

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ



NGUYỄN CÔNG THẠNH

**NGHIÊN CỨU KỸ THUẬT
DEEPPAKES ỨNG DỤNG TRONG TẤN
CÔNG ĐỐI KHÁNG**

LUẬN VĂN THẠC SĨ

TP HỒ CHÍ MINH - 2023

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

NGUYỄN CÔNG THẠNH

NGHIÊN CỨU KỸ THUẬT
DEEPPAKES ỨNG DỤNG TRONG TẤN
CÔNG ĐỐI KHÁNG

Ngành: An Toàn Thông Tin

Mã số: 8.48.02.02

Khóa: 01

LUẬN VĂN THẠC SĨ

Người hướng dẫn khoa học:

TS. Phạm Duy Trung - Học viện Kỹ Thuật Mật Mã

TS. Nguyễn Trọng Khánh - Học viện Bưu Chính Viễn Thông

TP HỒ CHÍ MINH - 2023

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

Số: 230/TrQĐ-HVM

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Hà Nội, ngày 14 tháng 3 năm 2023

QUYẾT ĐỊNH

Về việc giao đề tài luận văn tốt nghiệp năm 2023
- Đào tạo thạc sĩ ngành ATTT tại TP. Hồ Chí Minh

GIÁM ĐỐC HỌC VIỆN KỸ THUẬT MẬT MÃ

Căn cứ Thông tư số 23/2021/TT-BGDĐT ngày 30 tháng 8 năm 2021 của Bộ trưởng Bộ Giáo dục và Đào tạo về việc ban hành quy chế tuyển sinh và đào tạo trình độ thạc sĩ;

Căn cứ Thông tư số 54/2014/TT-BQP ngày 12/06/2014 của Bộ trưởng Bộ Quốc phòng quy định chức năng, nhiệm vụ, quyền hạn và mối quan hệ công tác của Học viện Kỹ thuật mật mã;

Căn cứ Kết quả của Tiểu ban xét duyệt đề cương luận văn thạc sĩ năm 2023 - đào tạo trình độ thạc sĩ ngành An toàn thông tin tại TP. Hồ Chí Minh; Theo đề nghị của Trưởng phòng Sau đại học.

QUYẾT ĐỊNH:

Điều 1. Giao đề tài luận văn thạc sĩ “Nghiên cứu kỹ thuật deepfakes ứng dụng trong tấn công đối kháng” cho học viên:

Họ và tên: **Nguyễn Công Thạnh**

Khóa 01 (2021-2023)

Điều 2. Giao nhiệm vụ hướng dẫn luận văn thạc sĩ cho:

Cán bộ hướng dẫn 1: TS. Phạm Duy Trung

Đơn vị công tác: Học viện KTMM

Cán bộ hướng dẫn 2: TS. Nguyễn Trọng Khánh

Đơn vị công tác: Học viện Bưu chính viễn thông

Ngày nộp luận văn: 14 tháng 7 năm 2023.

Điều 3. Trưởng phòng Sau đại học, cán bộ hướng dẫn và học viên có tên trên chịu trách nhiệm thi hành Quyết định này. /.

Nơi nhận:

- Như điều 3;

- Lưu: VT, SDH. CH05.

**KT. GIÁM ĐỐC
PHÓ GIÁM ĐỐC**



Lương Thế Dũng

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

Số: 1026/QĐ-HVM

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh Phúc

Hà Nội, ngày 15 tháng 9 năm 2023

QUYẾT ĐỊNH

Về việc thành lập Hội đồng chấm luận văn tốt nghiệp
Thạc sĩ ngành An toàn thông tin tại TP. Hồ Chí Minh khóa 01

GIÁM ĐỐC HỌC VIỆN KỸ THUẬT MẬT MÃ

Căn cứ Thông tư 54/2014/TT-BQP ngày 12/6/2014 của Bộ trưởng Bộ Quốc phòng quy định chức năng, nhiệm vụ, quyền hạn và mối quan hệ công tác của Học viện Kỹ thuật mật mã;

Căn cứ Thông tư số 23/2021/TT-BGDĐT ngày 30 tháng 8 năm 2021 của Bộ trưởng Bộ Giáo dục và Đào tạo ban hành quy chế tuyển sinh và đào tạo trình độ thạc sĩ;

Căn cứ Kế hoạch số 1017/KH-HVM ngày 15 tháng 9 năm 2023 của Giám đốc Học viện Kỹ thuật mật mã về việc tổ chức bảo vệ Luận văn tốt nghiệp thạc sĩ ngành ATTT tại TP.HCM năm 2023 (Lớp CHAT1P);

Theo đề nghị của Trưởng phòng Đào tạo.

QUYẾT ĐỊNH:

Điều 1. Thành lập Hội đồng chấm Luận văn tốt nghiệp Thạc sĩ:

Học viên: **Nguyễn Công Thanh**

Khoá 01 - đào tạo trình độ thạc sĩ ngành An toàn thông tin tại TP.HCM.

Đề tài: **“Nghiên cứu kỹ thuật deepfakes ứng dụng trong tấn công đối kháng”**,
gồm các đồng chí sau:

1	PGS. TS. Lương Thế Dũng	Học viện Kỹ thuật mật mã	Chủ tịch
2	TS. Nguyễn Mạnh Thắng	Học viện Kỹ thuật mật mã	Thư ký
3	PGS. TS. Nguyễn Long Giang	Viện Hàn lâm KHVN	Phản biện 1
4	TS. Nguyễn Tuấn Anh	Học viện Kỹ thuật mật mã	Phản biện 2
5	PGS. TS. Trần Đăng Hưng	Đại học Sư phạm HN	Ủy viên

Điều 2. Hội đồng có nhiệm vụ đánh giá Luận văn tốt nghiệp của Học viên theo quy định hiện hành. Hội đồng tự giải thể sau khi hoàn thành nhiệm vụ.

Điều 3. Cử đồng chí Cao Thị Hồng, Trợ lý phòng Đào tạo làm Thư ký giúp Hội đồng hoàn thiện hồ sơ, thủ tục theo quy định.

Điều 4. Trưởng phòng Đào tạo và các đồng chí có tên trên chịu trách nhiệm thi hành Quyết định này. /

Nơi nhận:

- Như điều 4;
- BGĐ² (để báo cáo);
- Khoa ATTT;
- Các phòng: ĐT³, KH-TC;
- Phân hiệu HVKTMM;
- Lưu: VT, ĐT, CH15.

KT. GIÁM ĐỐC
PHÓ GIÁM ĐỐC



Lương Thế Dũng

LỜI CAM ĐOAN

Tôi xin cam đoan luận văn “NGHIÊN CỨU KỸ THUẬT DEEPPAKES ỨNG DỤNG TRONG TẤN CÔNG ĐỐI KHÁNG” này là công trình nghiên cứu của cá nhân, không sao chép của ai, những nội dung, kiến thức trình bày trong luận văn là do tôi tìm hiểu tài liệu, nghiên cứu và trình bày theo cách hiểu của bản thân dưới sự hướng dẫn của TS. Phạm Duy Trung và TS. Nguyễn Trọng Khánh. Các nội dung được trình bày trong luận văn và kết quả thực nghiệm là hoàn toàn trung thực.

Trong quá trình thực hiện luận văn, tôi có tham khảo các tài liệu liên quan từ các nguồn khác nhau và có ghi nguồn tham khảo tại phần tài liệu tham khảo ở cuối luận văn.

Nếu có bất cứ điều gì sai sót, tôi hoàn toàn chịu trách nhiệm.

Hồ Chí Minh, 18 tháng 09 năm 2023

Học viên

Nguyễn Công Thạnh

LỜI CẢM ƠN

Để hoàn thành chương trình khóa học và hoàn thành luận văn tốt nghiệp này, tôi đã nhận được sự hướng dẫn và giúp đỡ nhiệt tình của các thầy cô Học viện kỹ thuật Mật mã. Tôi xin chân thành cảm ơn các thầy cô giáo bộ môn đã hết lòng dạy dỗ và truyền đạt những kiến thức quý báu cho chúng tôi trong suốt quá trình học tập của mình.

Đồng thời, tôi xin gửi lời cảm ơn sâu sắc đến TS. Phạm Duy Trung và TS. Nguyễn Trọng Khánh - người đã trực tiếp hướng dẫn tôi thực hiện luận văn này. Thầy đã tận tình chỉ bảo, cung cấp cho tôi những kiến thức, tài liệu, phương pháp nghiên cứu một vấn đề khoa học và giúp tôi định hướng tốt các vấn đề trong quá trình thực hiện luận văn.

Tôi xin gửi lời cảm ơn đến đồng nghiệp trong TEK4.VN đã tạo điều kiện và giúp đỡ tôi trong suốt thời gian nghiên cứu làm khóa luận.

Cũng xin gửi lời cảm ơn chân thành tới Ban Giám đốc học viện, Phòng Đào tạo sau đại học - Học viện kỹ thuật Mật mã đã tạo điều kiện tốt nhất cho chúng tôi trong suốt quá trình học tập.

Tôi xin gửi lời cảm ơn đến các thành viên trong lớp Cao học An toàn thông tin CHAT1P đã giúp đỡ tôi trong suốt thời gian học tập vừa qua.

Cuối cùng tôi xin gửi lời cảm ơn đến gia đình, bạn bè - những người đã luôn bên cạnh động viên, giúp đỡ và khuyến khích tôi trong suốt quá trình học tập và thực hiện đề tài nghiên cứu của mình.

Tôi xin chân thành cảm ơn!

Hồ Chí Minh, Tháng 08 năm 2023

Học viên

Nguyễn Công Thạnh

MỤC LỤC

LỜI CAM ĐOAN	iii
LỜI CẢM ƠN.....	iv
MỤC LỤC	v
DANH MỤC TỪ VIẾT TẮT	vii
DANH MỤC HÌNH VẼ	viii
DANH MỤC BẢNG.....	vii
MỞ ĐẦU.....	1
CHƯƠNG 1: TỔNG QUAN VỀ HỌC MÁY VÀ TẤN CÔNG ĐỐI KHÁNG	3
1.1. Tổng quan về học sâu, mô hình học sâu	3
1.1.1. Học máy và học sâu	3
1.1.2. Các mô hình học sâu	5
1.2. Tổng quan về deepfakes	6
1.2.1. Deepfake và các vấn đề có liên quan.....	7
1.2.2. Phương pháp tạo ra deepfakes.....	9
1.3. Tổng quan về tấn công đối kháng	10
1.3.1. Khái niệm tấn công đối kháng	10
1.3.2. Phân loại tấn công đối kháng	12
1.3.3. Một số phương pháp tấn công theo sự hiểu biết về mô hình	13
1.3.4. Tổng quan về chống lại các cuộc tấn công đối kháng	21
1.4. Kết luận chương 1	23
CHƯƠNG 2: PHƯƠNG PHÁP TẤN CÔNG ĐỐI KHÁNG SỬ DỤNG DEEPFAKES.....	24
2.1. Tổng quan về tấn công đối kháng sử dụng deepfake.....	24
2.2. Quy trình tấn công đối kháng sử dụng deepfake	25
2.2.1. Các bước tấn công đối kháng sử dụng deepfakes	25

2.2.2. Xác định mục tiêu và thu thập dữ liệu huấn luyện	26
2.2.3. Xây dựng mô hình và tạo mẫu đối kháng.....	34
2.2.4. Thực hiện tấn công đối kháng.....	37
2.3. Phương pháp và tiêu chí đánh giá	37
2.4. Kết luận chương 2	39
CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ.....	41
3.1. Chuẩn bị dữ liệu huấn luyện	41
3.2. Huấn luyện mô hình	41
3.3. Thử nghiệm tạo mẫu đối kháng và đánh giá mô hình tạo mặt nạ nhiễu.....	48
3.3.1. Thử nghiệm với mô hình repvgg_a2	49
3.3.2. Thử nghiệm với mô hình shufflenetv2	51
3.3.3. Thử nghiệm với mô hình mobilenetv2	53
3.3.4. Thử nghiệm với mô hình vgg19_bn.....	55
3.3.5. Thử nghiệm với mô hình resnet56	57
3.4. Thử nghiệm tấn công đối kháng	58
3.4.1. Thử nghiệm tấn công đối kháng vào mô hình repvgg_a2.....	59
3.4.2. Thử nghiệm tấn công đối kháng vào mô hình shufflenetv2.....	61
3.4.3. Thử nghiệm tấn công đối kháng vào mô hình mobilenetv2.....	63
3.4.4. Thử nghiệm tấn công đối kháng vào mô hình vgg19_bn.....	65
3.4.5. Thử nghiệm tấn công đối kháng vào mô hình resnet56	67
3.5. Kết luận chương 3	69
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	70
TÀI LIỆU THAM KHẢO.....	71

DANH MỤC TỪ VIẾT TẮT

CHỮ VIẾT TẮT	NGHĨA TIẾNG ANH	NGHĨA TIẾNG VIỆT
AA	Adversarial Attack	Tấn công đối kháng
AI	Artificial Intelligence	Trí tuệ nhân tạo
ASR	Automatic Speech Recognition Systems	Hệ thống nhận diện tiếng nói tự động
ATN	Adversarial Transformation networks	Mạng chuyển đổi đối nghịch
CSDL	Database	Cơ sở dữ liệu
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
DL	Deep Learning	Học sâu
DNN	Deep Neural Network	Mạng thần kinh sâu
IoT	Internet of Things	Internet vạn vật
SMC	Secure Multiparty Computation	Tính toán bảo mật nhiều thành viên
ML	Machine Learning	Học máy
VCS	Voice Communication System	Điều khiển bằng giọng nói
SSIM	Structural Similarity Index	Chỉ số tương đồng cấu trúc
PSNR	Peak Signal-to-Noise Rati	Tỷ lệ tín hiệu trên nhiễu
Adv	Adversarial images	Ảnh đối kháng - ảnh giả
G	Generator	Mạng sinh
D	Discriminator	Mạng so sánh

DANH MỤC HÌNH VẼ

Hình 1.1. Thuật toán tạo deepfakes	7
Hình 1.2. Thực nghiệm tạo mẫu đối kháng thực hiện bởi Christian Szegedy, 2014	11
Hình 2.1. Sơ đồ mô phỏng các bước tấn công đối kháng sử dụng deepfake	26
Hình 2.2. Hình mẫu dataset CIFAR10.....	27
Hình 2.3. Sơ đồ quá trình huấn luyện mô hình và tạo mẫu đối kháng.....	35
Hình 3.1. Cấu hình máy tính sử dụng thực nghiệm.....	41
Hình 3.2. Cấu hình VGA sử dụng huấn luyện mô hình	42
Hình 3.3. Mô tả quá trình huấn luyện	43
Hình 3.4. Đồ thị biểu diễn quá trình huấn luyện mô hình	46
Hình 3.5. Đồ thị biểu diễn mô hình sau khi tuning	47
Hình 3.6. Ảnh mẫu thực nghiệm trên mô hình repvgg_a2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra	50
Hình 3.7. Ảnh mẫu thực nghiệm trên mô hình shufflenetv2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra.....	52
Hình 3.8. Ảnh mẫu thực nghiệm trên mô hình mobilenetv2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra.....	54
Hình 3.9. Ảnh mẫu thực nghiệm trên mô hình vgg19_bn, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra	56
Hình 3.10. Ảnh mẫu thực nghiệm trên mô hình resnet56, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra	58

DANH MỤC BẢNG

Bảng 2.1. Bảng thông số đánh giá các model đã công bố của dataset CIFAR10.....	27
Bảng 3.1. Bảng thông số điều chỉnh mô hình	46
Bảng 3.2. Bảng kết quả huấn luyện mô hình	48
Bảng 3.3. Kết quả thử nghiệm với mô hình shufflenetv2.....	51
Bảng 3.4. Kết quả thử nghiệm với mô hình mobilenetv2.....	53
Bảng 3.5. Kết quả thử nghiệm với mô hình vgg19_bn.....	55
Bảng 3.6. Kết quả thử nghiệm với mô hình resnet56	57
Bảng 3.7. Bảng kết quả tấn công đối kháng vào mô hình repvgg_a2.....	59
Bảng 3.8. Bảng trích xuất mẫu đối kháng với mô hình repvgg_a2.....	60
Bảng 3.9. Bảng kết quả tấn công đối kháng vào mô hình shufflenetv2	61
Bảng 3.10. Bảng trích xuất mẫu đối kháng với mô hình shufflenetv2.....	62
Bảng 3.11. Bảng kết quả tấn công đối kháng vào mô hình mobilenetv2	63
Bảng 3.12. Bảng trích xuất mẫu đối kháng với mô hình mobilenetv2.....	64
Bảng 3.13. Bảng kết quả tấn công đối kháng vào mô hình vgg19_bn.....	65
Bảng 3.14. Bảng trích xuất mẫu đối kháng với mô hình vgg19_bn.....	66
Bảng 3.15. Bảng kết quả tấn công đối kháng vào mô hình resnet56	67
Bảng 3.16. Bảng trích xuất mẫu đối kháng với mô hình resnet56.....	68
Bảng 3.17. Bảng tổng kết kết quả thử nghiệm tấn công đối kháng	69

MỞ ĐẦU

Ngày nay, công nghệ AI đang phát triển mạnh mẽ và có khả năng tạo ra các hình ảnh, âm thanh và video giả mạo rất chính xác. Việc tấn công vào mô hình học sâu có thể dẫn đến việc tạo ra các kết quả sai lệch hoặc giả mạo, đe dọa đến tính toàn vẹn và tin cậy của các hệ thống tự động hoạt động dựa trên mô hình học sâu. Đặc biệt, Deepfakes đang là một trong những nguy cơ lớn nhất đối với tính đúng đắn và tin cậy của thông tin truyền tải trên các dạng media đồ họa và âm thanh. Vì vậy, việc nghiên cứu tấn công đối kháng trong mô hình học sâu cần được thực hiện một cách kỹ càng và nghiêm túc để đảm bảo an toàn và tính toàn vẹn của các hệ thống tự động hoạt động trên mô hình học sâu. Điều này có thể dẫn đến việc tạo ra các Deepfakes chất lượng cao và gây mất tin tưởng với người dùng.

Với sự tiến bộ nhanh chóng đạt được những thành công đáng kể trong đa số các ứng dụng, Deep Learning (học sâu) đang được áp dụng trong nhiều lĩnh vực quan trọng về an toàn thông tin. Tuy nhiên, Deep Neural Network (mạng thần kinh sâu) gần đây đã được phát hiện dễ bị ảnh hưởng bởi mẫu đầu vào, được gọi là Adversarial examples (mẫu đối kháng). Những thay đổi làm chúng ta không thể nhận thấy nhưng có thể dễ dàng đánh lừa mạng nơ-ron trong giai đoạn thử nghiệm/triển khai. Lỗi hồng đối với các mẫu đối kháng trở thành một trong những rủi ro lớn đối với việc ứng dụng DNN trong các lĩnh vực quan trọng về an toàn. Do đó, các cuộc tấn công và phòng thủ dựa trên các mẫu đối kháng đặt ra vấn đề lớn.

Ngoài ra, những thách thức lớn trong các mẫu đối kháng và việc phát triển các giải pháp cũng được nhiều nhà nghiên cứu quan tâm. Phần lớn các DNN bị đánh lừa bởi các mẫu đối kháng (Adversarial Example) bằng cách thêm vào những nhiễu nhỏ trong ảnh đầu vào khiến không thể nhận ra bằng mắt thường của con người, nhưng làm đầu ra của mô hình cho dự đoán không chính xác. Việc này thúc đẩy tạo ra ngày càng nhiều mẫu đối kháng để xác định các lỗ hổng bảo mật của mô hình trước khi đưa chúng vào sử dụng. Bên cạnh đó, mẫu đối kháng cũng tạo điều kiện cho các thuật toán DNN khác nhau đạt được độ ổn định bằng cách cung cấp dữ liệu huấn luyện đa dạng hơn.

Với mong muốn phát hiện hạn chế và khắc phục nhược điểm của các mô hình DNN nói riêng và Machine Learning (ML) nói chung, đề tài tập trung nghiên cứu tiếp cận các phát hiện gần đây về các mẫu đối kháng cho DNN, tóm tắt và đề xuất một số phương pháp để tạo một số cuộc tấn công sử dụng mẫu đối kháng.

Nghiên cứu để bảo đảm độ an toàn cho mô hình học máy có thể giúp an toàn mạng và mô hình một cách chủ động hơn giảm thiểu nguy cơ bị xâm nhập, đánh lừa các hệ thống nhận biết từ đó tiết kiệm chi phí và tăng hiệu quả trong công việc của cơ quan, tổ chức. Do vậy, “NGHIÊN CỨU KỸ THUẬT DEEPFAKES ỨNG DỤNG TRONG TẤN CÔNG ĐỐI KHÁNG” là một hướng nghiên cứu đang được quan tâm đầu tư nghiên cứu, có tính cấp thiết và ý nghĩa khoa học, thực tiễn cao trong việc nâng cao độ an toàn thông tin trong lĩnh vực An ninh - Quốc phòng và Kinh tế - Xã hội, ...

Nghiên cứu sẽ tập trung vào việc tìm hiểu và phát triển kỹ thuật deepfakes, cũng như các kỹ thuật tấn công đối kháng bằng deepfakes được tạo ra nhằm phá vỡ các mô hình. Phạm vi của đề tài nghiên cứu bao gồm các bộ dữ liệu và các tiêu chí đánh giá phù hợp để đánh giá hiệu quả của các phương pháp đề xuất.

Từ mục tiêu “NGHIÊN CỨU KỸ THUẬT DEEPFAKES ỨNG DỤNG TRONG TẤN CÔNG ĐỐI KHÁNG”, luận văn sẽ làm rõ lý thuyết tổng quan về Deepfakes và các phương pháp tấn công mô hình học máy, sử dụng mô hình học máy để tạo deepfakes. Phân tích lựa chọn, thực nghiệm thuật toán để tạo deepfakes tấn công mô hình học máy. Bố cục của luận văn được chia làm 3 chương như sau:

CHƯƠNG 1: TỔNG QUAN VỀ HỌC MÁY VÀ TẤN CÔNG ĐỐI KHÁNG

Chương này luận văn trình bày tổng quan các kiến thức về học máy và tấn công đối kháng bao gồm các khái niệm, phương pháp tạo ra deepfakes và phương pháp tấn công đối kháng sử dụng deepfakes.

CHƯƠNG 2: PHƯƠNG PHÁP TẤN CÔNG ĐỐI KHÁNG SỬ DỤNG DEEPFAKES

Trong chương 2 luận văn trình bày về quy trình và kỹ thuật sử dụng học sâu để tạo ra mẫu đối kháng, mẫu đối kháng chính là kết quả deepfakes. Trình bày các định nghĩa và phương pháp đề xuất cải tiến để tạo ra deepfakes và tấn công vào mô hình học máy.

CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ

Trong chương 3 luận văn trình bày quá trình thực nghiệm và kết quả thực nghiệm. Định nghĩa các thông số đánh giá, và đánh giá kết quả đạt được.

Tôi đã rất cố gắng trình bày luận văn chính xác về nội dung, mạch lạc trong hình thức, song không thể tránh được những thiếu sót. Tôi rất mong nhận được những ý kiến chân thành của thầy cô để luận văn được hoàn thiện hơn.

Hồ Chí Minh, Tháng 09 năm 2023

Học viên

CHƯƠNG 1: TỔNG QUAN VỀ HỌC MÁY VÀ TẤN CÔNG ĐỐI KHÁNG

1.1. Tổng quan về học sâu, mô hình học sâu

1.1.1. Học máy và học sâu

Học máy là một nhánh con trực thuộc Trí Tuệ Nhân Tạo, đã sẵn sàng để thực hiện những nhảy vọt trong khoa học và công nghệ. Là một mô hình phát triển từ thế kỷ 20, nó đã trải qua một quá trình phát triển mạnh mẽ từ những mô hình đơn giản như cây quyết định, Naive Bayes đến những mô hình phức tạp như Mạng Nơ-ron và Học Sâu. Học máy đã mở ra những kỷ nguyên mới về việc xử lý dữ liệu và cách chúng ta tiếp cận với thông tin.

Học máy có thể được phân loại theo nhiều cách, nhưng ba loại chính là Học có giám sát, Học không giám sát, và Học tăng cường, mỗi loại đều mang lại những lợi ích riêng biệt và tạo ra những ứng dụng rộng rãi. Nó đã tìm thấy vị trí của mình trong nhiều lĩnh vực, từ các hệ thống đề xuất thông minh, dự đoán giá cả, phân loại văn bản, nhận dạng giọng nói và hình ảnh, đến sự can thiệp của y tế, bảo mật mạng, và đóng góp vào nghiên cứu khoa học.

Tuy nhiên, những thách thức cũng đồng hành cùng với những tiến bộ. Học máy vẫn còn đối diện với các vấn đề như sự phụ thuộc vào dữ liệu, việc hiểu rõ các mô hình, và những vấn đề đạo đức và pháp lý. Nhưng với sự phát triển không ngừng của Học Sâu, Học Chuyển Giao, Học Tự Động, và nhiều hướng nghiên cứu mới khác, tương lai của học máy đang mở rộng không ngừng. Chúng ta có thể tin tưởng rằng học máy sẽ tiếp tục cung cấp những cơ hội để phát triển công nghệ mới, thúc đẩy sự tiến bộ của xã hội và kinh tế.

Có nhiều mô hình khác nhau trong học máy, mỗi mô hình có ưu điểm, nhược điểm và lĩnh vực ứng dụng phù hợp khác nhau. Dưới đây là một số mô hình Học máy phổ biến:

- **Hồi quy tuyến tính (Linear Regression):** Đây là một trong những mô hình học máy đơn giản nhất. Mô hình này giả định có một mối quan hệ tuyến tính giữa các biến đầu vào và biến đầu ra. Nó được sử dụng rộng rãi trong các vấn đề dự đoán và phân tích thống kê.
- **Phân loại Logistic (Logistic Regression):** Mô hình này tương tự như hồi quy tuyến tính nhưng được sử dụng trong các vấn đề phân loại nhị phân.
- **Máy Véc-tơ Hỗ trợ (Support Vector Machines - SVM):** SVM là một mô hình phân loại mạnh mẽ được sử dụng trong các vấn đề phân loại nhị phân hoặc đa lớp.

- Cây quyết định (Decision Trees) và Rừng ngẫu nhiên (Random Forests): Cả hai mô hình này đều dựa trên cây quyết định. Rừng ngẫu nhiên là một kỹ thuật ensemble, nghĩa là nó kết hợp nhiều mô hình nhỏ (ở đây là cây quyết định) để tạo ra một mô hình mạnh mẽ.
- Mạng Nơ-ron (Neural Networks): Mạng Nơ-ron là cơ sở của Học Sâu (Deep Learning). Chúng có thể học từ dữ liệu phức tạp và tạo ra các mô hình có độ chính xác cao, đặc biệt trong các tác vụ như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, và nhiều tác vụ khác.
- K-Means: Đây là một thuật toán phân cụm không giám sát phổ biến. Nó phân loại dữ liệu vào các nhóm (hay cụm) dựa trên sự tương tự giữa các điểm dữ liệu.

Kỹ thuật Giảm chiều dữ liệu: Như PCA (Principal Component Analysis) hay t-SNE, các kỹ thuật này giúp giảm số lượng biến đầu vào (hay "đặc trưng") trong tập dữ liệu mà vẫn giữ được thông tin cần thiết.

Các mô hình Học máy này đều có khả năng học từ dữ liệu và dự đoán hoặc phân loại dữ liệu mới. Lựa chọn mô hình phù hợp phụ thuộc vào nhiều yếu tố như độ phức tạp của dữ liệu, số lượng dữ liệu, và nhu cầu cụ thể của bài toán.

Học Sâu (Deep Learning) là một nhánh của học máy, đại diện cho những phương pháp tiên tiến nhất và phức tạp nhất mà nói chung giả lập cách mà con người học và nhận thức về thế giới. Thông qua việc sử dụng các mạng nơ-ron nhiều tầng, học sâu có thể xử lý và mô hình hóa dữ liệu phức tạp ở mức độ rất cao.

Những mô hình học sâu được cấu trúc thành nhiều tầng, mỗi tầng được xem như một cấp độ của việc học. Từ những tính năng rất cơ bản ở những tầng đầu tiên, mô hình học và tạo ra những tính năng phức tạp hơn qua từng tầng. Điều này giúp cho học sâu đạt được những kết quả tuyệt vời trong nhiều lĩnh vực, như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, tự động lái xe và nhiều hơn nữa.

Có nhiều loại mô hình học sâu khác nhau, bao gồm Mạng Nơ-ron Tích Chập (Convolutional Neural Networks - CNNs) thường được sử dụng cho nhận dạng hình ảnh, Mạng Nơ-ron Hồi tiếp (Recurrent Neural Networks - RNNs) hay chuyên dùng cho dữ liệu chuỗi thời gian và xử lý ngôn ngữ tự nhiên, và Mạng Nơ-ron GAN (Generative Adversarial Networks) dùng để tạo ra dữ liệu giả mạo rất thuyết phục.

Tuy nhiên, học sâu cũng gặp phải một số thách thức. Việc huấn luyện các mô hình học sâu đòi hỏi rất nhiều dữ liệu và tài nguyên tính toán. Bên cạnh đó, việc hiểu rõ hóa và giải thích các mô hình học sâu cũng là một vấn đề, bởi vì chúng thường hoạt động như một "hộp đen".

Mặc dù vậy, học sâu vẫn đang tiếp tục đẩy mạnh giới hạn của những gì Học máy có thể đạt được, và ngày càng trở nên quan trọng hơn trong việc phát triển các ứng dụng Trí Tuệ Nhân Tạo tiên tiến.

1.1.2. Các mô hình học sâu

Mạng Nơ-ron Nhân tạo (Artificial Neural Networks - ANN): ANN là cốt lõi của học sâu. Chúng bao gồm nhiều lớp các nơ-ron, mỗi nơ-ron nhận đầu vào từ các nơ-ron của lớp trước và truyền đầu ra tới các nơ-ron của lớp tiếp theo. Các nơ-ron có thể học từ dữ liệu thông qua việc điều chỉnh trọng số của các liên kết giữa chúng trong quá trình đào tạo. Quá trình này được lặp đi lặp lại cho đến khi mô hình có thể dự đoán đầu ra chính xác từ đầu vào.

Mạng Nơ-ron Tích Chập (Convolutional Neural Networks - CNN): CNN là loại mạng nơ-ron phổ biến trong việc xử lý dữ liệu hình ảnh. CNN sử dụng các bộ lọc tích chập để quét qua hình ảnh và tìm hiểu các đặc trưng không gian. Điều này giúp cho CNN có khả năng phát hiện các đặc trưng như cạnh, góc, và các đặc trưng không gian khác. CNN đã đạt được nhiều thành công trong các tác vụ như phân loại hình ảnh, nhận diện khuôn mặt và nhận dạng đối tượng.

Mạng Nơ-ron Hồi Quy (Recurrent Neural Networks - RNN): RNN là mô hình học sâu được thiết kế để xử lý dữ liệu chuỗi, như văn bản hoặc chuỗi thời gian. Trái với ANN và CNN, RNN có khả năng "nhớ" thông tin từ các bước thời gian trước và sử dụng thông tin này để ảnh hưởng đến quyết định ở bước hiện tại. Một biến thể nổi bật của RNN là Mạng Nơ-ron Hồi Quy Dài ngắn hạn (Long Short-Term Memory - LSTM), giúp giải quyết vấn đề mất mát thông tin khi xử lý chuỗi dữ liệu dài.

Mạng Nơ-ron Cải Tiến (Generative Adversarial Networks - GANs): GANs là mô hình học sâu tạo ra dữ liệu mới giống như dữ liệu thực. GANs bao gồm hai mạng con: mạng sinh (generator) và mạng phân biệt (discriminator). Mạng sinh tạo ra dữ liệu giả, trong khi mạng phân biệt cố gắng phân biệt dữ liệu thực từ dữ liệu giả. Qua quá trình cạnh tranh này, cả hai mạng cùng nhau tạo ra dữ liệu giả ngày càng giống với dữ liệu thực.

Truyền thống học (Transfer Learning): Truyền thống học là một kỹ thuật quan trọng trong học sâu, giúp mô hình đã được đào tạo trên một tác vụ có thể được sử dụng lại cho một tác vụ khác. Điều này không chỉ tiết kiệm thời gian và nguồn lực đào tạo, mà còn tận dụng được kiến thức đã học được từ tác vụ gốc, cải thiện hiệu suất trên tác vụ mới.

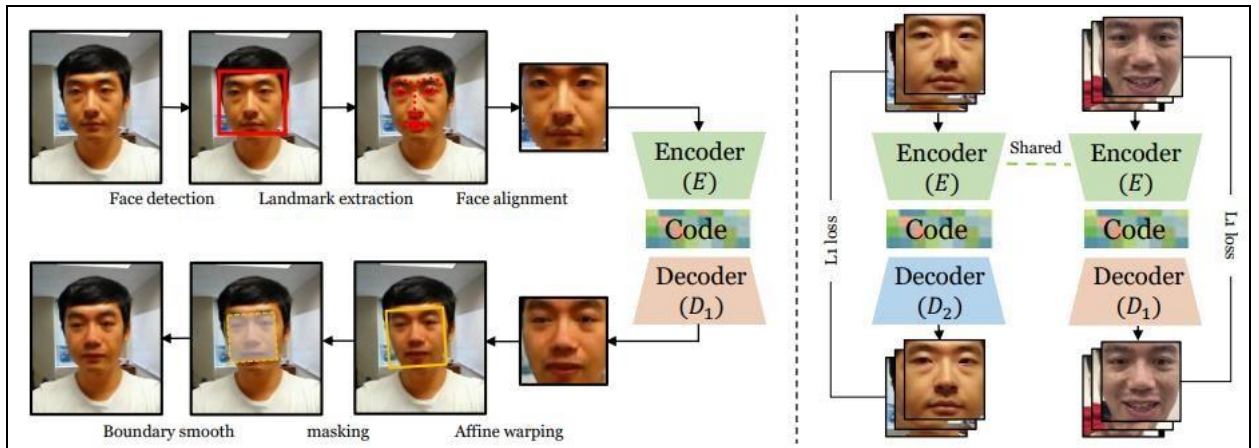
Học sâu và các mô hình học sâu đang mở ra nhiều hứa hẹn trong nhiều lĩnh vực khác nhau, từ nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, dự báo thời tiết, chẩn đoán y tế, và nhiều hơn nữa. Với sự phát triển không ngừng của công nghệ, học sâu sẽ tiếp tục đóng một vai trò quan trọng trong tương lai của trí tuệ nhân tạo.

1.2. Tổng quan về deepfakes

Deepfake là một kỹ thuật tổng hợp hình ảnh con người dựa trên trí tuệ nhân tạo. Khái niệm này được đặt theo tên của một người dùng ẩn danh trên mạng xã hội Reddit, người dùng có tên biệt danh là deepfakes - một từ ghép của “deep learning” và “fake”. Đây cũng chính là người chia sẻ video deepfake đầu tiên bằng cách giả mạo khuôn mặt của những người vô danh nổi tiếng vào các video có nội dung người lớn.

Các nội dung giả được tạo ra từ deepfake, có thể là hình ảnh, video, thậm chí là âm thanh mà rất khó phân biệt thật giả bằng tri giác con người theo cách thông thường.

Nguồn gốc của deepfake xuất phát từ một mạng nơ-ron có tên là Autoencoder. Thông thường, Autoencoder được hình thành bởi hai mạng nơ-ron tích chập (CNN) đó là Encoder và Decoder. Encoder chuyển đổi khuôn mặt từ dữ liệu đầu vào thành một vector mẫu. Vector mẫu ghi lại tất cả các thông tin như tư thế, biểu cảm, ánh sáng trong ảnh của một người, cho phép xác định danh tính của đối tượng. Để đảm bảo sự đồng bộ trong các vector mẫu, chỉ có duy nhất một Encoder được sử dụng xuyên suốt quá trình tạo deepfake. Mặt khác, mỗi danh tính có một bộ Decoder riêng, cho phép tạo ra khuôn mặt tương ứng của chủ thể từ vector mẫu.



Hình 1.1. Thuật toán tạo deepfakes

Ví dụ, khi thay thế khuôn mặt của Bob trong video bằng khuôn mặt của Alice. Sử dụng Encoder để trích xuất vector mẫu của Bob trong video và hiển thị nó bằng Decoder của Alice. Do đó, khuôn mặt Alice được tạo ra sẽ có cùng tư thế, ánh sáng và biểu cảm cảm xúc như khuôn mặt của Bob trong video gốc. Hình 1.1 minh họa thêm về thuật toán tạo deepfake với Autoencoder.

Khi deepfake trở nên phổ biến hơn, xã hội rất có thể sẽ cần phải thích ứng với việc phát hiện các video deepfake, giống như cách người dùng trực tuyến hiện đang chú ý đến việc phát hiện những loại tin giả khác. Có một số dấu hiệu để nhận dạng deepfake là:

- Khuôn mặt trong nhiều video deepfake mờ bất thường.
- Ở một số khuôn mặt được hoán đổi danh tính, màu da trông không tự nhiên.
- Lặp lại các bộ phận trên khuôn mặt: hai cằm, bốn lông mày...
- Sự không đồng nhất giữa các chi tiết ở vùng mặt và các bộ phận khác trên cơ thể. Ví dụ: mặt và cổ có màu da khác nhau.

1.2.1. Deepfake và các vấn đề có liên quan

Có rất nhiều vấn đề gây tác động tiêu cực từ deepfake. Những vấn đề này liên quan đến nhiều mặt của đời sống như: chính trị, kinh tế, an ninh quốc gia, nội dung khiêu dâm và xã hội học.

Chính trị là một vấn đề khá nhạy cảm không chỉ ở Việt Nam mà còn ở trên các quốc gia khác trên toàn thế giới. Với sự can thiệp ngày càng sâu của tội phạm mạng, nguy cơ tạo ra suy nghĩ lệch lạc và sự thiếu tin tưởng vào chính phủ, các nhà lãnh đạo sẽ tăng lên đáng kể. Theo The Washington Post nhiều video deepfake về Chủ tịch Hạ viện Mỹ - Nancy Pelosi đã được phát tán. Một video cho thấy, nạn

nhân nói lấp bấp và trong tình trạng say xỉn. Nhiều người, ngay cả cựu Tổng thống đương nhiệm Donald Trump, khi xem thông tin này đều tin rằng đây chính là Chủ tịch Hạ viện Pelosi. Theo hiệu ứng đám đông, video này liên tục được chia sẻ trên nhiều nền tảng mạng xã hội khiến lượt xem tăng nhiều lần và trở nên đáng tin cậy hơn. CBS News sau đó đã báo cáo rằng thực ra đó là một deepfake. Thực tế cho thấy, deepfake có thể ảnh hưởng lớn đến tương lai của một chính trị gia. Deepfake không chỉ là mối quan tâm của các chính trị gia về danh tiếng của họ, mà còn là mối quan tâm khi nói đến các cuộc tổng tuyển cử. Nếu một deepfake được nhắm vào một ứng cử viên chính trị với các hình ảnh, video phản cảm thì các cử tri hoàn toàn có thể bị dao động để chuyển phiếu bầu của họ sang một ứng cử viên đối lập hoặc không bỏ phiếu. Vì vậy, thông tin sai lệch có khả năng kiểm soát và phá vỡ nền chính trị trong nước theo nhiều cách.

Ngoài chính trị, deepfake cũng rất đáng lo ngại đối với an ninh quốc gia. Ví dụ, có rất nhiều video giả mạo nhắm đến nhà lãnh đạo tối cao của Triều Tiên - Kim Jong Un. Nội dung của các video này nói về các cuộc thực hiện phóng vũ khí hạt nhân chống lại Hoa Kỳ. Một ví dụ khác có thể là một video deepfake cho thấy một cuộc tấn công của Hoa Kỳ đang diễn ra nhằm vào các quốc gia khác. Mục tiêu nhằm kích động một cuộc phản công vô căn cứ. Những thông tin trên được sử dụng để gây ra sự hoảng loạn và tổn hại tới an ninh quốc gia.

Đôi khi, các tuyên bố công khai có thể tác động đến thị trường toàn cầu ngay lập tức. Ví dụ, thị trường toàn cầu có thể bị ảnh hưởng khi một nhân vật tài chính của chính phủ, các tỷ phú như Warren Buffet, Bill Gates hoặc người đứng đầu một quỹ đầu tư khổng lồ đưa ra nhận định về nền kinh tế. Ngay cả giá cổ phiếu của một công ty cũng có thể bị tác động (tích cực hoặc tiêu cực) bởi một tin ngắn.

“DeepTrace, một công ty sử dụng học sâu và thị giác máy tính để phát hiện deepfake, đã thực hiện một số thống kê với vấn đề deepfake. Thống kê cho biết, từ năm 2017 đến năm 2018, số lượng video deepfake đã tăng gần gấp đôi trong vòng 7 tháng. Một phát hiện quan trọng khác là 96% video deepfake miêu tả nội dung khiêu dâm không có sự đồng ý từ người mẫu.”

Nếu không thể phân biệt được deepfake, điều này đặt ra câu hỏi là liệu danh tính của một người có được bảo mật hay không. Hiện tại, các thiết bị di động sử dụng hệ thống sinh trắc học như dấu vân tay, nhận dạng khuôn mặt hoặc giọng nói. Nếu deepfake có thể vượt qua hoặc phá vỡ các hệ thống này, cho phép tin tặc truy cập vào điện thoại của người dùng hoặc tài khoản ngân hàng của họ, thì sẽ kéo theo rất nhiều hệ lụy khác.

Khi nhìn vào cơ hội mà công nghệ deepfakes mang lại, ta thấy rằng khả năng của nó không chỉ dừng lại ở việc tạo ra nội dung giả mạo. Trong ngành công nghiệp giải trí, deepfakes có thể được sử dụng để tái tạo hình ảnh hoặc giọng nói của những ngôi sao đã qua đời, hoặc tạo ra các cảnh quay hài hước, kịch tính mà không thể thực hiện được trong thực tế. Trong lĩnh vực công nghệ thông tin, deepfakes có thể được sử dụng để tạo ra trợ lý ảo chất lượng cao, trò chơi video sinh động hơn, hoặc các hình ảnh và video giả để huấn luyện mô hình AI khác.

Tuy nhiên, ngay cạnh những cơ hội tiềm năng đó, deepfakes cũng đặt ra những rủi ro và thách thức đáng kể. Thông tin giả mạo tạo ra bằng công nghệ deepfakes có thể được sử dụng để lừa dối, tấn công, hoặc phá hoại danh tiếng của người khác. Nó cũng có thể làm giảm lòng tin của người dùng vào thông tin trực tuyến, khi mọi người không thể tin tưởng vào những gì họ thấy hay nghe trên Internet.

Trước những thách thức này, nhiều nghiên cứu và phát triển đang được thực hiện để phát hiện và ngăn chặn deepfakes. Công nghệ phát hiện deepfakes, ví dụ như việc sử dụng học sâu để phân loại video giả và thực, hoặc việc xác định xem một video có bị chỉnh sửa không dựa trên các yếu tố như nhấp nháy mắt hoặc hơi thở, đang được nghiên cứu và cải thiện. Tuy nhiên, với mức độ tiên tiến của công nghệ deepfakes hiện nay, việc phân biệt giữa thực và giả vẫn là một thách thức lớn.

Vì vậy, với cả những cơ hội và thách thức mà deepfakes mang lại, cần một cách tiếp cận cân nhắc và thông minh với công nghệ này. Cần phải có cơ chế pháp lý và quy định vững chắc để giám sát và điều chỉnh việc sử dụng công nghệ này, trong khi cùng lúc tìm kiếm cách tận dụng những lợi ích mà nó mang lại.

1.2.2. Phương pháp tạo ra deepfakes

Để tạo ra một video hoặc hình ảnh deepfake, các bước cụ thể sau đây được sử dụng. Đây là một quy trình nâng cao yêu cầu kiến thức về lập trình và học máy.

Thu thập dữ liệu

Để tạo một deepfake, đầu tiên cần thu thập dữ liệu về đối tượng muốn tạo deepfake. Dữ liệu này thường dạng hình ảnh hoặc video. Cần có nhiều dữ liệu từ nhiều góc độ và điều kiện chiếu sáng khác nhau để mô hình có thể hiểu được hình dạng và diện mạo của người đó. Một số nguồn thông thường có thể bao gồm video từ YouTube, hình ảnh từ Google, hoặc các tập dữ liệu hình ảnh công cộng.

Tiền xử lý dữ liệu

Tiếp theo, cần xử lý dữ liệu để chuẩn bị cho việc huấn luyện mô hình. Bước này có thể bao gồm việc cắt, xoay, thay đổi kích cỡ, hoặc chuyển đổi định dạng của hình ảnh hoặc video. Điều quan trọng là cần xác định và cắt các khuôn mặt từ hình ảnh hoặc video. Có nhiều công cụ có sẵn để thực hiện việc này, bao gồm các thư viện như OpenCV.

Huấn luyện mô hình học sâu

Các deepfake thường được tạo ra bằng cách sử dụng một loại mô hình học sâu được gọi là mạng đối nghịch tạo sinh (GAN). Mô hình GAN bao gồm hai phần: một mô hình sinh (Generator) dùng để tạo ra hình ảnh giả, và một mô hình phân biệt (Discriminator) cố gắng phân biệt giữa hình ảnh thật và hình ảnh giả.

Trong quá trình huấn luyện, mô hình sinh và mô hình phân biệt cạnh tranh với nhau trong một trò chơi đối nghịch. Mô hình sinh cố gắng tạo ra hình ảnh giả mà mô hình phân biệt không thể phân biệt, trong khi mô hình phân biệt cố gắng cải thiện khả năng phân biệt của mình. Kết quả là cả hai mô hình đều trở nên tốt hơn theo thời gian.

Tạo hình ảnh deepfake

Khi mô hình đã được huấn luyện xong, có thể sử dụng mô hình sinh để tạo ra hình ảnh deepfake. Mô hình sẽ "vẽ" khuôn mặt của người mục tiêu dựa trên hình ảnh hoặc video đầu vào. Điều này được thực hiện bằng cách truyền hình ảnh hoặc video qua mô hình và sau đó lấy kết quả đầu ra.

Tinh chỉnh (post-processing)

Cuối cùng, sau khi tạo ra hình ảnh deepfake, có thể cần tinh chỉnh nó thêm. Điều này có thể bao gồm việc chỉnh sửa màu sắc, sáng tối, hoặc loại bỏ bất kỳ khuyết điểm nào xuất hiện trong hình ảnh. Mục tiêu ở đây là làm cho hình ảnh deepfake trông tự nhiên và thực sự như người thật.

1.3. Tổng quan về tấn công đối kháng

1.3.1. Khái niệm tấn công đối kháng




Cũng giống như rất nhiều hệ thống thông tin khác, khi có các phương pháp xây dựng mô hình tốt, nhanh, chính xác thì cũng sẽ có các phương pháp nhằm phá hoại, tấn công, đánh lừa mô hình. Đây là hai thái cực tồn tại song song và luôn mang tính thời sự. Có nhiều kiểu tấn công mạng nơ-ron tùy thuộc vào mục tiêu cũng như các giả thiết về sự hiểu biết (kiến thức) của kẻ tấn công đối với mô hình học máy. Nhìn chung thì đa phần những kẻ tấn công đều muốn bằng một cách nào đó tác động vào mô hình hoặc dữ liệu đầu vào để khiến cho mô hình đưa ra những

phán đoán sai lầm. Dựa trên hiểu biết của kẻ tấn công về mạng có thể chia thành hai cách tấn công đó là **White-box** (tấn công hộp trắng) và **Black-box** (tấn công hộp đen) theo cách hiểu như sau:

- **White box attacker:** là một kẻ tấn công có hiểu biết đầy đủ về đầu vào, đầu ra, kiến trúc của mạng, thứ tự sắp xếp các layer (lớp), các activation (tính năng) sử dụng, các trọng số đã được huấn luyện cũng như có toàn quyền truy cập, thay đổi các thông số đó. Kẻ tấn công dạng này có thể chiếm trọn quyền sử dụng và khiến mô hình hoạt động theo ý muốn của hắn.

- **Black box attacker:** là một kẻ tấn công chỉ biết được thông tin dựa trên đầu vào và đầu ra của mạng mà không hề biết gì về kiến trúc bên trong của mô hình cũng như các trọng số đã được huấn luyện. Các hướng tấn công sẽ tập trung vào việc thay đổi dữ liệu đầu vào nhằm mục đích đánh lừa mô hình AI. Mục đích của những kẻ tấn công này cũng chia thành hai loại chính là phân tấn công có không mục tiêu và tấn công có mục tiêu. Đối với dạng không có mục tiêu thì mục đích của kẻ tấn công chỉ nhằm khiến cho mô hình nhận dạng sai đi mà không cần quan tâm đến kết quả đầu ra. Ví dụ đưa ảnh con mèo vào thì mô hình không nhận ra đây là con mèo nữa tức là đã tấn công thành công. Còn đối với tấn công có mục tiêu thì lại nâng tầm tấn công lên một level mới. Ví dụ đưa ảnh con mèo vào bắt buộc đầu ra của mô hình sẽ nhận dạng thành con chó hoặc lò nướng bánh.

Ví dụ về phân loại gấu trúc để thấy rõ một số khái niệm, thuật toán cụ thể:

	$+ .007 \times$		$=$	
x		$\text{sign}(\nabla_x J(\theta, x, y))$		$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$
“panda”		“nematode”		“gibbon”
57.7% confidence		8.2% confidence		99.3 % confidence

Hình 1.2. Thực nghiệm tạo mẫu đối kháng thực hiện bởi Christian Szegedy, 2014

Trong đó x là ảnh input, y là nhãn của ảnh x . $J(\theta, x, y)$ là hàm loss sử dụng để training model θ . Quá trình huấn luyện sử dụng lan truyền ngược để tính toán đạo hàm và giải thuật sẽ căn cứ vào các giá trị đạo hàm để thay đổi input x đồng

thời tính toán sự thay đổi của hàm loss $\nabla_x J(\theta, \mathbf{x}, y)$. Khi thay đổi input x một lượng epsilon ϵ rất nhỏ theo hướng maximize loss $\text{sign}(\nabla_x J(\theta, \mathbf{x}, y))$ thì giá trị x sẽ bị phân loại sai thành classs **gibbon**.

1.3.2. Phân loại tấn công đối kháng

Các mô hình học máy hoàn thiện thường bị tấn công bằng cách nhận dữ liệu đầu vào là các mẫu đối kháng. Một mẫu đầu vào là “sạch” nếu nó là mẫu thô, chẳng hạn như một bức ảnh được lấy từ tập dữ liệu ImageNet. Nếu đối phương cố ý sửa đổi một mẫu để mô hình học máy phân loại sai thì mẫu đó được gọi là “mẫu đối kháng”. Dựa trên các khía cạnh khác nhau, chia tấn công đối kháng thành 3 nhóm dưới đây:

Trước hết, các cuộc tấn công có thể được phân loại theo loại kết quả mà đối thủ mong muốn:

- **Tấn công không mục tiêu:** Trong trường hợp này, mục tiêu của mẫu đối kháng là khiến bộ phân loại dự đoán không chính xác đối với bất kỳ đối tượng nào mà không quan trọng đối tượng cụ thể đó là gì.
- **Tấn công có mục tiêu:** Trong trường hợp này, kẻ thù muốn thay đổi dự đoán của bộ phân loại thành một số loại mục tiêu cụ thể.

Thứ hai, các kịch bản tấn công có thể được phân loại theo lượng kiến thức mà đối thủ có về mô hình:

- **Tấn công hộp trắng:** Trong tấn công hộp trắng, đối thủ có đầy đủ kiến thức về mô hình bao gồm loại mô hình, kiến trúc mô hình và các giá trị của tất cả các tham số và trọng số huấn luyện.
- **Hộp đen có đầu dò:** Trong trường hợp này, đối thủ không biết nhiều về mô hình, nhưng có thể thăm dò hoặc truy vấn mô hình, tức là cung cấp một số đầu vào và quan sát đầu ra. Có nhiều biến thể của kịch bản này - đối thủ có thể biết kiến trúc nhưng không biết các thông số hoặc đối thủ thậm chí có thể không biết kiến trúc mà thực hiện quan sát xác suất đầu ra cho từng lớp, hay đối thủ chỉ lựa chọn quan sát các lớp có khả năng nhất.
- **Hộp đen không thăm dò:** Trong kịch bản hộp đen không thăm dò, đối thủ có kiến thức hạn chế hoặc không có kiến thức về mô hình sẽ tấn công và không được phép thăm dò hoặc truy vấn mô hình trong khi xây dựng

các mẫu đối kháng. Trong trường hợp này, kẻ tấn công phải xây dựng các mẫu đối kháng để đánh lừa hầu hết các mô hình máy học.

Thứ ba, các cuộc tấn công có thể được phân loại bằng cách cung cấp dữ liệu vào mô hình máy học:

- **Tấn công kỹ thuật số:** Trong trường hợp này, đối thủ có quyền truy cập trực tiếp vào dữ liệu thực tế được đưa vào mô hình. Nói cách khác, đối thủ có thể chọn các giá trị như float32 cụ thể làm đầu vào cho mô hình. Trong cài đặt thực tế, điều này có thể xảy ra khi kẻ tấn công tải tệp PNG lên dịch vụ web và cố tình thiết kế tệp để khiến cho tệp đó được đọc không chính xác.

- **Tấn công vật lý:** Trong trường hợp này kẻ tấn công không có quyền truy cập trực tiếp vào biểu diễn kỹ thuật số được cung cấp cho mô hình. Thay vào đó, mô hình được cung cấp đầu vào thu được bởi các cảm biến như máy ảnh hoặc micrô. Kẻ thù có thể đặt các đối tượng trong môi trường vật lý mà máy ảnh nhìn thấy hoặc tạo ra âm thanh mà micrô nghe thấy. Biểu diễn kỹ thuật số chính xác mà cảm biến thu được sẽ thay đổi dựa trên các yếu tố như góc máy ảnh, khoảng cách đến micrô, ánh sáng xung quanh hoặc âm thanh trong môi trường, v.v. Điều này có nghĩa là kẻ tấn công có quyền kiểm soát kém chính xác hơn đối với đầu vào được cung cấp cho máy học.

1.3.3. Một số phương pháp tấn công theo sự hiểu biết về mô hình

Quy ước ký hiệu:

adv_x : Adversarial image.

x : Original input image.

y : Original input label.

ϵ : Multiplier to ensure the perturbations are small.

θ : Model parameters.

J : Loss.

- **Tấn công hộp trắng (White box digital attacks)**

L-BFGS: Đây là một trong những phương pháp đầu tiên để tìm ra các mẫu đối kháng cho mạng nơron. Ý tưởng của phương pháp này là giải quyết vấn đề tối ưu hóa sau:

$$\|x^{adv} - x\|_2 \rightarrow \min, \text{ sao cho } f(x^{adv}) = y_{\text{target}}, x^{adv} \in [0,1]^m$$

Các tác giả đề xuất sử dụng phương pháp tối ưu hóa L-BFGS để giải quyết vấn đề này. Một trong những nhược điểm chính của phương pháp này là tốc độ thực thi khá chậm. Phương pháp này không được thiết kế để chống lại sự phòng thủ chẳng hạn như giảm số lượng bit được sử dụng để lưu trữ mỗi điểm ảnh (pixel). Thay vào đó, phương pháp này được thiết kế để tìm ra sự nhiễu loạn nhỏ nhất dùng để tấn công, có nghĩa là phương pháp này đôi khi có thể thất bại chỉ bằng cách làm giảm chất lượng hình ảnh, ví dụ bằng cách làm tròn 8 bit của mỗi pixel.

Phương pháp dấu gradient nhanh (FGSM): Để kiểm tra ý tưởng rằng các mẫu đối kháng có thể được tìm thấy chỉ bằng cách sử dụng một phép gần đúng tuyến tính của mô hình mục tiêu. FGSM hoạt động bằng cách tuyến tính hóa hàm mất mát (hàm mục tiêu, hay hàm chi phí) trong vùng lân cận chuẩn L_∞ của một hình ảnh “sạch” và tìm chính xác cực đại của hàm tuyến tính bằng cách sử dụng phương trình dạng sau:

$$x^{adv} = x + \epsilon \text{sign}(\nabla_x J(x, y_{\text{true}}))$$

Các cuộc tấn công lặp lại: Tấn công L-BFGS có tỷ lệ thành công và chi phí tính toán cao. Tấn công FGSM có tỷ lệ thành công thấp (đặc biệt là khi người phòng thủ) cũng như chi phí tính toán thấp. Sự cân bằng tốt có được khi lặp đi lặp lại các thuật toán tối ưu hóa chuyên biệt để đạt được mục tiêu, sau một số lượng nhỏ (ví dụ: 40) lần lặp.

Một chiến lược nhằm tối ưu hóa thời gian là sử dụng FGSM (thường có thể đạt được giải pháp chấp nhận được qua một bước rất lớn) và thực thi qua một số bước nhưng với kích thước bước nhỏ hơn. Vì mỗi bước của FGSM được thiết kế để tiệm cận quanh giá trị bắt đầu của bước đó, nên phương pháp này giúp hội tụ nhanh chóng ngay cả với các gradient nhỏ. Từ đó, Kurakin và các cộng sự đã giới thiệu phương pháp lặp cơ bản (BIM) (còn được gọi là Phương pháp lặp FGSM (I-FGSM):

$$x_0^{adv} = X, x_{N+1}^{adv} = \text{Clip}_{X,\epsilon} \left\{ X_N^{adv} + \alpha \text{sign}(\nabla_X J(X_N^{adv}, y_{\text{true}})) \right\}$$

BIM có thể dễ dàng thực hiện thành công một cuộc tấn công có mục tiêu, được gọi là phương thức lớp mục tiêu lặp:

$$X_0^{adv} = X, X_{N+1}^{adv} = \text{Clip}_{X,\epsilon} \left\{ X_N^{adv} - \alpha \text{sign}(\nabla_X J(X_N^{adv}, y_{\text{target}})) \right\}$$

Các quan sát cho thấy, với đủ số lần lặp lại thì cách tấn công này hầu như luôn thành công trong việc đánh trúng lớp mục tiêu. Hơn nữa, Madry và các cộng sự đã chỉ ra rằng BIM có thể được cải thiện đáng kể bằng cách khởi tạo ngẫu nhiên từ một điểm trong lân cận ϵ . Cuộc tấn công này thường được gọi là **projected gradient descent**, nhưng tên gọi này có thể gây nhầm lẫn bởi vì: *thứ nhất*, thuật ngữ “projected gradient descent” đã đề cập đến một phương pháp tối ưu hóa tổng quát hơn so với việc sử dụng phương án cụ thể cho tấn công đối kháng; *thứ hai*, các cuộc tấn công khác sử dụng gradient và phép chiếu theo cùng một cách (cuộc tấn công giống như BIM ngoại trừ điểm khởi tạo) vì vậy tên gọi không phân biệt được cuộc tấn công này với các cuộc tấn công khác.

Mạng chuyển đổi đối nghịch (ATN): Một cách tiếp cận khác đã được đề xuất là huấn luyện một mô hình chung để tạo ra các mẫu đối kháng. Mô hình này lấy một hình ảnh “sạch” làm đầu vào và tạo ra một hình ảnh đối kháng tương ứng. Ưu điểm của cách tiếp cận này là, nếu bản thân mô hình sinh dữ liệu được thiết kế nhỏ, ATN có thể tạo ra các mẫu đối kháng nhanh hơn so với một thuật toán tối ưu tường minh. Về lý thuyết, cách tiếp cận này có thể nhanh hơn cả FGSM, nếu ATN được thiết kế để sử dụng ít tính toán hơn là cần thiết để chạy lan truyền ngược trên mô hình mục tiêu. (Tất nhiên ATN yêu cầu thêm thời gian để huấn luyện, nhưng khi chi phí này đã được trả thì số lượng rất lớn các mẫu sẽ có thể được tạo ra với chi phí thấp).

Tấn công vào các hệ thống không phân biệt được: Tất cả các cuộc tấn công được đề cập ở trên cần tính toán gradient của mô hình bị tấn công để tạo ra các mẫu đối kháng. Tuy nhiên, điều này không phải lúc nào cũng thực hiện được, chẳng hạn trong trường hợp mô hình chứa các phép toán không thể phân biệt. Khi đó, đối thủ có thể huấn luyện một mô hình thay thế và sử dụng khả năng xâm nhập của các mẫu đối kháng để thực hiện một cuộc tấn công vào hệ thống không phân biệt, tương tự như các cuộc tấn công hộp đen

Tấn công bằng cách áp dụng thuật toán C&W: Thuật toán C&W (Carlini and Wagner) là một trong những thuật toán tấn công được phát triển để thử nghiệm tính an toàn của các hệ thống phân loại máy học. Thuật toán này được tạo ra bởi Nicholas Carlini và David Wagner và được công bố trong bài báo có tựa đề "Towards Evaluating the Robustness of Neural Networks" vào năm 2017.

Mục tiêu của thuật toán C&W là tìm ra một vector chấm dứt (perturbation) nhỏ nhất có thể được thêm vào đầu vào mà vẫn giữ cho hệ thống phân loại nhận dạng sai lớp của dữ liệu, nhưng đối với con người, sự thay đổi này không thể phát

hiện được. Thật vậy, khi một hệ thống phân loại bị tấn công, nó có thể đưa ra dự đoán không chính xác cho các dữ liệu nhỏ sai số so với dữ liệu gốc.

Từng bước chính trong thuật toán C&W bao gồm:

Thiết lập hàm mục tiêu (objective function) có thể tối ưu hóa để tạo ra một chấm dứt nhỏ nhất và đảm bảo rằng hệ thống phân loại đưa ra dự đoán sai lớp cho dữ liệu tấn công.

Sử dụng thuật toán tối ưu hóa, chẳng hạn như tối ưu hóa gradient hoặc tối ưu hóa tiến hóa, để tìm ra chấm dứt tối ưu.

Điều chỉnh các ràng buộc (constraints) để đảm bảo rằng chấm dứt được tìm ra nằm trong một khoảng giá trị chấp nhận được để không làm thay đổi quá nhiều dữ liệu gốc.

Thuật toán C&W được sử dụng để đánh giá tính bảo mật của các hệ thống phân loại máy học và giúp cải thiện chúng bằng cách tăng cường độ tin cậy và độ ổn định trong môi trường có thể tấn công.

Tấn công PGD: PGD (Projected Gradient Descent) là một phương pháp tấn công mạnh mẽ và phổ biến và thường được sử dụng để đánh giá tính ổn định và độ tin cậy của các mô hình phân loại.

Ý tưởng chính của thuật toán PGD là tạo ra một chấm dứt (perturbation) nhỏ nhất cho dữ liệu đầu vào mà làm thay đổi dự đoán của mô hình mục tiêu. Chấm dứt này được tìm thấy bằng cách tối ưu hóa hàm mục tiêu sao cho dữ liệu bị tấn công vẫn nằm trong một tập hợp ràng buộc xác định, thường là một khoảng giá trị cụ thể. Thuật toán sử dụng phương pháp gradient descent để tối ưu hóa hàm mục tiêu này.

Các bước chính trong thuật toán PGD như sau:

Chọn ngẫu nhiên một điểm xuất phát gần điểm dữ liệu gốc để tạo mặt nạ ban đầu.

Lặp lại quá trình tối ưu hóa trong một số vòng lặp nhất định:

- a. Tính gradient của hàm mục tiêu tại mặt nạ hiện tại.
- b. Cộng dồn gradient vừa tính được vào mặt nạ hiện tại để tạo ra mặt nạ mới.
- c. Giới hạn mặt nạ mới vào tập hợp ràng buộc đã định nghĩa.

Trả về mặt nạ cuối cùng sau khi kết thúc số vòng lặp đã định trước.

Phương pháp PGD cho phép tạo ra những mặt nạ mạnh mẽ và khó phát hiện, đánh lừa các mô hình phân loại và làm thay đổi dự đoán của chúng. Điều này giúp đánh giá tính ổn định và độ tin cậy của các mô hình học máy và làm cơ sở để cải thiện tính bảo mật của chúng. Tuy nhiên, cũng cần lưu ý rằng việc sử dụng thuật toán PGD để thử nghiệm tính bảo mật có thể tốn nhiều tài nguyên tính toán, đặc biệt đối với các mô hình lớn và phức tạp.

Tấn công AdvGAN: Phương pháp AdvGAN (Adversarial Generative Adversarial Networks) là một phương pháp tấn công đối kháng dựa trên mạng sinh đối kháng và mạng phân loại mẫu đối kháng (GANs). Nó được đề xuất bởi tổ chức nghiên cứu Horizon Robotics vào năm 2018.

Ý tưởng chính của AdvGAN là sử dụng mạng sinh (generator) để tạo ra các mạng nhiễu (perturbations) nhằm tấn công mô hình phân loại (classifier). Mạng sinh trong AdvGAN được huấn luyện để tạo ra các chấm dứt có thể được thêm vào dữ liệu đầu vào sao cho mô hình phân loại đưa ra dự đoán sai lầm cho các dữ liệu này, nhưng dữ liệu bị tấn công vẫn giữ nguyên tính bất biến đối với con người.

Cấu trúc chính của AdvGAN bao gồm hai thành phần:

Mạng sinh (Generator): Mạng này nhận vào một điểm dữ liệu đầu vào và tạo ra một chấm dứt nhỏ được thêm vào điểm dữ liệu này. Mục tiêu của mạng sinh là tối ưu hóa sao cho mô hình phân loại đưa ra dự đoán sai khi nhìn vào dữ liệu bị tấn công.

Mạng phân loại (Classifier): Đây là một mô hình phân loại máy học thông thường, ví dụ như mạng nơ-ron tích chập, được sử dụng để đánh giá tính chính xác của dữ liệu.

Cách hoạt động của AdvGAN:

Đầu tiên, mạng sinh được huấn luyện bằng cách tối ưu hóa hàm mục tiêu để tạo ra chấm dứt tối ưu. Mục tiêu là làm thay đổi đầu vào ban đầu sao cho mô hình phân loại đưa ra dự đoán sai lầm.

Sau khi mạng sinh được huấn luyện, chúng ta có thể sử dụng nó để tạo ra các chấm dứt và thêm chúng vào dữ liệu đầu vào.

Dữ liệu bị tấn công sau khi được thêm chấm dứt sẽ được truyền qua mô hình phân loại, và mô hình này sẽ đưa ra dự đoán sai lầm do sự ảnh hưởng của chấm dứt.

Mục tiêu cuối cùng của AdvGAN là tạo ra các chấm dứt nhỏ nhất có thể sao cho dữ liệu bị tấn công vẫn giữ nguyên tính bất biến đối với con người, tức là con người vẫn có thể nhận diện và phân loại đúng dữ liệu bị tấn công.

AdvGAN đã được chứng minh là một phương pháp tấn công hiệu quả trong việc đánh giá tính bảo mật của các mô hình phân loại máy học, và nó đã đóng góp vào sự phát triển của lĩnh vực nghiên cứu bảo mật học máy.

Tấn công AI-GAN: Tấn công AI-GAN (Adversarial AI-Generated Adversarial Network) là một dạng tấn công đối kháng sử dụng GAN (Generative Adversarial Network) để tạo ra chấm dứt nhằm đánh lừa mô hình AI. Phương pháp này kết hợp sự cải tiến của tấn công GAN và mô hình AI để tạo ra các chấm dứt mạnh mẽ hơn và khó phát hiện hơn.

Ý tưởng chính của tấn công AI-GAN là sử dụng mạng sinh (generator) của GAN để tạo ra các chấm dứt có tính năng đối kháng cao đối với mô hình AI mục tiêu. Mạng sinh sẽ được huấn luyện để tạo ra những chấm dứt có thể thay đổi dữ liệu đầu vào một cách tinh vi, dẫn đến sự lệch lạc trong kết quả phân loại của mô hình AI.

Các bước chính trong tấn công AI-GAN như sau:

Xây dựng GAN: Bắt đầu bằng việc xây dựng một cặp GAN với mạng sinh và mạng phân loại. Mạng sinh nhận vào một vector ngẫu nhiên làm đầu vào và tạo ra các chấm dứt. Mục tiêu của mạng sinh là tạo ra các chấm dứt sao cho mô hình phân loại đưa ra dự đoán sai lầm cho các dữ liệu bị tấn công.

Huấn luyện GAN: Tiến hành huấn luyện cặp GAN để tối ưu hóa mạng sinh sao cho chấm dứt tạo ra có hiệu quả trong việc đánh lừa mô hình phân loại.

Tạo mặt nạ nhiễu: Sau khi mạng sinh được huấn luyện, sử dụng nó để tạo ra các chấm dứt và thêm chúng vào dữ liệu đầu vào.

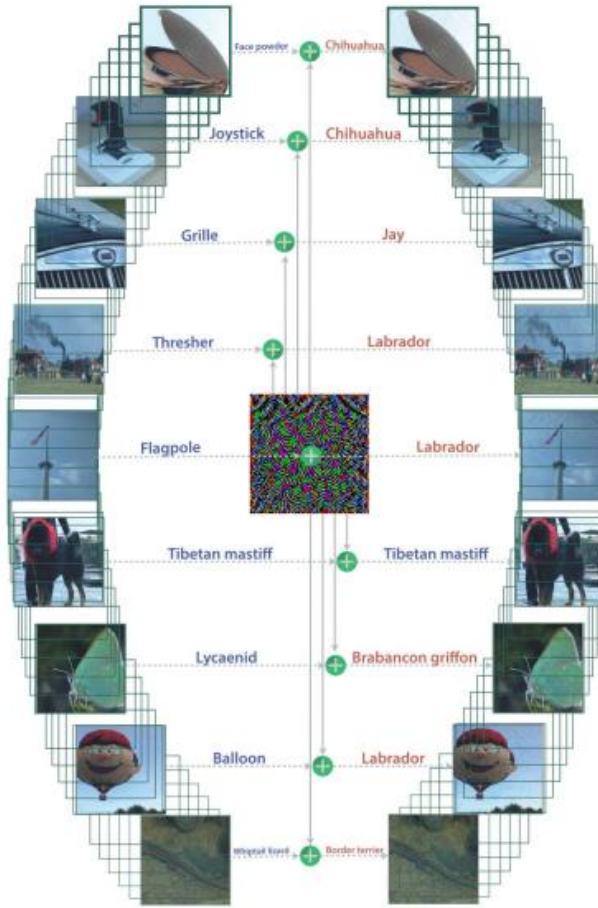
Đánh lừa mô hình AI: Dữ liệu bị tấn công sau khi được thêm chấm dứt sẽ được truyền qua mô hình AI mục tiêu, và mô hình này sẽ đưa ra dự đoán sai lầm do sự ảnh hưởng của chấm dứt.

Tấn công AI-GAN là một trong những phương pháp tấn công đối kháng nâng cao độ khó phát hiện của chấm dứt và làm đảo lộn tính bất biến của các mô hình AI. Nghiên cứu về tấn công và phòng thủ đối kháng ngày càng được quan tâm, để xây dựng các hệ thống AI bảo mật và đáng tin cậy hơn.

- **Tấn công hộp đen (Black box digital attacks)**

Các mẫu đối kháng được sinh ra giữa các mô hình khác nhau. Nói cách khác, một phần đáng kể các mẫu đối kháng khi đã đánh lừa được một mô hình thì cũng lừa được một mô hình khác. Thuộc tính này được gọi là “khả năng xâm nhập” và được sử dụng để tạo ra các mẫu đối kháng trong kịch bản hộp đen. Số lượng các mẫu đối kháng có thể xâm nhập thực tế có thể thay đổi từ vài phần trăm đến gần 100% tùy thuộc vào mô hình nguồn, mô hình mục tiêu, tập dữ liệu và các yếu tố khác. Những kẻ tấn công bằng kịch bản hộp đen có thể huấn luyện mô hình của riêng họ trên cùng một tập dữ liệu với mô hình mục tiêu, hoặc thậm chí huấn luyện mô hình của họ trên một tập dữ liệu khác được rút ra từ cùng một phân lớp. Khi đó, các mẫu đối kháng cho mô hình của đối thủ có cơ hội tốt để đánh lừa một mô hình mục tiêu không xác định.

Nếu kẻ tấn công không có tình huống hộp đen hoàn toàn mà được phép sử dụng các thiết bị thăm dò, để huấn luyện bản sao của chính kẻ tấn công của mô hình mục tiêu được gọi là “kẻ thay thế”. Cách tiếp cận này rất hiệu quả vì các mẫu đầu vào được gửi dưới dạng thăm dò không cần phải là các mẫu để huấn luyện thực tế; thay vào đó, chúng có thể là các giá trị đầu vào do kẻ tấn công chọn để tìm ra chính xác giới hạn quyết định của mô hình mục tiêu nằm ở đâu. Do đó, mô hình của kẻ tấn công được huấn luyện không chỉ trở thành một bộ phân loại tốt mà còn thực sự thiết kế ngược các chi tiết của mô hình mục tiêu, do đó, hai mô hình được định hướng một cách có hệ thống để có số lần xâm nhập cao.



Hình 1.3. Một ví dụ về đối kháng phổ biến đánh lừa mạng nơ-ron trên hình ảnh.

Hình ảnh bên trái: hình ảnh tự nhiên được dán nhãn ban đầu; hình ảnh trung tâm: nhiễu loạn phổ quát; hình ảnh bên phải: hình ảnh bị nhiễu với nhãn sai.

Trong kịch bản hộp đen hoàn chỉnh mà kẻ tấn công không thể gửi các đầu dò, một chiến lược để tăng tốc độ truyền là sử dụng một tập hợp nhiều mô hình làm mô hình nguồn cho các mẫu đối kháng. Ý tưởng cơ bản là nếu một mẫu đối kháng đánh lừa mọi mô hình trong tập hợp, thì nhiều khả năng nó sẽ tổng quát hóa và đánh lừa các mô hình bổ sung.

Cuối cùng, trong kịch bản hộp đen với các đầu dò, có thể chỉ chạy các thuật toán tối ưu hóa mà không sử dụng gradient để tấn công trực tiếp mô hình mục tiêu. Nhìn chung, thời gian cần thiết để tạo ra một mẫu đối kháng cao hơn nhiều so với khi sử dụng một kẻ thay thế, nhưng nếu chỉ cần một số lượng nhỏ các mẫu đối kháng được yêu cầu, thì các phương pháp này có thể có lợi thế hơn vì chúng có chi phí cố định ban đầu không cao khi huấn luyện thay thế.

1.3.4. Tổng quan về chống lại các cuộc tấn công đối kháng

Một số nghiên cứu có đề cập việc huấn luyện mẫu đối kháng là cách được thực hiện rộng rãi nhất để tăng độ bảo mật của các DNN. Bằng cách thêm mẫu đối kháng vào quy trình huấn luyện mô hình, các mô hình được huấn luyện mẫu đối kháng học cách chống lại những mẫu nhiễu ảnh hưởng đến sự chính xác của hàm. Tuy nhiên, quá trình này không mang lại sự mạnh mẽ cho các cuộc tấn công hộp đen do sự kết hợp của việc tạo ra các mẫu đối kháng và các tham số đã được huấn luyện. Huấn luyện đối kháng theo nhóm tăng cường dữ liệu huấn luyện với mẫu đối kháng được tạo ra không chỉ từ mô hình đang được huấn luyện, mà còn từ các mô hình học máy khác. Do đó, các mô hình đối thủ được huấn luyện theo tập hợp sẽ chống lại tốt các cuộc tấn công một bước và các cuộc tấn công hộp đen.

Vì nhiều đối kháng được tạo ra bởi nhiều phương pháp giống như nhiễu tần số cao đối với người quan sát nên nhóm tác giả đã đề xuất sử dụng tiền xử lý và khử nhiễu hình ảnh như một biện pháp bảo vệ hiệu quả để chống lại các mẫu đối kháng. Một sự khác biệt lớn trong các kỹ thuật tiền xử lý được nhóm tác giả đề xuất, như thực hiện nén JPEG hoặc áp dụng bộ lọc trung vị và giảm độ chính xác của dữ liệu được đưa vào huấn luyện. Các biện pháp phòng thủ như vậy có khả năng hoạt động tốt trước một số cuộc tấn công nhất định, các biện pháp phòng thủ đã được chứng minh là không thành công trong trường hợp hộp trắng do kẻ tấn công nhận thức được việc phòng thủ. Trong trường hợp hộp đen, biện pháp phòng thủ này có thể có hiệu quả trong thực tế.

Nhiều biện pháp bảo vệ do cố ý hoặc vô ý được gọi là "dấu gradient", hầu hết các cuộc tấn công hộp trắng hoạt động bằng cách tính toán nhiễu ảnh của mô hình và do đó thất bại nếu không thể tính toán các nhiễu hữu ích. Dấu Gradient có thể làm cho nhiễu trở nên vô dụng, bằng cách thay đổi mô hình khiến nó không thể phân biệt được hoặc làm cho nó không có nhiễu ở hầu hết các vị trí hoặc làm cho gradient hướng ra khỏi giới hạn quyết định. Dấu Gradient có thể làm giảm quá trình tối ưu hóa bởi sự giao thoa quyết định nhận biết đối tượng thuộc lớp ít nhiễu đều giống nhau, việc bảo vệ hệ thống dựa trên dấu gradient rất dễ bị chuyển đổi phương tấn công hộp đen. Nhiều biện pháp phòng thủ dựa trên việc phát hiện các mẫu đối kháng và từ chối nhận biết mẫu vào nếu có dấu hiệu giả mạo. Một cách khác, kẻ tấn công có thể tạo ra một cuộc tấn công đồng thời đánh lừa bộ phát hiện nghĩ rằng mẫu đối kháng là đầu vào hợp pháp và đánh lừa bộ phân loại đưa ra phân loại sai.

Một số biện pháp phòng thủ hoạt động là làm giảm độ chính xác của các mẫu sạch. Ví dụ, mạng RBF rất mạnh mẽ đối với các mẫu đối kháng trên tập dữ liệu nhỏ như MNIST nhưng có độ chính xác kém hơn nhiều trên MNIST sạch so với DNN. Mạng RBF đã cho thấy mức độ mạnh mẽ đối với các cuộc tấn công hộp trắng trên tập dữ liệu SmallNORB, nhưng vẫn chưa được đánh giá trên các tập dữ liệu khác vốn thường được sử dụng hơn trong các dữ liệu về mẫu đối kháng.

Cách phòng thủ phổ biến nhất trong các tài liệu nghiên cứu hiện nay có lẽ là huấn luyện đối kháng. Ý tưởng là đưa các mẫu đối kháng vào quá trình huấn luyện và huấn luyện mô hình dựa trên các mẫu đối kháng hoặc kết hợp các mẫu rõ ràng và mẫu đối kháng. Phương pháp này đã được áp dụng thành công cho các tập dữ liệu lớn, và có thể được thực hiện hiệu quả hơn bằng cách sử dụng các biểu diễn mã vectơ rời rạc thay vì biểu diễn số thực của đầu vào. Một nhược điểm chính của việc huấn luyện đối kháng là nó có xu hướng quá thích hợp với đòn tấn công cụ thể được sử dụng tại thời điểm huấn luyện. Điều này đã được khắc phục, ít nhất là trên các tập dữ liệu nhỏ, bằng cách thêm nhiễu trước khi khởi động trình tối ưu hóa cho cuộc tấn công. Một nhược điểm quan trọng khác của huấn luyện đối kháng là nó có xu hướng vô tình học cách làm mất nạt gradient hơn là thực sự di chuyển giới hạn quyết định. Điều này có thể được khắc phục phần lớn bằng cách huấn luyện các mẫu đối kháng được rút ra từ một tập hợp nhiều mô hình. Một nhược điểm quan trọng còn lại của huấn luyện đối kháng là nó có xu hướng quá phù hợp với vùng ràng buộc cụ thể được sử dụng để tạo ra các mẫu đối kháng (các mô hình được huấn luyện để chống lại các mẫu đối kháng trong một tiêu chuẩn tối đa có thể không chống lại các mẫu đối kháng dựa trên các sửa đổi lớn đối với điểm ảnh (pixel) ngay cả khi các mẫu đối kháng mới không có vẻ thách thức đặc biệt đối với người quan sát).

Hiện nay, sự cạnh tranh giữa các cuộc tấn công và phương pháp chống lại sự tấn công sử dụng các mẫu đối kháng trở thành một cuộc “chạy đua”: một phương pháp phòng chống tấn công được đề xuất để ngăn chặn các cuộc tấn công hiện có sau đó đã để lộ ra những hạn chế của phương pháp này là dễ bị tấn công bởi một số cuộc tấn công mới, và ngược lại.

Tóm lại, trên thực tế chưa có phương pháp luận để đánh giá về độ mạnh mẽ của mô hình, hay phương pháp bảo vệ nào bảo vệ hoàn hảo trước các tấn công vào mô hình học máy hay học sâu dù là tấn công hộp trắng hay hộp đen. Giải quyết những những vấn đề chưa được giải quyết là một hướng đi trong tương lai. Đây vẫn là một hướng nghiên cứu mang tính thời sự và đang được cải tiến liên

tục, nâng cao. Đây cũng là một phần luận án sẽ tập trung nghiên cứu và đề xuất giải pháp để nâng cao, đánh giá sự mạnh mẽ và đảm bảo độ an toàn cho mô hình học máy.

1.4.Kết luận chương 1

Trong chương 1 đã trình bày tổng quan về học máy, mô hình học sâu, deepfakes và tấn công đối kháng.

Gồm các khái niệm cơ bản của học máy và học sâu, cũng như cấu trúc và cách hoạt động của các mô hình học sâu. Học máy, và cụ thể hơn là học sâu, đã mang lại nhiều tiến bộ trong nhiều lĩnh vực khác nhau, từ thị giác máy tính đến xử lý ngôn ngữ tự nhiên.

Tiếp theo là deepfakes, một ứng dụng của học sâu có thể tạo ra video và âm thanh giả mạo chất lượng cao. Mặc dù deepfakes có tiềm năng trong một số ứng dụng chính đáng, nhưng chúng cũng tạo ra những mối đe dọa về bảo mật và quyền riêng tư.

Cuối cùng, trình bày về tấn công đối kháng, một lĩnh vực nghiên cứu đang ngày càng trở nên quan trọng trong học máy và trí tuệ nhân tạo. Những tấn công này, thực hiện bằng cách thay đổi dữ liệu đầu vào của mô hình học máy một cách tinh vi, có thể gây ra các dự đoán sai lệch và đưa ra hậu quả nghiêm trọng trong một số ứng dụng.

Tổng kết lại, mặc dù học máy và học sâu đã mang lại nhiều lợi ích, chúng cũng đi kèm với những thách thức và rủi ro đáng kể, đòi hỏi sự hiểu biết và nghiên cứu sâu sắc để đối phó. Trong các chương tiếp theo, sẽ trình bày sâu hơn vào cách thức hoạt động của tấn công đối kháng. Cụ thể là tấn công đối kháng sử dụng kết quả deepfakes.

CHƯƠNG 2: PHƯƠNG PHÁP TẤN CÔNG ĐỐI KHÁNG SỬ DỤNG DEEPPAKES

2.1. Tổng quan về tấn công đối kháng sử dụng deepfake

Trong một thế giới ngày càng số hóa, Deepfake đã tạo ra một đột phá vô cùng quan trọng trong lĩnh vực trí tuệ nhân tạo (AI) và học sâu. Bằng cách áp dụng các thuật toán học sâu phức tạp, Deepfake không chỉ có thể tạo ra hình ảnh và video giả mạo với độ phân giải cao và chất lượng tuyệt vời, mà còn có thể tái tạo giọng nói cực kỳ chính xác. Công nghệ này đang mở ra một thế giới mới về khả năng mô phỏng thực tế, nhưng cũng mang lại những nguy cơ và thách thức lớn về an ninh mạng, đặc biệt là khi nó được sử dụng trong các cuộc tấn công đối kháng nhằm đánh lừa các hệ thống học máy.

Những hình ảnh, âm thanh và video mà Deepfake tạo ra thực sự thuyết phục đến mức gần như không thể phân biệt được với thực tế. Điều này tạo ra một tiềm năng lớn cho những người tấn công muốn đánh lừa các hệ thống học máy. Các hệ thống nhận dạng khuôn mặt, giám sát video, nhận dạng giọng nói, đều có thể trở thành mục tiêu cho những cuộc tấn công đối kháng mà sử dụng Deepfake.

Hãy tưởng tượng một tình huống mà trong đó một hình ảnh Deepfake được tạo ra để mô phỏng khuôn mặt của một người nhất định. Hình ảnh này sau đó có thể được sử dụng để đánh lừa hệ thống nhận dạng khuôn mặt, khiến hệ thống nhầm lẫn giữa người thật và người giả. Trong một tình huống khác, một kẻ tấn công có thể sử dụng Deepfake để tạo ra một đoạn video giả mạo, với mục tiêu tạo ra một thông điệp lừa dối hoặc thậm chí tạo ra một sự hiểu lầm trong công chúng hoặc tổ chức nào đó.

Deepfake lại có thể đánh lừa đối với hệ thống học máy: Điều này chủ yếu phụ thuộc vào cách mà học máy, và đặc biệt là học sâu hoạt động. Học máy phụ thuộc vào việc tiếp nhận và học hỏi từ dữ liệu đầu vào để đưa ra dự đoán hoặc phân loại. Khi Deepfake làm biến dạng dữ liệu đầu vào này, nó cũng thay đổi cách mà hệ thống học máy đưa ra quyết định. Hậu quả của việc này có thể rất lớn, từ việc sai lệch trong việc phân loại hoặc dự đoán, cho đến việc tạo ra thông tin sai lệch có thể gây ra những hiểu lầm nghiêm trọng.

Dù Deepfake mang lại những nguy cơ tiềm ẩn, nó cũng tạo ra một lĩnh vực nghiên cứu hấp dẫn về việc phát hiện và chống lại những tấn công này. Nghiên cứu về cách phát hiện và ngăn chặn Deepfake không chỉ là cần thiết để bảo vệ các hệ thống học máy khỏi bị đánh lừa, mà còn quan trọng để bảo vệ xã hội khỏi những thông tin giả mạo và lạm dụng công nghệ.

Tấn công đối kháng sử dụng Deepfakes: Hãy tưởng tượng một hệ thống nhận dạng khuôn mặt. Hệ thống này được huấn luyện để phân biệt giữa khuôn mặt của các cá nhân khác nhau. Một kẻ tấn công có thể sử dụng Deepfake để tạo ra một hình ảnh khuôn mặt giả mạo của một người nào đó, và sau đó sử dụng hình ảnh này để đánh lừa hệ thống nhận dạng khuôn mặt.

Tương tự, trong một hệ thống nhận dạng giọng nói, một kẻ tấn công có thể sử dụng Deepfake để tạo ra một đoạn âm thanh giả mạo của một người nào đó, và sau đó sử dụng đoạn âm thanh này để đánh lừa hệ thống nhận dạng giọng nói. Nếu thành công, kẻ tấn công có thể truy cập không hợp pháp vào thông tin cá nhân, hoặc thậm chí thực hiện các hành động mà người bị đánh lừa không muốn.

Nói chung, tấn công đối kháng sử dụng Deepfakes đặt ra những thách thức lớn cho an ninh mạng. Bằng cách tạo ra hình ảnh, âm thanh và video giả mạo nhưng lại rất thuyết phục, Deepfakes có thể đánh lừa các hệ thống học máy, từ đó gây ra những hậu quả nghiêm trọng về an ninh và quyền riêng tư. Do đó, việc phát hiện và ngăn chặn Deepfakes đã trở thành một lĩnh vực nghiên cứu quan trọng.

2.2. Quy trình tấn công đối kháng sử dụng deepfake

2.2.1. Các bước tấn công đối kháng sử dụng deepfakes

Tấn công đối kháng thông qua Deepfakes đòi hỏi sự chuẩn bị kỹ lưỡng và phải tuân theo một số bước quan trọng. Dưới đây sẽ mô tả các bước cụ thể của quy trình này:

Xác định mục tiêu: Bước đầu tiên trong việc xây dựng một tấn công đối kháng sử dụng Deepfakes là xác định hệ thống mục tiêu. Điều này có thể là một hệ thống nhận dạng khuôn mặt, nhận dạng giọng nói, phân loại hình ảnh, hoặc bất kỳ hệ thống học máy nào khác mà có thể bị ảnh hưởng bởi dữ liệu đầu vào bị thay đổi. Việc hiểu rõ về hệ thống mục tiêu và cách nó hoạt động là yếu tố quan trọng để xây dựng một tấn công hiệu quả.

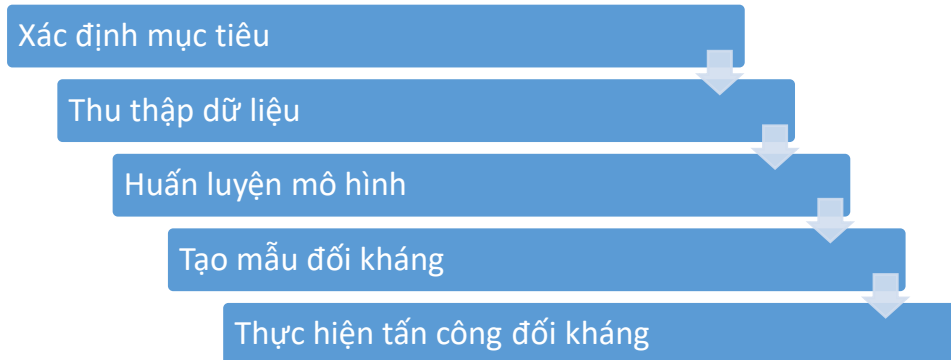
Thu thập dữ liệu huấn luyện: Trong quá trình này, kẻ tấn công cần thu thập một lượng lớn dữ liệu liên quan đến mục tiêu. Đối với tấn công nhận dạng khuôn mặt, đó có thể là hình ảnh khuôn mặt của người đang bị tấn công. Đối với tấn công nhận dạng giọng nói, đó có thể là các đoạn ghi âm của giọng nói của họ.

Tạo dữ liệu giả mạo bằng Deepfakes: Sử dụng dữ liệu huấn luyện đã thu thập, kẻ tấn công tiếp theo sẽ sử dụng phương pháp Deepfake để tạo ra dữ liệu giả mạo. Điều này có thể đòi hỏi việc sử dụng các thuật toán học sâu phức tạp như

autoencoders hoặc Generative Adversarial Networks (GANs). Kết quả là dữ liệu giả mạo thuyết phục, có thể đánh lừa được hệ thống mục tiêu.

Thực hiện tấn công đối kháng: Kẻ tấn công tiếp theo sẽ sử dụng dữ liệu giả mạo này để tấn công hệ thống mục tiêu. Điều này có thể thực hiện bằng cách đưa dữ liệu giả mạo vào hệ thống như một đầu vào hợp lệ, nhằm đánh lừa hệ thống đưa ra quyết định sai lệch.

Các bước này có thể được mô phỏng theo hình sau:



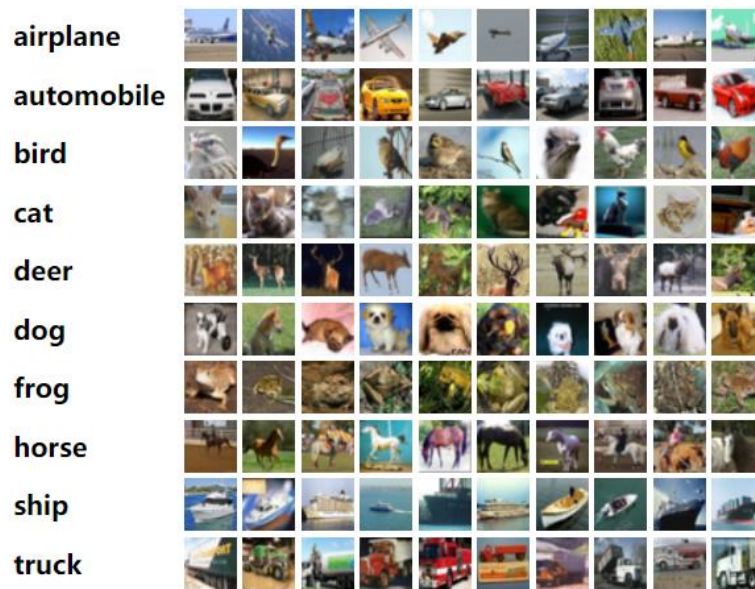
Hình 2.1. Sơ đồ mô phỏng các bước tấn công đối kháng

2.2.2. Xác định mục tiêu và thu thập dữ liệu huấn luyện

Mục tiêu của phương pháp là tạo ra một ảnh giả không thể phân biệt bằng mắt thường so với ảnh gốc ban đầu. Sau đó sử dụng tấn công đối kháng vào các mô hình học máy.

Lựa chọn dữ liệu:

Luận văn sử dụng dataset “cifar10” để nghiên cứu. Bộ dữ liệu CIFAR-10 (Viện nghiên cứu nâng cao Canada, 10 lớp) là một tập hợp con của bộ dữ liệu Tiny Images và bao gồm 60.000 hình ảnh màu 32x32. Các hình ảnh được gắn nhãn với một trong 10 loại loại trừ lẫn nhau: máy bay, ô tô, chim, mèo, nai, chó, ếch, ngựa, tàu và xe tải. Có 6000 hình ảnh mỗi lớp với 5000 hình ảnh đào tạo và 1000 hình ảnh thử nghiệm mỗi lớp.



Hình 2.2. Hình mẫu dataset CIFAR10

Nguồn <https://www.cs.toronto.edu/~kriz/cifar.html>

Mười lớp chính của dataset là airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck

Mô hình mục tiêu tấn công:

Từ bảng khảo sát các model công bố đã phân lớp thành công trên bộ dữ liệu cifar 10 sau:

Bảng 2.1. Bảng thông số đánh giá các model đã công bố của dataset CIFAR10

Model	Top-1 Acc.(%)	Top-5 Acc.(%)	Params.(M)	MAdds(M)
resnet20	92.60	99.81	0.27	40.81
resnet32	93.53	99.77	0.47	69.12
resnet44	94.01	99.77	0.66	97.44
resnet56	94.37	99.83	0.86	125.75
vgg11_bn	92.79	99.72	9.76	153.29
vgg13_bn	94.00	99.77	9.94	228.79
vgg16_bn	94.16	99.71	15.25	313.73
vgg19_bn	93.91	99.64	20.57	398.66

mobilenetv2_x0_5	92.88	99.86	0.70	27.97
mobilenetv2_x0_75	93.72	99.79	1.37	59.31
mobilenetv2_x1_0	93.79	99.73	2.24	87.98
mobilenetv2_x1_4	94.22	99.80	4.33	170.07
shufflenetv2_x0_5	90.13	99.70	0.35	10.90
shufflenetv2_x1_0	92.98	99.73	1.26	45.00
shufflenetv2_x1_5	93.55	99.77	2.49	94.26
shufflenetv2_x2_0	93.81	99.79	5.37	187.81
repvgg_a0	94.39	99.82	7.84	489.08
repvgg_a1	94.89	99.83	12.82	851.33
repvgg_a2	94.98	99.82	26.82	1850.10

Tiến hành lựa chọn ngẫu nhiên 5 model để thực hiện nghiên cứu tấn công là resnet56 (với độ chính xác 94.37%), vgg19_bn (với độ chính xác 93.91%), mobilenetv2_x1_4 (với độ chính xác 94.22%), shufflenetv2_x2_0 (với độ chính xác 93.81%), repvgg_a2 (với độ chính xác 94.98%).

• **resnet56 (với độ chính xác 94.37%)**

Model được công bố năm 2020 bởi [21] Wei Deng, Qi Feng, Liyao Gao, Faming Liang, Guang Lin. ResNet-56 là một biến thể của kiến trúc ResNet (Residual Network) được thiết kế cho bài toán phân loại hình ảnh. Kiến trúc này chứa tổng cộng 56 lớp convolution và fully connected layers.

Trong ResNet-56, kiến trúc cơ bản là "Residual Block", và mô hình sẽ được xây dựng bằng cách lặp lại các khối này nhiều lần:

Residual Block:

Mục tiêu của Residual Block là học được sự thay đổi (residual) của dữ liệu đầu vào thay vì cố gắng học một ánh xạ hoàn toàn mới.

Mỗi khối cơ bản trong ResNet-56 bao gồm hai lớp convolution với batch normalization và activation function ReLU.

Đầu ra của lớp convolution đầu tiên sẽ được truyền qua một lớp batch normalization, sau đó là hàm kích hoạt ReLU.

Sau đó, đầu ra này sẽ được truyền qua lớp convolution thứ hai với cùng số lượng bộ lọc và kích thước.

Đầu ra cuối cùng của khối cơ bản này sẽ được cộng với đầu vào ban đầu thông qua kết nối "skip connection". Kết quả của phép cộng này sẽ được truyền qua một hàm kích hoạt ReLU.

Kiến trúc chính:

Kiến trúc ResNet-56 bao gồm 9 khối Residual Blocks. Các khối này sẽ có số lượng lớp convolution và số lượng bộ lọc khác nhau.

Thông thường, mạng ResNet có số lượng lớp convolution tương đối lớn, vì vậy kiến trúc ResNet-56 sẽ giúp giảm thiểu vấn đề gradient biến mất.

Global Average Pooling và Fully Connected Layer:

Sau khi các khối Residual Blocks, mô hình sử dụng lớp Global Average Pooling để chuyển đổi mảng 3D thành vector 1D.

Cuối cùng, sử dụng một fully connected layer để thực hiện phân loại vào các lớp.

ResNet-56 là một mô hình mạng nơ-ron sâu đơn giản, nhưng lại có khả năng giải quyết vấn đề biến mất gradient, giúp nâng cao hiệu suất đối với nhiều bài toán phân loại hình ảnh.

•vgg19_bn (với độ chính xác 93.91%)

Model được công bố năm 2021 bởi [22] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, Florian Tramer. VGG19_bn là một biến thể của kiến trúc VGG (Visual Geometry Group) được mở rộng và cải tiến bằng cách thêm các lớp Batch Normalization. Kiến trúc VGG nổi tiếng với sự sâu và các lớp convolution 3x3 liên tiếp. Sự thêm vào của Batch Normalization giúp cải thiện hiệu suất huấn luyện và tăng tốc quá trình hội tụ.

Kiến trúc chính của VGG19_bn:

Convolutional Layers:

VGG19_bn bao gồm một loạt các lớp convolution với bộ lọc kích thước 3x3 và bước nhảy 1 (stride = 1).

Các lớp convolution được xếp chồng lên nhau, và số lượng bộ lọc tăng dần theo cấu trúc: 64-128-256-256-512-512-512-512.

Batch Normalization:

Lớp Batch Normalization được thêm vào sau mỗi lớp convolution. Batch Normalization giúp chuẩn hóa đầu ra của lớp trước khi áp dụng hàm kích hoạt, giúp cải thiện quá trình học và kiểm soát biến mất gradient.

Activation Function:

Thường là hàm kích hoạt ReLU (Rectified Linear Activation) sau mỗi lớp convolution và Batch Normalization.

Max Pooling Layers:

Sau mỗi nhóm 2 lớp convolution, VGG19_bn sử dụng lớp Max Pooling với kích thước cửa sổ 2x2 và bước nhảy 2 (stride = 2) để giảm kích thước không gian của đặc trưng.

Fully Connected Layers:

Sau khi các lớp convolution và max pooling, mô hình sẽ sử dụng các lớp fully connected để thực hiện phân loại.

VGG19_bn thường có 3 lớp fully connected với số lượng đơn vị ẩn giảm dần: 4096-4096-1000 (đối với VGG19 ban đầu), tương ứng với số lượng lớp phân loại.

Ưu điểm của VGG19_bn:

Độ sâu và biểu diễn mạnh mẽ: Với nhiều lớp convolution liên tiếp và độ sâu, VGG19_bn có khả năng học các đặc trưng phức tạp và biểu diễn sâu sắc.

Batch Normalization: Sự thêm vào của Batch Normalization giúp kiểm soát biến mất gradient, tăng tốc độ hội tụ và cải thiện khả năng huấn luyện.

VGG19_bn là một kiến trúc mạng nơ-ron sâu mạnh mẽ và hiệu quả, thường được sử dụng trong các bài toán phân loại hình ảnh và trích xuất đặc trưng.

•mobilenetv2_x1_4 (với độ chính xác 94.22%)

Model công bố 2018 bởi [23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen.

MobileNetV2 là một mô hình mạng nơ-ron sâu được thiết kế đặc biệt để có hiệu suất cao trong việc giảm kích thước mô hình và tăng tốc độ tính toán, đặc biệt là trên các thiết bị có tài nguyên hạn chế như điện thoại di động.

Kiến trúc chính của MobileNetV2:

Depthwise Separable Convolution:

Một phần quan trọng của MobileNetV2 là lớp Depthwise Separable Convolution, bao gồm hai giai đoạn: depthwise convolution và pointwise convolution.

Depthwise convolution: thực hiện convolution trên mỗi kênh riêng biệt của dữ liệu đầu vào.

Pointwise convolution: sau đó thực hiện một lớp convolution 1×1 thông qua tất cả các kênh đồng thời. Điều này giúp giảm số lượng tham số và tính toán.

Inverted Residual Blocks:

MobileNetV2 sử dụng khối Inverted Residual Block để tạo độ sâu và cải thiện khả năng biểu diễn.

Khối Inverted Residual Block bao gồm depthwise convolution, pointwise convolution, và skip connection (kết nối bỏ qua).

Bottleneck Structures:

MobileNetV2 sử dụng bottleneck structures, tức là sử dụng lớp convolution 1×1 để giảm kích thước đầu vào trước khi áp dụng depthwise convolution.

Linear Bottlenecks and Expansion Layers:

Linear bottleneck là một phiên bản của bottleneck structure với hệ số mở rộng là 1. Nó giúp giảm số lượng tham số và tính toán trong các khối.

Expansion layer thực hiện mở rộng số kênh của đặc trưng trước khi áp dụng depthwise convolution và pointwise convolution.

Ưu điểm của MobileNetV2:

Hiệu suất tính toán: MobileNetV2 tối ưu hóa cho việc tính toán trên các thiết bị có tài nguyên hạn chế như điện thoại di động, giúp tăng tốc quá trình dự đoán.

Kích thước nhỏ gọn: Kiến trúc kiểu Depthwise Separable Convolution giúp giảm kích thước mô hình so với các kiến trúc truyền thống.

Khả năng di động cao: MobileNetV2 là một sự kết hợp tốt giữa hiệu suất và khả năng di động, thích hợp cho các ứng dụng yêu cầu phân loại hình ảnh trên thiết bị di động.

MobileNetV2 là một kiến trúc mạng nơ-ron sâu đáng chú ý trong lĩnh vực học máy và thị giác máy tính, đặc biệt là trong các ứng dụng di động và nhúng.

- shufflenetv2_x2_0 (với độ chính xác 93.81%)

Model công bố 2022 bởi [24] Longqing Ye trong bài AugShuffleNet: Communicate More, Compute Less.

ShuffleNetV2 là một kiến trúc mạng nơ-ron sâu được thiết kế đặc biệt để cải thiện hiệu suất tính toán và khả năng di động trong các ứng dụng thị giác máy tính, đặc biệt là trên các thiết bị có tài nguyên hạn chế như điện thoại di động và thiết bị nhúng. Được giới thiệu bởi sự kết hợp của các ý tưởng như depthwise convolution, channel shuffle, và group convolution, ShuffleNetV2 mang lại hiệu suất tốt với kích thước mô hình nhỏ gọn.

Kiến trúc chính của ShuffleNetV2:

Ý tưởng chính của ShuffleNetV2 là "channel shuffle", cho phép thông tin từ các kênh đầu vào được trộn lẫn nhau để tạo ra sự kết hợp đa dạng và cải thiện khả năng biểu diễn.

Quá trình này giúp mô hình học cách kết hợp thông tin từ các kênh khác nhau để tạo ra các đặc trưng phức tạp và hiệu quả.

ShuffleNetV2 sử dụng depthwise convolution trong các khối cơ bản, trong đó mỗi kênh của dữ liệu đầu vào được xử lý riêng biệt.

Depthwise convolution giúp giảm số lượng tham số và tính toán so với các lớp convolution truyền thống.

Group Convolution: Một biến thể khác của ShuffleNetV2 sử dụng group convolution để thay thế depthwise convolution trong một số trường hợp. Group convolution chia các kênh đầu vào thành các nhóm nhỏ và thực hiện convolution trên từng nhóm.

Bottleneck Structures: ShuffleNetV2 sử dụng khối bottleneck để tạo độ sâu và cải thiện khả năng biểu diễn.

Khối bottleneck bao gồm depthwise convolution, pointwise convolution, và channel shuffle.

Ưu điểm của ShuffleNetV2:

Hiệu suất tính toán: ShuffleNetV2 được thiết kế để tối ưu hóa việc tính toán, giúp tăng tốc độ tính toán và giảm thời gian dự đoán.

Khả năng di động: Kiến trúc này rất thích hợp cho các ứng dụng di động và nhúng do kích thước mô hình nhỏ gọn và khả năng thích ứng tốt với tài nguyên hạn chế.

Hiệu suất tốt: ShuffleNetV2 có khả năng biểu diễn tốt và cho hiệu suất đáng chú ý trên nhiều tập dữ liệu thị giác máy tính.

ShuffleNetV2 là một ví dụ xuất sắc về việc sáng tạo kiến trúc mạng nơ-ron để đáp ứng các yêu cầu của các ứng dụng thị giác máy tính hiệu suất cao và có tính di động.

• repvgg_a2 (với độ chính xác 94.98%)

Model được công bố năm 2021 bởi [25] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, Jian Sun trong bài RepVGG: Making VGG-style ConvNets Great Again.

RepVGG là một kiến trúc mạng nơ-ron được thiết kế để cân bằng giữa hiệu suất và tính di động trong các ứng dụng thị giác máy tính. Mục tiêu của RepVGG là tạo ra một kiến trúc mạng đơn giản và hiệu quả mà vẫn đảm bảo tính linh hoạt trong việc điều chỉnh hiệu suất và kích thước mô hình.

Kiến trúc chính của RepVGG:

RepVGG sử dụng một kỹ thuật gọi là "reparameterization trick" để chuyển đổi lớp convolution thành lớp có thể đặt vị trí (position-wise layer). Lớp convolution được chia thành hai phần: một phần tương tự Fully Connected (FC) và một phần tương tự Depthwise Separable Convolution (DSConv).

RepVGG sử dụng reparameterization trick để tạo ra một phần đầy đủ và một phần thụt lùi. Sự kết hợp này giúp kiểm soát overfitting và tạo ra tính linh hoạt trong việc điều chỉnh hiệu suất mô hình.

Bottleneck Structures: Kiến trúc của RepVGG bao gồm các khối bottleneck để tạo ra độ sâu và cải thiện khả năng biểu diễn. Khối bottleneck kết hợp giữa reparameterization và lớp Batch Normalization để tối ưu hóa quá trình học.

RepVGG đạt được kích thước mô hình nhỏ gọn bằng cách tối ưu hóa cấu trúc lớp convolution và sử dụng reparameterization trick.

Ưu điểm của RepVGG:

Hiệu suất và tính di động: RepVGG cân bằng giữa hiệu suất và tính di động, cho phép tạo ra các mô hình mạng nơ-ron có khả năng dự đoán nhanh chóng trên các thiết bị có tài nguyên hạn chế.

Linh hoạt và đa dạng hiệu suất: RepVGG cung cấp nhiều phiên bản mô hình với khả năng linh hoạt trong việc tinh chỉnh hiệu suất dựa trên yêu cầu ứng dụng cụ thể.

RepVGG là một kiến trúc mạng nơ-ron đáng chú ý, có khả năng linh hoạt trong việc cân bằng giữa hiệu suất, kích thước mô hình và tính di động.

2.2.3. Xây dựng mô hình và tạo mẫu đối kháng

Xây dựng mô hình:

Ý tưởng phương pháp là với bất kỳ mô hình phân loại cho trước, xây dựng một mô hình sinh ảnh nhiễu, kết hợp với hình ảnh gốc nhằm đánh lừa mô hình phân loại. Phương pháp này chỉ quan tâm đến đầu vào và đầu ra của mô hình mục tiêu do đó nó là phương pháp tấn công hộp đen.

Các bước thực hiện:

Bước (a): Với đầu vào là một hình ảnh, đầu ra là nhiễu được tạo ra từ hình ảnh đã cho trước đó.

Bước (b): Sử dụng nhiễu thu được từ (a) kết hợp với hình gốc đầu vào, thu được hình ảnh mới (so sánh kết quả này và hình đầu vào, mắt thường không thể phân biệt được sự khác biệt).

Bước (c): Đưa kết quả từ (b) qua mô hình phân loại muốn đánh lừa

Bước (d): Kết quả thu được từ (c) sẽ khác so với kết quả đưa hình gốc qua model phân loại.

Tổng quan về mô hình được hiển thị trong Hình 2.3.

Ví dụ cụ thể:

Bài toán là đưa vào hình ảnh con mèo kết hợp với nhiễu được sinh vào mô hình nhận dạng, đánh lừa mô hình nhận dạng thành không phải con mèo, đây là tấn công không có mục tiêu. Mục tiêu sẽ để mô hình tự chọn một trong những loại nhiễu sinh ra chiếm tỉ lệ % phân loại lớn nhất.

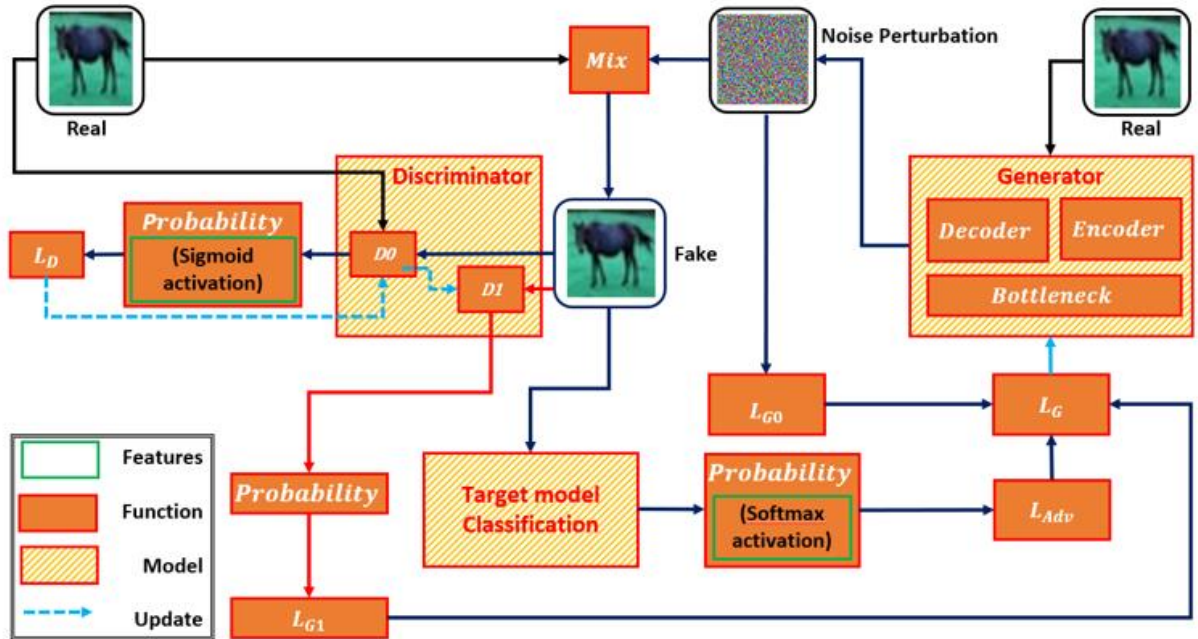
Input: mèo

Noise mask: máy bay: 90%, ngựa 70%,.....

Output: Noise_mask + ORG Image

Thì sẽ chọn mặt nạ nhiễu mang những đặt trưng của máy bay để ghép với hình gốc và đánh lừa mô hình.

Cuộc tấn công được xem là thành công khi bức ảnh được đưa vào mô hình, mô hình phân loại sai phân lớp ban đầu của bức ảnh và phải đúng với phân lớp là dê. Nếu phân lớp ra phân loại khác với dê nghĩa là tấn công chưa thành công.



Hình 2.3. Sơ đồ quá trình huấn luyện mô hình và tạo mẫu đối kháng

Cụ thể các bước thiết lập mô hình sẽ được trình bày như sau:

Đầu vào của mô hình là hình ảnh trong bộ dữ liệu cifar10.

Thông thường mạng GAN chỉ sử dụng nhiễu z làm đầu vào cho trình tạo G để tạo dữ liệu giả $G(z)$ gần giống với dữ liệu thực và nó được dùng để đánh lừa Discriminator. Discriminator sẽ thực hiện phân biệt giữa dữ liệu được sinh ra $D(G(z))$ và dữ liệu thực $D(x)$ và nhận ý. Cụ thể, công thức tổng quan của quy trình của GAN đối với các mô hình thường như sau:

$$V(D, G) = E_{x \sim p_{data}(x)} + E_{z \sim p_z(x)} \left[\log \log \left(1 - D(G(z|y)) \right) \right] \quad (20)$$

Mô hình đề xuất:

Model sử dụng mạng GAN gồm 2 mạng là mạng sinh (G - Generator) và mạng so sánh (D - Discriminator). Hình ảnh đầu vào sẽ được thuật toán CNN (deeplearning) khi mà xây dựng model với model mục tiêu (input model) sau quá trình forwarding, loss caculation, backpropagation các kernel (CNN) sẽ được update thông số (weight) một cách tự động. Và thông qua quá trình upsampling tạo thành model Generator còn lại qua quá trình downsampling để tạo ra mặt nạ nhiễu và sau đó kiểm tra kết quả phân lớp.

Nếu một mô hình phân loại f được đào tạo trên tập dữ liệu $X \subseteq R^n$ với n là một hình ảnh trong tập hình ảnh đầu vào. Và giả sử (x_i, y_i) là trường hợp thứ i trong dữ liệu huấn luyện, trong đó $x_i \in X$ được tạo từ một số dữ liệu không xác định và $y_i \in Y$ là nhãn.

Mô hình phân loại f được huấn luyện lên ảnh tự nhiên và đạt độ chính xác cao. Mục đích của một kẻ tấn công là tạo ra một ví dụ về đối thủ x_{adv} là một cái gì đó, một ảnh nào đó có thể đánh lừa f để đưa ra một dự đoán sai. Tức là khiến f nhận dạng x_i có nhãn là y_i thành một nhãn khác y_i .

Mạng **Generator** G lấy một hình ảnh sạch x và nhãn lớp đích t làm đầu vào để tạo nhiễu loạn. t được lấy mẫu có phần trăm lớn nhì từ mô hình mục tiêu phân lớp tập dữ liệu trừ kết quả cao nhất là nhãn ban đầu.

Hàm Mix bao gồm 2 thành phần :

$$Mix = x_{adv} = x + (K * pert)$$

Trong đó $pert$ là mặt nạ nhiễu tạo ra bởi mạng Generator

Hàm mất mát của mạng D bao gồm bốn phần:

L_D để phân biệt ảnh thật/ảnh nhiễu

L_{D0} để phân biệt ảnh thật và mặt nạ nhiễu

K là tỷ lệ mặt nạ nhiễu so với kích cỡ ảnh ban đầu

L_{D1} để phân loại trên các mặt nạ nhiễu được tạo bởi kẻ tấn công và mạng G .

$$L_D = E [\log P (S = real | x_{real})] + K * E [\log P (S = pert | x_{pert})]$$

$$L_{D0} = E [\log P (C = y | x_{pert})]$$

$$L_{D1} = L_{G1} = E[\sqrt{(d_fake - (d_fake * 0 + 1))^2}]$$

Trong đó y đại diện cho nhãn ban đầu.

" d_fake " đại diện cho kết quả của mô hình $D0$ biểu diễn dưới dạng ma trận với số chiều tùy vào $bath_size$ đưa vào

"($d_fake * 0 + 1$)" là ma trận 1 tương đồng ví dụ " d_fake " là $\begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$ thì " $(d_fake * 0 + 1)$ " sẽ là $\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$, tất cả vị trí đều là 1.

Mục tiêu của Discriminator là tối thiểu hóa L_{D1} . Do đó phải tối đa hóa L_D và L_{D0}

Hàm mất mát của mạng G bao gồm ba phần:

L_{G0} để tính mất mát của mặt nạ nhiễu,

L_{adv} kết quả mục tiêu hoàn thành sau khi được so sánh với mô hình Target ban đầu

K là tỷ lệ mặt nạ nhiễu so với kích cỡ ảnh ban đầu,

$L_{G1} = L_{D1}$ là kết quả sau Discriminator (kiểm tra).

Các hàm được tính như sau:

$L_{G0} = Avg(\text{Max}_{0 \leq i \leq n}(\text{pert}(i) - \alpha; 0))$ (trung bình của giá trị lớn nhất mặt nạ nhiễu thứ i trừ đi alpha so sánh với 0)

$L_{adv} = \sum (\text{Max}_{0 \leq i \leq n}(x_{adv} - x, 0))$ (Tổng của x_{adv} giá trị lớn nhất (ảnh giả) thứ i trừ đi x (ban đầu) thứ i so sánh với 0)

Mục tiêu của Generator là tối thiểu hóa $L_G = \gamma L_{G0} + \alpha L_{adv} + \beta L_{G1}$ trong đó γ, α, β là các tham số để điều chỉnh mô hình (tuning).

2.2.4. Thực hiện tấn công đối kháng

Qua quá trình huấn luyện ta thu được model G và D. Sử dụng ảnh mẫu đã được phân lớp có gắn nhãn từ dataset. Ta tiến hành thực hiện sinh ra mặt nạ nhiễu và tối ưu giá trị K (tỷ lệ mặt nạ nhiễu so với kích cỡ ảnh ban đầu) để sinh ra x'' (ảnh giả). Dùng ảnh giả tấn công đối kháng vào model ban đầu.

2.3. Phương pháp và tiêu chí đánh giá

Để chọn lựa được K và đánh giá được độ khác nhau của ảnh giả và ảnh ban đầu luận văn sử dụng các phương pháp đánh giá giữa ảnh thực và ảnh giả như sau:

- Sử dụng phương pháp SSIM để kiểm tra độ giống nhau về cấu trúc của hai ảnh. SSIM - Structural Similarity Index (Chỉ số tương đồng cấu trúc), là một phương pháp được sử dụng để đo lường sự tương đồng giữa hai hình ảnh. Điều đặc biệt về SSIM là nó cố gắng mô phỏng cách con người cảm nhận và phản ánh về mặt thị giác. Thay vì chỉ sử dụng các đặc trưng cơ bản như giá trị màu của các pixel, SSIM lấy cảm nhận thị giác vào xem xét. SSIM xem xét các yếu tố sau đây trong việc so sánh hai hình ảnh:
 - Cường độ độ tương phản (Luminance): SSIM đo sự tương đồng về cường độ độ sáng và độ tương phản giữa các vùng của hai hình ảnh.

- Cấu trúc (Structure): SSIM xem xét sự tương đồng về cấu trúc và hình dạng của các đối tượng trong hình ảnh.
- Biến đổi màu (Texture): SSIM đo sự tương đồng về mẫu, kết cấu, và biến đổi màu sắc trong hình ảnh.
- Công thức tính SSIM như sau:

$$SSIM(x, y) = ((2\mu_x\mu_y + C1) * (2\sigma_{xy} + C2)) / ((\mu_x^2 + \mu_y^2 + C1) * (\sigma_x^2 + \sigma_y^2 + C2))$$

Trong đó:

x và y là hai hình ảnh cần so sánh

μ_x và μ_y là giá trị trung bình của x và y

σ_x^2 và σ_y^2 là phương sai của x và y

σ_{xy} là covariance của x và y .

$C1$ và $C2$ là hai hằng số để tránh chia cho 0 trong trường hợp biểu thức có mẫu số nhỏ.

Kết quả của SSIM là một giá trị nằm trong khoảng từ -1 đến 1, với giá trị càng gần 1 thể hiện sự tương đồng càng cao giữa hai hình ảnh.

- Sử dụng phương pháp PSNR để đánh giá độ giống nhau của hai ảnh. PSNR - Peak Signal-to-Noise Ratio (Tỷ lệ tín hiệu - nhiễu cực đại), là một chỉ số được sử dụng để đo lường chất lượng hình ảnh sau khi nó đã được nén hoặc xử lý. PSNR thường được sử dụng để so sánh giữa hình ảnh gốc và hình ảnh đã được xử lý, ví dụ như sau khi áp dụng các thuật toán nén như JPEG hoặc sau khi thực hiện các thay đổi trên hình ảnh. Công thức để tính PSNR là:

$$PSNR = 10 * \log_{10} \frac{MAX^2}{MSE}$$

Trong đó:

MAX là giá trị tối đa mà pixel có thể đạt trong hình ảnh. Với hình ảnh 8-bit, MAX thường là 255.

MSE là giá trị trung bình của bình phương sai (Mean Squared Error) giữa các pixel trong hình ảnh gốc và hình ảnh đã xử lý.

Giá trị PSNR càng cao, thì hình ảnh đã xử lý càng giống với hình ảnh gốc, và hiệu suất nén càng tốt.

- Sử dụng các công thức tính Norm để đo lường sự thay đổi giữa hai bức ảnh:

- L_0 chỉ số này đo lường số lượng pixel khác nhau giữa hai hình ảnh. Nếu hai hình ảnh giống nhau hoàn toàn, L_0 bằng 0; nếu chúng hoàn toàn khác nhau, L_0 sẽ bằng tổng số pixel trong hình ảnh. Nếu so sánh hai hình ảnh mà chỉ có một pixel khác nhau, thì L_0 sẽ bằng 1.
- L_1 còn gọi là "norm chuẩn bậc 1" hoặc "chuẩn Manhattan," là một phép đo "độ lớn tuyệt đối" của một vector hoặc tensor. Trong trường hợp của một tensor, L1 norm được tính bằng cách lấy tổng giá trị tuyệt đối của tất cả các phần tử trong tensor. Công thức tính như sau:

$$L_1 = MAE = \frac{1}{n} \sum_{i=1}^n |y_i - x_i|$$

Trong đó:

n là số lượng mẫu trong tập dữ liệu.

y_i là giá trị tensor của ảnh giả thứ i

x_i là giá trị tensor của ảnh thực thứ i

- L_2 (Norm chuẩn bậc 2) thường được sử dụng trong bài toán học máy để đo lường mức độ sai lệch giữa dự đoán và giá trị thực tế, và nó thường tương đương với bình phương của sai số trung bình, hay còn gọi là Mean Squared Error (MSE). Công thức tính như sau:

$$L_2 = MSE = \frac{1}{n} \sum_{i=1}^n (y_i - x_i)^2$$

Trong đó:

n là số lượng mẫu trong tập dữ liệu.

y_i là giá trị tensor của ảnh giả thứ i

x_i là giá trị tensor của ảnh thực thứ i

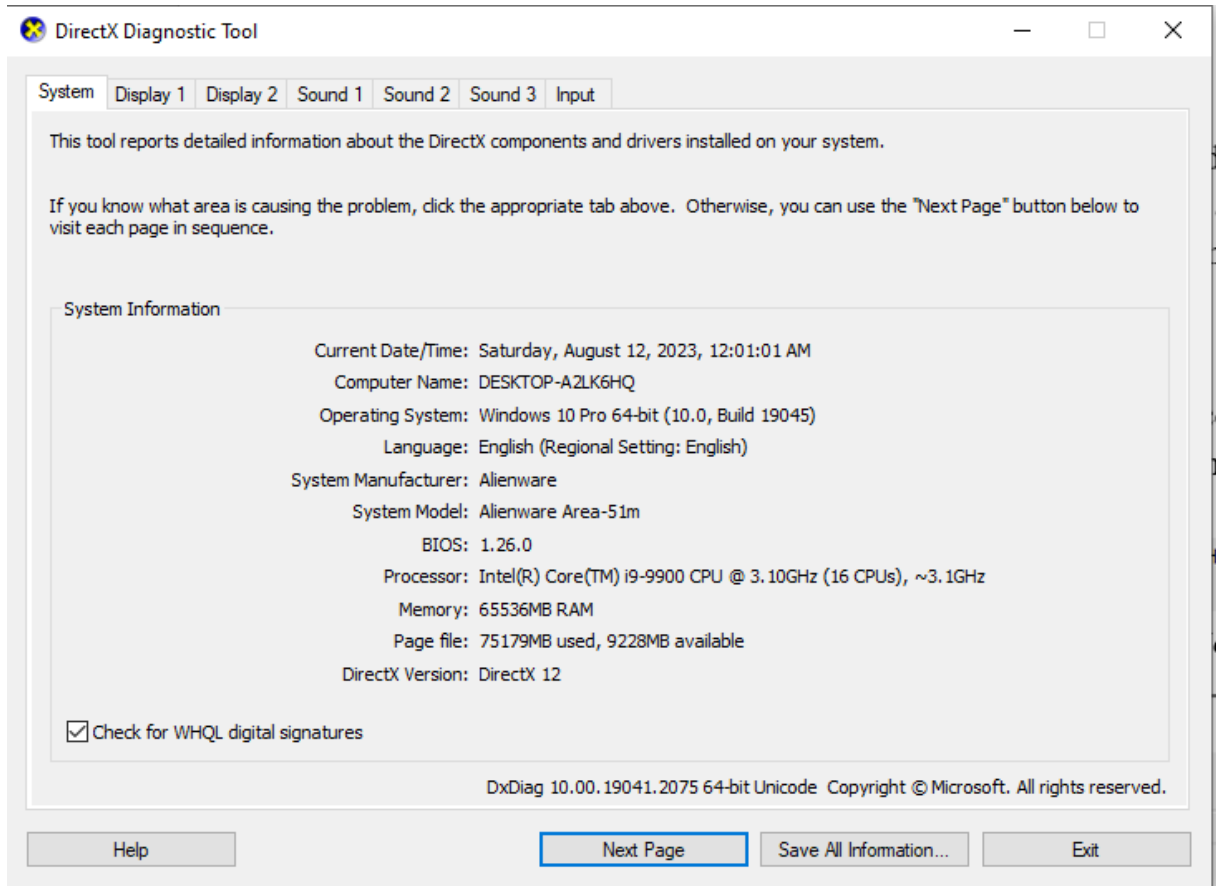
2.4. Kết luận chương 2

Qua chương 2 luận văn đã tìm hiểu được một số phương pháp tấn công đối kháng. Đề xuất một phương pháp, quy trình tấn công đối kháng sử dụng deepfakes được cải tiến các kiến trúc của mạng Generator như cải tiến resnetblock trong bottleneck cụ thể là 4 resnetblock(32) so với 1 resnetblock(32) của nghiên cứu AIGAN [20], đồng thời luận văn cũng đề xuất đánh giá hàm loss Generator sau khi huấn luyện Discriminator một lần với tám ảnh cần đánh giá để hạn chế False Positive trong việc phân loại thật giả, mục đích cuối là để giá trị hàm loss

Discriminator được thấp nhất. Trong chương 3 luận văn sẽ trình bày kết quả thực nghiệm và đánh giá phương pháp.

CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ

3.1. Chuẩn bị dữ liệu huấn luyện



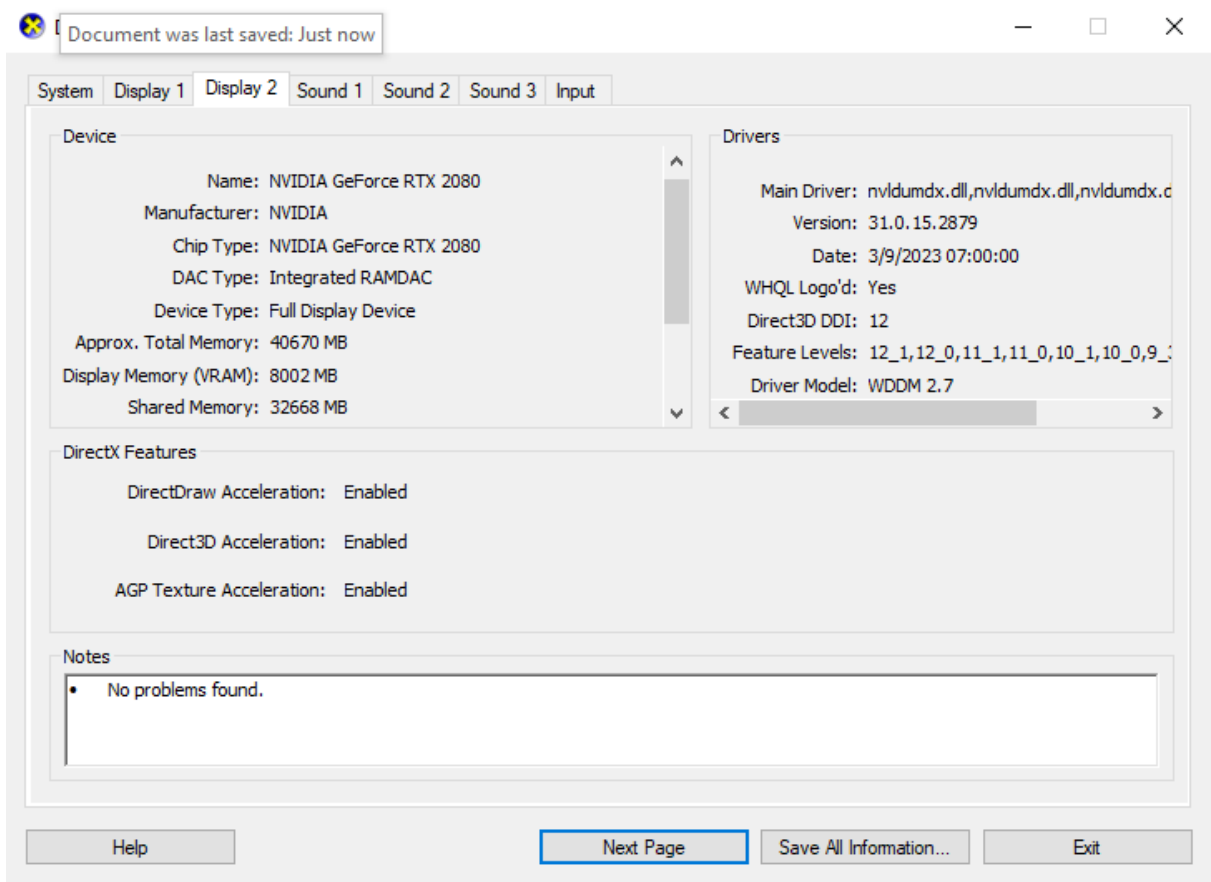
Hình 3.1. Cấu hình máy tính sử dụng thực nghiệm

Bộ dữ liệu CIFAR-10 (Viện nghiên cứu nâng cao Canada, 10 lớp) là một tập hợp con của bộ dữ liệu Tiny Images và bao gồm 60.000 hình ảnh màu 32x32. Các hình ảnh được gắn nhãn với một trong 10 loại trừ lẫn nhau: máy bay, ô tô, chim, mèo, nai, chó, ếch, ngựa, tàu và xe tải.

3.2. Huấn luyện mô hình

Mục tiêu của nghiên cứu là huấn luyện được mô hình Generator để sinh mặt nạ nhiễu và mô hình Discriminator để nhận diện ảnh thật và ảnh giả. Do đó việc huấn luyện này nhằm mục đích huấn luyện ra hai mô hình 1 Generator và 1 Discriminator.

Môi trường chạy thuật toán sử dụng bản phân phối Anaconda Python 3.11.3, framework Pytorch 1.8 và thư viện CUDA 10.1, trên hệ thống máy tính cấu hình CPU I9 9900, RAM 64GB, GPU RTX2080 8GB VRAM.



Hình 3.2. Cấu hình VGA sử dụng huấn luyện mô hình

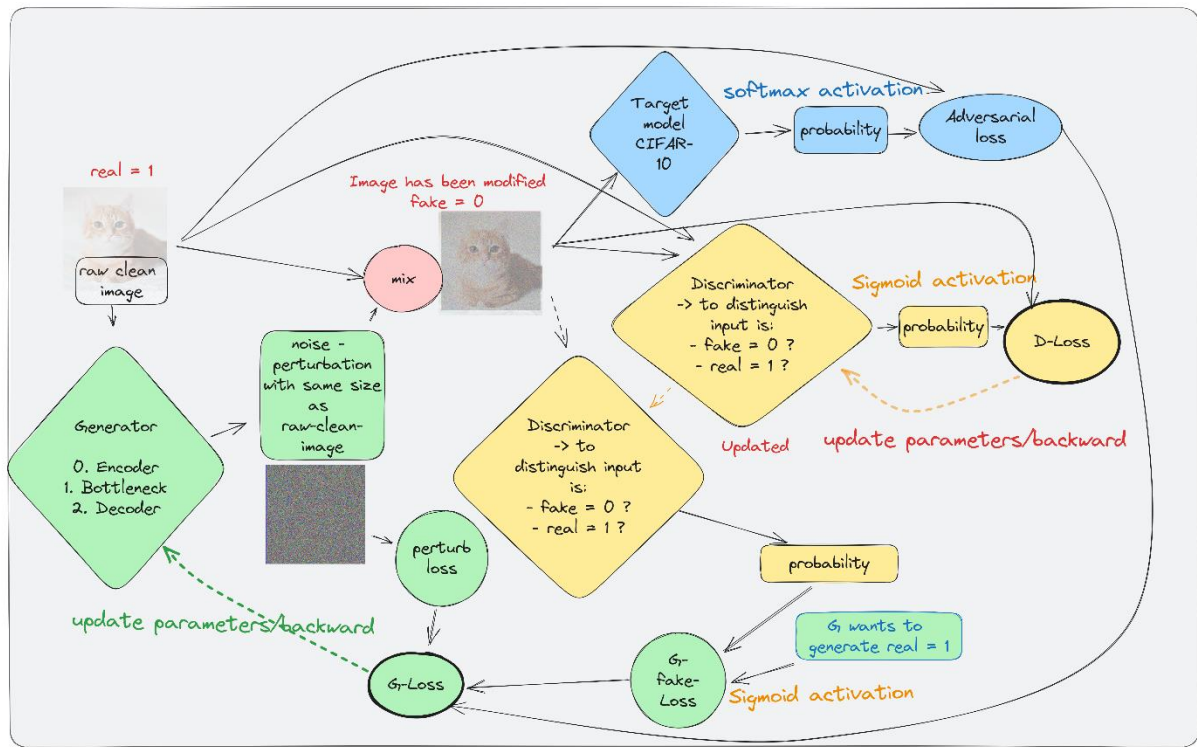
Quá trình huấn luyện là một quá trình mất nhiều thời gian, cần thực hiện nhiều thử nghiệm ở mỗi lần chạy thuật toán để tìm ra tệp tin trọng số có kết quả mong đợi. Một số điểm cần chú ý trong quá trình này đó là:

- Việc tăng kích thước đầu vào của mô hình có thể cải thiện độ chính xác của mô hình, tuy nhiên số lượng tham số trong mạng cũng sẽ tăng lên rất nhanh, đòi hỏi nhiều tài nguyên hệ thống hơn.
- Việc thực huấn luyện mô hình chạy trên GPU sẽ nhanh hơn rất nhiều lần so với chạy trên CPU.
- Phải kiểm tra kỹ tệp tin nhãn và tệp tin hình ảnh tương ứng trong tệp dữ liệu, tránh sai sót về định dạng đánh nhãn.

Với dữ liệu từ bộ dataset cifar10, tiến hành huấn luyện mô hình với kích thước batch là 128, kích thước ảnh đầu vào là 32x32, ngưỡng hỗn loạn của mô hình là 0.5, sau 600 epochs thì mô hình hội tụ.

Thực hiện huấn luyện riêng biệt với 5 mô hình được chọn ở chương 2 là: resnet56 (với độ chính xác 94.37%), vgg19_bn (với độ chính xác 93.91%),

mobilenetv2_x1_4 (với độ chính xác 94.22%), shufflenetv2_x2_0 (với độ chính xác 93.81%), repvgg_a2 (với độ chính xác 94.98%).



Hình 3.3. Mô tả quá trình huấn luyện

Huấn luyện mô hình Generator là quá trình tạo ra mặt nạ nhiễu từ đây mô hình tính toán được perturb-loss, sau đó mặt nạ nhiễu sẽ được kết hợp với ảnh gốc để sinh ra ảnh giả, ảnh giả này được đưa qua nhận diện bởi mô hình mục tiêu tấn công và tính được kết quả adversarial loss. Mặc khác ảnh giả được đưa qua mô hình Discriminator để nhận diện xem là giả hay thật ở đây mô hình gán nhãn real là 1 và fake là 0 sau đó sẽ tính được D-loss và cập nhật lại mô hình Discriminator. Sau khi mô hình Discriminator được cập nhật thông số kết quả nhận diện real và fake của ảnh giả sẽ được tính toán thành G-fake-loss vì mục tiêu của mô hình Generator muốn mô hình mục tiêu ban đầu nhận diện ảnh giả này là nhãn 1 (real). Kết hợp perturb-loss, adversarial loss, G-fake-loss thành G-loss và cập nhật lại mô hình Generator. Việc huấn luyện một vòng như vậy lặp đi lặp lại cho đến khi số lượng ảnh đầu vào hết thì kết thúc 1 epoch. Mô hình huấn luyện liên tục 600 epochs.

Các bước thực hiện huấn luyện và xây dựng mô hình được thể hiện qua mã giả sau:

```

Input:  $x$ 
Output: AE (Adversarial example)
 $T = \text{Total images} / 128$ 
 $iter = 1;$ 
while ( $iter < T$ ) do
     $pert = \text{Generator}(iter):$ 
        1. Encoder( $iter$ );
        2. Bottleneck(Encoder);
        3. Decoder(Bottleneck);
     $AE = x_{adv} = x \oplus (k \times pert) ;$ 
    if ( $\text{Discriminator}(AE, x) == 0$ ) then
        Update Discriminator
    end
    if ( $\text{Discriminator}(x, AE) == 1$ ) then
        Update Discriminator
    end
end
end

```

Mã nguồn huấn luyện mô hình:

```

if __name__ == "__main__":
    use_cuda = True
    image_nc = 3
    batch_size = 128

    # Define what device we are using
    print("CUDA Available: ", torch.cuda.is_available())
    device = torch.device("cuda" if (use_cuda and
    torch.cuda.is_available()) else "cpu")

    cifar_dataset = torchvision.datasets.CIFAR10('./dataset', train=True,
    transform=transforms.ToTensor(), download=True)

    train_dataloader = DataLoader(cifar_dataset, batch_size=batch_size,
    shuffle=False, num_workers=1)

    # training the target model
    target_model = CIFAR_target_net().to(device)
    target_model.train()

    opt_model = torch.optim.Adam(target_model.parameters(), lr=0.001)
    epochs = 80
    for epoch in range(epochs):
        loss_epoch = 0
        if epoch == 40:
            opt_model = torch.optim.Adam(target_model.parameters(),
            lr=0.0001)

        for i, data in enumerate(train_dataloader, 0):
            train_imgs, train_labels = data
            train_imgs, train_labels = train_imgs.to(device),
            train_labels.to(device)

            logits_model = target_model(train_imgs)
            loss_model = F.cross_entropy(logits_model, train_labels)
            loss_epoch += loss_model
            opt_model.zero_grad()
            loss_model.backward()
            opt_model.step()

        print('loss in epoch %d: %f' % (epoch, loss_epoch.item()))

    # save model
    targeted_model_file_name = './CIFAR_target_model.pth'
    torch.save(target_model.state_dict(), targeted_model_file_name)

```

```

target_model.eval()

# Cifar test dataset

cifar_dataset_test = torchvision.datasets.CIFAR10('./dataset',
train=False, transform=transforms.ToTensor(), download=True)

test_dataloader = DataLoader(cifar_dataset_test,
batch_size=batch_size, shuffle=True, num_workers=1)

num_correct = 0

for i, data in enumerate(test_dataloader, 0):

    test_img, test_label = data

    test_img, test_label = test_img.to(device), test_label.to(device)

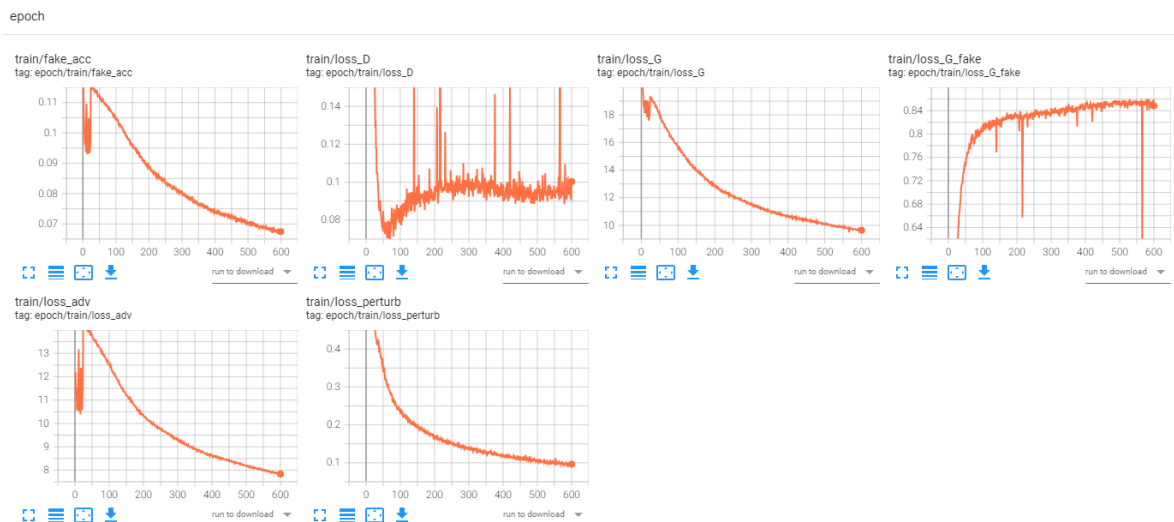
    pred_lab = torch.argmax(target_model(test_img), 1)

    num_correct += torch.sum(pred_lab==test_label,0)

print('accuracy in testing
set: %f\n'%(num_correct.item()/len(cifar_dataset_test)))

```

Kết quả huấn luyện của mô hình resnet56:



Hình 3.4. Đồ thị biểu diễn quá trình huấn luyện mô hình

Kết quả tensorboard cho thấy mô hình đã đi đúng mục tiêu là giảm các giá trị loss tuy nhiên còn chưa ổn định.

Thực hiện điều chỉnh mô hình, tiến hành thay đổi các thông số và huấn luyện nhiều lần để có bộ thông số tốt nhất.

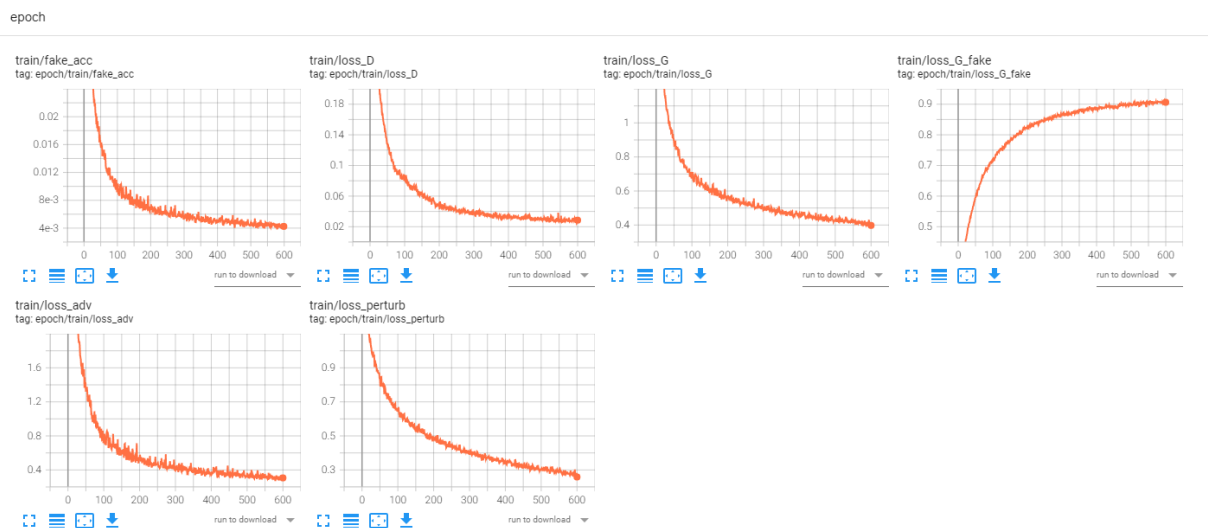
Bảng 3.1. Bảng thông số điều chỉnh mô hình

- smooth = 0.1 (do kết quả độ chính xác của mô hình resnet56 là 94.37% nên ta xem mỗi bức ảnh đưa vào

sẽ có 90% là nhãn đúng, ví dụ bức ảnh có nhãn là con mèo thì chỉ có 90% là con mèo)

- $lr = 1e - 4$ (giảm initialize optimizers)
- Thay đổi các thông số $\lambda = 0.2$, $\alpha = 0.2$, $\beta = 0.6$. Do mục tiêu là ta giảm G-loss mà công thức tính $loss_G = \lambda * loss_{adv} + \alpha * loss_{G_fake} + \beta * loss_{perturb}$. Nên qua quá trình thực hiện huấn luyện nhiều lần ta chọn bộ thông số này là tối ưu G-loss nhất.
- $C_TRESH = 0.5$ (ngưỡng hỗn loạn của mô hình)

Sau khi thực hiện điều chỉnh đã thu được mô hình mong muốn.



Hình 3.5. Đồ thị biểu diễn mô hình sau khi tuning

Thực hiện huấn luyện và điều chỉnh cho từng mô hình. Sau đó từ kết quả huấn luyện ta tính độ chính xác mô hình bằng $1 - \text{fake_acc}$ lần lượt như sau:

Bảng 3.2. Bảng kết quả huấn luyện mô hình

Model	Epoch	Độ chính xác	Thời gian huấn luyện
resnet56	600	99.58%	19 giờ 12 phút 30s
vgg19_bn	600	98.69%	21 giờ 18 phút 43s
mobilenetv2	600	98.84%	16 giờ 21 phút 37s
shufflenetv2	600	99.22%	28 giờ 49 phút 37s
repvgg_a2	600	98.78%	24 giờ 50 phút 30s

Từ bảng kết quả huấn luyện cho thấy muốn thực hiện tấn công một mô hình học sâu bằng phương pháp sử dụng ảnh tấn công đối kháng phải mất rất nhiều thời gian. Chỉ riêng thời gian huấn luyện ở ảnh có kích cỡ 32x32 thì đã mất hàng chục giờ để tạo được mô hình tạo mặt nạ nhiễu.

3.3. Thử nghiệm tạo mẫu đối kháng và đánh giá mô hình tạo mặt nạ nhiễu

Kịch bản thử nghiệm: Sử dụng mô hình Generator đã được huấn luyện ở 3.2 để sinh ra mặt nạ nhiễu sau đó nhân với tỷ lệ K (K chạy từ 0.1 đến 1.0 tương ứng là tỷ lệ mặt nạ nhiễu) sau đó cộng với ảnh ban đầu để cho ra kết quả ảnh giả. Dùng ảnh giả được tạo ra đưa vào mô hình phân lớp ban đầu để kiểm tra kết quả ảnh giả có đánh lừa được mô hình phân lớp hay không. Nếu ảnh giả có kết quả phân lớp khác với ảnh ban đầu thì nâng giá trị As lên 1. Quá trình lặp đi lặp lại đến hết bộ ảnh thử nghiệm. Sau đó tiếp tục hiện hành thay đổi tỷ lệ K để thử nghiệm.

Các chỉ số sử dụng trong thử nghiệm:

Gt: số ảnh trong tập dữ liệu dùng để làm đầu vào thử nghiệm

As: số ảnh đã được thay đổi nhãn thành nhãn mục tiêu thành công.

Công thức đánh giá như sau:

$$Adv = \frac{As}{gt} (* 100\%)$$

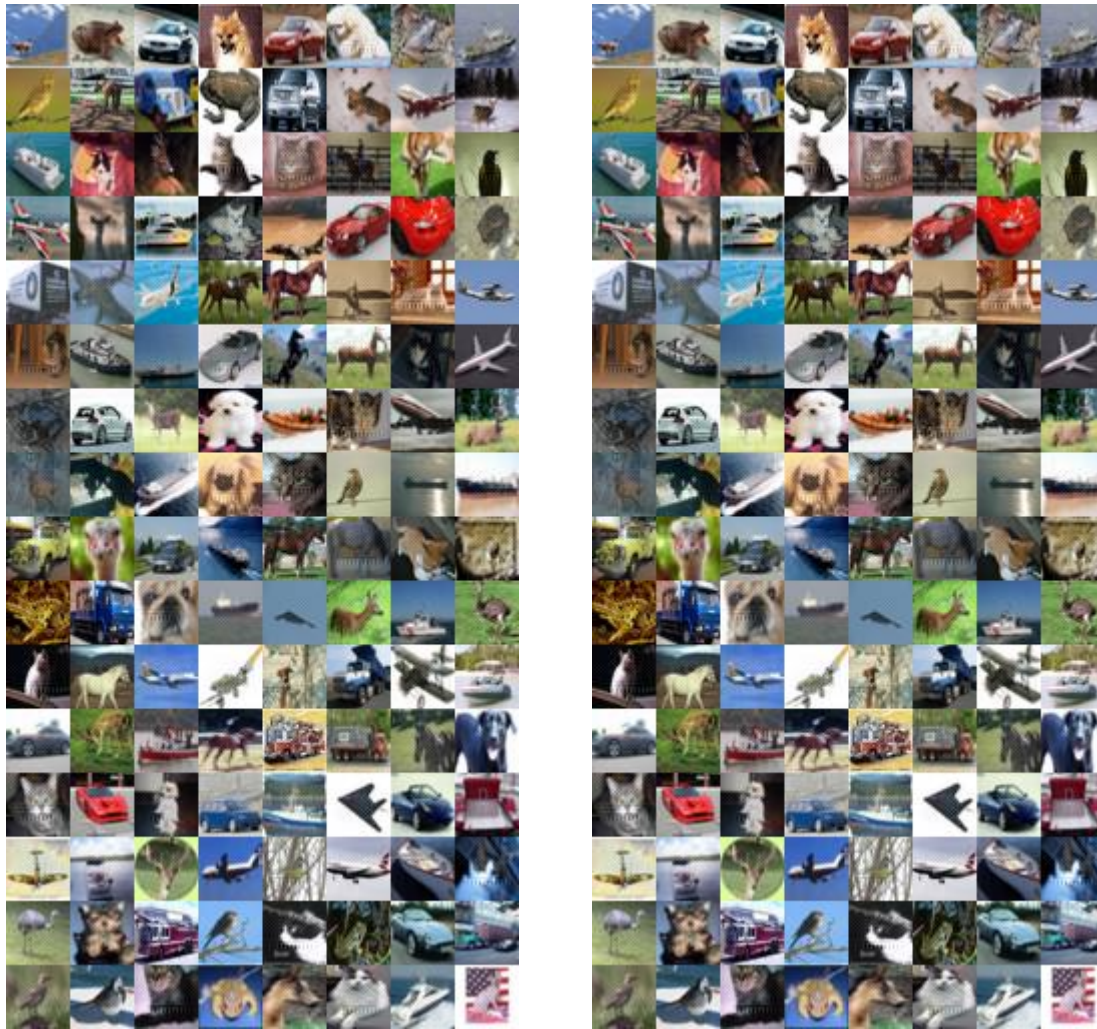
Mục tiêu của thử nghiệm: Chọn được tỷ lệ k tốt nhất để sinh ra ảnh giả và sinh ra ảnh giả.

3.3.1. Thử nghiệm với mô hình repvgg_a2

Bảng 3.2. Kết quả thử nghiệm với mô hình repvgg_a2

Model	K	Gt	As	Adv	Time	TB/adv s	SSIM	PSNR	L_0	L_1	L_2
repvgg_a2	0.1	4261	97	2.28%	151	1.56	0.9999999995	46.90	0.0012	0.000013	0.00000000047
repvgg_a2	0.2	4261	350	8.21%	147	0.42	0.9999999992	45.72	0.0015	0.000019	0.00000000080
repvgg_a2	0.3	4261	732	17.18%	167	0.23	0.9999999988	44.88	0.0018	0.000023	0.00000000118
repvgg_a2	0.4	4261	1210	28.40%	175	0.14	0.9999999984	44.21	0.0022	0.000028	0.00000000163
repvgg_a2	0.5	4261	1774	41.63%	154	0.09	0.9999999978	43.63	0.0025	0.000032	0.00000000216
repvgg_a2	0.6	4261	2319	54.42%	159	0.07	0.9999999972	43.12	0.0028	0.000036	0.00000000277
repvgg_a2	0.7	4261	2812	65.99%	167	0.06	0.9999999965	42.66	0.0031	0.000041	0.00000000348
repvgg_a2	0.8	4261	3212	75.38%	170	0.05	0.9999999957	42.22	0.0035	0.000045	0.00000000431
repvgg_a2	0.9	4261	3504	82.23%	171	0.05	0.9999999948	41.83	0.0038	0.000049	0.00000000522
repvgg_a2	1	4261	3707	87.00%	174	0.05	0.9999999938	41.47	0.0042	0.000054	0.00000000624

Sau quá trình thử nghiệm thì luận văn chọn tỷ lệ K từ khoảng 0.1 đến 1 để thử nghiệm và cho thấy nếu K càng lớn thì tỷ lệ Adv càng cao nhưng ảnh giả sẽ khác biệt so với ảnh ban đầu minh chứng thông qua kết quả PSNR ngày càng thấp đi. Ngược lại K càng bé thì Adv càng thấp nhưng gần như ảnh ban đầu không khác biệt gì so với ảnh giả minh chứng thông qua kết quả PSNR ngày càng thấp đi. Đồng thời SSIM của từng trường hợp K từ 0.1 đến 1 rất gần 1 cho thấy ảnh giả được tạo ra rất giống ảnh thật do SSIM rất gần 1. Quan sát thêm các chỉ số L_0 L_1 L_2 cũng rất thấp cho thấy sự thay đổi của hai ảnh là không đáng kể. Cuối cùng luận văn cũng chọn được giá trị K lý tưởng là 0.7 thỏa mãn kỳ vọng gần 50% ảnh giả được tạo ra từ tập dữ liệu.



Hình 3.6. Ảnh mẫu thực nghiệm trên mô hình repvgg_a2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra

3.3.2. Thử nghiệm với mô hình shufflenetv2

Bảng 3.3. Kết quả thử nghiệm với mô hình shufflenetv2

Model	K	Gt	As	Adv	Time	TB/ adv s	SSIM	PSNR	L_0	L_1	L_2
shufflenetv2	0.1	3072	42	1.37%	134	3.19	0.9999999995	46.75	0.0012	0.000014	0.00000000050
shufflenetv2	0.2	3072	123	4.00%	143	1.16	0.9999999991	45.58	0.0016	0.000020	0.00000000086
shufflenetv2	0.3	3072	246	8.01%	149	0.61	0.9999999988	44.86	0.0019	0.000024	0.00000000117
shufflenetv2	0.4	3072	399	12.99%	147	0.37	0.9999999984	44.22	0.0021	0.000028	0.00000000156
shufflenetv2	0.5	3072	577	18.78%	146	0.25	0.9999999980	43.63	0.0025	0.000033	0.00000000205
shufflenetv2	0.6	3072	826	26.89%	151	0.18	0.9999999973	3.08	0.0028	0.000037	0.00000000267
shufflenetv2	0.7	3072	1118	36.39%	148	0.13	0.9999999966	42.61	0.0031	0.000042	0.00000000335
shufflenetv2	0.8	3072	1401	45.61%	159	0.11	0.9999999959	42.17	0.0035	0.000046	0.00000000411
shufflenetv2	0.9	3072	1681	54.72%	159	0.09	0.9999999950	41.78	0.0038	0.000051	0.00000000495
shufflenetv2	1	3072	1900	61.85%	158	0.08	0.9999999942	41.44	0.0041	0.000055	0.00000000583

Sau quá trình thử nghiệm thì luận văn chọn tỷ lệ K từ khoảng 0.1 đến 1 để thử nghiệm và cho thấy nếu K càng lớn thì tỷ lệ Adv càng cao nhưng ảnh giả sẽ khác biệt so với ảnh ban đầu minh chứng thông qua kết quả PSNR ngày càng thấp đi. Ngược lại K càng bé thì Adv càng thấp nhưng gần như ảnh ban đầu không khác biệt gì so với ảnh giả minh chứng thông qua kết quả PSNR ngày càng thấp đi. Đồng thời SSIM của từng trường hợp K từ 0.1 đến 1 rất gần 1 cho thấy ảnh giả được tạo ra rất giống ảnh thật do SSIM rất gần 1. Quan sát thêm các chỉ số L_0 L_1 L_2 cũng rất thấp cho thấy sự thay đổi của hai ảnh là không đáng kể. Cuối cùng luận văn cũng chọn được giá trị K lý tưởng là 0.7 thỏa mãn kỳ vọng gần 50% ảnh giả được tạo ra từ tập dữ liệu.



Hình 3.7. Ảnh mẫu thực nghiệm trên mô hình shufflenetv2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra

3.3.3. Thử nghiệm với mô hình mobilenetv2

Bảng 3.4. Kết quả thử nghiệm với mô hình mobilenetv2

Model	K	Gt	As	Adv	Time	TB/adv s	SSIM	PSNR	L_0	L_1	L_2
mobilenetv2	0.1	3691	65	1.76%	165	2.54	0.999999992	47.20	0.0011	0.000012	0.00000000041
mobilenetv2	0.2	3691	237	6.42%	184	0.78	0.999999985	45.76	0.0015	0.000019	0.00000000075
mobilenetv2	0.3	3691	548	14.85%	163	0.30	0.999999979	45.02	0.0018	0.000024	0.00000000105
mobilenetv2	0.4	3691	940	25.47%	174	0.19	0.999999973	44.48	0.0020	0.000027	0.00000000034
mobilenetv2	0.5	3691	1422	38.53%	187	0.13	0.999999966	44.00	0.0022	0.000031	0.00000000169
mobilenetv2	0.6	3691	1875	50.80%	199	0.11	0.999999958	43.54	0.0025	0.000034	0.00000000210
mobilenetv2	0.7	3691	2322	62.91%	190	0.08	0.999999949	43.12	0.0028	0.000038	0.00000000256
mobilenetv2	0.8	3691	2641	71.55%	205	0.08	0.999999939	42.74	0.0030	0.000042	0.00000000006
mobilenetv2	0.9	3691	2869	77.73%	230	0.08	0.999999928	42.41	0.0032	0.000046	0.00000000358
mobilenetv2	1	3691	3060	82.90%	226	0.07	0.999999917	42.09	0.0035	0.000049	0.00000000417

Sau quá trình thử nghiệm thì luận văn chọn tỷ lệ K từ khoảng 0.1 đến 1 để thử nghiệm và cho thấy nếu K càng lớn thì tỷ lệ Adv càng cao nhưng ảnh giả sẽ khác biệt so với ảnh ban đầu minh chứng thông qua kết quả PSNR ngày càng thấp đi. Ngược lại K càng bé thì Adv càng thấp nhưng gần như ảnh ban đầu không khác biệt gì so với ảnh giả minh chứng thông qua kết quả PSNR ngày càng thấp đi. Đồng thời SSIM của từng trường hợp K từ 0.1 đến 1 rất gần 1 cho thấy ảnh giả được tạo ra rất giống ảnh thật do SSIM rất gần 1. Quan sát thêm các chỉ số L_0 L_1 L_2 cũng rất thấp cho thấy sự thay đổi của hai ảnh là không đáng kể. Cuối cùng luận văn cũng chọn được giá trị K lý tưởng là 0.7 thỏa mãn kỳ vọng gần 50% ảnh giả được tạo ra từ tập dữ liệu.



Hình 3.8. Ảnh mẫu thực nghiệm trên mô hình mobilenetv2, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra

3.3.4. Thử nghiệm với mô hình vgg19_bn

Bảng 3.5. Kết quả thử nghiệm với mô hình vgg19_bn

Model	K	Gt	As	Adv	Time	TB/adv s	SSIM	PSNR	L ₀	L ₁	L ₂
vgg19_bn	0.1	2886	45	1.56%	110	2.44	0.9999999994	46.37	0.0013	0.000015	0.00000000059
vgg19_bn	0.2	2886	131	4.54%	109	0.83	0.9999999988	44.65	0.0019	0.000025	0.00000000123
vgg19_bn	0.3	2886	260	9.01%	139	0.53	0.9999999981	43.69	0.0024	0.000031	0.00000000192
vgg19_bn	0.4	2886	443	15.35%	131	0.30	0.9999999972	42.90	0.0029	0.000037	0.00000000277
vgg19_bn	0.5	2886	693	24.01%	103	0.15	0.9999999963	42.29	0.0033	0.000042	0.00000000370
vgg19_bn	0.6	2886	1027	35.59%	110	0.11	0.9999999952	41.72	0.0038	0.000048	0.00000000482
vgg19_bn	0.7	2886	1365	47.30%	106	0.08	0.9999999939	41.21	0.0043	0.000053	0.00000000611
vgg19_bn	0.8	2886	1668	57.80%	138	0.08	0.9999999925	40.77	0.0047	0.000058	0.00000000752
vgg19_bn	0.9	2886	1920	66.53%	120	0.06	0.9999999910	40.38	0.0052	0.000062	0.00000000903
vgg19_bn	1	2886	2129	73.77%	120	0.06	0.9999999893	40.02	0.0056	0.000067	0.00000001069

Sau quá trình thử nghiệm thì luận văn chọn tỷ lệ K từ khoảng 0.1 đến 1 để thử nghiệm và cho thấy nếu K càng lớn thì tỷ lệ Adv càng cao nhưng ảnh giả sẽ khác biệt so với ảnh ban đầu minh chứng thông qua kết quả PSNR ngày càng thấp đi. Ngược lại K càng bé thì Adv càng thấp nhưng gần như ảnh ban đầu không khác biệt gì so với ảnh giả minh chứng thông qua kết quả PSNR ngày càng thấp đi. Đồng thời SSIM của từng trường hợp K từ 0.1 đến 1 rất gần 1 cho thấy ảnh giả được tạo ra rất giống ảnh thật do SSIM rất gần 1. Quan sát thêm các chỉ số L_0 L_1 L_2 cũng rất thấp cho thấy sự thay đổi của hai ảnh là không đáng kể. Cuối cùng luận văn cũng chọn được giá trị K lý tưởng là 0.7 thỏa mãn kỳ vọng gần 50% ảnh giả được tạo ra từ tập dữ liệu.



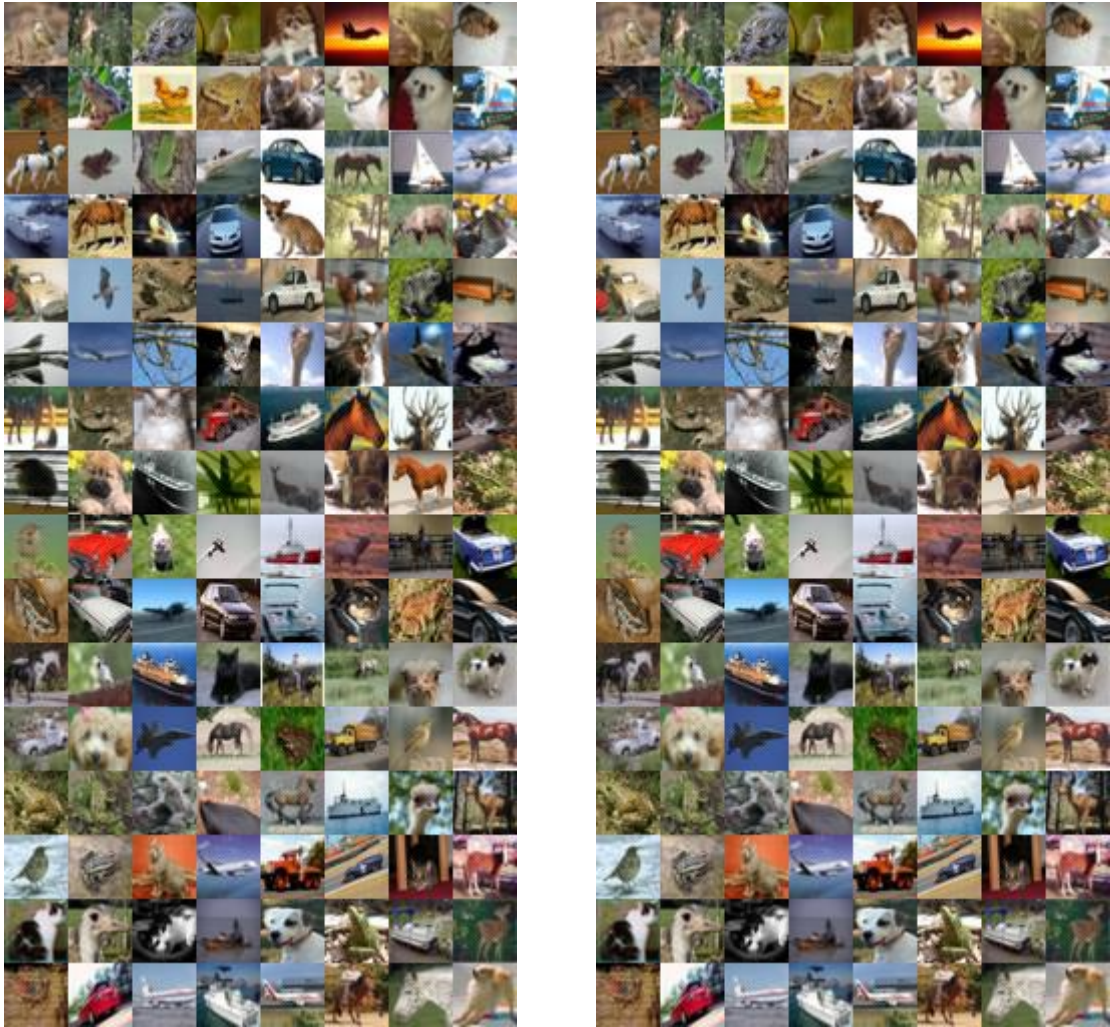
Hình 3.9. Ảnh mẫu thực nghiệm trên mô hình vgg19_bn, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra

3.3.5. Thử nghiệm với mô hình resnet56

Bảng 3.6. Kết quả thử nghiệm với mô hình resnet56

Model	K	Gt	As	Adv	Time s	TB/adv s	SSIM	PSNR	L_0	L_1	L_2
resnet56	0.1	5222	160	3.06%	252	1.58	0.999999999	47.44	0.0011	0.000012	0.0000000004
resnet56	0.2	5222	485	9.29%	258	0.53	0.999999998	45.61	0.0015	0.000020	0.0000000008
resnet56	0.3	5222	931	17.83%	273	0.29	0.999999998	44.97	0.0018	0.000024	0.0000000011
resnet56	0.4	5222	1588	30.41%	269	0.17	0.999999997	44.42	0.0020	0.000028	0.0000000014
resnet56	0.5	5222	2346	44.93%	286	0.12	0.999999996	43.90	0.0023	0.000031	0.0000000018
resnet56	0.6	5222	3087	59.12%	301	0.10	0.999999995	43.40	0.0026	0.000036	0.0000000023
resnet56	0.7	5222	3691	70.68%	289	0.08	0.999999994	42.96	0.0029	0.000040	0.0000000029
resnet56	0.8	5222	4153	79.53%	327	0.08	0.999999993	42.56	0.0032	0.000044	0.0000000035
resnet56	0.9	5222	4452	85.25%	365	0.08	0.999999992	42.21	0.0034	0.000048	0.0000000041
resnet56	1	5222	4666	89.35%	412	0.09	0.999999990	41.88	0.0037	0.000051	0.0000000048

Sau quá trình thử nghiệm thì luận văn chọn tỷ lệ K từ khoảng 0.1 đến 1 để thử nghiệm và cho thấy nếu K càng lớn thì tỷ lệ Adv càng cao nhưng ảnh giả sẽ khác biệt so với ảnh ban đầu minh chứng thông qua kết quả PSNR ngày càng thấp đi. Ngược lại K càng bé thì Adv càng thấp nhưng gần như ảnh ban đầu không khác biệt gì so với ảnh giả minh chứng thông qua kết quả PSNR ngày càng thấp đi. Đồng thời SSIM của từng trường hợp K từ 0.1 đến 1 rất gần 1 cho thấy ảnh giả được tạo ra rất giống ảnh thật do SSIM rất gần 1. Quan sát thêm các chỉ số L_0 L_1 L_2 cũng rất thấp cho thấy sự thay đổi của hai ảnh là không đáng kể. Cuối cùng luận văn cũng chọn được giá trị K lý tưởng là 0.7 thỏa mãn kỳ vọng gần 50% ảnh giả được tạo ra từ tập dữ liệu.



Hình 3.10. Ảnh mẫu thực nghiệm trên mô hình resnet56, bên trái là ảnh ban đầu, bên phải là ảnh giả được tạo ra

3.4. Thử nghiệm tấn công đối kháng

Kịch bản thử nghiệm: Sau khi thử nghiệm ở phần 3.3 luận văn đã chọn được K để tạo ảnh giả cho từng model phân lớp mục tiêu và thực hiện thử nghiệm tấn công đối kháng vào mô hình mục tiêu.

Với từng lớp đối tượng của dataset cifar 10 luận văn tiến hành phân loại mẫu đối kháng theo từng lớp. Và thực hiện đưa cả ảnh giả và ảnh thật vào model phân lớp mục tiêu để lấy kết quả phân lớp. Nếu kết quả phân lớp của ảnh giả khác với kết quả phân lớp của ảnh thật thì đánh dấu ảnh giả đó được tạo thành công và đạt được mục tiêu thay đổi kết quả phân lớp.

Mục tiêu của thử nghiệm: Là để đánh giá lại ảnh giả được tạo bởi thử nghiệm ở phần 3.3 có thực sự thay đổi được kết quả phân lớp hay không.

3.4.1. Thử nghiệm tấn công đối kháng vào mô hình repvgg_a2


Với mô hình gốc phân loại bộ dữ liệu cifar10 là repvgg_a2 ta tiến hành thử nghiệm tấn công đối kháng với 2812 ảnh giả ta thu được kết quả sau:

Bảng 3.7. Bảng kết quả tấn công đối kháng vào mô hình repvgg_a2

	Adv	Success	Tỷ lệ
airplanes	396	396	100%
cars	226	226	100%
birds	124	124	100%
cats	274	274	100%
deer	240	240	100%
dogs	323	323	100%
frogs	151	151	100%
horses	279	279	100%
ships	558	558	100%
trucks	241	241	100%
Tổng	2812	2812	100%

Như vậy với 2812 ảnh giả được tạo ra không có ảnh nào thất bại khi tấn công làm sai lệch nhận dạng của mô hình repvgg_a2. Dưới đây là bảng trích xuất mẫu của từng lớp trong bộ dữ liệu.

Bảng 3.8. Bảng trích xuất mẫu đối kháng với mô hình repvgg_a2

Nhãn gốc	Ảnh gốc	Mặt nạ nhiễu	Ảnh giả	Nhãn mục tiêu
airplaine				cats
ships				cats
dogs				cats
trucks				cats
cars				frogs
frogs				cats
horses				dogs
deers				cats
birds				cats
cats				dogs

3.4.2. Thử nghiệm tấn công đối kháng vào mô hình shufflenetv2









Với mô hình gốc phân loại bộ dữ liệu cifar10 là shufflenetv2 ta tiến hành thử nghiệm tấn công đối kháng với 1118 ảnh giả ta thu được kết quả sau:

Bảng 3.9. Bảng kết quả tấn công đối kháng vào mô hình shufflenetv2

	Adv	Success	Tỷ lệ
airplanes	121	121	100%
cars	48	48	100%
birds	50	50	100%
cats	146	146	100%
deer	109	109	100%
dogs	99	99	100%
frogs	34	34	100%
horses	124	124	100%
ships	342	342	100%
trucks	45	45	100%
Tổng	1118	1118	100%

Như vậy với 1118 ảnh giả được tạo ra không có ảnh nào thất bại khi tấn công làm sai lệch nhận dạng của mô hình shufflenetv2. Dưới đây là bảng trích xuất mẫu của từng lớp trong bộ dữ liệu.

Bảng 3.10. Bảng trích xuất mẫu đối kháng với mô hình shufflenetv2

Nhãn gốc	Ảnh gốc	Mặt nạ nhiễu	Ảnh giả	Nhãn mục tiêu
airplaine				dogs
ships				dogs
dogs				dogs
trucks				dogs
cars				frogs
frogs				cats
horses				dogs
deers				frogs
birds				dogs
cats				frogs

3.4.3. Thử nghiệm tấn công đối kháng vào mô hình mobilenetv2














Với mô hình gốc phân loại bộ dữ liệu cifar10 là mobilenetv2 ta tiến hành thử nghiệm tấn công đối kháng với 2322 ảnh giả ta thu được kết quả sau:

Bảng 3.11. Bảng kết quả tấn công đối kháng vào mô hình mobilenetv2

	Adv	Success	Tỷ lệ
airplanes	352	352	100%
cars	138	138	100%
birds	45	45	100%
cats	171	171	100%
deer	61	61	100%
dogs	625	625	100%
frogs	183	183	100%
horses	113	113	100%
ships	458	458	100%
trucks	176	176	100%
Tổng	2322	2322	100%

Như vậy với 2322 ảnh giả được tạo ra không có ảnh nào thất bại khi tấn công làm sai lệch nhận dạng của mô hình mobilenetv2. Dưới đây là bảng trích xuất mẫu của từng lớp trong bộ dữ liệu.

Bảng 3.12. Bảng trích xuất mẫu đối kháng với mô hình mobilenetv2

Nhãn gốc	Ảnh gốc	Mặt nạ nhiễu	Ảnh giả	Nhãn mục tiêu
airplaine				frogs
ships				cats
dogs				dogs
trucks				cats
cars				frogs
frogs				cats
horses				dogs
deers				cats
birds				cats
cats				dogs

3.4.4. Thử nghiệm tấn công đối kháng vào mô hình vgg19_bn













Với mô hình gốc phân loại bộ dữ liệu cifar10 là vgg19_bn ta tiến hành thử nghiệm tấn công đối kháng với 2886 ảnh giả ta thu được kết quả sau:

Bảng 3.13. Bảng kết quả tấn công đối kháng vào mô hình vgg19_bn

	Adv	Success	Tỷ lệ
airplanes	111	111	100%
cars	36	36	100%
birds	50	50	100%
cats	348	348	100%
deer	115	115	100%
dogs	197	197	100%
frogs	35	35	100%
horses	78	78	100%
ships	380	380	100%
trucks	15	15	100%
Tổng	2886	2886	100%

Như vậy với 2886 ảnh giả được tạo ra không có ảnh nào thất bại khi tấn công làm sai lệch nhận dạng của mô hình vgg19_bn. Dưới đây là bảng trích xuất mẫu của từng lớp trong bộ dữ liệu.

Bảng 3.14. Bảng trích xuất mẫu đối kháng với mô hình vgg19_bn

Nhãn gốc	Ảnh gốc	Mặt nạ nhiễu	Ảnh giả	Nhãn mục tiêu
airplaine				cats
ships				cats
dogs				cats
trucks				dogs
cars				frogs
frogs				cats
horses				dogs
deers				cats
birds				cats
cats				dogs

3.4.5. Thử nghiệm tấn công đối kháng vào mô hình resnet56


Với mô hình gốc phân loại bộ dữ liệu cifar10 là resnet56 ta tiến hành thử nghiệm tấn công đối kháng với 3691 ảnh giả ta thu được kết quả sau:

Bảng 3.15. Bảng kết quả tấn công đối kháng vào mô hình resnet56

	Adv	Success	Tỷ lệ
airplanes	512	512	100%
cars	325	325	100%
birds	320	320	100%
cats	391	391	100%
deer	361	361	100%
dogs	423	423	100%
frogs	253	253	100%
horses	355	355	100%
ships	504	504	100%
trucks	247	247	100%
Tổng	3691	3691	100%

Như vậy với 3691 ảnh giả được tạo ra không có ảnh nào thất bại khi tấn công làm sai lệch nhận dạng của mô hình resnet56. Dưới đây là bảng trích xuất mẫu của từng lớp trong bộ dữ liệu.

Bảng 3.16. Bảng trích xuất mẫu đối kháng với mô hình resnet56

Nhãn gốc	Ảnh gốc	Mặt nạ nhiễu	Ảnh giả	Nhãn mục tiêu
airplaine				cats
ships				cats
dogs				cats
trucks				cats
cars				frogs
frogs				cats
horses				dogs
deers				cats
birds				cats
cats				dogs

3.5. Kết luận chương 3

Trong chương 3 luận văn đã huấn luyện thành công mô hình và thực hiện tạo ảnh giả thành công. Với tỷ lệ ảnh giả tấn công đối thành công vào mô hình mục tiêu tương đối tốt và đạt sự kỳ vọng gần 50% bộ dữ liệu thử nghiệm.

Bảng 3.17. Bảng tổng kết kết quả thử nghiệm tấn công đối kháng

MODEL	K	GT	ADV	TỶ LỆ	THỜI GIAN	SSIM	PSNR
repvgg_a2	0.7	4261	2812	65.99%	167	0.9999999965	42.66
shufflenetv2	0.7	3072	1118	36.39%	148	0.9999999966	42.61
mobilenetv2	0.7	3691	2332	62.91%	190	0.9999999949	43.12
vgg19_bn	0.7	2886	1365	47.30%	106	0.9999999939	41.21
resnet56	0.7	5222	3691	70.68%	289	0.999999994	42.96

Từ kết quả trên cho thấy luận văn đã thành công trong việc nghiên cứu xây dựng, thử nghiệm mô hình tạo mẫu đối kháng (deepfakes) với tỷ lệ tấn công hợp thành công cao. Cao hơn kết quả tối đa của các bài báo công bố gần đây như AIGAN [20] tỷ lệ Adv cao nhất của họ trên tập dữ liệu cifar10 là 10.17% và tương tự trong bài báo cũng trích dẫn các kết quả cao nhất của FSRM là 5.76%, C&W là 8.74%, PGD là 9.22%, AdvGAN là 10.19%.

Ngoài ra luận văn cũng công bố toàn bộ mã nguồn sử dụng và phát triển tại github: https://github.com/congthanh96/Deepfake_Adversarial

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Luận văn đã nghiên cứu trình bày tổng quan về học máy, tấn công đối kháng, các phương pháp tấn công đối kháng.

Luận văn nghiên cứu trình bày tổng quan về deepfakes, kỹ thuật ứng dụng deepfakes trong tấn công đối kháng.

Luận văn đã đề xuất được phương pháp tạo mẫu đối kháng (deepfakes) và thực hiện huấn luyện mô hình Generator và Discriminator thành công. Thực hiện điều chỉnh mô hình bằng cách thay đổi các thông số và huấn luyện nhiều lần để tạo ra mô hình tốt nhất.

Trong luận văn này đã đề xuất được mô hình tạo mẫu đối kháng theo lý thuyết mạng sinh đối kháng và đã thực hiện huấn luyện hai mô hình thành phần cấu tạo chính nên GAN là Generator và Discriminator thành công. Thực hiện điều chỉnh mô hình bằng cách thay đổi các thông số và huấn luyện nhiều lần để tạo ra mô hình tốt nhất. Luận văn cũng thực hiện thử nghiệm thành công tạo mẫu đối kháng trên tập dữ liệu cifar10 và tấn công đối kháng vào các model mục tiêu như repvgg_a2, shufflenetv2, mobilenetv2, vgg19_bn, resnet56 với tỷ lệ trung bình 45% so với tập dữ liệu thử nghiệm. Cao hơn so với tỷ lệ tạo mẫu đối kháng thành công của một số phương pháp khác AIGAN là 10.17% (Cao hơn ~ 34%), FSRM là 5.76% (Cao hơn ~ 34%), C&W là 8.74% (Cao hơn ~ 36%), PGD là 9.22% (Cao hơn ~ 35%), AdvGAN là 10.19% (Cao hơn ~ 34%).

Kết quả thực nghiệm cho thấy, nghiên cứu đã cải thiện và phát triển AIGAN để tạo ra mẫu đối kháng hiệu quả nhất. Tuy nhiên việc chọn epochs để kết thúc huấn luyện và chọn model còn chưa đạt kết quả như mong đợi do số thực hiện huấn luyện còn ít do hạn chế về mặt thời gian và số lượng thực nghiệm trên nhiều mô hình. Mong muốn trong thời gian tới sẽ thực hiện huấn luyện với số epochs lớn, với bộ dữ liệu có kích cỡ hình ảnh lớn hơn. Hướng phát triển trong tương lai là thực hiện trên bộ dữ liệu ảnh có kích cỡ cao hơn như 64x64, 256x256, 512x512, ... và thực hiện huấn luyện mô hình với thời gian dài và thử nghiệm với nhiều thông số khác để đưa ra mô hình sinh dữ liệu đối kháng đạt hiệu quả cao hơn kết quả trước đó.

TÀI LIỆU THAM KHẢO

- [1] Atencia-Linares, P., Artiga (2022), “Deepfakes, shallow epistemic graves”, *Doi: 10.1007/s11229-022-04003-3*.
- [2] A. Kurakin, I. Goodfellow, and S. Bengio (2016), “Adversarial examples in the physical world”, *arXiv:1607.02533*.
- [3] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille (2017), “Adversarial examples for semantic segmentation and object detection”, *International Conference on Computer Vision IEEE*.
- [4] D. Castelvechi (2016), “Can we open the black box of AI?”, *Nature News*, vol. 538, no. 7623, p. 20.
- [5] Eran Segalis, Eran Galili (2022), “OGAN: Disrupting Deepfakes with an Adversarial Attack that Survives Training”, *Doi:10.48550/arXiv.2006.12247*.
- [6] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu (2017), “Dolphinattack: Inaudible voice commands,” *arXiv:1708.09537*.
- [7] Haoxuan Qiu, Yanhui Du and Tianliang Lu (2022), “The Framework of Cross-Domain and Model Adversarial Attack against Deepfake”, *Doi:10.3390/fi14020046*.
- [8] Huang, H., Wang, Y., Chen, Z., Zhang, Y., Li, Y., Tang, Z., Chu, W., Chen, J., Lin, W., & Ma, K.-K (2022), “CMUA-Watermark: A Cross-Model Universal Adversarial Watermark for Combating Deepfakes”, *AAAI Conference on Artificial Intelligence*, 36(1), 989-997, *Doi: 10.1609/aaai.v36i1.19982*.
- [9] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song (2017), “Robust physical-world attacks on deep learning models,” *arXiv:1707.08945*.
- [10] Jan Kietzmann, Linda W. Lee, Ian P. McCarthy, Tim C. Kietzmann (2022), “Deepfakes: Trick or treat?”, *Doi: 10.1016/j.bushor.2019.11.006*.
- [11] Longqing Ye (2022), “AugShuffleNet: Communicate More, Compute Less”, *arXiv:2203.06589*.

- [12] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen (2018), “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, *arXiv:1801.04381*.
- [13] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, Florian Tramèr (2021), “Membership Inference Attacks From First Principles”, *arXiv:2112.03570v2*.
- [14] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, (2016), “Hidden voice commands,” *USENIX Security Symposium*, pp. 513–530.
- [15] Paarth Neekhara, Shehzeen Hussain, Malhar Jere, Farinaz Koushanfar, Julian McAuley (2019), "Adversarial Deepfakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples", *Doi: 10.13140/RG.2.2.26227.48168*.
- [16] P. W. Koh and P. Liang, (2017), “Understanding black-box predictions via influence functions,” *International Conference on Machine Learning*.
- [17] R. Shwartz-Ziv and N. Tishby (2017), “Opening the black box of deep neural networks via information,” *arXiv:1703.00810*.
- [18] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra (2016), “Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization,” *arXiv:1610.02391*.
- [19] Shehzeen Hussain, Paarth Neekhara, Malhar Jere, Farinaz Koushanfar, Julian McAuley (2021), “Adversarial Deepfakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples”, *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3348-3357.
- [20] Tao Bai, Jun Zhao, Jinlin Zhu, Shoudong Han, Jiefeng Chen, Bo Li, Alex Kot (2021), “AI-GAN: ATTACK-INSPIRED GENERATION OF ADVERSARIAL EXAMPLES”, *arXiv:2002.02196v*.

- [21] Umme Sara1, Morium Akter, Mohammad Shorif Uddin (2019), “Image Quality Assessment through FSIM, SSIM,MSE and PSNR—A Comparative Study”, *Doi:10.4236/jcc.2019.73002*.
- [22] W. Knight, (2017), “The dark secret at the heart of ai,” *MIT Technology Review*.
- [23] Wei Deng, Qi Feng, Liyao Gao, Faming Liang, Guang Lin (2020), “Non-convex Learning via Replica Exchange Stochastic Gradient MCMC”, *ICML*.
- [24] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, Jian Sun (2021), “RepVGG: Making VGG-style ConvNets Great Again”, *arXiv:2101.03697v1*.
- [25] Ying Xu, Kiran Raja, Marius Pedersen (2022), “Supervised Contrastive Learning for Generalizable and Explainable DeepFakes Detection”, *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, pp. 379-389.
- [26] Z. C. Lipton, (2016), “The mythos of model interpretability,” *International Conference on Machine Learning (ICML) Workshop*.

Hồ Chí Minh, ngày 18 tháng 09 năm 2023

CÁN BỘ HƯỚNG DẪN

(Ký, ghi rõ họ tên)

HỌC VIÊN

(Ký, ghi rõ họ tên)

TS. Phạm Duy Trung TS. Nguyễn Trọng Khánh

Nguyễn Công Thạnh

Xác nhận của Trưởng tiểu ban

BẢN XÁC NHẬN CHỈNH SỬA LUẬN VĂN

Họ và tên học viên: NGUYỄN CÔNG THẠNH

Ngày sinh: 12/12/1996 Nơi sinh: Bình Định

Ngành/chuyên ngành: An toàn thông tin Khóa: 01

Tôi đã bảo vệ luận văn với đề tài: “NGHIÊN CỨU KỸ THUẬT DEEPPAKES
ỨNG DỤNG TRONG TẤN CÔNG ĐỐI KHÁNG”

tại Hội đồng chấm luận văn ngày 23 tháng 09 năm 2023

Tôi đã chỉnh sửa nội dung luận văn thạc sĩ với đề tài trên theo góp ý, yêu cầu của Hội đồng. Cụ thể như sau:

- Chỉnh sửa lại bố cục luận văn, các lý thuyết về đánh giá đưa về chương 2. Chương 3 thay đổi thêm mã giả, mã nguồn của luận văn.
- Làm rõ tấn công đối kháng, mô hình học sâu, cải tiến trong luận văn, đóng góp của luận văn.

Nay tôi xin báo cáo đã hoàn thành chỉnh sửa luận văn như trên và đề nghị Hội đồng chấm luận văn, cán bộ hướng dẫn xác nhận.

Hà Nội, ngày 01 tháng 10 năm 2023.

CÁN BỘ HƯỚNG DẪN

HỌC VIÊN

TS. Phạm Duy Trung

TS. Nguyễn Trọng Khánh

THƯ KÝ

Nguyễn Công Thành

CHỦ TỊCH HỘI ĐỒNG

TS. Nguyễn Mạnh Thắng

PGS TS. Lương Thế Dũng