Interpretable Deep Learning under Fire

Zhang, X., Wang, N., Shen, H., Ji, S., Luo, X., & Wang, T.

Usenix 20’

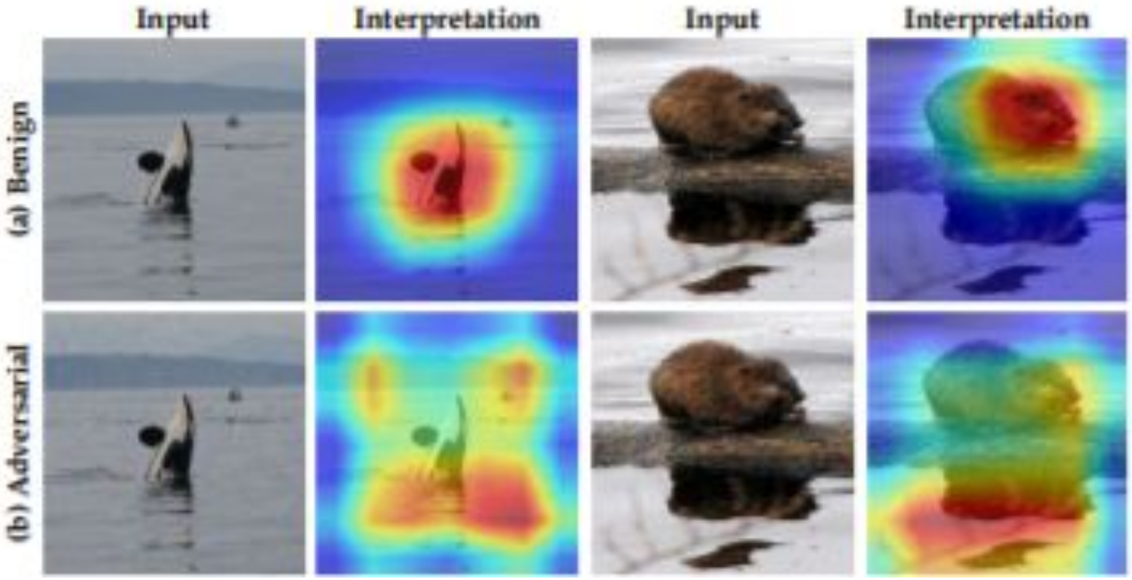
Explaining Deep Learning

Deep learning systems vulnerabilities

Stop Yield Speed Limit

(a) Normal (b) Attack

Interpretability to the rescue!



Interpretable Deep learning

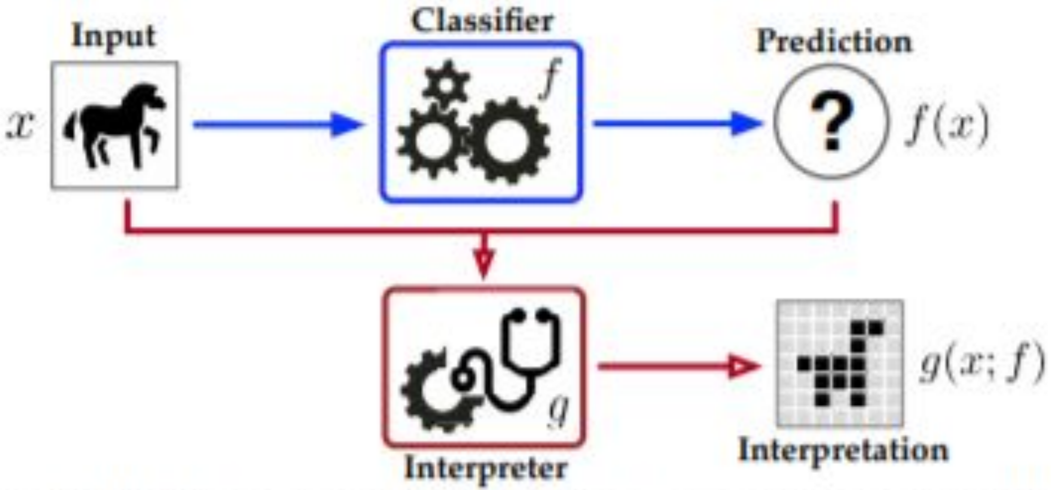
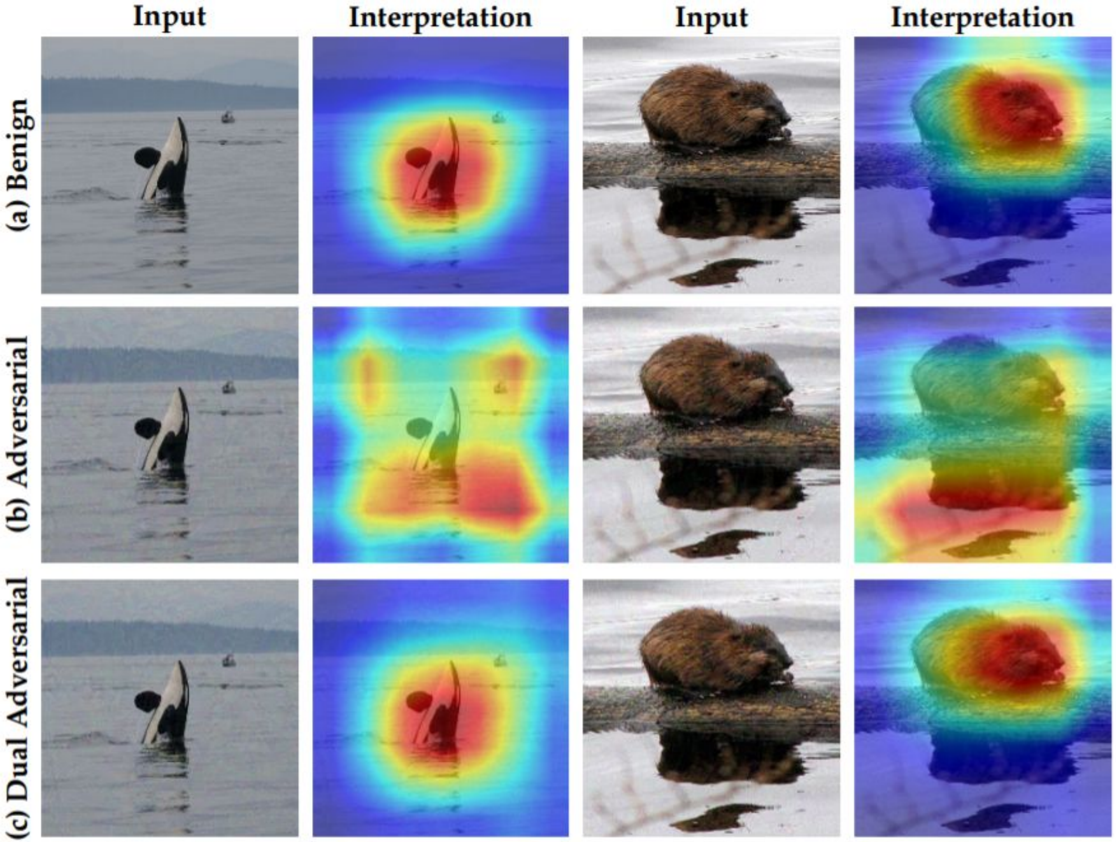


Figure 2: Workflow of an interpretable deep learning system (IDLS).

Can we trust interpreters?



Paper in a Nutshell

Explore vulnerabilities of deep learning interpreters (on computer vision)

➔ Proposes attack ADV^2 that targets both model and its interpreter

➔ Explore different interpreters and models

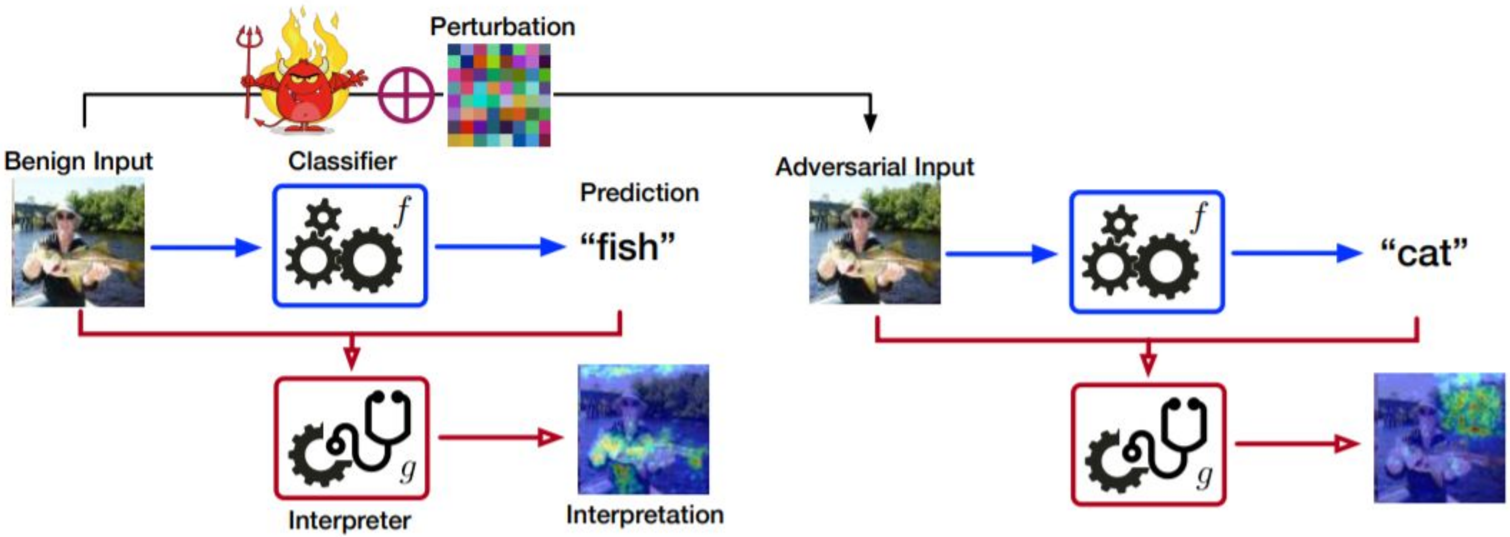
➔ Provide explanation of how to improve interpreters and resent mitigations strategies

Threat model

Paper considers white box settings

➔ The adversary has complete access to the classifier and the interpreter

ADV^2


Perturbation
Benign Input
Classifier
Prediction
"fish"
Adversarial Input
"cat"
Interpreter
Interpretation

ADV^2

Find smallest perturbation that modifies the model prediction and the interpretation to given targets.

ADV^2 generates an adversarial input **x∗** by modifying a benign input **x∘** such that

- (i) **x∗** is misclassified by **f** to a target class **c_t**, **f(x∗)** = **c_t**
- (ii) **x∗** triggers **g** to generate a target attribution map **m_t**, **g(x∗; f)** = **m_t**
- (iii) The difference between **x∗** and **x∘**, Δ**(x∗, x∘)**, is imperceptible

$$
\begin{aligned}
\min_x \quad & \ell_{\text{prd}}(f(x), c_t) + \lambda \ell_{\text{int}}(g(x; f), m_t) \\
\text{s.t.} \quad & \Delta(x, x_\circ) \leq \varepsilon
\end{aligned}
$$

where **f** is the model, **g** the interpreter, the $\ell_{\textbf{prd}}$ is the prediction loss and $\ell_{\textbf{int}}$ is the interpreter loss

Interpreters

1. Target different interpreters
    a. **Back-Propagation-Guided**: Gradient saliency
    b. **Representation-Guided**: class activation mapping
    c. **Model-Guided**: meta-model outputs attribution map
    d. **Perturbation-Guided** : Adds noise or occlusion to input features and observe output changes

ADV^2 vs Back-Propagation-Guided IDPs

1. Gradient Saliency (**GRAD**) Interpreter

Compute gradient with respect to each input feature

2. Attack:

   - Perform gradient updates with gradient smoothing for ReLU

$$x^{(i+1)} = \Pi_{\mathcal{B}_\varepsilon(x_\circ)}\left(x^{(i)} - \alpha \operatorname{sgn}\left(\nabla_x \ell_{\mathrm{adv}}\left(x^{(i)}\right)\right)\right)$$

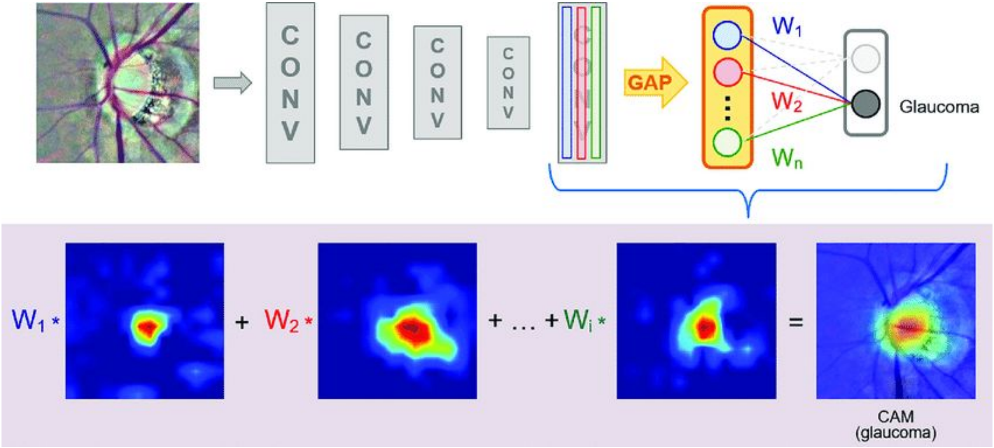
where **f** is the model, **g** the interpreter, the $\ell_{\mathbf{adv}}$ is the adversarial loss

ADV^2 vs Representation-Guided IDPs

1. Class activation mapping (**CAM**) Interpreter

   Perform average pooling on output of last CNN layers and compute weighted average of features for each class

2. Attack: Series of gradient updates



ADV^2 vs Model Guided IDPs

1. Real Time Image Saliency (**RTS**) Interpreter
   - Build surrogate model (meta-model) that outputs feature map.
   - Encoder (Resnet) extracts features
   - U-NET , model trained to output attribution maps

2. Attack: Series of gradient updates + encoder loss $\ell_{\mathbf{enc}}$**(enc(x), enc(ct))**

ADV^2 vs Perturbation-Guided IDPs

1. **MASK** Interpreter

- Adds noise to pixels and checks whether if influences the prediction
- Finds the most important features with minimum noise

Algorithm 1: ADV^2 against MASK.

Input: $x_\circ$: benign input; c_t: target class; m_t: target map; f: target DNN;
g: MASK interpreter

Output: x_*: adversarial input

initialize x and m as $x_\circ$ and $g(x_\circ; f)$;

while *not converged* **do**

```
// update m
```

update m by gradient descent along $\nabla_m \ell_{\text{map}}(m; x)$;

```
// update x with single-step lookahead
```

update x by gradient descent along $\nabla_x \ell_{\text{adv}}\left(x, m - \xi \nabla_m \ell_{\text{map}}(m; x)\right)$;

return x;

Experiments

- ➔ **Q1:** Is it effective against classifiers?
- ➔ **Q2:** Is it effective against interpreters?
- ➔ **Q3:** Is it evasive with respect to attack detection methods?
- ➔ **Q5:** Is it flexible to adopt alternative attack frameworks?
- ➔ **Q6: Why does it work?**

Experiments setup

- ➔ **Dataset**: ImageNet (1.2 million images from 1,000 classes)
- ➔ **Classifiers**: ResNet-50, DenseNet-169
- ➔ **Interpreters:**
  - GRAD
  - CAM
  - RTS
  - MASK

Experiments setup

Optimization:

→ **Based on PGD: Pixel-wise perturbation**

- Iterative optimizer for 1000 iterations
- Run for 400 iterations as ADV only then as ADV^2
- Label smoothing (avoid zero-gradient)

RQ1. Attack Effectiveness (Prediction)

$$\text{Attack Success Rate (ASR)} = \frac{\#\text{successful trials}}{\#\text{total trials}}$$

	ResNet				DenseNet			
	GRAD	CAM	MASK	RTS	GRAD	CAM	MASK	RTS
P	100% (1.0)				100% (1.0)			
A	100% (0.99)	100% (1.0)	98% (0.99)	100% (1.0)	100% (0.98)	100% (1.0)	96% (0.98)	100% (1.0)

Table 3. Effectiveness of PGD (P) and ADV^2 (A) against different classifiers and interpreters in terms of ASR (MC).

RQ2. Attack Effectiveness (Interpretation)

Metrics:

- **Visualisations**
- **Lp measure:** L1 norm between benign and adversarial features maps
- **IoU Test (Intersection over Union)**: $IoU(m) = |O(m) \cap O(m_\circ)| / |O(m) \cup O(m_\circ)|$, where $O(m)$ denotes the set of non-zero dimensions in m

Visualization

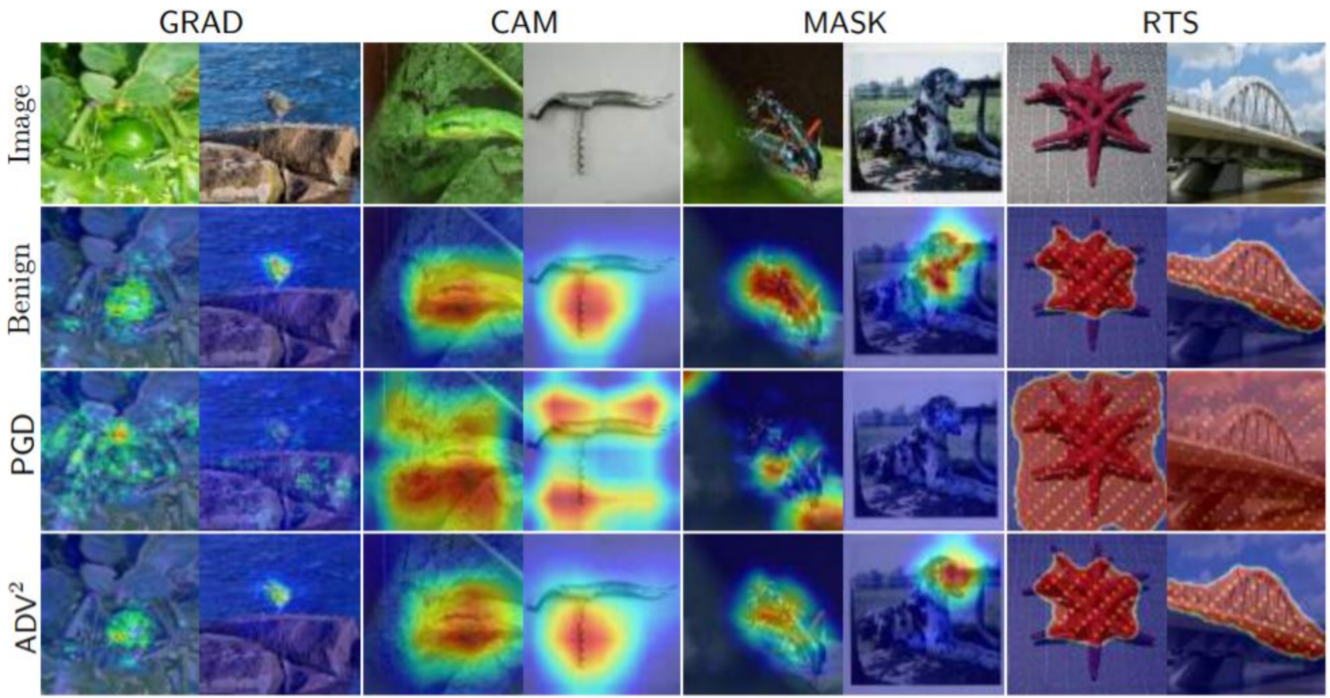


Figure 4: Attribution maps of benign and adversarial (PGD, ADV^2) inputs with respect to GRAD, CAM, MASK, and RTS on ResNet.

L1 Similarity

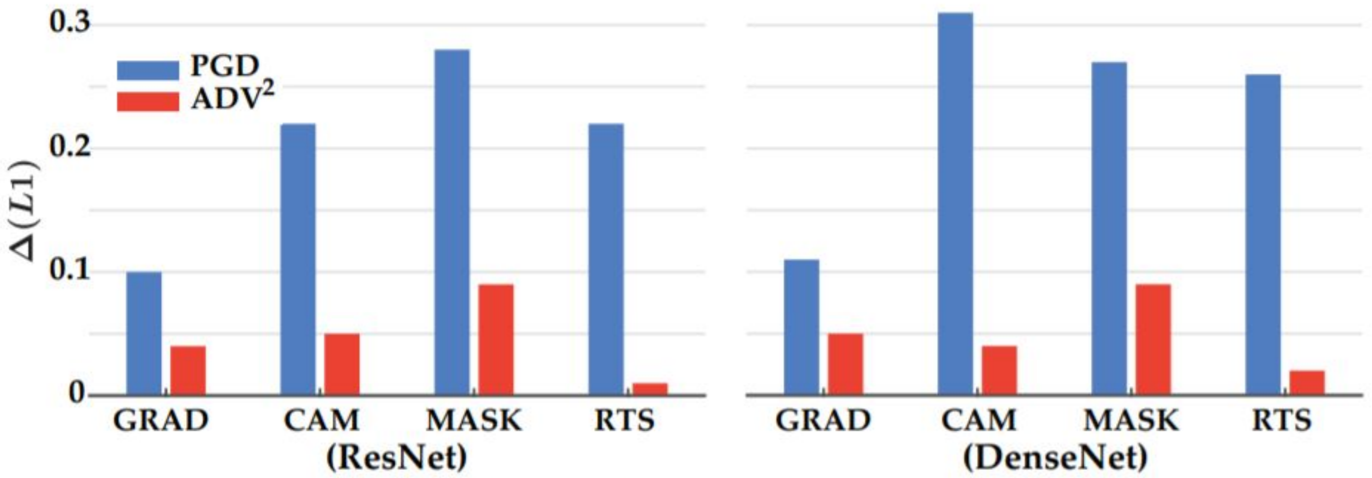


Figure 5: Average $\mathcal{L}_1$ distance between benign and adversarial (PGD, ADV^2) attribution maps.

Intersection over Union (IoU)

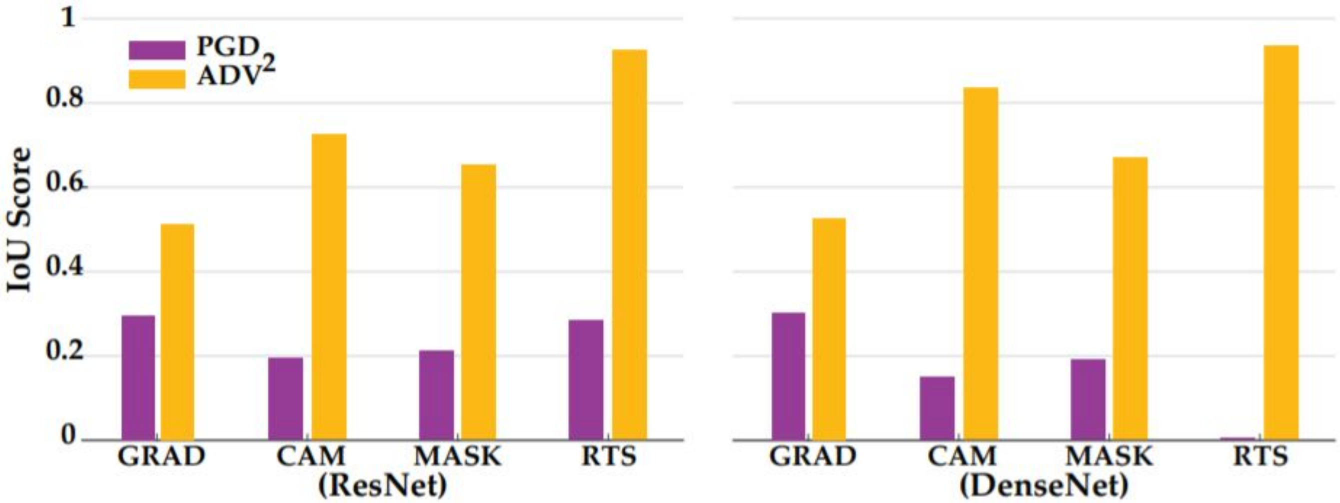


Figure 6: IoU scores of adversarial attribution maps (PGD, ADV^2) with respect to benign maps.

Use Case: Skin Cancer detection

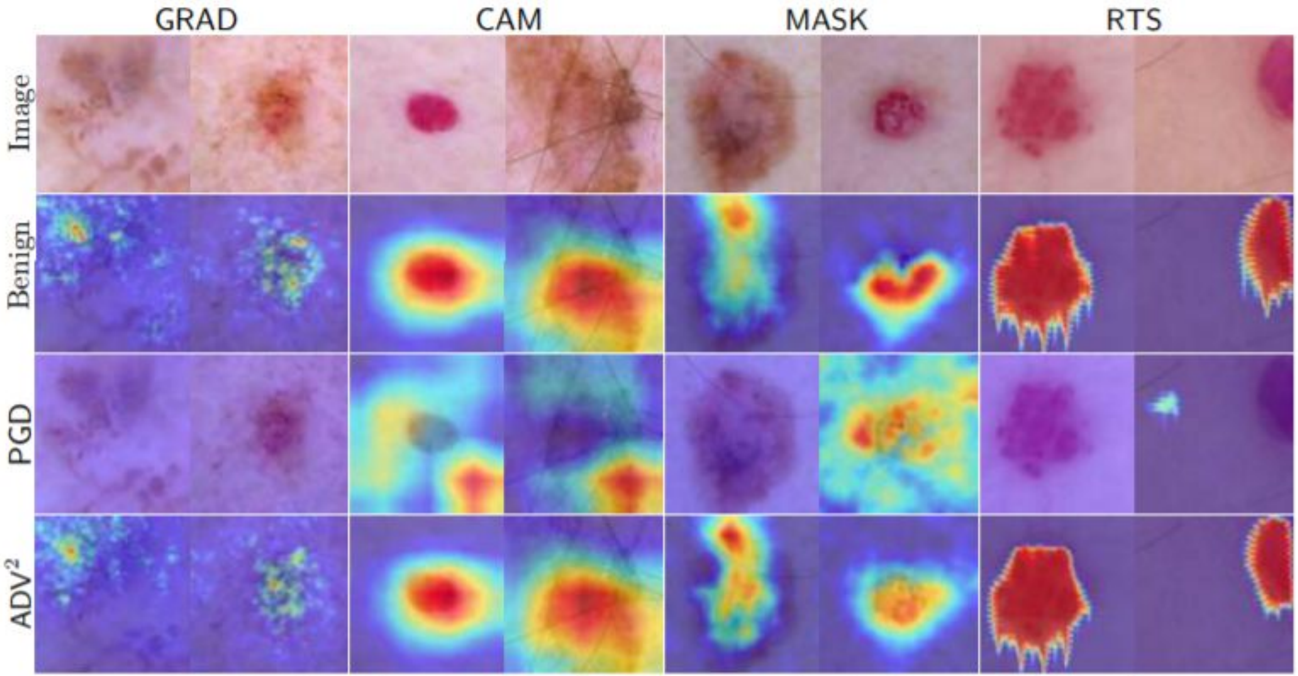


Figure 7: Attribution maps of benign and adversarial (ADV^2) inputs in the skin cancer screening application.

Skin cancer classification

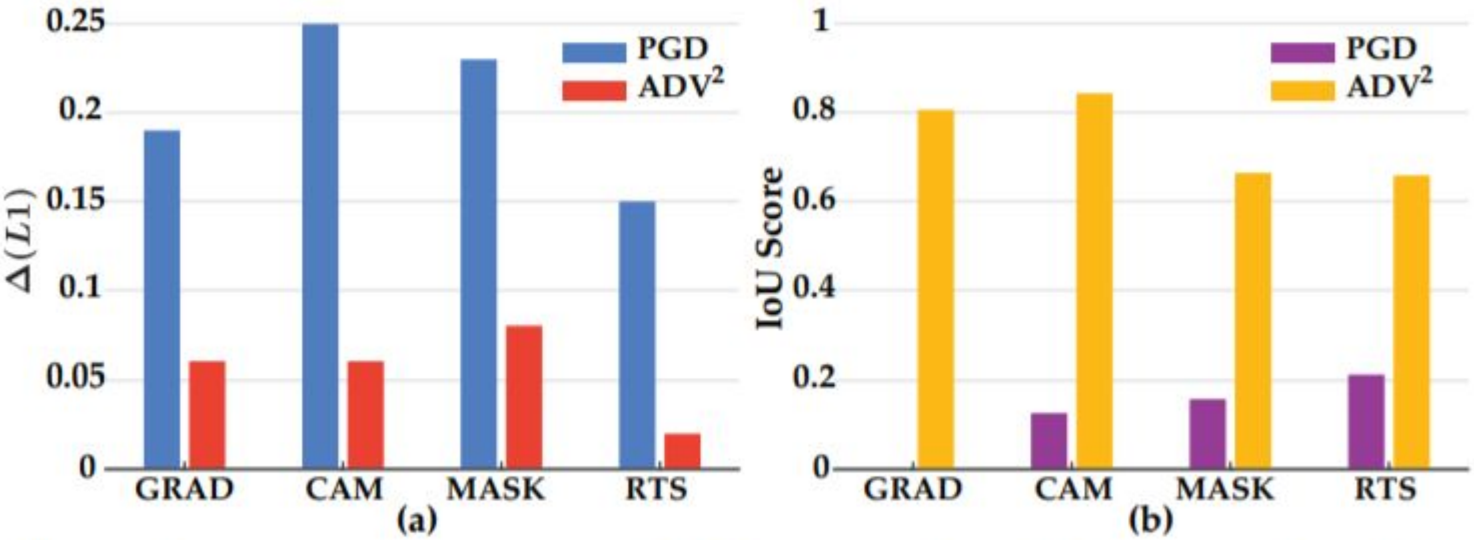


Figure 8: $\mathcal{L}_1$ measures (a) and IoU scores (b) of adversarial attribution maps (PGD, ADV^2) with respect to benign maps.

RQ3. Attack Evasiveness

Detection: Feature squeezing

”Squeeze” multiple inputs into single feature space and compares to non-squeezed predictions

Squeezers types:

- **Bit depth reductions**
- **Local smoothing**
- **Non local-smoothing**

Detection

Squeezer	Setting	PGD	MASK-A	RTS-A
Bit Depth Reduction	2-bit	92.3%	84.1%	94.0%
	3-bit	72.7%	89.2%	88.3%
L. Smoothing	3×3	97.3%	98.6%	99.0%
N. Smoothing	11-3-4	52.3%	74.7%	75.3%

Table 4. Detectability of adversarial inputs by PGD, basic ADV^2 (A)

Detection

Squeezer	Setting	PGD	MASK-A	RTS-A	MASK-A*	RTS-A*
Bit Depth Reduction	2-bit	92.3%	84.1%	94.0%	11.7%	29.4%
	3-bit	72.7%	89.2%	88.3%	35.9%	13.9%
L. Smoothing	3×3	97.3%	98.6%	99.0%	16.5%	3.4%
N. Smoothing	11-3-4	52.3%	74.7%	75.3%	51.7%	29.4%

Table 4. Detectability of adversarial inputs by PGD, basic ADV^2 (A), and adaptive ADV^2 (A*) using feature squeezing.

Adaptive attack: Add loss term $\ell_{\mathbf{sqz}}$**(f(x), f(ψ(x))** to minimize cross entropy between predictions of original input **x** and squeeze inputs **ψ(x)**

Does it transfer to other frameworks?

Attack Transfer

Implement ADV based on spatial-transformations (**STADV-based**)

- Replace pixels by another pixel
- Instead of adding noise (PGD)

Impact of samples size per user

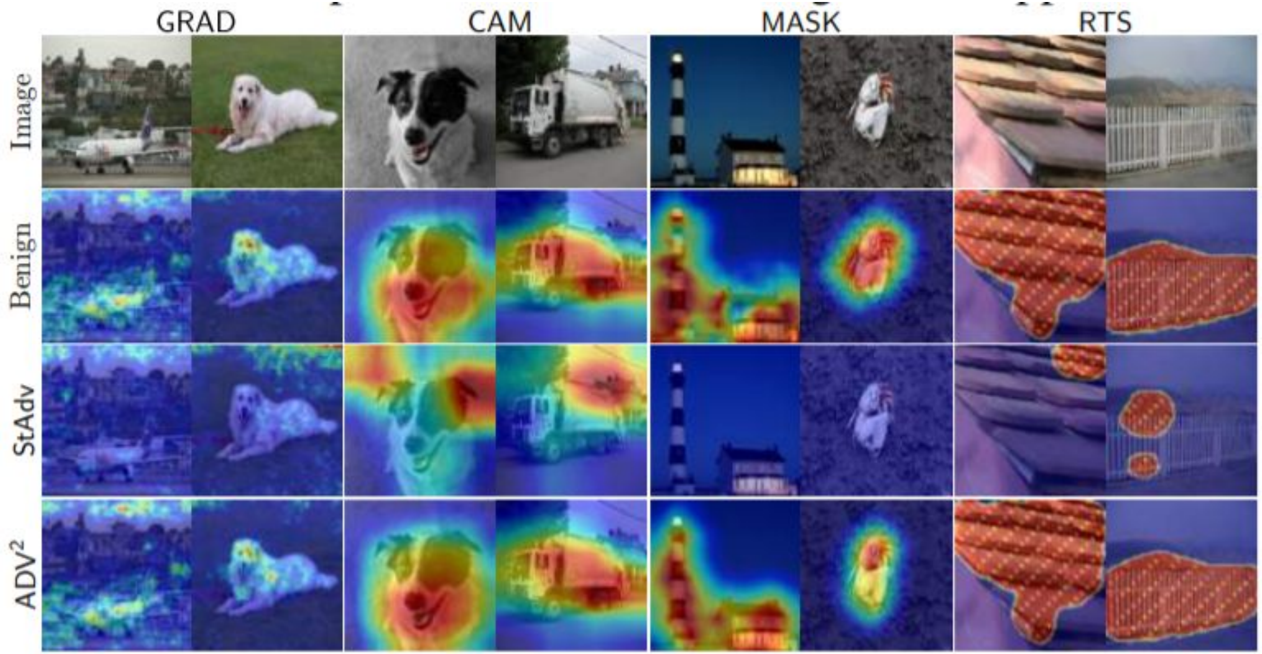


Figure 9: Attribution maps of benign and adversarial (STADV, STADV-based ADV^2) inputs with respect to GRAD, CAM, MASK, and RTS on ResNet.

Impact of samples size per user

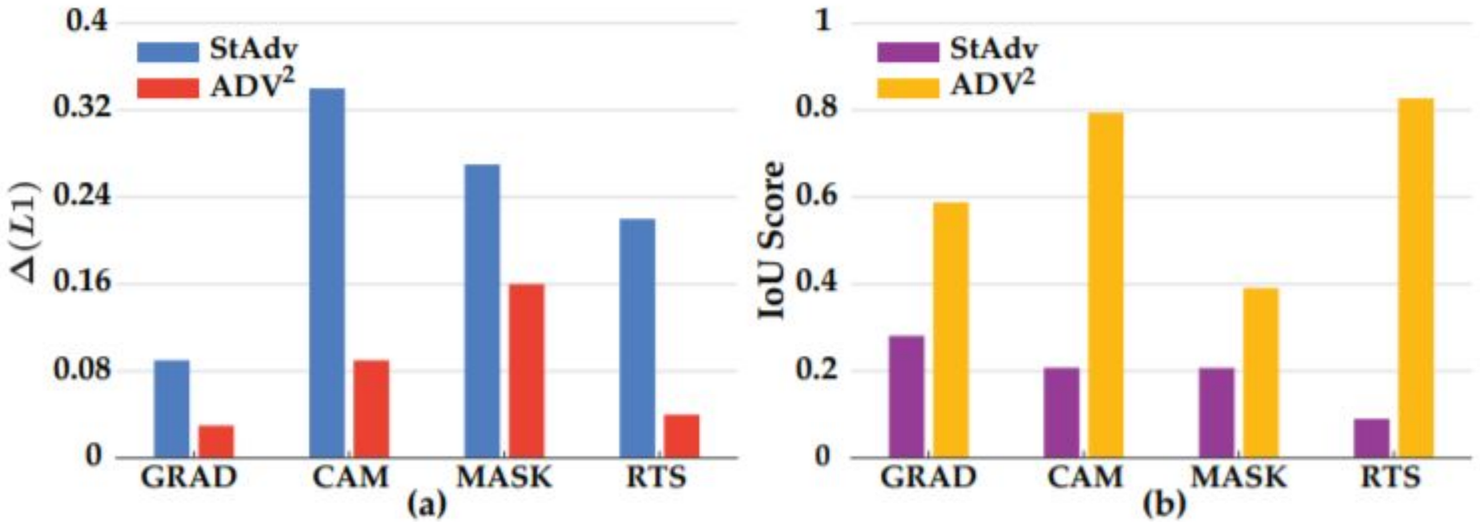


Figure 10: $\mathcal{L}_1$ measures (a) and IoU scores (b) of adversarial attribution maps (STADV, STADV-based ADV^2) with respect to benign maps on ResNet.

How is it possible?

Root of Attack Vulnerability

Intuition: Gap between prediction and interpretation

1. Try to generate random shapes as attribution map
2. Try to generate another class attribute map
3. Measure transferability of attack amon interpreters

Root of Attack Vulnerability

	Grad	Cam	Mask	Rts
Adv^2	100% (0.98)	100% (1.0)	99% (0.95)	100% (1.0)

Table 6. ASR (MC) of Adv^2 targeting random patch interpretations.

	Grad	Cam	Mask	Rts
Adv^2	100% (0.99)	100% (0.99)	100% (0.99)	100% (1.0)

Table 8. ASR (MC) of Adv^2 with random class interpretations.

Root of Attack Vulnerability

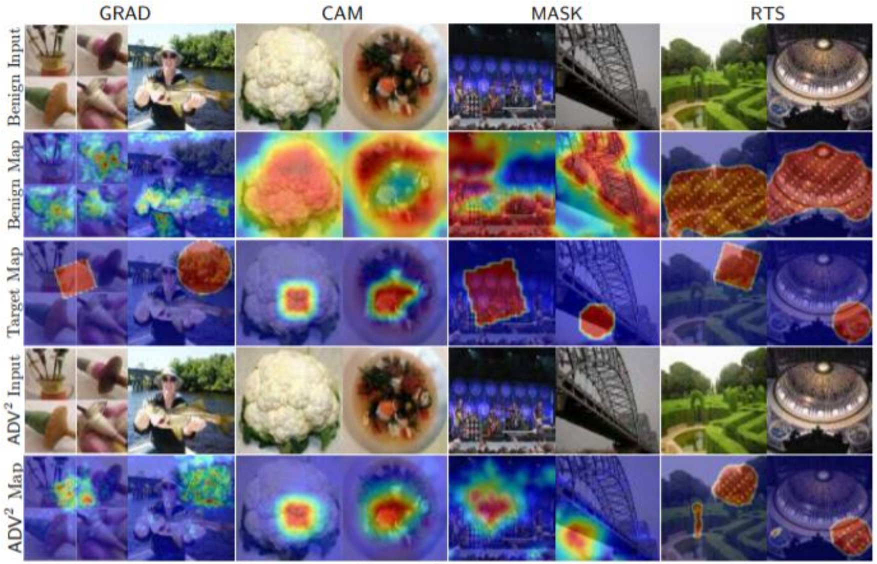


Figure 11: Visualization of ADV^2 targeting random patch interpretations across different interpreters on ResNet.

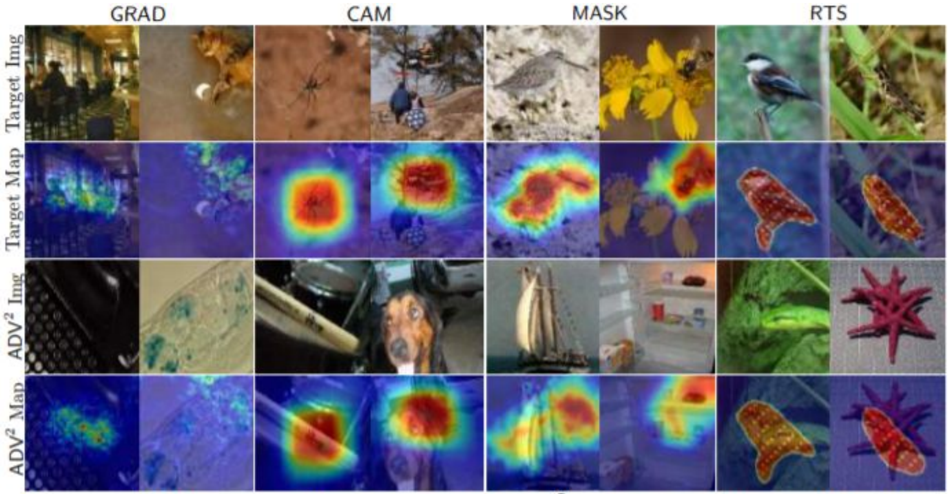


Figure 12: Target and adversarial (ADV^2) inputs and their attribution maps on ResNet.

Root of Attack Vulnerability

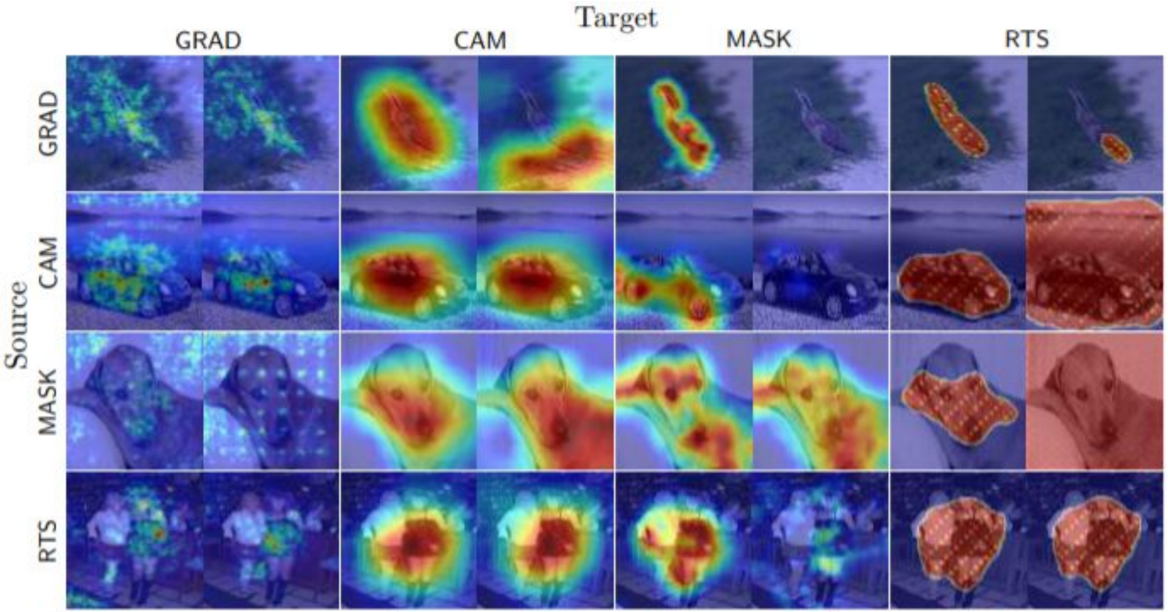


Figure 14: Visualization of attribution maps of adversarial inputs across different interpreters on ResNet.

	GRAD	CAM	MASK	RTS
GRAD	0.04	0.24	0.22	0.24
CAM	0.09	0.05	0.18	0.13
MASK	0.12	0.34	0.09	0.74
RTS	0.10	0.17	0.20	0.01
PGD	0.10	0.22	0.28	0.22

Table 9. L1 distance between attribution maps of adversarial (ADV2 , PGD) on ResNet (row/column as source/target).

Root of Attack Vulnerability

Possible explanation:

➔ Each interpreter targets a different part of the model
  - Don’t fully explain the model

➔ Attack only need to ensure one aspect is preserved for the specific interpreter
  - Poor transferability

Mitigations

- ➔ Defense 1: **Ensemble of interpreters**
- ➔ Defense 2: **Adversarial Interpretation training**

Defense 1 : Ensemble of Interpreters

Use multiple interpreters to analyse predictions

Challenges:

- Differences in interpretations
- Adversary might adapt to ensemble

Defense 2 : Adversarial Interpreter Training

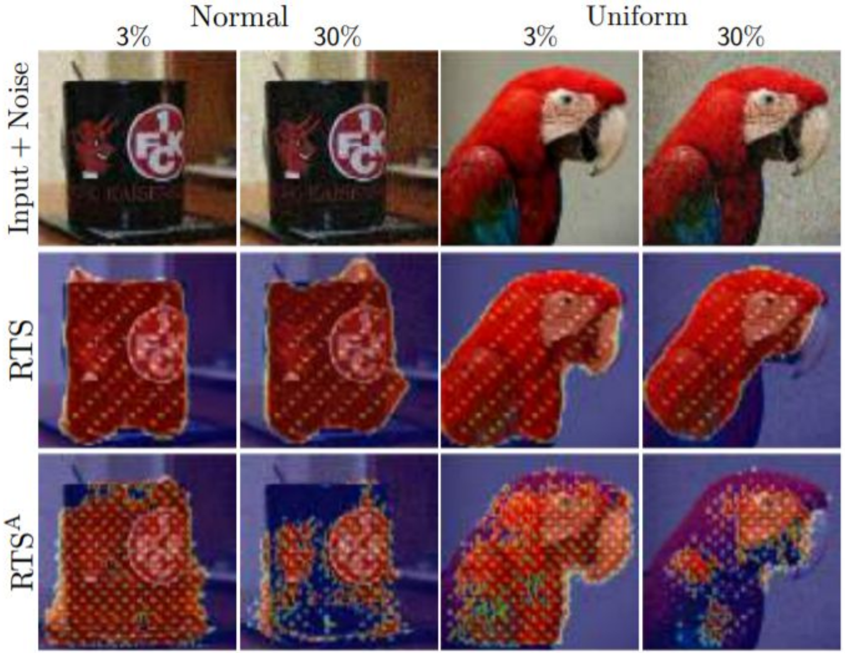
Minimise prediction-interpretation gap

- Introduce adversarial loss to maximize L1 distance between benign and adversarial samples for trainable interpreters

Experiments on RTS (model-based IDLS) vs adversary trained RTS

1. Compare sensitivity to perturbation
2. Compare L1 measures

Defense 2 : Adversarial Interpreter Training



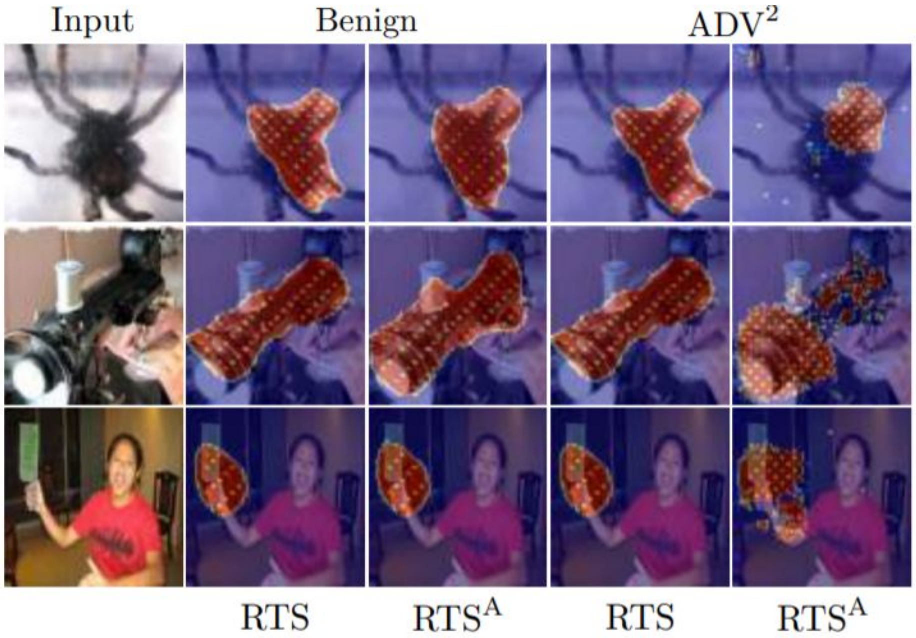
$\mathcal{L}_1$ measures

Noise	$\mathbb{N}$	$\mathbb{U}$
0%	0.01	0.01
	0.01	0.01
3%	0.01	0.01
	0.02	0.03
30%	0.09	0.10
	0.17	0.17

RTS RTS$^{\text{A}}$

Figure 15: Attribution maps generated by RTS and RTS$^{\text{A}}$ under different noise levels and types (normal $\mathbb{N}$, unifrom $\mathbb{U}$) on ResNet.

Defense 2 : Adversarial Interpreter Training



	RTS	RTS^A
Benign	\	0.03
ADV^2	0.01	0.10

$\mathcal{L}_1$ measures

Figure 16: Attribution maps of benign and adversarial (ADV^2) inputs with respect to RTS and RTS^A on ResNet.

Discussion & Limitations

1. The work present adversarial attack on DNN models and various interpreters
   a. Specific to CV
   b. What about LIME and SHAP?

2. Assume that the adversary has white-box knowledge
   a. In future work, investigate black box settings

3. Present attack effectiveness, stealth and adaptations, investigate into the cause of the vulnerability and propose mitigations

Conclusion

1. Work present a systematic study on attacking CNN models and their Interpreters
2. ADV^2 is effective on different models, optimizers and different interpreter types
3. Identify the prediction-interpreter gap as the possible cause
4. Possible countermeasures are interpreters ensemble and adversarial training
5. Show that interpreters can offer false sense of security

Relevant Papers

1. **Interpreters spoofing**
   a. "**Evaluating explanation methods for deep learning in security**." Warnecke, Alexander, et al. *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2020.
2. **Interpreter Uncertainty modelling**
   a. "**How Much Should I Trust You? Modeling Uncertainty of Black Box Explanations**.",Slack, Dylan, et al. *arXiv preprint arXiv:2008.05030* (2020).

Thank you!