We can **detect adversarial examples** in Neural Nets by leveraging **topological** information from **under-optimized edges.**

Detecting by Dissecting: Using Persistent Homology to catch Adversarial Examples in Deep Nets

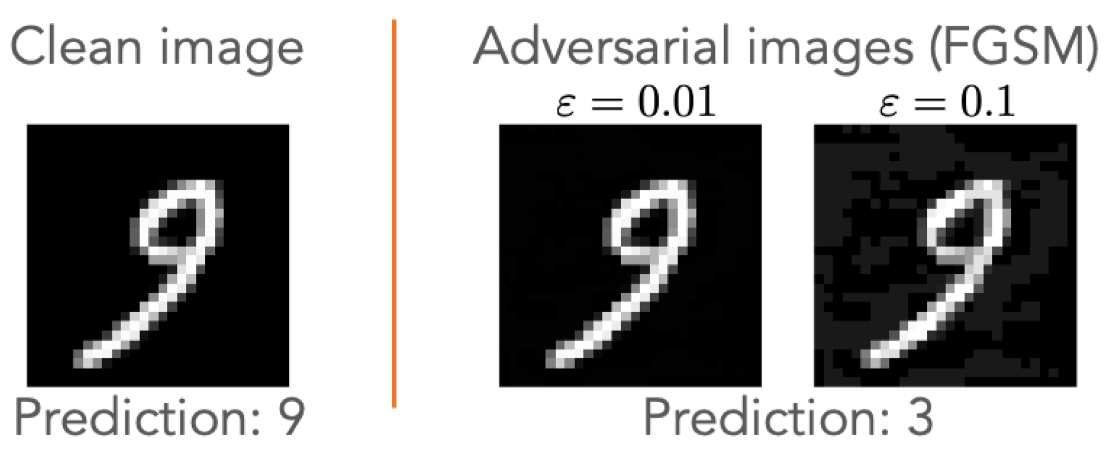
Morgane Goibert, Thomas Ricatte, Elvis Dohmatob

Criteo AI Lab

criteo AI Lab

Introduction

- Adversarial examples: $x^{adv} = x + \delta, ||\delta|| \leq \varepsilon$, whose objective is to fool Neural Nets, i.e $h(x^{adv}) \neq y$.
- Different attack algorithms (FGSM, DeepFool, CW, etc.) or different strenght (more or less subtle attacks).



- What to do against attacks: defend or detect. Defend tries to give the correct label to an adversarial input. Detect tries to flag adversarial inputs (and afterwards, human in the loop).
- No complete understanding of the phenomenon

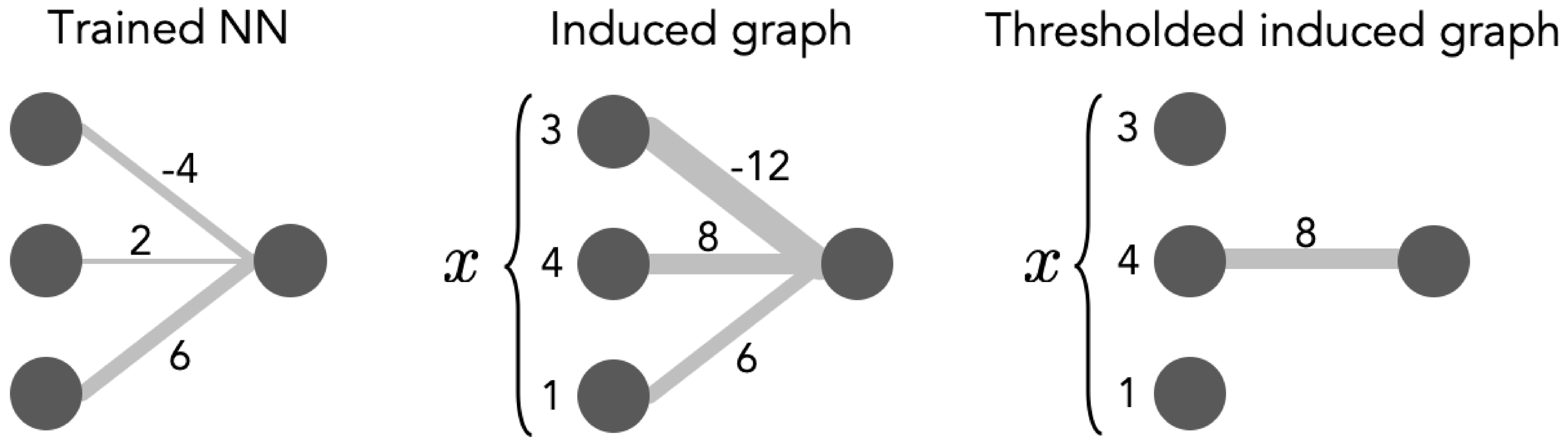
Contributions

Main Takeaways

- Two *detection methods*: Raw Graph and *Persistence Diagram*, based on topological information, better than baselines.
- *Under-optimized edges* are a major flaw for Neural Nets' robustness. Removing them by pruning helps better robustness.
- Unified protocol for evaluating adversarial examples detectors.

Methods

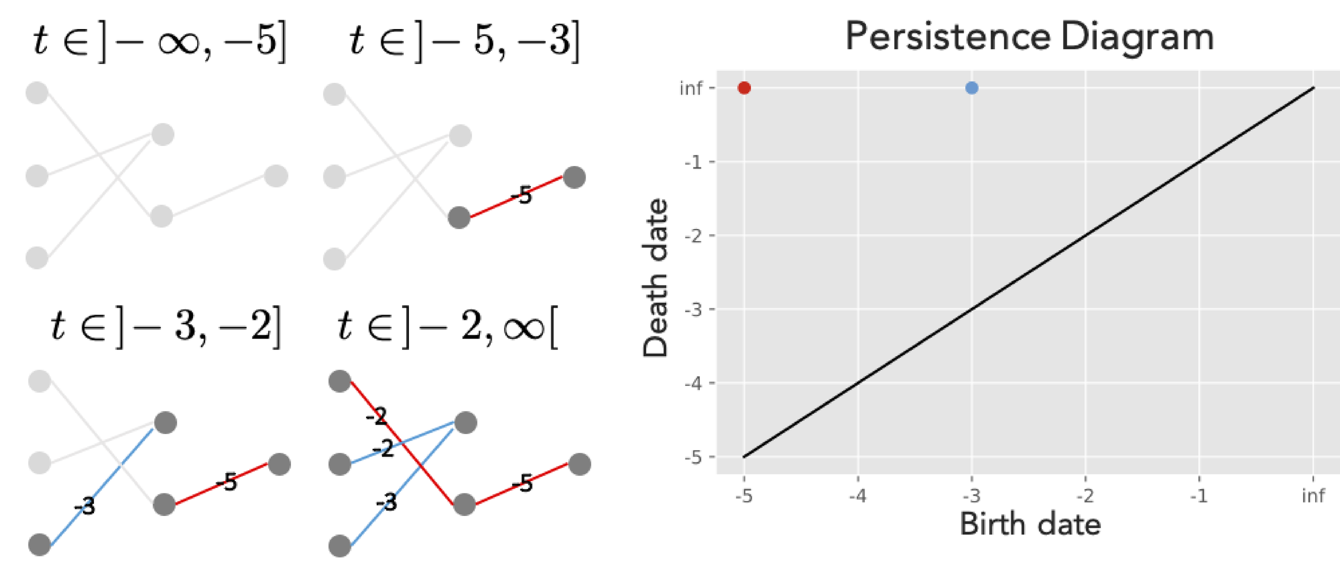
Thresholded induced graph. *Information* from both a *trained Neural Net* and an *input*.



- *Trained Neural Net* g has parameters W_l for layer $l \in \{1,..,L\}$.
- For input x, *activation value* $g(x)_l$ is the activation value of layer l.
- *Induced graph* for NN g and input x: $G(g,x) = G(V,E)$, $V = \{1,\dots,n_1 + \dots + n_L\}$, $E = \{u^l, v^{l+1}, w^l_{u,v} \subseteq V^2 \times \mathbb{R}\}$ where $w^l_{u,v} = [g(x)_l]_u \times (W_l)_{u,v}$.
- *Thresholded induced graph* $G^q(g,x)$: we keep an edge (u,v) iff $|(W_l^{init})_{u,v} - (W_l)_{u,v}| < q_l$, with q_l threshold for layer l ("Magnitude Increase" method). *Reducing parameter space dimension*: $q_1 = \dots = q_L = q$ or 0.

Raw Graph. Simply use the *weights* of $G^q(g,x)$ as *features*, so the feature mapping is $\Phi_{RG}(x,g) = \mathrm{Vec}(W)$.
Use classical RBF kernel $K_{RG}(x,x') = \exp\left(-\frac{1}{2\sigma^2}||\Phi_{RG}(x,g) - \Phi_{RG}(x',g)||^2\right)$.

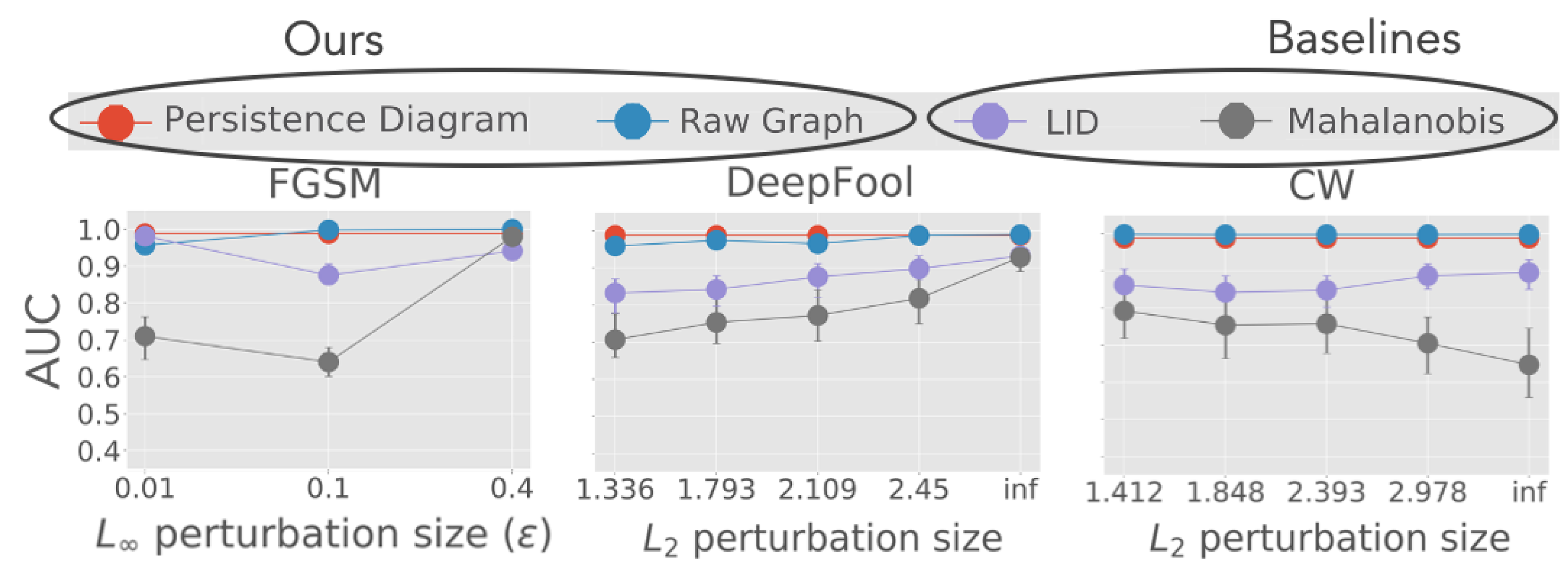
Persistence Diagram. The representation of topological information, in a weighted graph, through time.



Use the zeroth-dimensional *persistence diagram* of $\tilde{G}^q(x,g) = (V, -|W|)$ where $G^q(x,g) = (V,W)$ as *features*, so the feature mapping is $\Phi_{PD}(x,g) := \mathrm{PD}(\tilde{G}^q(x,g))$. We use the *Sliced-Wasserstein Kernel*: $K_{PD}(x,x') = \exp\left(-\frac{1}{2\sigma^2}\mathrm{SW}(\Phi_{PD}(x,g), \Phi_{PD}(x',g))\right)$.

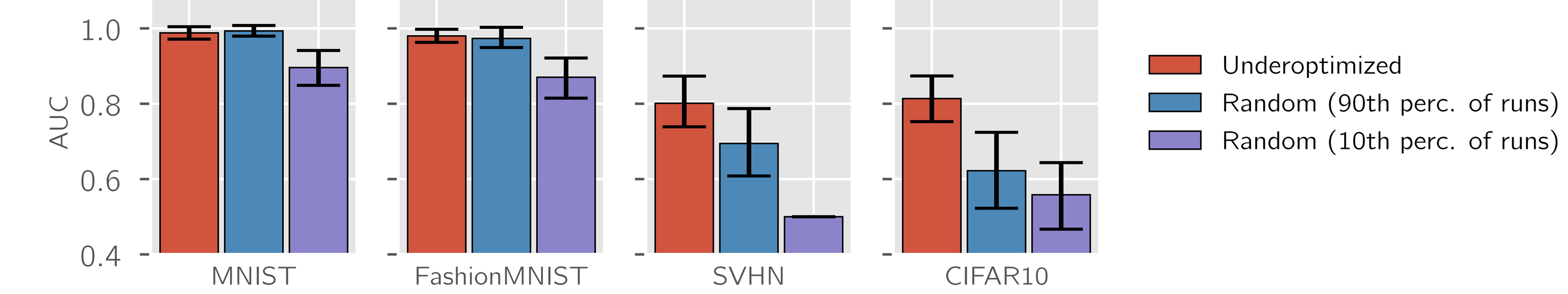
Detection Results

- *Unsupervised experiments*: better for generalizing to any attacks.
- Better or competitive with baselines.
- Illustration: AUC results on MNIST LeNet (unsupervised).



Under-optimized edges

When we threshold *using under-optimized edges* (red), we get *better results* than when we select the same number of *random edges* (blue, 90th percentile and purple, 10th percentile).



Removing under-optimized edges ⇔ Pruning (relevant ratio) ⇒ *robustness*.