.8	**48.5**
	Racket Radiator Radio	75.3	46.7	76.1	17.1	**29.6**
	Space objects	59.3	43.3	76.2	20.4	**22.9**
ResNet152	Frogs	74.0	48.0	78.3	17.2	**30.8**
	Sharks	80.0	61.3	78.3	12.9	**48.4**
	Aircrafts	86.0	78.7	78.3	16.1	**62.6**
	Racket Radiator Radio	72.7	44.0	78.3	14.4	**29.6**
	Space objects	64.7	36.7	78.4	16.7	**20.0**

[1] CD-UAP: Class-Discriminative Universal Adversarial Perturbations; Zhang*, Benz*, Imtiaz, Kweon; AAAI 2020

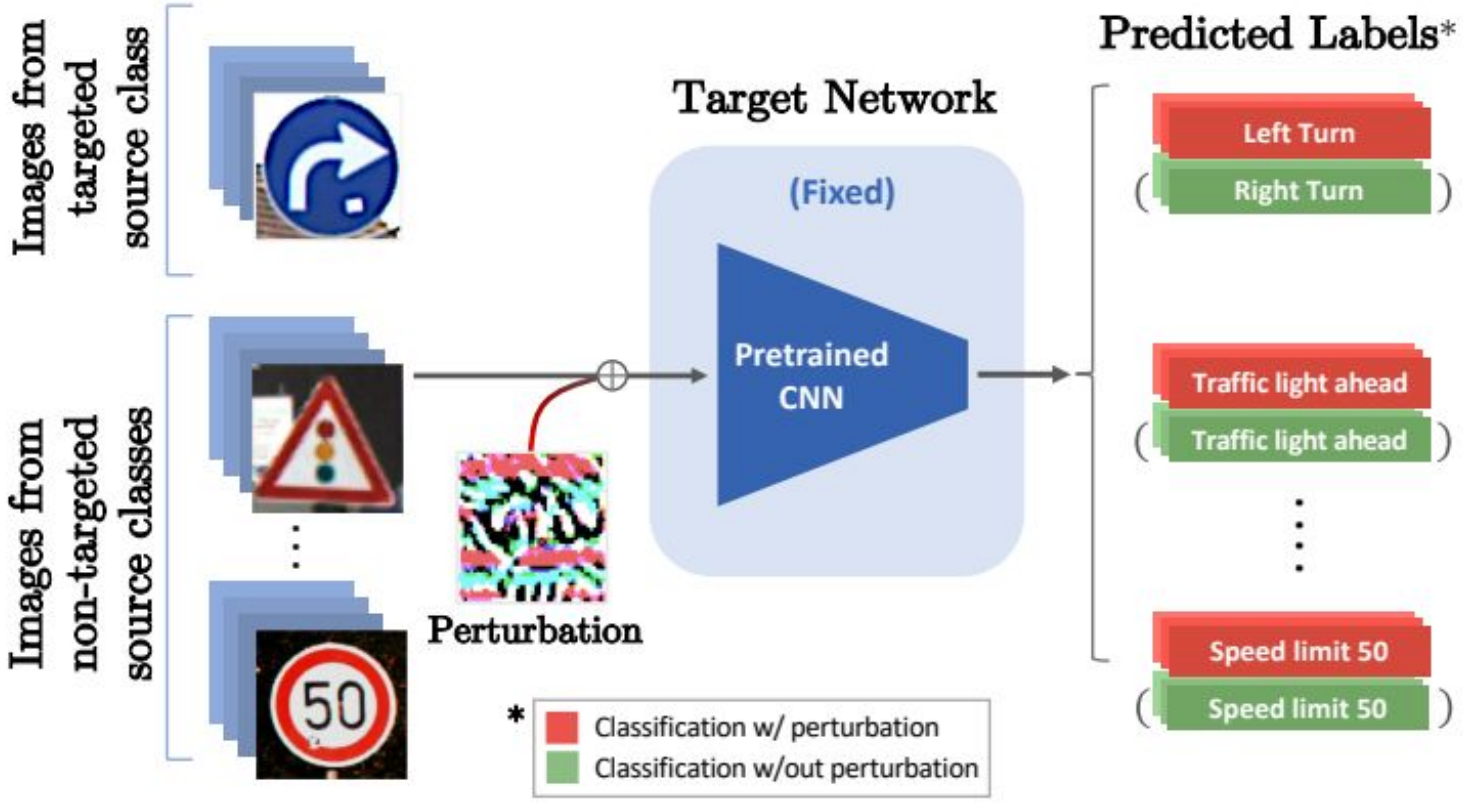
Double targeted universal adversarial perturbations (ACCV 2020)

Philipp Benz*, Chaoning Zhang*, Tooba Imtiaz, In-So Kweon
* indicates equal contribution

Korea Advanced Institute of Science and Technology (KAIST)

Double Targeted UAPs

UAP to fool one targeted source class to a sink class, while having limited adversarial effect on other non-targeted source classes to avoid raising suspicion.



[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Loss Design

The loss consists of a targeted (L_t) and a non-targeted part (L_{nt})

$$\mathcal{L} = \mathcal{L}_t + \alpha \mathcal{L}_{nt}$$

Objective of the targeted part: Fool the network by shifting the prediction from the source class into the sink class
This is broken down into two subtasks:

(1) Decreasing the logit value for the originally predicted class L_p to not being the highest logit anymore
(2) Increasing the logit for the sink class L_{sink}, to be the highest logit

$$\mathcal{L}_t = \mathcal{L}_{t1} + \mathcal{L}_{t2}, \text{ with}$$

$$\mathcal{L}_{t1} = \max(\hat{L}_p(x_t + \delta) - \max_{i \neq p}(\hat{L}_i(x_t + \delta)), 0) \qquad p = \arg\max(\hat{L}(x_t))$$

$$\mathcal{L}_{t2} = \max(\max_{i \neq y_{\text{sink}}}(\hat{L}_i(x_t + \delta) - \hat{L}_{\text{sink}}(x_t + \delta)), -D)$$

Objective of the non-targeted part: Achieve good classification performance
Adopt the cross-entropy function

$$\mathcal{L}_{nt} = \mathcal{X}(\hat{L}(x_{nt} + \delta), \mathbb{1}(\hat{F}(x_{nt})))$$

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Algorithm

A simple, yet effective algorithm

Algorithm 1: Double Targeted Attack Algorithm

Input: Data distribution X, Classifier $\hat{F}$, Loss function $\mathcal{L}$, Mini-batch size m, Number of iterations I, Perturbation magnitude ϵ

Output: Perturbation vector δ

$X_t \subseteq X$ ▷ Subset

$X_{nt} \subseteq X$ ▷ Subset

$\delta \leftarrow 0$ ▷ Initialize

for *iteration* $= 1, \ldots, I$ **do**

- $B_t \sim X_t$: $|B_t| = \frac{m}{2}$ ▷ Randomly sample
- $B_{nt} \sim X_{nt}$: $|B_{nt}| = \frac{m}{2}$ ▷ Randomly sample
- $B \leftarrow B_t \bigcup B_{nt}$ ▷ Concatenate
- $g_\delta \leftarrow \mathop{\mathbb{E}}_{B}[\nabla_\delta \mathcal{L}]$ ▷ Calculate gradient
- $\delta \leftarrow \texttt{Optim}(g_\delta)$ ▷ Update perturbation
- $\delta \leftarrow \frac{\delta}{||\delta||_p}\epsilon$ ▷ Projection

end

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Quantitative Results

DTA achieves reasonable performance for different mapping scenarios on a wide range of datasets.

1. The targeted fooling ratio for the targeted classes (κ_t) is reasonably high.
2. A significant gap between κ_t and κ_{nt} indicates that the crafted perturbation is discriminative between targeted class and non-targeted classes

Table: Experimental results for the Double Targeted Attack (DTA) for the datasets CIFAR-10, GTSRB, EuroSAT, YCB and ImageNet under 10 scenarios S_0 to S_9. All numbers are reported in %.

Dataset	Model	S_0		S_1		S_2		S_3		S_4		S_5		S_6		S_7		S_8		S_9		Avg	
		κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}
CIFAR-10	VGG-16	77.5	20.5	83.5	22.0	78.2	14.7	81.4	21.5	73.0	18.6	79.1	14.2	75.1	15.1	76.7	24.6	75.0	20.3	86.2	16.6	78.6	18.8
	ResNet-20	78.8	26.1	84.6	28.0	84.0	24.3	84.2	26.9	77.1	22.0	82.1	21.3	83.8	14.7	72.9	33.2	80.0	27.8	89.8	22.3	81.7	24.7
GTSRB	VGG-16	89.0	0.2	100	1.1	87.1	1.2	72.2	0.6	91.0	1.3	83.6	2.4	88.3	1.1	80.0	0.7	95.0	1.9	81.1	1.7	86.7	1.2
	ResNet-20	84.3	0.5	100	1.6	53.1	0.2	77.8	1.8	87.6	2.9	77.1	4.4	70.0	2.7	88.3	1.2	80.0	0.3	64.4	0.7	78.3	1.6
EuroSAT	ResNet-50	96.2	33.0	98.8	18.0	95.2	31.1	96.6	22.1	99.2	28.7	95.0	24.0	94.4	44.3	96.3	17.6	96.3	24.5	91.2	22.7	95.9	26.6
	Inception-V3	94.3	28.7	95.2	18.9	93.8	41.4	99.2	56.3	93.0	29.4	93.0	24.2	91.6	34.6	96.0	21.8	96.8	31.6	89.2	18.8	94.2	30.6
YCB	ResNet-50	100	14.5	100	24.2	100	32.4	96.7	38.0	100	33.5	99.2	38.3	100	44.4	99.2	41.7	100	19.0	100	33.1	99.5	31.9
	Inception-V3	100	16.6	100	30.0	100	38.7	99.2	31.2	100	12.9	98.3	20.0	100	32.2	100	36.6	100	17.3	100	39.2	99.8	27.5
ImageNet	VGG-16	72.0	10.3	96.0	19.5	90.0	19.5	82.0	28.3	74.0	15.9	82.0	13.0	66.0	8.9	64.0	12.9	66.0	21.5	70.0	26.1	76.2	17.6
	ResNet-50	74.0	13.9	94.0	21.4	82.0	15.2	72.0	20.9	62.0	13.6	84.0	15.5	72.0	9.8	66.0	21.4	66.0	17.3	62.0	18.1	73.4	16.7
	Inception-V3	78.0	10.0	86.0	15.7	86.0	12.2	78.0	15.6	58.0	9.5	76.0	12.9	70.0	8.9	72.0	15.7	62.0	18.9	66.0	17.8	73.2	13.7
	MobileNet-V2	74.0	11.3	94.0	17.0	88.0	20.4	70.0	15.3	72.0	16.0	84.0	15.0	74.0	14.5	74.0	21.7	72.0	18.8	70.0	21.9	77.2	17.2

κ_t: targeted fooling ratio for the targeted source samples

κ_{nt}: targeted fooling ratio for the non-targeted source samples

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Multi2One attack

Attack multiple source classes (MS) to be misclassified as one sink class.

Table: Experimental results for the universal Multi2One targeted perturbation on ImageNet under 4 scenarios MS_0 to MS_3 attacking three source classes to one sink class. All numbers are reported in %.

Model	MS_0		MS_1		MS_2		MS_3		Avg	
	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}
VGG16	63.3	24.9	69.3	33.7	76.0	25.8	69.3	26.8	69.5	27.8
ResNet-50	64.0	30.1	63.3	32.3	78.7	29.2	62.7	23.2	67.2	28.7
Inception-V3	58.0	19.4	56.7	23.8	66.7	19.0	66.7	20.8	62.0	20.8
MobileNet-V2	68.0	27.2	66.0	28.0	74.0	25.6	66.0	24.4	68.5	26.3

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Qualitative Results

original images

amplified version of the corresponding perturbations

adversarial example

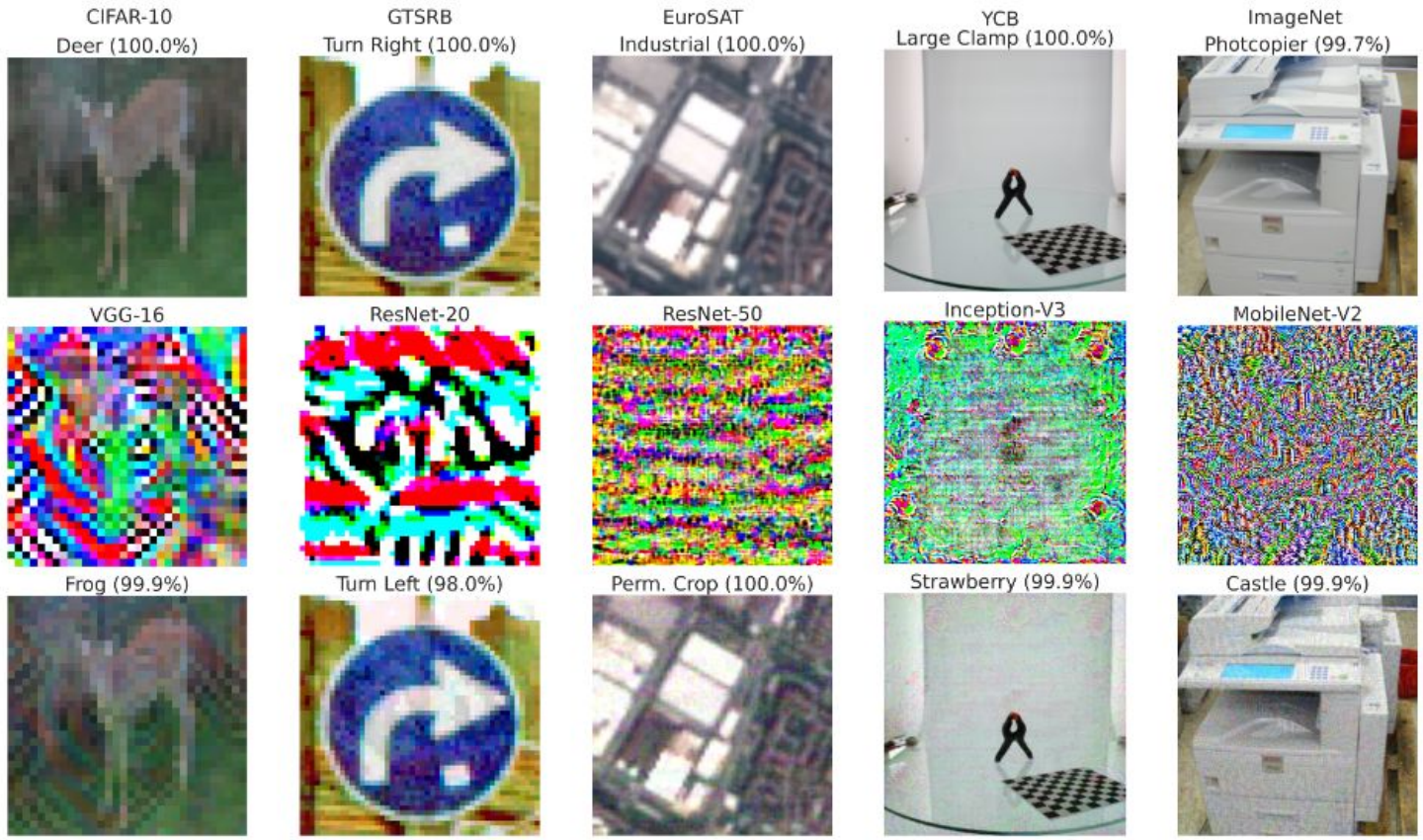


Figure: Examples of adversarial perturbations for various datasets and networks. The confidence values of the network and the predicted labels are stated above the images. The target network is indicated above the amplified perturbation.

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Physical Attack - Double Targeted Patch

Table: Quantitative results for the generated DT-Patch on ImageNet

Hammer → Hummingbird		Screwdriver → Go-Kart		Coffee Mug → Chocolate Sauce	
κ_t	κ_{nt}	κ_t	κ_{nt}	κ_t	κ_{nt}
80.0	42.7	92.0	44.9	96.0	41.6

Figure: Real-world examples of the DT-Patch for the three different scenarios

[1] Double targeted universal adversarial perturbations; Benz*, Zhang*, Imtiaz, Kweon; ACCV 2020

Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations (CVPR 2020)

Chaoning Zhang*, Philipp Benz*, Tooba Imtiaz, In-So Kweon
* indicates equal contribution

Korea Advanced Institute of Science and Technology (KAIST)

Background: Adversarial Examples are not Bugs they are Features

Adversarial Examples can be directly attributed to the presence of non-robust features [1]

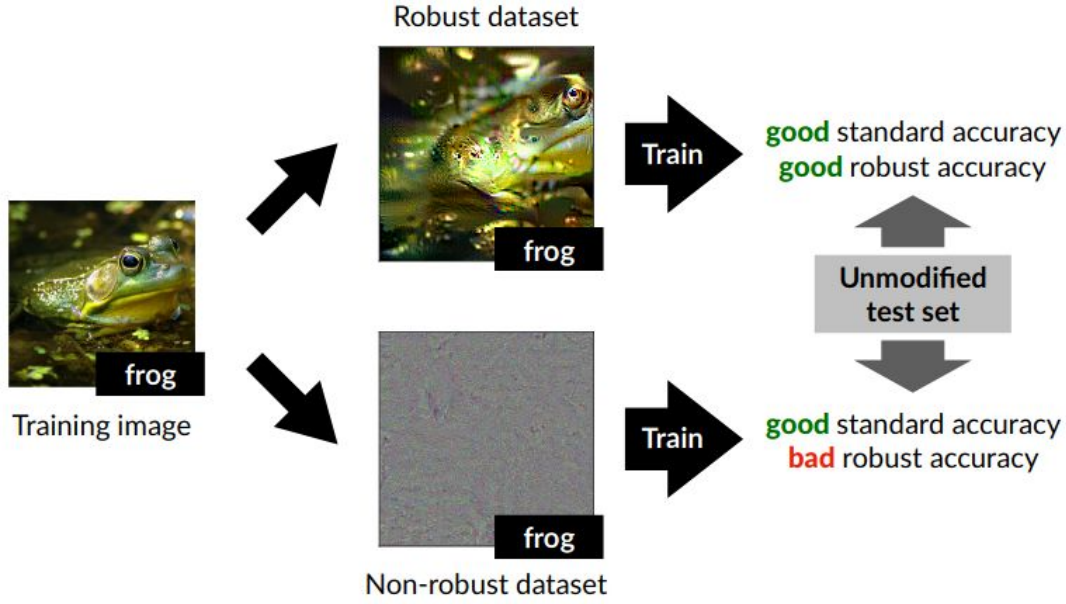


Figure 1: Disentanglement of features into combinations of robust and non-robust features [1]

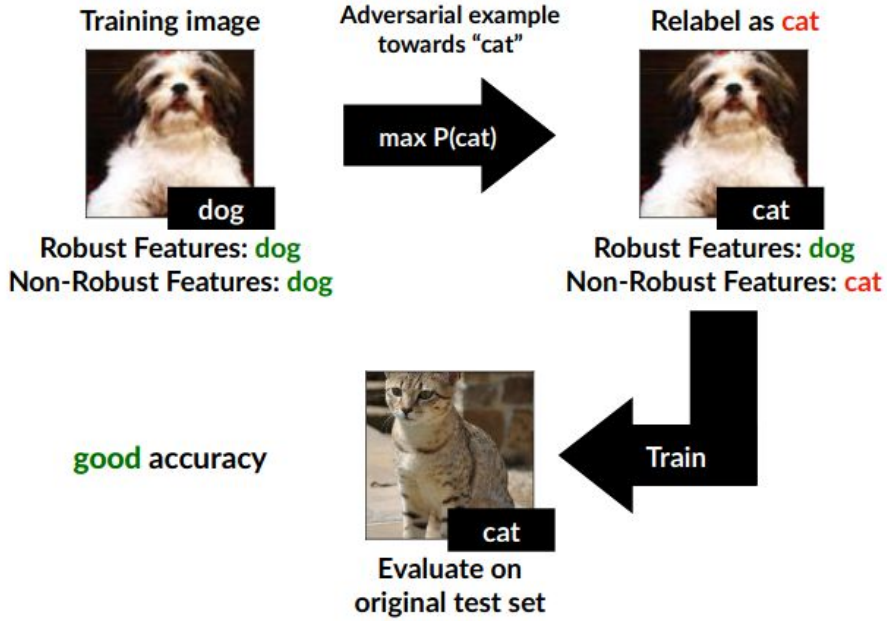


Figure 2: A dataset appearing mislabeled to humans (via adversarial examples) can results in good accuracy on the original test set [1]

[1] Adversarial Examples Are Not Bugs, They Are Features; Ilyas, Santurkar, Tsipras, Engstrom, Tran, Madry; NeurIPS 2019

Background: Adversarial Examples are not Bugs they are Features

The existence of adversarial examples can be attributed to the existence of Non-robust features in the dataset.

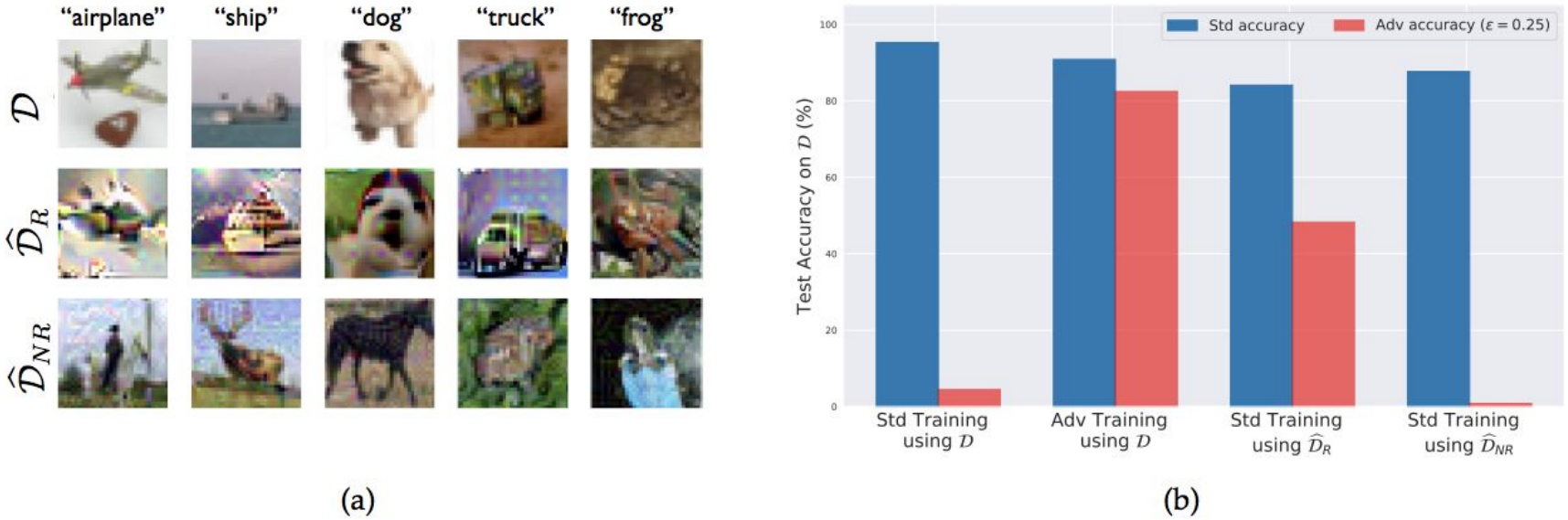


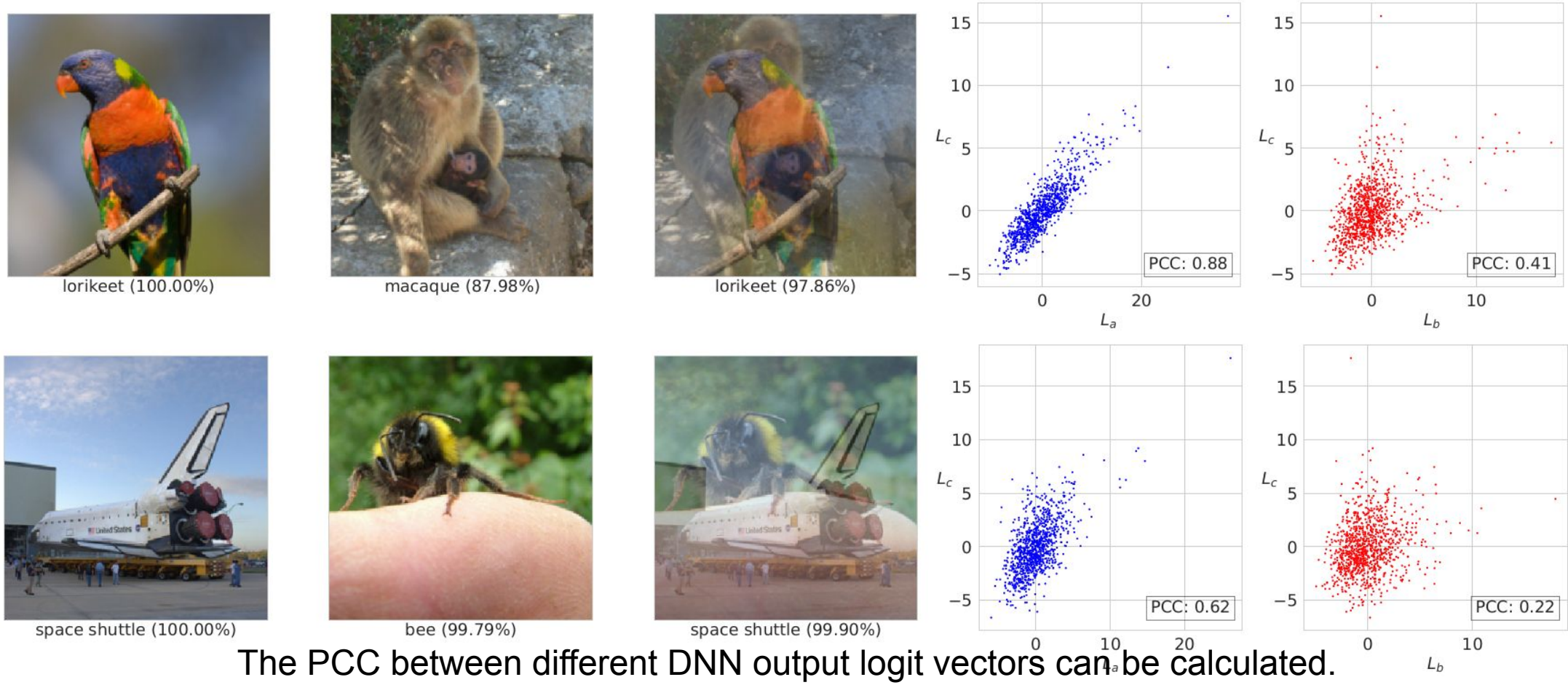
Figure 2: **Left**: Random samples from our variants of the CIFAR-10 [Kri09] training set: the original training set; the *robust training set* $\widehat{\mathcal{D}}_R$, restricted to features used by a robust model; and the *non-robust training set* $\widehat{\mathcal{D}}_{NR}$, restricted to features relevant to a standard model (labels appear incorrect to humans). **Right**: Standard and robust accuracy on the CIFAR-10 test set ($\mathcal{D}$) for models trained with: (i) standard training (on $\mathcal{D}$) ; (ii) standard training on $\widehat{\mathcal{D}}_{NR}$; (iii) adversarial training (on $\mathcal{D}$); and (iv) standard training on $\widehat{\mathcal{D}}_R$. Models trained on $\widehat{\mathcal{D}}_R$ and $\widehat{\mathcal{D}}_{NR}$ reflect the original models used to create them: notably, standard training on $\widehat{\mathcal{D}}_R$ yields nontrivial robust accuracy. Results for Restricted-ImageNet [Tsi+19] are in D.8 Figure 12.

[1] Adversarial Examples Are Not Bugs, They Are Features; Ilyas, Santurkar, Tsipras, Engstrom, Tran, Madry; NeurIPS 2019

PCC Analysis

The Pearson correlation coefficient (PCC) is a widely adopted metric to measure the linear correlation between two variables.

$$\mathrm{PCC}_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$



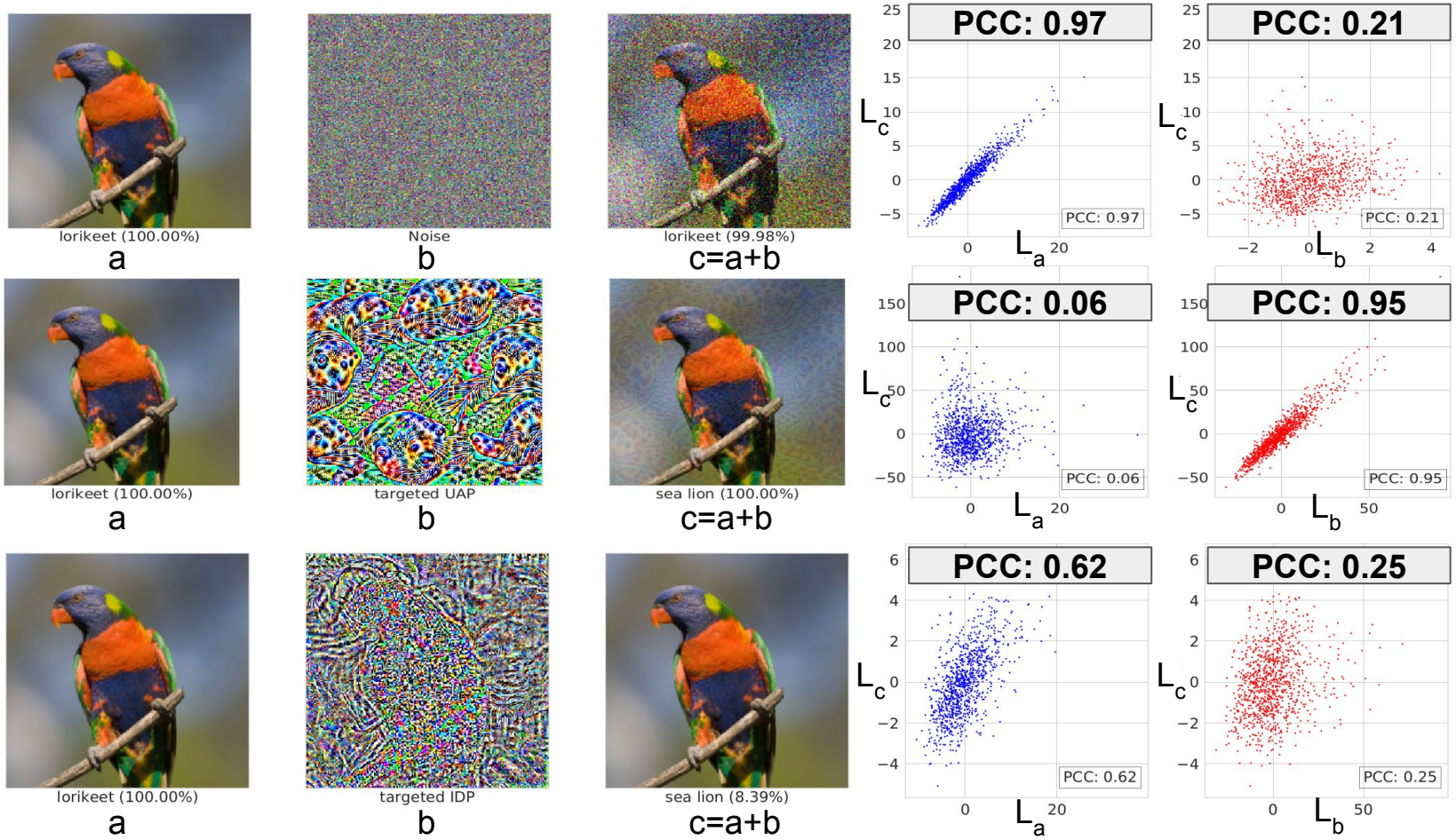
The PCC between different DNN output logit vectors can be calculated.

This provides insight about the “stronger” contribution of the two inputs to the final outcome. A higher PCC value indicates the more significant contributor.

[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

PCC Analysis

Treat the DNN logits as a vector for feature representation and use them to analyze the mutual influence of two independent inputs based on the Pearson correlation coefficient (PCC)



"Universal perturbations contain dominant features, and images behave like noise to them"

Image-dependant perturbations seem to not contain features by themselves

PCC-Analysis result for one sample image `lorikeet'. Three scenarios of input combinations are considered: 1: image + noise; 2: image + targeted UAP; 3: image + targeted image-dependant AE. The columns show input a, input b, input c=a+b, logit vector analysis of L_c over L_a and vector analysis of L_c over L_b

[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

Noise Perspective vs. Feature Perspective

Noise Perspective (Prior works)

- Treat the targeted UAP as noise (“bug”) to the sample to be attacked
- Requires the samples from the training dataset in the UAP generation process
- Explicitly designed to let individual samples cross the decision boundary
- Assumes that the attack generalizes to unseen samples

Requires the original training dataset
Slow: ~2 hours

Feature Perspective (Ours)

- **UAPs contain features of a certain class**
- Treatment of the **images as noise** to the generated UAP during the optimization process in order to be recognizable by the target network
- **No need for semantic features** as in the original training dataset samples
- **Proxy datasets** as background noise: Downloaded from the Internet, MS-COCO, Pascal VOC, Places365

Requires no original training dataset
Fast: ~2 minutes

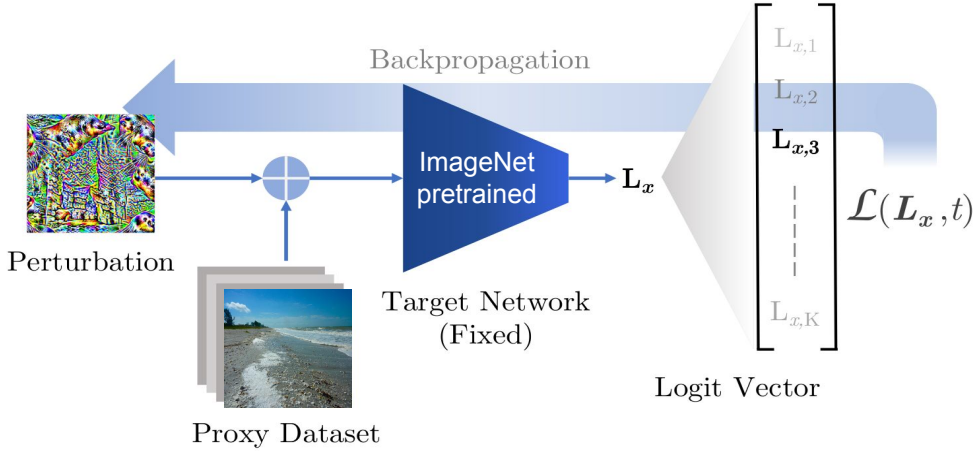
[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

Data-free Targeted UAP

If images just behave like noise to the features in UAPs. Can we just use another dataset to craft UAPs? Yes we can!

UAP generation stage

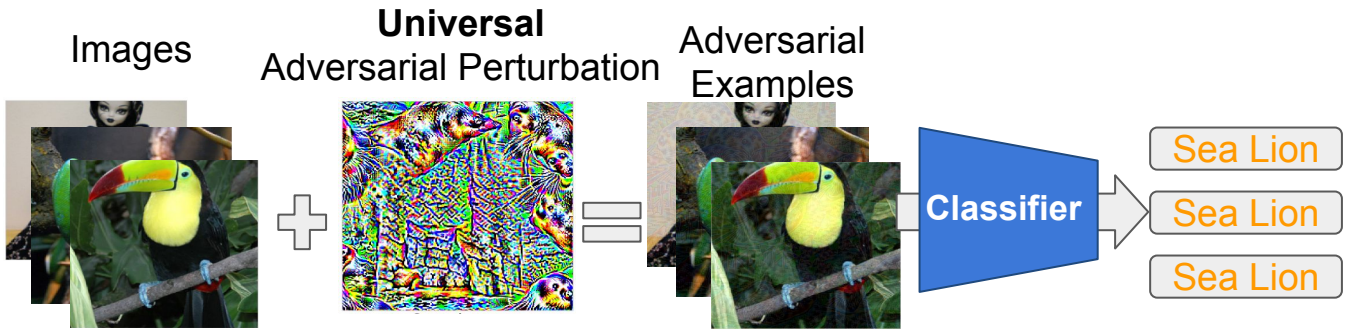
Craft the perturbation with the help of some proxy dataset



As a loss function you can use the cross-entropy between the output logit and some (randomly chosen) target class.
For example: Choose target class sea lion

UAP testing stage

Test the perturbation on the ImageNet validation dataset



Is the sample misclassified?
Good

Is the sample correctly classified?
Bad

Figure 1: Inference with adversarial examples formed with a targeted universal adversarial perturbation crafted for target class “sea lion”.

[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

Data-free Targeted UAP: Results

The proposed approach outperforms previous data-dependant and data-free UAP generation methods

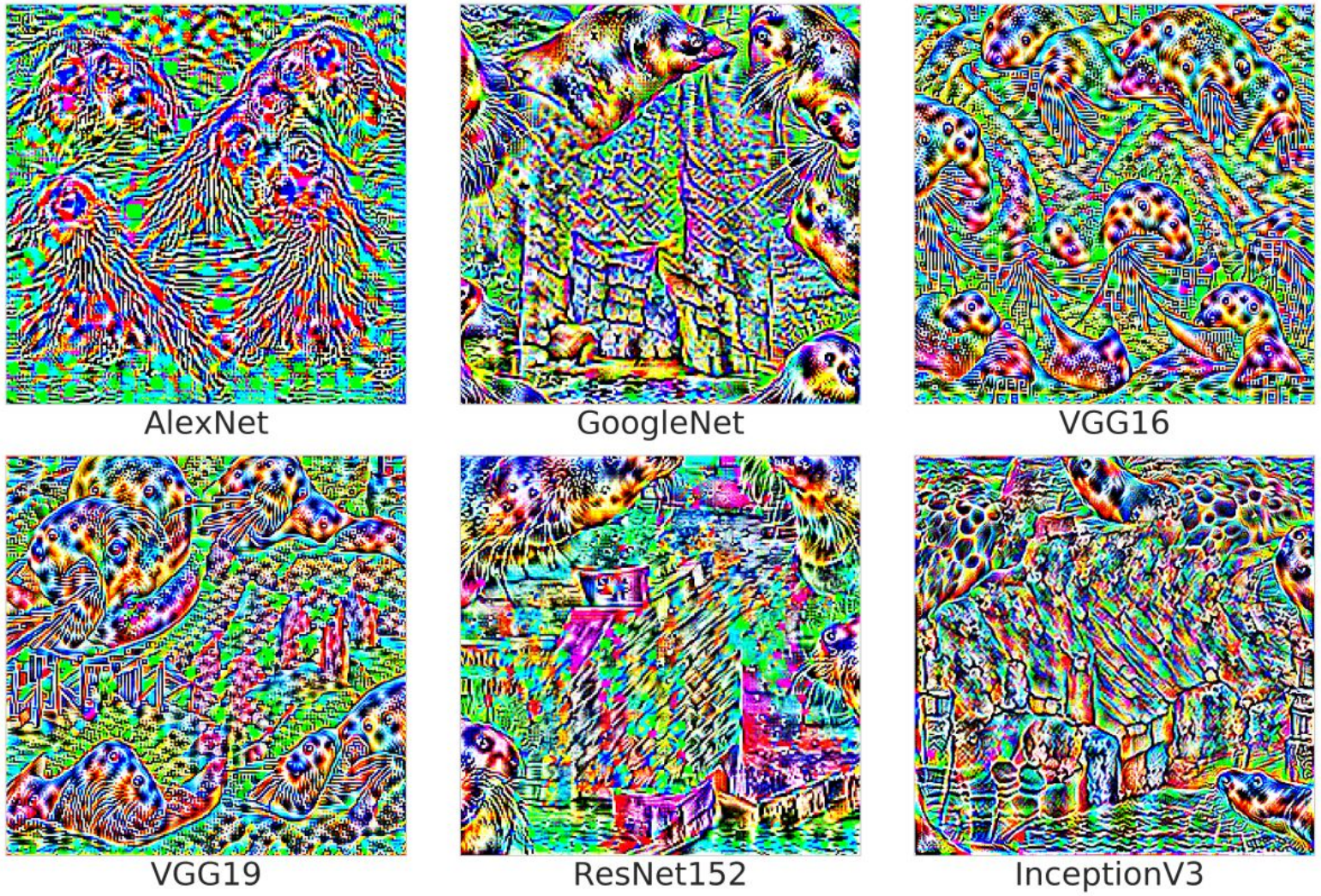
Table 2: Comparison to other methods. The results are divided in universal attacks with access to the original ImageNet training data (upper) and data-free methods (lower). The metric is reported in the non-targeted fooling ratio (%)

Method	AlexNet[1]	GoogleNet	VGG16	VGG19	ResNet152
UAP [14]	93.3	78.9	78.3	77.8	84.0
GAP [19]	-	82.7	83.7	80.1	-
Ours(ImageNet [11])	**96.17**	**88.94**	**94.30**	**94.98**	**90.08**
FFF [18]	80.92	56.44	47.10	43.62	-
AAA [21]	89.04	75.28	71.59	72.84	60.72
GD-UAP [17]	87.02	71.44	63.08	64.67	37.3
Ours (COCO [12])	89.9	**76.8**	**92.2**	**91.6**	**79.9**
Ours (VOC [5])	89.9	76.7	**92.2**	90.5	79.1
Ours (Places365 [29])	**90.0**	76.4	92.1	91.5	78.0

[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

Can you see the features?

The UAPs were generated for target class sea lion. Sea lion features become visible in the UAPs!



[1] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging

Chaoning Zhang*, Philipp Benz*, Adil Karjauv*, Geng Sun, In-So Kweon
* indicates equal contribution

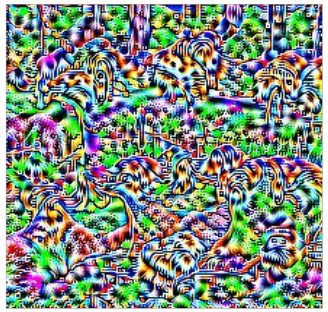
Korea Advanced Institute of Science and Technology (KAIST)

What you see is not what the machine sees

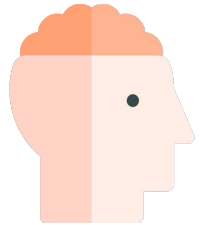
Universal Adversarial Attack

Original Image

Perturbation (Cat feature)



Perturbed Image



What?

Dog!

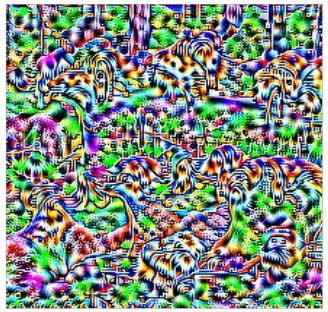
Cat!

What you see is not what the machine sees

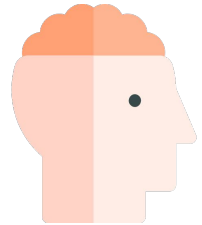
Universal Adversarial Attack

Original Image

Perturbation (Cat feature)



Perturbed Image



What?

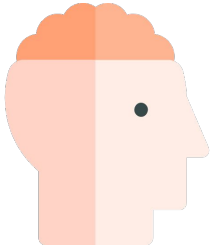
Dog!

Cat!

Universal Deep Hiding

Perturbation (Encoded secret)

Encode



Secret?

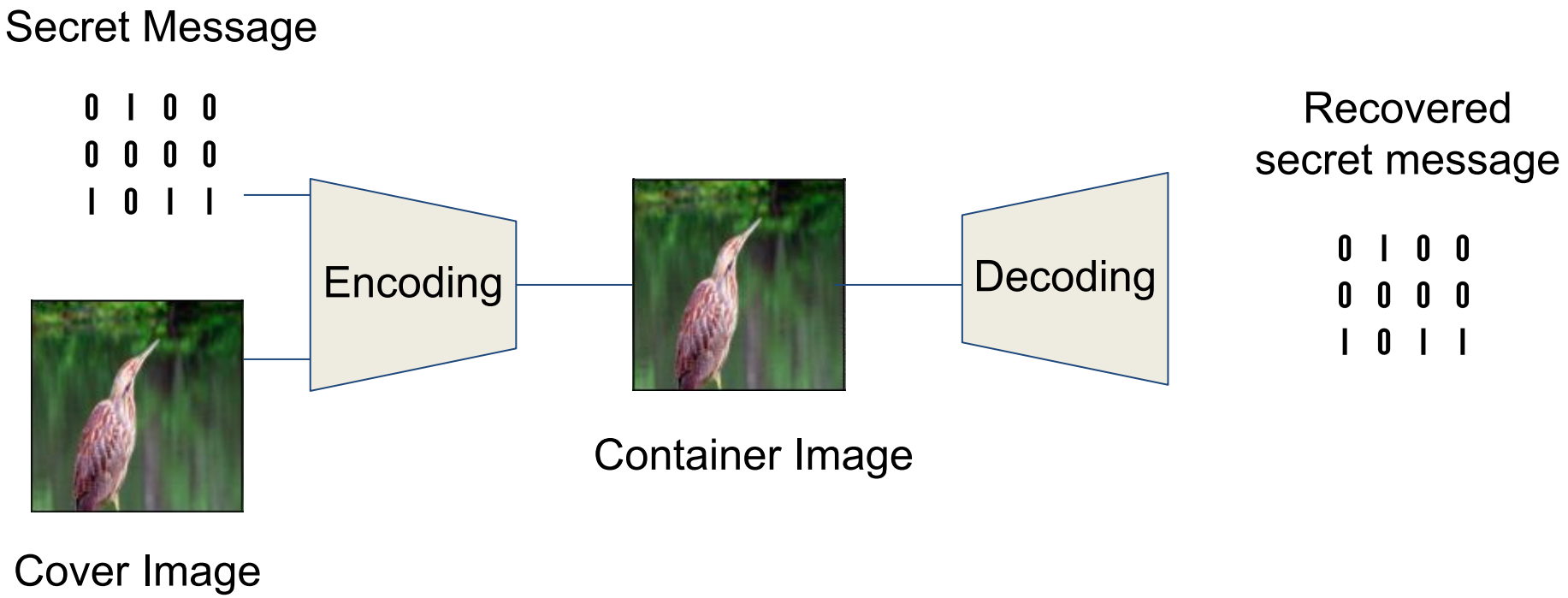
No!

Yes!

Decode

Digital Steganography

- Definition: Hiding secret message within a digital file, such as images or videos.
- Challenge: Trade-off between Capacity vs. secrecy

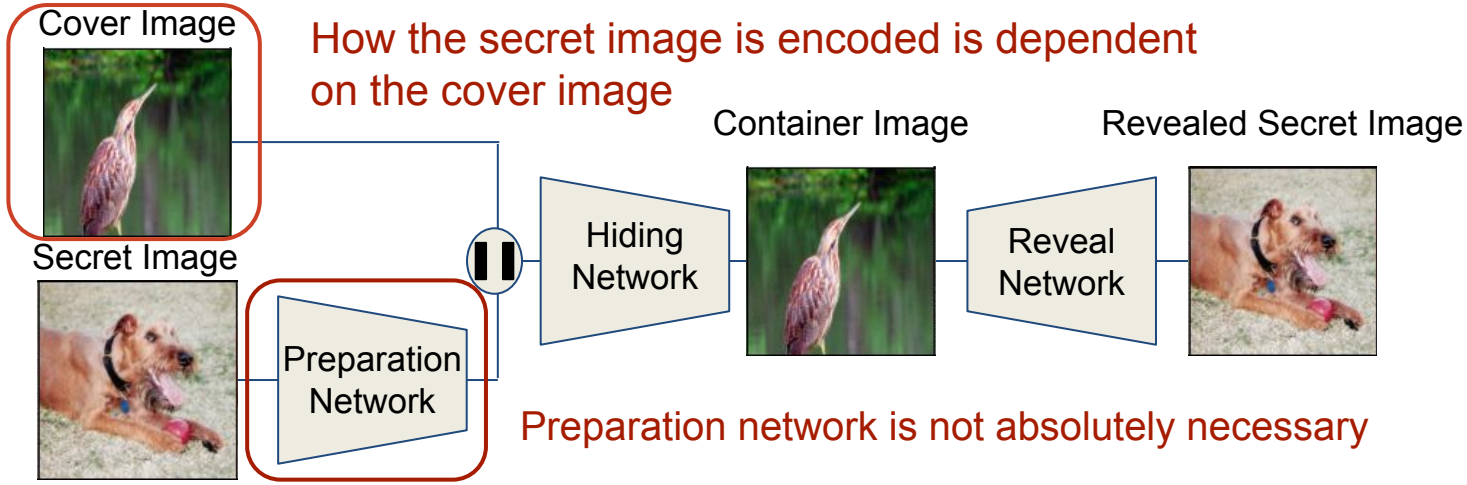


Problems of existing solutions

Traditional steganography vs. Deep steganography

Traditional Steganography	Deep Steganography [1]
Secret: Binary message (<0.5 bits per pixel)	Secret: a full image (24 bits per pixel)
Requires perfect encoding/decoding	Highlighting visual quality
Evading steganalysis is important	Inevitable to be detected due to high capacity

Hiding Images in Plain Sight: Deep Steganography [1]



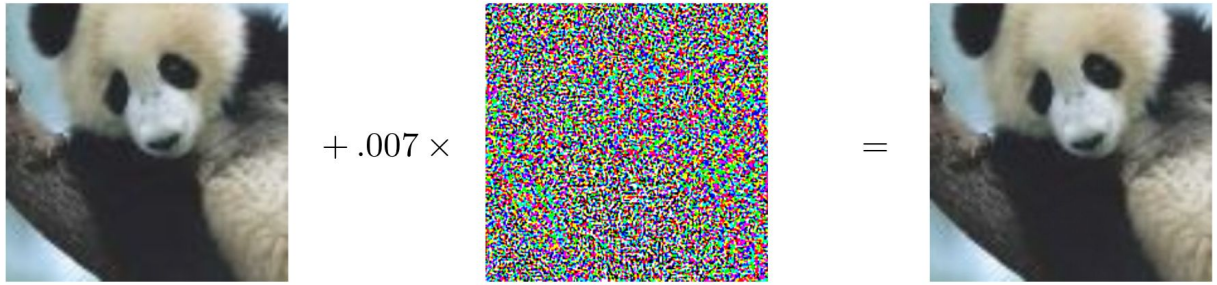
[1] Hiding images in plain sight: Deep steganography; Baluja; NeurIPS 2017

Deep Neural Networks are sensitive to invisible perturbations, leading to misclassifications.
Universal adversarial perturbations exists to fool most images.

Image

Image-dependent
Adversarial Perturbation [1]

Adversarial Example

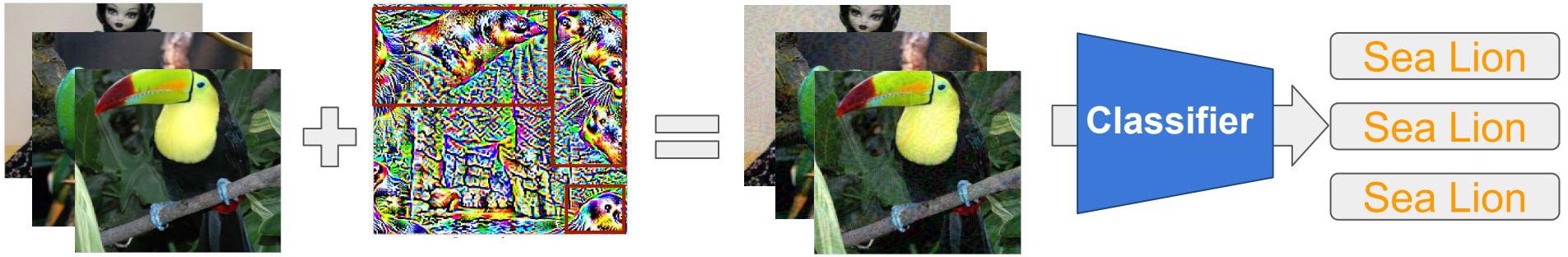


Cover-**D**ependent
Deep **H**iding
(Existing DDH)

Images

Universal
Adversarial Perturbation [2]

Adversarial Examples

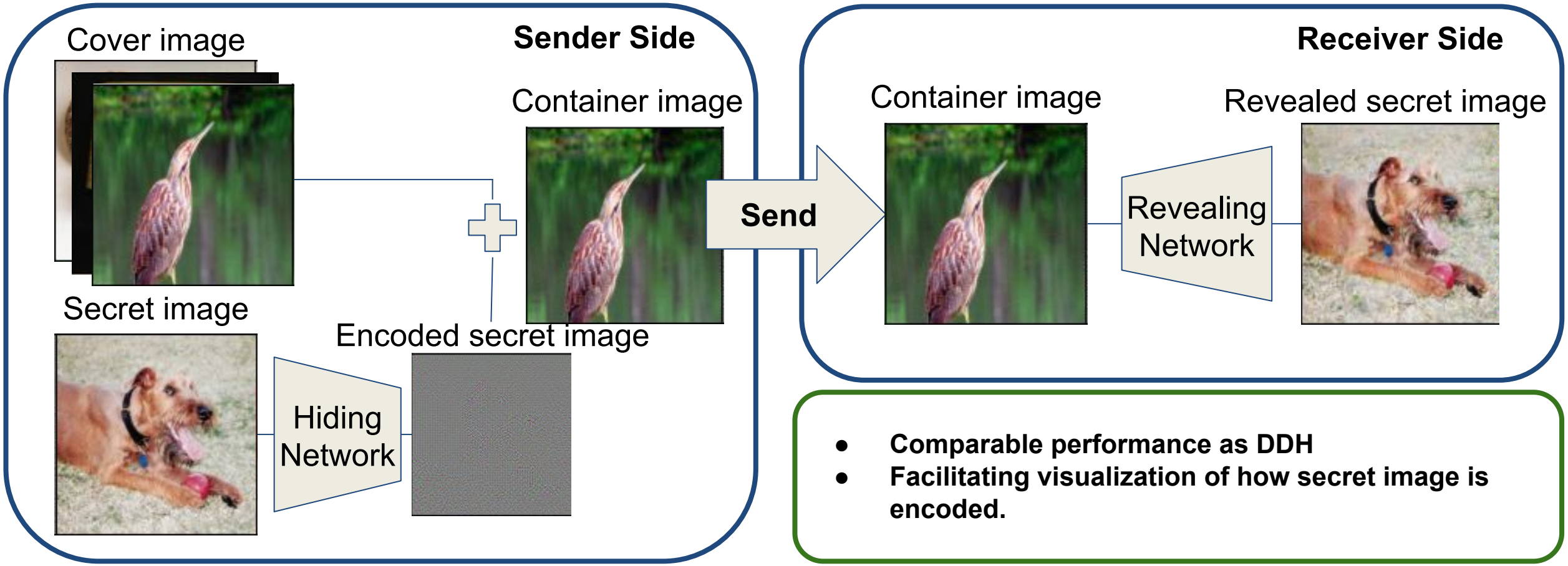


Universal
Deep **H**iding
(Our UDH)

[1] Explaining and Harnessing Adversarial Examples; Goodfellow, Shlens, Szegedy; ICLR 2015
[2] Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations; Zhang*, Benz*, Imtiaz, Kweon; CVPR 2020

Universal Deep Hiding (UDH)

A novel Deep Hiding meta-architecture to hide secret image in a **cover-agnostic** manner.



- **Comparable performance as DDH**
- **Facilitating visualization of how secret image is encoded.**

A secret image is fed to the hiding network to yield an encoded secret image, which can be added to a random cover image to form a container image. The revealing network then retrieves the secret image from the container.

[1] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

The Hiding network encodes secret image in high-frequency representation. The reveal network easily extracts high-frequency encoded secret image from container

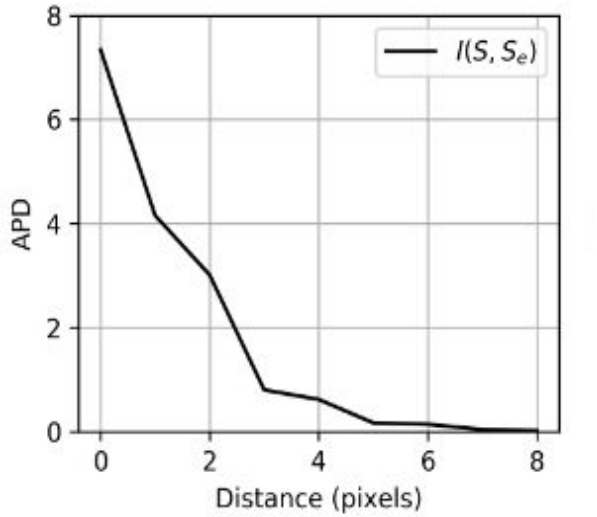


Figure 1: Local encoding. Influence of setting one pixel in the secret image to zero on the encoded secret image.

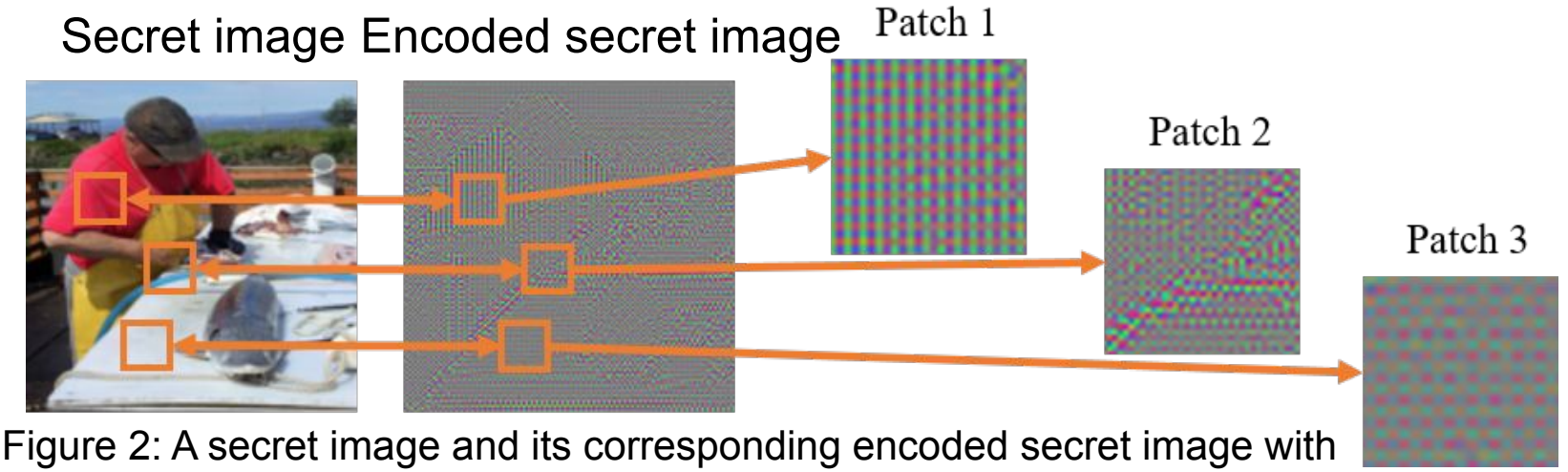


Figure 2: A secret image and its corresponding encoded secret image with zoomed patches for the **UDH** meta architecture.

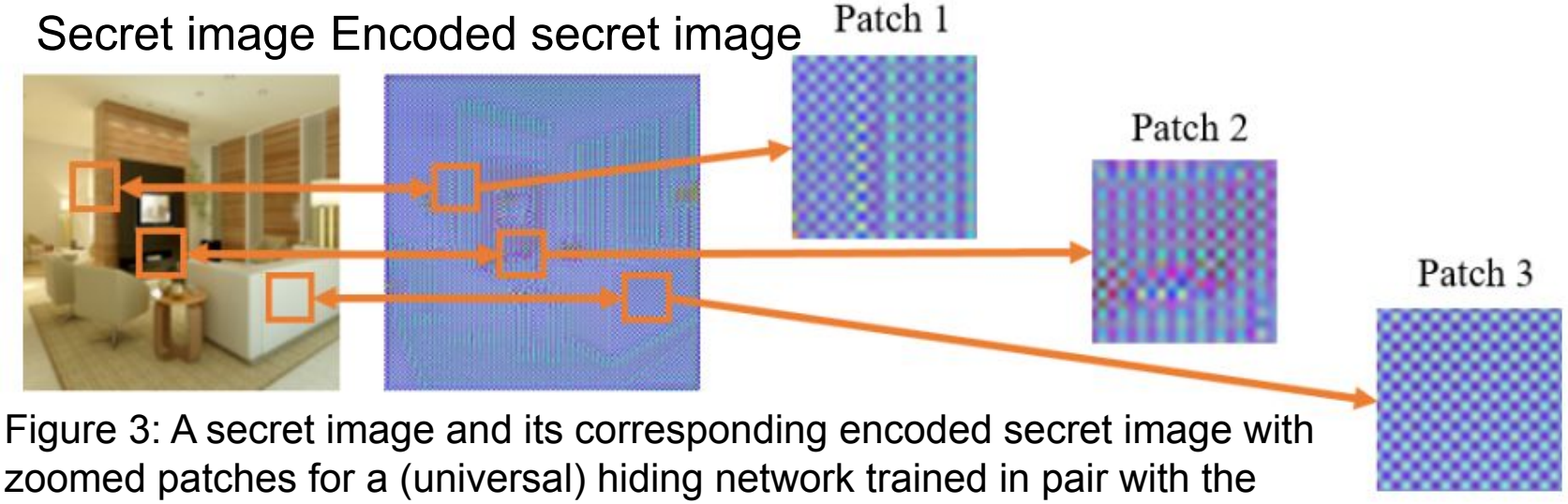


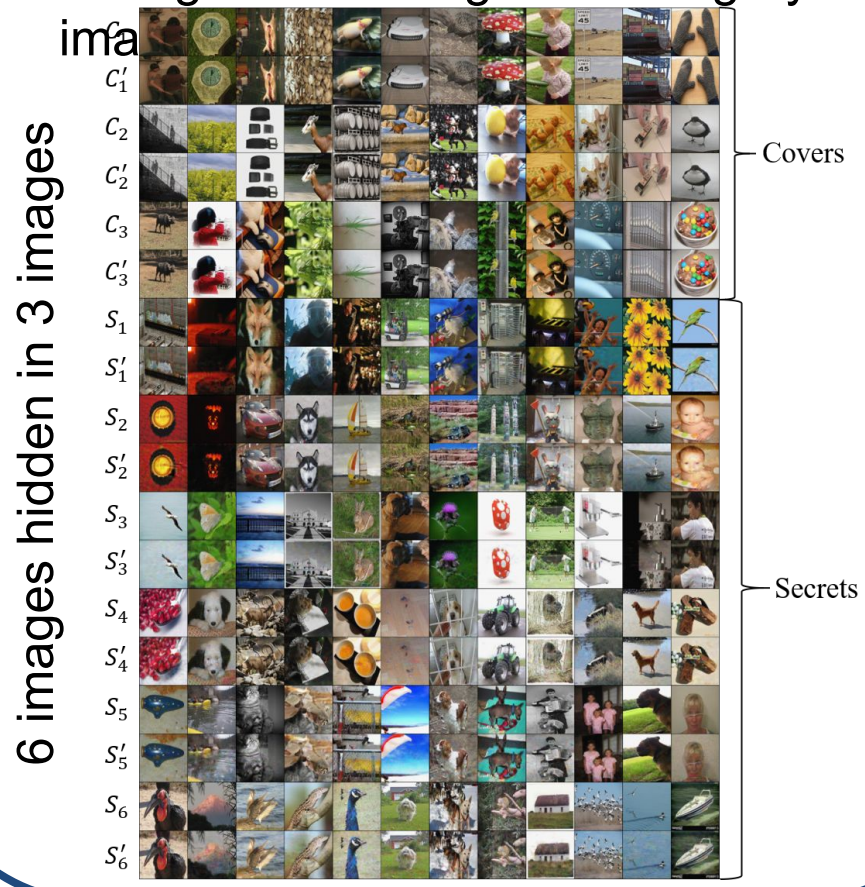
Figure 3: A secret image and its corresponding encoded secret image with zoomed patches for a (universal) hiding network trained in pair with the **fixed** reveal network from the **DDH** meta architecture.

[1] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Beyond hiding one image in one image

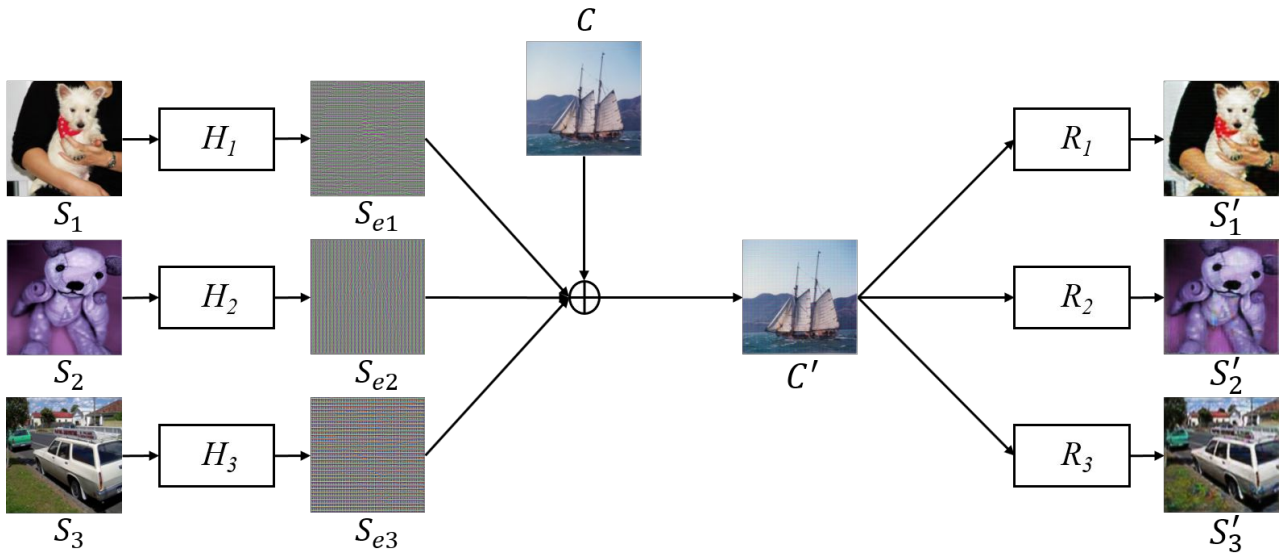
Hiding *M* Images in *N* Images

- Hiding many images in few images
- Hiding 2 color images in one gray image



Different Secrets for Different Recipients

Hiding multiple secret images in one cover so that different recipients retrieve their respective secrets



[1] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Beyond Digital Steganography

Watermarking

- Definition: Hiding secret message for proving the ownership
- Challenge: robustness to distortions

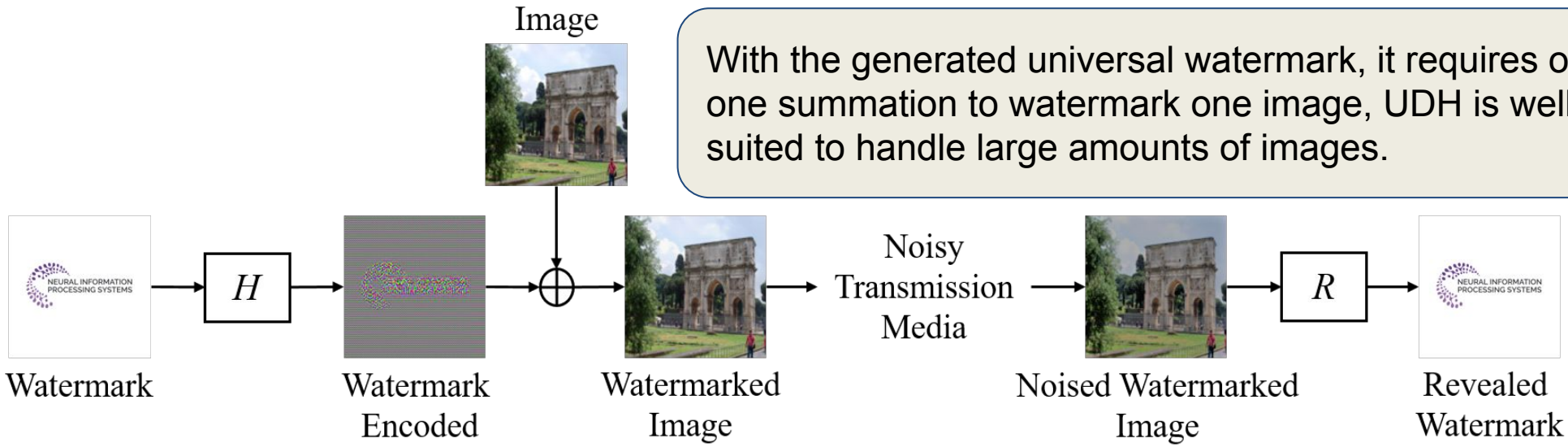
Photographic steganography or Light Field Messaging (LFM)

- Definition: Hiding and transmitting secret message by taking a photo on screen display
- Challenge: robustness to light effect

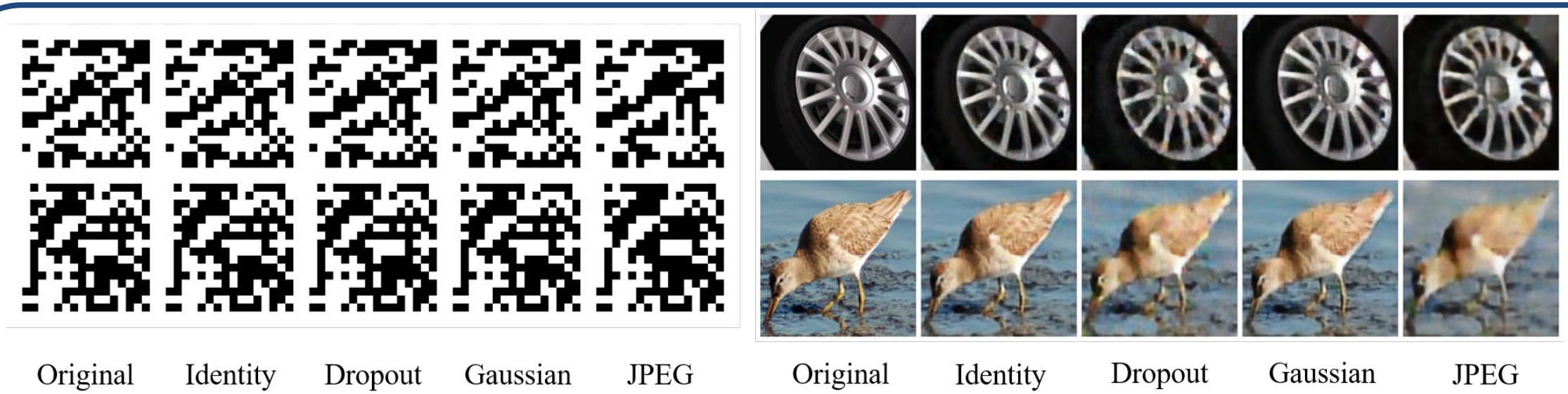
With the UDH, we are the first to demonstrate the possibility of hiding a full image for both applications.

[1] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Universal Deep Watermarking



With the generated universal watermark, it requires only one summation to watermark one image, UDH is well suited to handle large amounts of images.

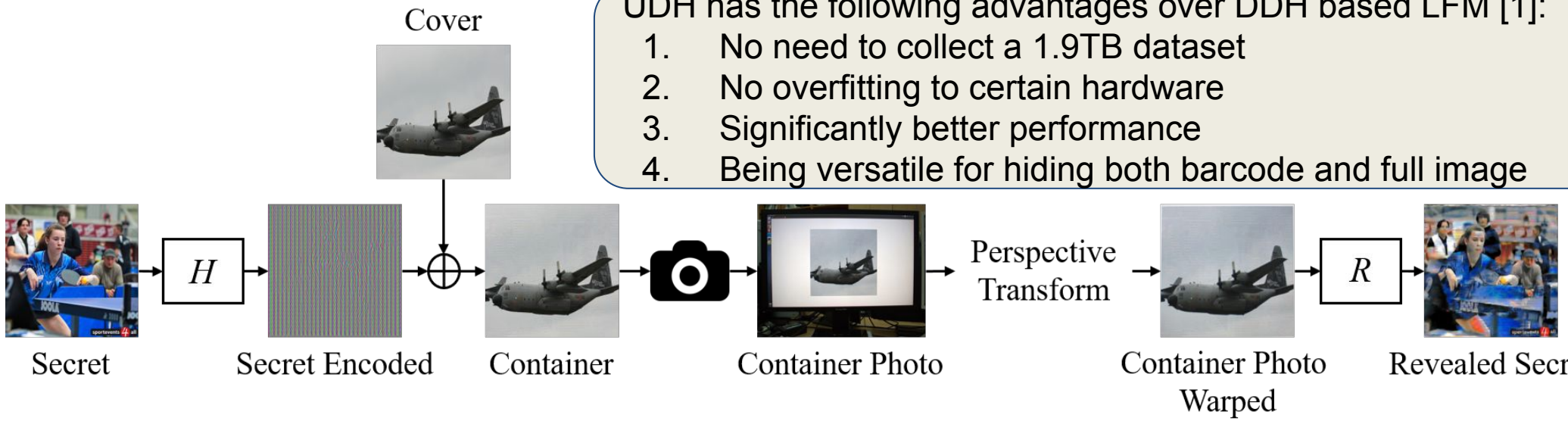


[1] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Light Field Messaging (Photographic Steganography)

UDH has the following advantages over DDH based LFM [1]:

1. No need to collect a 1.9TB dataset
2. No overfitting to certain hardware
3. Significantly better performance
4. Being versatile for hiding both barcode and full image



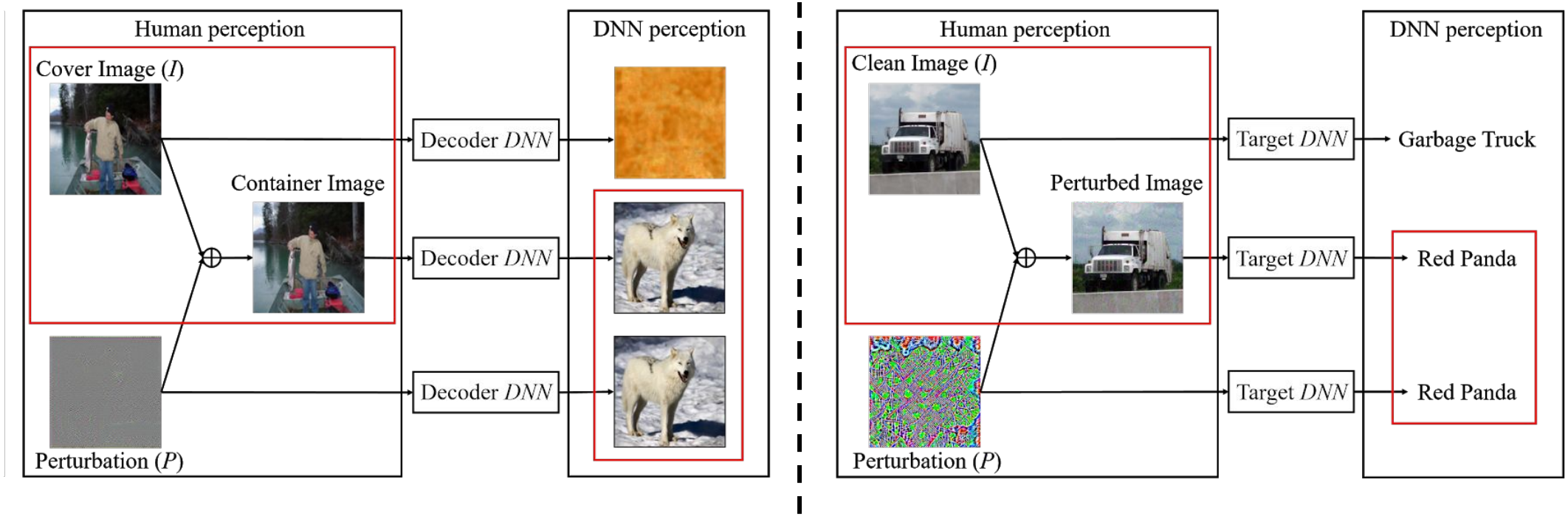
[1] Light field messaging with deep photographic steganography; Wengrowski, Dana K.; CVPR 2019
[2] UDH: Universal Deep Hiding for Steganography, Watermarking and Light Field Messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Universal Adversarial Perturbations Through the Lens of Deep Steganography: Towards a Unified Fourier Perspective (AAAI 2021)

Chaoning Zhang*, Philipp Benz*, Adil Karjauv, In-So Kweon
* indicates equal contribution

Korea Advanced Institute of Science and Technology (KAIST)

Universal Perturbations Dominate over Images



Universal Data Hiding [1]

Universal Adversarial Attack

There exists a **misalignment** between **human vision** and **model perception** for both universal data hiding and universal adversarial attack

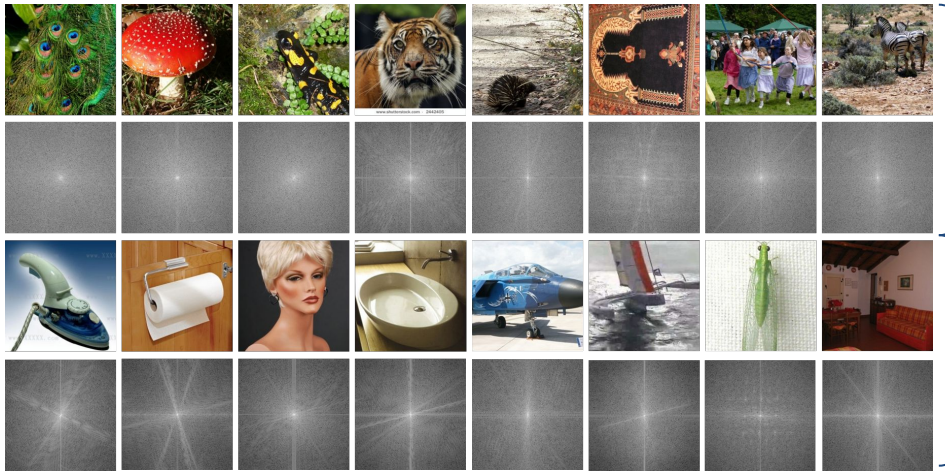
[1] UDH: Universal deep hiding for steganography, watermarking and light field messaging. Zhang*, Benz*, Karjauv*, Sun, Kweon, NeurIPS 2020.

Universal yet there exists class-wise discrepancy

For robust and non-robust classes, the targeted attack rate is **40%** and **100%** respectively.

3 most robust classes

3 least robust classes



high frequency

low frequency

Considering the most and least robust classes against adversarial attack,
the **most robust classes** have relatively **high frequency** and
the **least robust** classes have relatively **low frequency**

A Unified Frequency Perspective on UAP and UDH

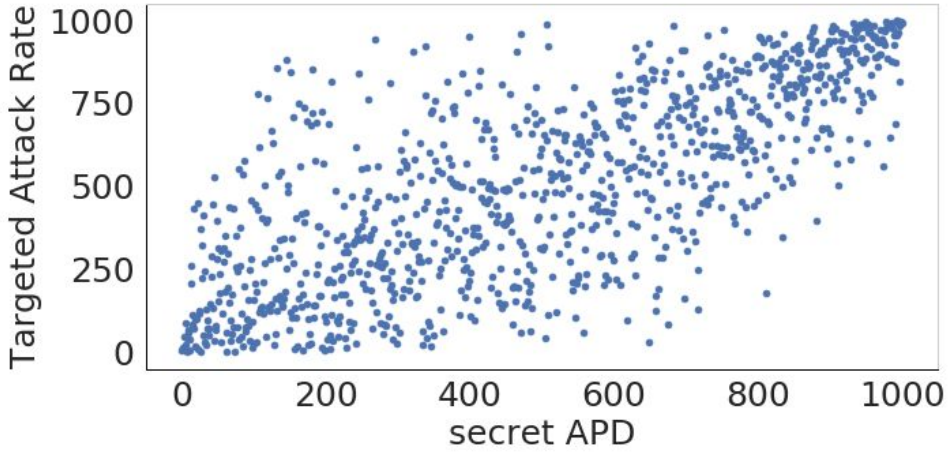
We order the 1000 classes in ImageNet based on three metrics:

(a) Targeted attack rate (for UAP)

(b) Secret APD (for UDH)

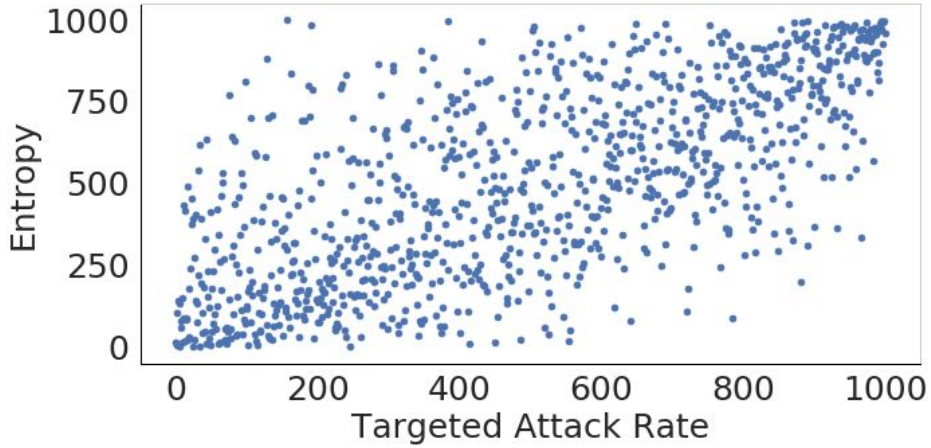
(c) Entropy of Fourier images (for frequency)

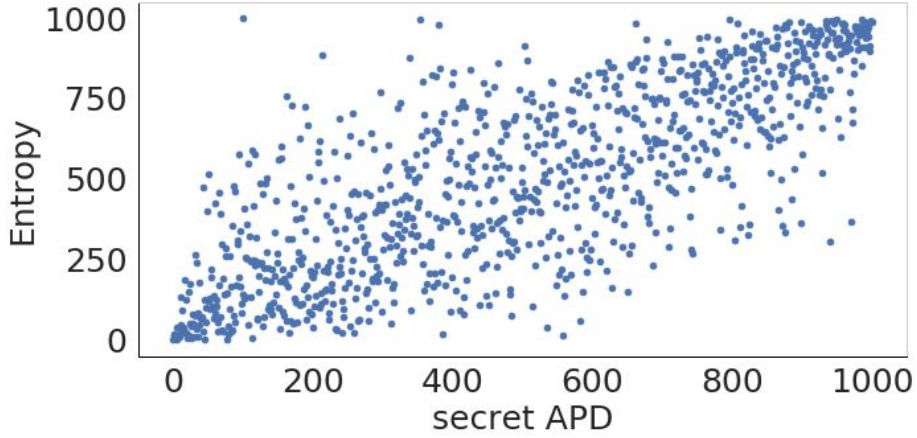
Surprisingly, we find that the ordering is highly correlated between two different tasks.



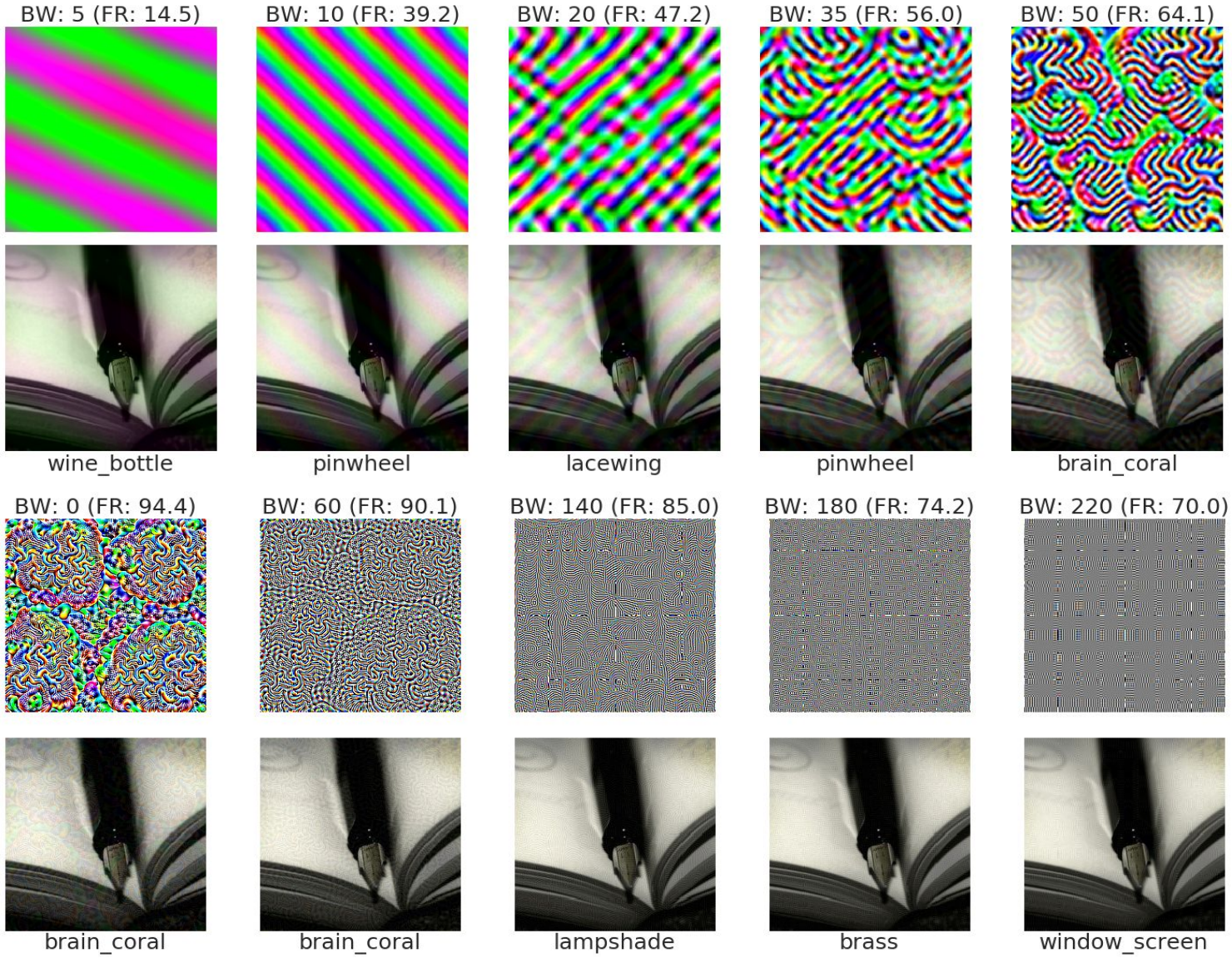
What is the factor that determines this high correlation?

Frequency! Images from HF class are systematically more robust to UAP.





Why do UAPs dominate images?

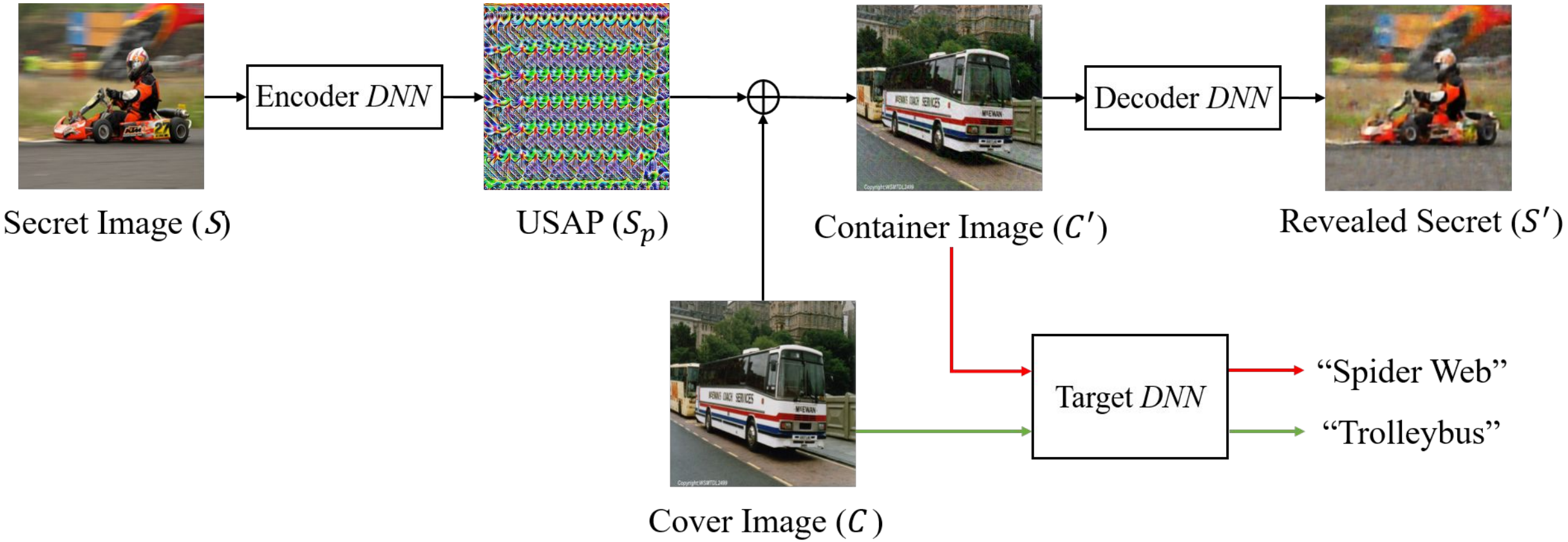


The UAP fails when HF content is removed.
On the contrary, removing LF content from the UAP has trivial effect.

UAPs (of smaller magnitude) dominate images, why?

→**DNNs are sensitive to HF content & UAPs have HF properties**

Universal Secret Adversarial Perturbation (USAP)



Takeaway

1. What you see is not what the machine sees

2. Joint Investigation and Explanation: Adversarial Attack & Data Hiding

Thank You

Philipp Benz
https://phibenz.github.io
pbenz@kaist.ac.kr

Chaoning Zhang
https://chaoningzhang.github.io
chaoningzhang1990@gmail.com