Attacks on deep learning and explainable AI

Konstantin Arkhipenko [arkhipenko<at>ispras<dot>ru, t.me/arxikv]

November 10, 2021

Ivannikov Institute for System Programming of the RAS (ISP RAS)

- **Deep learning** is SOTA in many tasks (computer vision, NLP, ...)
- However, **poor robustness and interpretability** of DNNs limits their applications in safety-critical environments

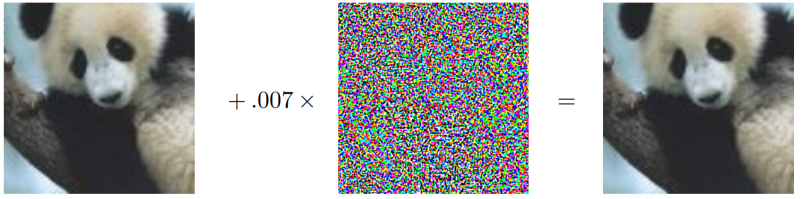
Contents

Techniques

- Training: data poisoning (to allow future intrusion)
- Testing (inference): evasion (**adversarial examples**), model extraction and inversion

Knowledge

- White-box: complete information about the model, including model architecture, parameters, loss function and data
- Black-box: samples and oracle only
- Gray-box: partial information (many different settings examined in the papers; typically the parameters are unknown)

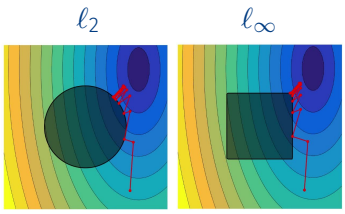


- Note: $J(\boldsymbol{\theta}, \boldsymbol{x}, y)$ is the loss function value where $\boldsymbol{\theta}$ are the learned parameters, and y is the ground truth label corresponding to input image $\boldsymbol{x}$

- PGD is used in many SOTA white-box attacks
- Find adversarial example x' for an input x within the ℓ_p-ball B of radius ε: $\|x - x'\|_p \leq \varepsilon$
- Repeatedly set ($x_0 := x$):

$$x_{i+1} = \mathrm{Proj}_B(x_i + \alpha g) \qquad g = \arg\max_{\|v\|_p \leq 1} v^T [\nabla_x L(x, y)\big|_{x=x_i}]$$

- **Targeted attack**: replace $L(x, y)$ with $-L(x, y')$ where y' is the target label
- Loss L is typically cross-entropy, however, other objectives can be used in much stronger **adaptive** attacks



- **Idea**: train on adversarial examples

$$\mathbb{E}_{x,y}[\max_{\delta \in \mathcal{S}} L(\theta; x + \delta, y)] \to \min_{\theta}$$

- Most widely used defense method (typically increases accuracy on adversarial examples from $\sim$0% to 30-70%)
- Very diverse defense methods have been proposed to further increase the accuracy
- However, **none** of these defenses have been shown to be robust to **adaptive** attacks targeting these defenses (several such claims were unvalidated by Tramer et al. 2020)

- **Numerical gradient estimation**
  - Too many oracle queries ($O(d)$ where d is input dimensionality)
  - Not possible if class probabilities are not available (methods for this case use even more oracle queries)
- **Substitute model** (Papernot et al. 2016, untargeted attack):

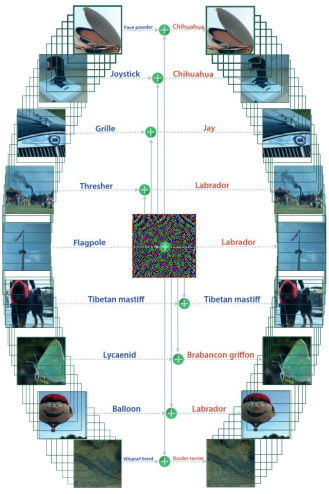
Algorithm 1 - Substitute DNN Training: for oracle $\tilde{O}$, a maximum number max_ρ of substitute training epochs, a substitute architecture F, and an initial training set S_0.

Input: $\tilde{O}$, max_ρ, S_0, λ
1: Define architecture F
2: **for** $\rho \in 0$ .. $max_\rho - 1$ **do**
3: // *Label the substitute training set*
4: $D \leftarrow \left\{(\vec{x}, \tilde{O}(\vec{x})) : \vec{x} \in S_\rho\right\}$
5: // *Train F on D to evaluate parameters θ_F*
6: $\theta_F \leftarrow \text{train}(F, D)$
7: // *Perform Jacobian-based dataset augmentation*
8: $S_{\rho+1} \leftarrow \{\vec{x} + \lambda \cdot \text{sgn}(J_F[\tilde{O}(\vec{x})]) : \vec{x} \in S_\rho\} \cup S_\rho$
9: **end for**
10: **return** θ_F

Transferability of adversarial attacks

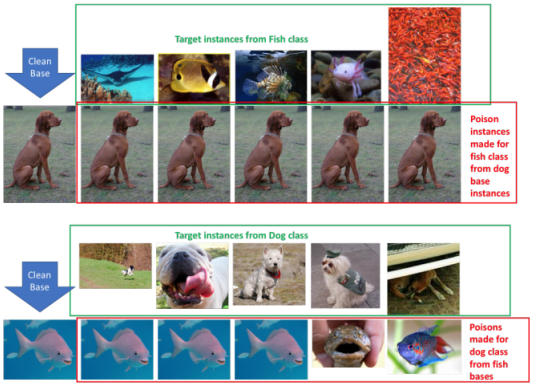
ISP RAS

- Liu et al. 2016 showed that targeted attacks are more diffucult to transfer, but this can be done using ensembles of models
  - The resulting adversarial examples are more likely to transfer to other models
- Meanwhile, Moosavi-Dezfooli et al. 2016 showed existence of **universal perturbations** which can transfer across different datasets and models
- As shown by Ilyas et al. 2019 (“Adversarial examples are not bugs, they are features”), these phenomena may be related to hidden patterns in datasets



- Add examples to the training set to manipulate the behavior of the model at test time
- Different settings possible depending on whether the adversary can control the labeling process:
  - The adversary has **full control** over the training process, the victim is provided with the final model parameters (**Trojaned model**)
  - The adversary can alter the data samples but cannot control the labeling process
- In the second setting, imperceptible perturbations are reasonable, while in the former can alter data samples in any way

$$p = \arg\min_x \|f(x) - f(t)\|_2^2 + \beta\|x - b\|_2^2$$

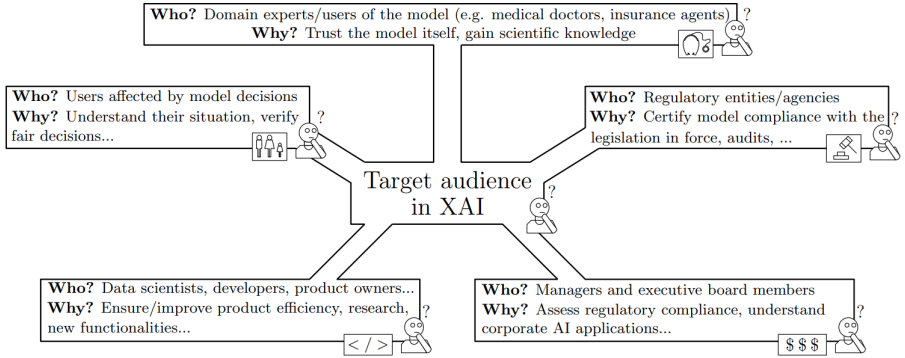


Algorithm 1 Poisoning Example Generation

Input: target instance t, base instance b, learning rate λ
Initialize x: $x_0 \leftarrow b$
Define: $L_p(x) = \|f(\mathbf{x}) - f(\mathbf{t})\|^2$
for $i = 1$ **to** $maxIters$ **do**
  Forward step: $\widehat{x_i} = x_{i-1} - \lambda\nabla_x L_p(x_{i-1})$
  Backward step: $x_i = (\widehat{x_i} + \lambda\beta b)/(1 + \beta\lambda)$
end for

Contents

- Arrieta et al. 2020 “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”



Human-centered

- **Information transfer rate** and **trust coefficient** (Schmidt & Biessmann 2019)
  - **Idea**: the better an explanation is, the faster and more accurately an user will reproduce the decisions of the model
  - Compute ITR **before** and **after** showing explanations to the user to assess trust
  - **Target audience**: end users, executives, domain experts

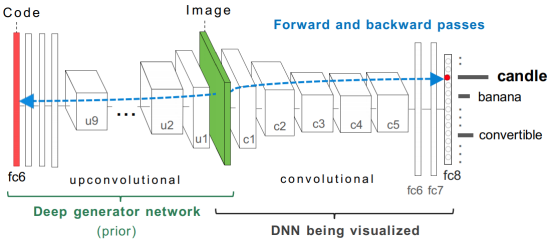
Metric-centered

- **Fidelity**: removing the relevant features (according to an explanation) should significantly affect prediction score/accuracy

Post-hoc explainability

- Interpretable models (e.g. linear regression, decision trees) are often not sufficient for complex tasks
- **Post-hoc explainability**: try to enhance interpretability of complex or black-box models (such as DNNs) by various means:
  - **Feature relevance explanation**
  - **Explanations by example**: activation maximization, prototypes, counterfactuals
  - **Text explanations**: generate texts that help explaining the results from the model
  - **Visual explanation**: e.g. dimensionality reduction
  - **Explanation by simplificiation**: techniques in which a whole new system is rebuilt based on the trained model to be explained
  - **Local explanations**: segment the solution space and give explanations to less complex solution subspaces

- **Synthesizing the preferred inputs for neurons via deep generator networks**, Nguyen et al. 2016
- Use image generator network (deconv) to activate the output neurons



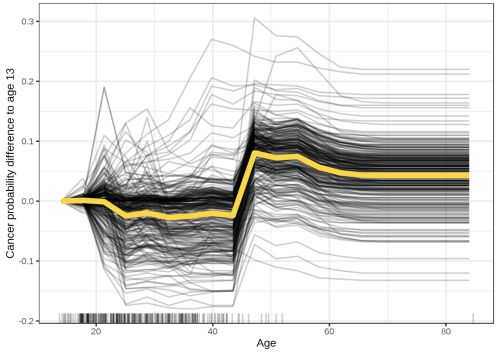
White/black-box

- White-box: need gradient access or even specific propagation rules for all layer (operation) types
- Black-box: only input data and predictions are used

Data type

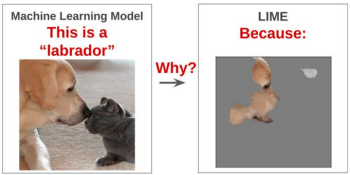
- Tabular data: meaningful real-valued/ordinal/binary/categorical features
- Images: saliency maps
- NLP: token/n-gram highlighting, visualizing attention weights

Partial dependence plot / Individual conditional expectation

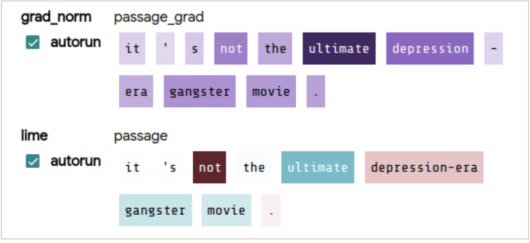


- Fix all features except one
- ICE: average of PDPs for all data samples

- **Global surrogate** (knowledge distillation): train a new interpretable model which replicates the one being explained
- **Local surrogate** (LIME, Ribeiro et al. 2016):
  - Create a dataset where each sample is a perturbed version of the original sample being explained
  - Ask the (black-box) oracle for predictions and use them as ground-truth for a new interpretable model
  - Need some way to perturb samples and maximum allowed perturbation level



- Perturbation method: remove one of the input tokens (alternatively, replace with some mask token)
- Substitute model: linear regression (regression) or logistic regression (classification)
- Use substitute model weights to obtain input token importances
- **Language Interpretability Tool** (PAIR-code, Tenney et al. 2020):



Pixel/patch sensitivity maps

- Occlusion sensitivity
  - Apply gray patches on the input image iteratively and see the model confidence
- Vanilla gradient, gradient $\times$ input
- Integrated gradients
- SmoothGrad

CAM (class activation map) based methods

- Grad-CAM, Grad-CAM++, ScoreCAM, ...

Implementations: tf-explain, PAIR-code/saliency, tf-keras-vis, Captum (pytorch)
Out of scope: LRP, DeepLIFT, DeepSHAP

- Given an input image $\boldsymbol{x}$ and baseline image $\boldsymbol{x}'$:

$$[\mathrm{IG}(\boldsymbol{x})]_i \triangleq (x_i - x_i') \cdot \int_{\alpha=0}^{1} \frac{\partial f_c(\boldsymbol{x}' + \alpha(\boldsymbol{x} - \boldsymbol{x}'))}{\partial x_i} d\alpha$$

- The baseline is typically zero (black) image
- Riemann sum approximation (m is the number of steps):

$$[\mathrm{IG}_{approx}(\boldsymbol{x})]_i = (x_i - x_i') \cdot \sum_{k=1}^{m} \frac{\partial f_c(\boldsymbol{x}' + \frac{k}{m}(\boldsymbol{x} - \boldsymbol{x}'))}{\partial x_i} \cdot \frac{1}{m}$$

Why not vanilla gradient?

- Model saturation (perturbing a single pixel) may have no effect on prediction
- IG have a number of desirable properties not present in Vanilla gradient

Sensitivity(a), violated by Gradient

- For every input and baseline that differ in one feature but have different predictions then the differing feature should be given a non-zero attribution

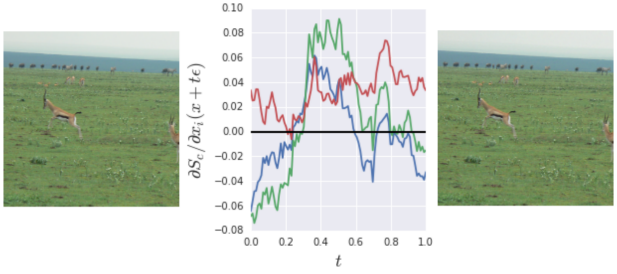
Sensitivity(b)

- If the function implemented by the deep network does not depend (mathematically) on some variable, then the attribution to that variable is always zero

Completeness (implies Sensitivity(a), violated by Gradient)

- The attributions add up to the difference between the output at the input x and the baseline x'

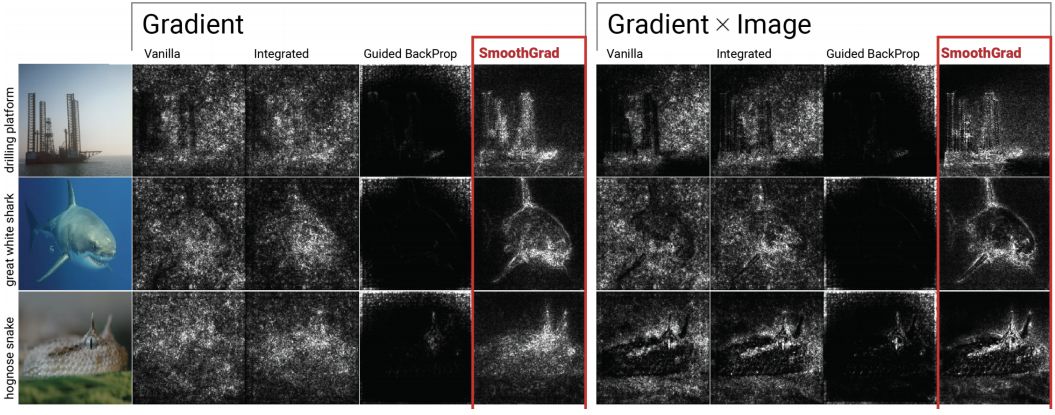
- Motivation: the gradient may fluctuate sharply at small scales:



Method

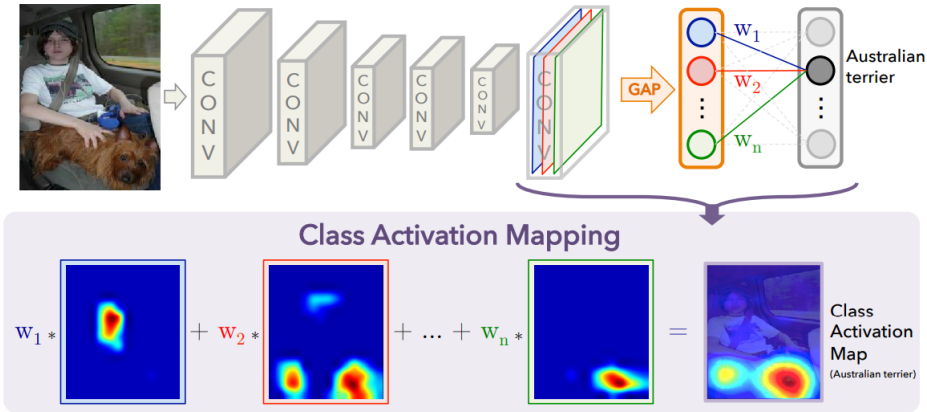
- Apply Vanilla gradient to N noisy images obtained from the original image by adding Gaussian noise $\mathcal{N}(0, \sigma^2)$
- Can be combined with Grad $\times$ input, Guided BP or CAM-based methods (e.g. Smooth Grad-CAM++ by Omeiza et al. 2019)

SmoothGrad vs Guided BP vs Integrated gradients



- Simple gradient saliency such as (the norm of) Vanilla gradient, Gradient × input can be used for NLP tasks

Image saliency: class activation maps (Zhou et al. 2015)



- Initially proposed as a localization method for ILSVRC 2014 dataset

Assumption

- The last convolutional layer uses **global average pooling** and is immediately followed by softmax layer
  - Such models can be suboptimal on some datasets and are restricted to image classification

Definition

- Global average pooling ($\boldsymbol{A}$ is the last conv layer activations for a single input image in channels-last format):

$$[\mathrm{GAP}(\boldsymbol{A})]_k = \frac{1}{h_a w_a} \sum_{i,j} A_{ijk}$$

- The score (logit) f_c for class c:

$$f_c(\boldsymbol{x}) = \sum w_k^c \cdot [\mathrm{GAP}(\boldsymbol{A})]_k$$

Definition

- Class activation map $\mathrm{CAM}(\boldsymbol{x}, c)$ for input image $\boldsymbol{x}$ and class c:

$$[\mathrm{CAM}(\boldsymbol{x}, c)]_{ij} \triangleq \sum_k w_k^c A_{ijk}$$

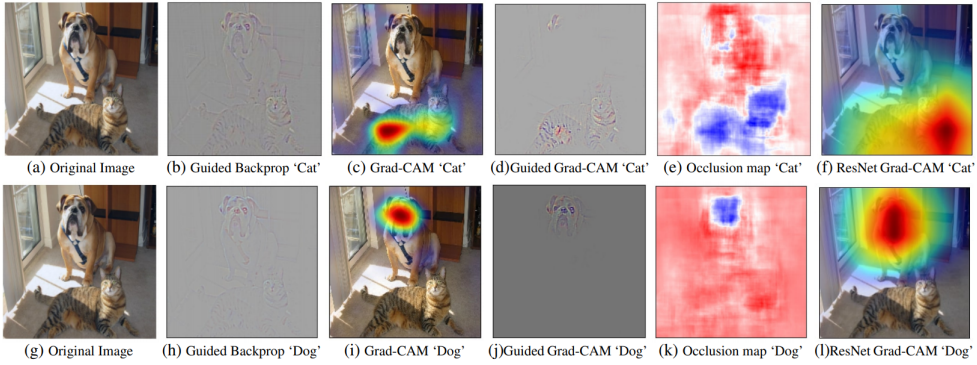
- Use bilinear interpolation to match the input image size

Completeness property

- The activation map sums up to the score f_c thanks to the usage of global average pooling:

$$\frac{1}{h_a w_a} \sum_{i,j} [\mathrm{CAM}(\boldsymbol{x}, c)]_{ij} = f_c(\boldsymbol{x})$$

Image saliency: Grad-CAM (Selvaraju et al. 2016)



(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d)Guided Grad-CAM 'Cat' (e) Occlusion map 'Cat' (f) ResNet Grad-CAM 'Cat'

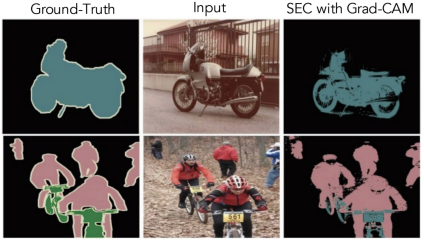
(g) Original Image (h) Guided Backprop 'Dog' (i) Grad-CAM 'Dog' (j)Guided Grad-CAM 'Dog' (k) Occlusion map 'Dog' (l)ResNet Grad-CAM 'Dog'

- Does not rely on CAM assumption
  - Also equivalent to CAM if the assumption holds
- Can be applied to any layer of the network

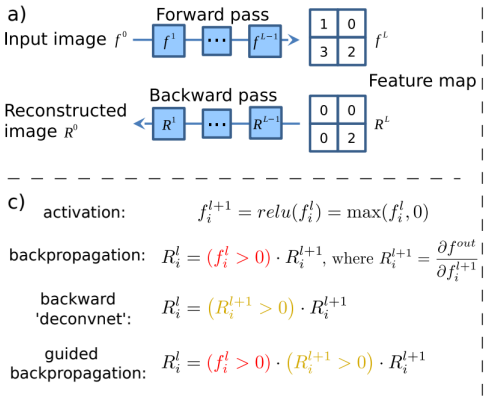
- Use **gradients** of the score $f_c(x)$ with respect to activations of some convolutional layer (typically the last one but this is not required):

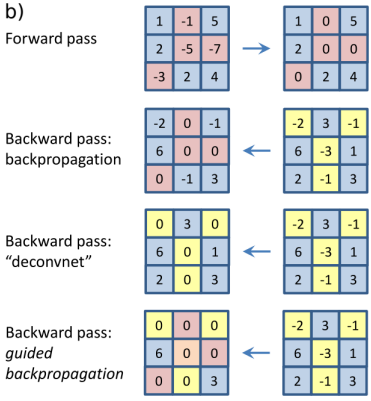
$$w_k^c = \frac{1}{h_a w_a} \sum_{i,j} \frac{\partial f_c(x)}{\partial A_{ijk}}$$

- The paper reports SOTA results on ILSVRC-15 weakly-supervised localization
- An example of weakly-supervised segmentation is provided as well:

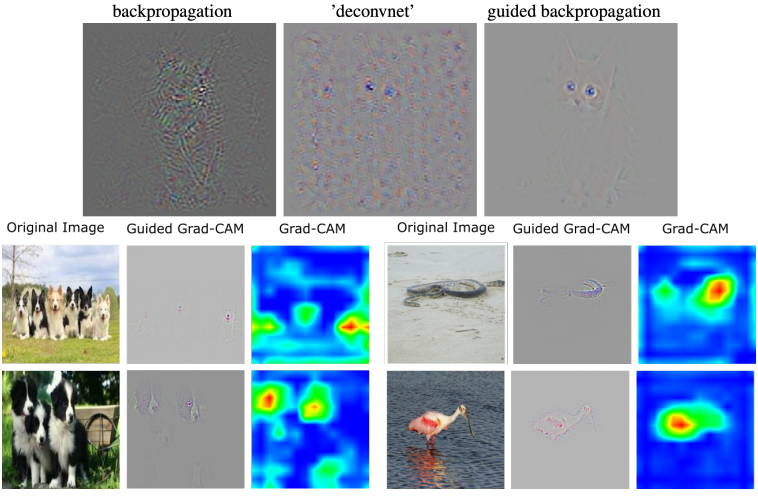


- Guided backpropagation (Springenberg et al. 2014):

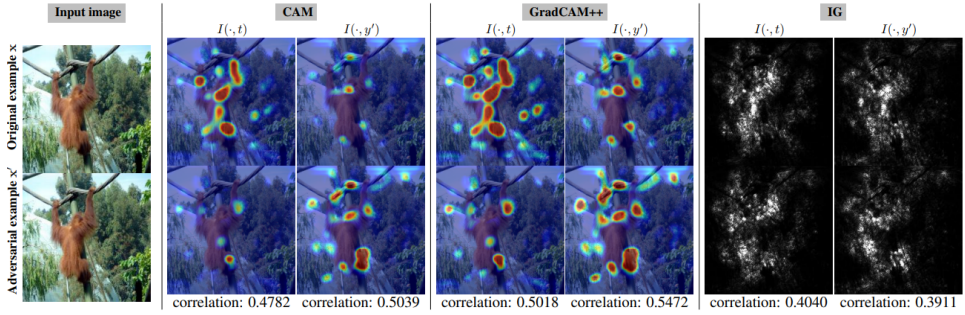




- Combine Grad-CAM with guided BP:



Boopathy et al. 2020



- Interpretation discrepancy between benign image x and adversarial image x':

$$\mathcal{D}(x, x') = \frac{1}{C} \sum_{c \in \mathcal{C}} \|\mathrm{CAM}(x, c) - \mathrm{CAM}(x', c)\|_p$$

- **Proposition** (by the paper): is not difficult to prevent adversarial examples from having large interpretation discrepancy with respect to **a single class label**, but not both original y and target y'
- ℓ_1 2-class interpretation discrepancy:

$$\mathcal{D}_{2,\ell_1}(x, x') = \frac{1}{2}(\|\mathrm{CAM}(x, y) - \mathrm{CAM}(x', y)\|_1 + \|\mathrm{CAM}(x, y') - \mathrm{CAM}(x', y')\|_1)$$

- Adversarial training:

$$\mathbb{E}_{x,y}[f_{train}(\theta; x, y) + \gamma \cdot \max_{\|\delta\|_\infty \leq \epsilon} \mathcal{D}_{2,\ell_1}(x, x + \delta)] \to \min_{\theta}$$

Notes:

- 200-step PGD accuracy under different perturbation sizes ϵ
- **Adv** is vanilla adversarial training
- **Int** and **Int2** are interpretability-aware training with 1-class and 2-class ID, respectively

Method	$\epsilon = 0$	0.05	0.1	0.2	0.3	0.35	0.4
MNIST, `Small`							
Normal	1.000	0.530	0.045	0.000	0.000	0.000	0.000
Adv	0.980	0.960	0.940	0.925	0.890	0.010	0.000
TRADES	0.970	0.970	0.955	0.930	0.885	0.000	0.000
IG-Norm	0.985	0.950	0.895	0.410	0.005	0.000	0.000
IG-Norm-Sum	0.975	0.955	0.935	0.910	0.880	0.115	0.000
Int-one-class	0.975	0.635	0.330	0.140	0.125	0.115	0.080
Int	0.950	0.930	0.905	0.840	**0.790**	**0.180**	**0.140**
Int-Adv	0.935	0.945	0.905	0.880	**0.855**	**0.355**	**0.175**
Int2	0.950	0.945	0.935	0.890	**0.845**	**0.555**	**0.385**
Int2-Adv	0.955	0.925	0.915	0.880	**0.840**	**0.655**	**0.620**
	$\epsilon = 0$	2/255	4/255	6/255	8/255	9/255	10/255
CIFAR-10, `WResnet`							
Normal	0.765	0.250	0.070	0.060	0.060	0.060	0.060
Adv	0.720	0.605	0.485	0.330	0.170	0.145	0.085
TRADES	0.765	0.610	0.460	0.295	0.170	0.140	0.100
Int-one-class	0.685	0.505	0.360	0.190	0.065	0.040	0.025
Int	0.735	0.630	0.485	0.365	**0.270**	**0.240**	**0.210**
Int-Adv	0.665	0.585	0.510	0.385	**0.320**	**0.300**	**0.280**
Int2	0.690	0.595	0.465	0.360	**0.290**	**0.245**	**0.220**
Int2-Adv	0.680	0.585	0.485	0.405	**0.335**	**0.310**	**0.285**

- **Goal**: detect adversarial examples using **m**ulti-**l**ayer **l**eave-**o**ne-**o**ut attributions
- Leave-one-out (LOO) attributions:

$$\phi(x)_i \triangleq f(x)_c - f(x_{(i)})_c \qquad c = \arg\max_{j \in C} f(x)_j$$

  - $f(x)$ is the probability vector for input x, $x_{(i)}$ is obtained by masking i-th pixel in x, C is the set of class labels
- Measure **interquartile range** (IQR) for the attribution map:

$$\mathrm{IQR}(\phi(x)) \triangleq Q_{\phi(x)}(0.75) - Q_{\phi(x)}(0.25)$$

- Multi-layer: compute LOO attributions for **hidden layer activations** as well
- Feed multiple IQR values into logistic regression detector

- XAI and adversarial attacks is a rather new field receiving attention in the last 5 years
- There are apparent connections between XAI and adversarial robustness as claimed by a number of papers

Thank you