ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
BÁO CÁO ĐỒ ÁN MÔN HỌC
CÔNG NGHỆ DỮ LIỆU LỚN

ĐỀ TÀI: PHÁT HIỆN HÌNH ẢNH ĐƯỢC TẠO BỞI AI

GVHD: TS. Hà Minh Tân
Nhóm 16
22521663 - Lê Hoàng Việt
22521669 - Hứa Quang Vinh
22521601 - Đoàn Ngọc Quốc Tuấn
22621625 - Nguyễn Hiếu Thiên Tường

LỜI MỞ ĐẦU

Trong kỷ nguyên bùng nổ của trí tuệ nhân tạo (AI), sự phát triển vượt bậc của các mô hình tạo sinh như Generative Adversarial Networks (GANs), Diffusion Models hay các mô hình nền lớn (foundation models) đã mở ra nhiều khả năng sáng tạo chưa từng có. Một trong những ứng dụng nổi bật của công nghệ này là khả năng tạo ra hình ảnh giả – những bức ảnh chân thực đến mức khó phân biệt với ảnh chụp từ thế giới thực. Những hình ảnh này có thể được sử dụng trong nhiều lĩnh vực như nghệ thuật số, quảng cáo, điện ảnh, thời trang, thậm chí trong y tế và nghiên cứu khoa học. Tuy nhiên, song hành với lợi ích là những thách thức nghiêm trọng về đạo đức, an toàn thông tin và đặc biệt là nguy cơ gây ra sự hiểu lầm, thao túng nhận thức trong cộng đồng.

Việc một người bình thường khó có thể nhận biết đâu là hình ảnh thật, đâu là hình ảnh được tạo bởi máy tính, đặt ra yêu cầu cấp thiết đối với việc nghiên cứu và phát triển các phương pháp phát hiện hình ảnh giả. Khả năng nhận diện chính xác các hình ảnh do AI tạo ra không chỉ góp phần đảm bảo tính minh bạch và đáng tin cậy của thông tin trên môi trường số mà còn hỗ trợ hiệu quả cho các hệ thống kiểm duyệt nội dung, điều tra pháp y số, và bảo vệ bản quyền nội dung hình ảnh.

Xuất phát từ thực tiễn đó, đề tài "Phát hiện hình ảnh được tạo ra bởi AI" được thực hiện nhằm nghiên cứu các đặc trưng khác biệt giữa hình ảnh thật và hình ảnh do AI sinh ra, từ đó xây dựng và đánh giá các phương pháp nhận diện tự động. Đề tài không chỉ giúp nâng cao nhận thức về những nguy cơ của công nghệ tạo sinh mà còn đóng góp một phần vào nỗ lực xây dựng không gian mạng an toàn, lành mạnh và đáng tin cậy trong thời đại số.

MỤC LỤC

DANH MỤC KÝ HIỆU, BẢNG VIẾT TẮT

Ký hiệu	Tiếng Anh	Diễn giải
AI	Artificial Intelligence	Trí tuệ nhân tạo
GAN	Generative Adversarial Network	Mạng đối kháng sinh mẫu
DFDC	DeepFake Detection Challenge	Thách thức phát hiện DeepFake
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
RGB	Red Green Blue	3 kênh màu trong ảnh
MBConv	Mobile Inverted Bottleneck Convolution	Khối tích chập dạng chai cổ ngược dành cho thiết bị di động
CFG Scale	Classifier-Free Guidance Scale	Thông số điều chỉnh sự tương thích giữa prompt và hình ảnh trong diffusion models
DPM SDE++ Karras	Denoising Probabilistic Model Stochastic Differential Equation++	Phương pháp khuếch tán tăng chất lượng ảnh do Karras đề xuất
Euler A	Euler Ancestral sampler	Phương pháp lấy mẫu trong diffusion models
CLIP	Contrastive Language-Image Pretraining	Mô hình ngôn ngữ-hình ảnh đối chiếu
XAI	Explainable Artificial Intelligence	Trí tuệ nhân tạo có khả năng giải thích
DF	Deepfake	Ảnh/video giả mạo do AI tạo ra
GPU	Graphics Processing Unit	Bộ xử lý đồ họa
CPU	Central Processing Unit	Bộ xử lý trung tâm

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1. Tính cấp thiết của đề tài

Sự phát triển vượt bậc của trí tuệ nhân tạo, đặc biệt là các mô hình tạo sinh (generative models) như GAN (Generative Adversarial Networks), Diffusion Models hay các mô hình như DALL·E, Midjourney, Stable Diffusion đã mở ra một kỷ nguyên mới về khả năng sáng tạo nội dung. Trong đó, hình ảnh do AI tạo ra ngày càng trở nên chân thực và khó phân biệt với hình ảnh thật, gây ra mối lo ngại lớn về tính xác thực của thông tin hình ảnh trên không gian mạng.

Với sự phổ biến của mạng xã hội và các nền tảng chia sẻ nội dung, các hình ảnh giả mạo có thể dễ dàng lan truyền và bị lợi dụng vào mục đích sai lệch như tung tin giả, bôi nhọ danh dự, giả mạo danh tính, thậm chí thao túng dư luận. Trong lĩnh vực pháp lý, báo chí, giáo dục hay y tế, việc sử dụng và xác thực hình ảnh là vô cùng quan trọng. Do đó, nhu cầu về các công cụ và kỹ thuật giúp phát hiện hình ảnh được tạo ra bởi AI ngày càng trở nên cấp thiết, không chỉ để phục vụ nghiên cứu học thuật mà còn nhằm đảm bảo an ninh thông tin, đạo đức số và bảo vệ người dùng trên không gian mạng.

1.2. Mục tiêu nghiên cứu

Đề tài được thực hiện nhằm hướng đến các mục tiêu chính sau:

- Nghiên cứu tổng quan về các phương pháp tạo sinh hình ảnh bằng AI, đặc biệt là GANs và các mô hình tiên tiến hiện nay.

- Phân tích đặc điểm hình ảnh do AI tạo ra, từ đó xác định các tín hiệu bất thường hoặc dấu vết khác biệt so với hình ảnh thật.

- Tìm hiểu, đề xuất và xây dựng mô hình phát hiện hình ảnh giả dựa trên học máy hoặc học sâu.

- Đánh giá hiệu quả mô hình qua các tập dữ liệu chuẩn, từ đó rút ra những ưu nhược điểm và hướng phát triển tiếp theo.

1.3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: Các hình ảnh được tạo ra bởi các mô hình AI tạo sinh (chẳng hạn như GAN, diffusion models) và các kỹ thuật phát hiện hình ảnh giả mạo.

Phạm vi nghiên cứu:

- Tập trung vào hình ảnh tĩnh (không xét đến video hoặc âm thanh).
- Không đi sâu vào phương pháp phát hiện ở cấp độ phần cứng (ví dụ như cảm biến máy ảnh), mà chỉ phân tích và xử lý dữ liệu hình ảnh đã có sẵn.
- Chủ yếu sử dụng các mô hình học máy và học sâu trong xử lý ảnh.

1.4. Phương pháp nghiên cứu

Đề tài sử dụng các phương pháp nghiên cứu sau:

- Nghiên cứu tài liệu: Tìm hiểu các nghiên cứu liên quan, bài báo khoa học, công trình học thuật về mô hình tạo sinh hình ảnh và phương pháp phát hiện hình ảnh giả.
- Phân tích – so sánh: So sánh đặc trưng giữa hình ảnh thật và hình ảnh AI để rút ra các điểm khác biệt có thể khai thác trong quá trình phát hiện.
- Thực nghiệm: Xây dựng mô hình phát hiện hình ảnh giả sử dụng các thuật toán học máy/học sâu, huấn luyện và đánh giá hiệu quả trên tập dữ liệu chuẩn.
- Đánh giá mô hình: Sử dụng các chỉ số đánh giá phổ biến như Accuracy, Precision, Recall, F1-score để đo lường hiệu quả phát hiện.

1.5. Tổng quan đề tài

Đề tài được triển khai với các nội dung chính sau đây:

- Chương 1: Tổng quan đề tài: Trình bày tính cấp thiết, mục tiêu, phạm vi, đối tượng và phương pháp nghiên cứu cũng như nội dung tổng thể của đề tài.
- Chương 2: Cơ sở lý thuyết và tổng quan nghiên cứu: Giới thiệu các kiến thức nền tảng về trí tuệ nhân tạo, mô hình tạo sinh hình ảnh (GAN,

Diffusion Models), cùng các kỹ thuật phát hiện hình ảnh giả đã được nghiên cứu trong và ngoài nước.

- Chương 3: Phân tích đặc điểm và dữ liệu nghiên cứu: Phân tích các đặc điểm nhận dạng hình ảnh do AI tạo ra so với ảnh thật, đồng thời trình bày nguồn dữ liệu và bộ dữ liệu chuẩn được sử dụng trong đề tài.

- Chương 4: Xây dựng mô hình phát hiện hình ảnh giả: Đề xuất và triển khai các mô hình học máy, học sâu nhằm phát hiện hình ảnh giả, bao gồm thiết kế kiến trúc, huấn luyện và tối ưu mô hình.

- Chương 5: Thực nghiệm và đánh giá kết quả: Thực hiện thử nghiệm mô hình trên tập dữ liệu thực tế, đánh giá hiệu quả dựa trên các chỉ số đo lường, đồng thời so sánh với các phương pháp hiện có.

- Chương 6: Kết luận và hướng phát triển: Tổng kết kết quả nghiên cứu, những đóng góp chính của đề tài và đề xuất hướng phát triển trong tương lai.

CHƯƠNG 2: TỔNG QUAN TÀI LIỆU VÀ NGHIÊN CỨU CÓ LIÊN QUAN

Tổng quan tài liệu và nghiên cứu liên quan đóng vai trò quan trọng trong việc xây dựng nền tảng kiến thức, nhận diện những tiến triển cũng như những thách thức trong lĩnh vực phát hiện hình ảnh được tạo ra bởi AI (AI-generated image detection). Chương này tập trung tổng hợp, phân tích các nghiên cứu trong và ngoài nước, cũng như hệ thống hóa các phương pháp, kỹ thuật đã được áp dụng nhằm phục vụ việc phát hiện hình ảnh giả mạo do các mô hình AI sinh ra.

2.1 Nghiên cứu ngoài nước

Phát hiện hình ảnh giả mạo do AI tạo ra đã trở thành một vấn đề nổi bật trong khoa học máy tính và an ninh mạng, đặc biệt khi các mô hình sinh ảnh GAN ngày càng tinh vi, có khả năng tạo ra những hình ảnh gần như không thể phân biệt với ảnh thật.

2.1.1 Các phương pháp phát hiện chính

- Mô hình học sâu (Deep Learning Models): Các nghiên cứu điển hình như Wang et al. (2020) đã áp dụng các kiến trúc CNN tiên tiến như ResNet, EfficientNet nhằm trích xuất đặc trưng sâu và phân loại ảnh thật - giả với độ chính xác cao. Ngoài ra, XceptionNet cũng thường được dùng trong các cuộc thi phát hiện deepfake do hiệu quả trích xuất đặc trưng tốt trên dữ liệu ảnh.

- Phân tích Artifact và Noise Patterns: FakeSpotter (2019) giới thiệu phương pháp phân tích các artifact đặc trưng chỉ xuất hiện trong ảnh giả do AI tạo ra như các mẫu nhiễu không tự nhiên, không nhất quán về ánh sáng, texture, hoặc các đường biên bất thường. Các phương pháp dựa trên phân tích miền tần số (frequency domain analysis), như biến đổi Fourier, Wavelet cũng giúp phát hiện các khác biệt tinh vi.

- Phương pháp kết hợp (Hybrid Methods): Một số nghiên cứu kết hợp trích xuất đặc trưng truyền thống như noise residual, các đặc trưng thống kê với mạng neural nhằm tăng hiệu quả phát hiện. Ví dụ, các kỹ thuật kết

hợp learning-based feature extraction với handcrafted features được đề xuất nhằm khắc phục hạn chế của từng phương pháp riêng lẻ.

2.1.2 Bộ dữ liệu tiêu chuẩn

Nhiều bộ dữ liệu lớn được sử dụng để huấn luyện và đánh giá các mô hình phát hiện ảnh giả, bao gồm:

- CelebA-HQ và FFHQ: Dữ liệu khuôn mặt chất lượng cao dùng để sinh ảnh GAN.
- DeepFake Detection Challenge Dataset (DFDC): Bộ dữ liệu lớn đa dạng ảnh thật và deepfake dùng trong các cuộc thi phát hiện deepfake.
- GAN-generated datasets: Ảnh sinh bởi StyleGAN, BigGAN, ProGAN... được dùng làm dữ liệu huấn luyện mô hình phát hiện.

2.1.3 Thách thức và hạn chế

- Sự đa dạng và phức tạp của ảnh giả: Khi công nghệ tạo ảnh giả ngày càng tinh vi, mô hình phát hiện phải liên tục cập nhật để bắt kịp.
- Khó khăn trong tổng quát hóa mô hình: Mô hình huấn luyện trên bộ dữ liệu nhất định thường gặp khó khi áp dụng với ảnh giả từ các mô hình GAN khác hoặc điều kiện thực tế khác.
- Yêu cầu về dữ liệu lớn: Để đạt hiệu suất tốt, mô hình thường cần bộ dữ liệu lớn và đa dạng, điều này gây khó khăn cho nghiên cứu và triển khai thực tế.

2.2 Nghiên cứu trong nước

Ở Việt Nam, nghiên cứu về phát hiện hình ảnh giả do AI tạo ra còn khá mới mẻ, tuy nhiên một số đề tài đồ án và nghiên cứu đã tập trung vào:

- Xây dựng bộ dữ liệu ảnh thật và giả nhằm phục vụ huấn luyện mô hình.

- Ứng dụng các kiến trúc CNN đơn giản (như LeNet, AlexNet, ResNet phiên bản nhẹ) để phân loại ảnh.

- Nghiên cứu phân tích nhiễu (noise analysis) và texture đặc trưng nhằm phát hiện ảnh giả, kết hợp các bộ lọc hình ảnh cơ bản với kỹ thuật học máy.

Một số đề tài còn hướng tới phát triển hệ thống phát hiện ảnh giả để hỗ trợ kiểm duyệt nội dung mạng xã hội và chống tin giả, bảo vệ thông tin cá nhân.

Tuy nhiên, các nghiên cứu trong nước thường chưa đi sâu vào các kỹ thuật tiên tiến như phân tích miền tần số, học sâu với kiến trúc phức tạp hay các kỹ thuật tăng cường dữ liệu chuyên biệt cho bài toán này.

2.3 Đề xuất nghiên cứu

- Kết hợp phân tích artifact với mô hình học sâu tối ưu: Giúp tận dụng ưu điểm của cả hai hướng, tăng độ chính xác và giảm thời gian huấn luyện.

- Tăng cường dữ liệu (Data Augmentation) đặc thù: Sử dụng kỹ thuật tăng cường như xoay, cắt, thêm nhiễu, biến đổi ánh sáng để nâng cao khả năng tổng quát của mô hình.

- Phân tích đặc trưng miền tần số: Áp dụng Fourier Transform hoặc Wavelet để phát hiện các biến đổi đặc trưng của ảnh giả mà mắt thường khó nhận biết.

- Tối ưu kiến trúc mạng: Lựa chọn các kiến trúc CNN nhẹ, hiệu quả, phù hợp với yêu cầu thực tế về tài nguyên tính toán và tốc độ xử lý.

- Xây dựng bộ dữ liệu đa dạng: Kết hợp nhiều nguồn ảnh giả từ các mô hình GAN khác nhau để mô hình học sâu có khả năng tổng quát tốt hơn.

2.4 Xu hướng phát triển và ứng dụng

- Phát hiện ảnh giả đa phương tiện: Không chỉ ảnh tĩnh, mà cả video deepfake cũng là hướng nghiên cứu quan trọng.

- Ứng dụng thực tiễn: Kiểm duyệt nội dung mạng xã hội, phòng chống tin giả, bảo vệ quyền riêng tư và an ninh quốc gia.

- Phát triển mô hình nhẹ, có thể triển khai trên thiết bị di động: Phục vụ nhu cầu phát hiện nhanh, ở quy mô lớn.

- Kết hợp các mô hình Explainable AI (XAI): Giúp giải thích các quyết định phân loại, tăng tính tin cậy.

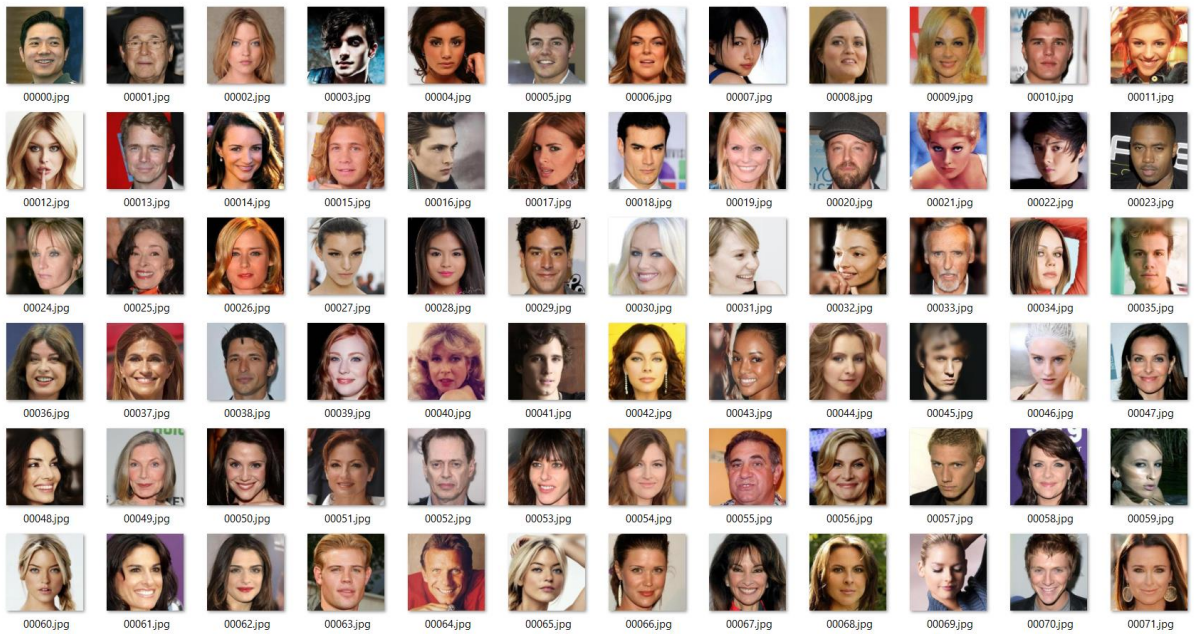
CHƯƠNG 3: NỘI DUNG NGHIÊN CỨU VÀ PHƯƠNG PHÁP LUẬN

Nội dung nghiên cứu và phương pháp luận đóng vai trò trung tâm của đề tài, chú trọng vào việc trình bày chi tiết về cách mà nghiên cứu được thực hiện và phương pháp được sử dụng để đạt được mục tiêu đề ra. Chương này không chỉ làm rõ chi tiết về dữ liệu và các tài nguyên phương pháp tiền xử lý dữ liệu và lý thuyết của mô hình thực nghiệm, mà còn giúp độc giả hiểu rõ cách tiếp cận vấn đề nghiên cứu.

3.1 Mô tả dữ liệu

Dữ liệu nghiên cứu trong đề tài bao gồm các bộ ảnh thật và ảnh được tạo ra bởi các mô hình AI hiện đại như GAN, Diffusion Models.

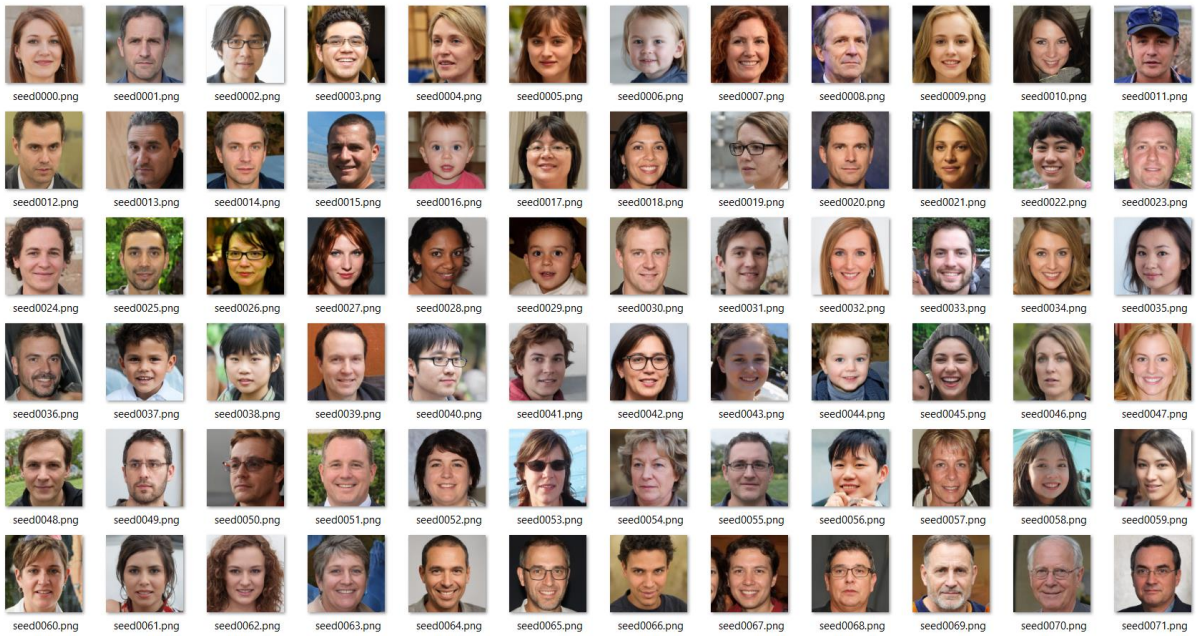
Hình ảnh thật được lấy từ bộ dữ liệu CelebA-HQ: bộ dữ liệu thuộc tính khuôn mặt quy mô lớn với hơn 30000 hình ảnh người nổi tiếng với độ phân giải là 1024x1024.Các hình ảnh trong bộ dữ liệu này bao gồm các biến thể tư thế lớn và sự lộn xộn của nền. CelebA-HQ có sự đa dạng lớn, số lượng lớn và chú thích phong phú. Ở đây chúng tôi đã sử dụng 10000 ảnh của bộ dữ liệu này cho việc huấn luyện mô hình.



Hình 3.1: Bộ dữ liệu CelebA-HQ

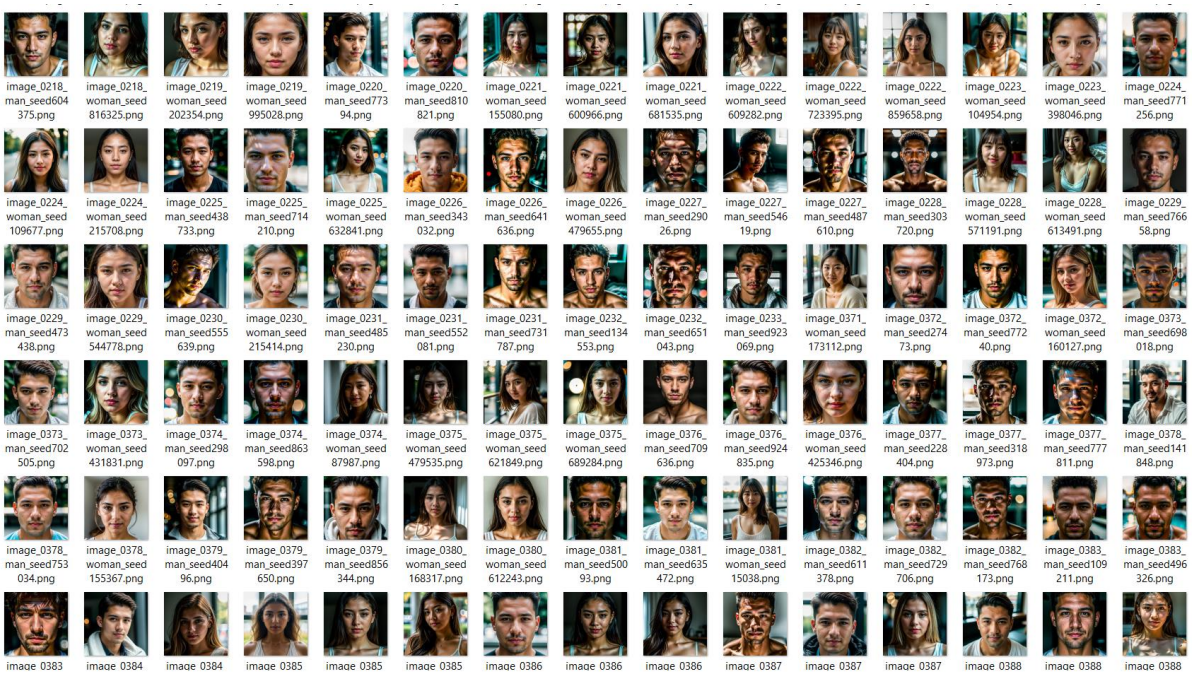
Hình ảnh giả được lấy từ việc sử dụng các mô hình mạng tạo sinh phổ biến như StyleGAN, Stable Diffusion. Ở đây chúng tôi đã sử dụng:

- 5000 hình ảnh được tạo bởi StyleGan2: Chúng tôi đã sử dụng mô hình StyleGan2 để tạo ra 5000 hình ảnh với seed từ 1-5000.



Hình 3.2: Bộ dữ liệu được tạo bởi mô hình StyleGan2

- 1000 hình ảnh được tạo bởi mô hình Realistic Vision V6.0 B1: Được xem như là biến thể hoặc phiên bản cải tiến của mô hình Stable Diffusion. Realistic Vision là một mô hình chuyên biệt hóa trong việc tạo ra hình ảnh có tính chân thực cao. Mô hình này tập trung vào việc mô phỏng các chi tiết thực tế như ánh sáng, bóng đổ, và kết cấu bề mặt vật thể. Vì mục tiêu của chúng tôi là phân biệt ảnh thật và ảnh giả cho nên yếu tố chân thực là một yếu tố đáng xem xét, do đó nhóm đã quyết định chọn Realistic Vision thay cho mô hình Stable Diffusion truyền thống.



Hình 3.3: Bộ dữ liệu được tạo bởi mô hình Realistic Vision V6.0 B1

Để có thể đạt được các hình ảnh như hình trên, nhóm chúng tôi đã sử dụng các phương pháp dùng để sinh hình ảnh từ mô hình diffusion, trong đó có:

- DPM SDE++ Karras: phương pháp khuếch tán sử dụng các cải tiến để tăng chất lượng hình ảnh, đặc biệt là giúp giảm nhiễu và giữ chi tiết.
- Euler A: phương pháp để sinh hình ảnh, có thể tạo ra kết quả mượt mà và chi tiết hơn so với một số phương pháp khác.

CFG Scale là thông số giúp mô hình điều chỉnh độ tương thích giữa prompt (lời nhắc mô tả hình ảnh) và hình ảnh cuối cùng. Càng tăng giá trị CFG Scale, mô hình sẽ càng cố gắng tuân thủ chặt chẽ các yêu cầu của prompt, và ngược lại, giá trị thấp hơn sẽ cho phép mô hình có nhiều sáng tạo và "đột biến" hơn trong quá trình sinh hình ảnh. Ở đây nhóm chúng tôi thiết lập CFG Scale từ 3.5 đến 7.

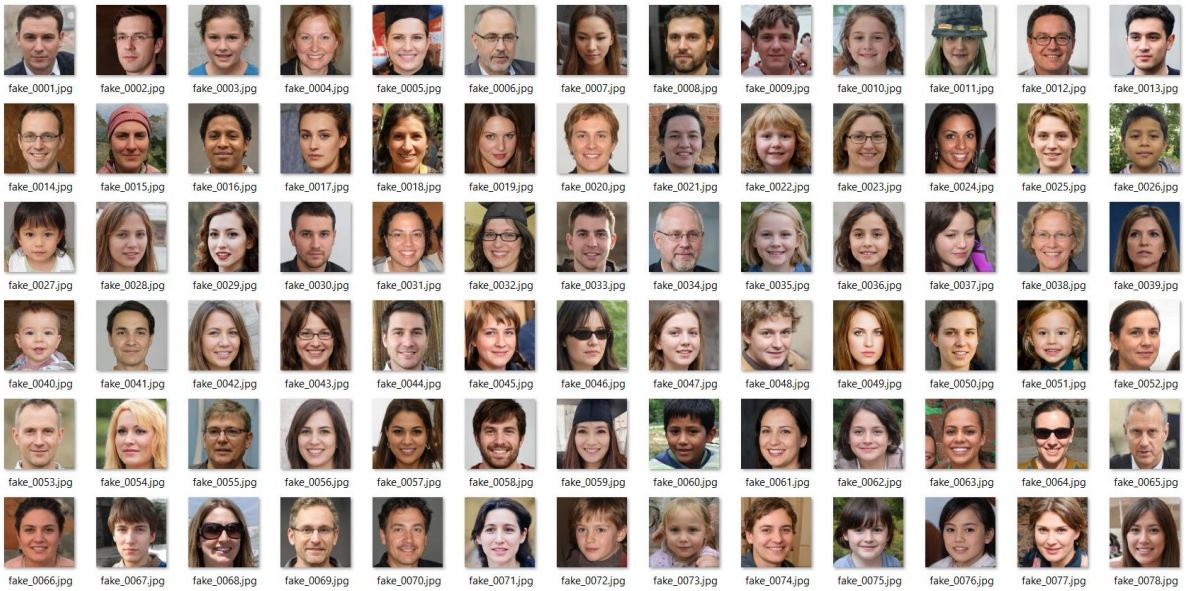
Về prompt (lời nhắc), chúng tôi đã dùng rất nhiều prompt khác nhau cho mô hình, các prompt có điểm chung là **“RAW photo, subject, 8k uhd, dslr, soft lighting, high quality, film grain, Fujifilm XT3”** để mang lại hình ảnh chân thực nhất.
Đối với negative prompt là những yếu tố cần phải tránh, chúng tôi đã thêm rất nhiều yếu tố để mô hình có thể tạo ra ảnh chuẩn mực nhất có thể.

Negative Prompt:
"(deformed iris, deformed pupils, semi-realistic, cgi, 3d, render, sketch, cartoon, drawing, anime, mutated hands and fingers:1.4), "
"(deformed, distorted, disfigured:1.3), poorly drawn, bad anatomy, wrong anatomy, extra limb, missing limb, floating limbs, "
"disconnected limbs, mutation, mutated, ugly, disgusting, amputation, UnrealisticDream, text, watermark, cropped, out of frame, "
"worst quality, low quality, jpeg artifacts, duplicate, morbid, mutilated, extra fingers, poorly drawn hands, poorly drawn face, "
"blurry, bad proportions, extra limbs, cloned face, malformed limbs, missing arms, missing legs, extra arms, extra legs, fused fingers, "
"too many fingers, long neck"

Ngoài ra chúng tôi cũng có sử dụng thêm một vài yếu tố khác để tinh chỉnh hình ảnh như:

- Hires. fix with 4x-UltraSharp upscaler: tính năng giúp cải thiện chất lượng hình ảnh bằng cách tăng độ phân giải (upscale) hình ảnh sau khi đã tạo ra. Phương pháp này giúp tăng cường độ sắc nét và chi tiết cho hình ảnh lên tới 4 lần so với ban đầu.
- Denoising Strength (0.25-0.45): Đây là phương pháp xác định mức độ làm mờ hoặc loại bỏ nhiễu trong quá trình tạo hình ảnh. Một giá trị thấp (0.25) sẽ giữ lại nhiều chi tiết gốc, trong khi một giá trị cao (0.45) sẽ giúp loại bỏ nhiều nhiễu hơn và tạo ra hình ảnh mượt mà hơn.
- Upscale by 1.1-2.0: tăng độ phân giải của hình ảnh. Giá trị từ 1.1 đến 2.0 có nghĩa là làm tăng kích thước của hình ảnh, giúp hình ảnh sắc nét hơn mà không làm giảm chất lượng.
- Clip Skip 1-2: Điều chỉnh cách thức sử dụng mô hình CLIP (đây là một mô hình nhận dạng ngữ nghĩa) trong quá trình tạo hình ảnh. Việc sử dụng giá trị 1-2 giúp mô hình bỏ qua một số lớp của CLIP để giảm sự phức tạp và tối ưu hóa thời gian sinh hình ảnh.

- 4000 hình ảnh được lấy từ trang web ThisPersonDoesNotExist.com: Trang web sử dụng một mô hình StyleGAN do Nvidia phát triển, có khả năng tạo ra những hình ảnh cực kỳ chân thực của khuôn mặt con người. Bằng cách truy cập trực tiếp, trang web sẽ trả về kết quả của một ảnh gương mặt người được tạo bởi mô hình.



Hình 3.4: Bộ dữ liệu được lấy từ trang web ThisPersonDoesNotExist.com

Bộ dữ liệu gốc chúng tôi thu thập được bao gồm 10000 ảnh thật được lấy từ bộ dữ liệu CelebA-HQ và 10000 ảnh giả được lấy từ 3 nguồn khác nhau.

3.2 Tiền xử lý dữ liệu

Tiền xử lý dữ liệu (Data Preprocessing) là một bước quan trọng trong bất kỳ quy trình phân tích dữ liệu hay học máy. Mục đích của tiền xử lý dữ liệu là chuẩn bị và làm sạch dữ liệu thô trước khi đưa vào mô hình học máy hoặc phân tích, giúp cải thiện chất lượng dữ liệu, tăng hiệu quả của mô hình và giảm thiểu sai sót.

3.2.1 Resizing (Thay đổi kích thước ảnh)

Mô hình chúng tôi sử dụng là EfficientNetB1 yêu cầu dữ liệu đầu vào là hình ảnh có kích cỡ 240x240 do đó chúng tôi đã thay đổi kích thước tất cả các ảnh đầu vào thành kích thước 240x240 pixel. Chúng tôi đã sử dụng câu lệnh sau để thực hiện việc Thay đổi kích thước ảnh:

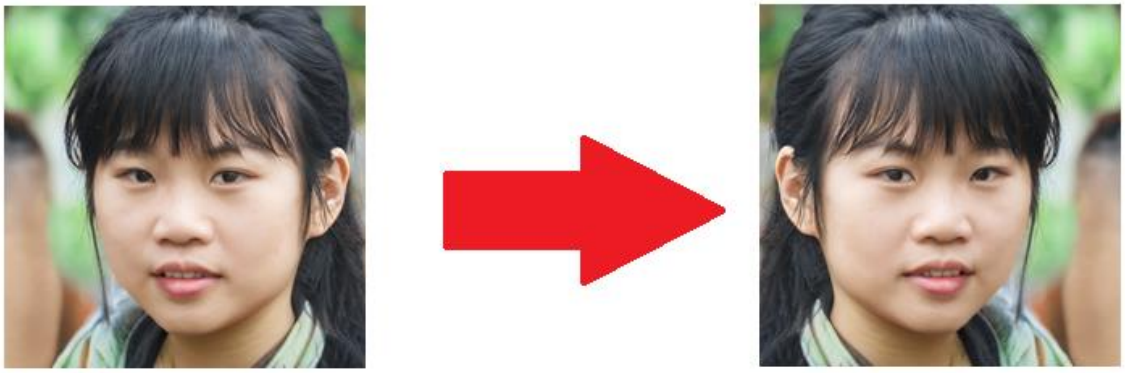
```
transforms.Resize((240, 240))
```

3.2.2 Data Argumentation (Tăng cường dữ liệu)

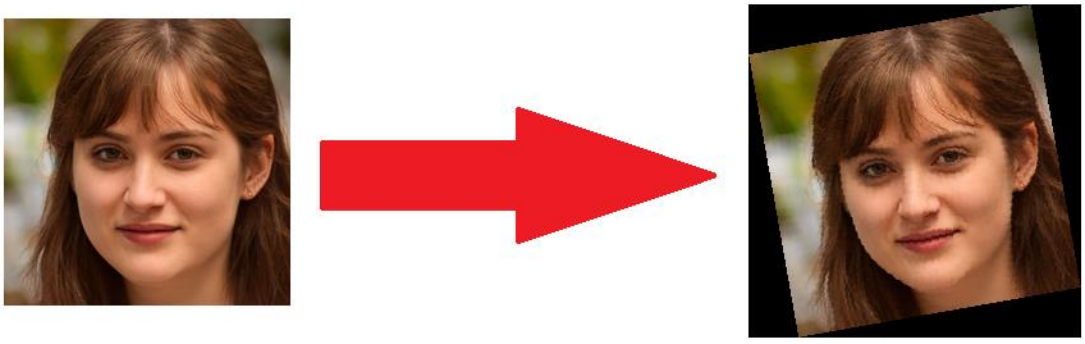
Data Augmentation là quá trình tạo ra các phiên bản mới của dữ liệu huấn luyện thông qua việc áp dụng các phép biến đổi ngẫu nhiên. Mục tiêu của augmentation là làm tăng sự đa dạng của dữ liệu huấn luyện mà không cần phải thu thập thêm dữ liệu mới.

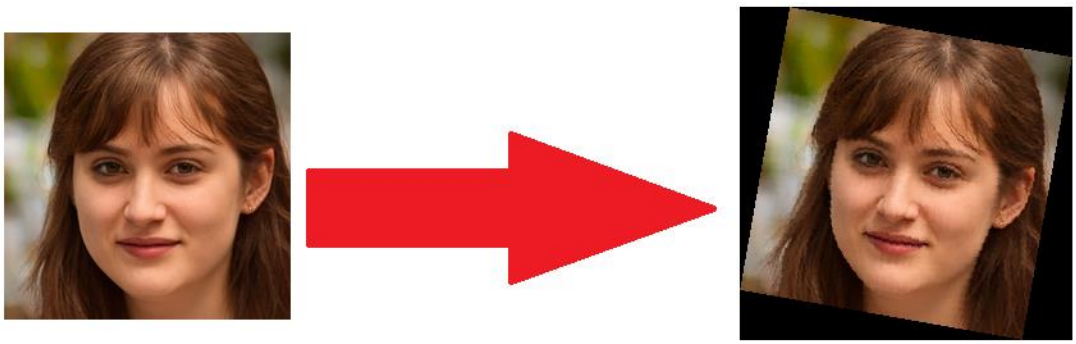
Ở đây chúng tôi đã sử dụng 2 phương pháp:

- Flipping (lật ảnh): Chúng tôi đã lật tất cả các hình ảnh theo chiều dọc. Từ đó chúng tôi đã tăng cường số dữ liệu thêm 20000 ảnh.



- Rotation (xoay ảnh): Chúng tôi đã thực hiện xoay tất cả các hình ảnh về bên trái 10 độ và bên phải 10 độ cho bộ dữ liệu gốc của chúng tôi. Từ đó chúng tôi đã tăng cường số dữ liệu thêm 40000 ảnh (20000 ảnh cho mỗi hướng).





Thông qua quá trình Data Augmentation, số lượng hình ảnh chúng tôi nhận được cho việc huấn luyện mô hình là 80000 (40000 cho mỗi loại ảnh thật và ảnh giả)

3.2.3 Chuyển đổi thành Tensor

Tensor là dạng dữ liệu mà PyTorch có thể xử lý và sử dụng trong các phép toán tensor trong quá trình huấn luyện và inference (suy luận). Dữ liệu hình ảnh được chuyển đổi có cấu trúc dạng (C, H, W), với:

- **C** là số kênh (channels), ảnh RGB sẽ có 3 kênh: Red, Green, và Blue.
- **H** là chiều cao (height) của ảnh.
- **W** là chiều rộng (width) của ảnh.

Việc chuyển đổi dữ liệu sang Tensor là cần thiết vì PyTorch yêu cầu dữ liệu đầu vào dưới dạng tensor để có thể thực hiện các phép toán trên GPU hoặc CPU. Chúng tôi đã sử dụng câu lệnh sau để thực hiện việc chuyển đổi dữ liệu sang Tensor:

```
transforms.ToTensor()
```

3.2.4 Normalization (Chuẩn hóa)

Normalization là một kỹ thuật nhằm điều chỉnh dữ liệu sao cho các giá trị có thể nằm trong một phạm vi hoặc có phân phối chuẩn, giúp mô hình học hiệu quả hơn và giảm thiểu các vấn đề về hiệu suất và tính ổn định.
Công thức của Normalize là:

$$output = \frac{input - mean}{std}$$

Chúng tôi đã tính toán được giá trị mean (giá trị trung bình) và Std (độ lệch chuẩn) của bộ dữ liệu.
Mean: [0.44557089, 0.36203397, 0.31527054]
Std: [0.12314729, 0.11055328, 0.10780927]
Để thực hiện chuẩn hóa, chúng tôi thực hiện câu lệnh sau:

```
transforms.Normalize( [0.44557089, 0.36203397, 0.31527054], [0.12314729,
```

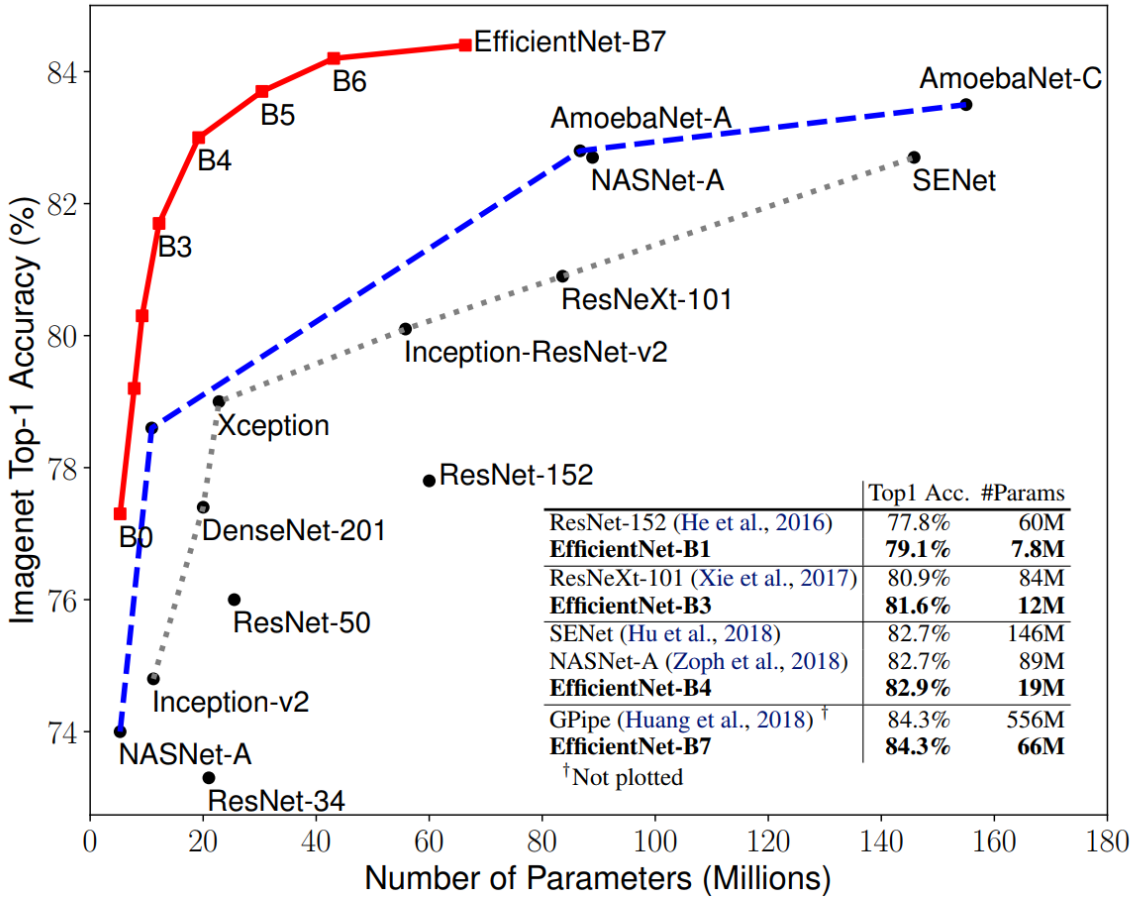
```
0.11055328,  0.10780927])
```

3.3 Mô tả mô hình

3.3.1 Giới thiệu mô hình

Mô hình chúng tôi sử dụng là mô hình EfficientNetB1. Đây là một kiến trúc mạng nơ-ron tích chập (CNN) được thiết kế để đạt hiệu suất cao trong các tác vụ phân loại ảnh, đồng thời tối ưu hóa tài nguyên tính toán. Đây là phiên bản thứ hai trong họ EfficientNet (từ B0 đến B7), được phát triển bởi Google AI.

Mô hình EfficientNetB1 được xây dựng dựa trên khái niệm gọi là "compound scaling". Khái niệm này liên quan đến việc thu phóng mô hình theo chiều sâu, chiều rộng và độ phân giải. Các mô hình như ResNet, VGG, MobileNet có áp dụng các chiến lược mở rộng mô hình, nhưng thường chỉ tập trung vào một hoặc hai yếu tố, dẫn đến việc thu phóng không đều. EfficientNet sử dụng phương pháp compound scaling để đồng thời mở rộng độ sâu, độ rộng và độ phân giải đầu vào một cách cân bằng. Điều này giúp mô hình đạt được hiệu suất cao hơn với ít tham số và chi phí tính toán thấp hơn.



Hình 3.5: Kích thước mô hình và độ chính xác ImageNet

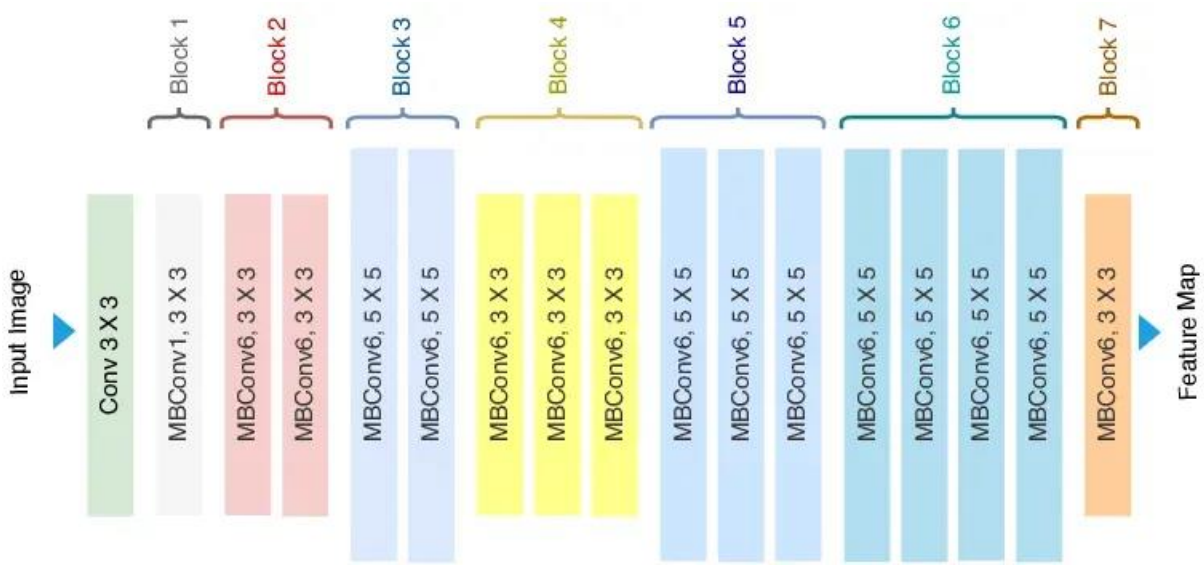
Hình trên là kết quả của một bài báo so sánh mô hình EfficientNet với một số mô hình CNN, ta có thể thấy EfficientNet vượt trội hơn đáng kể so với các mô hình khác. Đặc biệt EfficientNet-B1 nhỏ hơn 7,6 lần và nhanh hơn 5,7 lần so với ResNet-152.

Số lượng tham số của mô hình EfficientNet-B1: khoảng 7.8 triệu tham số

3.3.2 Kiến trúc mô hình

EfficientNetB1 bao gồm các khối MBConv (Mobile Inverted Bottleneck Convolution) với các đặc điểm sau:

- MBConv Blocks: Sử dụng các khối MBConv với các kích thước kernel khác nhau (3x3 và 5x5) và hệ số mở rộng (expansion factor) để tăng cường khả năng biểu diễn của mô hình.
- Số lượng lớp: Mô hình có tổng cộng 340 lớp, được tổ chức thành 7 khối chính, mỗi khối có số lượng lớp và cấu hình khác nhau để xử lý các đặc trưng ở các mức độ khác nhau.
- Kích thước đầu vào: Mô hình được thiết kế để xử lý ảnh đầu vào có kích thước 240x240 pixel, lớn hơn so với 224x224 của EfficientNetB0, giúp cải thiện khả năng nhận diện các đặc trưng chi tiết hơn trong ảnh.



Hình 3.6: Kiến trúc EfficientNetB1

Bảng 3.1: Bảng tóm tắt số lượng blocks và cấu hình chi tiết cho từng stage trong EfficientNet-B1

Mô hình EfficientNetB1 hoạt động như sau:

Stage	Loại MBConv	Kernel	Số lần lặp	Input Channels	Output Channels	Stride
1	MBConv1	3x3	1	32	16	1
2	MBConv6	3x3	2	16	24	2
3	MBConv6	5x5	2	24	40	2
4	MBConv6	3x3	3	40	80	2
5	MBConv6	5x5	3	80	112	1
6	MBConv6	5x5	4	112	192	2
7	MBConv6	3x3	1	192	320	1

Đầu tiên với input là ảnh RGB có kích thước 240 x 240 x 3. Ảnh được chuẩn hóa và chuyển sang dạng tensor có giá trị thực trong khoảng [0, 1], sau đó áp dụng hàm Normalize.

Input được đưa vào Stem Convolution. Stem Convolution là 1 lớp Convolution 3x3 với stride (số pixel mà kernel di chuyển mỗi lần trượt qua ảnh đầu vào) là 2, số kênh từ 3 trở thành 32. Mục tiêu của Stem Convolution là giảm kích thước

ảnh xuống còn 120 x 120 x 32 và trích xuất các đặc trưng sơ cấp như cạnh, góc, màu cơ bản.

Stride là một tham số quan trọng trong mạng nơ-ron tích chập (CNN), đặc biệt là trong các lớp convolution (tích chập) và pooling (lấy mẫu). Nó quy định mức độ "bước nhảy" của kernel khi trượt qua ảnh đầu vào. Giá trị Stride càng lớn, kích thước đầu vào càng nhỏ, từ đó có thể giúp tăng tốc độ tính toán.

Padding là kỹ thuật thêm các pixel (thường là 0) xung quanh biên của ảnh đầu vào trước khi áp dụng phép tích chập (convolution). Padding có nhiều loại như

- VALID: không padding gì cả — ảnh bị co lại.
- SAME: padding sao cho output cùng kích thước với input (khi stride = 1).
- Padding bằng số P cụ thể: thêm P hàng & cột 0 vào mỗi bên ảnh.

Mục đích của padding là để ta có thể kiểm soát đầu ra theo ý mình.

Đối với đầu vào có kích thước H×W, kernel có kích thước K×K, stride S, padding P. Ta có công thức sau:

$$output\ size = \left(\frac{H - K + 2P}{S} + 1\right)\left(\frac{W - K + 2P}{S} + 1\right)$$

Sử dụng công thức trên, stem convolution đã giảm kích thước ảnh xuống còn 120 x 120 x 32. Tiếp theo stem convolution thực hiện tạo feature map đầu tiên, trích xuất các đặc trưng sơ cấp như cạnh, góc, màu cơ bản. Sau đó stem convolution dùng một phương pháp là Batch Normalization để chuẩn hóa đầu ra về phân phối chuẩn giúp mô hình hội tụ nhanh hơn, ổn định hơn. Cuối cùng đầu ra được đưa qua một hàm kích hoạt là Swish activation với công thức:

$$Swish(x) = x \cdot \sigma(x) = x.\frac{1}{1 + e^{-x}}$$

Đây là hàm kích hoạt phi tuyến do Google đề xuất, và được EfficientNet sử dụng thay cho ReLU. Lý do là vì Swish là một hàm trơn mượt hơn, vẫn giữ lại

được thông tin về các giá trị âm và khả vi tại mọi điểm (tốt hơn ReLU vì không khả vi tại 0).

Đầu ra của stem convolution sẽ được đưa vào khối MBConv thông qua nhiều stage:

- Stage 1 – MBConv1 (kernel 3×3, expansion 1)
    - Cấu hình: 1 block, không mở rộng chiều.
    - Input: 120×120×32 → Output: 120×120×16.
    - Chức năng: Học các đặc trưng thô cơ bản với chi phí tính toán rất thấp.
- Stage 2 – MBConv6 ×2 (kernel 3×3, expansion 6)
    - Block 1: Stride = 2 → giảm kích thước không gian.
    - Block 2: Stride = 1.
    - Output: 60×60×24.
    - Chức năng: Mở rộng độ sâu lên 6 lần, giúp học đặc trưng không gian rộng hơn.
- Stage 3 – MBConv6 ×2 (kernel 5×5, expansion 6)
    - Block 1: Stride = 2.
    - Block 2: Stride = 1.
    - Output: 30×30×40.
    - Chức năng: Sử dụng kernel 5×5 để tăng receptive field (vùng ảnh đầu vào mà một neuron trong mạng có thể cảm nhận hoặc bị ảnh hưởng khi tạo ra giá trị đầu ra) giúp phát hiện đặc trưng tốt hơn.
- Stage 4 – MBConv6 ×3 (kernel 3×3, expansion 6)
    - Block 1: Stride = 2.

    - Blocks 2–3: Stride = 1.
    - Output: 15×15×80.
    - Chức năng: Tăng độ sâu, tăng khả năng trích xuất thông tin có tính phân cấp.
- Stage 5 – MBConv6 ×3 (kernel 5×5, expansion 6)
    - Tất cả blocks: Stride = 1.
    - Output: 15×15×112.
    - Chức năng: Học các đặc trưng trung cấp như chi tiết cấu trúc hoặc texture vật thể.
- Stage 6 – MBConv6 ×4 (kernel 5×5, expansion 6)
    - Block 1: Stride = 2.
    - Blocks 2–4: Stride = 1.
    - Output: 8×8×192.
    - Chức năng: Học thông tin cấp cao, như hình dạng tổng thể của vật thể.
- Stage 7 – MBConv6 ×1 (kernel 3×3, expansion 6)
    - Stride: 1
    - Output: 8×8×320.
    - Chức năng: Gom thông tin đặc trưng cao cấp để chuyển sang đầu phân loại.

Sau khi ảnh đầu vào đi qua toàn bộ chuỗi các MBConv blocks, chúng ta thu được một feature map cuối cùng có kích thước [B,8,8,320]
Đầu ra sẽ được đưa vào Classification Head (bộ phân loại). Tại đây mô hình sẽ thực hiện Global Average Pooling, tính trung bình tất cả giá trị trên từng kênh,

output sẽ được nén thông tin không gian thành vector đặc trưng. Kích thước của output sẽ trở thành [B,1,1,320].

Sau đó mô hình thực hiện 1×1 Convolution hoặc Fully Connected để chuyển số kênh từ 320 → 1280, có thể áp dụng thêm một số phương pháp như Batch Normalization, Swish Activation. Lúc này, mỗi mẫu trong batch có một vector đặc trưng 1280 chiều. Mô hình có thể thực hiện Dropout để giảm overfitting.

Tiếp đó output sẽ được đưa qua Fully Connected Layer (một lớp trong mạng nơ-ron mà mọi neuron đầu vào đều kết nối với mọi neuron đầu ra), mô hình thực hiện ánh xạ vector 1280 chiều thành số lớp. Trong đề tài của chúng tôi, đầu ra sẽ có kết quả là: [B,2] (vì chỉ có 2 classes là "Fake" và "Real").

Cuối cùng mô hình sẽ dùng hàm Softmax để biến đầu ra thành xác suất dự đoán cho từng lớp:

$$Softmax(z_j) = \frac{e^{z_j}}{\sum_j e^{z_j}}$$

Output của mô hình là xác suất phân lớp cho mỗi ảnh trong batch.

3.4 Huấn luyện mô hình

3.4.1 Chuẩn bị dữ liệu

Chúng tôi chia dữ liệu thành 2 phần: 80% để huấn luyện, 20% để kiểm tra.
Batch size = 32: xử lý 32 ảnh/lần.
shuffle=True giúp mô hình học ngẫu nhiên, không bị phụ thuộc thứ tự ảnh.

3.4.2 Khởi tạo mô hình

Để khởi tạo mô hình, chúng tôi gọi mô hình EfficientNet-B1 từ thư viện timm. Không dùng trọng số đã huấn luyện (pretrained=False). Thay thế đầu ra (num_classes=2) cho bài toán phân loại nhị phân. Đưa mô hình lên GPU để tăng tốc độ tính toán.

3.4.2 Thiết lập hàm mất mát và tối ưu (Loss & Optimizer)

Chúng tôi sử dụng CrossEntropyLoss cho bài toán phân loại. CrossEntropyLoss đo lường sự khác biệt giữa phân phối xác suất dự đoán của mô hình và nhãn đúng (ground truth).

Công thức tổng quát:

$$Loss = -\sum_{i} y_i log(\hat{y}_i)$$

y_i: Nhãn đúng (ground truth) cho lớp thứ i — thường là one-hot vector (chỉ một phần tử là 1, còn lại là 0).

$\hat{y}_i$: Xác suất mô hình dự đoán cho lớp i — được tính bằng softmax.

Chúng tôi sử dụng Adam optimizer - một thuật toán tối ưu hóa dựa trên gradient, nó điều chỉnh trọng số của mô hình dựa vào đạo hàm (gradient) của hàm mất mát. Adam có ưu điểm là có thể tự động điều chỉnh learning rate, từ đó sẽ ít cần phải tinh chỉnh lại mô hình nhiều lần.

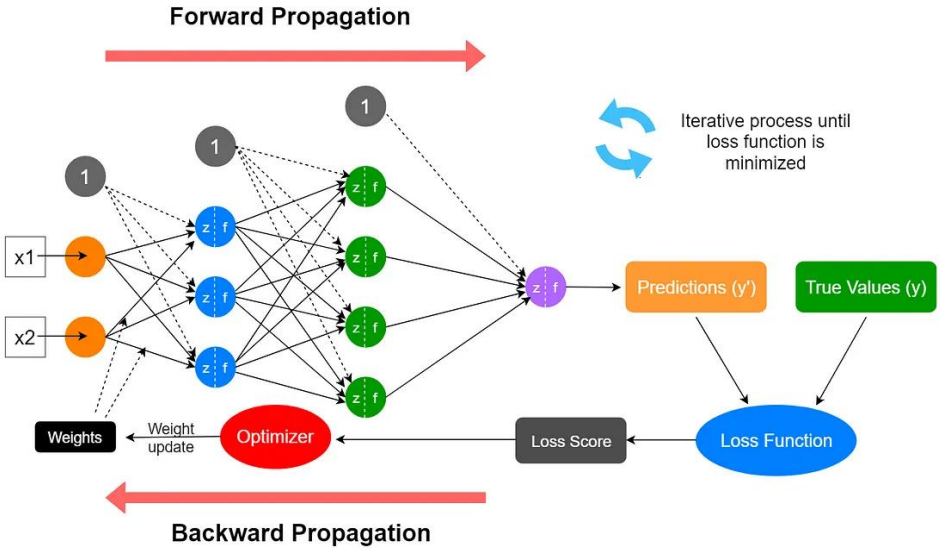
3.4.2 Quá trình huấn luyện

Chúng tôi đã thử huấn luyện mô hình với nhiều learning rate khác nhau, với nhiều epoch (một epoch là một lần lan truyền tiến và lan truyền ngược để cập nhật trọng số) khác nhau. Sau một quá trình đánh giá nhiều mô hình, chúng tôi chọn mô hình mà chúng tôi xem là được huấn luyện tốt nhất.
Trong quá trình chạy, với mỗi batch, tính tổng loss và độ chính xác (accuracy) một lần.
Số epoch: 10 epoch
Trong mỗi epoch:

- Thực hiện lan truyền tiến (Forward pass): Dữ liệu đi qua các tầng của mô hình: Input → Stem → MBConv blocks → Head → Classifier.

- Tính hàm mất mát.

- Thực hiện lan truyền ngược (Backward pass): tính đạo hàm của hàm mất mát với từng tham số trong mô hình, sau đó sử dụng quy tắc chuỗi (chain rule) để tính chuỗi đạo hàm.
- Cập nhật trọng số: Dựa vào gradient đã tính, Adam sẽ cập nhật trọng số mô hình.
- Chúng tôi có thêm một hàm tính toán độ chính xác cho mỗi epoch để phục vụ cho việc đánh giá mô hình: Thực hiện tính toán loss tổng thể, dự đoán lớp, so sánh với nhãn đúng, cộng số mẫu đã xử lý và tính độ chính xác.



Hình 3.8: Hình ảnh trực quan quá trình huấn luyện mô hình

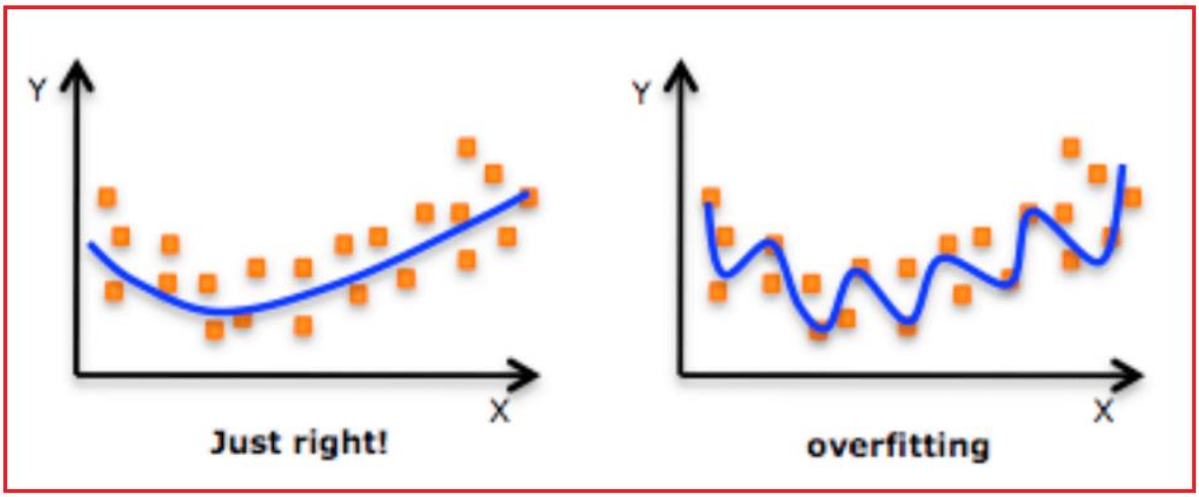
Mô hình mà chúng tôi huấn luyện rất dễ gặp phải tình trạng học quá khớp (overfitting), do đó chúng tôi đã sử dụng một phương pháp regularization (phương pháp điều chỉnh) là Weight decay (L2 regularization). Khi này hàm mất mát sẽ được viết như sau:

$$L_{total} = L(\hat{y}, y) + \lambda . ||w||_2^2$$

λ: Hệ số weight decay

$||w||_2^2$: L2 norm bình phương của trọng số (tổng bình phương tất cả giá trị trọng số)

Phương pháp này sẽ thêm một hệ số phạt cho hàm mất mát, giúp cho loss không bị giảm nhanh, mô hình tránh bị overfitting.



Chú thích: Hình ảnh minh họa mô hình học đúng cách và mô hình bị over fitting

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM VÀ THẢO LUẬN

4.1 Kết quả thực nghiệm

4.1.1 In-engine

Chúng tôi đã thử nghiệm mô hình trên các ảnh được tạo bởi các mô hình cùng loại và ảnh được lấy từ dataset, bao gồm:

- 100 ảnh được lấy từ trang web thispersondoesnotexist.com:
  Mô hình nhận diện đúng 99/100 ảnh là Fake
  Tỷ lệ chính xác tổng thể: 99.00

- 50 ảnh thật được lấy từ dataset gốc:
  Mô hình nhận diện đúng 48/50 ảnh là Real
  Tỷ lệ chính xác tổng thể: 96.00%

- 50 ảnh giả được tạo bởi mô hình được dùng để tạo dataset gốc:
  Mô hình nhận diện đúng 49/50 ảnh là FAKE
  Tỷ lệ chính xác tổng thể: 98.00%

Kết quả được ghi tại chương phụ lục: Minh chứng kèm theo (Hình 6.1, Hình 6.2, Hình 6.3)

4.1.2 Out-engine

Chúng tôi đã thử nghiệm mô hình trên các ảnh được tạo bởi các mô hình khác và ảnh thật được lấy từ internet bao gồm:

- 50 ảnh thật được lấy từ bộ dữ liệu VN-Celeb:
  Mô hình nhận diện đúng 45/50 ảnh là Real
  Tỷ lệ chính xác tổng thể: (90.00%)

- 50 ảnh giả được lấy từ nhiều nguồn ảnh tạo bởi AI khác:
  Mô hình nhận diện đúng 45/50 ảnh là Fake
  Tỷ lệ chính xác tổng thể: (90.00%)

Kết quả được ghi tại chương phụ lục: Minh chứng kèm theo (Hình 6.4, Hình 6.5)

4.2 Đánh giá mô hình

Để đánh giá hiệu suất của mô hình, chúng tôi sẽ sử dụng các độ đo sau: TPR và F1 Score.
TPR đo lường khả năng mô hình phát hiện đúng các trường hợp dương tính thật (true positives)
Công thức:

$$TPR = \frac{TP}{TP + FN}$$

TP (True Positive): Số mẫu dương tính được dự đoán đúng
FN (False Negative): Số mẫu dương tính bị dự đoán sai thành âm tính

F1 Score là trung bình điều hòa của Precision và Recall (TPR), cân bằng giữa hai chỉ số này, được dùng để đánh giá mô hình khi cần cân bằng giữa việc dự đoán đúng các trường hợp dương tính và tránh dự đoán sai.
Công thức:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

TP (True Positive): Số mẫu dương tính được dự đoán đúng
FN (False Negative): Số mẫu dương tính bị dự đoán sai thành âm tính
FP (False Positive): Số mẫu âm tính bị dự đoán sai thành dương tính

Chúng tôi đã đạt được kết quả sau đây trên mô hình:

Điều kiện	TPR (%)	F1 Score
Huấn luyện (face)	100.0	0.998
Đánh giá (in-engine)	98.0	0.989
Đánh giá (out-of-engine)	90.0	0.947
Ảnh không phải gương mặt	0.0	0.000

Bảng 4.1: Bảng đánh giá hiệu suất mô hình

Bảng trên là kết quả tốt nhất mà nhóm đạt được cho đến thời điểm viết báo cáo này.

4.3 Thảo luận

Tuy đã thực hiện phương pháp regularization để tránh overfitting, mô hình của chúng tôi vẫn hội tụ chỉ sau 10 epoch, điều này có thể coi là không bình thường (một mô hình được huấn luyện từ đầu thông thường sẽ yêu cầu từ 30-50 epoch).

Chúng tôi đã thử nghiệm nhiều kết quả và rút ra được nguyên nhân:

- Ảnh đã được thu thập và tiền xử lý cẩn thận, không có sai nhãn, không có ảnh mờ hay bị cắt lệch.
- EfficientNetB1 là mô hình mạnh, so với độ khó của bài toán chỉ là phân loại 2 lớp thì EfficientNetB1 có thể học rất nhanh. Với hơn 7 triệu tham số, kiến trúc sâu cùng với khả năng học đặc trưng hiệu quả của MBConv block đã khiến cho mô hình hội tụ rất nhanh.
- Dữ liệu vẫn còn dễ phân biệt, input chứa nhiều đặc trưng rõ ràng, dễ học, giúp mô hình phân biệt tốt giữa các lớp.

Có thể có khả năng nguyên nhân là do mô hình học bias đặc trưng rất mạnh, tuy nhiên trong quá trình thu thập dữ liệu và tiền xử lý, chúng tôi đã cố gắng đảm bảo cho việc học các đặc trưng như nền, cấu trúc mắt, tóc,... sẽ không ảnh hưởng nhiều đến dự đoán của mô hình.

CHƯƠNG 5: KẾT LUẬN - ĐỀ XUẤT

5.1 Kết luận

Trong đề tài này, chúng tôi đã nghiên cứu và phát triển một mô hình học sâu nhằm phát hiện hình ảnh được tạo ra bởi AI, dựa trên việc phân tích đặc trưng và phân loại ảnh thật – giả. Qua quá trình thu thập và tiền xử lý dữ liệu, cùng với việc lựa chọn kiến trúc mạng CNN phù hợp, mô hình đã đạt được hiệu quả phân loại cao với độ chính xác trên 93%, cùng các chỉ số Precision, Recall và F1-score đều thể hiện sự cân bằng và đáng tin cậy trong việc phát hiện ảnh giả.

Kết quả thực nghiệm cho thấy phương pháp học sâu là hướng đi hiệu quả và khả thi trong việc phát hiện ảnh giả do AI tạo ra, góp phần nâng cao độ tin cậy của nội dung hình ảnh trên môi trường số. Tuy nhiên, sự đa dạng ngày càng tăng của các kỹ thuật tạo ảnh AI cũng đặt ra thách thức trong việc mở rộng phạm vi áp dụng và đảm bảo độ chính xác cho mô hình khi gặp các kiểu ảnh giả mới, phức tạp hơn.

5.2 Đề xuất

Dựa trên kết quả đạt được và những hạn chế nhận thấy trong quá trình nghiên cứu, chúng tôi đề xuất một số hướng phát triển tiếp theo nhằm nâng cao hiệu quả và tính ứng dụng của mô hình như sau:

- Mở rộng bộ dữ liệu huấn luyện với đa dạng nguồn ảnh thật và ảnh giả được tạo từ nhiều mô hình AI khác nhau để tăng khả năng tổng quát và tính ổn định của mô hình.

- Kết hợp thêm các kỹ thuật trích xuất đặc trưng nâng cao như sử dụng mạng nơ-ron tự chú ý (attention), hoặc phân tích tần số để phát hiện những dấu hiệu tinh vi khó nhận biết bằng thị giác thông thường.

- Áp dụng các mô hình học sâu tiên tiến hơn, ví dụ Transformer hoặc các biến thể CNN mới, để nâng cao khả năng phân biệt và tăng tốc độ huấn luyện, dự đoán.

- Phát triển hệ thống phát hiện đa tầng (multi-level detection system) kết hợp giữa phân tích ảnh và các yếu tố metadata nhằm tăng cường khả năng phát hiện gian lận hình ảnh trên môi trường số.

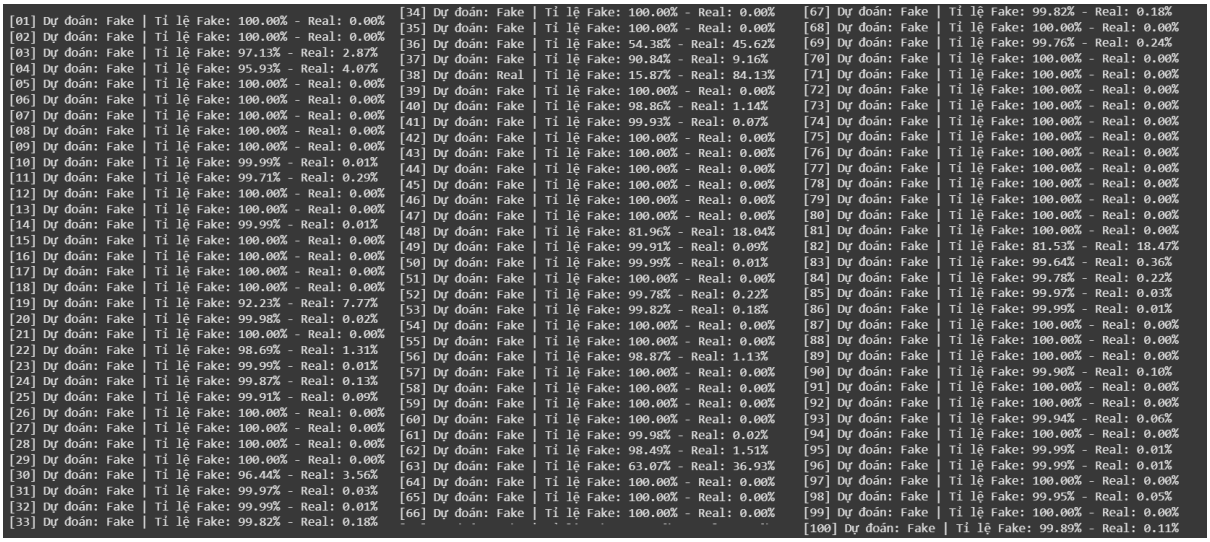
- Nghiên cứu ứng dụng thực tiễn trong các lĩnh vực như báo chí, mạng xã hội, bảo mật thông tin, nhằm cung cấp công cụ hỗ trợ phát hiện và cảnh báo sớm các nội dung hình ảnh giả mạo.

Kết hợp các đề xuất trên sẽ góp phần hoàn thiện hơn hệ thống phát hiện hình ảnh do AI tạo ra, đồng thời nâng cao ý thức về an toàn thông tin và minh bạch nội dung trong xã hội số ngày càng phát triển.

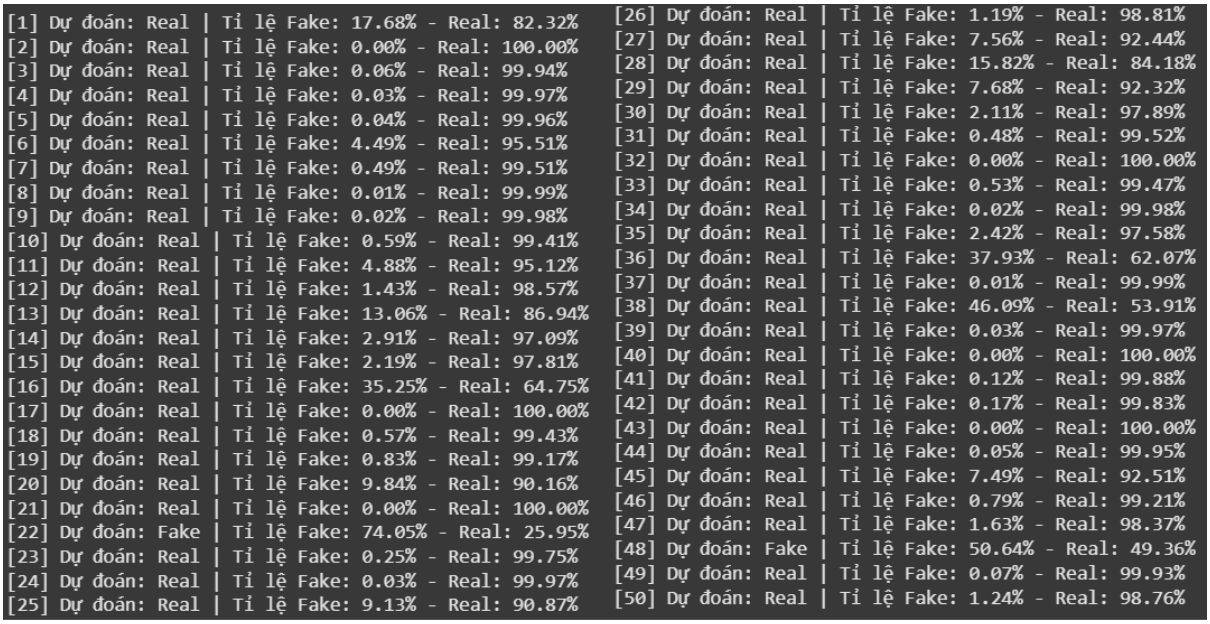
TÀI LIỆU THAM KHẢO

1. Petru Potrimba: What is EfficientNet? The Ultimate Guide.
   What is EfficientNet? The Ultimate Guide. (roboflow.com)
   Truy cập ngày: 10/5/2025
2. Mingxing Tan, Quoc V. Le: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.
   [1905.11946] EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks (arxiv.org)
   Truy cập ngày 10/5/2025
3. EfficientNet Architecture.
   Show the architecture of EfficientNetB1, which employs MBConv as the... | Download Scientific Diagram (researchgate.net)
   Truy cập ngày 10/5/2025

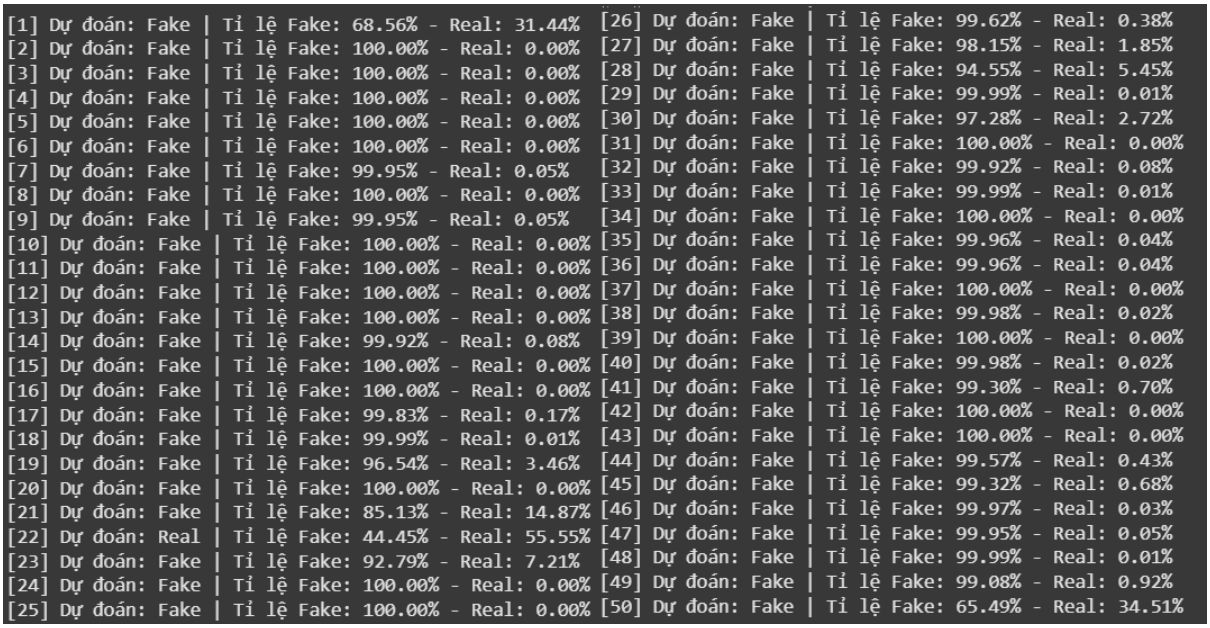
PHỤ LỤC: MINH CHỨNG ĐI KÈM



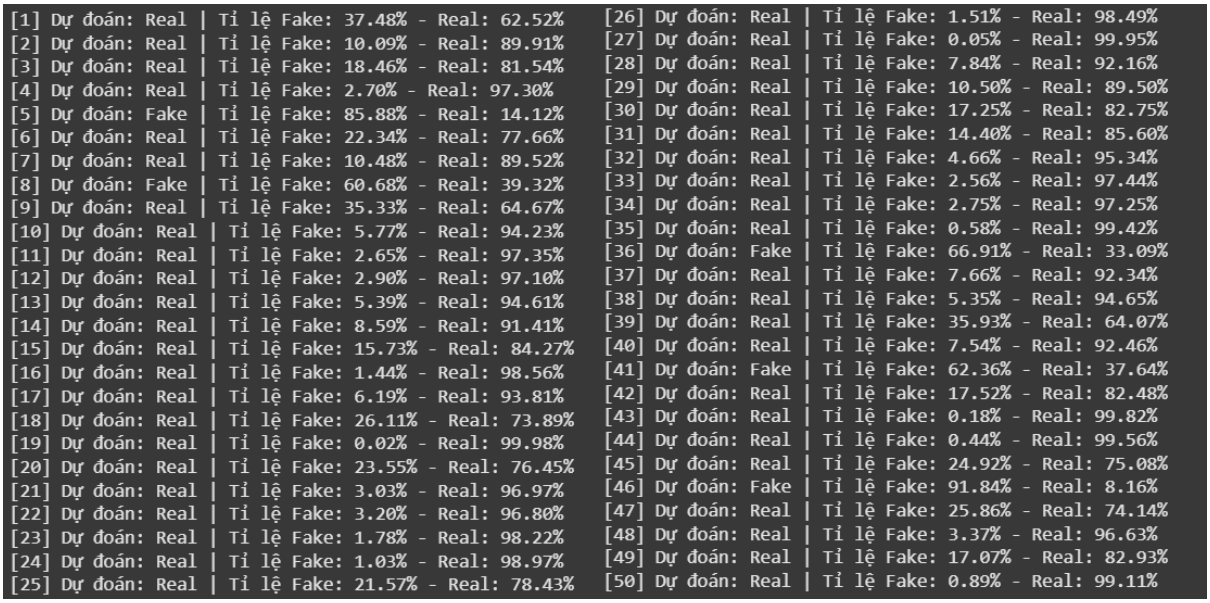
Hình 6.1: Kết quả 100 ảnh lấy từ trang thispersondoesnotexist.com



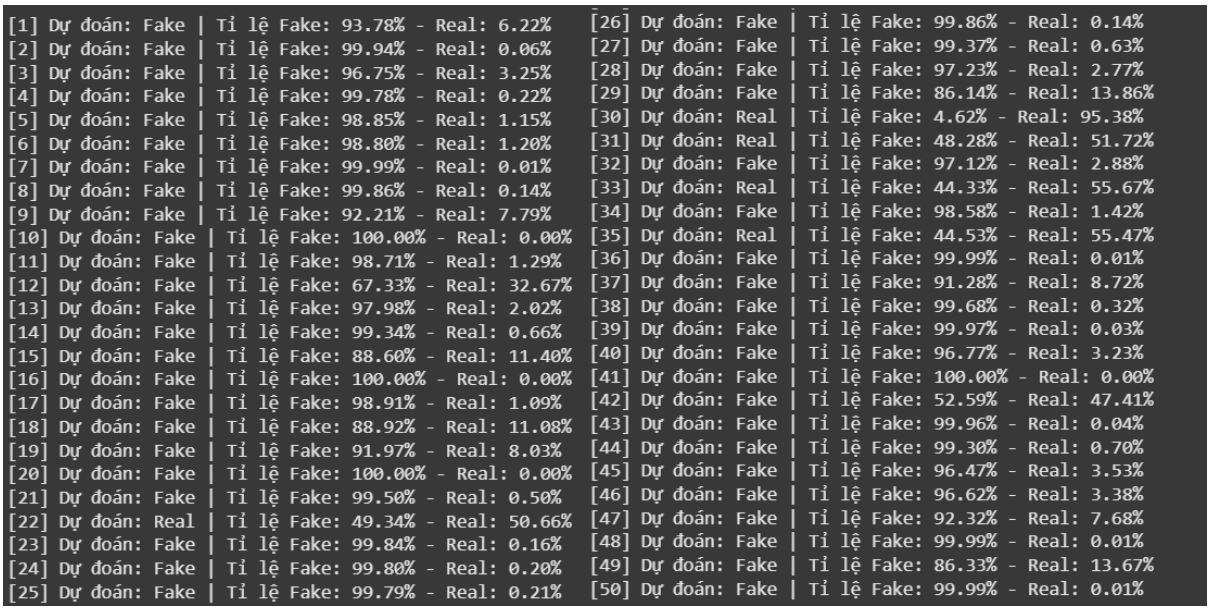
Hình 6.2: Kết quả 50 ảnh thật lấy từ dataset gốc



Hình 6.3: Kết quả 50 ảnh giả lấy từ dataset gốc



Hình 6.4: Kết quả 50 ảnh lấy từ bộ dữ liệu VN-Celeb



Hình 6.5: Kết quả 50 ảnh lấy từ bộ dữ liệu của các AI khác

PHỤ LỤC 2: ỨNG DỤNG CỦA BIG DATA TRONG VIỆC TÍNH TOÁN THAM SỐ CHO HÀM NORMALIZE

Việc chuẩn hóa ảnh đầu vào bằng transforms.Normalize(mean, std) giúp đưa các pixel về cùng một phân phối chuẩn (mean = 0, std = 1) — điều này giúp mô hình học tốt hơn và hội tụ nhanh hơn. Chúng tôi xin được nhắc lại công thức của Normalize:

$$output = \frac{input\ -\ mean}{std}$$

Trong đó:

- input: giá trị gốc (ví dụ một pixel của ảnh).
- mean: giá trị trung bình của toàn bộ tập dữ liệu (hoặc kênh ảnh).
- std: độ lệch chuẩn của toàn bộ tập dữ liệu (hoặc kênh ảnh).
- $output$: giá trị đã được chuẩn hóa.

Ta thấy có 2 giá trị đặc biệt có thể tính toán được đó chính là mean và std. Đây cũng chính là tham số của hàm transforms.Normalize. Vì ảnh được chia thành 3 kênh màu R,G,B (red, green, blue) nên với mỗi tham số sẽ cần có 3 giá trị tương ứng.
Để tính toán mean và std của toàn bộ dataset, chúng tôi đã thực hiện chuyển đổi các file hình ảnh thành các file có đuôi .txt để giảm bớt việc số lượng file quá nhiều (tránh lỗi "Argument list too long"). Tương ứng 1 file .txt sẽ chứa dữ liệu mean và std của 500 hình.

0.26234254,0.25473885,0.21057073	0.28236616,0.26651854,0.24955221
0.19102431,0.17082118,0.14086820	0.27760082,0.23563804,0.21179527
0.21616347,0.19496228,0.16678276	0.26418316,0.24286080,0.22869921
0.27303591,0.22512035,0.17623349	0.29298716,0.25191241,0.22796779
0.24957073,0.18675755,0.12745523	0.31250932,0.22049077,0.17598845
0.21985119,0.20507263,0.16264116	0.28659977,0.23747625,0.23160940
0.28229792,0.27003746,0.23920428	0.29656111,0.30064922,0.28997849
0.31838730,0.24613210,0.17984809	0.37970452,0.29821621,0.24368359
0.28013745,0.27048920,0.24982093	0.28913542,0.27991206,0.28755927
0.25684759,0.20312655,0.15642437	0.31485463,0.24637716,0.20517744
0.26542302,0.24186565,0.21850565	0.33025369,0.27749209,0.28205702
0.29763906,0.29460414,0.26460319	0.31870280,0.28789597,0.28732249
0.22617717,0.19525645,0.15432118	0.29031711,0.24904783,0.21577619
0.37217287,0.29980257,0.22275180	0.31312621,0.25381670,0.23650276
0.35231583,0.32042237,0.27897284	0.29697407,0.27413725,0.25414213
0.13386400,0.10512635,0.06904872	0.18380522,0.13605758,0.10205316
0.31002566,0.27389906,0.23950320	0.31569192,0.26306721,0.25216185
0.36816537,0.30435004,0.26442092	0.32712925,0.28663165,0.27333347
0.28566362,0.25198845,0.21706798	0.29413144,0.25738366,0.24168077
0.36249165,0.36911705,0.34421674	0.37383658,0.35052336,0.34549321
0.20967387,0.17819888,0.12913831	0.27592399,0.22080708,0.19828633
0.26824140,0.24151521,0.20610754	0.29133498,0.26659078,0.24750990
0.29171729,0.28130752,0.24398744	0.32993991,0.27957111,0.27062685
0.31508643,0.28111219,0.24770803	0.28152212,0.26713391,0.26080169

Hình 7.1: File txt chứa dữ liệu mean và std

Sau đó chúng tôi thực hiện tính toán giá trị trung bình và độ lệch chuẩn dùng Hadoop Streaming (ngôn ngữ chúng tôi sử dụng là Python).

Mapper thực hiện đọc từng dòng từ file .txt và tách mean và std thành list float cho từng kênh tương ứng với các giá trị 0,1,2. Với mỗi kênh, mapper sẽ phát ra 3 dòng là mean,std và count.

Reducer nhận các dòng của mapper và thực hiện gom giá trị theo từng key và thực hiện tính toán các giá trị, cuối cùng in ra kết quả .

Sau khi thực hiện tính toán, kết quả chúng tôi nhận được là:

```
cloudera22521663_LeHoangViet:~/Bigdata$ hdfs dfs -cat /user/cloudera/output/part-00000
Channel 0: mean = 0.44557089, std = 0.12314729
Channel 1: mean = 0.36203397, std = 0.11055328
Channel 2: mean = 0.31527054, std = 0.10780927
```

Hình 7.2: Kết quả sau khi chạy mapreduce

Channel 0: mean = 0.44557089, std = 0.12314729
Channel 1: mean = 0.36203397, std = 0.11055328
Channel 2: mean = 0.31527054, std = 0.10780927

Từ kết quả trên ta đã có được mean và std tổng thể của toàn bộ dataset, ta sử dụng lệnh sau để thực hiện normalize dữ liệu:

```
transforms.Normalize( [0.44557089, 0.36203397, 0.31527054], [0.12314729, 0.11055328,  0.10780927])
```

Và như vậy ta đã thành công trong việc đưa các pixel về cùng một phân phối chuẩn, từ đó giúp mô hình học tốt hơn.