Generative Adversarial Perturbations

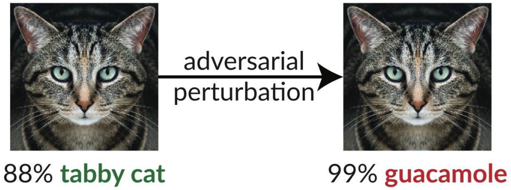
Omid Poursaeed, Isay Katsman, Bicheng Gao, Serge Belongie

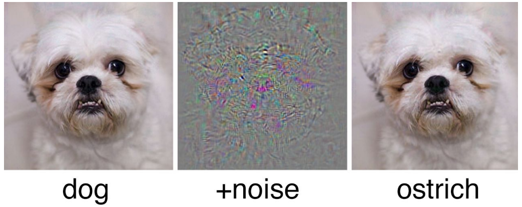
CVPR 2018

Adversarial Examples

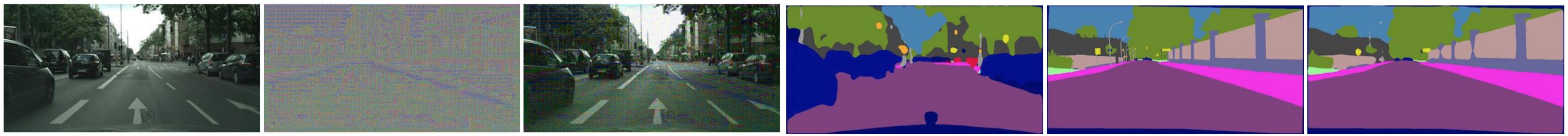
Slightly perturbed images resembling natural images but maliciously crafted to fool trained models.

- **Classification:**





- **Semantic Segmentation:**

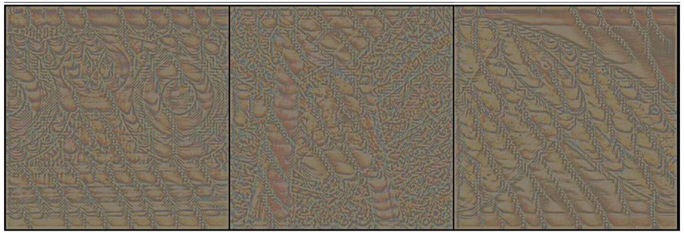


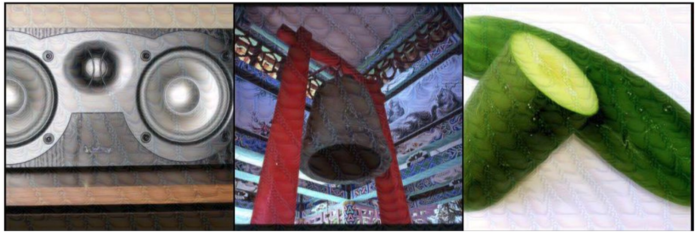
(a) Original image (b) Perturbation (c) Perturbed image (d) Prediction for original image (e) Target (f) Prediction for perturbed image

Adversarial Examples

Universal Perturbations: Universal perturbations are fixed perturbations which when added to natural images can significantly degrade the accuracy of the pre-trained network

Image-dependent Perturbations: Image-dependent perturbations can vary for different images in the dataset

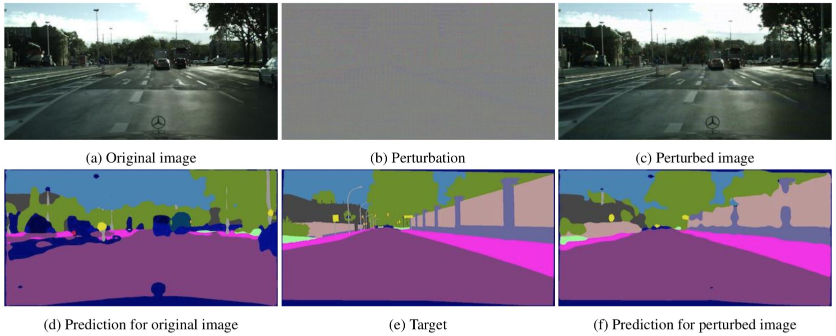




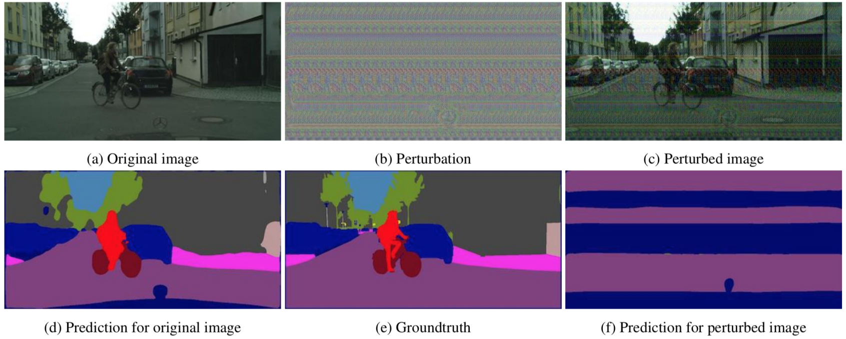
Adversarial Examples

Targeted Attacks: We seek adversarial images that can change the prediction of a model to a specific target

Non-targeted Attacks: We want to generate adversarial examples for which the model's prediction is any label other than the ground-truth



Targeted Attack



Non-targeted Attack

Problem Formulation

y_x : ground-truth label for image $x \in [0,1]^n$

n : number of pixels in the image

C : number of classes

Classification Network $\mathcal{K} : [0,1]^n \rightarrow \{1, \ldots, C\}$

Semantic Segmentation Network $\mathcal{K} : [0,1]^n \rightarrow \{1, \ldots, C\}^n$

$\delta \in [0,1]^n$: additive perturbation

t : target label

$k(x)$: Output probabilities of the network for the input x

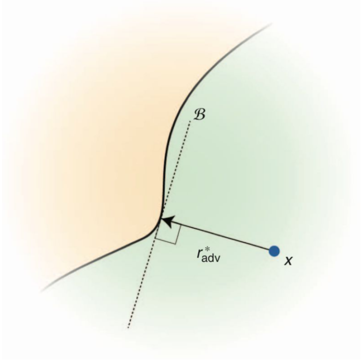
$\mathcal{K}(x) = \arg\max k(x)$

Non-targeted Perturbation:

$$\begin{aligned} &\text{Minimize}_\delta \; \|\delta\|_p \\ &\quad \text{s.t. } \mathcal{K}(x+\delta) \neq y_x \\ &\quad x+\delta \in [0,1]^n \end{aligned}$$

Targeted Perturbations:

$$\begin{aligned} &\text{Minimize}_\delta \; \|\delta\|_p \\ &\quad \text{s.t. } \mathcal{K}(x+\delta) = t \\ &\quad x+\delta \in [0,1]^n \end{aligned}$$



Often intractable optimization problem

Optimization-based methods

- Intriguing Properties of Neural Networks (Szegedy et al. 2014)

$$\text{Minimize}_\delta \ \ \|\delta\|_p + c \ . \ Loss(x+\delta, t)$$
$$\text{s.t.} \ \ x+\delta \in [0,1]^n$$

- Towards Evaluating the Robustness of Neural Networks (Carlini et al. 2017)

$$\text{Minimize}_\delta \ \ \|\delta\|_p + c \ . (log(k(x+\delta))_t - max_{i \neq t}\{log(k(x+\delta))_i\})^+$$
$$\text{s.t.} \ \ x+\delta \in [0,1]^n$$
$$k(x+\delta) : \text{output probabilities of network for input } x+\delta$$

Iteratively updates the perturbation to minimize the loss

Perform a search to find the best positive value for c

Advantage: Very good performance - **Drawback:** Slow at inference time

Fast Gradient Sign Method (FGSM)

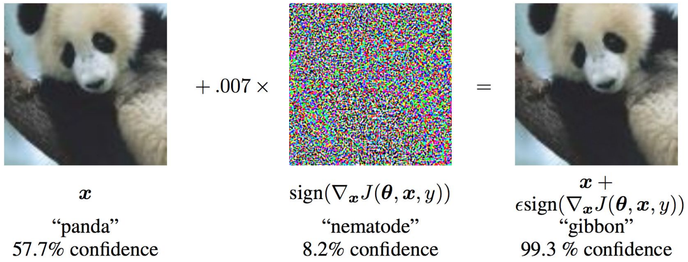
Uses a linear approximation of the loss function at the perturbed sample

$$Loss(x + \delta, y_x) \approx Loss(x, y_x) + \delta \ .\nabla_x Loss(x, y_x)$$

$$\text{Maximize}_\delta \ Loss(x, y_x) + \delta \ . \ \nabla_x Loss(x, y_x)$$

$$s.t. \ \ \|\delta\|_\infty \leq \epsilon$$

$$\Rightarrow \delta = \epsilon \ . \ sign(\nabla_x Loss(x, y_x))$$



Advantage: Fast computation (single forward and backward pass through the network)

Drawback: Linear approximation of the loss surface is not very accurate especially when the sample is far away from the decision boundary

Iterative FGSM

Applies FGSM multiple times with smaller step sizes

$$x^{(0)} = x$$
$$x^{(n+1)} = clip_{\,[x-\epsilon,\ x+\epsilon]}(x^{(n)} + \alpha \ . \ sign(\nabla_{x^{(n)}} Loss(x^{(n)}, y_x))$$

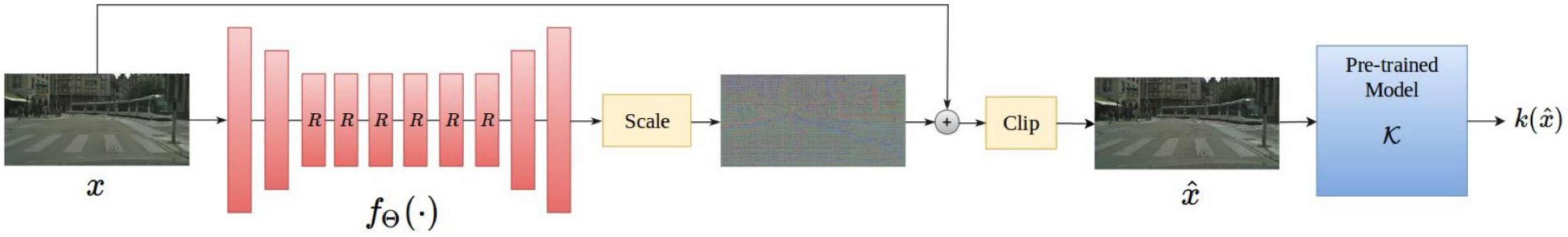
“Basic iter.”; L_∞ distance to clean image = 32

Advantage: More accurate results (better approximation of the loss surface)

Drawback: Slow at inference time (requires multiple forward and backward passes through the network)

Our Approach: Image-dependent Perturbations

Using a generator to learn the perturbation from the input image



Similar architecture can be used across different tasks (classification, segmentation, etc.)

Advantage: Fast at inference time (single forward pass through the generator)

Drawback: Needs to train an additional network

Our Approach: Image-dependent Perturbations

$$\begin{aligned}\text{Maximize}_\theta \; & Loss_{\mathcal{K}}(x + G_\theta(x), y_x) \\ \text{s.t.} \; & \|G_\theta(x)\|_p \leq \epsilon \\ & x + G_\theta(x) \in [0,1]^n\end{aligned}$$

$$\begin{aligned}\text{Minimize}_\theta \; & -\log(Loss_{\mathcal{K}}(x + G_\theta(x), y_x)) \triangleq Loss_{\mathcal{G}} \\ \text{s.t.} \; & \|G_\theta(x)\|_p \leq \epsilon \\ & x + G_\theta(x) \in [0,1]^n\end{aligned}$$

Generator's Loss Function:

- Non-targeted Attacks:

$$Loss_{\mathcal{G}}(x, y_x) \triangleq -\log Loss_{\mathcal{K}}(x, y_x)$$

$$\text{Least Likely Class Loss: } Loss_{\mathcal{G}}(x, y_x) \triangleq \log Loss_{\mathcal{K}}(x, \arg\min(k(x)))$$

- Targeted Attacks:

$$Loss_{\mathcal{G}}(x, t) \triangleq \log Loss_{\mathcal{K}}(x, t)$$

Universal Perturbations

- Universal Adversarial Perturbations (Dezfooli et al. 2017)
  - Creating the universal perturbation by adding image-dependent perturbations and scaling the result

$$X \triangleq \{x_1, \ldots, x_m\}$$

Initialize u $= 0$

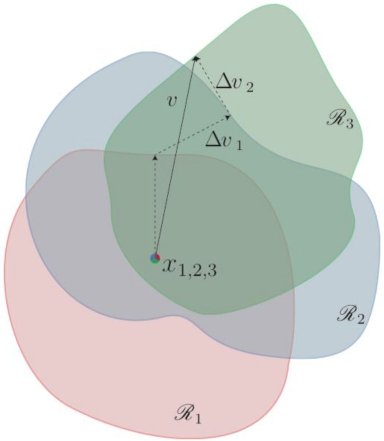
$$Err(X_u) \triangleq \frac{1}{m} \Sigma_{j=1}^{m} \mathbb{1}_{\mathcal{K}(x_j) \neq \mathcal{K}(x_j + u)}$$

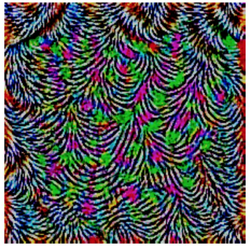
β : threshold on error rate

While $Err(X_u) \leq 1 - \beta$:

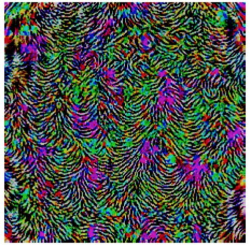
  for i in $\{1, \ldots, m\}$:

$$u \leftarrow \epsilon \frac{u + \delta_{x_i + u}}{\|u + \delta_{x_i + u}\|_p}$$





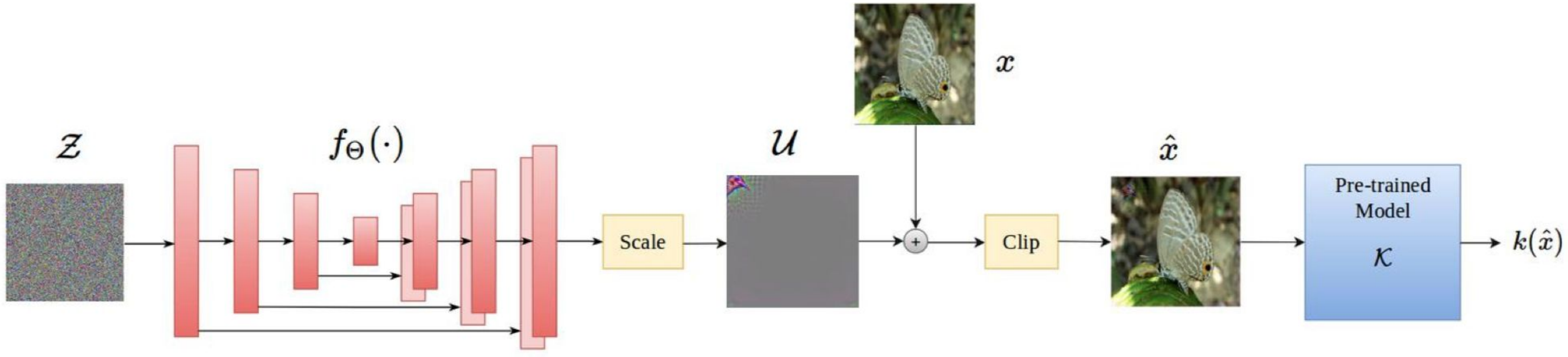
(c) VGG-16



(f) ResNet-152

Our Approach: Universal Perturbations

Transforming a randomly sampled pattern to the universal perturbation



Advantage: Improves the performance on universal perturbations

Drawback: Needs to train an additional network

Our Approach: Universal Perturbations

$$\text{Sample } z \sim \text{Uniform}[0,1]^n$$
$$u = G_\theta(z)$$
$$\begin{aligned} \text{Maximize}_\theta\ & Loss_{\mathcal{K}}(x + G_\theta(z), y_x) \\ \text{s.t.}\ & \|G_\theta(z)\|_p \leq \epsilon \\ & x + G_\theta(z) \in [0,1]^n \end{aligned}$$

$$\begin{aligned} \text{Minimize}_\theta\ & -\log(Loss_{\mathcal{K}}(x + G_\theta(z), y_x)) \triangleq Loss_{\mathcal{G}} \\ \text{s.t.}\ & \|G_\theta(z)\|_p \leq \epsilon \\ & x + G_\theta(z) \in [0,1]^n \end{aligned}$$

Generator's Loss Function:

- Non-targeted attacks:

$$Loss_{\mathcal{G}}(x, y_x) \triangleq -\log Loss_{\mathcal{K}}(x, y_x)$$
$$\text{Least Likely Class Loss: } Loss_{\mathcal{G}}(x, y_x) \triangleq \log Loss_{\mathcal{K}}(x, \arg\min(k(x)))$$

- Targeted attacks:

$$Loss_{\mathcal{G}}(x, t) \triangleq \log Loss_{\mathcal{K}}(x, t)$$

Fooling Multiple Networks

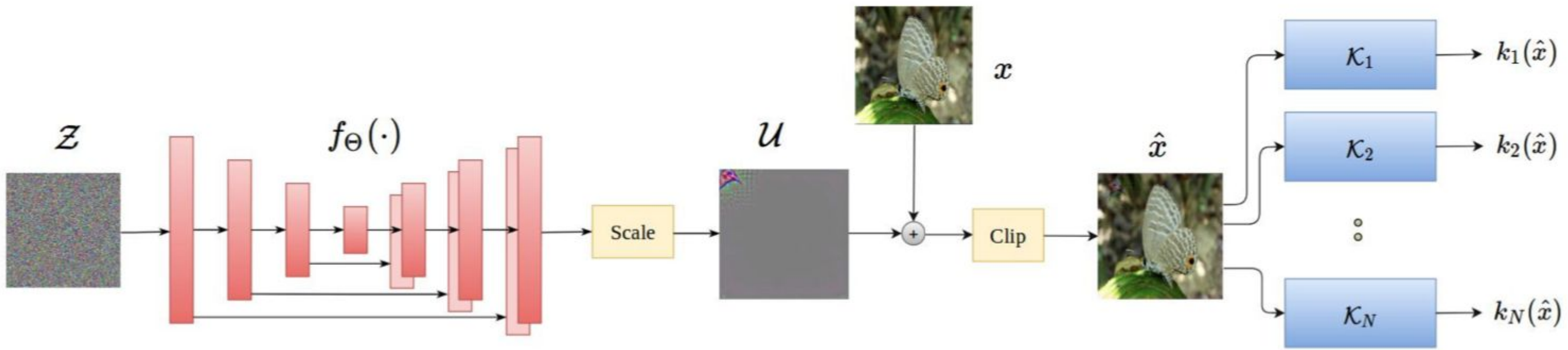


Figure 3: Architecture for training a model to fool multiple target networks. The fooling loss for training the generator is a linear combination of fooling losses of target models.

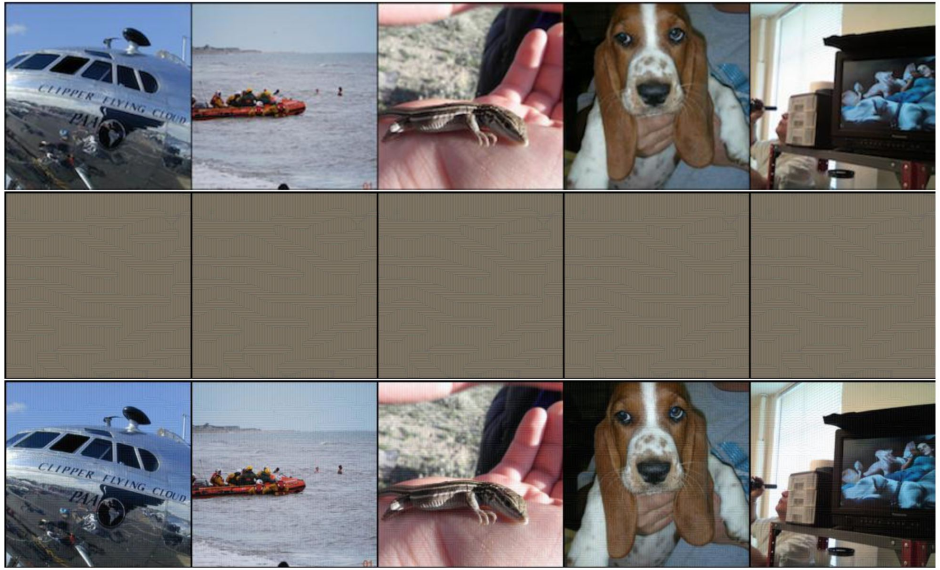
$$l_{multi-fool} = \lambda_1 \cdot l_{fool_1} + \cdots + \lambda_m \cdot l_{fool_m}$$

Results on Classification

Non-targeted Universal Perturbations:

		VGG16	VGG19	Inception[5]
$L_\infty = 10$	**GAP**	**83.7%**	**80.1%**	**82.7%**[6]
	UAP	78.8%	77.8%	78.9%

Table 2: Fooling rates of non-targeted universal perturbations using L_∞ norm as the metric.



(d) Target model: VGG-19, Fooling ratio: 80.1%

Results on Classification

Non-targeted Universal Perturbations:

		VGG16	VGG19	Inception[5]
$L_\infty = 10$	**GAP**	**83.7%**	**80.1%**	**82.7%**[6]
	UAP	78.8%	77.8%	78.9%

Table 2: Fooling rates of non-targeted universal perturbations using L_∞ norm as the metric.

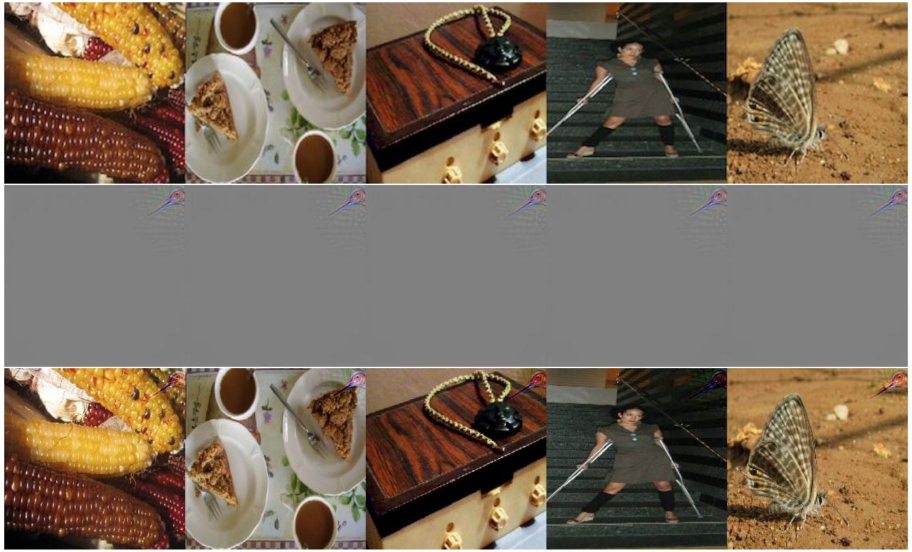
(c) Target model: Inception-v3, Fooling ratio: 79.2%

Results on Classification

Non-targeted Universal Perturbations:

		VGG16	VGG19	ResNet152
$L_2 = 2000$	**GAP**	**93.9%**	**94.9%**	79.5%
	UAP	90.3%	84.5%	**88.5%**

Table 1: Fooling rates of non-targeted universal perturbations for various classifiers pre-trained on ImageNet. Our method (GAP) is compared with Universal Adversarial Perturbations (UAP) [35] using L_2 norm as the metric.



(a) Target model: VGG-19, Fooling ratio: 94.9%

Results on Classification

Targeted Universal Perturbations

- The most challenging task
- Average target accuracy on 10 random targets: 52.0%
  - Model: Inception-v3

(a) Target: Soccer Ball, Top-1 target accuracy: 74.1%

(b) Target: Knot, Top-1 target accuracy: 63.6%

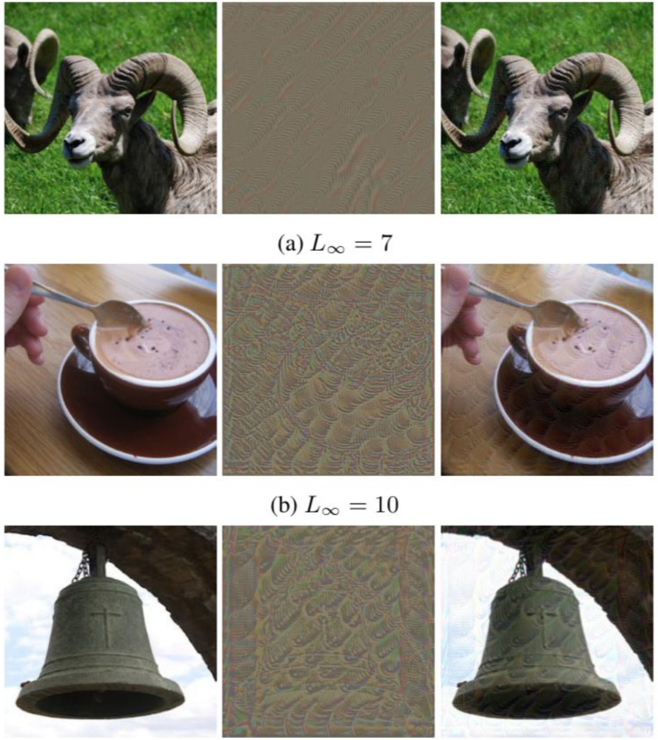
(c) Target: Finch, Top-1 target accuracy: 61.8%

Results on Classification

Non-targeted Image-dependent Perturbations

	$L_\infty = 7$	$L_\infty = 10$	$L_\infty = 13$
VGG16	66.9% (30.0%)	80.8% (17.7%)	88.5% (10.6%)
VGG19	68.4% (28.8%)	84.1% (14.6%)	90.7% (8.6%)
Inception-v3	85.3% (13.7%)	98.3% (1.7%)	99.5% (0.5%)

Table 3: Fooling ratios (pre-trained models' accuracies) for non-targeted image-dependent perturbations.



(a) $L_\infty = 7$

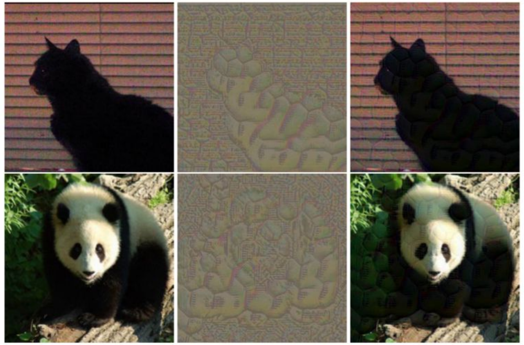
(b) $L_\infty = 10$

(c) $L_\infty = 13$

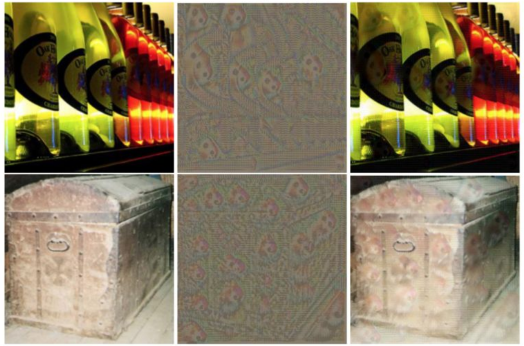
Results on Classification

Targeted Image-dependent Perturbations

- Average target accuracy on 10 random targets: 89.1%
  - Model: Inception-v3



(a) Target: Soccer Ball, Top-1 target accuracy: 91.3%



(b) Target: Hamster, Top-1 target accuracy: 87.4%

Transferability

Adversarial examples created for one network can also fool other networks

Decision boundaries of different networks are correlated

	VGG16	VGG19	ResNet152
VGG16	**93.9%**	89.6%	52.2%
VGG19	88.0%	**94.9%**	49.0%
ResNet152	31.9%	30.6%	**79.5%**
VGG16 + VGG19	90.5%	90.1%	**54.1%**

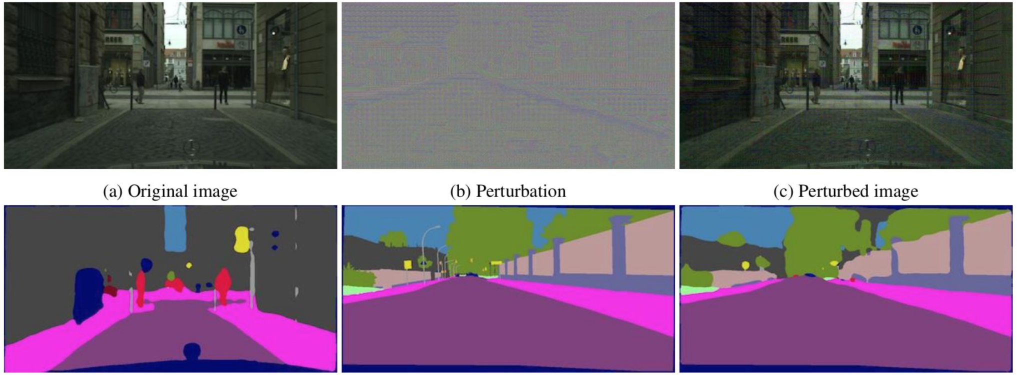
Table 4: Transferability of non-targeted universal perturbations. The network is trained to fool the pre-trained model shown in each row, and is tested on the model shown in each column. Perturbation magnitude is set to $L_2 = 2000$. The last row indicates joint training on VGG-16 and VGG-19.

Results on Semantic Segmentation

Targeted Universal Perturbations

	$L_\infty = 5$	$L_\infty = 10$	$L_\infty = 20$
GAP (Ours)	79.5%	**92.1%**	**97.2%**
UAP-Seg [34]	**80.3%**	91.0%	96.3%

Table 5: Success rate of targeted universal perturbations for fooling the FCN-8s segmentation model. Results are obtained on the validation set of the Cityscapes dataset.



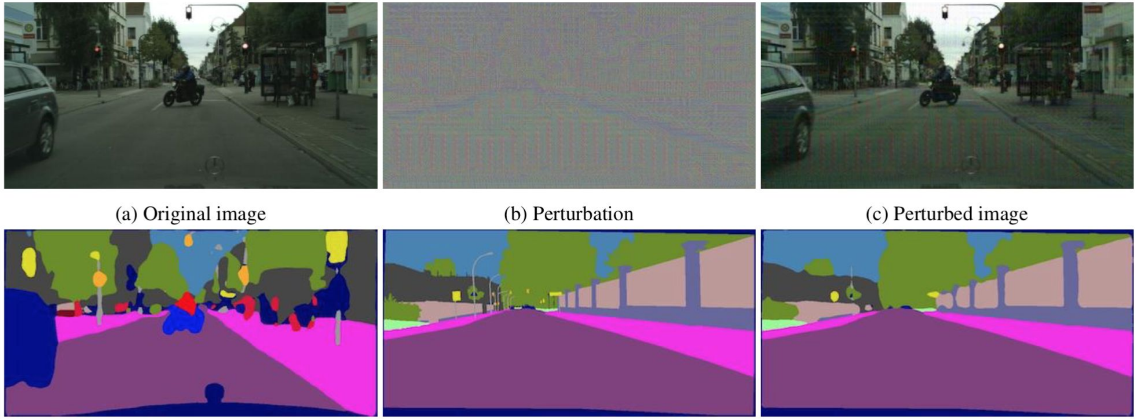
(a) Original image (b) Perturbation (c) Perturbed image
(d) Prediction for original image (e) Target (f) Prediction for perturbed image

Results on Semantic Segmentation

Targeted Image-dependent Perturbations

	$L_\infty = 5$	$L_\infty = 10$	$L_\infty = 20$
GAP	87.0%	96.3%	98.2%

Table 7: Success rate of targeted image-dependent perturbations for fooling FCN-8s on the Cityscapes dataset.



(a) Original image (b) Perturbation (c) Perturbed image

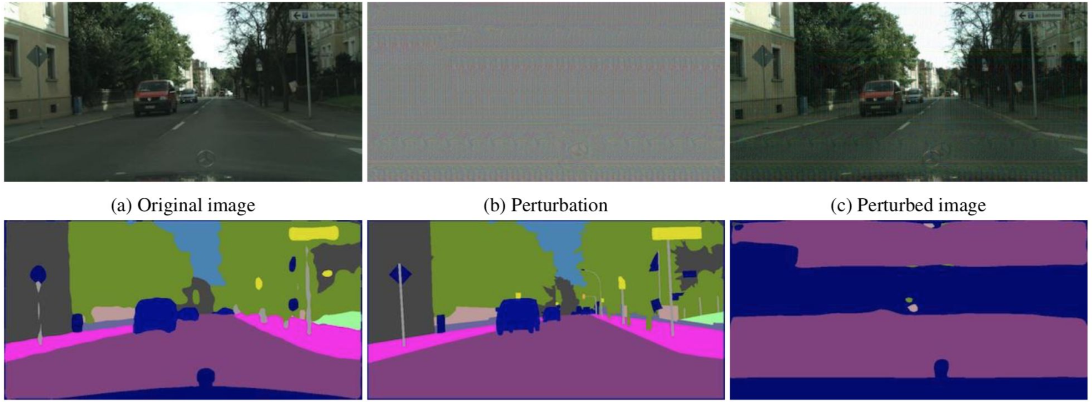
(d) Prediction for original image (e) Target (f) Prediction for perturbed image

Results on Semantic Segmentation

Non-targeted Perturbations

Task	$L_\infty = 5$	$L_\infty = 10$	$L_\infty = 20$
Universal	12.8%	4.0%	2.1%
Image-dependent	6.9%	2.1%	0.4%

Table 6: Mean IoU of non-targeted perturbations for fooling the FCN-8s segmentation model on the Cityscapes dataset.



(a) Original image (b) Perturbation (c) Perturbed image

(d) Prediction for original image (e) Groundtruth (f) Prediction for perturbed image

Figure 21: Non-targeted universal perturbations with $L_\infty = 10$.

Runtime Analysis

Task	Architecture	Titan Xp	Tesla K40
Non-targeted	ResNet Gen. 6 blocks, 50 filters	0.27 ms	4.7 ms
Targeted	ResNet Gen. 6 blocks, 57 filters	0.28 ms	4.8 ms

Table 8: Average inference time per image and generator's architecture for image-dependent classification tasks. Target model is Inception-v3.

Architecture	Titan Xp	Tesla K40m
U-Net Generator: 8 layers, 200 filters	132.8 ms	511.7 ms
ResNet Generator: 9 blocks, 145 filters	335.7 ms	2396.9 ms

Table 9: Average inference time per image and generator's architecture for the semantic segmentation task. Targeted image-dependent perturbations are considered with FCN-8s as the pre-trained model.

Resistance to Gaussian Blur

Destruction Rate: Fraction of images that are no longer misclassified after blur

	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$	$\sigma = 1.25$
GAP	0.0%	0.8%	3.2%	8.0%
I-FGSM	0.0%	0.5%	8.0%	23.0%

Table 10: Destruction Rate of non-targeted image-dependent perturbations for the classification task. Perturbation norm is set to $L_\infty = 16$.

	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$	$\sigma = 1.25$
$L_\infty = 5$	83.2%	76.9%	66.0%	57.1 %
$L_\infty = 10$	94.8%	90.1%	80.0%	69.6%
$L_\infty = 20$	97.5%	95.7%	89.3%	78.8%

Table 11: Success rate of targeted image-dependent perturbations for the segmentation task after applying Gaussian filters.

Contributions

- We present a unifying framework for creating universal and image-dependent perturbations for both classification and semantic segmentation tasks

- Improve the state-of-the-art performance in universal perturbations by leveraging generative models instead of current iterative methods.

- Present the first effective targeted universal perturbations. (This is the most challenging task as we are constrained to have a single perturbation pattern and the prediction should match a specific target).

- Our attacks are considerably faster than iterative and optimization-based methods at inference time. We can generate perturbations in the order of milliseconds.