

TIỂU LUẬN

Khai phá dữ liệu và học máy
Tấn công Elastic-Net vào mạng thần kinh sâu
qua một số mẫu đối nghịch

Đại học Quốc gia Hà Nội
Đại học Khoa học Tự nhiên
Khoa Toán cơ tin

Giảng viên:

Trần Trọng Hiếu

Học viên:

Nguyễn Mạnh Linh, Đào Thị Thu Hồng

Mục lục

1	Giới thiệu	1
2	Các nghiên cứu liên quan	4
2.1	Tấn công vào DNNs	4
2.2	Phòng thủ trong DNNs	5
3	EAD: Tấn công Elastic-Net vào mạng DNNs	6
3.1	Sơ bộ về hiệu chỉnh Elastic-Net	6
3.2	Xây dựng thuật toán EAD	6
3.3	Thuật toán EAD	7
4	Đánh giá hiệu năng	9
4.1	Các phương pháp đối sánh	9
4.2	Thiết kế thực nghiệm	9
4.3	Đánh giá các độ đo	10
4.4	Phân tích độ nạy và luật quyết định	11

Tóm tắt

Các nghiên cứu gần đây đã chỉ ra tính dễ bị tổn thương của các mạng nơ-ron sâu (Deep Neural Networks - DNNs) đến các mẫu đối nghịch - một cách trực quan, hình ảnh đối nghịch không thể phân biệt có thể được tạo ra một cách dễ dàng khiến các mô hình được huấn luyện tốt cũng phân loại ảnh sai. Các phương pháp hiện có để tạo ra mẫu đối nghịch thường dựa trên thông số biến dạng L_2 và L_∞ . Mặc dù trong thực tế độ biến dạng L_1 là quan trọng và quyết định đến tính thưa của nhiễu lại ít được xem xét. Trong bài này, chúng tôi thiết kế một quy trình tấn công DNNs thông qua mẫu đối nghịch như một bài toán tối ưu hóa sử dụng hiệu chỉnh elastic-net. Tấn công DNNs bằng elastic-net (EAD) với tham số L_1 được thêm vào cùng với tấn công L_2 . Kết quả thực nghiệm trên các tập dữ liệu MNIST, CIFAR10 và ImageNet chỉ ra rằng EAD có thể mang lại một tập mẫu đối nghịch với độ nhiễu L_1 nhỏ và đạt được hiệu suất tấn công tương đương với các phương pháp hiện đại nhất qua các kịch bản tấn công khác nhau. Quan trọng hơn, EAD dẫn đến sự cải tiến trong việc tấn công DNN và gợi ý những hiểu biết mới về hiệu chỉnh L_1 để cải thiện bảo mật trong các mô hình học máy.

1 Giới thiệu

Mạng nơ ron sâu (DNNs) đã đạt hiệu quả rất tốt với các bài toán trong học máy và trí tuệ nhân tạo như phân loại ảnh, nhận diện giọng nói, dịch máy và trò chơi. Mặc dù DNNs rất hiệu quả nhưng một số nghiên cứu gần đây đã chứng minh DNNs rất dễ "tổn thương" với các mẫu đối nghịch (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2015). Ví dụ, một hình ảnh với nhiễu được thiết kế cẩn thận có thể làm cho một DNNs đã được huấn luyện phân loại sai. Tệ hơn nữa, các mẫu đối nghịch được tạo ra hầu như không thể phân biệt được bằng mắt người.



Hình 1.1: Minh họa trực quan về mẫu đối nghịch được sinh bởi EAD. Hình gốc (đà điểu) được lấy từ tập ImageNet. Các mẫu đối nghịch bị phân loại sai với mô hình Inception-v3.

Ví dụ trực quan trên thể hiện 3 mẫu đối nghịch của một hình con đà điểu ("ostrich") được sinh ra bằng thuật toán của chúng tôi. Các mẫu này được mô hình Inception-v3 (Szegedy et al. 2016) phân loại thành "safe", "shoe shop" và "vacuum".

Sự thiếu mạnh mẽ của DNNs thể hiện trước các mẫu đối nghịch đã làm dấy lên những lo ngại nghiêm trọng về vấn đề bảo mật các ứng dụng, bao gồm nhận dạng tín hiệu giao thông và phát hiện phần mềm độc hại. Hơn nữa, vượt ra ngoài không gian kỹ thuật số, các nhà nghiên cứu đã chỉ ra rằng những mẫu đối nghịch này vẫn có hiệu quả trong thế giới vật chất trong việc đánh lừa DNNs (Kurakin, Goodfellow, and Bengio 2016a; Evtimov et al. 2017). Do tính mạnh mẽ và ý nghĩa bảo mật, các phương tiện tạo ra các mẫu đối nghịch được gọi là các cuộc tấn công (*attacks*) vào DNNs. Cụ thể, các cuộc tấn công có chủ đích (*targeted attacks*) nhằm mục đích tạo ra các mẫu đối nghịch được phân loại nhầm thành các lớp mục tiêu cụ thể và các cuộc tấn công không nhắm mục tiêu (*untargeted attacks*) nhằm mục đích để tạo ra các mẫu đối nghịch không được phân loại như lớp học ban đầu. Các cuộc tấn công chuyển giao (*transfer attacks*) nhằm mục đích tạo ra các mẫu đối nghịch có thể chuyển từ mô hình DNN này sang mô hình DNN khác. Ngoài việc đánh giá mức độ mạnh mẽ của DNNs, Các mẫu đối nghịch có thể được sử dụng để huấn luyện một mô hình mạnh có khả năng chống chịu với những xáo trộn của đối nghịch, được gọi

là huấn luyện đối nghịch (*adversarial training*) (Madry et al. 2017). Chúng cũng có được sử dụng để giải thích DNNs (Koh và Liang 2017; Dong et al. 2017).

Trong bài báo này, chúng tôi sử dụng các mẫu đối nghịch để tấn công phân loại ảnh dựa trên mạng nơ ron tích chập. Mẫu đối nghịch được tạo ra để làm sai lệch kết quả dự đoán và phải đảm bảo hình mới tạo ra giống với hình gốc. Trong quá khứ, sự giống nhau giữa mẫu đối nghịch được tạo ra và hình gốc được đo bằng các tham số độ méo (*distortion metrics*) khác nhau. Một metric thường được sử dụng là chuẩn L_q với $\|x\|_q = (\sum_{i=1}^p |x_i|^q)^{\frac{1}{q}}$ kí hiệu chuẩn L_q của vector p chiều $x = [x_1, \dots, x_p]$ với $q \geq 1$. Đặc biệt, khi tạo ra các mẫu đối nghịch, độ méo L_∞ được sử dụng để đánh giá sự thay đổi tối đa giá trị pixel (Goodfellow, Shlens, and Szegedy 2015), trong khi độ méo L_2 được sử dụng để cải thiện chất lượng hình ảnh (Carlini and Wagner 2017b). Tuy nhiên, trong thực tế, chuẩn L_1 được sử dụng rộng rãi trong các bài toán phục hồi ảnh (Fu et al. 2006) cũng như phục hồi tính thưa (Candès and Wakin 2008), Các mẫu đối nghịch dựa trên L_1 chưa được xem xét một cách kĩ càng. 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) lỗi đo lường số lượng pixel thay đổi (độ thưa) gây ra bởi nhiễu. Để lấp đầy khoảng trống này, chúng tôi xem xét một thuật toán tấn công dựa trên hiệu chỉnh *elastic-net*, được gọi là tấn công *elastic-net* vào DNNs (*elastic-net attacks to DNNs - EAD*). Hiệu chỉnh *elastic-net* là tổ hợp tuyến tính của các hàm penalty L_1 và L_2 và nó cũng là công cụ tiêu chuẩn cho bài toán lựa chọn tính chất đặc trưng cho dữ liệu nhiều chiều (Zou and Hastie 2005). Trong bài toán tấn công DNNs, EAD mở ra hướng nghiên cứu mới từ cách tấn công hiện đại dựa trên L_2 (Carlini and Wagner 2017b) và cũng đề xuất tấn công hiệu quả hơn theo định hướng L_1 so với các phương pháp tấn công hiện có.

Để khám phá hiệu quả của tấn công dựa trên L_1 , chúng tôi tiến hành các thử nghiệm trên tập dữ liệu MNIST, CIFAR10, và ImageNet trong các tình huống tấn công khác nhau. So sánh với các phương pháp tấn công hiện có dựa trên L_2 và L_∞ (Kurakin, Goodfellow, and Bengio 2016b; Carlini and Wagner 2017b), EAD có thể đạt được tỉ lệ tấn công thành công tương tự khi phá vỡ DNNs được phòng thủ hoặc không phòng thủ (Papernot et al. 2016b). Quan trọng hơn, chúng tôi chỉ ra rằng, tấn công L_1 đạt được hiệu suất vượt trội so với các cuộc tấn công L_2 và L_∞ trong các cuộc tấn công chuyển giao với các mẫu đối thủ được huấn luyện bổ sung. Với tập dữ liệu khó nhất (MNIST), các kết quả từ EAD cải thiện tấn công chuyển giao vào DNN không được phòng thủ hay được phòng thủ đạt tỉ lệ thành công gần 99%. Thêm vào đó việc huấn luyện kèm theo với mẫu đối nghịch dựa trên L_1 và L_2 có thể tăng cường khả năng phục hồi của DNNs đối với các nhiễu loạn. Những kết quả này gợi ý rằng EAD mang lại tập mẫu đối nghịch khác biệt nhưng hiệu quả hơn. Hơn thế nữa, đánh giá các cuộc tấn công dựa trên độ méo L_1 cung cấp thêm hiểu biết mới

về học máy đối kháng và sự bảo mật của DNNs, gợi ý rằng L_1 có thể bổ sung cho L_2 và L_∞ thúc đẩy các framework về học máy đối kháng hoàn thiện hơn.

2 Các nghiên cứu liên quan

Trong phần này, chúng tôi tổng hợp các nghiên cứu liên quan về tấn công và phòng thủ DNNs.

2.1 Tấn công vào DNNs

FGM và 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. Các phương pháp đạo hàm nhanh (*fast gradient methods - FGM*) sử dụng gradient ∇J của hàm mất mát trong tập huấn luyện J với \mathbf{x}_0 để tạo ra các mẫu đối nghịch (Goodfellow, Shlens, and Szegedy 2015). Với tấn công L_∞ , \mathbf{x} được tính bằng công thức:

$$\mathbf{x} = \mathbf{x}_0 - \epsilon \times \text{sign}(\nabla J(\mathbf{x}_0, t)) \quad (2.1)$$

Trong đó ϵ là độ biến dạng L_∞ giữa \mathbf{x} và \mathbf{x}_0 và $\text{sign}(\nabla J)$ là dấu của gradient. Với tấn công L_1 và L_2 , \mathbf{x} được tính bằng công thức:

$$\mathbf{x} = \mathbf{x}_0 - \epsilon \frac{\nabla J(\mathbf{x}_0, t)}{\|\nabla J(\mathbf{x}_0, t)\|_q} \quad (2.2)$$

với $q = 1, 2$ và ϵ là độ méo tương quan. Các phương pháp lặp đạo hàm nhanh (*Iterative fast gradient methods (I-FGM)*) được trình bày trong (Kurakin, Goodfellow, and Bengio 2016b) là phương pháp lặp sửa dụng FGM với một độ méo mịn hơn. Các cuộc tấn công không nhắm mục tiêu sử dụng FGM và I-FGM có thể được thực hiện theo cách tương tự nhau.

C&W attack: 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 toán EAD của chúng tôi (sẽ được trình bày trong phần sau). Tấn công C&W được coi là một trong những tấn công mạnh nhất đối với DNNs vì nó có thể phá vỡ cả những DNNs được phòng thủ và chất lọc, nó cũng có thể đạt được hiệu suất đáng kể với tấn công chuyển giao.

DeepFool: là một thuật toán tấn công không nhắm mục tiêu L_2 (Moosavi-Dezfooli, Fawzi, and Frossard 2016) dựa trên lý thuyết về phép chiếu tới siêu phẳng phân tách gần nhất trong phân lớp. Nó cũng được sử dụng để tạo ra một nhiễu loạn phổ quát nhằm đánh lừa các DNN được huấn luyện trên các hình ảnh tự nhiên (Moosavi-Dezfooli et al. 2016).

2.2 Phòng thủ trong DNNs

Defensive distillation: Chứng cất phòng thủ (Papernot et al. 2016b) bảo vệ chống lại sự nhiễu loạn của đối nghịch bằng cách sử dụng kỹ thuật chưng cất trong (Hinton, Vinyals, and Dean 2015) để huấn luyện lại cùng một mạng với xác suất lớp được dự đoán bởi mạng ban đầu. Phương pháp này đưa ra tham số nhiệt độ T trong lớp softmax để tăng cường độ mạnh đối với những xáo trộn của mẫu đối nghịch.

Adversarial training: Có thể được triển khai theo 1 số cách khác nhau. Hướng tiếp cận chuẩn là thêm vào tập dữ liệu huấn luyện ban đầu các mẫu đối nghịch đã được sửa đúng nhãn, sau đó huấn luyện lại mạng. (Zheng et al. 2016; Madry et al. 2017; Tramèr et al. 2017; Zantedeschi, Nicolae, and Rawat 2017) đề xuất điều chỉnh training loss hoặc kiến trúc mạng để tăng cường sức mạnh của DNNs trước mẫu đối nghịch.

Detection methods: Sử dụng các test thống kê để phân biệt các mẫu đối nghịch với mẫu gốc (Feinman et al. 2017; Grosse et al. 2017; Lu, Issaranon, and Forsyth 2017; Xu, Evans, and Qi 2017). Tuy nhiên, 10 phương pháp phát hiện khác nhau đã không thể phát hiện được tấn công C&W (Carlini and Wagner 2017a).

3 EAD: Tấn công Elastic-Net vào mạng DNNs

3.1 Sơ bộ về hiệu chỉnh Elastic-Net

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). Nó được xem là công cụ hiệu chỉnh trong đó tổ hợp tuyến tính hàm phạt (*penalty*) L_1 và L_2 . Nhìn chung, hiệu chỉnh elastic-net được sử dụng trong bài toán cực tiểu hóa sau đây:

$$\text{minimize}_{\mathbf{z} \in \mathcal{Z}} f(\mathbf{z}) + \lambda_1 \|\mathbf{z}\|_1 + \lambda_2 \|\mathbf{z}\|_2^2 \quad (3.1)$$

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 $\lambda_1 \|\mathbf{z}\|_1 + \lambda_2 \|\mathbf{z}\|_2^2$ là được gọi là hiệu chỉnh elastic-net của \mathbf{z} . Với bài toán hồi quy chuẩn, hàm mất mát $f(\mathbf{z})$ là hàm lỗi trung bình bình phương (*mean squared error - MSE*), vector \mathbf{z} biểu diễn trọng số của các thuộc tính và $\mathcal{Z} = \mathbb{R}^p$. Hiệu chỉnh elastic-net trong phương trình 3.1 chính là công thức Lasso khi $\lambda_2 = 0$ và trở thành công thức hồi quy Ridge khi $\lambda_1 = 0$. (Zou and Hastie 2005) đã chỉ ra rằng hiệu chỉnh elastic-net có thể chọn ra 1 nhóm các thuộc tính tương quan mạnh, mà có thể khắc phục sự thiếu sót của việc chọn lựa thuộc tính nhiều chiều khi sử dụng 1 mình hồi quy Lasso hoặc Ridge.

3.2 Xây dựng thuật toán EAD

Từ tấn công C&W (Carlini and Wagner 2017b), ta thêm vào cùng hàm mất mát f để sinh ra các mẫu đối nghịch. Đặc biệt, 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:

$$f(\mathbf{x}, t) = \max \left(\max_{j \neq t} [\mathbf{Logit}(\mathbf{x})]_j - [\mathbf{Logit}(\mathbf{x})]_t, -\kappa \right) \quad (3.2)$$

Trong đó $\mathbf{Logit}(\mathbf{x}) = [\mathbf{Logit}(\mathbf{x})_1, \dots, \mathbf{Logit}(\mathbf{x})_K] \in \mathbb{R}^K$ 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 $[\mathbf{Logit}(\mathbf{x})]_t$.

Cần lưu ý rằng, thành phần $[\mathbf{Logit}(\mathbf{x})]_t$ là xác suất dự đoán x có nhãn t theo luật phân loại của hàm softmax:

$$\text{Prob}(\text{Label}(\mathbf{x}) = t) = \frac{\exp([\mathbf{Logit}(\mathbf{x})]_t)}{\sum_{j=1}^K \exp([\mathbf{Logit}(\mathbf{x})]_j)} \quad (3.3)$$

Do đó, hàm mất mát trong phương trình 3.2 có mục đích là để cho ra nhãn t là lớp có xác suất 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 3.2 trở thành:

$$f(x) = \max \left([\mathbf{Logit}(\mathbf{x})]_{t_0} - \max_{j \neq t} [\mathbf{Logit}(\mathbf{x})]_j, -\kappa \right) \quad (3.4)$$

Trong bài viết này, chúng tôi tập trung vào tấn công nhắm mục tiêu vì nó khó hơn là không nhắm mục tiêu. Thuật toán EAD có thể được áp dụng trực tiếp vào tấn công không nhắm mục tiêu bằng cách thay $f(x, t)$ trong phương trình 3.2 thành $f(x)$ trong phương trình 3.4.

Ngoài ra, để thao túng kết quả dự đoán thông qua hàm mất mát trong 3.2, 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:

$$\begin{aligned} & \text{minimize}_{\mathbf{x}} \quad c \times f(\mathbf{x}, t) + \beta \|\mathbf{x} - \mathbf{x}_0\|_1 + \|\mathbf{x} - \mathbf{x}_0\|_2^2 \\ & \text{st } \mathbf{x} \in [0, 1]^p \end{aligned} \quad (3.5)$$

Với $f(x, t)$ được xác định trong phương trình 3.2, $c, \beta \geq 0$ lần lượt là các tham số hiệu chỉnh của hàm mất mát f và hàm phạt L_1 . Miền ràng buộc $\mathbf{x} \in [0, 1]^p$ giữ cho \mathbf{x} trong một không gian ảnh mà có thể dễ dàng thỏa mãn điều kiện bằng cách chia mỗi giá trị của pixel cho giá trị lớn nhất (ví dụ 255). Để xác định Khoảng cách (nhiều) giữa \mathbf{x} và \mathbf{x}_0 bằng $\delta = \mathbf{x} - \mathbf{x}_0$, công thức EAD trong 3.5 được tạo ra để tìm một mẫu đối nghịch \mathbf{x} mà sẽ được phân loại vào lớp mục tiêu t , đồng thời cực tiểu hóa δ trong hàm mất mát elastic-net $\beta \|\delta\|_1 + \|\delta\|_2^2$. Đáng chú ý, công thức tấn công C&W (Carlini and Wagner 2017b) trở thành trường hợp đặc biệt của EAD trong công thức 3.5 khi $\beta = 0$, không phụ thuộc vào L_1 của δ . Tuy nhiên thành phần phạt L_1 là một hiệu trực quan để tạo ra mẫu đối nghịch do $\|\delta\|_1 = \sum_{i=1}^p |\delta_i|$ biểu diễn tổng sai khác của nhiều và được sử dụng rộng rãi như là hàm thay thế để phát triển các nhiễu thưa. Phần đánh giá hiệu năng sẽ chứng minh rằng khi đưa thành phần L_1 vào nhiễu sẽ sinh ra tập phân biệt các mẫu đối nghịch, và nó cải thiện khả năng chuyển giao tấn công và triển khai huấn luyện đối nghịch.

3.3 Thuật toán EAD

Khi giải bài toán EAD trong 3.5 mà không có thành phần L_1 , Carlini and Wagner sử dụng kỹ thuật COV (change-of-variable) qua hàm \tanh trên \mathbf{x} nhằm mục đích loại bỏ ràng buộc $\mathbf{x} \in [0, 1]^p$ (Carlini and Wagner 2017b). Khi $\beta > 0$, ta thấy COV sẽ không hiệu quả để giải bài toán 3.5 vì mẫu đối nghịch tương ứng sẽ không nhạy

cảm với thay đổi trên β (xem phần đánh giá hiệu năng để biết chi tiết). Vì L_1 không khả vi trên toàn không gian, COV không giải được bài toán 3.5 do không dùng được phương pháp dưới đạo hàm (*subgradient*) trong việc giải bài toán tối ưu (Duchi and Singer 2009).

Để giải bài toán EAD trong 3.5 để sinh ra các mẫu đối nghịch, bài viết này đề xuất sử dụng thuật toán ISTA (*iterative shrinkage-thresholding algorithm* – *thuật toán lặp với ngưỡng biến đổi*) (Beck and Teboulle 2009). ISTA là một thuật toán tối ưu bậc nhất phổ biến, ISTA thêm một bước điều chỉnh ngưỡng trong mỗi bước lặp. Cụ thể, đặt $g(\mathbf{x}) = c \times f(\mathbf{x}) + \|\mathbf{x} - \mathbf{x}_0\|_2^2$ và đặt $\nabla g(\mathbf{x})$ là gradient của $g(\mathbf{x})$ được tính bởi mạng DNN. Tại bước lặp thứ $k+1$, mẫu đối nghịch $\mathbf{x}^{(k+1)}$ của \mathbf{x}_0 được tính như sau:

$$\mathbf{x}^{(k+1)} = S_\beta(\mathbf{x}^{(k)} - \alpha_k \nabla g(\mathbf{x}^{(k)})) \quad (3.6)$$

Trong đó, α_k là độ dài bước tại bước lặp thứ $k+1$, $S_\beta : \mathbb{R}^p \rightarrow \mathbb{R}^p$ là hàm của phép chiếu biến đổi ngưỡng trên từng phần tử, được xác định bởi:

$$[S_\beta(\mathbf{z})]_i = \begin{cases} \min\{\mathbf{z}_i - \beta, 1\} & \text{nếu } \mathbf{z}_i - \mathbf{x}_{0i} > \beta; \\ \mathbf{x}_{0i} & \text{nếu } |\mathbf{z}_i - \mathbf{x}_{0i}| \leq \beta; \\ \max\{\mathbf{z}_i + \beta, 0\} & \text{nếu } \mathbf{z}_i - \mathbf{x}_{0i} < -\beta \end{cases} \quad (3.7)$$

Với $i \in \{1, \dots, p\}$. Nếu $|\mathbf{z}_i - \mathbf{x}_{0i}| > \beta$, 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 $[0, 1]$. Mặt khác, nếu $|\mathbf{z}_i - \mathbf{x}_{0i}| \leq \beta$, nó đặt ngưỡng \mathbf{z}_i bằng cách thiết lập $[S_\beta(\mathbf{z})]_i = \mathbf{x}_{0i}$. Chúng minh thuật toán tối ưu sử dụng 3.6 để giải bài toán EAD trong 3.5 được trình bày trong tài liệu (<https://arxiv.org/abs/1709.04114>)

Thuật toán EAD để tạo mẫu đối nghịch được tổng kết trong thuật toán 1. Để tính toán hiệu quả, ta triển khai FISTA (fast ISTA) cho EAD, nó sẽ cho tốc độ hội tụ tối ưu với phương pháp tối ưu hóa bậc 1 (Beck and Teboulle 2009). Vector $\mathbf{y}^{(k)}$ trong thuật toán 1 kết hợp với momen trong $\mathbf{x}^{(k)}$. Trong phép lặp EAD, bước lặp $\mathbf{y}^{(k)}$ được coi là mẫu đối nghịch thành công của \mathbf{x}_0 nếu mô hình dự báo thành lớp đích t . Mẫu đối nghịch \mathbf{x} cuối cùng được lựa chọn từ tất cả các mẫu đối nghịch thành công. Trong bài viết này, chúng ta xem xét 2 luật để chọn \mathbf{x} từ tập các mẫu đối nghịch thành công: elastic-net tối thiểu (EN) và nhiễu L_1 tương ứng với \mathbf{x}_0 . Sự ảnh hưởng của β , κ và luật quyết định lên EAD sẽ được nghiên cứu trong phần tiếp theo.

4 Đánh giá hiệu năng

Trong phần này, chúng ta so sánh EAD với các thuật toán tấn công tiên tiến hiện nay trên 3 tập dữ liệu ảnh: MNIST, CIFAR10 và ImageNet. Tác giả bài viết muốn cho thấy:

- EAD có thể thu được hiệu năng tấn công tương tự tấn công C&W đối với mạng DNNs phòng thủ và mạng DNN không phòng thủ do C&W là trường hợp đặc biệt của EAD khi $\beta = 0$.
- So sánh với các phương pháp tấn công hiện tại dựa trên L_1 như FGM, I-FGM, các mẫu đối nghịch thu được từ EAD có nhiều L_1 nhỏ hơn đáng kể và tỷ lệ tấn công thành công cao hơn.
- Các mẫu đối nghịch tạo bởi EAD có thể hỗ trợ tấn công chuyển giao và triển khai huấn luyện đối nghịch.

4.1 Các phương pháp đối sánh

Bài viết này so sánh EAD với các phương pháp tấn công nhằm đích hiệu quả nhất có sử dụng các loại hiệu chỉnh khác nhau:

- C&W: tấn công nhằm đích dùng hiệu chỉnh L_2 được đề xuất bởi Carlini and agner (Carlini and Wagner 2017b). Đây là trường hợp đặc biệt của EAD khi $\beta = 0$.
- FGM (fast gradient method): được đề xuất bởi (Goodfellow, Shlens, and Szegedy 2015). FGM sử dụng nhiều loại hiệu chỉnh với các phiên bản FGM- L_1 , FGM- L_1 , FGM- L_∞ .
- I-FGM: (FGM lặp): được đề xuất bởi (Kurakin, Goodfellow, and Bengio 2016b). I-FGM sử dụng nhiều loại hiệu chỉnh I-FGM- L_1 , I-FGM- L_2 , I-FGM- L_∞ .

4.2 Thiết kế thực nghiệm

Thực nghiệm sử dụng framework của Carlini và Wagner. Với cả tấn công EAD và C&W, tác giả đều sử dụng các tham số cấu hình mặc định, trong đó triển khai 9 bước tìm kiếm nhị phân trên tham số hiệu chỉnh (bắt đầu từ 0.001) và chạy vòng lặp cho mỗi bước với learning rate khởi tạo . Để tìm các mẫu đối nghịch thành công, tác giả sử dụng thuật toán tối ưu tham chiếu (ADAM) cho C&W, và sử dụng thuật

toán chiếu FISTA (thuật toán 1) với square-root decaying learning rate cho EAD. Tương tự tấn công C&W, mẫu đối nghịch cuối cùng tìm được từ EAD được chọn là mẫu nhiễu ít nhất giữa các mẫu đối nghịch thành công. Độ nhạy cảm của L1 (tham số κ) và tác động của luật chọn trong EAD sẽ được bàn đến ở phần sau. Tác giả đặt tham số tấn công chuyển giao $\kappa = 0$ cho cả EAD và C&W.

Tác giả triển khai FGM và I-FGM bằng gói thư viện CleverHans. Tham số nhiễu tốt nhất ϵ được xác định bằng phương pháp tìm kiếm lưới fine-grained trên mỗi ảnh, ϵ nhỏ nhất trong lưới thể hiện tấn công thành công sẽ được ghi lại. Với I-FGM, tác giả thực hiện 10 vòng lặp FGM (10 là giá trị mặc định) với ϵ -ball clipping. Tham số nhiễu ϵ' trong mỗi vòng lặp FGM được đặt là $\epsilon/10$, đã được chứng minh hiệu quả trong (Tramer et al. 2017). Dải giá trị của lưới và độ phân giải của 2 phương pháp trên được đề cập cụ thể trong tài liệu bổ sung 1.

Việc phân loại ảnh cho tập MNIST và CIFAR10 được huấn luyện trên mô hình DNN bởi Carlini and Wagner. Để phân loại cho tập ImageNet, người ta dùng mô hình Inception-v3 (Szegedy et al. 2016). Đối với tập MNIST và CIFAR10, tác giả lấy ngẫu nhiên trong tập test 1000 ảnh đã được phân loại đúng để tấn công làm phân loại sai. Với tập ImageNet, tác giả chọn ngẫu nhiên 100 ảnh đã được phân loại đúng và 9 ảnh được phân loại sai để tấn công. Tất cả thực nghiệm được triển khai trên thiết bị phần cứng Intel E5-2690 v3 CPU, 40 GB RAM, single NVIDIA K80 GPU.

4.3 Đánh giá các độ đo

Theo cách đánh giá trong (Carlini and Wagner 2017b), tác giả báo cáo tỷ lệ tấn công thành công và độ nhiễu của mẫu đối nghịch trong mỗi phương pháp. Tỷ lệ tấn công thành công (ASR) là phần trăm mẫu đối nghịch được phân loại vào lớp đích (khác lớp gốc). Các trung bình L_1 , L_2 và L_∞ của các mẫu đối nghịch thành công cũng được ghi chép báo cáo. Cụ thể, các trường hợp sau được quan tâm:

- **Trường hợp tốt nhất:** Tấn công ít khó nhất trong số các tấn công nhắm đích sử dụng nhiễu.
- **Trường hợp trung bình:** Tấn công nhắm đích vào ngẫu nhiên 1 lớp gán nhãn sai.
- **Trường hợp xấu nhất:** tấn công khó nhất trong số những tấn công nhắm đích sử dụng nhiễu.

4.4 Phân tích độ nạy và luật quyết định

Tác giả kiểm tra sự cần thiết của việc sử dụng thuật toán 1 để giải bài toán tấn công bằng hiệu chỉnh elastic-net trong (3.5) bằng cách so sánh nó với thuật toán COV thuần. Trong tài liệu (Carlini and Wagner 2017b), Carlini và Wagner đã loại bỏ điều kiện ràng buộc $\mathbf{x} \in [0, 1]^p$ bằng cách thay \mathbf{x} bằng $\frac{\mathbf{1} + \tanh \mathbf{w}}{2}$ với $\mathbf{w} \in \mathbb{R}^p$ và $\mathbf{1} \in \mathbb{R}^p$ là vector các với các phần tử 1.