Tấn công Elastic-Net vào DNN qua một số mẫu đối nghịch

Khai phá dữ liệu và học máy

Nguyễn Mạnh Linh, Đào Thị Thu Hồng

Khoa Toán-Cơ-Tin học
Đại học Khoa học Tự nhiên

2022/12

Outline

Giới thiệu

Sự dễ tổn thương của DNNs

Hình 1: Các mẫu đối nghịch bị phân loại sai bởi mô hình Inception-V3

Giới thiệu - Các loại tấn công

1. Tấn công nhắm đích - *targeted attacks*
2. Tấn công không nhắm đích - *untargeted attacks*

Giới thiệu: Huấn luyện đối nghịch

Huấn luyện đối nghịch (*adverarial training*) (Madry et al. 2017): Sử dụng mẫu đối nghịch để huấn luyện một mô hình mạnh có khả năng chống chịu với các nhiễu của mẫu đối nghịch.

Giới thiệu: Tấn công phân loại ảnh

1. Tấn công phân loại ảnh dựa trên mạng neuron tích chập
2. Mẫu đối nghịch được tạo ra để làm sai lệch kết quả phân loại
3. Mẫu mới được tạo ra (gần) giống với mẫu gốc

Giới thiệu: Độ nhiễu

$\|x\|_q = (\sum_{i=1}^{p} |x_i|^q)^{\frac{1}{q}}$ kí hiệu chuẩn L_q của vector p chiều
$x = [x_1, ..., x_p]$ với $q \geq 1$

- L_∞: Đánh giá sự thay đổi tối đa các pixel (oodfellow, Shlens, and Szegedy 2015)
- L_2: Cải thiện chất lượng hình ảnh (Carlini and Wagner 2017b)
- L_1: Sử dụng trong các bài toán phục hồi ảnh (Fu et al. 2006)

Trong bài toán mẫu đối nghịch, độ biến dạng L_1 đánh giá tổng các thay đổi trong nhiễu loạn và đóng vai trò là một thành phần (hàm) lồi đo lường số lượng pixel thay đổi (độ thưa) gây ra bởi nhiễu

Giới thiệu - Các tập dữ liệu thử nghiệm

Tấn công dựa trên L_1 với các tập dữ liệu

- MNIST
- CIFAR-10
- ImageNet

Được so sánh với các tấn công dựa trên L_2 và L_∞

Outline

Tấn công DNNs - FGM, I-FGM

- Kí hiệu $\mathbf{x_0}$ và $\mathbf{x}$ lần lượt là mẫu gốc và mẫu đối nghịch, t là lớp mục tiêu cần tấn công.
- Tấn công dựa trên L_∞

$$\mathbf{x} = \mathbf{x_0} - \epsilon \times \text{sign}(\nabla J(\mathbf{x_0}, t)) \tag{1}$$

với ϵ là độ biến dạng L_∞ giữa $\mathbf{x}$ và $\mathbf{x_0}$ và $\text{sign}(\nabla J)$ là dấu của gradient

- Tấn công dựa trên L_1 và L_2

$$\mathbf{x} = \mathbf{x_0} - \epsilon \frac{\nabla J(\mathbf{x_0}, t)}{\|\nabla J(\mathbf{x_0}, t)\|_q} \tag{2}$$

với $q = 1, 2$ và ϵ là độ méo tương quan

Tấn công DNNs - C&W

- Thay vì sử dụng hàm mất mát trên tập huấn luyện Carlini và Wagner đã thiết kế một hiệu chỉnh L_2 trong hàm mất mát dựa trên lớp logit trong DNNs để sinh ra các mẫu đối nghịch (Carlini and Wagner 2017b)
- Công thức này hóa ra là một trường hợp riêng của thuật toát EAD (sẽ được trình bày trong phần sau)

Phòng thủ DNNs

- Defensive distillation - Chưng cất phòng thủ (Papernot et al.2016b)
- Adversarial training - Huấn luyện đối nghịch (Zheng et al. 2016; Madry et al. 2017; Tram‘er et al. 2017; Zantedeschi, Nicolae, and Rawat 2017)
- Detection methods - Phương pháp dò tìm (Feinman et al. 2017; Grosse et al. 2017; Lu, Issaranon, and Forsyth 2017; Xu, Evans, and Qi 2017)

Outline

EAD - Tóm lược

- Hiệu chỉnh elastic-net là công nghệ được sử dụng rộng rãi trong việc giải quyết các bài toán lựa chọn thuộc tính nhiều chiều (Zou and Hastie 2005)
- Nhìn chung, hiệu chỉnh elastic-net được sử dụng trong bài toán cực tiểu hóa sau đây:

$$\text{minimize}_{\mathbf{z}\in\mathcal{Z}}\ f(\mathbf{z}) + \lambda_1\|\mathbf{z}\|_1 + \lambda_2\|\mathbf{z}\|_2^2 \tag{3}$$

  Trong đó $\mathbf{z}$ là vector của p biến tối ưu, $\mathcal{Z}$ là tập nghiệm chấp nhận được, $f(\mathbf{z})$ là hàm mất mát, $\|\mathbf{z}\|_q$ là chuẩn q của $\mathbf{z}$ và $\lambda_1, \lambda_2 \geq 0$ tương ứng là các tham số hiệu chỉnh L_1 và L_2
- Biểu thức $\lambda_1\|\mathbf{z}\|_1 + \lambda_2\|\mathbf{z}\|_2^2$ được gọi là hiệu chỉnh elactic-net của $\mathbf{z}$

EAD - Xây dựng

Cho trước 1 ảnh $\mathbf{x_0}$ và nhãn đúng của nó là t_0, gọi $\mathbf{x}$ là mẫu đối nghịch của $\mathbf{x_0}$ với lớp đích nhắm đến là $t \neq t_0$. Hàm mất mát $f(\mathbf{x})$ cho tấn công nhắm đích là:

$$f(\mathbf{x}, t) = \max\left(\max_{j \neq t}[\mathbf{Logit}(\mathbf{x})]_j - [\mathbf{Logit}(\mathbf{x})]_t, -\kappa\right) \tag{4}$$

Trong đó $\mathbf{Logit}(\mathbf{x}) = [\mathbf{Logit}(\mathbf{x})_1, ..., \mathbf{Logit}(\mathbf{x})_K] \in \mathbb{R}^K$ là lớp logit (lớp trước softmax) biểu diễn cho $\mathbf{x}$ trong mạng DNN, K là số lượng lớp cần phân loại, $\kappa > 0$ là tham số tin cậy, nó đảm bảo một khoảng cách cố định giữa $\max_{j \neq t}[\mathbf{Logit}(\mathbf{x})]_j$ và $[\mathbf{Logit}(\mathbf{x})]_t$.

EAD - Xây dựng

Thành phần $[\mathbf{Logit}(x)]_t$ là xác xuất dự đoán x có nhãn t theo luật phân loại của hàm softmax:

$$\text{Prob}(\text{Label}(\mathbf{x}) = t) = \frac{\exp([\mathbf{Logit}(\mathbf{x})]_t)}{\sum_{j=1}^{K} \exp([\mathbf{Logit}(\mathbf{x})]_j)} \tag{5}$$

EAD - Xây dựng

Do đó, hàm mất mát trong phương trình 4 có mục đích là để cho ra nhãn t là lớp có xác xuất cao nhất của x và tham số κ đảm bảo sự phân biệt giữa lớp t và lớp dự đoán gần nhất khác với t. Với tấn công không nhắm mục tiêu, hàm mất mát trong phương trình 4 trở thành:

$$f(x) = \max\left([\mathbf{Logit}(\mathbf{x})]_{t_0} - \max_{j \neq t}[\mathbf{Logit}(\mathbf{x})]_j, -\kappa\right) \tag{6}$$

EAD - Xây dựng

Hiệu chỉnh elastic-net còn tạo ra mẫu đối nghịch tương tự với ảnh gốc. Công thức tấn công elastic-net vào mạng DNNs (EAD) để tạo ra mẫu đối nghịch $(\mathbf{x}, t)$ cho ảnh gốc $(\mathbf{x_0}, t_0)$ như sau:

$$\begin{aligned} &\text{minimize}_{\mathbf{x}}\ c \times f(\mathbf{x}, t) + \beta \|\mathbf{x} - x_0\|_1 + \|\mathbf{x} - \mathbf{x_0}\|_2^2 \\ &\text{st } \mathbf{x} \in [0,1]^p \end{aligned} \tag{7}$$

Với $f(x, t)$ được xác định trong phương trình (4), $c, \beta \geq 0$ lần lượt là các tham số hiệu chỉnh của hàm mất mát f và hàm phạt L_1.

Thuật toán 1 Tấn công Elastic-net vào DNNs (EAD)

Input: Ảnh gốc và nhãn của nó $(\mathbf{x}_0, t_0)$, lớp mục tiêu t, tham số chuyển giao κ, tham số hiệu chỉnh β, độ dài bước α_k, số bước lặp I

Output: mẫu đối nghịch $\mathbf{x}$

Khởi tạo: $\mathbf{x}^{(0)} = \mathbf{y}^{(0)} = \mathbf{x}_0$

for $k = 0$ to $I - 1$ do

$$\mathbf{x}^{(k+1)} = S_\beta(\mathbf{y}^{(k)} - \alpha_k \nabla g(\mathbf{y}^{(k)}))$$
$$\mathbf{y}^{(k+1)} = \mathbf{x}^{(k+1)} + \frac{k}{k+3}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)})$$

end for

Luật quyết định: tìm $\mathbf{x}$ từ tập các mẫu thành công trong $\{\mathbf{x}^k\}_{k=1}^I$ (luật EN, luật L_1).

EAD - Thuật toán

Trong đó

$$[S_\beta(\mathbf{z})]_i = \begin{cases} \min\{\mathbf{z}_i - \beta, 1\} & \text{nếu } \mathbf{z}_i - \mathbf{x_0}_i > \beta; \\ \mathbf{x_0}_i & \text{nếu } |\mathbf{z}_i - \mathbf{x_0}_i| \leq \beta; \\ \max\{\mathbf{z}_i + \beta, 0\} & \text{nếu } \mathbf{z}_i - \mathbf{x_0}_i < -\beta \end{cases} \tag{8}$$

Với $i \in \{1, ..., p\}$. Nếu $|\mathbf{z}_i - \mathbf{x_0}_i| > \beta$, thành phần $\mathbf{z}_i$ được co lại với hệ số β và chiếu thành phần kết quả lên miền ràng buộc chấp nhận được thuộc đoạn $[0, 1]$.

Outline

Thực nghiệm: Dữ liệu và phương pháp

- Tập dữ liệu: MNIST, CIFAR10, ImageNet
  - MNIST và CIFAR10 được huấn luyện trên mô hình DNN bởi Carlini và Wagner
  - ImageNet sử dụng mô hình InceptionV3
- Phương pháp tấn công:
  - EAD
  - C&W
  - FGM
  - I-FGM
- Phần cứng: Intel E5-2690 v3 CPU, 40 GB RAM, NVIDIA K80 GPU

Thực nghiệm: Độ đo

- ASR: Tỉ lệ tấn công thành công
- L_1, L_2 và L_∞: Các khoảng cách giữa mẫu đối nghịch và ảnh gốc

Thực nghiệm: Các trường hợp quan tâm

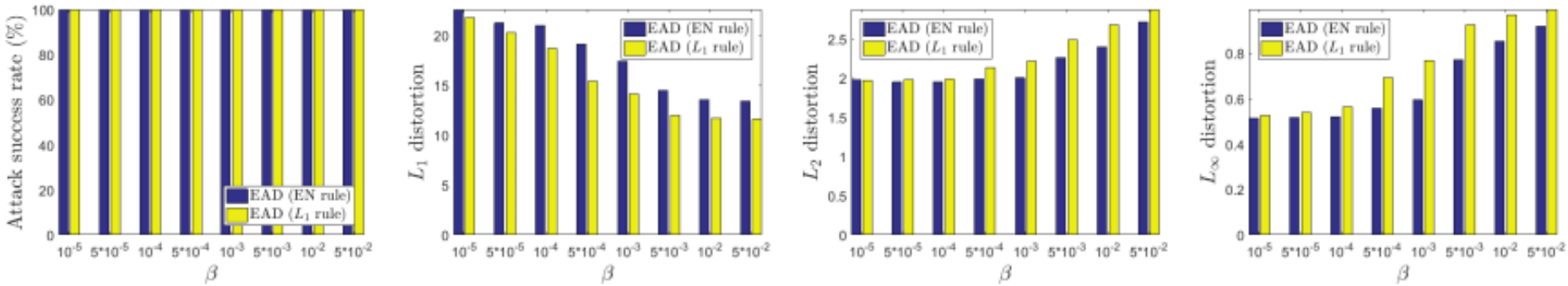
- **Trường hợp tốt nhất (best case):** tấn công dễ nhất về phương diện nhiễu, trong số các tấn công nhắm tới tất cả các lớp sai nhãn.
- **Trường hợp trung bình (average case):** tấn công nhắm ngẫu nhiên vào 1 lớp sai nhãn.
- **Trường hợp xấu nhất (worst case):** tấn công khó nhất về phương diện nhiễu, trong số những tấn công nhắm tới tất cả các lớp sai nhãn.

Thực nghiệm: Độ nhạy

		Best case				Average case				Worst case			
Optimization method	β	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞
COV	0	100	13.93	1.377	0.379	100	22.46	1.972	0.514	99.8	32.3	2.639	0.663
	10^{-5}	100	13.92	1.377	0.379	100	22.66	1.98	0.508	99.5	32.33	2.64	0.663
	10^{-4}	100	13.91	1.377	0.379	100	23.11	2.013	0.517	100	32.32	2.639	0.664
	10^{-3}	100	13.8	1.377	0.381	100	22.42	1.977	0.512	99.9	32.2	2.639	0.664
	10^{-2}	100	12.98	1.38	0.389	100	22.27	2.026	0.53	99.5	31.41	2.643	0.673
EAD (EN rule)	0	100	14.04	1.369	0.376	100	22.63	1.953	0.512	99.8	31.43	2.51	0.644
	10^{-5}	100	13.66	1.369	0.378	100	22.6	1.98	0.515	99.9	30.79	2.507	0.648
	10^{-4}	100	12.79	1.372	0.388	100	20.98	1.951	0.521	100	29.21	2.514	0.667
	10^{-3}	100	9.808	1.427	0.452	100	17.4	2.001	0.594	100	25.52	2.582	0.748
	10^{-2}	100	7.271	1.718	0.674	100	13.56	2.395	0.852	100	20.77	3.021	0.976

Bảng 4.1: So sánh COV và EAD trong việc tìm ra công thức elastic-net trên tập MNIST. ASR là tỷ lệ tấn công thành công. Mặc dù 2 phương pháp trên đều đạt được tỷ lệ tấn công thành công như nhau, COV không hiệu quả trong việc tạo ra các mẫu đối nghịch L_1. Khi β tăng lên, EAD tạo ra các mẫu đối nghịch L_1 ít biến dạng hơn trong khi COV thì không bị ảnh hưởng nhiều khi thay đổi β.

Thực nghiệm: Luật quyết định



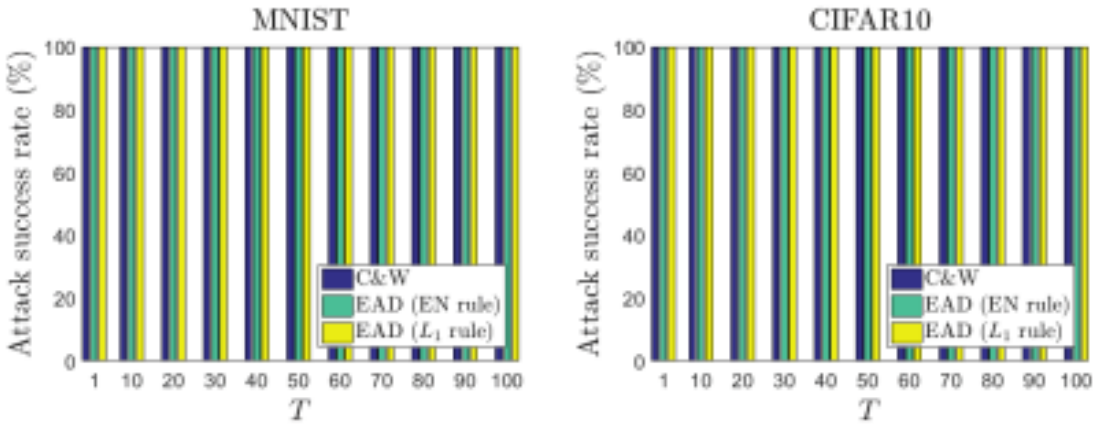
Hình 2: So sánh luật quyết định EN và L_1 trong EAD trên tập MNIST với nhiều tham số hiệu chỉnh L_1 β (trường hợp trung bình). So sánh với luật chọn EN tại cùng giá trị β thì luật chọn L_1 thu được các mẫu ít nhiễu L_1 hơn, nhưng đổi lại có thể bị nhiễu L_2, L_∞ nhiều hơn.

Thực nghiệm: ASR, nhiễu trên MNIST, CIFAR10, ImageNet

	MNIST				CIFAR 10				ImageNet			
Attack method	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞
C&W (L_2)	**100**	22.46	**1.972**	0.514	**100**	13.62	**0.392**	0.044	**100**	232.2	**0.705**	0.03
FGM-L_1	39	53.5	4.186	0.782	48.8	51.97	1.48	0.152	1	61	0.187	0.007
FGM-L_2	34.6	39.15	3.284	0.747	42.8	39.5	1.157	0.136	1	2338	6.823	0.25
FGM-L_∞	42.5	127.2	6.09	0.296	52.3	127.81	2.373	0.047	3	3655	7.102	0.014
I-FGM-L_1	**100**	32.94	2.606	0.591	**100**	17.53	0.502	0.055	77	526.4	1.609	0.054
I-FGM-L_2	**100**	30.32	2.41	0.561	**100**	17.12	0.489	0.054	**100**	774.1	2.358	0.086
I-FGM-L_∞	**100**	71.39	3.472	**0.227**	**100**	33.3	0.68	**0.018**	**100**	864.2	2.079	**0.01**
EAD (EN rule)	**100**	**17.4**	2.001	0.594	**100**	**8.18**	0.502	0.097	**100**	**69.47**	1.563	0.238
EAD (L_1 rule)	**100**	**14.11**	2.211	0.768	**100**	**6.066**	0.613	0.17	**100**	**40.9**	1.598	0.293

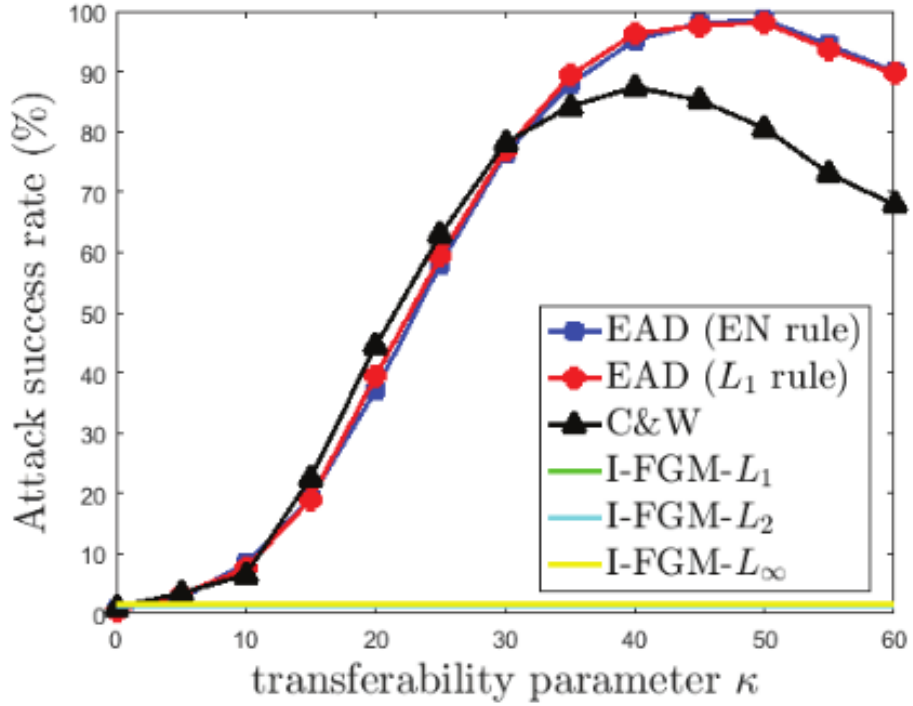
Bảng 4.2: So sánh các tấn công trên các tập dữ liệu MNIST, CIFAR10 và ImageNet. ASR là tỷ lệ tấn công thành công (%). Giá trị nhiễu trong bảng đo được trên giá trị trung bình của các mẫu thành công. EAD, C&W và I-FGM-L_∞ thu được các mẫu ít nhiễu nhất trên các chuẩn L_1,L_2,L_∞ tương ứng. Kết quả đầy đủ được báo cáo trong tài liệu mở rộng.

Thực nghiệm: Phá vỡ chắt lọc phòng thủ



Hình 3: ASR (trường hợp trung bình) của C&W và EAD trên tập MNIST và CIFAR10 với các tham số nhiệt T khác nhau cho chắt lọc phòng thủ. Cả 2 phương pháp đều phá vỡ thành công chắt lọc phòng thủ.

Thực nghiệm: Chuyển giao tấn công



Hình 4: Khả năng chuyển giao tấn công (trường hợp trung bình) từ mạng không phòng thủ sang mạng chắt lọc phòng thủ trên tập dữ liệu MNIST với các tham số κ khác nhau. EAD có thể đạt ASR gần 99% khi $\kappa = 50$, trong khi ASR lớn nhất của C&W là gần 88% khi $\kappa = 40$.

Thực nghiệm: Huấn luyện đối nghịch bổ sung

Attack method	Adversarial training	Average case			
		ASR	L_1	L_2	L_∞
C&W (L_2)	None	100	22.46	1.972	0.514
	EAD	100	26.11	2.468	0.643
	C&W	100	24.97	2.47	0.684
	EAD + C&W	100	27.32	2.513	0.653
EAD (L_1 rule)	None	100	14.11	2.211	0.768
	EAD	100	17.04	2.653	0.86
	C&W	100	15.49	2.628	0.892
	EAD + C&W	100	16.83	2.66	0.87

Bảng 4.3: Huấn luyện đối nghịch sử dụng tấn công C&W và EAD (luật L_1) trên tập dữ liệu MNIST. ASR là tỷ lệ tấn công thành công. Kết hợp các mẫu L_1 bổ sung cho huấn luyện đối nghịch và tăng cường độ khó của tấn công về phương diện nhiễu. Kết quả đầy đủ có trong tài liệu mở rộng.

Outline

Nhận xét: Thời gian tấn công

Phương pháp	EAD-EN	FGM-L1	FGM-L2	IFGM-L1	IFGM-L2
Thời gian (s)	42810	893	1034	3366	9922

Bảng 1: Thời gian tấn công của các thuật toán trên tập CIFAR10

Thời gian tấn công của thuật toán EAD với luật EN là khoảng 11 giờ và gấp khoảng 48 lần khi so sánh với thuật toán nhanh nhất là FGM-L1!

Mở rộng: Thuật toán tổng quát

- Thuật toán FISTA: cực tiểu hóa hàm $f(\mathbf{x}, t)$ trong phương trình 5. Để tính được gradient ∇g ta cần có ràng buộc hàm mất mát của mô hình gốc f phải trơn
- Với một hàm f (lồi) bất kì mà không cần ràng buộc về tính trơn của nó. (Shao, Weijia, Fikret Sivrikaya, & Sahin Albayrak 2022) đã giới thiệu một thuật toán hiệu quả để cực tiểu hóa hàm mục tiêu tổ hợp bằng cập nhật mũ

Mở rộng: Tấn công hệ thống điểm danh GHTK

- EAD tấn công hệ thống nhận diện bằng khuôn mặt
- Tấn công leo thang đặc quyền khi sử dụng khuôn mặt của một nhân viên với quyền thấp hơn để đánh lừa mô hình nhận diện thành một nhân viên với quyền cao hơn
- Framework foolbox [1] đã tích hợp sẵn tấn công EAD và C&W để sinh ra các mẫu đối nghịch

[1] https://github.com/bethgelab/foolbox

Outline

Kết luận

Nhóm tác giả đã đề xuất mô hình tấn công bằng hiệu chỉnh elastic-net để tạo ra các mẫu đối nghịch trong tấn công DNN. Các kết quả thực nghiệm trên các tập dữ liệu MNIST, CIFAR10 và ImageNet cho thấy các mẫu L_1 tạo bởi EAD có thể đạt được tỷ lệ thành công tương đương với các phương pháp tấn công tiên tiến dựa trên L_2 và L_∞ khi phá vỡ mạng không phòng thủ và phòng thủ chưng cất. Ngoài ra, EAD có thể cải thiện khả năng chuyển giao tấn công và huấn luyện đối nghịch bổ sung. Các kết quả của nhóm tác giả đã chứng minh hiệu quả của EAD và đưa ra hướng mới sử dụng mẫu đối nghịch L_1 trong việc huấn luyện đối nghịch và tăng cường an ninh cho DNN.