Minimally Distorted Interpretable Adversarial Attacks

Shpresim Sadiku

(Technische Universität Berlin & Zuse Institute Berlin)

RIKEN AIP Seminar · August 28, 2023

Outline

- Adversarial Examples in Image Classification
- Adversarial Attack Generation
- Sparse Adversarial Attacks
  - Structured Sparsity
- Proximal Operator of $\|\cdot\|_p^p$ for $p = \frac{1}{2}$
- Group-wise Sparsity
  - Group-wise Sparse Attacks Generation
- Evaluation
- Results
  - CIFAR-10 Results
  - NIPS Results
- Visual Comparison
- Interpretability
- Speed

Joint work with...

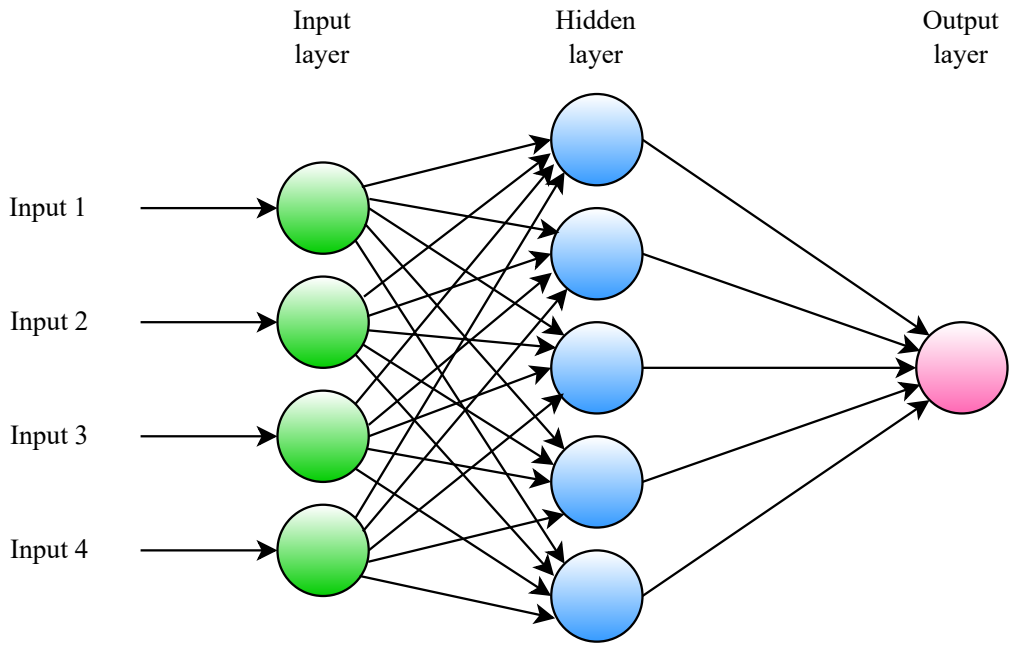
Moritz Wagner
TU Berlin/ZIB

Sebastian Pokutta
TU Berlin/ZIB

Deep Neural Networks for Image Classification

Image Classifiers (DNNs)

- High success rate
- **Robustness?**



Adversarial Attacks in Image Classification

- Image space: $\mathcal{X} = [I_{min}, I_{max}]^{C \times M \times N}$
- Classifier: mapping $\mathcal{K} : \mathcal{X} \to \{1, ..., L\}$
- Implemented by

$$\mathcal{K}(\mathbf{x}) = \underset{k=1,\ldots,L}{\arg\max}\, F_k(\mathbf{x})$$

  for some mapping $F : \mathcal{X} \to \mathbb{R}^L$

- **Adversarial examples:**

correctly classified image + small perturbation = incorrectly classified image

$$\mathbf{x} + \mathbf{w} = \mathbf{y}, \quad \|\mathbf{w}\|_{\ell_\mathbf{p}} < \epsilon$$

visually indistinguishable

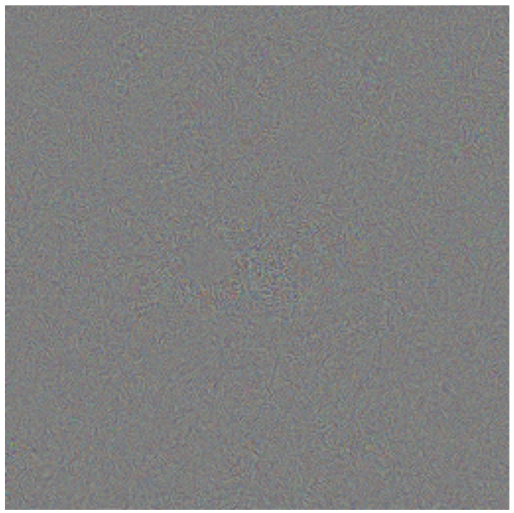
but

$$\mathcal{K}(\mathbf{x}) \neq \mathcal{K}(\mathbf{y})$$

Spot the difference

Original
Label: 986 (daisy)

Perturbation scaled by 15
$\varepsilon = 0.03$



PGD adversarial example
Prediction: 524 (crutch)

Adversarial Attack Generation

- White-box attack (F is known)
- Input image $\mathbf{x} \in \mathcal{X}$, correct label $l \in \mathbb{N}$
- Target label $t \in \mathbb{N}, t \neq l$
- Goal of an adversary: succeed under minimal distortion

$$\min_{\mathbf{x}_{adv}:\mathcal{K}(\mathbf{x}_{adv})\neq l} \|\mathbf{x}_{adv} - \mathbf{x}\|_p$$

- $\mathcal{L} : \mathcal{X} \times \mathbb{N} \to \mathbb{R}$ classification loss function (e.g. cross-entropy loss)
- Approximate the constrained minimization problem by its Lagrangian formulation

$$\min_{\mathbf{x}_{adv}\in\mathcal{X}} \mathcal{L}(\mathbf{x}_{adv}, t) + \lambda\|\mathbf{x}_{adv} - \mathbf{x}\|_p^p$$

  for $\lambda > 0$
- If $\mathbf{w} := \mathbf{x}_{adv} - \mathbf{x}$

$$\min_{\mathbf{w}:\mathbf{x}+\mathbf{w}\in\mathcal{X}} \mathcal{L}(\mathbf{x} + \mathbf{w}, t) + \lambda\|\mathbf{w}\|_p^p$$

Sparse Adversarial Attacks

- Most methods solve the problem limited for values of $p \geq 1$
- Existing sparse adversarial attacks
  - Greedy approaches that pick only one pixel at a time
  - Local search
  - Evolutionary Algorithms
  - Relaxation of combinatorial problem ℓ_0 via ℓ_1 ball

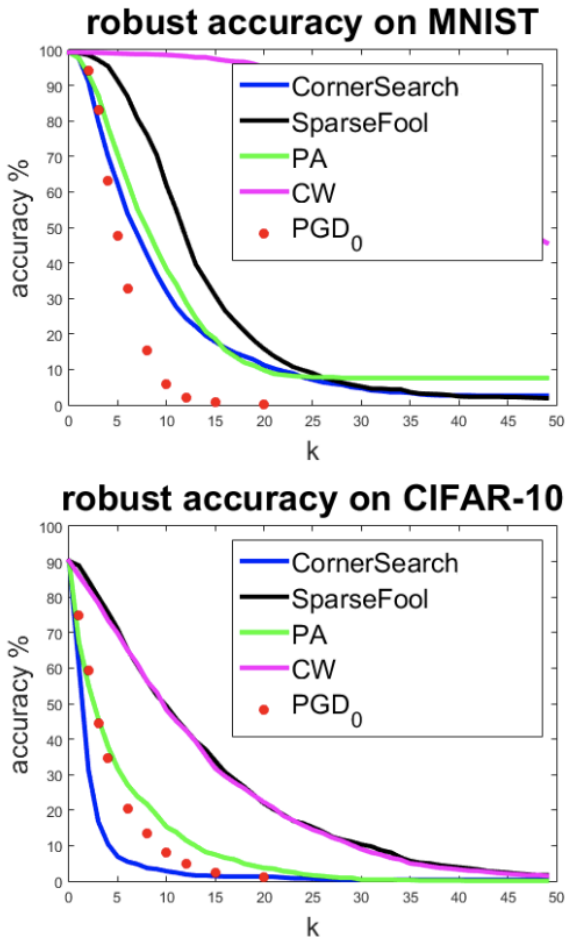


Figure 1: Robust accuracy of classifier when the attack is allowed to perturb at most k pixels (Croce and Hein, 2019)

Limitations and Pitfalls

Problems encountered with existing methods

1. Fail to produce attacks that simultaneously have high sparsity and low magnitude of the changed pixels
2. Overly complicated/slow techniques

Question

Can we develop a method that is simple, sparse **and** interpretable?

Non-convex ℓ_p norm perturbations

- Can we generate adversarial examples with a non-convex loss in a non-convex ℓ_p neighborhood of the input image?
  - Idea: Step away ℓ_0 combinatorial problem while remaining as continuous but sparser than ℓ_1 ball
  - For $0 < p < 1$

$$\|\mathbf{w}\|_p = \left(\sum_i |w_i|^p \right)^{\frac{1}{p}}$$

    is a quasi-norm

- How do we find

$$\hat{\mathbf{w}} \in \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{x} + \mathbf{w}, t) + \lambda \|\mathbf{w}\|_p^p$$

so that $\mathbf{x}_{adv} = \text{clip}_{\mathcal{X}}(\mathbf{x} + \hat{\mathbf{w}})$?

Sparse Adversarial Attack Generation

- Forward-backward splitting algorithm to find $\hat{\mathbf{w}}$

Forward-Backward Splitting Algorithm

Require: Image $\mathbf{x} \in \mathcal{X}$, target label t, loss function $\mathcal{L}$, sparsity parameter $\lambda > 0$, step sizes α_k, number of iterations K

1. Initialize $\mathbf{w}^{(0)} = \mathbf{0}$
2. **for** $k = 1, ..., K$ **do**
3. $\quad \mathbf{w}_{k+1} = \text{prox}_{\alpha_k \lambda \|\cdot\|_p^p} \left(\mathbf{w}_k - \alpha_k \nabla_{\mathbf{w}_k} \mathcal{L}(\mathbf{x} + \mathbf{w}_k, t) \right)$
4. **end for**
5. **return** $\hat{\mathbf{w}} = \mathbf{w}_K$

- Closed-form solution for the proximal operator of $\| \cdot \|_p^p$ for $p = \frac{1}{2}$ (Cao et al., 2013)

Proximal operator of $\|\cdot\|_p^p$

- Recall the definition of the proximal operator

$$\text{prox}_{\alpha\lambda\|\cdot\|_p^p}(\mathbf{z}) := \arg\min_y \frac{1}{2\alpha\lambda}\|\mathbf{z}-\mathbf{y}\|^2 + \|\mathbf{y}\|_p^p$$

  $\alpha > 0$, $\lambda > 0$ given parameters

- $\|\cdot\|_p^p$ is a separable function
  $\hookrightarrow$ Sufficient to solve the proximal operator when $n = 1$

$$\left[\text{prox}_{\alpha\lambda\|\cdot\|_p^p}(\mathbf{z})\right]_i = \begin{cases} \frac{2}{3}|z_i|\left(1+\cos(\frac{2\pi}{3} - \frac{2\phi_{2\alpha\lambda}z_i}{3})\right), & \text{if } z_i > g(2\alpha\lambda) \\ 0, & \text{if } |z_i| \leq g(2\alpha\lambda) \\ -\frac{2}{3}|z_i|\left(1+\cos(\frac{2\pi}{3} - \frac{2\phi_{2\alpha\lambda}z_i}{3})\right), & \text{if } z_i < -g(2\alpha\lambda) \end{cases}$$

  with

$$\phi_{2\alpha\lambda} = \arccos\left(\frac{2\alpha\lambda}{8}\left(\frac{|z_i|}{3}\right)^{-\frac{3}{2}}\right), \quad g(2\alpha\lambda) = \frac{\sqrt[3]{54}}{4}(2\alpha\lambda)^{\frac{2}{3}}$$

Group-wise Sparsity

- Group-wise sparse adversarial attacks
  - Group lasso penalty (Xu et al., 2019)
  - Nuclear group norm to impose structure (Kazemi et al., 2023)

- Use a vector of trade-off parameters $\lambda \in \mathbb{R}_{\geq 0}^{C \times M \times N}$ instead of a single parameter
  - Adjust each entry separately
  - Decrease $\lambda_{:,i,j}$ for pixels (i, j) close to already perturbed pixels

$$\left[\operatorname{prox}_{\lambda \|\cdot\|_p^p}(\mathbf{z})\right]_i := \left[\operatorname{prox}_{\lambda_i \|\cdot\|_p^p}(\mathbf{z})\right]_i$$

- After computing an iterate

$$\mathbf{w}^{(k)} = \operatorname{prox}_{\lambda^{(k-1)} \alpha_{k-1} \|\cdot\|_p^p}\left(\mathbf{w}^{(k-1)} - \alpha_{k-1} \nabla_{\mathbf{w}^{(k-1)}} \mathcal{L}(\mathbf{x} + \mathbf{w}^{(k-1)}, t)\right)$$

  we adjust $\lambda^{(k)}$ according to `AdjustLambda`

AdjustLambda

- Build a mask

$$\mathbf{m} = \operatorname{sign}\left(\sum_{c=1}^{C} |\mathbf{w}^{(k)}|_{c,:,:}\right) \in \{0,1\}^{M\times N}$$

- $2D$-convolve $\mathbf{m}$ with Gaussian blur kernel $\mathbf{K} \in \mathbb{R}^{n\times n}$

$$\mathbf{M} = \mathbf{m} * *\mathbf{K} \in [0,1]^{M\times N}$$

- Construct

$$\overline{\mathbf{M}}_{ij} = \begin{cases} \mathbf{M}_{ij} + 1 & \text{if } \mathbf{M}_{ij} \neq 0, \\ q & \text{else,} \end{cases}$$

  for $0 < q \leq 1$

- Compute trade-off parameters for next iteration

$$\lambda_{:,i,j}^{(k+1)} = \frac{\lambda_{:,i,j}^{(0)}}{\overline{\mathbf{M}}_{i,j}}.$$

`FixLambda`

- Update λ only up to some iteration $\hat{k}$ to achieve better sparsity and less clusters
- For all iterations after that, fix vector of trade-off parameters (`FixLambda`) to $\hat{\lambda}$

$$\hat{\lambda}_{:,i,j} = \begin{cases} 0, & \text{if } \lambda^{(\hat{k})}_{:,i,j} < \lambda^{(0)}_{:,i,j} \\ \infty, & \text{otherwise.} \end{cases}$$

- Perturbations for pixels (i, j) with $\hat{\lambda}_{:,i,j} = \infty$ will stay at 0

Group-Wise Sparse Adversarial Attacks

Group-Wise Sparse Adversarial Attack Algorithm

Require: Image $\mathbf{x} \in \mathcal{X}$, target label t, loss function $\mathcal{L}$, sparsity parameter $\lambda > 0$, step sizes α_k, number of iterations $\hat{k}, K$

1. Initialize $\mathbf{w}^{(0)} = \mathbf{0}, \quad \lambda^{(0)} = \lambda \mathbf{1}$
2. **for** $k = 1, ..., \hat{k}$ **do**
3. $\quad \mathbf{w}_{k+1} = \text{prox}_{\alpha_k \lambda^{(k)} \|\cdot\|_p^p} (\mathbf{w}_k - \alpha_k \nabla_{\mathbf{w}_k} \mathcal{L}(\mathbf{x} + \mathbf{w}_k, t))$
   $\quad \lambda^{(k+1)} = \texttt{AdjustLambda}(\lambda^{(0)}, \mathbf{w}^{(k+1)})$
4. **end for**
5. $\hat{\lambda} = \texttt{FixLambda}(\lambda^{(\hat{k})}, \lambda^{(0)})$
6. **for** $k = \hat{k} + 1, ..., K$ **do**
7. $\quad \mathbf{w}_{k+1} = \text{prox}_{\alpha_k \hat{\lambda} \|\cdot\|_p^p} (\mathbf{w}_k - \alpha_k \nabla_{\mathbf{w}_k} \mathcal{L}(\mathbf{x} + \mathbf{w}_k, t))$
8. **end for**
9. **return** $\hat{\mathbf{w}} = \mathbf{w}_K$

Group-wise Sparsity in Action

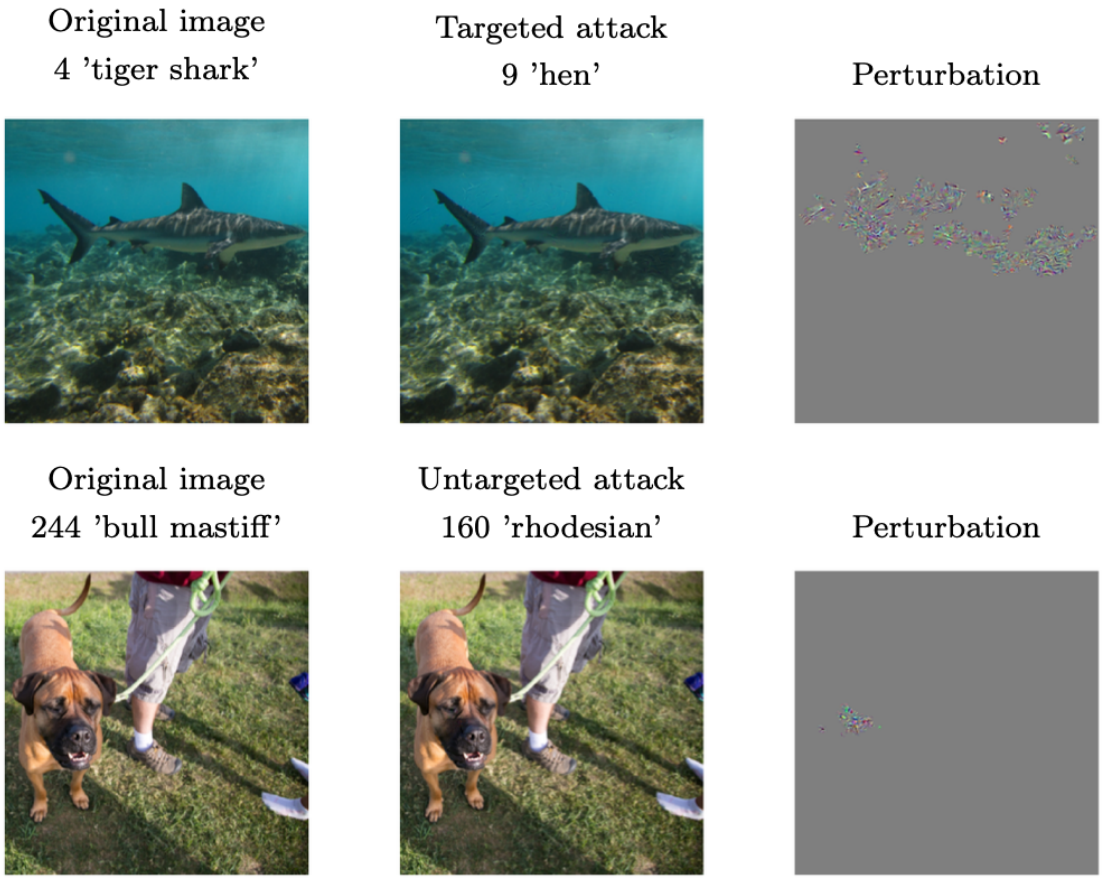


Figure 2: Examples of adversarial examples generated by our attack.

Evaluation

- Input images: $(\mathbf{x}^{(i)})_{0<i\leq n}$
- Corresponding perturbations: $(\delta^{(i)})_{0<i\leq n}$
- Number of successful adversarial examples: $m \leq n$

- Attack Success Rate (**ASR**)

$$\text{ASR} = \frac{m}{n}$$

- Average number of changed pixels (**ACP**)

$$\text{ACP} = \frac{1}{n}\sum_{i=1}^{n}\frac{\|\Delta^{(i)}\|_0}{MN}, \quad \Delta^{(i)} = \frac{1}{c}\sum_{j=1}^{C}|\delta^{(i)}_{[:,:,j]}| \in \mathbb{R}^{M\times N}$$

Evaluation

- Compute average number of clusters (**ANC**) of perturbed pixels
  1. Build a mask

$$\mathbf{m}^{(i)} = \text{sign}\left(\sum_{c=1}^{C} |\delta^{(i)}|_{c,:,:}\right) \in \{0,1\}^{M \times N}$$

  2. Run depth-first search (DFS) on $\mathbf{m}$
     - Treat adjacent $1-$entries as neighboring nodes
  3. Rerun DFS starting from every $1-$entry that another DFS run has not yet discovered
  4. Number of DFS runs until all $1-$entries are discovered is number of clusters
  5. Compute ANC for n adversarial examples

- Compute ANC on perturbations that have been blurred by a 3×3 Gaussian blur kernel - $\mathbf{ANC}_{\text{blur}}$

CIFAR-10 Results

Attack	ASR	ACP	ANC	ANC_{blur}
Ours	100%	91.4	1.8	1.3
StrAttack	100%	116.1	4.7	2.1
FWnucl	95.6%	456.4	1.3	1.3

Figure 3: Comparison of untargeted StrAttack, FWnucl, and our attack. Tested on 1000 images from the CIFAR-10 dataset with a ResNet20 model.

	Attack	ASR	ACP	ANC	ANC_{blur}
Best Case	Ours	100%	81.6	1.0	1.0
	StrAttack	100%	85.3	2.3	1.0
	FWnucl	100%	278.1	1.0	1.0
Average Case	Ours	100%	226.9	1.8	1.2
	StrAttack	100%	228.0	5.5	1.9
	FWnucl	82.6%	392.4	1.6	1.5
Worst Case	Ours	100%	409.6	3.4	2.0
	StrAttack	100%	395.7	9.3	3.7
	FWnucl	41.1%	468.3	2.7	2.2

Figure 4: Comparison of targeted StrAttack, FWnucl, and our attack. Tested on 100 images from the CIFAR-10 dataset with 9 target labels each and with a ResNet20 model.

NIPS Results

Model	Attack	ASR	ACP	ANC	ANC_{blur}
ResNet50	Ours	100%	1809.5	10.6	5.6
	StrAttack	100%	8733.5	12.9	11.2
	FWnucl	55.1%	14725.8	3.7	2.7

Figure 5: Comparison of untargeted StrAttack, FWnucl, and our attack. Tested on the NIPS2017 dataset.

Model		Attack	ASR	ACP	ANC	ANC_{blur}
ResNet50	Best Case	Ours	100%	1986.4	2.6	2.4
		StrAttack	100%	5869.5	4.4	4.3
		FWnucl	31.8%	6531.2	2.4	2.2
	Average Case	Ours	100%	5876.3	14.0	7.1
		StrAttack	100%	14348.6	15.5	13.1
		FWnucl	24.4%	18461.2	10.1	7.3
	Worst Case	Ours	100%	11929.5	44.8	13.6
		StrAttack	100%	25262.2	22.0	19.3
		FWnucl	11.9%	35972.8	32.0	15.8

Figure 6: Comparison of targeted StrAttack, FWnucl, and our attack. Tested on 100 images from the NIPS2017 dataset with 10 target labels each.

Visual comparison

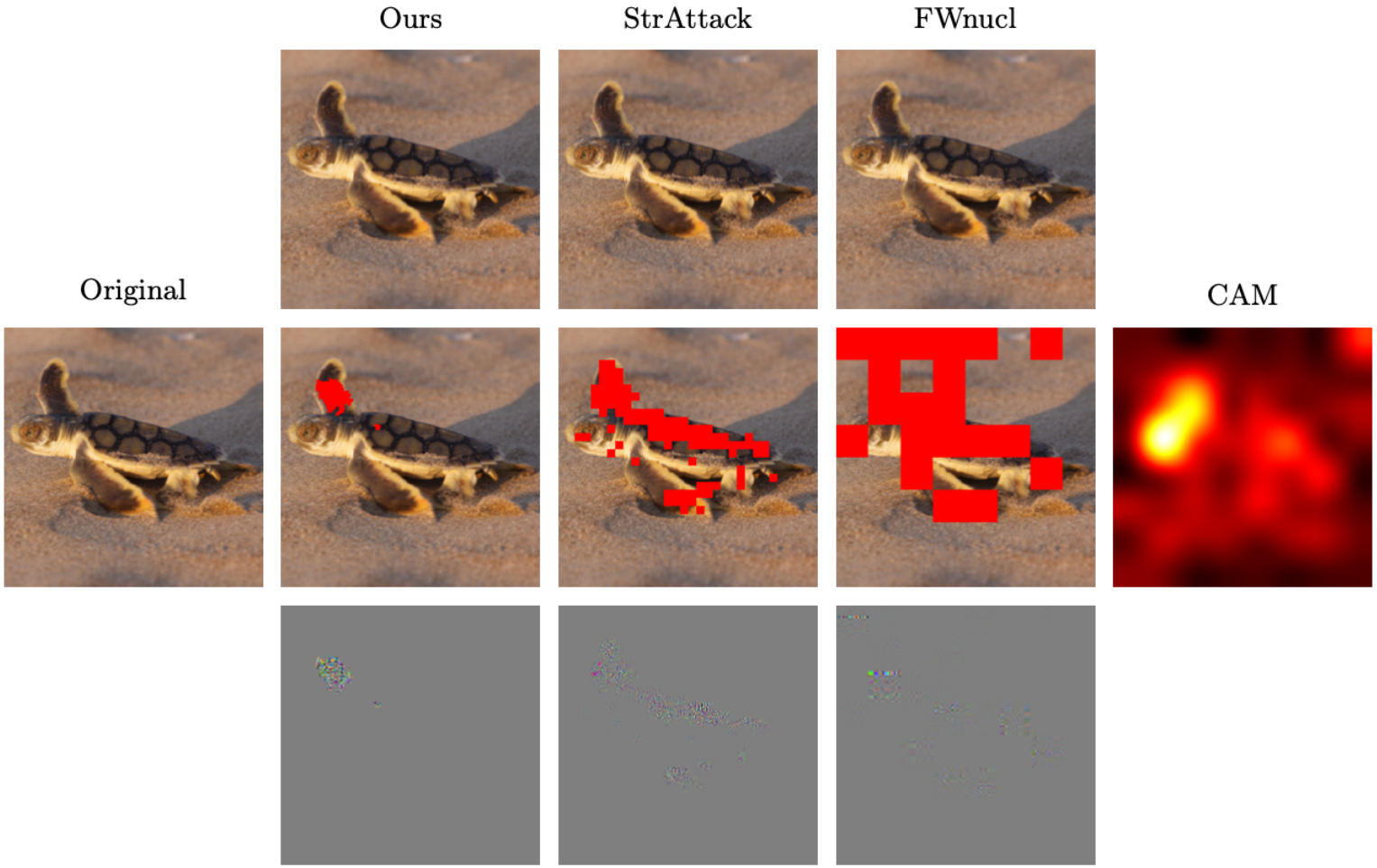


Figure 7: Visual comparison of successful, untargeted adversarial examples for our attack, StrAttack, and FWnucl. (Top row) adversarial examples, (middle row) perturbed pixels highlighted in red, (bottom row) perturbations scaled by 5.

Adversarial Saliency Map (ASM)

- $\mathbf{x} \in \mathbb{R}^d$ vectorized image
- True label t_0, target label t
- $Z(\mathbf{x})$ the logits of a classifier
- Adversarial Saliency Map (ASM) (Xu et al., 2019)

$$\mathrm{ASM}(\mathbf{x}, t)[i] := \begin{cases} 0 & \text{if } \frac{\partial Z(\mathbf{x})_t}{\partial \mathbf{x}_i} < 0 \text{ or } \frac{\partial Z(\mathbf{x})_{t_0}}{\partial \mathbf{x}_i} > 0 \\ \left(\frac{\partial Z(\mathbf{x})_t}{\partial \mathbf{x}_i}\right) \left|\frac{\partial Z(\mathbf{x})_{t_0}}{\partial \mathbf{x}_i}\right| & \text{otherwise} \end{cases}$$

  - The higher the value of $\mathrm{ASM}(\mathbf{x}, t) \in \mathbb{R}^d_{\geq 0}$, the more important the pixel
- Compute a binary mask $\mathbf{B}_{ASM} \in \{0, 1\}^d$ by

$$\mathbf{B}_{ASM}[i] := \begin{cases} 1 & \text{if } \mathrm{ASM}(\mathbf{x}, t)[i] > \nu \\ 0 & \text{otherwise} \end{cases}$$

  where ν is some percentile of the entries of $\mathrm{ASM}(\mathbf{x}, t)$
- Given an adversarial perturbation $\mathbf{x}_{adv} - \mathbf{x} =: \delta \in \mathbb{R}^d$, compute the interpretability score (**IS**)

$$\mathrm{IS}(\delta) = \frac{\|\mathbf{B}_{ASM} \odot \delta\|}{\|\delta\|}.$$

Interpretability

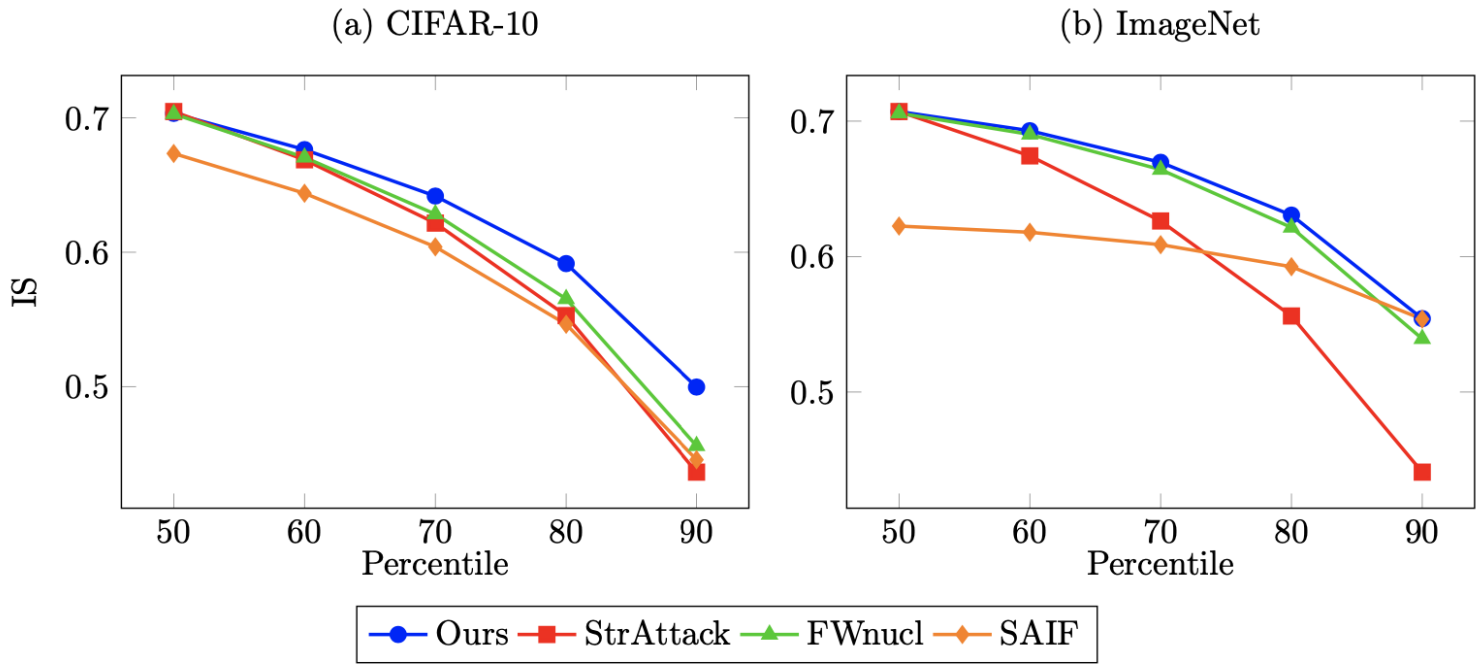


Figure 8: IS vs. percentile for targeted versions of our attack, StrAttack, FWnucl, and SAIF. Evaluated with a CIFAR-10 ResNet20 classifier (a) and an ImageNet ResNet50 classifier (b).

Speed

	Untargeted		Targeted	
	CIFAR-10	ImageNet	CIFAR-10	ImageNet
Attack	**ResNet20**	**ResNet50**	**ResNet20**	**ResNet50**
Ours	0.39s	20.6s	0.44s	20.1s
StrAttack	1.30s	48.7s	1.28s	49.2s
FWnucl	0.77s	31.6s	0.82s	31.9s

Figure 9: Comparison of computation time per image for StrAttack, FWnucl, and our attack. Tested on 1000 images from the CIFAR10 dataset for ResNet20 and on the NIPS2017 dataset for ResNet50 and InceptionV3.

Future Work

- Improve sparsity using proximal operator of piece-wise exponential function
- Given this vulnerability of NNs, design SOA defense strategies
  - Integrate sparse interpretable adversarial attacks in the adversarial training procedure
- Generate sparse interpretable adversarial attacks in a black-box setting and for real-world scenarios

THANK YOU!

Slides available at:

`www.shpresimsadiku.com`

Check related information on Twitter at:

`@shpresimsadiku`