

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN**

**TRẦN THANH TÙNG – VÒNG CẢNH CHI**

**MÔ HÌNH ĐỐI KHÁNG SINH MẪU CHO CHUYỂN ĐỔI GIỮA CÁC MIỀN  
KHÔNG GIAN ẢNH - ỨNG DỤNG TRÊN VIDEO  
KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT  
CHƯƠNG TRÌNH CHÍNH QUY**

**Tp. Hồ Chí Minh, tháng 02/2022**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**  
**KHOA CÔNG NGHỆ THÔNG TIN**

**TRẦN THANH TÙNG – 18120258**  
**VÒNG CẢNH CHI – 18210293**

**MÔ HÌNH ĐỐI KHÁNG SINH MẪU CHO CHUYỂN ĐỔI GIỮA CÁC MIỀN  
KHÔNG GIAN ẢNH - ỨNG DỤNG TRÊN VIDEO**

**KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT**  
**CHƯƠNG TRÌNH CHÍNH QUY**

**GIÁO VIÊN HƯỚNG DẪN**  
**PGS.TS. Lê Hoàng Thái**

**Tp. Hồ Chí Minh, tháng 02/2022**

# Lời cảm ơn

Trước hết, chúng tôi muốn gửi lời cảm ơn sâu sắc đối với PSG.TS Lê Hoàng Thái, thầy trực tiếp hướng dẫn và hỗ trợ nhóm trong suốt quá trình thực hiện khóa luận.. Khóa luận này chắc chắn không thể hoàn thành nếu không có sự tận tâm của thầy.

Chúng tôi chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên, Đại học Quốc gia Tp. Hồ Chí Minh đã tạo điều kiện thuận lợi cho chúng tôi trong 4 năm học tập tại trường và thực hiện khóa luận.

Chúng tôi xin chân thành cảm ơn quý thầy cô thuộc Khoa Công Nghệ Thông Tin đã tận tình giảng dạy, trang bị cho chúng tôi những kiến thức quý báu trong suốt quá trình học tập để có thể thực hiện được đề tài.

Chúng tôi cũng rất biết ơn gia đình và bạn bè, những người đã giúp đỡ, cổ vũ chúng tôi rất nhiều trong lúc chúng tôi gặp khó khăn cũng như trong suốt quá trình thực hiện đồ án.

Mặc dù đã cố gắng hoàn thành khóa luận trong phạm vi và khả năng cho phép nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong nhận được sự góp ý và tận tình chỉ bảo của quý thầy cô.

Một lần nữa, chúng tôi xin chân thành cảm ơn và mong nhận được sự chỉ bảo của quý thầy cô để khóa luận tốt nghiệp được hoàn chỉnh hơn.

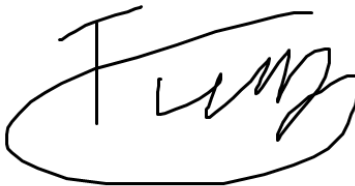

Thành phố Hồ Chí Minh, tháng 02 năm 2022

Nhóm sinh viên thực hiện

Trần Thanh Tùng – Vòng Cảnh Chi

# Đề cương chi tiết

<b>Tên đề tài:</b> Mô hình đối kháng sinh mẫu cho chuyển đổi giữa các miền không gian ảnh - ứng dụng trên video
<b>Giáo viên hướng dẫn:</b> PGS.TS Lê Hoàng Thái
<b>Thời gian thực hiện:</b> 6 tháng
<b>Sinh viên thực hiện:</b> Trần Thanh Tùng – 18120258 Vòng Cảnh Chi – 18120293
<b>Loại đề tài:</b> Nghiên cứu
<b>Nội dung đề tài:</b> Đây là dạng đề tài theo nghiên cứu lí thuyết để kiểm soát mẫu sinh ra bởi mô hình đối kháng sinh mẫu của bài toán chuyển đổi giữa các miền không gian ảnh. Chúng tôi thực hiện điều này thông qua việc kiểm soát miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu theo các cách tiếp cận sẵn có và tự đề xuất. Sau đó đánh giá các phương pháp theo kết quả định tính và định lượng để so sánh mức độ hiệu quả của các phương pháp.
<b>Kế hoạch thực hiện:</b> <p>Mỗi thành viên trong nhóm phải nắm được bản chất của việc tinh chỉnh phong cách hoạt họa cũng như cách mà bộ tạo, bộ phân biệt hoạt động trong GANs. Các thành viên phải có trách nhiệm truyền đạt cho nhau những kiến thức mà mình tìm hiểu chính để mọi thành viên cùng nắm được vấn đề. Công việc được chia thành 2 phần chính như sau:</p> <ul style="list-style-type: none"><li>• Tìm hiểu và áp dụng phương pháp học không giám sát (Sefa) trong tinh chỉnh phong cách hoạt họa: Chịu trách nhiệm chính – Vòng Cảnh Chi.</li><li>• Tìm hiểu và áp dụng phương pháp học có giám sát (sử dụng mạng Deep Danbooru) trong tinh chỉnh phong cách hoạt họa: Chịu trách nhiệm chính – Trần Thanh Tùng.</li></ul>

<p>Thời gian và nội dung thực hiện cụ thể:</p> <ul style="list-style-type: none"> <li>- Tháng 9/2021 – 10/2021: Tìm hiểu về mô hình sinh đối kháng sinh mẫu cho chuyển đổi giữa các miền không gian ảnh GANs N’ Rose.</li> <li>- Tháng 10/2021 – 12/2021: Tìm hiểu hai phương pháp tinh chỉnh phong cách không giám sát và có giám sát.</li> <li>- Tháng 12/2021 – 2/2022: Thực hiện đánh giá và áp dụng phương pháp tinh chỉnh phong cách đã chọn để tạo các đoạn phim hoạt hình.</li> <li>- Tháng 2/2022 – 3/2022: Viết báo cáo luận văn và chuẩn bị cho bảo vệ đề tài.</li> </ul>	
<p><b>Xác nhận của GVHD</b></p>	<p><i>TP. Hồ Chí Minh, ngày 05 tháng 10 năm 2021</i></p> <p><b>NHÓM SINH VIÊN THỰC HIỆN</b> (Ký và ghi rõ họ tên)</p> <div data-bbox="890 831 1246 1016" data-label="Text">  </div> <p>Trần Thanh Tùng</p> <div data-bbox="954 1093 1310 1294" data-label="Text">  </div> <p>Vòng Cảnh Chi</p>

# Mục lục

Lời cảm ơn .....	iii
Đề cương chi tiết.....	iv
Mục lục .....	vi
Danh sách các hình.....	xi
Danh sách các bảng.....	xiii
Tóm tắt .....	xiv
<b>Chương 1 Giới thiệu về đề tài.....</b>	<b>1</b>
1.1 Lý do nghiên cứu .....	1
1.2 Mục tiêu nghiên cứu .....	2
1.3 Cách tiếp cận .....	2
1.4 Đóng góp .....	3
<b>Chương 2 Phát biểu bài toán.....</b>	<b>4</b>
2.1 Tổng quan về mô hình sinh mẫu .....	4
2.1.1 Khái niệm.....	4
2.1.2 Khả năng của mô hình sinh mẫu.....	4
2.1.3 Các loại mô hình sinh mẫu: .....	4
2.1.3.1 Các mạng Bayes:.....	4
2.1.3.2 Bộ mã hóa tự động biến thể: .....	5
2.1.3.3 Máy Boltzmann.....	6
2.1.3.4 Mạng đối kháng sinh mẫu.....	7
2.2 Tổng quan về bài toán chuyển đổi giữa các miền không gian ảnh bằng mô hình đối kháng sinh mẫu .....	7

2.2.1	Khái niệm.....	7
2.2.2	Các dạng bài toán.....	10
2.2.2.1	Có giám sát.....	10
2.2.2.2	Không giám sát .....	10
2.2.3	Các phương thức .....	11
2.2.3.1	Uni-modal .....	11
2.2.3.2	Mutli-modal .....	12
2.3	Bài toán chuyển miền ảnh mặt người sang ảnh hoạt họa .....	13
2.3.1	Mô tả bài toán .....	13
2.3.2	Các cách tiếp cận .....	13
2.4	Bài toán kiểm soát miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu.. .....	14
2.4.1	Khái niệm.....	14
2.4.2	Kiểm soát miền không gian tiềm ẩn .....	16
<b>Chương 3</b>	<b>Cơ sở lý thuyết .....</b>	<b>17</b>
3.1	Mô hình đối kháng sinh mẫu .....	17
3.1.1	Định nghĩa.....	17
3.1.2	Kiến trúc .....	17
3.2	StyleGAN .....	20
3.2.1	Định nghĩa.....	20
3.2.2	Progressive Growing .....	21
3.2.3	Mạng ánh xạ.....	22
3.2.4	AdaIN (Adaptive Instance Normalization).....	22
3.2.5	Stochastic variation.....	23

3.3	StyleGAN2 .....	25
3.3.1	Chuẩn hóa thay vì sử dụng AdaIN .....	25
3.3.1.1	Vấn đề của AdaIN.....	25
3.3.1.2	Chuẩn hóa .....	25
3.3.2	Giải quyết vấn đề của Progressive Growing .....	27
3.3.2.1	Vấn đề của Progressive Growing.....	27
3.3.2.2	Giải pháp .....	27
3.3.3	Cung cấp tính liên tục cho không gian tiềm ẩn .....	29
3.4	GANs N' Roses .....	30
3.4.1	Kiến trúc Encoder – Decoder .....	31
3.4.2	Đảm bảo đa dạng phong cách.....	33
3.4.3	Các độ mất mát .....	34
3.5	Kiểm soát miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu.....	37
3.5.1	Sơ bộ .....	37
3.5.2	Không giám sát .....	38
3.5.3	Có giám sát .....	39
<b>Chương 4</b>	<b>Phương pháp đề xuất .....</b>	<b>41</b>
4.1	Giới thiệu phương pháp thao túng miền không gian tiềm ẩn được đề xuất thử nghiệm .....	41
4.2	Quy trình thao túng miền không gian tiềm ẩn có giám sát.....	43
4.2.1	Sinh ảnh hoạt họa ngẫu nhiên:.....	43
4.2.2	Gán nhãn:.....	43
4.2.3	Các phương pháp tìm vector ngữ nghĩa được đề xuất.....	43
4.2.3.1	Hồi quy:.....	43



4.2.3.2	Phân rã ma trận để tìm vector riêng .....	44
4.2.3.3	So sánh giữa 2 phương pháp về mặt lý thuyết .....	45
4.2.3.4	Các kỹ thuật phụ họa bổ sung .....	46
4.3	Tổng quan quy trình của chương trình .....	46
<b>Chương 5</b>	<b>Thực nghiệm và kết quả .....</b>	<b>48</b>
5.1	Dữ liệu .....	48
5.2	Độ đo .....	48
5.2.1	FID .....	48
5.2.2	DFID .....	50
5.2.3	LPIPS .....	51
5.2.4	Re-scoring .....	<b>Error! Bookmark not defined.</b>
5.3	Quy trình thực nghiệm .....	52
5.3.1	Quá trình tạo các vector nhãn: .....	52
5.3.1.1	Sinh ảnh hoạt họa ngẫu nhiên: .....	52
5.3.1.2	Gán nhãn: .....	52
5.3.1.3	Tìm vector ngữ nghĩa: .....	53
5.3.2	Chỉnh sửa ảnh sử dụng các vector nhãn: .....	54
5.3.3	Đo các độ đo: .....	54
5.3.3.1	Đo FID: .....	54
5.3.3.2	Đo DFID: .....	55
5.3.3.3	Đo LPIPS: .....	55
5.3.3.4	Đo Re-scoring: .....	55
5.4	So sánh định tính .....	55
5.5	So sánh định lượng: .....	57

5.6	Ứng dụng trên video: .....	62
<b>Chương 6</b>	<b>Kết luận và hướng phát triển .....</b>	<b>63</b>
6.1	Kết luận.....	63
6.2	Hướng phát triển.....	63
<b>Tài liệu tham khảo</b> .....		<b>64</b>

# Danh sách các hình

Hình 2.1. Các mẫu sinh bởi mạng PixelCNN [13] .....	5
Hình 2.2 Một mẫu sinh của VAE [14] .....	6
Hình 2.3 Chuyển đổi ảnh chụp phong cảnh sang tranh vẽ bằng CycleGAN[16] .....	8
Hình 2.4 Chuyển miền ảnh bản đồ sang ảnh chụp từ vệ tinh với pix2pix [17] .....	9
Hình 2.5 Ảnh mờ và ảnh rõ nét tương ứng [18].....	10
Hình 2.6 Chuyển miền ảnh từ một giống chó này sang các giống chó khác [19] ....	11
Hình 2.7 Ứng dụng chuyển ảnh phác họa sang ảnh pokemon [17] .....	12
Hình 2.8 Chuyển ảnh khuôn mặt người [21] .....	12
Hình 3.1 Kiến trúc mô hình đối kháng sinh mẫu [12] .....	17
Hình 3.2 Kiến trúc mạng sinh được sử dụng trong StyleGAN [34] .....	21
Hình 3.3. Trục quan quá trình AdaIN [37] .....	23
Hình 3.4 Thêm nhiễu vào mỗi khối sinh của mạng sinh [37].....	24
Hình 3.5 AdaIN tạo ra nhiễu trong ảnh sinh ra bởi StyleGAN [1] .....	25
Hình 3.6 Vấn đề của Progressive growing [1] .....	27
Hình 3.7 Kiến trúc được sử dụng trong StyleGAN2 .....	28
Hình 3.8 So sánh các chi tiết khuôn mặt của ảnh sinh bởi StyleGAN và StyleGAN2 [38] .....	29
Hình 3.9. Kiến trúc Generator được sử dụng [3] .....	30
Hình 3.10. Kiến trúc của bộ Discriminator được sử dụng [3] .....	31
Hình 3.11. Mô hình Encoder – Decoder [40] .....	31
Hình 4.1 Ví dụ của việc không bảo toàn được một số chi tiết khi chuyển miền ảnh .....	42
Hình 4.2 Minh họa sự chỉnh sửa của SeFa.....	42
Hình 4.3 Sơ đồ quá trình tạo các vector nhãn .....	43
Hình 4.4 Quy trình phân rã ma trận tìm vector riêng.....	45
Hình 4.5 Tổng quan quy trình sinh ảnh của chương trình .....	47

Hình 5.1 Quy trình tính FID.....	49
Hình 5.2 Trực quan quá trình đo Re-scoring .....	<b>Error! Bookmark not defined.</b>
Hình 5.3. Minh họa chỉnh sửa bằng các phương pháp được đề xuất.....	55
Hình 5.4 Dùng vector nhãn của các phương pháp để chỉnh màu tóc sang màu xanh .....	56
Hình 5.5 Minh họa ảnh hưởng của việc sử dụng cặp nhãn đối nghịch.....	57
Hình 5.6 Ứng dụng cho phong cách đã chỉnh sửa trên video .....	62

# Danh sách các bảng

Bảng 5.1 Kiến trúc mạng DeepDanbooru .....	53
Bảng 5.2 Kết quả độ đo FID, DFID và LPIPS của các phương pháp chỉnh sửa .....	<b>Error! Bookmark not defined.</b>
Bảng 5.3 Kết quả độ đo Re-scoring của SeFa.....	<b>Error! Bookmark not defined.</b>
Bảng 5.4 Kết quả độ đo Re-scoring của hồi quy Lasso .....	<b>Error! Bookmark not defined.</b>
Bảng 5.5 Kết quả độ đo Re-scoring của hồi quy tuyến tính ...	<b>Error! Bookmark not defined.</b>
Bảng 5.6 Kết quả độ đo Re-scoring của phương pháp phân rã ma trận .....	<b>Error! Bookmark not defined.</b>

# Tóm tắt

Chuyển đổi giữa các miền không gian ảnh hiện nay đang là một vấn đề được giới nghiên cứu về học máy quan tâm và đã có nhiều ứng dụng trong thực tế. Một cách tiếp cận hiệu quả và được sử dụng nhiều trong thời gian gần đây chính là mô hình đối kháng sinh mẫu. Tuy nhiên việc chuyển đổi giữa các miền không gian ảnh khác nhau cần những mô hình đối kháng sinh mẫu khác nhau, một trong những miền không gian rất khó để chuyển đổi là ảnh hoạt họa, nguyên nhân xuất phát từ bản chất của ảnh là các đường nét rõ ràng, cũng như màu sắc của ảnh hoạt họa là các khối tách rời, phân biệt. Bằng sự kết hợp giữa mô hình đối kháng sinh mẫu đột phá [1] làm nền tảng, với kiến trúc Encoder-Decoder [2] và sự tăng cường dữ liệu trong quá trình huấn luyện, một nghiên cứu gần đây [3] đã thành công trong việc chuyển miền ảnh mặt người thật sang miền ảnh mặt người hoạt họa, với sự đa dạng phong cách giữa các ảnh hoạt họa được sinh ra từ ảnh gốc. Từ sự đa dạng giữa các phong cách này, nhóm chúng tôi nhận thấy tiềm năng về việc ứng dụng làm phim hoạt hình bằng cách kiểm soát có chủ đích của các ảnh được sinh ra, bởi nó giúp người làm phim có nhiều lựa chọn trong việc thiết kế nhân vật cũng như chỉnh sửa những chi tiết mong muốn. Việc này được nhóm thực hiện bằng cách cải tiến kỹ thuật thao túng miền không gian tiềm ẩn trong mô hình đối kháng sinh mẫu. Kết quả thu được là các ảnh sinh ra từ quá trình chỉnh sửa mang những thuộc tính chỉnh sửa mong muốn mà vẫn giữ được sự đa dạng về phong cách.

## **Chương 1**

# **Giới thiệu về đề tài**

### **1.1 Lý do nghiên cứu**

Một trong những công đoạn tiêu tốn thời gian nhất trong quá trình làm phim hoạt hình chính là việc dựng hình cho các nhân vật. Theo một nguồn tham khảo từ trang Ehancedrawing [4] thì việc vẽ một ảnh mặt hoạt họa có thể tốn trung bình từ 2 đến 3 giờ đồng hồ. Cứ mỗi giây trong một phim hoạt hình bình thường cần có 24 khung hình [5] để khắc hoạt chuyển động nhân vật trong khoảng thời gian đó, cho rằng một bộ phim hoạt hình 90 phút có ít nhất 45 phút xuất hiện mặt người thì nhóm họa sĩ cần phải tốn từ 225 ngày tới 337 ngày chỉ để vẽ ảnh mặt hoạt họa. Các nhà làm phim hiện đại thường sử dụng công nghệ theo dõi chuyển động của diễn viên bằng các bộ đồ gắn kèm cảm biến, sau đó các chuyển động này được đưa cho nhóm họa sĩ để tạo hình theo [6]. Tuy nhiên quá trình này đòi hỏi chi phí tốn kém rất lớn, kèm với trình độ cao của các bên tham gia. Chính vì thế, việc sinh ảnh hoạt họa tự động từ ảnh người thật có thể giúp việc làm phim hoạt hình nhanh hơn, tiết kiệm chi phí, cũng như giúp các nhà làm phim nghiệp dư dễ tiếp cận hơn.

Một trong những yêu cầu tối quan trọng đối với việc làm phim hoạt hình là thiết kế, tạo hình cho nhân vật. Quá trình này phụ thuộc rất nhiều vào nguồn cảm hứng và trí tưởng tượng của nhà thiết kế. Đôi khi việc dựa trên các thiết kế có sẵn rất dễ dẫn đến các vấn đề liên quan tới bản quyền. Việc có sẵn một nguồn phong phú các phong cách hoạt họa để tham khảo giúp ích cho nhà thiết kế rất nhiều trong việc định hình nhân vật. Sau đó, để đến được với tạo hình cuối cùng của nhân vật, tác giả phải thử thiết kế từng chi tiết để phù hợp với nhân vật mong muốn, như là liệu nhân vật đó hợp với tóc dài hay tóc ngắn hơn, mắt đen hay mắt xanh,...

Chính vì 2 lý do trên nên nhóm đã ứng dụng tính đa dạng của việc sinh ảnh hoạt họa từ ảnh mặt người từ mô hình GANs N' Roses [3] kết hợp với quá trình thao túng miền không gian tiềm ẩn trong mô hình để hỗ trợ quá trình làm phim hoạt hình.

## 1.2 Mục tiêu nghiên cứu

Mục tiêu nghiên cứu mà nhóm nhắm đến là áp dụng kỹ thuật thao túng miền không gian tiềm ẩn của bộ đối kháng sinh mẫu trong mô hình GANs N' Roses của Chong và cộng sự [3], thông qua đó chỉnh sửa ảnh được tạo ra để hỗ trợ quá trình làm phim hoạt hình.

## 1.3 Cách tiếp cận

Để thao túng miền không gian tiềm ẩn của bộ đối kháng sinh mẫu, nhóm tiếp cận theo 2 phương pháp:

- Phương pháp không giám sát (Sefa [7]): Sử dụng phương pháp tiếp cận tương tự PCA [8] để tìm ra  $k$  vector riêng tốt nhất mà có ảnh hưởng đến phong cách hoạt họa nhất của ma trận trọng số  $A^T A$ . Mỗi lớp khác nhau sẽ có ảnh hưởng khác nhau đến phong cách. Ta sẽ thực hiện việc kéo trên các vector riêng tương ứng tìm được để thay đổi phong cách theo ý muốn.
- Phương pháp có giám sát (Deep Danbooru [9]): Sử dụng mạng Deep Danbooru, một biến thể của mạng Resnet [10] được huấn luyện để gán nhãn cho ảnh hoạt họa đầu vào. Sau đó sử dụng một số phương pháp được nhóm áp dụng vào để tìm các vector tiềm ẩn tương ứng với nhãn đó. Sau đó đơn giản là thực hiện tinh chỉnh ảnh hoạt họa bằng các vector trên theo ý muốn.

Tập dữ liệu đào tạo mà nhóm tác giả GANs N' Roses đã sử dụng là tập dữ liệu selfie2anime [11]. Đây là tập dữ liệu phục vụ việc đào tạo các mô hình từ ảnh mặt người sang ảnh hoạt họa. Ở đề tài này, chúng tôi sẽ không thực hiện việc đào tạo thêm dữ liệu mà chỉ tập trung vào công việc đã đề cập ở trên. Bởi vì chúng tôi muốn chứng minh rằng quá trình thao túng miền không gian tiềm ẩn không làm thay đổi những điểm đặc sắc của mô hình có sẵn.



## 1.4 Đóng góp

Nhóm kỳ vọng những cải tiến, tinh chỉnh kỹ thuật thao túng miền không gian tiềm ẩn mà nhóm thực hiện có thể đạt được sự ổn định và mức độ tinh chỉnh cần thiết cho quá trình tạo ra ảnh hoạt họa phục vụ cho việc làm phim hoạt hình. Đồng thời cho thấy rằng kỹ thuật thao túng miền không gian tiềm ẩn này đáng tin cậy và có thể ứng dụng ở nhiều mô hình khác.

## Chương 2

# Phát biểu bài toán

## 2.1 Tổng quan về mô hình sinh mẫu

### 2.1.1 Khái niệm

Về cơ bản, mô hình sinh mẫu có thể tạo ra dữ liệu mới bằng cách học phân phối của tập dữ liệu mục tiêu.

Về mặt toán học: “Với một tập dữ liệu  $X$  và một tập nhãn  $Y$  tương ứng cho trước, mô hình sinh mẫu là mô hình thống kê của xác suất đồng thời của  $P(X, Y)$  hoặc  $P(X)$  nếu không tồn tại tập nhãn  $Y$ .”

Tổng quát hóa, một mô hình sinh mẫu có thể tái hiện lại phân phối của tập dữ liệu bằng cách tạo ra những dữ liệu giả trông giống như thật được lấy từ phân phối đó [12].

### 2.1.2 Khả năng của mô hình sinh mẫu

Một mô hình sinh mẫu có khả năng sẽ phải học những mối quan hệ tương quan giữa các biến như là “một con mèo thường xuyên có 4 chân”, “một con chim thường hay xuất hiện cùng với bầu trời”.

Một mô hình sinh mẫu cố gắng mô hình hóa dữ liệu bằng cách đặt chúng ở trong một miền không gian nào đấy. Chẳng hạn, cho một tập hình ảnh của số 0 và số 1 với tập nhãn tương ứng của chúng, mô hình sinh mẫu sẽ cố gắng tạo ra số 0 và số 1 giống với các mẫu thật của chúng trong miền không gian dữ liệu tức là nó phải giả lập mô hình phân phối cho không gian của dữ liệu [12].

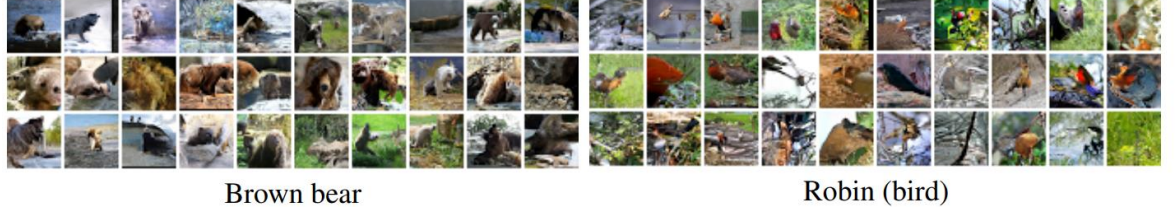
### 2.1.3 Các loại mô hình sinh mẫu:

#### 2.1.3.1 Các mạng Bayes:

Các mạng này sử dụng quy tắc nhân xích về xác suất để phân tích xác suất phân phối trên một vector thành một tích các phân tử của vector.

$$p_{\text{mô hình}}(\mathbf{x}) = \prod_{i=1}^n p_{\text{mô hình}}(x_i | x_1, \dots, x_{i-1}) \quad (2.1)$$

Mô hình phổ biến nhất của họ này là PixelCNN.

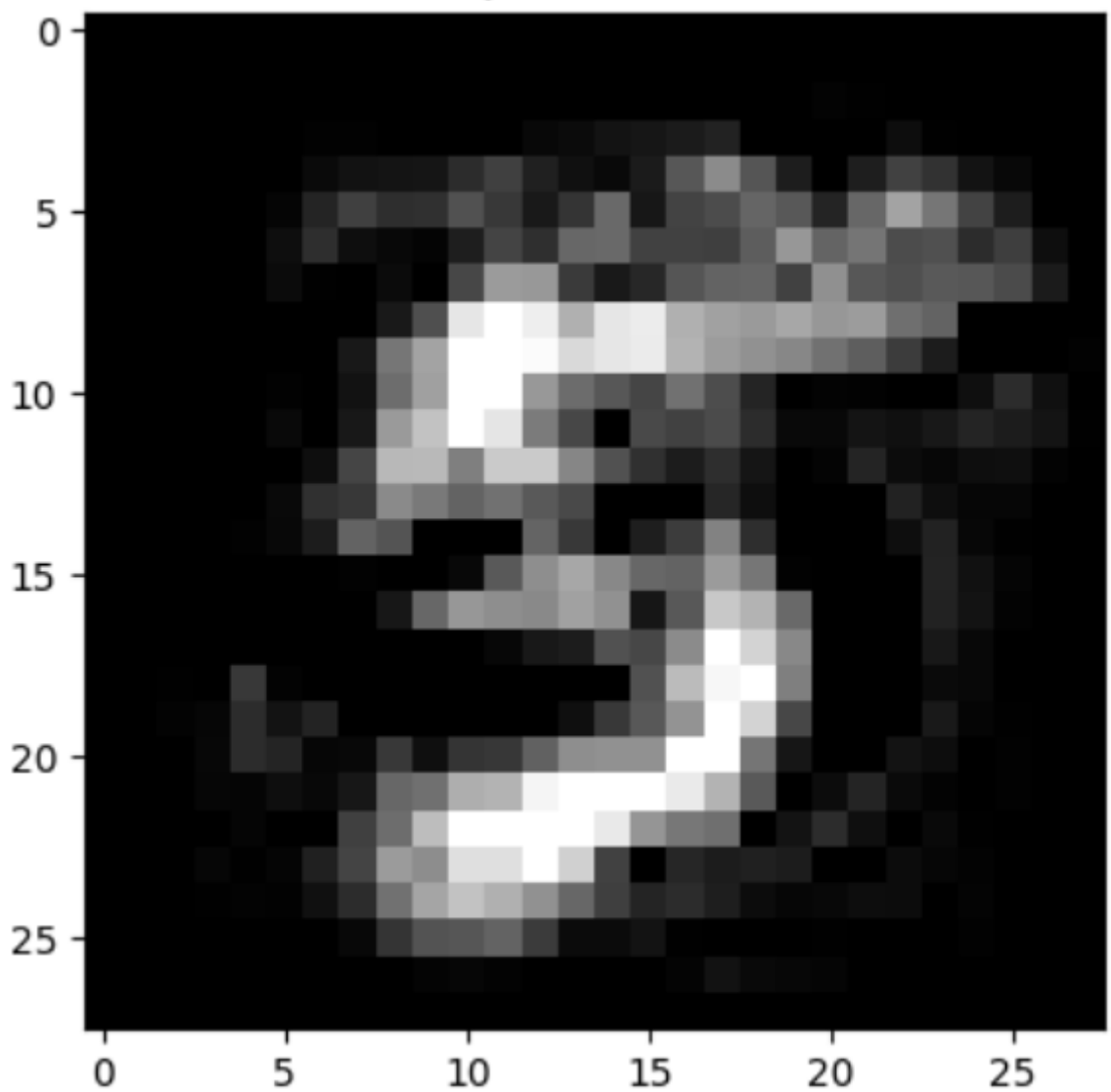


**Hình 2.1. Các mẫu sinh bởi mạng PixelCNN [13]**

#### 2.1.3.2 Bộ mã hóa tự động biến thể:

Bộ mã hóa này hoạt động bằng cách loại biến ngẫu nhiên  $z$  khỏi hàm mật độ  $\log p(\mathbf{x})$  thông qua việc xấp xỉ gần đúng. Mô hình sẽ tối đa hóa cận dưới về logarit likelihood của dữ liệu.

Điểm mạnh và hạn chế của phương pháp này là mô hình sẽ gần như giữ nguyên nếu như phân phối của  $p$  hoàn hảo và ngược lại. Một hạn chế khác là các mẫu được sinh ra thường có chất lượng kém.



**Hình 2.2 Một mẫu sinh của VAE [14]**

#### *2.1.3.3 Máy Boltzmann*

Máy Boltzmann là một loại mạng lặp ngẫu nhiên. Một máy Boltzmann có thể được định nghĩa bởi một hàm năng lượng và xác suất bởi một trạng thái, tỷ lệ với mỗi giá trị năng lượng. Máy Boltzmann sử dụng phương pháp Monte Carlo để chuyển các trạng thái thành phân phối xác suất. Chính vì điều này mà chúng gặp hạn chế là không thể sinh mẫu với số chiều không gian lớn.

#### 2.1.3.4 Mạng đối kháng sinh mẫu

Mạng đối kháng sinh mẫu khắc phục được các yếu điểm của các mô hình ở trên. Khác với mạng Bayes, mạng đối kháng sinh mẫu sử dụng mã tiềm ẩn và có thể sinh dữ liệu song song. So với bộ mã hóa tự động biến thể thì mô mạng đối kháng sinh mẫu gần như nhất quán. Hơn nữa, mô hình đối kháng sinh mẫu không sử dụng xích Markov. Và hơn hết là các mẫu mà mô hình đối kháng sinh mẫu sinh ra cho chất lượng tốt nhất so với phần còn lại. [15]

## 2.2 Tổng quan về bài toán chuyển đổi giữa các miền không gian ảnh bằng mô hình đối kháng sinh mẫu

### 2.2.1 Khái niệm

Chuyển đổi giữa các miền không gian ảnh là việc chuyển ảnh từ một nguồn gốc tới nguồn mục tiêu trong khi vẫn giữ được nguyên bản các nội dung của ảnh và phong cách của ảnh sau chuyển đổi mang hơi hướng của miền mục tiêu.

Vậy nội dung và phong cách của ảnh là gì?

Nội dung của ảnh bao gồm cấu trúc của các vật thể có trong ảnh.

Phong cách của ảnh có thể bao gồm màu sắc, các đường nét thể hiện cấu trúc vật thể, phong nền,...

Ví dụ, chuyển miền không gian từ ảnh chụp thực tế sang tranh vẽ theo phong cách của một số họa sĩ nổi tiếng:



**Hình 2.3 Chuyển đổi ảnh chụp phong cảnh sang tranh vẽ bằng  
CycleGAN[16]**

Hay chuyển từ ảnh bản đồ sang ảnh chụp từ vệ tinh:



**Hình 2.4 Chuyển miền ảnh bản đồ sang ảnh chụp từ vệ tinh với pix2pix [17]**

Quá trình chuyển miền ảnh có thể mô tả một cách toán học như sau:

$x_A$  thuộc miền ảnh gốc  $A$ ,

$B$  là miền ảnh mục tiêu,

Một ánh xạ  $G_{A \rightarrow B}$  sẽ tạo ra một ảnh  $x_{AB} \in B$  không thể phân biệt được so với ảnh  $x_B \in B$  với nguồn được lấy từ ảnh  $x_A \in A$ . Tổng kết quá trình chuyển miền ảnh sẽ là:



$$x_{AB} \in B: x_{AB} = G_{A \rightarrow B}(x_A) \quad (2.2)$$

### 2.2.2 Các dạng bài toán

Bài toán chuyển đổi giữa các miền không gian ảnh thường được chia thành 2 kiểu là có giám sát và không giám sát.

#### 2.2.2.1 Có giám sát

Đối với bài toán có giám sát, tập dữ liệu đầu vào có có từng cặp ảnh tương ứng với đầu vào và đầu ra của bài toán, ví dụ như bài toán chuyển từ ảnh mờ sang ảnh rõ nét tương ứng.

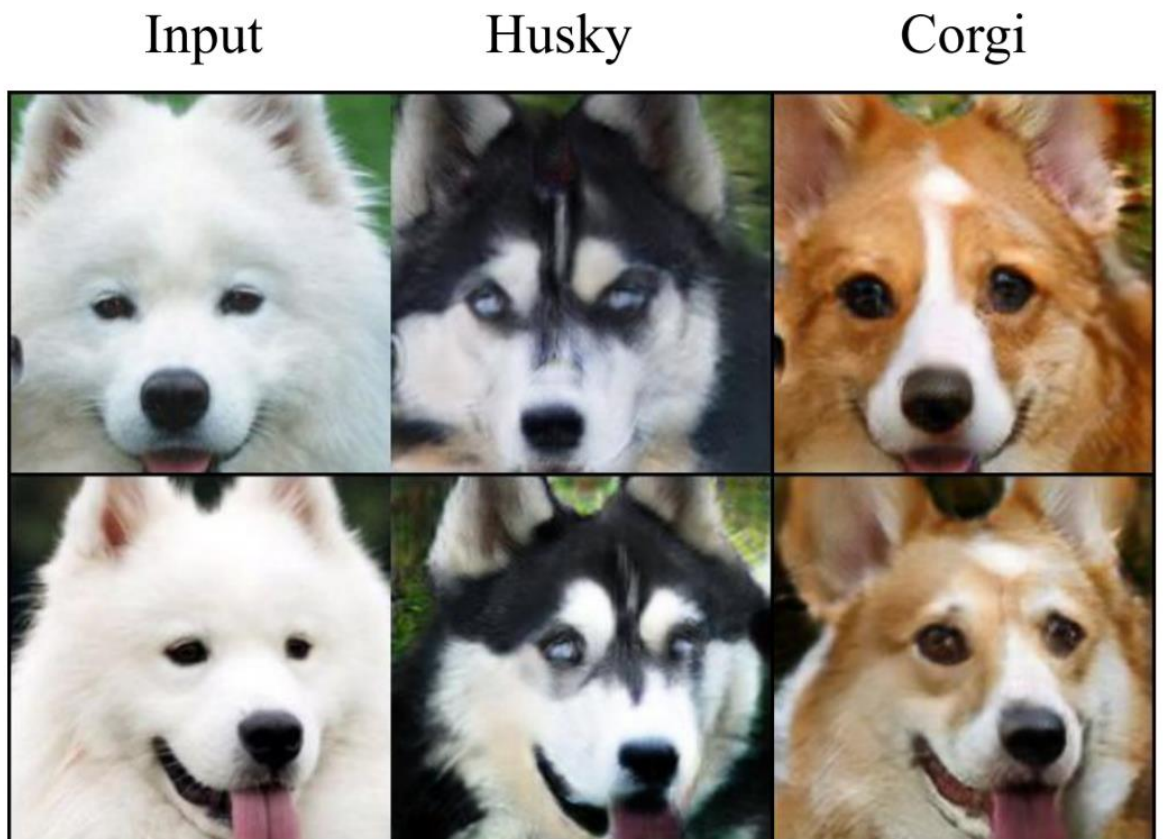


**Hình 2.5 Ảnh mờ và ảnh rõ nét tương ứng [18]**

#### 2.2.2.2 Không giám sát

Ở bài toán không giám sát, tập dữ liệu đầu vào không có từng cặp ảnh đầu vào và đầu ra, ví dụ như bài toán chuyển ảnh của một giống chó sang những giống chó khác, đối với bài toán này ta không thể có hình ảnh của một con samoyed khi chuyển sang giống husky hay corgi thì sẽ trông như thế nào để tạo thành cặp.





**Hình 2.6 Chuyển miền ảnh từ một giống chó này sang các giống chó khác**  
[19]

### 2.2.3 Các phương thức

Có 2 dạng phương thức cho kết quả của bài toán chuyển miền ảnh là uni-modal và multi-modal.

#### 2.2.3.1 *Uni-modal*

Uni-modal là kiểu mô hình chỉ cho một ảnh kết quả đầu ra với mỗi ảnh đầu vào, ví dụ như là chuyển ảnh phác thảo thành ảnh của một con pokemon [20].

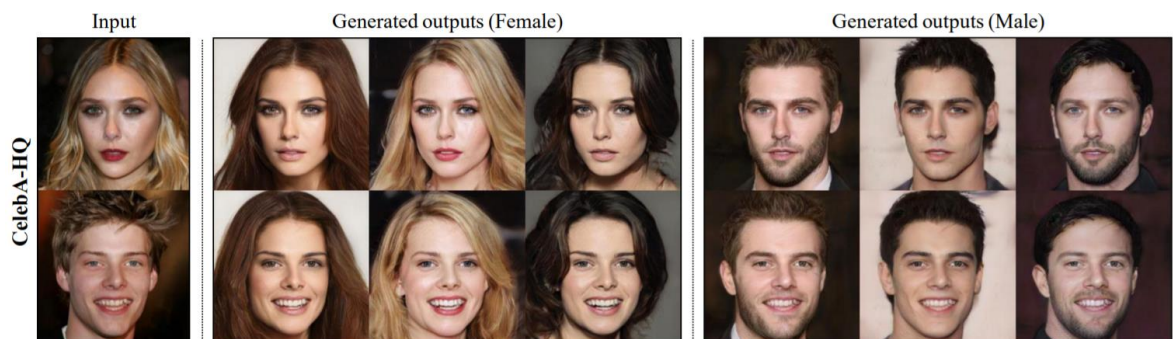
# Sketch → Pokemon



**Hình 2.7 Ứng dụng chuyển ảnh phác họa sang ảnh pokemon [17]**

## 2.2.3.2 Mutli-modal

Multi-modal là kiểu mô hình có thể cho nhiều ảnh kết quả đầu ra với mỗi ảnh đầu vào, ví dụ như là chuyển ảnh chụp khuôn mặt một người sang những khuôn mặt khác có phong cách khác nhau nhưng vẫn giữ kết cấu mặt.



**Hình 2.8 Chuyển ảnh khuôn mặt người [21]**

## **2.3 Bài toán chuyển miền ảnh mặt người sang ảnh hoạt họa**

### **2.3.1 Mô tả bài toán**

Đây là bài toán chuyển miền không gian ảnh một chiều với miền gốc là ảnh mặt người và miền đích là ảnh hoạt họa.

Ảnh mặt người chủ yếu là những bức ảnh selfie cận mặt hoặc chỉ có phần thân trên chứ không có toàn bộ người, những ảnh này đa dạng về sắc tộc, màu da, góc độ chụp, độ sáng, chất lượng ảnh (do chất lượng thiết bị thu hình).

Ảnh hoạt họa là những bức ảnh cận mặt hoặc chỉ có phần thân trên của những nhân vật anime, các nhân vật này đến từ nhiều bộ phim hoạt hình khác nhau, có những phong cách vẽ phong phú đa dạng, trải dài từ cổ điển đến hiện đại.

Khi chuyển miền không gian ảnh, ta kỳ vọng rằng những chi tiết thuộc về kết cấu, vị trí khuôn mặt người như là: vị trí các bộ phận mắt, miệng,... góc nghiêng của khuôn mặt, tóc,... được bảo toàn, còn những chi tiết thuộc về kiểu dáng, phong cách mang tính chất điện ảnh.

### **2.3.2 Các cách tiếp cận**

Do tính chất của ảnh hoạt họa (anime) đã được đề cập, rất khó để một mô hình sinh loại ảnh này được theo yêu cầu.

Gwern – một nhà khoa học máy tính đã dành thời gian để sinh mẫu ảnh anime bằng rất nhiều phương pháp đã rút ra một số kết luận.

Khi Gwern bắt đầu với DCGAN và khoảng 5,000 – 10,000 ảnh hoạt họa, kết quả vô cùng tệ hại, không thể nào tiệm cận với kết quả của DCGAN trên tập CelebA [22], ảnh sinh ra thường có những đốm màu đỏ hay hoàn toàn bị lỗi.

Sau đó ông có thử thêm rất nhiều phương pháp khác có thể kể đến như: StackGAN [23]/StackGAN++ [24], PixelRNN [25], PixelCNN[13], Wasserstein GAN [26], Glow [27], GAN-QP [28], MSG-GAN [29], SaGAN [30], VGAN [31], BigGAN [32]. Các mô hình trên đều cho ra một kết quả âm đạm khi ảnh anime được sinh ra

không đạt chất lượng yêu cầu. ProGAN[33] có thể nói là một ngoại lệ khi những mẫu được sinh ra đạt được mức độ giống với anime thật tuy nhiên phải đánh đổi thời gian huấn luyện rất lớn.

Chỉ đến khi StyleGAN [34] được áp dụng, mô hình của các nhà nghiên cứu của Nvidia không chỉ đạt chất lượng mẫu như ProGAN mà còn nhanh hơn đáng kể trong quá trình huấn luyện. StyleGAN học hiệu quả ở nhiều độ phân giải, cộng thêm việc kiểm soát ảnh được sinh ra bằng cách kiểm soát miền không gian tiềm ẩn của nó.

## **2.4 Bài toán kiểm soát miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu**

### **2.4.1 Khái niệm**

#### a) Thể hiện tiềm ẩn của một ảnh

Một ảnh RGB với kích thước  $256 \times 256$  sẽ cần lưu ở trong một mảng có số chiều là  $(256, 256, 3)$  (3 kênh màu). Như vậy để mô tả ảnh này ta cần có tới  $256 \times 256 \times 3 = 196608$  chiều dữ liệu.

Tuy nhiên, ta có một cách nén nào đó cho lưu 196608 chiều dữ liệu này về một không gian 4 chiều, ví dụ vector này có dạng  $\vec{v}_1 = [2, 5, 15, 7]$ . Vector này được gọi là một thể hiện tiềm ẩn của ảnh ban đầu.

#### b) Không gian tiềm ẩn

Vector được mô tả ở trên và các vector cùng họ với nó hợp thành không gian tiềm ẩn. Trong miền không gian này, những ảnh giống nhau như là ảnh của 2 cái xe đạp sẽ có vector tiềm ẩn tương trưng cho chúng nằm ở gần nhau, trong khi những ảnh khác nhau như là xe đạp và máy bay thì ngược lại.

Trong mô hình đối kháng sinh mẫu, không gian tiềm ẩn là một không gian giả định có chứa các vector (là thể hiện của ảnh) mà bộ sinh mẫu có thể chuyển từ nó sang một ảnh nào đó.

#### c) Không gian $Z$

Không gian  $Z$  là không gian chứa tất cả các vector  $z$ , là vector bao gồm các giá trị ngẫu nhiên từ phân phối chuẩn.

Nguyên nhân cho sự tồn tại của không gian  $Z$  là mạng học sâu thông thường có thể học rất tốt cách khớp một đầu vào  $X$  với một đầu ra  $Y$  nào đó, tuy nhiên không thể tạo ra dữ liệu mới. Giải pháp được đưa ra là sử dụng nhiễu ngẫu nhiên từ một phân phối xác suất đơn giản và sử dụng mạng học sâu để biến đổi phân phối xác suất ngẫu nhiên đơn giản này thành phân phối của dữ liệu huấn luyện. [35]

d) Mã phong cách/vector phong cách

GAN thông thường bị giới hạn bởi việc chỉ có thể tạo ra những bản sao của bộ dữ liệu huấn luyện trong khi StyleGAN cho phép sinh ra các mẫu trung thực hơn với nhiều biến thể về phong nền, kiểu tóc, kính,...

StyleGAN đạt được điều này bằng cách: thay vì truyền vector  $z$  thẳng vào bộ sinh mẫu, StyleGAN sẽ truyền nó vào một mạng ánh xạ để tạo ra một vector  $w$  (vector phong cách/mã phong cách). vector này sau đó được truyền vào bộ sinh mẫu ở nhiều những lớp khác nhau.

e) Không gian  $W$

Tương tự như không gian  $Z$ , không gian  $W$  là không gian chứa tất cả các vector  $w$  đã được định nghĩa ở trên, không gian  $W$  hay còn được gọi là miền không gian phong cách. Không gian  $W$  đóng vai trò quan trọng trong StyleGAN vì nó cho phép người ta kiểm soát các thuộc tính/chi tiết của ảnh. Nguyên nhân vì không gian  $W$  không bị xáo trộn, tức là một vùng không gian nào đó của không  $W$  sẽ chịu trách nhiệm cho một phần thông tin duy nhất nào đó của ảnh. Chẳng hạn, miền không gian đầu tiên có thể quyết định màu tóc của nhân vật, miền không gian thứ 2 có thể quyết định kích cỡ mắt của nhân vật,... Vì thế người ta có thể chỉnh sửa ảnh của một người từ màu tóc đen sang màu tóc xanh từ việc biến đổi vector  $w$  một cách thích hợp.

### 2.4.2 Kiểm soát miền không gian tiềm ẩn

Về cơ bản, ta thực hiện việc kiểm soát miền không gian tiềm ẩn bằng cách biến đổi vector  $w$  theo những hướng trong miền không gian tiềm ẩn mà cho ta kết quả mong đợi trên ảnh được tạo ra.

#### a) Ánh xạ ảnh vào miền không gian tiềm ẩn

Đây là quá trình tìm một thể hiện tiềm ẩn của một ảnh cho trước trong miền không gian phong cách  $W$  nghĩa là ta sẽ tìm một vector phong cách  $w$  nào đó mà khi truyền vector này vào bộ sinh mẫu của StyleGAN sẽ cho ta một ảnh giống y hệt ảnh ban đầu.

#### b) Các cách tiếp cận

- Không giám sát: một cách tiếp cận vô cùng sáng tạo được giới thiệu vào năm 2020 tên gọi là SeFa [7], bằng cách tìm  $k$  vector riêng có ảnh hưởng nhất đến phong cách hoạt hóa của ma trận  $A^T A$  với  $A$  là ma trận trọng số của bộ sinh mẫu. Sau đó sử dụng các vector tìm được này để thay đổi vector phong cách  $w$ .
- Có giám sát: bằng cách sử dụng một mô hình khác để gán nhãn những ảnh được tạo ra, ta có thể dùng các kỹ thuật như là hồi quy, phân rã ma trận hay SVM [36],... để tìm ra những vector  $w$  ứng với các nhãn.

# Cơ sở lý thuyết

### 3.1.1 Định nghĩa

- Generative network (mạng sinh - Generator): Tạo ra dữ liệu giả và mục tiêu là sinh ra được các dữ liệu giống thật nhất, có khả năng đánh lừa được Discriminative network.

### 3.1.2 Kiến trúc

```
graph LR; RI[Real images] --> S1[Sample]; RI --> D[Discriminator]; G[Generator] --> S2[Sample]; S1 --> D; S2 --> D; D --> DL[Discriminator loss]; D --> GL[Generator loss];
```

The diagram illustrates the architecture of a Generative Adversarial Network (GAN). It shows the flow of data and the calculation of losses. Real images are sampled and fed into the discriminator. The generator takes random input to produce samples, which are also fed into the discriminator. The discriminator outputs two losses: the discriminator loss and the generator loss.

a) Bộ phân biệt (Discriminator):

Bộ phân biệt của GANs là một mô hình phân lớp. Nó sẽ cố gắng phân lớp ảnh đầu vào để quyết định rằng đây là ảnh thật hay là ảnh giả (ảnh được sinh ra bởi bộ sinh mẫu).

Trong quá trình huấn luyện, bộ phân biệt sẽ được truyền dữ liệu thật và giả (được sinh từ bộ sinh mẫu).

Độ mất mát của bộ phân biệt sẽ phạt bộ phân biệt nếu nó nhầm lẫn giữa ảnh thật và ảnh giả.

Bộ phân biệt cập nhật trọng số nhờ vào quá trình lan truyền ngược từ độ lỗi qua mạng phân biệt của nó.

b) Bộ sinh mẫu (Generator):

Bộ sinh mẫu là mô hình phụ trách tạo ra dữ liệu giả từ phản hồi của bộ phân biệt. Nó sẽ cố gắng khiến cho bộ phân biệt phân lớp nhầm ảnh được sinh ra là ảnh thật.

Để huấn luyện bộ sinh mẫu thì các phần của GAN được sử dụng bao gồm:

- Đầu vào ngẫu nhiên.
- Bộ sinh mẫu.
- Bộ phân biệt.
- Đầu ra của bộ phân biệt.
- Độ mất mát của bộ sinh mẫu sẽ phạt bộ sinh mẫu nếu nó không đánh lừa được bộ phân biệt.

Nguyên nhân của việc phải sử dụng đầu vào ngẫu nhiên là nhờ đó ta có thể khiến cho mô hình tạo ra đa dạng các dữ liệu, lấy các mẫu đa dạng từ phân phối mục tiêu.

Tuy nhiên đó không phải là tất cả, GAN dùng 1 mẹo là: **sử dụng bộ phân biệt để huấn luyện bộ sinh mẫu.**



Thông thường để huấn luyện một mạng trí tuệ nhân tạo, ta cập nhật trọng số của mạng để giảm thiểu độ lỗi của mô hình. Tuy nhiên, bộ sinh mẫu trong GAN không trực tiếp kết nối với độ lỗi mà chúng ta đang muốn tạo ảnh hưởng mà nó kết nối thông qua bộ phân biệt.

Như vậy quá trình huấn luyện bộ sinh mẫu có thể được mô tả như sau:

1. Lấy nhiều ngẫu nhiên.
2. Bộ sinh mẫu tạo đầu ra từ nhiều ngẫu nhiên.
3. Bộ phân biệt phân lớp đầu ra từ bộ sinh mẫu.
4. Tính độ lỗi của bộ phân biệt.
5. Lan truyền ngược trên cả 2 bộ sinh mẫu và bộ phân biệt để lấy gradient.
6. Sử dụng gradient để cập nhật trọng số cho bộ sinh mẫu [12].

#### c) Huấn luyện GAN

Như vậy có thể tổng kết quá trình huấn luyện một mô hình đối kháng sinh mẫu như sau: Giả sử dữ liệu của chúng ta ở dạng hình ảnh (image), Generator sẽ tạo ra các hình ảnh giả (fake image), Discriminator sẽ học cách phân biệt thật giả. Discriminator càng cố gắng phân biệt thật giả thì Generator sẽ dựa vào feedback (output của Discriminator) để cải thiện khả năng tạo hình ảnh giả của mình, từ đó tăng khả năng đánh lừa Discriminator. Quá trình sẽ được lặp đi lặp lại, Generator tăng khả năng tạo ảnh giả, Discriminator cải thiện khả năng phân biệt thật giả. Đến khi nào Discriminator không thể phân biệt được đâu là ảnh thật và đâu là ảnh giả do Generator tạo ra thì quá trình dừng lại.

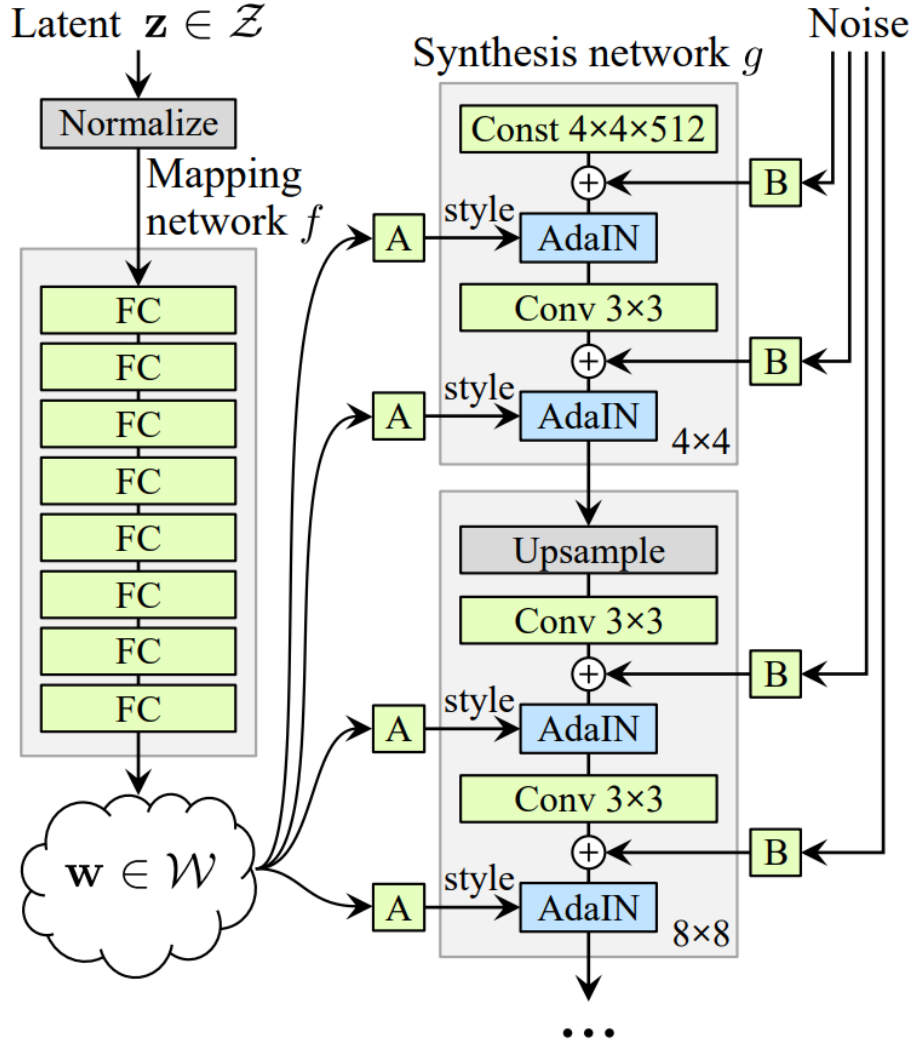
## 3.2 StyleGAN

### 3.2.1 Định nghĩa

StyleGAN là một kiến trúc GAN mà các nhà nghiên cứu của Nvidia đã thiết kế lại bộ sinh mẫu trong mô hình nhằm kiểm soát quá trình sinh ảnh, từ tổng quan (khung nền, kiểu tóc, góc nghiêng mặt,...) tới các chi tiết (màu mắt, màu tóc,...).

Các kỹ thuật quan trọng trong StyleGAN bao gồm:

- Progressive Growing.
- Mạng ánh xạ.
- AdaIN.
- Stochastic variation.



Hình 3.2 Kiến trúc mạng sinh được sử dụng trong StyleGAN [34]

### 3.2.2 Progressive Growing

Đây là một kỹ thuật lần được được giới thiệu trong PGGAN [33], trong kỹ thuật này cả bộ sinh mẫu và bộ phân biệt ban đầu được huấn luyện trên ảnh với độ phân giải  $4 \times 4$ . Khi mô hình đã đạt được độ ổn định nhất định thì sẽ được huấn luyện với ảnh gấp đôi độ phân giải  $8 \times 8$ ,  $16 \times 16$ , ... Một khối sinh sẽ được thêm vào mỗi lần kích thước ảnh tăng lên.

### 3.2.3 Mạng ánh xạ

Đối với một không gian tiềm ẩn, ta mong muốn nó bao gồm những không gian con tuyến tính, với mỗi không gian con này kiểm soát một nhân tố của ảnh được sinh ra. Tuy nhiên nếu sử dụng không gian  $Z$  thì xác suất lấy mẫu từ  $Z$  phải tương ứng với các nhân tố tương ứng trong tập dữ liệu huấn luyện  $\rightarrow$  điều này là rất khó để xảy ra.

Mạng ánh xạ (mapping network) lấy đầu vào là biến ngẫu nhiên từ một phân phối đơn giản – không gian  $Z$  (ở kiến trúc StyleGAN là phân phối chuẩn) và ánh xạ nó vào không gian tiềm ẩn trung gian  $W$  (hay còn gọi là không gian phong cách). Nhờ quá trình học các mẫu liên tục của mạng ánh xạ này mà  $W$  có thể thích ứng với phân phối của dữ liệu huấn luyện. Việc ánh xạ này “mở”  $W$  để khiến cho các nhân tố trở nên tuyến tính hay nói cách khác là làm cho không gian  $W$  linh hoạt hơn, dễ kiểm soát hơn. Các nhà nghiên cứu của Nvidia cho rằng chính mạng ánh xạ này đã tạo áp lực lên bộ sinh mẫu vì sẽ dễ dàng để sinh ảnh trông thật hơn nếu dựa trên một không gian linh hoạt thay vì một không gian không gian rời rạc.

### 3.2.4 AdaIN (Adaptive Instance Normalization)

AdaIN giữ vai trò chuyển thông tin từ vector phong cách  $w$  hợp với các đầu ra của lớp tích chập tạo thành một biểu hiện trực quan (ảnh) của mạng sinh.

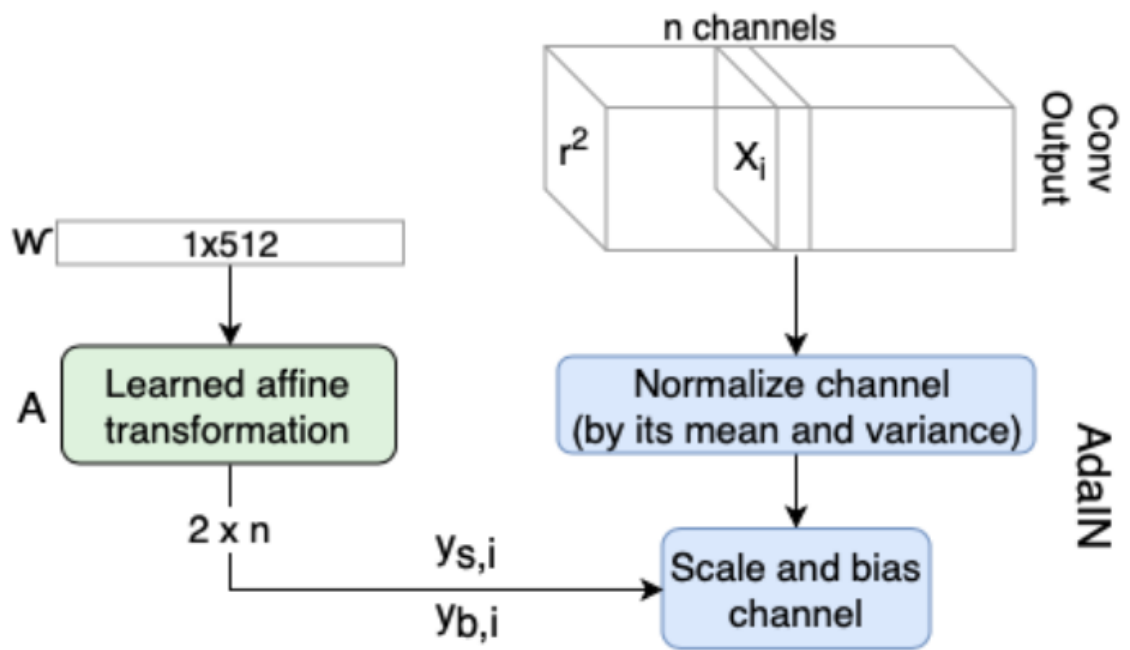
Các phép biến đổi affine sẽ chuyên biệt hóa vector phong cách  $w$  thành  $y = (y_s, y_b)$ , với  $y_s, y_b$  lần lượt là nhân tố scale và bias sẽ kiểm soát quá trình AdaIN.

Cơ chế của AdaIN như sau:

$$\text{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i} \quad (3.1)$$

Với mỗi  $\mathbf{x}_i$  là đầu ra của lớp tích chập nằm trong các khối sinh.

Sau mỗi lớp tích chập, ta cần chuẩn hóa đầu ra (feature map) theo phản tương ứng của vector phong cách  $w$ , đó là lý do tại sao mô-đun AdaIN lại được sử dụng.

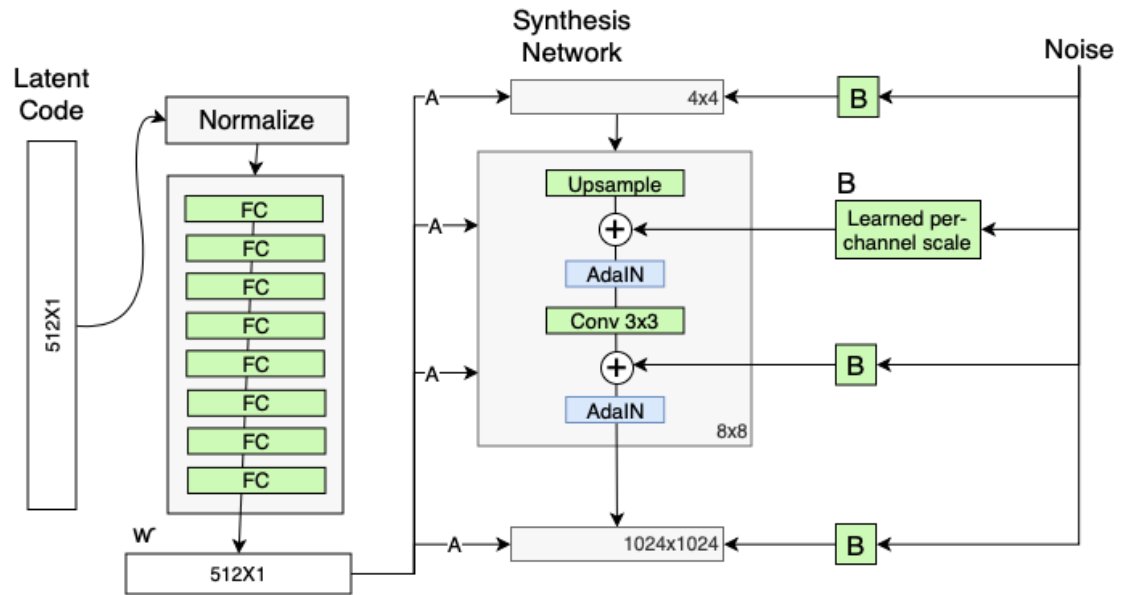


Hình 3.3. Trực quan quá trình AdaIN [37]

### 3.2.5 Stochastic variation

Stochastic variation – biến thể ngẫu nhiên là một cải tiến cho phép StyleGAN có thể sinh ra những ảnh có sự đa dạng về các chi tiết hơn so với tập dữ liệu huấn luyện, các chi tiết này có thể là: tàn nhang, nếp nhăn, khuyết tai, mắt kính,...

StyleGAN thực hiện điều này bằng cách thêm nhiễu ngẫu nhiên vào trước mô-đun AdaIN để thay đổi một ít chi tiết trên biểu hiện trực quan (ảnh) của mạng sinh.



**Hình 3.4 Thêm nhiễu vào mỗi khối sinh của mạng sinh [37]**

## 3.3 StyleGAN2

StyleGAN2 là một phiên bản cải tiến của StyleGAN ở các điểm sau:

- Thay AdaIN bằng phép chuẩn hóa trọng số của mạng tích chập.
- Cải tiến việc ảnh sinh ra không tự nhiên do Progressive Growing.
- Tăng chất lượng ảnh sinh ra bằng cách cung cấp tính liên tục cho không gian tiềm ẩn.

### 3.3.1 Chuẩn hóa thay vì sử dụng AdaIN

#### 3.3.1.1 Vấn đề của AdaIN

Khi sinh ảnh, StyleGAN có xu hướng tạo ra các nhiễu có hình dạng như một giọt nước. Khi các nhà nghiên cứu của Nvidia loại bỏ AdaIN ra khỏi mô hình thì những nhiễu này cũng biến mất theo.



**Hình 3.5 AdaIN tạo ra nhiễu trong ảnh sinh ra bởi StyleGAN [1]**

Chính vì vấn đề này mà một cách chuẩn hóa mới đã được sử dụng trong StyleGAN2.

#### 3.3.1.2 Chuẩn hóa

Thay vì chuẩn hóa bằng cách sử dụng dữ liệu thật, trong StyleGAN2 người ta chuẩn hóa bằng cách sử dụng dữ liệu xấp xỉ. Xem lại công thức 3.1, AdaIN được thực hiện gồm các bước như sau: chuẩn hóa feature map  $x_i \rightarrow$  biến đổi tuyến tính bằng  $y_s, y_b$  được tạo ra bởi các biến đổi affine đã học trên vector  $w$ . Tuy nhiên, AdaIN nằm trong một khối sinh nên thứ tự của quá trình sẽ là như sau: biến đổi tuyến tính bằng  $y_s, y_b$  được tạo ra bởi các biến đổi affine đã học trên vector  $w \rightarrow$  Tích chập  $\rightarrow$  chuẩn hóa feature map.

Đối với StyleGAN2 quá trình này được thực hiện hoàn toàn trên trọng số của các lớp tích chập – tức là dữ liệu xấp xỉ, chứ không hoạt động trên feature map như là AdaIN.

Quá trình biến đổi tuyến tính scale bằng  $y_s$  có thể thay thế bằng cách nhân một hệ số  $s_i$  vào trọng số tích chập thay vì nhân vào feature map đầu ra của lớp tích chập:

$$w'_{ijk} = s_i \cdot w_{ijk} \quad (3.2)$$

Với  $w$  và  $w'$  lần lượt là là trọng số ban đầu và trọng số sau khi biến đổi của lớp tích chập,  $s_i$  là giá trị cho tỉ lệ từng đầu vào của feature map thứ  $i$ ,  $j$  và  $k$  là chỉ số hàng và cột lớp tích chập. Như vậy quá trình này đã thay thế cho 2 bước đầu tiên của AdaIN.

Tiếp theo quá trình chuẩn hóa được thực hiện như sau:

Giả sử rằng input của quá trình tích chập có phân phối chuẩn, độ lệch chuẩn của kết quả đầu ra sẽ là:

$$\sigma_j = \sqrt{\sum_{i,k} w'_{ijk}{}^2} \quad (3.3)$$

Để chuẩn hóa kết quả đầu ra theo đơn vị của độ lệch chuẩn thì ta chỉ cần nhân với một tỷ lệ  $1/\sigma_j$ . Như vậy ta có thể tích hợp điều này vào trọng số của lớp tích chập:

$$w''_{ijk} = \frac{w'_{ijk}}{\sqrt{\sum_{i,k} w'_{ijk}{}^2 + \epsilon}} \quad (3.4)$$

Với  $\epsilon$  là một hằng có giá trị rất nhỏ nhằm tránh việc chia cho 0.

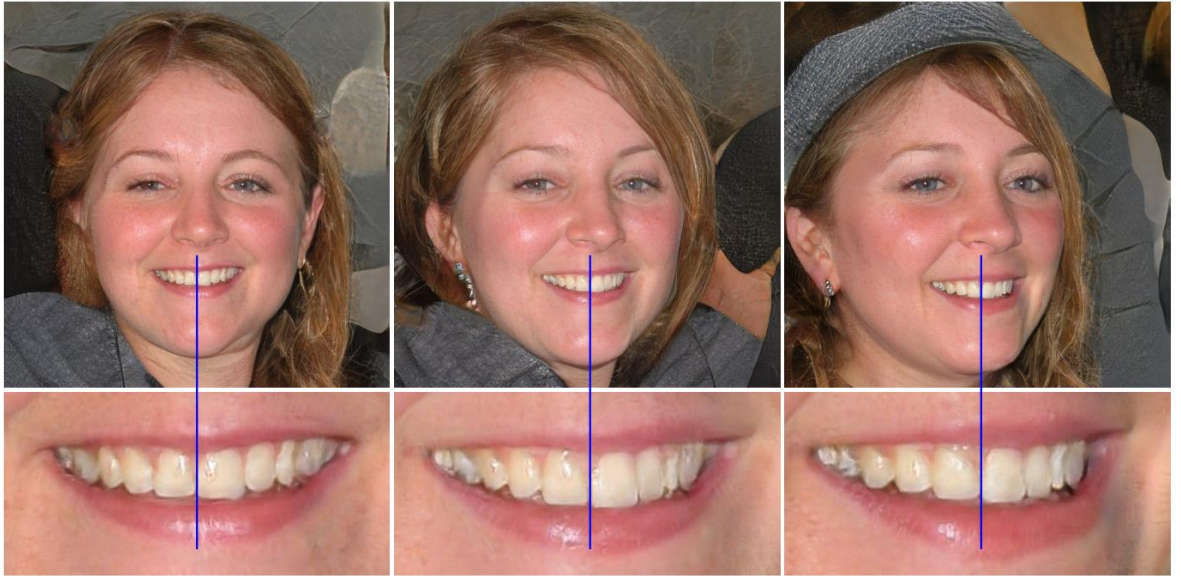
Như vậy cả 3 bước của AdaIN đều được thay thế bằng các chỉnh sửa trong việc thực hiện tích chập.



### 3.3.2 Giải quyết vấn đề của Progressive Growing

#### 3.3.2.1 Vấn đề của Progressive Growing

Progressive Growing khiến cho bộ sinh mẫu có một thiên kiến về vị trí của những chi tiết trên ảnh. Như ở ví dụ bên dưới, tuy tư thế của khuôn mặt thay đổi nhưng răng của nhân vật vẫn giữ nguyên:

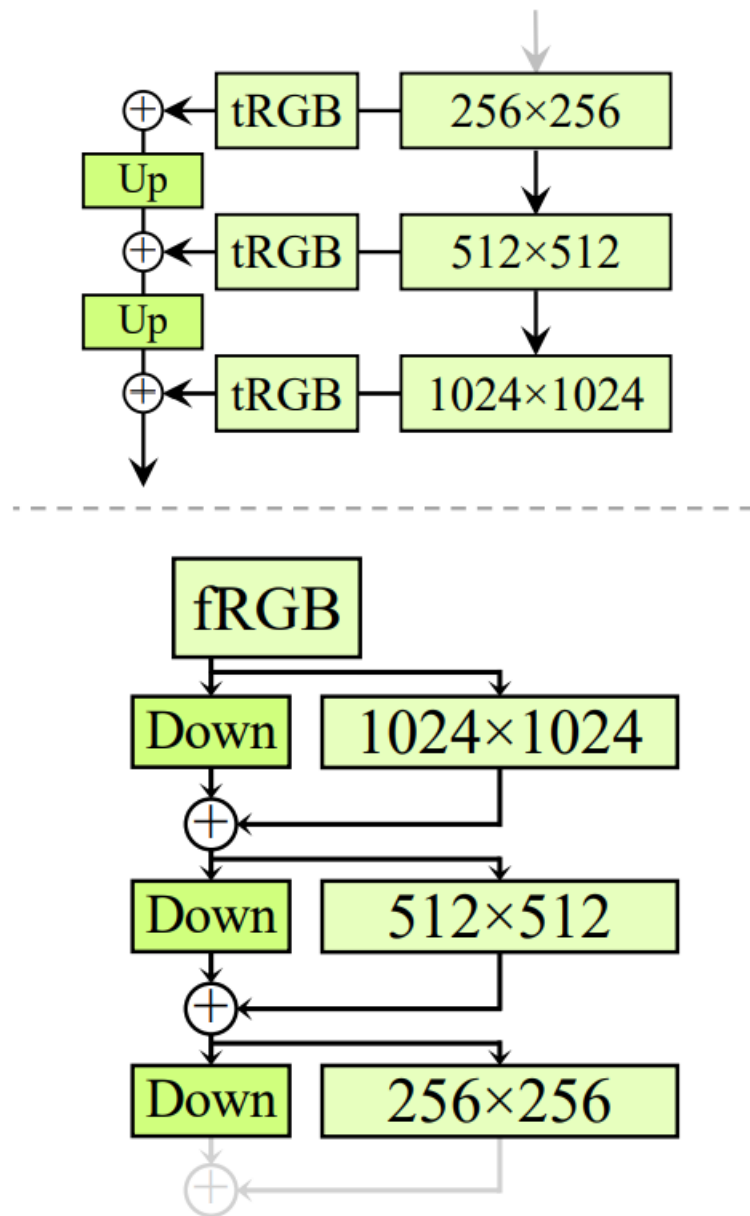


**Hình 3.6 Vấn đề của Progressive growing [1]**

Chính vì điều này mà các tác giả của StyleGAN2 đã thay thế progressive growing bằng một cấu trúc mạng riêng.

#### 3.3.2.2 Giải pháp

Không giống như progressive growing – sẽ thêm dần các bộ sinh mẫu và bộ phân biệt cho các ảnh có độ phân giải cao hơn. Mạng được các nhà nghiên cứu của Nvidia đề xuất có phần tương tự như MSG-GAN [29]:



**Hình 3.7 Kiến trúc được sử dụng trong StyleGAN2**

Với khối nằm trên đường gạch nối là kiến trúc của bộ sinh mẫu, còn ở dưới là kiến trúc bộ phân biệt. **Up** và **Down** tượng trưng cho up sampling và down sampling, **tRGB** và **fRGB** chuyển đổi giữa RGB và dữ liệu nhiều chiều trên mỗi điểm ảnh.

Kết quả của cải tiến này mang lại giúp cho các chi tiết của ảnh sinh ra dịch chuyển theo cùng hướng với hướng của khuôn mặt.



**Hình 3.8 So sánh các chi tiết khuôn mặt của ảnh sinh bởi StyleGAN và StyleGAN2 [38]**

### 3.3.3 Cung cấp tính liên tục cho không gian tiềm ẩn

StyleGAN2 cung cấp tính liên tục cho miền không gian tiềm ẩn bằng cách thêm một độ lỗi vào Generator:

$$\mathcal{L}_{pl} = \mathbb{E}_{\mathbf{w}} \mathbb{E}_{\mathbf{y}} (\|\mathbf{J}_{\mathbf{w}}^T \mathbf{y}\| - a)^2 \quad (3.5)$$

Với  $\mathbf{J}_{\mathbf{w}}$  là ma trận Jacobian  $\mathbf{J}_{\mathbf{w}} = \partial g(\mathbf{w}) / \partial \mathbf{w}$ .

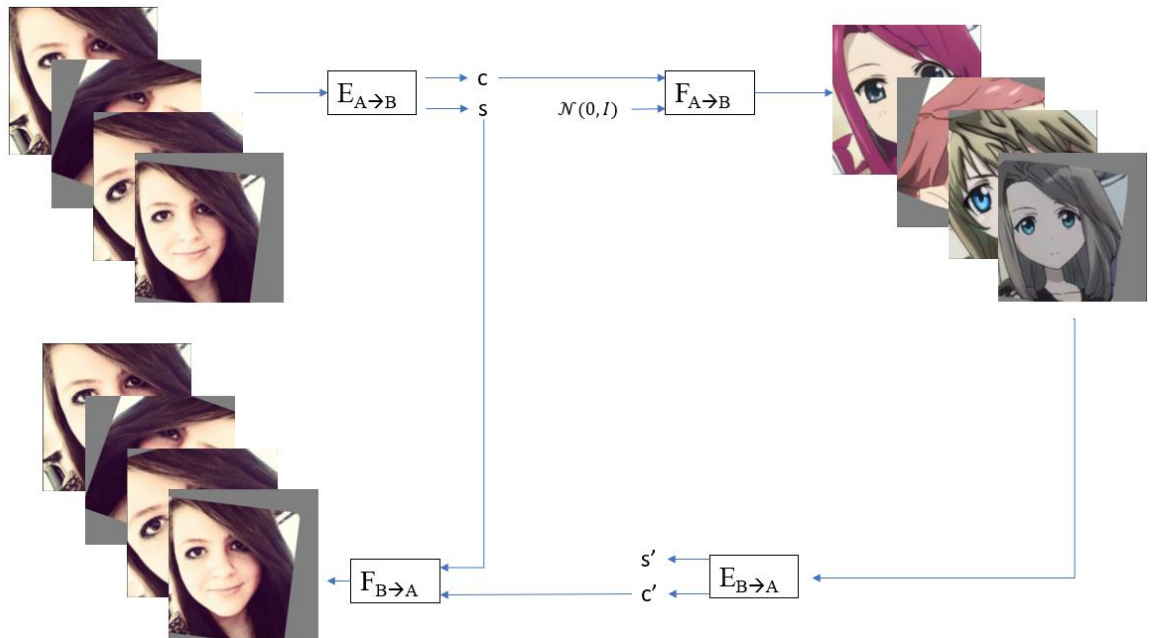
Bằng cách này, Generator sẽ buộc phải giảm thiểu những thay đổi của sự xáo trộn các biến tiềm ẩn nhất có thể.

Nhờ điều này mà mô hình hoạt động nhất quán và đáng tin cậy hơn, giúp khám phá kiến trúc dễ dàng hơn cũng như việc khám phá miền không gian tiềm ẩn một cách trơn tru hơn. [38]

### 3.4 GANs N' Roses

GANs N' Roses là một framework được thiết kế để chuyển miền ảnh mặt người sang ảnh hoạt họa với trọng tâm tối đa sự đa dạng của các phong cách ảnh hoạt họa được sinh ra. Framework được thiết kế dựa trên kiến trúc Encoder-Decoder, với bộ Decoder  $F$  là bộ Generator sử dụng kiến trúc StyleGAN2.

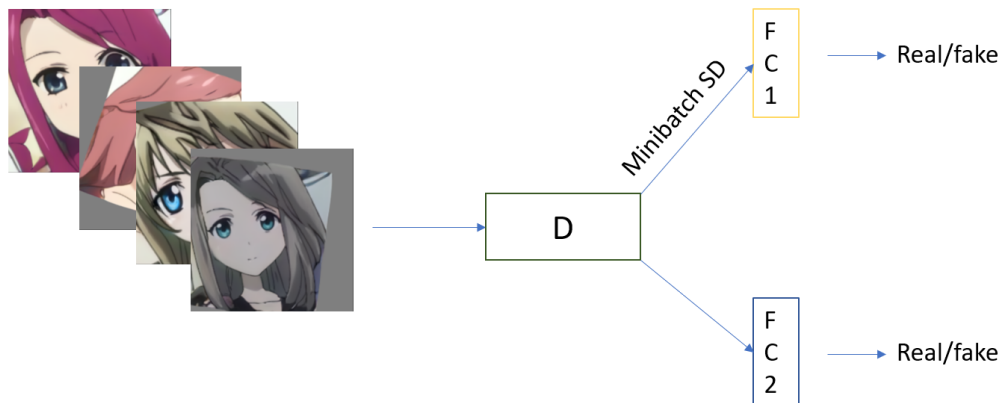
Ý tưởng chính của GANs N' Rose là định nghĩa nội dung là vị trí của sự vật còn phong cách là kiểu dáng tức là sự vật trông như thế nào. Việc phân biệt nội dung và phong cách có thể đạt được thông qua phương pháp tăng cường dữ liệu. Chọn một tập các phép tăng cường dữ liệu tương ứng mà khiến cho phong cách không thay đổi dưới những phép tăng cường này, còn nội dung thì có.



**Hình 3.9. Kiến trúc Generator được sử dụng [3]**

Bộ generator hoạt động theo cả 2 chiều thuận và nghịch. Để đảm bảo rằng mã nội dung được giữ nguyên sau quá trình encode và decode ảnh hoạt họa.

### Discriminator

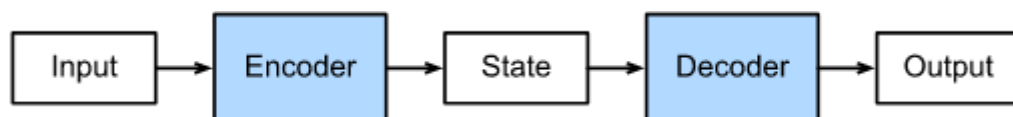


**Hình 3.10. Kiến trúc của bộ Discriminator được sử dụng [3]**

Bộ discriminator trong kiến trúc của GANs N' Rose có 2 nhánh phân biệt. Một là các ảnh riêng lẻ, hai là một batch các ảnh hoạt họa được sinh ra khi áp dụng tăng cường dữ liệu vào ảnh gốc. Nguyên nhân nhánh thứ hai xuất hiện vì GAN thường có xu hướng chỉ lấy 1 tập con của tập dữ liệu huấn luyện, chính vì thế giảm độ đa dạng của dữ liệu được sinh ra. Sử dụng phương pháp được đề cập trong PGGAN [1], ở lớp áp chót của bộ discriminator của GANs N' Rose được tính toán độ lệch chuẩn cho từng thuộc tính sau đó truyền chúng vào một lớp FC.

#### 3.4.1 Kiến trúc Encoder – Decoder

Kiến trúc Encoder – Decoder được giới thiệu lần đầu vào năm 2015 [2] và được ứng dụng rộng rãi trong lĩnh vực học sâu. Phương pháp ban đầu đề xuất một mạng LSTM [39] xử lý chuỗi đầu vào thành một vector với số chiều cố định. Vector này sau đó được chuyển thành một chuỗi đầu ra bởi một mạng LSTM khác. Kiến trúc này được sử dụng rộng rãi trong các tác vụ như dịch máy bởi nó tốt hơn bất kỳ phương pháp thống kê cổ điển nào khác.



**Hình 3.11. Mô hình Encoder – Decoder [40]**

Kiến trúc Encoder – Decoder GANs được ra mắt giới nghiên cứu lần đầu trong ALI [41] và BiGAN [42]. Như mô hình GANs tiêu chuẩn, kiến trúc này bao gồm một 2 phần: một là bộ phân biệt Discriminator  $D$ , cho ra đầu ra là một số thuộc khoảng  $[0,1]$ , nhằm đánh giá mức độ thỏa mãn của dữ liệu đầu vào (dữ liệu được sinh ra) so với dữ liệu thật. Hai là bộ sinh mẫu Generator  $G$  – Decoder  $F$ . Tùy thuộc vào kiến trúc mô hình mà bộ Encoder  $E$  có thể hỗ trợ bộ sinh mẫu Generator  $G$  hay bộ phân biệt Discriminator  $D$ .

Ta kỳ vọng rằng Encoder và Decoder sẽ nghịch đảo cho nhau tức là:

$E(G(z)) \approx z$  và  $G(E(x)) \approx x$ , và phân phối của  $(z, G(z))$  và  $(E(x), x)$  gần như giống nhau. Nếu giả định trên xảy ra thì bộ sinh mẫu Generator  $G$  phải cho những đầu ra tương trưng cho giá trị  $E(x)$  có ý nghĩa.

Đối với bài toán của chúng ta, với 2 miền  $\mathcal{X}$  và  $\mathcal{Y}$  lần lượt là miền ảnh mặt người thật selfie và miền ảnh mặt anime, gọi  $x$  là một mẫu ngẫu nhiên thuộc miền ảnh mặt người:  $x \in \mathcal{X}$ , mục tiêu là tạo ra một tập  $\hat{Y} \in \mathcal{Y}$  đa dạng các ảnh hoạt họa có nội dung tương ứng với ảnh gốc  $x$ .

Bộ sinh mẫu của GAN's N' Roses được tạo nên bởi một cặp Encoder  $E$  và Decoder  $F$ , cho mỗi hướng của việc chuyển miền ảnh  $\mathcal{X} \rightarrow \mathcal{Y}$  và  $\mathcal{Y} \rightarrow \mathcal{X}$ .

Đối với chiều thuận tức là  $\mathcal{X} \rightarrow \mathcal{Y}$ , Encoder  $E$  sẽ tách  $x$ , thành content code  $c(x)$  và style code  $s(x)$ . Decoder  $F$  sử dụng  $c(x)$  và một style code  $s_z$  ngẫu nhiên để tạo thành một ảnh thích hợp  $\hat{y}$  thuộc miền  $\mathcal{Y}$ .

Đối với chiều nghịch (để tính độ lỗi), Encoder  $E$  tách ảnh mặt anime mới được tạo thành  $\hat{y}$  thành content code  $c(\hat{y})$  và style code  $s(\hat{y})$ . Decoder  $F$  sử dụng content code của ảnh anime mới được tạo thành  $c(\hat{y})$  và style code của ảnh mặt người thật ban đầu  $s(x)$  nhằm tái tạo ảnh mặt người thật ban đầu.

Như vậy, nội dung ảnh được sinh ra sẽ được kiểm soát bởi content code và phong cách của nó sẽ được kiểm soát bởi style code.

### 3.4.2 Đảm bảo đa dạng phong cách

Có 3 hướng chính để đảm bảo sự đa dạng phong cách được sinh ra như sau:

Đầu tiên, tạo mẫu từ một style code ngẫu nhiên  $s_z$ , nhưng không có gì đảm bảo sự đa dạng của phong cách – Vì Decoder  $F$  có thể gian lận bằng cách bỏ qua style code và chỉ tạo ra một phong cách cho một khuôn mặt đầu vào  $\rightarrow$  không đảm bảo việc có thể sinh ra nhiều phong cách cho một input.

Thứ hai, bộ sinh mẫu có thể được thiết kế để style code  $s_z$  có thể được truy hồi ngược lại từ Decoder  $F$ , nhưng cũng không có gì đảm bảo sự đa dạng – Decoder  $F$  có thể gian lận bằng cách giấu style code  $s_z$  vào một vài điểm ảnh của ảnh được tạo ra hay chỉ thay đổi tổng quan màu sắc của ảnh theo  $s_z$ .

Thứ ba, có thể đặt một hình phạt cụ thể khiến cho Decoder  $F$  sinh những ảnh với style code khác nhau sẽ khác nhau, nhưng cũng không đảm bảo là sẽ đạt được sự đa dạng, Decoder  $F$  có thể gian lận bằng cách thay đổi màu sắc tổng quan của ảnh để khiến cho chúng khác biệt với nhau.

Các chiến lược ở trên đều có yếu điểm và không đạt được sự hiệu quả mong muốn. Định nghĩa về nội dung và phong cách của GANs N' Rose ban đầu chính là một biện pháp hiệu quả để giải quyết bài toán này. Gọi  $x_i \in \mathcal{X}$  tượng trưng cho một ảnh bất kỳ thuộc miền ảnh người thật,  $T(\cdot)$  là hàm sẽ áp một phép tăng cường dữ liệu ngẫu nhiên từ tập các phép tăng cường đã được chỉ định trước vào ảnh đó,  $P(C)$  là phân phối của các content code,  $P(Y)$  là phân phối của miền ảnh hoạt họa thật. Vậy thì  $c(x_i) \sim P(C)$ . Bởi vì như trên đã định nghĩa rằng phong cách sẽ không thay đổi dưới phép tăng cường dữ liệu và nội dung thì có, nếu phép tăng cường dữ liệu hợp lý thì  $c(T(x_i)) \sim P(C)$ , tức là áp một phép tăng cường ngẫu nhiên vào ảnh sẽ tạo ra một content code có phân phối tương tự như content code trước đó. Phép tăng cường dữ liệu ảnh cho bộ phân biệt sẽ không hiệu quả nếu giả định trên không thỏa mãn.

Khi hoạt động, Decoder  $F$  được tiếp cận những content code tạo ra bởi quá trình Encode ảnh mặt người thật, và style code là mẫu của một phân phối đã biết nào đó. Xét một batch những mẫu sinh ra bằng cách:

- (a) Chọn một ảnh mặt người thật.
- (b) Áp các phép tăng cường dữ liệu ngẫu nhiên vào ảnh đó.
- (c) Tính style code của những ảnh sinh ra từ quá trình tăng cường để lấy  $c_i$ .
- (d) Lấy mẫu style code  $z_i$  từ phân phối  $P(S)$  của miền phong cách và sử dụng Decoder  $F$  tạo ra một ảnh hoạt họa  $\hat{Y}_i = F(c_i, z_i; \theta)$ .

Bởi vì  $C$  và  $S$  là phân biệt với nhau, nên chúng sẽ độc lập với nhau. Tiếp đến,  $(c_i, z_i) \sim P(C)P(S)$  và vì phép tăng cường  $\theta$  là đúng nên khi sinh một batch  $\hat{Y}_i$  sẽ là các mẫu không thể phân biệt với những mẫu thật từ  $P(Y)$  với cùng kích thước batch.

Giờ đây giả định rằng bộ sinh mẫu đã được huấn luyện với những điều kiện như ở trên, tạo ra một batch  $\hat{Y}_i$  không thể phân biệt với những mẫu thật từ  $P(Y)$  với cùng kích thước batch. Bởi vì  $P(Y | C = c_i) \propto P(Y, C = c_i)$ , nên mô hình  $P(\hat{Y}_i | C = c_i)$  sẽ đúng.

Tức là, với  $T_n(\cdot)$  là hàm của  $n$  phép tăng cường dữ liệu ngẫu nhiên từ tập tăng cường được chỉ định, áp vào ảnh ta sẽ thu được một lô content code  $\{x_1, x_2, \dots, x_n\} = T_n(x)$ , sau đó có được  $(c_i, s_i) = E(x_i)$ . Lấy các style code  $z_i$  từ phân phối  $P(S)$  của miền phong cách hợp với những content code phía trên ta sẽ được một lô các cặp  $\mathcal{B} = \{(c_1, z_1), \dots, (c_n, z_n)\}$ . Cuối cùng sử dụng Decoder  $F$  để tạo ra một batch  $\hat{Y}_i = F(c_i, z_i; \theta)$  là các mẫu không thể phân biệt với những mẫu thật từ  $P(Y)$  với cùng kích thước batch.

### 3.4.3 Các độ mất mát

**Độ mất nhất quán phong cách:** Dưới phép tăng cường dữ liệu  $T(\cdot)$ , style code là không đổi, nên trong một batch  $\mathcal{B}$ , ta kỳ vọng rằng tất cả các style code  $s_i$  là đồng nhất.

$$\mathcal{L}_{\text{scon}} = \text{Var}(s) \quad (3.6)$$

**Độ mất bảo toàn của mã nội dung trong một chu kỳ:** Ta muốn đảm bảo rằng ảnh hoạt họa giữ được content code. Sinh một ảnh hoạt họa  $\hat{y}_i$  từ ảnh người thật  $x_i$  bằng cách sử dụng cặp content code từ  $x_i$  và một style code  $z_i$  ( $c(x_i), z_i$ ), sau đó ánh



xạ ảnh hoạt hoạ vừa tạo trở ngược lại thành ảnh mặt người thật bằng  $(c(\hat{y}_i), s_i(x_i))$  tức là sử dụng mã nội dung được lấy ra từ việc truyền ảnh hoạt hoạ vừa tạo vào Encoder  $\mathbf{E}$  theo hướng  $\mathcal{Y} \rightarrow \mathcal{X}$ . Ta kỳ vọng kết quả của quá trình này sẽ là ảnh gốc ban đầu, bởi vì ta sử dụng chính mã phong cách của nó, và mã nội dung được bảo toàn. Việc xóa trộn các style code trong một batch (của tất cả các ảnh trong cùng một batch  $\mathcal{B}$ , vì các ảnh này nên có cùng một style code) có ích bởi vì điều này sự bảo toàn phong cách. Ta có:

$$\begin{aligned} c'_i, s'_i &= E_{\mathcal{Y} \rightarrow \mathcal{X}}(\hat{y}_i) \\ \hat{x}_i &= F_{\mathcal{Y} \rightarrow \mathcal{X}}(c'_i, s'_i) \end{aligned} \quad (3.7)$$

Với  $i \neq j$  và  $\hat{x}_i$  là ảnh mặt người gốc được tái tạo, độ mất bảo toàn của content code trong một chu kỳ được tính như sau:

$$\mathcal{L}_{\text{cyc}} = \mathbb{E}_x[\|\hat{x}_i - x_i\|_2] \quad (3.8)$$

Ngoài ra còn sử dụng độ mất mát LPIPS[43].

**Độ mất tính đối kháng của mô hình:** Tiêu chuẩn của mô hình (một batch ảnh được sinh  $\hat{Y}_i$  gồm các mẫu không thể phân biệt với những mẫu thật từ  $P(Y)$  với cùng kích thước) áp dụng cho batch chứ không phải là những ảnh bất kỳ. Vì vậy, ta kỳ vọng những thuộc tính của các mẫu trong cùng một lô sẽ giống nhau thay vì chỉ một vài thuộc tính. Trong quá trình chạy, Discriminator luôn phải làm việc với các ảnh phân biệt chứ không phải là các lô. Discriminator với phương pháp đối kháng sẽ khác, bởi vì nó được huấn luyện và sử dụng với một kích thước lô cụ thể. Discriminator đa dạng của GANs N' Roses sử dụng mẹo về mini-batch từng xuất hiện trong PGGAN[33] để tận dụng điểm khác biệt này. Ở lớp áp chót của Bộ phân biệt, ta tính độ lệch chuẩn cho từng đặc trưng và truyền nó và một lớp fully connected. Bộ phân biệt đa dạng của GANs N' Roses cho ra kết quả logit thật/giả và logit của độ lệch chuẩn. Điều này nghĩa là bộ phân biệt có khả năng xác định điểm khác biệt có thể tin cậy được trong một lô biến thể. Nhóm tác giả sau đó áp độ mất mát đối kháng không bão hòa  $\mathcal{L}_{\text{adv}}$  từ chuẩn hóa R1 cho cả 2 nhánh của Bộ phân biệt. Các lược bỏ cho

thấy bộ phân biệt đa dạng này rất quan trọng, vì nó làm tăng hiệu quả của bộ phân biệt: bộ phân biệt sẽ dễ dàng phát hiện ra các sai lệch tinh vi giữa hai nguồn ảnh nếu nó xem toàn bộ các batch.

Vậy tổng độ mất mát là:

$$\mathcal{L} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{scon}}\mathcal{L}_{\text{scon}} + \lambda_{LPIPS}\mathcal{L}_{LPIPS} + \lambda_{\text{cyc}}\mathcal{L}_{\text{cyc}} \quad (3.9)$$

## 3.5 Kiểm soát miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu

### 3.5.1 Sơ bộ

**Cơ chế sinh của GANs:** Bộ sinh mẫu  $G(\cdot)$  trong GANs học cách ánh xạ từ miền không gian tiềm ẩn  $d$  chiều  $\mathcal{Z} \subseteq \mathbb{R}^d$  vào một miền không gian ảnh  $\mathcal{I} \subseteq \mathbb{R}^{H \times W \times C}$ , với  $\mathbf{I} = G(\mathbf{z})$ .  $\mathbf{z} \in \mathcal{Z}$  và  $\mathbf{I} \in \mathcal{I}$  tượng trưng cho mã tiềm ẩn đầu vào và ảnh đầu ra tương ứng. Bộ sinh mẫu  $G(\cdot)$  bao gồm nhiều lớp sẽ ánh xạ miền tiềm ẩn tới miền ảnh cuối cùng theo từng bước. Mỗi bước học cách biến đổi từ không gian này sang không gian khác. Ta có thể định nghĩa bước đầu tiên của quá trình biến đổi này như sau:

$$G_1(\mathbf{z}) \triangleq \mathbf{y} = \mathbf{A}\mathbf{z} + \mathbf{b} \quad (3.10)$$

Với  $\mathbf{y} \in \mathbb{R}^m$  là miền không gian  $m$  chiều của projected code.  $\mathbf{A} \in \mathbb{R}^{m \times d}$  và  $\mathbf{b} \in \mathbb{R}^m$  tượng trưng cho trọng số của generator và bias được sử dụng ở bước biến đổi đầu tiên  $G_1(\cdot)$ .

**Kiểm soát mô hình trong miền tiềm ẩn GAN:** Miền không gian tiềm ẩn của GANs gần đây đã được chứng minh rằng [36], [44] có thể sử dụng để rút ra những thông tin mang tính ngữ nghĩa phong phú. Những thông tin này có thể ứng dụng để chỉnh sửa hình ảnh bằng cách sử dụng một vector số học hợp lý tượng trưng cho thông tin đó. Hơn thế nữa, các công trình trước đây đã đề xuất [36], [44], [45] sử dụng một hướng nhất định  $\mathbf{n} \in \mathbb{R}^d$  trong miền tiềm ẩn để tượng trưng cho cấu trúc ngữ nghĩa ta muốn hướng đến. Sau khi nhận diện được hướng trong miền không gian tiềm ẩn có ý nghĩa, việc kiểm soát có thể thực hiện được thông qua mô hình sau:

$$\text{edit}(G(\mathbf{z})) = G(\mathbf{z}') = G(\mathbf{z} + \alpha \mathbf{n}) \quad (3.11)$$

thường được sử dụng ở trong các phép tiếp cận đã có từ trước. Ở đây  $\text{edit}(\cdot)$  tượng trưng cho phép chỉnh sửa. Hay nói cách khác, ta có thể chỉnh sửa ảnh tạo thành bằng cách chuyển dịch mã tiềm ẩn  $\mathbf{z}$  tuyến tính theo hướng  $\mathbf{n}$ ,  $\alpha$  tượng trưng cho hệ số kiểm soát mức độ chỉnh sửa.

Để tìm hướng chỉnh sửa  $\mathbf{n}$ , các nhóm nghiên cứu trước đây thường tiếp cận theo 2 phương là có giám sát và không giám sát.

### 3.5.2 Không giám sát

Trong những năm gần đây, nhiều nhóm nghiên cứu đã đề xuất những phương pháp kiểm soát miền không gian tiềm ẩn theo cách không giám sát như Voynov và Babenko [45] tạo một ma trận ứng viên và một bộ phân lớp mà những hướng có ý nghĩa trong ma trận đó có thể được nhận dạng bởi bộ phân lớp, Erik Härkönen và cộng sự [46] thực hiện PCA trên dữ liệu mẫu để tìm các hướng chính trong không gian tiềm ẩn. Nhưng ấn tượng nhất là Yujun Shen và cộng sự, đề xuất một phương pháp khác biệt hoàn toàn với những phương pháp trước đó, khi không cần phải huấn luyện hay lấy mẫu, phương pháp này được nhóm nghiên cứu đặt tên là Sefa - thừa số hóa ngữ nghĩa không giám sát (Semantics Factorization).

SeFa là một phương pháp closed-form để khám phá các hướng tiềm ẩn trong GANs. Bằng cách quan sát cơ chế sinh của GANs, SeFa có thể nhận diện các hướng có ý nghĩa trong không gian tiềm ẩn bằng cách hiệu quả bằng cách dựa vào trọng số của mô hình.

Mục tiêu của SeFa là tìm được các nhân tố mang ý nghĩa trong miền không gian tiềm ẩn của GANs (ví dụ như là hướng của  $\mathbf{n}$ ). Một bộ sinh mẫu GANs có thể được xem như là hàm nhiều bước ánh xạ từ miền tiềm ẩn vào miền ảnh. Xem xét bước ánh xạ đầu tiên, sự kiểm soát mô hình có thể đơn giản hóa như sau:

$$\begin{aligned}\mathbf{y}' &\triangleq G_1(\mathbf{z}') = G_1(\mathbf{z} + \alpha\mathbf{n}) \\ &= \mathbf{A}\mathbf{z} + \mathbf{b} + \alpha\mathbf{A}\mathbf{n} = \mathbf{y} + \alpha\mathbf{A}\mathbf{n}.\end{aligned}\tag{3.12}$$

Như vậy, với một mã tiềm ẩn  $\mathbf{z}$  bất kỳ cùng với một hướng tiềm ẩn cụ thể  $\mathbf{n}$ , quá trình chỉnh sửa luôn có thể thực hiện được bằng cách thêm một lượng  $\alpha\mathbf{A}\mathbf{n}$  vào projected code sau bước đầu tiên. Từ góc nhìn này, các trọng số của  $\mathbf{A}$  sẽ chứa những kiến thức cần thiết cho việc biến đổi ảnh. Vì thế chúng ta tìm những hướng tiềm ẩn quan trọng bằng cách phân rã ma trận trọng số  $\mathbf{A}$ .

Như vậy thừa số hóa ngữ nghĩa không giám sát - SeFa có thể được thực hiện bằng cách giải phương trình tối ưu sau:

$$\mathbf{n}^* = \arg \max_{\{\mathbf{n} \in \mathbb{R}^d : \mathbf{n}^T \mathbf{n} = 1\}} \|\mathbf{A} \mathbf{n}\|_2^2 \quad (3.13)$$

Với  $\|\cdot\|_2$  là  $l_2$  norm. Phương trình này hướng tới việc tìm những hướng có thể tạo ra sự biến đổi lớn sau khi ánh xạ bằng  $\mathbf{A}$ . Nghĩa là, nếu có một hướng  $\mathbf{n}'$  được ánh xạ tới zero-norm vector, ví dụ như  $\mathbf{A} \mathbf{n}' = 0$ , quá trình chỉnh sửa như ở trên sẽ biến  $\mathbf{y}' = \mathbf{y}$ , nghĩa là không thay đổi gì cả.

Khi phải tìm  $k$  hướng quan trọng nhất, ta mở rộng phương trình (3.13) trên thành

$$\mathbf{N}^* = \arg \max_{\{\mathbf{N} \in \mathbb{R}^{d \times k} : \mathbf{n}_i^T \mathbf{n}_i = 1 \forall i=1, \dots, k\}} \sum_{i=1}^k \|\mathbf{A} \mathbf{n}_i\|_2^2 \quad (3.14)$$

Với  $\mathbf{N} = [\mathbf{n}_1, \mathbf{n}_2, \dots, \mathbf{n}_k]$  tương ứng với  $k$  hướng có ý nghĩa nhất. Để giải phương trình trên ta sử dụng phương pháp nhân tử Lagrange  $\{\lambda_i\}_{i=1}^k$  biến phương trình (3.14) thành:

$$\begin{aligned} \mathbf{N}^* &= \arg \max_{\mathbf{N} \in \mathbb{R}^{d \times k}} \sum_{i=1}^k \|\mathbf{A} \mathbf{n}_i\|_2^2 - \sum_{i=1}^k \lambda_i (\mathbf{n}_i^T \mathbf{n}_i - 1) \\ &= \arg \max_{\mathbf{N} \in \mathbb{R}^{d \times k}} \sum_{i=1}^k (\mathbf{n}_i^T \mathbf{A}^T \mathbf{A} \mathbf{n}_i - \lambda_i \mathbf{n}_i^T \mathbf{n}_i + \lambda_i) \end{aligned} \quad (3.15)$$

Bằng cách lấy đạo hàm cho mỗi  $\mathbf{n}_i$ , ta có

$$2\mathbf{A}^T \mathbf{A} \mathbf{n}_i - 2\lambda_i \mathbf{n}_i = 0 \quad (3.16)$$

Tất cả các nghiệm của phương trình trên là vector riêng của ma trận  $\mathbf{A}^T \mathbf{A}$ . Để lấy được giá trị tối đại và khiến cho  $\{\mathbf{n}_i\}_{i=1}^k$  phân biệt lẫn nhau, ta chọn  $\mathbf{N}$  vector riêng của  $\mathbf{A}^T \mathbf{A}$  có  $k$  trị riêng lớn nhất.

### 3.5.3 Có giám sát

Một số phương pháp có giám sát được đề xuất trong những năm gần đây bao gồm những hướng chính như sau:

- Donahue và cộng sự đề xuất [32], [42] việc sử dụng một Encoder. Tức là nếu StyleGAN học cách ánh xạ từ không gian tiềm ẩn  $w \rightarrow$  ảnh, thì ta sẽ huấn luyện thêm một bộ Encoder theo cách có giám sát để chuyển ảnh  $\rightarrow w$ .
- Antonia Creswell và Anil A Bharath [47] đề xuất lan truyền ngược độ lỗi trên từng điểm ảnh hay các chi tiết để tối ưu latent code. Một mạng nơ ron nhân tạo có 3 thành phần: đầu vào, tham số mô hình, đầu ra/độ lỗi, vì thế ta có tiếp cận theo 3 hướng để lan truyền ngược. Đầu tiên ta có thể giữ nguyên đầu vào, thay đổi tham số mô hình để giảm độ lỗi  $\rightarrow$  đây là quá trình huấn luyện mô hình. Cách thứ 2 là giữ nguyên đầu vào, thay đổi đầu ra để thay đổi thông số của các lớp  $\rightarrow$  trực quan hóa và khám phá mạng nơ ron nhân tạo. Cuối cùng là giữ nguyên các tham số và đầu ra của mô hình, sử dụng gradient để tìm một tập các đầu vào cụ thể mà tạo ra đầu ra với độ lỗi thấp nhất.

Trong trường hợp cụ thể của ta, cho một hình ảnh mong muốn tùy ý *target*, bộ sinh mẫu có thể khởi tạo một  $z$  ngẫu nhiên, cho nó qua mạng sinh mẫu để tạo ra một ảnh, so sánh ảnh này với *target* theo mức độ từng điểm ảnh để tìm ra độ lỗi, sau đó lan truyền ngược để điều chỉnh đầu vào làm cho nó giống với hình ảnh mong muốn *target* hơn.

- Sử dụng một mạng phân lớp để gán nhãn các ảnh được tạo ra bởi mô hình, sau đó dùng những nhãn này để học những vùng nào miền không gian tiềm ẩn ứng với những ngữ nghĩa mà ta mong muốn [48].

## Chương 4

# Phương pháp đề xuất

### 4.1 Giới thiệu phương pháp thao túng miền không gian tiềm ẩn được đề xuất thử nghiệm

Từ cơ sở lý thuyết phía trên, ta tiến hành thử nghiệm việc chỉnh sửa ảnh bằng cách thao túng miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu bằng phương pháp có giám sát: sử dụng một mạng phân lớp pre-train để gán nhãn cho các ảnh được sinh ra bởi GANs N' Roses, sau đó nhóm sẽ dùng kết quả của quá trình này nhằm tìm các vector tượng trưng cho các hướng ngữ nghĩa mong muốn trong miền không gian tiềm ẩn, cuối cùng sử dụng những vector này để thao túng nhằm chỉnh sửa ảnh theo ý muốn.

Nguyên nhân phương pháp này được sử dụng vì trong quá trình khám phá mô hình GANs N' Roses để thực hiện mục đích làm phim hoạt hình, nhóm nhận thấy rằng GANs N' Roses khi chuyển đổi miền không gian ảnh, một số chi tiết như màu mắt, khẩu hình miệng (mở miệng, khép miệng, cười,...), độ dài của tóc của ảnh chụp người thật khi chuyển qua miền không gian hoạt họa không được bảo toàn.



**Hình 4.1 Ví dụ của việc không bảo toàn được một số chi tiết khi chuyển miền ảnh**

Phương pháp đề xuất sẽ được so sánh đánh giá với phương pháp chỉnh sửa được chính các tác giả của GANs N' Roses giới thiệu SeFa (không giám sát) nhằm so sánh đánh giá mức độ hiệu quả của quá trình chỉnh sửa. Tại sao nhóm không dùng chính SeFa để chỉnh sửa mà lại phải sử dụng một phương pháp có giám sát? Nguyên nhân chính là do bản chất của SeFa là một phương pháp không giám sát, chính vì thế nó không cho phép người dùng có được sự toàn quyền trong chỉnh sửa các chi tiết, rất khó để chỉnh sửa một số chi tiết nhỏ như (miệng, tóc, lông mày,...) mà không ảnh hưởng tới các phần khác của ảnh.

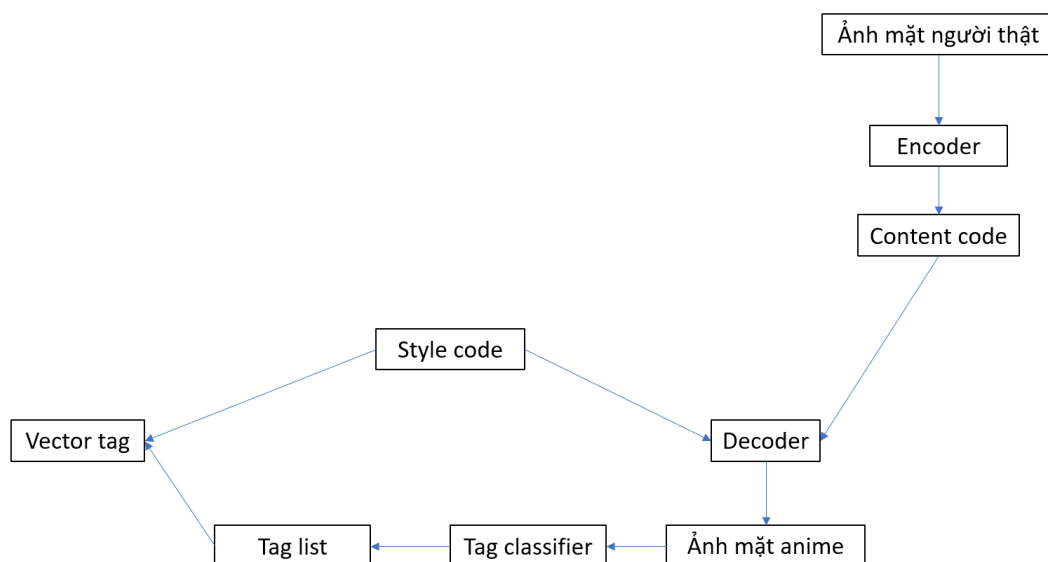


**Hình 4.2 Minh họa sự chỉnh sửa của SeFa**



## 4.2 Quy trình thao túng miền không gian tiềm ẩn có giám sát

Quy trình thực hiện có thể tóm tắt bằng sơ đồ dưới đây:



**Hình 4.3 Sơ đồ quá trình tạo các vector nhãn**

### 4.2.1 Sinh ảnh hoạt họa ngẫu nhiên:

Sử dụng Decoder – Generator để sinh ảnh hoạt họa ngẫu nhiên và lưu lại style code cho mỗi lần sinh ảnh. Đây có thể coi như là một bước lấy mẫu cho việc khám phá miền không gian tiềm ẩn.

### 4.2.2 Gán nhãn:

Từ những ảnh sinh ra ở mục trước, ta sử dụng một mạng trí tuệ nhân tạo có sẵn pre-train để gán nhãn cho các ảnh này. Mỗi ảnh sinh ra sẽ có một bộ nhãn tương ứng với nó hay nói cách khác mỗi style code sẽ có một bộ nhãn tương ứng.

### 4.2.3 Các phương pháp tìm vector ngữ nghĩa được đề xuất

#### 4.2.3.1 Hồi quy:

Đối với hồi quy, ta xây dựng vector dòng mục tiêu bằng cách lấy từng nhãn của các bộ nhãn rồi đếm xem thử nó xuất hiện trong những ảnh nào, nếu có thì ở vị trí đó nó được gán giá trị bằng 1, các vị trí còn lại được gán bằng 0. Tức là vector dòng sẽ

có kích thước là bằng số ảnh được sinh ra đặt là  $\mathbf{M}$  với giá trị của các phần tử là 0 hoặc 1. Nghĩa là sẽ hồi quy như sau:

$$X_{\mathbf{M},512}A_{512,1} = N_{\mathbf{M},1} \quad (4.1)$$

Ma trận  $X$  là ma trận chứa các vector phong cách của quá trình sinh ảnh.

$N$  là vector mục tiêu

Vector  $A$  chính là vector tượng trưng cho nhãn được hồi quy.

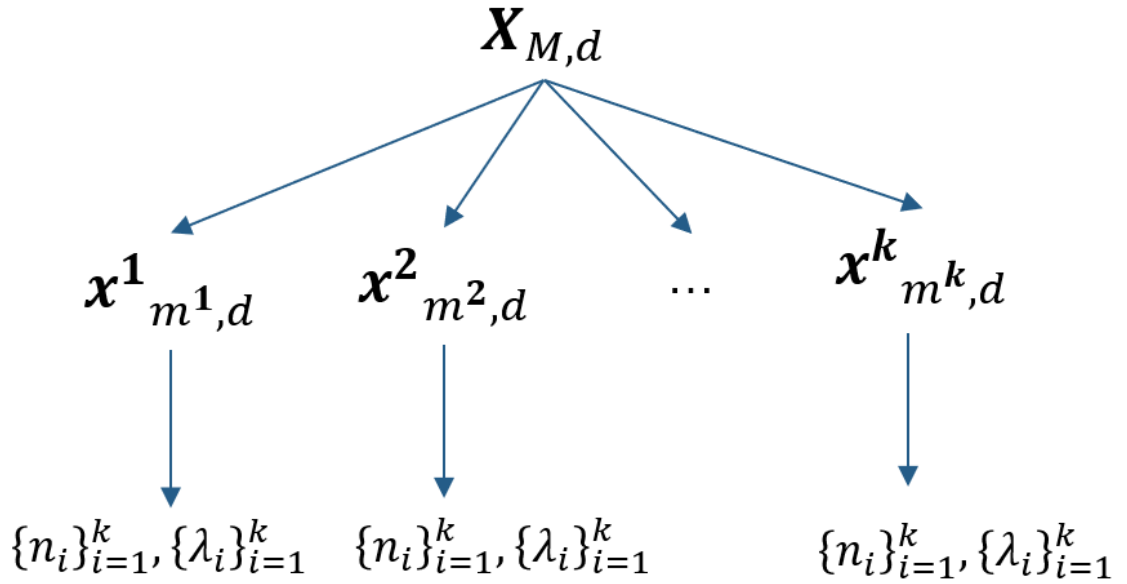
Tiến hành hồi quy lần lượt như vậy cho tất cả các nhãn ta sẽ hoàn thành quá trình hồi quy.

#### 4.2.3.2 Phân rã ma trận để tìm vector riêng

Ta sử dụng ma trận chứa các vector phong cách của quá trình sinh ảnh để làm đầu vào cho quá trình này. Xét từng nhãn trong bộ nhãn sau khi lọc, ta tiến hành lọc các vector phong cách tương ứng với ảnh này để xây dựng một ma trận chứa các vector phong cách đó. Nếu như giả định về tính liên tục của miền không gian tiềm ẩn là chính xác thì ma trận này sẽ là miền không gian con thuộc miền không gian mẹ quy định chi tiết ảnh ứng với nhãn đang được xét.

Vector riêng trong tiếng anh gọi là eigen vector vốn được mượn từ tiếng Đức *eigen* nghĩa là “đặc trưng”. Như tên gọi của nó, ta tìm các vector đặc trưng này để làm đặc trưng cho miền không gian tiềm ẩn ứng với nhãn đang được xét.

Quy trình thực hiện có thể được mô tả như hình dưới đây:



**Hình 4.4 Quy trình phân rã ma trận tìm vector riêng**

Đối với quá trình phân rã ma trận:

- Ta làm lần lượt cho mỗi nhãn, xét từng bộ tag, nếu trong bộ tag đó có nhãn ta đang xét thì thêm vector phong cách của bộ nhãn đó vào ma trận của nhãn đang xét.
- Tìm vector riêng của ma trận này.
- Làm lần lượt như vậy cho tất cả các nhãn sẽ hoàn thành quá trình phân rã ma trận.

Kết quả của quá trình này sẽ là một ma trận chứa các vector riêng đặc trưng cho từng miền không gian tiềm ẩn ứng với từng nhãn được xét.

#### 4.2.3.3 So sánh giữa 2 phương pháp về mặt lý thuyết

Có thể mô tả hồi quy như việc tìm một đường thẳng đi qua không gian sao cho khoảng cách từ các điểm đang được xét đến đường thẳng đó là nhỏ nhất. Chính vì điều này mà vector thu được từ quá trình hồi quy có thể không chính xác trong việc tượng trưng cho nhãn đó nhất có thể, dẫn đến hiệu năng chỉnh sửa các chi tiết ảnh có thể không được như ý muốn.

Trong khi đó, việc phân rã ma trận để tìm các vector riêng ứng với mỗi miền không gian quy định chi tiết ảnh trong toàn miền không gian tiềm ẩn sẽ cho ra các vector đặc trưng nhất của miền không gian đó, chính vì thế ta kỳ vọng rằng khi chỉnh sửa bằng các vector này sẽ ảnh hưởng lên ảnh hiệu quả hơn.

#### 4.2.3.4 Các kỹ thuật phụ họa bổ sung

Một điểm đặc biệt cần lưu ý là một ảnh có thể có nhiều nhãn được gán - các nhãn đi kèm với nhau, chính vì thế mà vector nhãn thu được từ quá trình hồi quy/ phân rã ma trận có thể không chỉ quy định riêng cho một phần ảnh tương ứng với nhãn mà có thể cho cả các phần khác của ảnh.

Điều này làm cho việc chỉnh sửa bằng các vector nhãn tuy có thể thay đổi phần mong muốn, nhưng cũng có thể kéo theo những phần khác mà người chỉnh sửa không mong muốn.

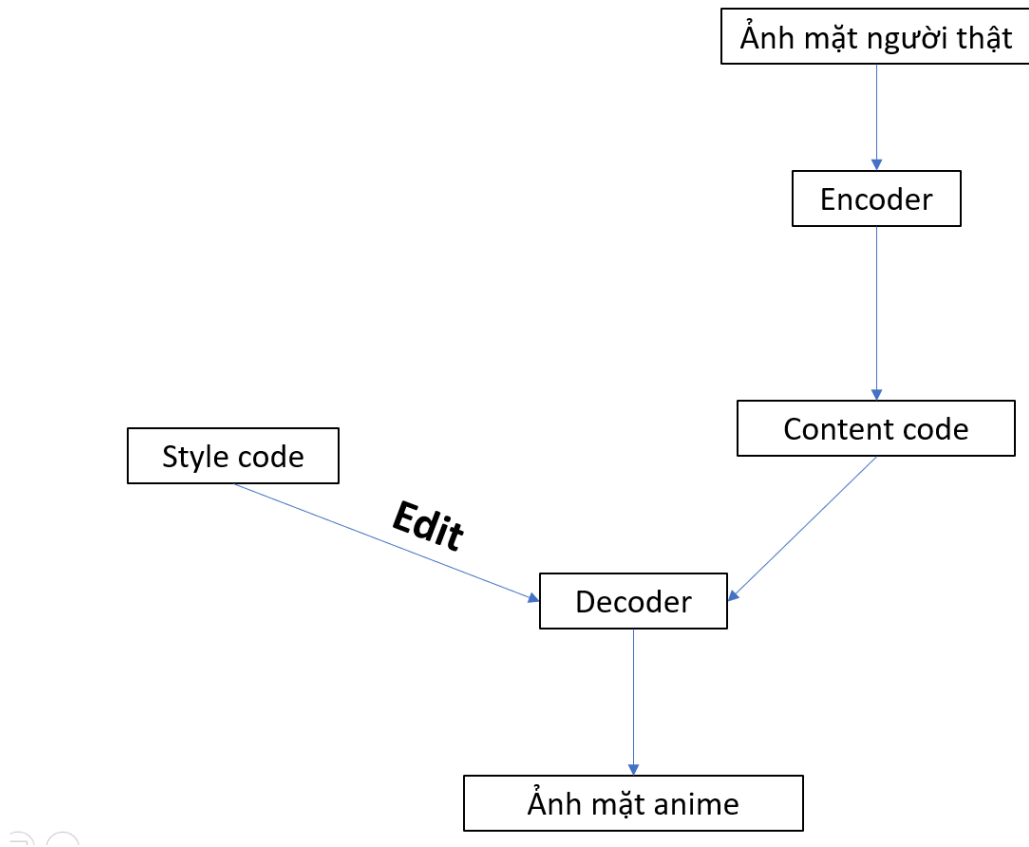
Nhằm khắc phục điều này, nhóm đề xuất việc sử dụng các cặp nhãn đối nghịch trong quá trình chỉnh sửa. Ví dụ cặp nhãn tóc dài – tóc ngắn, miệng mở - miệng khép, mắt mở - mắt đóng,... Hoặc chúng ta có thể ứng dụng để chỉnh màu tóc, màu mắt: màu tóc đen – tóc đỏ, mắt xanh – mắt vàng,...

Khi chỉnh sửa, chẳng hạn nếu ta muốn làm cho tóc nhân vật dài hơn thì sẽ cộng vector tóc dài và trừ vector tóc ngắn vào vector phong cách ban đầu hay muốn chuyển nhân vật đang có tóc đen thành tóc đỏ thì ta sẽ trừ vector tóc đen và cộng vector tóc đỏ vào vector phong cách để sinh ảnh.

Bằng cách làm như vậy, ta có thể đạt được hiệu quả các quá trình chỉnh sửa tốt hơn mà không làm biến dạng ảnh quá nhiều. Nguyên nhân bởi vì ta ổn định được các miền không gian khác mà không cần chỉnh sửa bằng cách sử dụng các cặp nhãn này, chính vì thế mà ảnh sau khi được chỉnh sửa có thể mang chi tiết ta mong muốn và những phần khác được tương đối bảo toàn.

### 4.3 Tổng quan quy trình của chương trình

Quy trình thực hiện của chương trình được nhóm mô tả theo hình ảnh dưới đây:



**Hình 4.5 Tổng quan quy trình sinh ảnh của chương trình**

Như vậy quy trình hoạt động được dựa trên quy trình của GANs N' Roses với sự bổ sung bước chỉnh sửa vector phong cách trước khi được cho vào Decoder để hợp với content code tạo thành ảnh mặt anime.

## Chương 5

# Thực nghiệm và kết quả

### 5.1 Dữ liệu

Nhóm thực nghiệm trên bộ dữ liệu selfie2anime – giống như bộ dữ liệu mà các tác giả GANs N’ Roses đã làm để có một kết quả so sánh công bằng.

Bộ dữ liệu selfie2anime được thu thập bởi Kim và cộng sự để sử dụng cho mô hình U-GAT-IT [49] của họ. Tập dữ liệu selfie được chia làm 2 tập là tập train và tập test với kích thước lần lượt là 3400 ảnh và 100 ảnh. Các ảnh này đều là ảnh selfie của nữ giới và có kích thước 256 x 256. Tập dữ liệu anime được các tác giả thu thập từ Anime-Planet [50] và được cắt phần mặt nhân vật ra sử dụng một bộ dò mặt nhân vật anime [51], sau đó các ảnh này được chọn thủ công để tạo nên 2 tập là tập train và tập test với kích thước lần lượt là 3400 ảnh và 100 ảnh như tập selfie. Các ảnh anime này sau đó được thay đổi kích thước về 256 x 256 bằng cách sử dụng phương pháp super resolution [52].

### 5.2 Độ đo

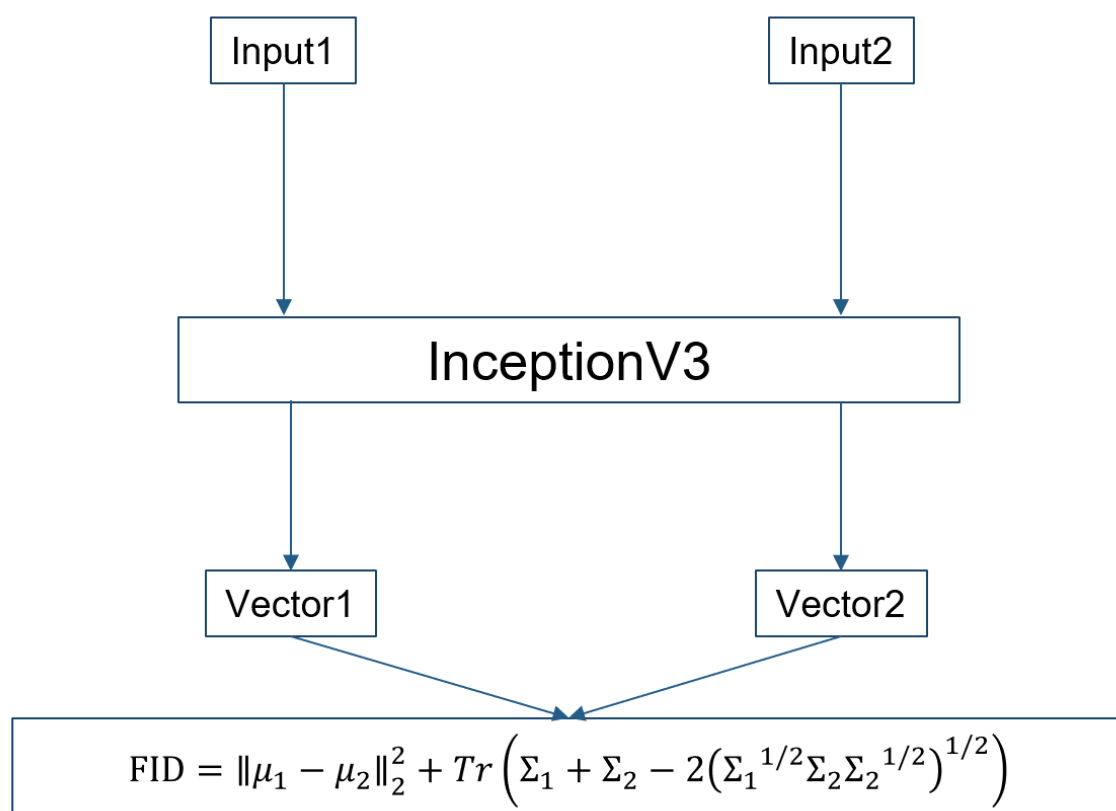
Nhóm sử dụng các độ đo tương tự như độ đo gốc được sử dụng trong GANs N’ Roses là FID, DFID, LPIPS để đánh giá được tính bảo toàn các kết quả sẵn có của mô hình gốc khi thêm việc thao túng miền không gian tiềm ẩn cộng với độ đo Re-scoring để đánh giá kết quả của việc chỉnh sửa ảnh.

#### 5.2.1 FID

FID (Fréchet Inception Distance) là độ đo được phát triển để ước lượng hiệu năng thực thi của các mạng GAN và dùng để ước lượng chất lượng tập ảnh được mô hình sinh ra so với tập ảnh gốc. Mục tiêu phát triển hệ số FID là ước lượng, đánh giá ảnh được sinh dựa trên số liệu thống kê từ tập ảnh được sinh so với số liệu thống kê từ

tập hình ảnh thực. Chỉ số FID càng thấp thì chất lượng ảnh được sinh càng cao tức là ảnh sinh ra càng gần với ảnh thực – càng giống thực.

Cách tính FID như sau:



**Hình 5.1 Quy trình tính FID**

Mỗi ảnh trong tập ảnh gốc khi cho qua InceptionV3 sẽ được 1 vector có kích thước 2048, từ đó ta sẽ tìm được 1 phân phối chuẩn đa chiều fit nhất tất cả vector này với  $\text{mean} = \mu_X$  và  $\text{std} = \Sigma_X$ .

Tương tự mỗi ảnh trong tập ảnh tái tạo khi cho qua InceptionV3 sẽ được 1 vector  $2048 * 1$ , từ đó ta sẽ tìm được 1 phân phối chuẩn đa chiều fit nhất tất cả vector này với  $\text{mean} = \mu_X$  và  $\text{std} = \Sigma_X$ .

Đối với phân phối chuẩn đơn biến, giả sử 2 đường cong giữa người và chó là 2 phân phối chuẩn. Ta có được công thức tính khoảng cách giữa 2 phân phối chuẩn này

là bình phương khoảng cách giữa trung bình cộng với bình phương khoảng cách giữa độ lệch chuẩn của 2 phân phối.

$$d(X, Y) = (\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2 \quad (5.1)$$

Trong đó:

- $\mu_X$  và  $\mu_Y$  là giá trị trung bình của 2 hàm phân phối cho ta cảm giác về trung tâm.
- $\sigma_X$  và  $\sigma_Y$  là độ lệch chuẩn của 2 hàm phân phối cho ta cảm giác về mức độ lan truyền.

Tương tự trường hợp đơn biến, trường hợp đa biến ứng dụng cho vector, để các ảnh tái tạo sinh ra giống với các ảnh trong dataset thì ta mong muốn 2 phân phối chuẩn đa biến này giống nhau, hay mean và variance giống nhau:

$$\text{FID} = \|\mu - \mu_w\|_2^2 + \text{Tr}(\Sigma + \Sigma_w - 2(\Sigma^{1/2}\Sigma_w\Sigma^{1/2})^{1/2}) \quad (5.2)$$

Trong đó:

•  $\mu_w$  và  $\mu$  là vector trung bình từng thành phần của tập ảnh gốc và tái tạo, tức là các vector 2048 chiều mà mỗi thành phần là trung bình của các thành phần trong vector đặc trưng tương ứng của từng ảnh.

•  $\Sigma_w$  và  $\Sigma$  là các ma trận hiệp phương sai của các vector đặc trưng của ảnh gốc và ảnh tái tạo.

•  $\text{Tr}$  là đề cập đến ma trận, lấy tổng các phần tử dọc theo đường chéo chính của ma trận.

### 5.2.2 DFID

DFID (Diversity Fréchet Inception Distance) là một độ đo được các tác giả của GANs N' Roses tự định nghĩa để đo độ đa dạng của multi-modal framework. Trong GANs N' Roses khi áp  $M$  phép tăng cường dữ liệu ngẫu nhiên vào ảnh selfie được thử nghiệm sẽ được  $M$  mẫu từ phân phối content. Như vậy khi sinh ảnh anime ta sẽ có được một tập có kích thước tương ứng. Điều này đúng với bất kỳ ảnh nào được lựa chọn, như vậy DFID được tính như sau:



$$DFID = \frac{1}{N} \sum_{i=1}^N FID(\{G(T_M(x_i), z) \mid z \sim \mathcal{N}(0, I)\}, Y) \quad (5.3)$$

### 5.2.3 LPIPS

LPIPS (Learned Perceptual Image Patch Similarity) là một độ đo thể hiện mức độ tương đồng giữa 2 ảnh sử dụng mạng học sâu. Khác với FID có thể đo giữa 2 tập dữ liệu, LPIPS chỉ có thể đo độ tương đồng giữa 2 ảnh, độ đo càng thấp thì giữa 2 ảnh càng có nhiều sự tương đồng.

LPIPS được tính bằng cách sử dụng một mạng VGG16 [53], LPIPS được tính thông qua việc sử dụng các activation của các layer trong mạng.

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \|w_l \odot (\hat{y}_{hw}^l - \hat{y}_{0hw}^l)\|_2^2 \quad (5.4)$$

Trong đó:

- $\hat{y}_{hw}^l, \hat{y}_{0hw}^l \in \mathbb{R}^{H_l \times W_l \times C_l}$  tương ứng với activation của  $x, x_0$  của một channel trong layer  $l$ .
- $w_l \in \mathbb{R}^{C_l}$  là vector được sử dụng để scale  $(\hat{y}_{hw}^l - \hat{y}_{0hw}^l)$ .
- $H_l W_l$  tượng trưng cho tất cả các channel trong layer  $l$ .

### 5.2.4 Attribute Correlation

Attribute Correlation là một độ đo được sử dụng để đánh giá kết quả của việc chỉnh sửa ảnh[36] cụ thể ở đây là chỉnh sửa các chi tiết trên khuôn mặt. Độ đo này thể hiện rằng sự tương quan giữa các vector nhãn được dùng để chỉnh sửa. Công thức của độ đo:

$$Attribute\ Correlation = \frac{n_1 n_2}{|n_1| |n_2|} \quad (5.5)$$

Trong đó  $n_1, n_2$  là 2 vector được đo độ tương quan

Ta mong muốn rằng giữa 2 vector nhãn khác nhau sẽ có độ tương quan = 0, tức là khi chỉnh sửa bằng vector nhãn này, thì chi tiết mà vector nhãn kia quy định sẽ không thay đổi.

## 5.3 Quy trình thực nghiệm

### 5.3.1 Quá trình tạo các vector nhãn:

Như đã mô tả ở hình 4.3, ta lần lượt làm theo các bước như sau:

#### 5.3.1.1 Sinh ảnh hoạt họa ngẫu nhiên:

Sử dụng các ảnh mặt người thật, ta tiến hành sinh ảnh hoạt họa bằng cách lấy nội dung từ ảnh người thật hợp với ba phong cách được chọn ngẫu nhiên để cho ra ba ảnh hoạt họa. Kết thúc quá trình này cho ta được 10000 ảnh hoạt họa với phong cách ngẫu nhiên. Nguyên nhân vì ta muốn đa dạng các loại ảnh hoạt họa có thể sinh ra đồng thời việc có nhiều ảnh sẽ mang ý nghĩa tích cực trong việc hồi quy/phân rã ma trận. Khi sinh ảnh, ta lưu lại vector phong cách để hồi quy/phân rã ma trận ở bước sau.

#### 5.3.1.2 Gán nhãn:

Do bộ dataset của mô hình không có nhãn cho trước, vì thế ta phải sử dụng một mạng phân lớp pre-train (để gán các nhãn này). Deepdanbooru [9] là một mạng Resnet được chỉnh sửa để phân lớp ảnh hoạt họa, được huấn luyện với hơn 2 triệu ảnh hoạt họa thu thập ở trên trang web Danbooru [54]. Các ảnh khi cho vào mạng Deepdanbooru sẽ được scale về kích thước 299x299, sau đó thông qua hàng loạt các lớp tích chập và lớp nơ ron nhân tạo sẽ trả về các nhãn ứng với ảnh đó. Hay nói cách khác là các nhãn tương ứng với vector phong cách của ảnh đó. Sau quá trình này ta đã được 10000 bộ nhãn ứng với 10000 ảnh được sinh ra.

Kiến trúc mạng Deepdanbooru:

Output size	Layer
$256 \times 256$	$7 \times 7, 64, \text{stride } 2$
$128 \times 128$	$3 \times 3 \text{ max pool, stride } 2$
	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 2$
$64 \times 64$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 7$
$32 \times 32$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 2 \text{ } 1024 \end{bmatrix} \times 19$
$16 \times 16$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 2 \text{ } 1024 \end{bmatrix} \times 19$
$8 \times 8$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 2$
$4 \times 4$	$\begin{bmatrix} 1 \times 1, 1024 \\ 3 \times 3, 1024 \\ 1 \times 1, 4096 \end{bmatrix} \times 2$

**Bảng 5.1 Kiến trúc mạng DeepDanbooru**

#### 5.3.1.3 Tìm vector ngữ nghĩa:

Trước khi thực hiện, ta tiến hành lọc lại dữ liệu. Đầu tiên, chỉ những nhãn nào xuất hiện dưới 9800 lần, nguyên nhân của việc này vì nếu nhãn xuất hiện quá nhiều thì đó sẽ là những nhãn mà ảnh nào cũng có nên không có ý nghĩa về mặt chỉnh sửa.

Đối với hồi quy ta sử dụng các kỹ thuật:

1. Hồi quy tuyến tính

2. Hồi quy Lasso
3. Logistic
4. SVM

Các kỹ thuật được cài đặt sử dụng thư viện sklearn [55].

Đối với phân rã ma trận ta sử dụng kỹ thuật: singular value decomposition của thư viện numpy [56].

### 5.3.2 Chỉnh sửa ảnh sử dụng các vector nhãn:

Sau khi đã thu được các vector nhãn, ta có thể chỉnh sửa ảnh theo công thức (3.11), tức là thêm  $\alpha \mathbf{n}$  vào vector phong cách gốc ban đầu.

### 5.3.3 Đo các độ đo:

Như đã đề cập ở phần 4, bởi GANs N' Roses khi chuyển đổi miền không gian ảnh, một số chi tiết như màu mắt, khẩu hình miệng (mở miệng, khép miệng, cười,...), độ dài của tóc của ảnh chụp người thật khi chuyển qua miền không gian hoạt họa không được bảo toàn, chính vì thế khi đo các độ đo nhóm sẽ tập trung vào các tag hướng đến những chi tiết này bao gồm các cặp tag như sau:

- Open mouth – Closed mouth
- Short hair – Long hair
- Closed Eyes

Sau khi đã sinh ảnh, ta sử dụng các cặp tag đối nghịch này bằng cách cộng chúng vào vector phong cách ban đầu với cặp hệ số trái dấu:

#### 5.3.3.1 Đo FID:

Để đo FID, sử dụng 100 ảnh selfie trong thư mục testA của tập dữ liệu selfie2anime để sinh ảnh anime từ các ảnh này sau đó chỉnh sửa chúng theo các vector nhãn phía trên, sau đó ta tiến hành so tập ảnh này với tập testB bao gồm 100 ảnh anime để đánh giá mức độ giống thật của các ảnh được sinh ra.

#### 5.3.3.2 Đo DFID:

Với mỗi ảnh đầu vào từ thư mục testA, ta áp 1000 phép tăng cường dữ liệu vào ảnh đó và sinh ảnh anime, sau đó chỉnh sửa các ảnh này theo các vector nhãn đã đề cập. So các ảnh này với tập testB để tìm FID. Làm lần như vậy cho tất cả các ảnh trong thư mục testA và lấy trung bình kết quả.

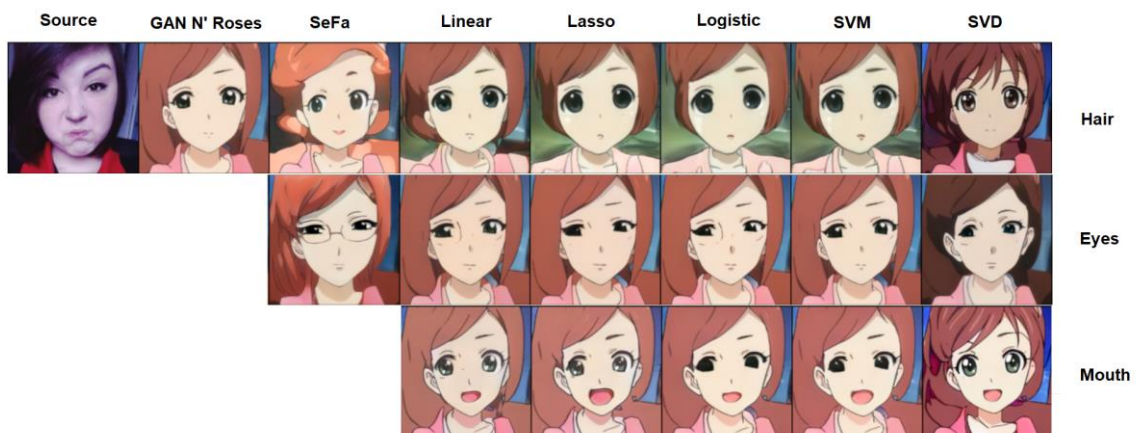
#### 5.3.3.3 Đo LPIPS:

Với mỗi ảnh đầu vào từ thư mục testA, ta sinh ngẫu nhiên 10 ảnh anime từ ảnh này và chỉnh sửa chúng, sau đó đo LPIPS giữa các ảnh đôi một. Làm như vậy cho tất cả các ảnh trong thư mục testA và lấy trung bình kết quả.

#### 5.3.3.4 Đo Re-scoring:

Sử dụng mạng ResNet50 [10] được train trên tập trainB của bộ dữ liệu selfie2anime với 3 nhãn bao gồm: miệng mở, tóc ngắn, mắt nhắm. Với mỗi ảnh đầu vào của thư mục testA, ta sinh ngẫu nhiên ảnh anime sau đó chỉnh sửa nó, dùng mạng Resnet để gán nhãn ảnh trước khi chỉnh sửa và sau khi chỉnh sửa để đánh giá mức độ ảnh hưởng của quá trình chỉnh sửa.

## 5.4 So sánh định tính



**Hình 5.2. Minh họa chỉnh sửa bằng các phương pháp được đề xuất**

Từ hình 5.2 ta rút ra được một số nhận xét như sau:

- SeFa ít làm thay đổi những chi tiết của ảnh nhất, rất khó để chỉnh sửa một chi tiết theo ý mình.
- Sử dụng các vector từ quá trình hồi quy tuyến tính thể hiện tốt khi chỉnh sửa chi tiết mắt và miệng, tuy nhiên đối với chỉnh tóc thì không đạt được mục tiêu đề ra.
- Sử dụng các vector từ quá trình hồi quy Lasso cho kết quả tạm chấp nhận được khi chỉnh sửa tóc và miệng, tuy nhiên đối với cỡ mắt thì không hiệu quả.
- Sử dụng các vector từ quá trình phân rã ma trận cho ta kết quả tốt nhất khi chỉnh sửa cả tóc, miệng và mắt, tuy nhiên đối với chi tiết mắt thì màu ảnh bị kéo theo thay đổi.

Ta cũng nên lưu ý rằng việc chỉnh sửa màu dễ hơn so với chỉnh sửa các chi tiết rất nhiều vì thế ta luôn đặt mục tiêu hiệu quả của việc chỉnh sửa làm thay đổi cấu trúc các chi tiết làm trọng tâm chứ không chú trọng vào màu sắc.



**Hình 5.3 Dùng vector nhãn của các phương pháp để chỉnh màu tóc sang màu xanh**

Từ kết quả trên ta thấy rằng hồi quy Lasso tỏ ra hiệu quả hơn so với phần còn lại ở phần chỉnh sửa màu, tuy nhiên nhìn chung việc chỉnh màu đạt hiệu quả cao cho tất cả các phương pháp.

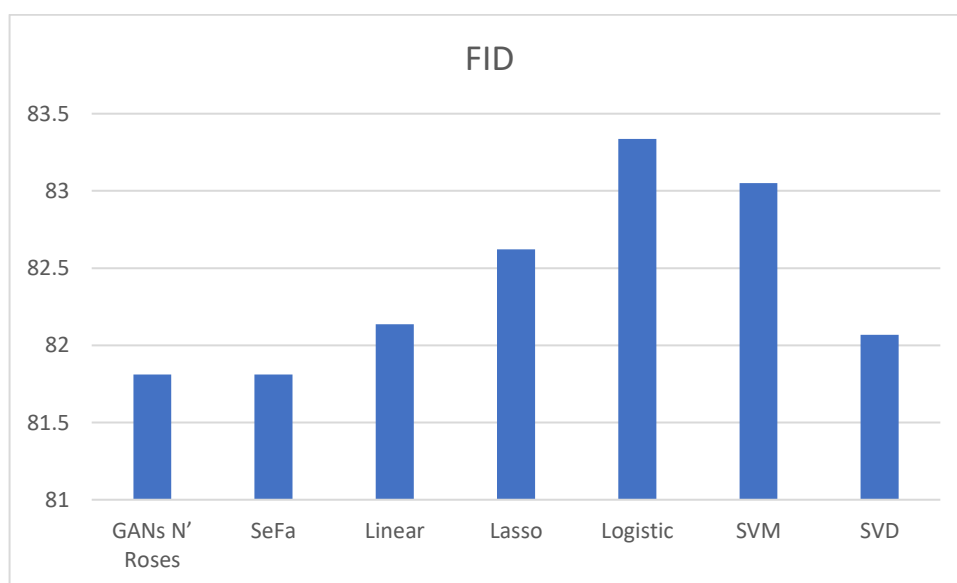


**Hình 5.4 Minh họa ảnh hưởng của việc sử dụng cặp nhãn đối nghịch**

Từ hình trên ta có thể thấy rằng khi không sử dụng cặp tag đối nghịch thì các phần khác không phải mục tiêu chỉnh sửa cũng bị kéo theo thay đổi tuy ở một mức chấp nhận được nhưng khiến cho kết quả không được như mong đợi. Chính việc áp dụng cặp nhãn đối nghịch khắc phục nhược điểm trên: thay đổi được mục tiêu cần chỉnh sửa mà không làm biến dạng những phần khác của ảnh, qua đó tăng tính hiệu quả của quá trình chỉnh sửa. Ngoài ra việc sử dụng cặp tag đối nghịch này còn tăng hiệu quả chỉnh sửa cho chi tiết ta hướng đến.

Từ kết quả trên có thể nói rằng chính việc sử dụng cặp nhãn đối xứng là phần bổ sung cần thiết cho phương pháp có giám sát được áp dụng.

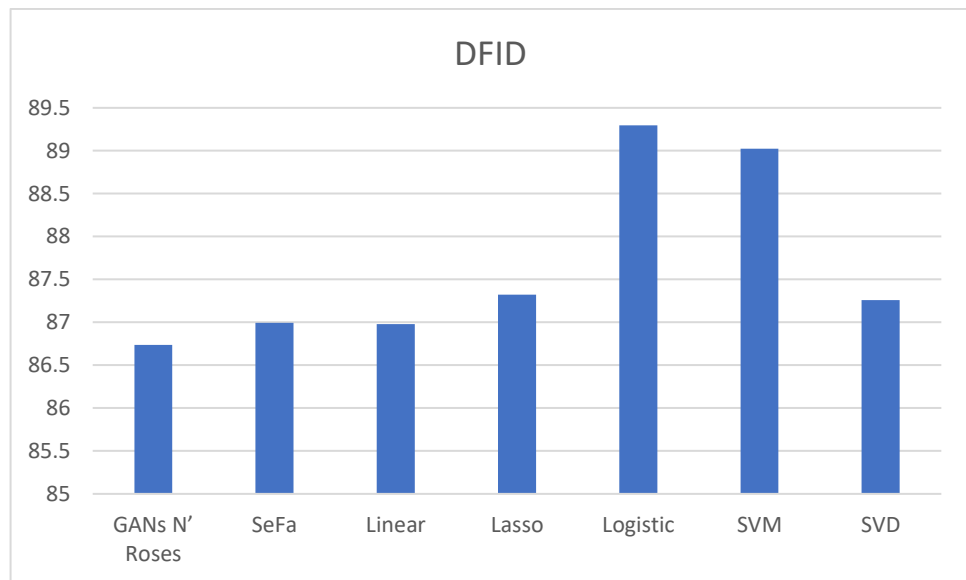
## 5.5 So sánh định lượng:



**Hình 5.5 Kết quả độ đo FID**

Ở trên hình thì SeFa cho kết quả gần với kết quả của mô hình gốc nhất, điều này đã được nằm trong dự đoán vì các vector ngữ nghĩa của SeFa là các vector riêng được lấy từ quá trình phân rã ma trận trọng số của bộ Generator, chính vì thế chúng là những vector có ý nhất với bộ generator, do đó giúp ảnh tạo ra giống thật nhất.

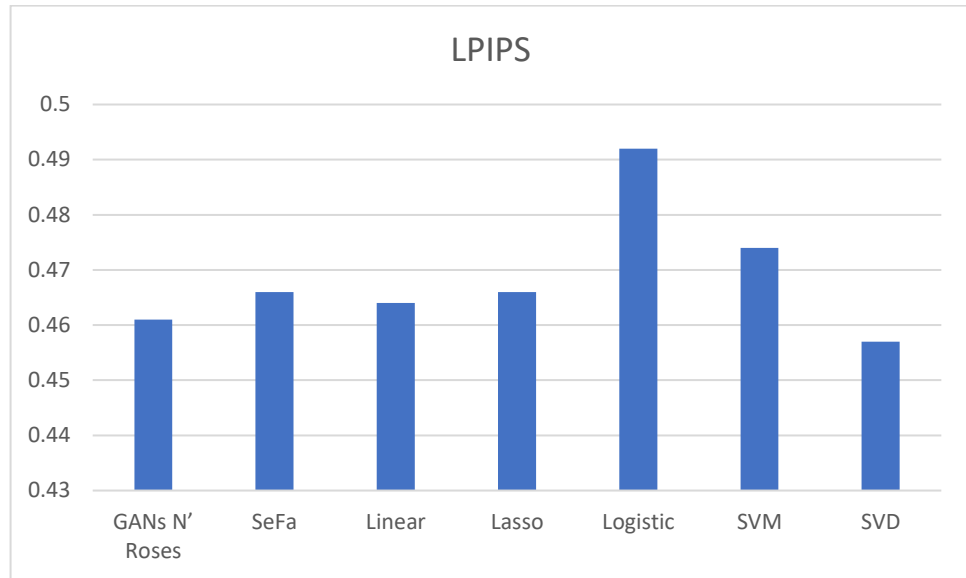
Linear thể hiện tốt nhất trong các phương pháp còn lại.



**Hình 5.6 Kết quả độ đo DFID**

Ở độ đo này thì SeFa và Linear cho kết quả tương đồng nhau.





**Hình 5.7 Kết quả độ đo LPIPS**

Logistic thể hiện tốt nhất ở độ đo này.

Có thể thấy rằng từ các độ đo trên thì việc thêm bước chỉnh sửa vẫn giữ được điểm mạnh của mô hình GANs N' Roses, trong đó thì Linear cho kết quả khả quan nhất trên các độ đo.

### Re scoring:

Không dùng tag đối nghịch:

open_mouth	1		
short_hair	0.057748	1	
half-closed_eyes	0.006926	0.016922	1
	open_mouth	short_hair	half-closed_eyes

linear

open_mouth	1		
short_hair	0.070174	1	
half-closed_eyes	0.071646	-0.032302	1
	open_mouth	short_hair	half-closed_eyes

logistic

open_mouth	1		
short_hair	-0.022995	1	
half-closed_eyes	-0.014277	0.030630	1
	open_mouth	short_hair	half-closed_eyes

lasso

open_mouth	1		
short_hair	0.069367	1	
half-closed_eyes	0.036207	0.075498	1
	open_mouth	short_hair	half-closed_eyes

SVC

open_mouth	1		
short_hair	0.820568	1	
half-closed_eyes	0.235060	0.048279	1
	open_mouth	short_hair	half-closed_eyes

svd

Dùng thêm tag đối nghịch:

open_mouth	1		
short_hair	0.057748	1	
half-closed_eyes	0.006926	0.016922	1
	open_mouth	short_hair	half-closed_eyes

open_mouth	1		
short_hair	0.000026	1	
half-closed_eyes	0.000562	0.014351	1
	open_mouth	short_hair	half-closed_eyes

Linear

open_mouth	1		
short_hair	0.067199	1	
half-closed_eyes	0.055971	0.006790	1
	open_mouth	short_hair	half-closed_eyes

Logistic

open_mouth	1		
short_hair	0.000834	1	
half-closed_eyes	-0.015304	0.015686	1
	open_mouth	short_hair	half-closed_eyes

Lasso

open_mouth	1		
short_hair	0.056574	1	
half-closed_eyes	0.034260	0.047268	1
	open_mouth	short_hair	half-closed_eyes

Svc

open_mouth	1		
short_hair	0.005951	1	
half-closed_eyes	0.284650	-0.081187	1
	open_mouth	short_hair	half-closed_eyes

svd

Từ các bảng độ đo Re-scoring trên ta có một số nhận xét sơ bộ:

- Khi sử dụng tag đối nghịch, hệ số tương quan giữa các vector của phương pháp đều giảm cho thấy việc hiệu quả của việc áp dụng phương pháp này.
- Đặc biệt bảng của phương pháp phân rã ma trận, hiệu quả của phương pháp này tác động rất mạnh

## 5.6 Ứng dụng trên video:



**Hình 5.8 Ứng dụng cho phong cách đã chỉnh sửa trên video**

Ở đây chúng tôi sử dụng GANs N' Roses để chuyển video mặt người sang mặt anime theo từng khung hình, sau đó chúng tôi chỉnh sửa vector phong cách để thay đổi nhân vật (khiến nhân vật chuyển sang tóc nâu). Như có thể thấy ở hình trên, các khung hình khi so sánh giữa nhân vật hoạt hình được khi chưa chỉnh sửa và đã chỉnh sửa đều có những hoạt họa giống nhau, không bị thêm hay bớt những chi tiết ngoài ý muốn. Điều này chứng minh cho việc chỉnh sửa vector phong cách một cách hợp lý vẫn bảo toàn được khả năng ứng dụng trên video của mô hình GANs N' Roses gốc.

## Chương 6

# Kết luận và hướng phát triển

### 6.1 Kết luận

Trong khóa luận này, từ kết quả nghiên cứu của Min Jin Chong và cộng sự [3] kết hợp với quy trình chỉnh sửa, nhóm đã đề xuất và cải tiến quy trình chỉnh sửa ảnh hoạt họa được tạo ra bởi quá trình đối kháng sinh mẫu bằng phương pháp có giám sát với mục tiêu cho hiệu quả chỉnh sửa tốt hơn .

Các chỉnh sửa ảnh được dựa trên việc thao túng miền không gian tiềm ẩn của mô hình đối kháng sinh mẫu, bằng kết hợp giữa các phương pháp tìm vector ngữ nghĩa cục bộ của miền không gian tiềm ẩn với sử dụng các vector ngữ nghĩa đối nghịch giúp ổn định quá trình chỉnh sửa và cho hiệu năng chỉnh sửa tốt hơn.

Các kết quả được đo trên chính tập dữ liệu mà Chong và cộng sự đã thực hiện để chứng minh việc chỉnh sửa không làm mất đi tính độc đáo gốc của mô hình, đồng thời nhóm cũng thêm một độ đo để so sánh sự hiệu quả giữa các phương pháp chỉnh sửa với nhau.

Nhờ sự thành công trong việc áp dụng các kỹ thuật tìm vector ngữ nghĩa và các kỹ thuật phụ họa mà kết quả chỉnh sửa mà nhóm đạt được tốt hơn nhiều so với phương pháp chỉnh sửa mà Chong và cộng sự đề xuất trong mô hình của họ.

### 6.2 Hướng phát triển

Có thể phát triển nghiên cứu thành một phần mềm giúp cho những người làm phim hoạt hình có thêm công cụ để tạo hình, hoạt hình cho quá trình thiết kế nhân vật và tạo chuyển động nhân vật. Ngoài ra có thể ứng dụng phương pháp thao túng miền không gian tiềm ẩn được nhóm đề xuất trên các mô hình đối kháng sinh mẫu khác nhằm tạo mẫu theo ý muốn của người sử dụng.

# Tài liệu tham khảo

- [1] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, và T. Aila, “Analyzing and Improving the Image Quality of StyleGAN”, *ArXiv191204958 Cs Eess Stat*, tháng 3 2020, Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1912.04958>
- [2] I. Sutskever, O. Vinyals, và Q. V. Le, “Sequence to Sequence Learning with Neural Networks”, *ArXiv14093215 Cs*, tháng 12 2014, Truy cập: 25 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1409.3215>
- [3] M. J. Chong và D. Forsyth, “GANs N’ Roses: Stable, Controllable, Diverse Image to Image Translation (works for videos too!)”, *ArXiv210606561 Cs*, tháng 6 2021, Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/2106.06561>
- [4] S. Ardila, “How Long Does It Take To Learn To Draw Anime? All Skill Levels”, *Enhance Drawing*. <https://enhancedrawing.com/how-long-does-it-take-to-learn-to-draw-anime/> (truy cập 18 Tháng Giêng 2022).
- [5] “How to Control Speed of Your Animation- FPS and Frame Repetition”. <https://www.kdanmobile.com/animation-desk/software-windows/animation-motion-speed-FPS> (truy cập 18 Tháng Giêng 2022).
- [6] “VFX Motion Capture | Media & Entertainment”, *Vicon*. <https://www.vicon.com/applications/vfx/> (truy cập 18 Tháng Giêng 2022).
- [7] “[2007.06600] Closed-Form Factorization of Latent Semantics in GANs”. <https://arxiv.org/abs/2007.06600> (truy cập 24 Tháng Chạp 2021).
- [8] K. Pearson, “LIII. On lines and planes of closest fit to systems of points in space”, *Lond. Edinb. Dublin Philos. Mag. J. Sci.*, vol 2, số p.h 11, tr 559–572, tháng 11 1901, doi: 10.1080/14786440109462720.
- [9] K. Kim, *DeepDanbooru*. 2021. Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <https://github.com/KichangKim/DeepDanbooru>
- [10] K. He, X. Zhang, S. Ren, và J. Sun, “Deep Residual Learning for Image Recognition”, *ArXiv151203385 Cs*, tháng 12 2015, Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1512.03385>
- [11] “Papers with Code - selfie2anime Dataset”. <https://paperswithcode.com/dataset/selfie2anime> (truy cập 19 Tháng Giêng 2022).
- [12] “Background: What is a Generative Model? | Generative Adversarial Networks”, *Google Developers*. <https://developers.google.com/machine-learning/gan/generative> (truy cập 22 Tháng Giêng 2022).
- [13] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, và K. Kavukcuoglu, “Conditional Image Generation with PixelCNN Decoders”, *ArXiv160605328 Cs*, tháng 6 2016, Truy cập: 22 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1606.05328>
- [14] “Generative Modeling: What is a Variational Autoencoder (VAE)?”, *MLQ.ai*, 1 Tháng Sáu 2021. <https://www.mlq.ai/what-is-a-variational-autoencoder/> (truy cập 22 Tháng Giêng 2022).
- [15] K. Sudhir, “Generative Adversarial Networks- History and Overview”, *Medium*, 22 Tháng Sáu 2017. <https://towardsdatascience.com/generative-adversarial-networks-history-and-overview-7effbb713545> (truy cập 22 Tháng Giêng 2022).

- [16] J.-Y. Zhu, T. Park, P. Isola, và A. A. Efros, “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”, *ArXiv170310593 Cs*, tháng 8 2020, Truy cập: 19 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1703.10593>
- [17] P. Isola, J.-Y. Zhu, T. Zhou, và A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks”, *ArXiv161107004 Cs*, tháng 11 2018, Truy cập: 19 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1611.07004>
- [18] X. Xu, D. Sun, J. Pan, Y. Zhang, H. Pfister, và M.-H. Yang, “Learning to Super-Resolve Blurry Face and Text Images”, trong *2017 IEEE International Conference on Computer Vision (ICCV)*, tháng 10 2017, tr 251–260. doi: 10.1109/ICCV.2017.36.
- [19] M.-Y. Liu, T. Breuel, và J. Kautz, “Unsupervised Image-to-Image Translation Networks”, trong *Advances in Neural Information Processing Systems*, 2017, vol 30. Truy cập: 20 Tháng Giêng 2022. [Online]. Available at: <https://proceedings.neurips.cc/paper/2017/hash/dc6a6489640ca02b0d42dabeb8e46bb7-Abstract.html>
- [20] “The Official Pokémon Website | Pokemon.com”. <https://www.pokemon.com/us/> (truy cập 20 Tháng Giêng 2022).
- [21] Y. Choi, Y. Uh, J. Yoo, và J.-W. Ha, “StarGAN v2: Diverse Image Synthesis for Multiple Domains”, *ArXiv191201865 Cs*, tháng 4 2020, Truy cập: 20 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1912.01865>
- [22] “CelebFaces Attributes (CelebA) Dataset”. <https://kaggle.com/jessicali9530/celeba-dataset> (truy cập 24 Tháng Giêng 2022).
- [23] H. Zhang và c.s., “StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks”, *ArXiv161203242 Cs Stat*, tháng 8 2017, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1612.03242>
- [24] H. Zhang và c.s., “StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks”, *ArXiv171010916 Cs Stat*, tháng 6 2018, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1710.10916>
- [25] A. van den Oord, N. Kalchbrenner, và K. Kavukcuoglu, “Pixel Recurrent Neural Networks”, *ArXiv160106759 Cs*, tháng 8 2016, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1601.06759>
- [26] M. Arjovsky, S. Chintala, và L. Bottou, “Wasserstein GAN”, *ArXiv170107875 Cs Stat*, tháng 12 2017, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1701.07875>
- [27] D. P. Kingma và P. Dhariwal, “Glow: Generative Flow with Invertible 1x1 Convolutions”, *ArXiv180703039 Cs Stat*, tháng 7 2018, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1807.03039>
- [28] J. Su, “GAN-QP: A Novel GAN Framework without Gradient Vanishing and Lipschitz Constraint”, *ArXiv181107296 Cs Stat*, tháng 12 2018, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1811.07296>
- [29] A. Karnewar và O. Wang, “MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks”, *ArXiv190306048 Cs Stat*, tháng 6 2020, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1903.06048>
- [30] H. Zhang, I. Goodfellow, D. Metaxas, và A. Odena, “Self-Attention Generative Adversarial Networks”, *ArXiv180508318 Cs Stat*, tháng 6 2019, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1805.08318>
- [31] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, và S. Levine, “Variational Discriminator Bottleneck: Improving Imitation Learning, Inverse RL, and GANs by

- Constraining Information Flow”, *ArXiv181000821 Cs Stat*, tháng 8 2020, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1810.00821>
- [32] A. Brock, J. Donahue, và K. Simonyan, “Large Scale GAN Training for High Fidelity Natural Image Synthesis”, *ArXiv180911096 Cs Stat*, tháng 2 2019, Truy cập: 24 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/1809.11096>
- [33] T. Karras, T. Aila, S. Laine, và J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation”, *ArXiv171010196 Cs Stat*, tháng 2 2018, Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1710.10196>
- [34] T. Karras, S. Laine, và T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks”, *ArXiv181204948 Cs Stat*, tháng 3 2019, Truy cập: 23 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1812.04948>
- [35] “machine learning - How does a Generator (GAN) create samples similar to the data space from a vector of random numbers?”, *Cross Validated*. <https://stats.stackexchange.com/questions/278623/how-does-a-generator-gan-create-samples-similar-to-the-data-space-from-a-vector-of-random-numbers> (truy cập 24 Tháng Giêng 2022).
- [36] Y. Shen, C. Yang, X. Tang, và B. Zhou, “InterFaceGAN: Interpreting the Disentangled Face Representation Learned by GANs”, *ArXiv200509635 Cs Eess*, tháng 10 2020, Truy cập: 25 Tháng Giêng 2022. [Online]. Available at: <http://arxiv.org/abs/2005.09635>
- [37] R. Horev, “Explained: A Style-Based Generator Architecture for GANs - Generating and Tuning Realistic...”, *Medium*, 2 Tháng Giêng 2019. <https://towardsdatascience.com/explained-a-style-based-generator-architecture-for-gans-generating-and-tuning-realistic-6cb2be0f431> (truy cập 28 Tháng Giêng 2022).
- [38] A. FUJII, “From GAN basic to StyleGAN2”, *Analytics Vidhya*, 12 Tháng Hai 2021. <https://medium.com/analytics-vidhya/from-gan-basic-to-stylegan2-680add7abe82> (truy cập 7 Tháng Hai 2022).
- [39] S. Hochreiter và J. Schmidhuber, “Long Short-Term Memory”, *Neural Comput.*, vol 9, số p.h 8, tr 1735–1780, tháng 11 1997, doi: 10.1162/neco.1997.9.8.1735.
- [40] “9.6. Encoder-Decoder Architecture — Dive into Deep Learning 0.17.1 documentation”. [https://d2l.ai/chapter\\_recurrent-modern/encoder-decoder.html](https://d2l.ai/chapter_recurrent-modern/encoder-decoder.html) (truy cập 25 Tháng Chạp 2021).
- [41] V. Dumoulin và c.s., “Adversarially Learned Inference”, *ArXiv160600704 Cs Stat*, tháng 2 2017, Truy cập: 25 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1606.00704>
- [42] J. Donahue, P. Krähenbühl, và T. Darrell, “Adversarial Feature Learning”, *ArXiv160509782 Cs Stat*, tháng 4 2017, Truy cập: 25 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1605.09782>
- [43] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, và O. Wang, “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”, *ArXiv180103924 Cs*, tháng 4 2018, Truy cập: 24 Tháng Chạp 2021. [Online]. Available at: <http://arxiv.org/abs/1801.03924>
- [44] C. Yang, Y. Shen, và B. Zhou, “Semantic Hierarchy Emerges in Deep Generative Representations for Scene Synthesis”, *ArXiv191109267 Cs*, tháng 2 2020, Truy cập: 8 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/1911.09267>
- [45] A. Voynov và A. Babenko, “Unsupervised Discovery of Interpretable Directions in the GAN Latent Space”, *ArXiv200203754 Cs Stat*, tháng 6 2020, Truy cập: 8 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/2002.03754>



- [46] E. Härkönen, A. Hertzmann, J. Lehtinen, và S. Paris, “GANSpace: Discovering Interpretable GAN Controls”, *ArXiv200402546 Cs*, tháng 12 2020, Truy cập: 8 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/2004.02546>
- [47] A. Creswell và A. A. Bharath, “Inverting The Generator Of A Generative Adversarial Network (II)”, *ArXiv180205701 Cs*, tháng 2 2018, Truy cập: 8 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/1802.05701>
- [48] Y. Shen, J. Gu, X. Tang, và B. Zhou, “Interpreting the Latent Space of GANs for Semantic Face Editing”, *ArXiv190710786 Cs*, tháng 3 2020, Truy cập: 8 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/1907.10786>
- [49] J. Kim, M. Kim, H. Kang, và K. Lee, “U-GAT-IT: Unsupervised Generative Attentional Networks with Adaptive Layer-Instance Normalization for Image-to-Image Translation”, *ArXiv190710830 Cs Eess*, tháng 4 2020, Truy cập: 15 Tháng Hai 2022. [Online]. Available at: <http://arxiv.org/abs/1907.10830>
- [50] “Anime Recommendations, Reviews, Manga and More! | Anime-Planet”. <https://www.anime-planet.com/> (truy cập 15 Tháng Hai 2022).
- [51] nagadomi, *lbpcascade\_animeface*. 2022. Truy cập: 15 Tháng Hai 2022. [Online]. Available at: [https://github.com/nagadomi/lbpcascade\\_animeface](https://github.com/nagadomi/lbpcascade_animeface)
- [52] nagadomi, *waifu2x*. 2022. Truy cập: 15 Tháng Hai 2022. [Online]. Available at: <https://github.com/nagadomi/waifu2x>
- [53] “[1409.1556] Very Deep Convolutional Networks for Large-Scale Image Recognition”. <https://arxiv.org/abs/1409.1556> (truy cập 3 Tháng Ba 2022).
- [54] “Danbooru: Anime Image Board”, *Danbooru*. <https://danbooru.donmai.us/> (truy cập 25 Tháng Hai 2022).
- [55] “API Reference”, *scikit-learn*. <https://scikit-learn/stable/modules/classes.html> (truy cập 16 Tháng Hai 2022).
- [56] “numpy.linalg.svd — NumPy v1.22 Manual”. <https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html> (truy cập 16 Tháng Hai 2022).