ChatGPT

Mạng Đối nghịch Tạo sinh trong Chuyển đổi Ảnh Chân dung sang Phong cách Anime

Chương 1: Mở đầu

1.1. Tính cấp thiết và Ý nghĩa của Đề tài

Sự phát triển mạnh mẽ của nghệ thuật kỹ thuật số cùng với thị trường Anime/Manga toàn cầu đã tạo động lực lớn cho các nghiên cứu về chuyển đổi phong cách hình ảnh. Theo thống kê năm 2024, quy mô thị trường anime toàn cầu đạt khoảng 81,96 tỷ USD và dự kiến vượt 200 tỷ USD vào năm 2034 [1]. Công nghệ AI hiện nay cũng đang dần xâm nhập vào quy trình sản xuất anime – ví dụ, các công cụ **Generative AI** đã có thể tự động hoá việc vẽ phông nền và tô màu, giúp giảm bớt công việc lặp lại cho các hoạ sĩ [2]. Trong bối cảnh đó, nhu cầu **tự động hoá chuyển đổi ảnh thực sang phong cách anime** trở nên cấp thiết, nhằm phục vụ cộng đồng người hâm mộ khổng lồ và ngành công nghiệp sáng tạo nội dung đang bùng nổ.

Tuy nhiên, việc chuyển một ảnh chân dung người thật sang tranh anime là thách thức không nhỏ. Phong cách anime có đặc trưng rất khác biệt (đường nét đơn giản, mắt to, màu sắc phẳng, v.v.), trong khi ảnh chụp chứa nhiều chi tiết thực tế phức tạp. Các phương pháp truyền thống dựa trên **Neural Style Transfer** thường gặp khó khăn trong việc giữ được nội dung gốc và **dễ sinh ra nhiễu, tạo tác (artifacts)** khi khác biệt phong cách quá lớn [3] [4]. Mặt khác, vẽ tay thủ công bởi các hoạ sĩ tuy chất lượng cao nhưng tốn kém thời gian và công sức. Do đó, bài toán đặt ra là làm thế nào để tự động hoá quá trình này mà vẫn đảm bảo **chất lượng cao** và **bảo toàn được đặc điểm nhận dạng** của đối tượng trong ảnh gốc.

Mạng Đối nghịch Tạo sinh (GAN) nổi lên như một công nghệ đột phá có thể giải quyết các nhiệm vụ tổng hợp hình ảnh phức tạp. Được Ian Goodfellow giới thiệu năm 2014 và được Yann LeCun ca ngợi là "ý tưởng thú vị nhất của ML trong 10 năm" [5], GAN đã chứng tỏ hiệu quả vượt trội trong việc sinh ảnh chân thực từ dữ liệu huấn luyện. Đặc biệt trong bài toán **dịch chuyển phong cách** (style transfer), GANs cung cấp một khuôn khổ linh hoạt để huấn luyện mô hình tạo ảnh anime từ ảnh thật mà không đòi hỏi dữ liệu ảnh cặp một-một. Nhờ GAN, những tiến bộ gần đây cho thấy khả năng tạo các hình ảnh **anime hóa** ngày càng sắc nét và chính xác hơn, mở ra hướng tiếp cận mới cho bài toán này [6] [7].

1.2. Mục tiêu Nghiên cứu

Mục tiêu tổng quát: Xây dựng và đánh giá một mô hình chuyển đổi ảnh chân dung người thật sang phong cách anime **chất lượng cao** dựa trên GAN, đảm bảo giữ được các nét nhận dạng chính của khuôn mặt gốc.

Mục tiêu cụ thể: - **Lựa chọn kiến trúc GAN phù hợp:** Nghiên cứu các kiến trúc GAN tiên tiến và chọn mô hình nền tảng hiệu quả nhất cho bài toán (dự kiến sử dụng kiến trúc **Double-Tail GAN (DTGAN)** – phiên bản AnimeGAN thế hệ mới). - **Xây dựng tập dữ liệu và tiền xử lý:** Thu thập hoặc tinh chỉnh tập dữ liệu gồm ảnh chân dung thực và ảnh anime tương ứng (dạng không ghép cặp), áp dụng các bước tiền xử lý như căn chỉnh khuôn mặt, chuẩn hóa kích thước, tăng cường dữ liệu. - **Đề xuất hàm mất mát và chiến lược huấn luyện tối ưu:** Thiết kế bộ hàm loss chuyên biệt (kết hợp giữa loss truyền thống của GAN và các loss phong

cách/anime đặc thù) cùng lịch trình huấn luyện thích hợp nhằm tăng độ ổn định và làm nổi bật chi tiết ảnh đầu ra. - **Đánh giá mô hình:** Đánh giá chất lượng ảnh anime sinh ra bằng cả phương pháp **định lượng** (ví dụ: FID, PSNR, LPIPS) và **định tính** (đánh giá cảm quan, khảo sát người dùng), so sánh với các phương pháp chuyển đổi phong cách hiện có để xác định hiệu quả của mô hình đề xuất.

1.3. Đối tượng và Phạm vi Nghiên cứu

Đối tượng nghiên cứu: Các kiến trúc mạng **Generative Adversarial Networks (GANs)** phục vụ cho nhiệm vụ dịch chuyển phong cách hình ảnh, đặc biệt là các mô hình chuyên dành cho chuyển ảnh người sang hoạt hình/anime. Ngoài ra, đề tài cũng liên quan đến các kỹ thuật thị giác máy tính và học sâu hỗ trợ, như mô hình **encoder-decoder**, các phương pháp **xử lý ảnh** (phát hiện/căn chỉnh khuôn mặt) và các hàm **tổn thất perceptual**.

Phạm vi: Đề tài tập trung vào **ảnh chân dung người** (face portraits) và phong cách **Anime 2D**. Cụ thể, ảnh đầu vào giới hạn ở ảnh khuôn mặt người thật (có thể chụp từ camera hoặc ảnh selfie), và ảnh đầu ra hướng đến phong cách tranh vẽ nhân vật anime 2D (phong cách hoạt hình Nhật Bản). Mô hình được huấn luyện và đánh giá trên tập dữ liệu ảnh chân dung và ảnh anime *không ghép cặp*. Những khía cạnh ngoài phạm vi bao gồm chuyển đổi phong cách cho video, phong cách 3D hoặc các phong cách hoạt hình khác (cartoon kiểu phương Tây, tranh phác hoạ, v.v.), cũng như không đi sâu vào các kỹ thuật diffusion models hay transformer mới hơn (chỉ tập trung vào GAN truyền thống trong giai đoạn 2020-2025).

1.4. Cấu trúc Luận văn

Luận văn được tổ chức thành **x chương** như sau:

- **Chương 1: Mở đầu** – Trình bày sự cần thiết, mục tiêu, phạm vi của đề tài và khái quát nội dung nghiên cứu.
- **Chương 2: Cơ sở Lý thuyết và Tổng quan Nghiên cứu** – Tổng hợp nền tảng lý thuyết về học sâu, thị giác máy, kiến trúc GAN, và các nghiên cứu liên quan trong lĩnh vực chuyển ảnh chân dung sang phong cách anime.
- **Chương 3: Phương pháp Nghiên cứu Đề xuất** – Mô tả chi tiết mô hình GAN đề xuất (kiến trúc **Double-Tail GAN**), các cải tiến kỹ thuật (như **LADE** – adaptive denormalization, hàm mất mát mới), cùng quy trình huấn luyện.
- **Chương 4: Thực nghiệm và Kết quả** – Trình bày thiết lập thực nghiệm, quá trình huấn luyện mô hình trên tập dữ liệu thu thập, và kết quả đánh giá so sánh với các phương pháp khác. Phân tích các kết quả định lượng và định tính, minh hoạ bằng các hình ảnh đầu vào/đầu ra.
- **Chương 5: Kết luận và Hướng phát triển** – Tóm tắt những đóng góp chính của luận văn, thảo luận những hạn chế còn tồn tại và đề xuất hướng nghiên cứu trong tương lai (mở rộng sang video, phong cách khác, ứng dụng diffusion model, v.v.).

Chương 2: Cơ sở Lý thuyết và Tổng quan Nghiên cứu

2.1. Cơ sở Lý thuyết về Học sâu và Thị giác Máy tính

Mạng nơ-ron tích chập (CNN): Đây là nền tảng của nhiều mô hình xử lý ảnh hiện đại. CNN bao gồm các **lớp tích chập (convolutional layers)** để trích xuất đặc trưng không gian từ ảnh, xen kẽ với các lớp **pooling** để giảm kích thước và chọn lọc đặc trưng nổi bật. Các **hàm kích hoạt (activation)** phi tuyến (ReLU,

LeakyReLU, v.v.) được sử dụng sau mỗi tầng để tăng tính phi tuyến, cho phép mạng học các biểu diễn phức tạp. Kiến trúc CNN điển hình có dạng *feature extractor* (trích đặc trưng qua nhiều conv layer + pooling) và *classifier* (các lớp kết nối đầy đủ để phân loại dựa trên đặc trưng). Trong bài toán dịch phong cách, CNN thường đóng vai trò làm bộ phận **encoder** trích xuất nội dung ảnh gốc, hoặc làm **discriminator** đánh giá ảnh sinh ra.

Mô hình mã hoá - giải mã (Encoder–Decoder): Đây là kiến trúc phổ biến trong các bài toán sinh ảnh. **Encoder** (thường là CNN) sẽ nén ảnh đầu vào thành một **đại diện tiềm ẩn (latent representation)** gọn hơn, chứa các thông tin nội dung chính. Sau đó **Decoder** (thường là mạng tích chập chuyển vị hoặc Upsampling) sẽ từ biểu diễn tiềm ẩn đó tái tạo thành ảnh đầu ra mong muốn (ví dụ ảnh phong cách anime). Để cải thiện chi tiết, giữa encoder và decoder thường có các kết nối tắt (skip connections) như trong mô hình **U-Net**. Kiến trúc encoder-decoder còn kết hợp với các **ResNet blocks** (khối mạng phần dư) ở phần giữa để giúp lan truyền thông tin tốt hơn và ổn định quá trình huấn luyện. Nhiều mô hình chuyển style ảnh (CartoonGAN, AnimeGAN, v.v.) đã sử dụng kiến trúc encoder-decoder như vậy làm nền tảng [8].

2.2. Kiến trúc Mạng Đối nghịch Tạo sinh (GAN)

Nguyên lý hoạt động: Mạng GAN bao gồm hai thành phần chính: **Mạng Sinh (Generator)** và **Mạng Phân biệt (Discriminator)**. Hai mạng này được huấn luyện đồng thời trong một **trò chơi hai người có tổng bằng không (zero-sum game)** [9]. Cụ thể, Generator nhận đầu vào là một véc-tơ nhiễu ngẫu nhiên (hoặc một ảnh nguồn cần chuyển đổi) và tìm cách **tạo ra ảnh giả trông giống ảnh thật**, trong khi Discriminator nhận vào ảnh (cả thật lẫn giả) và cố gắng **phân biệt ảnh nào thật, ảnh nào do Generator tạo ra**. Mục tiêu của Generator là "đánh lừa" Discriminator (tạo ảnh giả mà Discriminator nghĩ là thật), còn Discriminator thì cố gắng phân biệt chính xác. Quá trình này thiết lập nên một cơ chế học cạnh tranh: Generator không được học trực tiếp từ dữ liệu thật mà học **gián tiếp thông qua phản hồi của Discriminator** về mức độ ảnh trông "thật" [10]. Theo thời gian, Generator cải thiện chất lượng ảnh sinh ra, còn Discriminator cũng nâng cao khả năng phân biệt – cả hai đạt tới điểm cân bằng Nash khi ảnh giả không thể bị phân biệt với ảnh thật.

Hàm mất mát gốc (minimax GAN loss): Goodfellow et al. (2014) đã đề xuất hàm mục tiêu cho GAN dưới dạng tối ưu **minimax hai người chơi** [11]. Hàm mất mát cổ điển gồm hai phần: *(i)* loss cho Discriminator, thường là kỳ vọng của [$\ln D(x)$ + $\ln(1 - D(G(z)))$] – Discriminator cố gắng tối đa hóa biểu thức này (phân loại đúng ảnh thật x là thật và ảnh giả $G(z)$ là giả); *(ii)* loss cho Generator, ban đầu được định nghĩa là tối thiểu hóa $\ln(1 - D(G(z)))$ (tương đương Generator cố gắng làm Discriminator nhầm lẫn ảnh giả thành thật). Tuy nhiên dạng tối ưu này dẫn đến hiện tượng **gradient bão hòa** (generator dễ ngừng học khi Discriminator quá mạnh). Do đó, mẹo thực hành thường dùng là thay hàm loss Generator thành **$-\ln(D(G(z)))$** (gọi là *non-saturating loss*) để Generator tối đa hóa xác suất ảnh giả được phân loại là thật, giúp gradient ổn định hơn [12].

Các vấn đề trong huấn luyện GAN: Việc huấn luyện GAN nổi tiếng là khó khăn do đặc thù tối ưu hai mạng song song. Hai vấn đề phổ biến nhất là: **Mode Collapse** và **Training Instability**. *Mode collapse* xảy ra khi Generator **chỉ học được một vài mẫu đầu ra và lặp đi lặp lại**, bỏ sót nhiều mode (khả năng đa dạng) của dữ liệu thật [13]. Ví dụ, một GAN huấn luyện sinh chữ số có thể sụp đổ về chỉ sinh toàn số "0" nếu Discriminator từng tạm thời ưu ái phân biệt số 0 tốt hơn [14]. Nguyên nhân có thể do Generator cập nhật quá nhanh so với Discriminator, dẫn đến kẹt ở cực trị cục bộ [14]. Vấn đề thứ hai, *training instability*, là việc GAN không hội tụ: sự cân bằng giữa hai mạng rất mong manh, Discriminator quá mạnh sẽ làm Generator **mất gradient (vanishing gradient)**, ngược lại Discriminator quá yếu sẽ làm Generator **không học được gì**. Quá trình huấn luyện thường "đổ vỡ" thành các trạng thái thất bại như: Generator cho ra ảnh vô nghĩa, hoặc

dao động không ổn định [15]. Nhiều kỹ thuật đã được đề xuất để cải thiện ổn định, ví dụ: **two time-scale update rule (TTUR)** – cập nhật Generator với tốc độ chậm hơn Discriminator [16], dùng hàm mất mát cải tiến như **Wasserstein GAN** để tránh vanishing gradient, và áp dụng **huấn luyện curriculum** (tăng dần độ khó, ví dụ từ ảnh độ phân giải thấp lên cao) [15]. Mặc dù vậy, huấn luyện GAN đến nay vẫn đòi hỏi kinh nghiệm tinh chỉnh lớn và nghiên cứu vẫn đang tiếp tục nhằm giải quyết triệt để các vấn đề trên.

2.3. Các mô hình GAN tiêu biểu cho chuyển đổi ảnh sang Anime

2.3.1. CycleGAN và các biến thể (cho dữ liệu không cặp)

Một trong những bước ngoặt cho bài toán **dịch ảnh không cần dữ liệu cặp** là mô hình **CycleGAN** (Zhu et al., 2017) [17]. Kiến trúc CycleGAN gồm hai cặp mạng Generator–Discriminator cho hai chiều chuyển đổi: Generator G học chuyển ảnh từ miền nguồn X (ví dụ ảnh thật) sang miền đích Y (ví dụ tranh anime), còn Generator F học chuyển ngược từ Y về X. Đi kèm, hai Discriminator D_X và D_Y đảm nhiệm phân biệt ảnh thuộc từng miền. **Điểm cốt lõi** của CycleGAN là khái niệm **chu trình nhất quán (cycle consistency)**: để tránh Generator G thay đổi tuỳ ý khiến mất nội dung gốc, ta thêm ràng buộc rằng khi chuyển ảnh qua hai vòng ($X \xrightarrow{G} Y \xrightarrow{F} X$) thì phải thu được ảnh gần giống ảnh ban đầu [18]. Hàm loss *cycle-consistency* ($||F(G(X)) - X||$) buộc $F(G(x)) \approx x$ và tương tự $G(F(y)) \approx y$, giúp bảo toàn cấu trúc nội dung ban đầu. Nhờ đó, CycleGAN có thể học dịch chuyển phong cách giữa hai tập ảnh **không có ghép cặp trực tiếp** mà vẫn giữ lại được bố cục và nội dung quan trọng.

Ưu điểm: CycleGAN mở ra hướng tiếp cận rất **linh hoạt** cho dịch hình ảnh không cặp, phù hợp khi khó thu thập dữ liệu ghép cặp (như ảnh người và tranh anime tương ứng) [17]. Mô hình này đã được áp dụng rộng rãi cho nhiều bài toán *image-to-image translation*, từ chuyển mùa trong ảnh phong cảnh cho đến biến ảnh thật thành tranh Monet [19]. Đối với bài toán selfie2anime, CycleGAN cung cấp nền tảng ban đầu để huấn luyện trên các tập ảnh người và ảnh anime tách biệt, không cần từng người phải có một bản vẽ anime tương ứng. Ngoài ra, việc bổ sung loss nhất quán giúp **bảo toàn nội dung chính**, tránh kết quả bị sai lệch quá xa so với đầu vào.

Nhược điểm: Mặc dù bảo toàn nội dung, CycleGAN *cơ bản* thường gặp khó trong việc **thay đổi các chi tiết hình học lớn** cần thiết cho phong cách anime (ví dụ: tỷ lệ khuôn mặt, kích thước mắt) [20]. Mô hình gốc không có cơ chế chú ý đến các vùng quan trọng (như khuôn mặt) nên dễ bỏ sót một số đặc trưng cần biến đổi mạnh. Kết quả có thể vẫn giữ quá nhiều nét từ ảnh thật, chưa thực sự "anime hóa" hoàn toàn, hoặc ngược lại nếu cố biến đổi mạnh thì lại **mất nhận dạng khuôn mặt**. Hơn nữa, CycleGAN đôi khi tạo ra các **artifact (nhiễu, lem màu)** do hai bộ generator hoạt động độc lập theo chu trình nhưng không có ràng buộc trực tiếp về cấu trúc ngoài cycle-loss. Để khắc phục, nhiều **biến thể của CycleGAN** đã ra đời. Một ví dụ nổi bật là **AttentionGAN** (2018) – thêm module *chú ý* để tập trung biến đổi vùng trọng tâm, hay đặc biệt là **U-GAT-IT** (Kim et al., ICLR 2020) – mô hình cải tiến cho bài toán **selfie2anime**. U-GAT-IT giới thiệu một module chú ý học được và một kỹ thuật chuẩn hoá mới gọi là **Adaptive Layer-Instance Normalization (AdaLIN)**, cho phép điều chỉnh linh hoạt mức độ thay đổi hình dạng so với thay đổi kết cấu [21]. Nhờ đó, U-GAT-IT có thể thực hiện những biến đổi hình dạng lớn (như tạo mắt to, khuôn mặt V-line kiểu anime) trong khi **vẫn giữ được các nét nhận dạng** của người gốc, đạt kết quả vượt trội so với CycleGAN trên bộ dữ liệu selfie2anime [22]. Mô hình này trở thành một backbone phổ biến cho ứng dụng chuyển ảnh chân dung sang phong cách hoạt hình.

2.3.2. StyleGAN (cho ảnh chất lượng cao và khả năng kiểm soát)

Trong khi CycleGAN tập trung vào dịch chuyển giữa hai miền hình ảnh, **StyleGAN** (Karras et al.) lại là một dòng mô hình GAN sinh ảnh chất lượng cao với **không gian ẩn rời rạc (disentangled latent space)**. StyleGAN ban đầu được thiết kế cho sinh ảnh chân dung người có độ phân giải rất cao, nhưng các ý tưởng kiến trúc của nó tỏ ra hữu ích cho bài toán chuyển phong cách với **mức độ kiểm soát chi tiết**.

Kiến trúc: Điểm khác biệt chính của StyleGAN là cách đưa latent code vào mạng sinh. Thay vì đưa trực tiếp vector z vào đầu mạng như GAN truyền thống, StyleGAN sử dụng một **mạng ánh xạ (Mapping Network)** để biến z thành một vector **w trong không gian trung gian W** [23]. Sau đó, mỗi layer của **mạng tổng hợp (Synthesis Network)** sẽ nhận một biến thể của w (thông qua các **Affine transform** tạo thành các "style code") và áp dụng vào feature map thông qua cơ chế **Adaptive Instance Normalization (AdaIN)** [23] [24]. Thêm vào đó, StyleGAN còn đưa các **bản đồ nhiễu (noise maps)** vào từng layer để tạo đa dạng chi tiết ngẫu nhiên (như tàn nhang, nếp nhăn...). Thiết kế này giúp không gian latent W bớt rối hơn không gian Z, nghĩa là các thuộc tính sinh ảnh (như kiểu tóc, màu mắt, nét mặt) có thể được điều khiển gần như độc lập [25]. Kết quả là StyleGAN có khả năng **tùy biến phong cách** từng thành phần ảnh rất tốt (ví dụ, thay nền nhưng giữ khuôn mặt, hoặc thay kiểu tóc nhưng giữ khuôn miệng).

Cơ chế chuyển đổi kiểu StyleGAN (GAN Inversion): Đối với bài toán chuyển ảnh thực sang ảnh anime bằng StyleGAN, cách tiếp cận phổ biến là dùng kỹ thuật **đảo ngược GAN (GAN inversion)**. Thay vì huấn luyện trực tiếp mô hình dịch, ta tận dụng một mô hình StyleGAN đã huấn luyện trên ảnh anime (ví dụ, sinh chân dung nhân vật anime). Để "anime hóa" một ảnh chân dung thực, ta chiếu (project) ảnh đó vào không gian latent của StyleGAN (tìm vector w sao cho $G(w)$ gần giống ảnh gốc) rồi sau đó dùng chính generator StyleGAN để tạo ảnh ở miền anime. Nói cách khác, StyleGAN đóng vai trò như một **hàm dựng hình phong cách anime**, ta chỉ cần tìm được mã latent tương ứng với ảnh gốc [26]. Kỹ thuật này đòi hỏi một quá trình tối ưu (hoặc dùng encoder) để tìm được latent phù hợp, nhưng kết quả thường rất ấn tượng: ảnh đầu ra sắc nét, giữ lại **thần thái và danh tính** gốc tốt hơn so với các mô hình GAN thông thường. Thêm vào đó, vì StyleGAN có không gian latent disentangled, ta còn có thể **điều chỉnh mức độ phong cách** sau khi invert – ví dụ, thay đổi một chút latent để làm ảnh anime "cute" hơn hoặc theo phong cách hoạ sĩ cụ thể.

Ưu điểm: StyleGAN (đặc biệt là StyleGAN2, 2020) hiện được xem là **state-of-the-art** trong sinh ảnh chân dung chất lượng cao [27]. Ảnh tạo ra có độ phân giải cao, chi tiết sắc nét, ít nhiễu. Khi áp dụng cho anime, StyleGAN cũng cho thấy khả năng **bảo toàn danh tính** nhân vật rất tốt – các nét mặt đặc trưng của ảnh gốc (hình dáng khuôn mặt, vị trí ngũ quan) được giữ lại rõ ràng trong phiên bản anime. Hơn nữa, StyleGAN cho **khả năng điều khiển phong cách linh hoạt**: ta có thể trộn style (style mixing) giữa các ảnh, hoặc dùng các phương pháp định hướng bằng văn bản như **StyleGAN-NADA (2021)** để điều chỉnh mô hình sang các phong cách mới chỉ dựa trên **mô tả ngôn ngữ tự nhiên** [26]. Ví dụ, StyleGAN-NADA cho phép biến một generator pre-train (huấn luyện trên ảnh người thật) thành generator phong cách anime chỉ bằng cách "đòi" mô hình sinh ra ảnh giống với cụm từ khóa *"anime face drawing"* trong không gian CLIP [28]. Điều này cho thấy tiềm năng to lớn của StyleGAN trong việc làm chủ các phong cách khác nhau mà không cần dữ liệu huấn luyện trực tiếp.

Nhược điểm: Đánh đổi cho chất lượng cao là **độ phức tạp** và **tài nguyên tính toán**. StyleGAN có kiến trúc lớn với hàng chục triệu tham số; việc huấn luyện từ đầu đòi hỏi lượng dữ liệu và sức mạnh tính toán rất lớn (thường chỉ các phòng thí nghiệm hoặc công ty lớn mới huấn luyện từ đầu). Quá trình **GAN inversion** để xử lý từng ảnh cũng tốn thời gian (phải tối ưu hoặc chạy qua một mạng encoder nặng). Ngoài ra, StyleGAN được huấn luyện rời rạc cho từng domain: một mô hình StyleGAN khó có thể bao quát nhiều phong cách

cùng lúc. Mặc dù có phương pháp như Multi-styleGAN, nhưng nói chung nếu muốn chuyển sang phong cách khác (ví dụ phong cách phim Pixar hay cartoon phương Tây), ta phải huấn luyện hoặc tinh chỉnh lại mô hình cho domain đó. Điều này khác với các mô hình dịch như CycleGAN hay AnimeGAN vốn được thiết kế (với discriminator riêng) để làm việc trực tiếp với 2 domain. Tóm lại, StyleGAN mang lại chất lượng xuất sắc và khả năng kiểm soát cao, nhưng **chi phí triển khai** (về tính toán, thiết kế) cũng cao và quy trình sử dụng phức tạp hơn so với các mô hình chuyên biệt cho dịch ảnh.

2.3.3. Double-Tail GAN – AnimeGAN thế hệ mới

Gần đây, một hướng tiếp cận mới trong bài toán ảnh sang anime là mô hình có kiến trúc **"hai đuôi" (Double-Tail GAN)**, tiêu biểu là nghiên cứu của Liu *et al.* (2024) đề xuất **AnimeGANv3 (DTGAN)** [29]. Mô hình này kế thừa từ loạt AnimeGAN trước đó (v1 năm 2019 [8], v2 năm 2020) với mục tiêu cải thiện chất lượng ảnh và tốc độ suy luận. Điểm độc đáo là **Generator của DTGAN có hai nhánh đầu ra**: một *nhánh hỗ trợ* và một *nhánh chính*.

- **Nhánh Hỗ trợ (Support Tail):** sinh ra ảnh anime sơ bộ, mang phong cách anime nhưng có thể còn thô, chứa nhiễu và các lỗi nhỏ. Đây giống như bước phác thảo ban đầu, tạo nền tảng cho ảnh anime.
- **Nhánh Chính (Main Tail):** nhận đầu vào chính là ảnh anime thô từ nhánh hỗ trợ, rồi **tinh chỉnh, sửa lỗi và thêm chi tiết** để tạo thành ảnh anime cuối cùng chất lượng cao. Trong giai đoạn suy luận (inference), nhánh hỗ trợ được lược bỏ, chỉ giữ lại nhánh chính – do ảnh đầu vào cho nhánh chính lúc này có thể được giả lập thông qua kết quả trung gian. Nhờ vậy, **mô hình triển khai rất nhẹ**. Theo báo cáo, DTGAN chỉ có khoảng *1,02 triệu tham số ở* pha inference, nhỏ hơn nhiều so với các model trước đây [30].

Kiến trúc tổng thể của DTGAN vẫn dựa trên sơ đồ **encoder–decoder với các khối ResNet** tương tự AnimeGAN trước đây, nhưng việc thêm "đuôi kép" giúp quá trình sinh ảnh diễn ra hai bước: **coarse-to-fine** (từ ảnh thô đến ảnh tinh). Ý tưởng này nhằm tách nhiệm vụ: nhánh đầu học **tái tạo phong cách tổng quan** (màu sắc, nét vẽ anime) còn nhánh sau tập trung **loại bỏ artifact và tăng độ nét**. Để hỗ trợ quá trình này, nhóm tác giả còn đề xuất một kỹ thuật chuẩn hoá mới gọi là **LADE – Linearly Adaptive Denormalization** [31]. Đây là một cơ chế **chuẩn hoá có học** được tích hợp trong generator, giúp điều chỉnh linh hoạt thống kê đặc trưng ở các tầng mạng nhằm **ngăn chặn hiện tượng nhiễu** trong ảnh sinh. Trước đó, AnimeGAN v2 (2020) đã cho thấy tầm quan trọng của lựa chọn hàm chuẩn hoá: việc chuyển từ Instance Normalization sang **Layer Normalization** giúp giảm nhiễu lốm đốm trong ảnh output [32]. Giờ đây, **LADE** tiến thêm một bước khi cho phép mô hình **học cách kết hợp các đặc trưng đã chuẩn hoá và chưa chuẩn hoá** một cách tuyến tính tối ưu, thích nghi với từng nội dung ảnh [31]. Kết quả là ảnh anime tạo ra mượt mà hơn, hạn chế tối đa các artifact như vùng màu gai hay lem nét.

Ưu điểm: Nhờ kiến trúc và kỹ thuật mới, DTGAN đạt được nhiều ưu điểm nổi trội: - **Chất lượng hình ảnh vượt trội:** Ảnh đầu ra có độ phân giải và sắc nét cao, các mảng màu được làm phẳng mịn đúng chất anime (nhờ *region smoothing*), đồng thời các đường viền chi tiết được giữ rõ ràng (nhờ *fine-grained refinement*) [33]. Các thử nghiệm cho thấy DTGAN tạo ảnh anime nhìn tự nhiên hơn và **ít lỗi nhiễu** hơn so với các phương pháp trước đó [34]. - **Tốc độ xử lý nhanh, mô hình gọn nhẹ:** Với chỉ ~1 triệu tham số khi triển khai, AnimeGANv3 rất nhẹ, cho phép suy luận gần như **thời gian thực** kể cả trên thiết bị cấu hình thấp [30]. Đây là điểm cải tiến quan trọng so với các mô hình nặng như U-GAT-IT hay StyleGAN. Kiến trúc hai đuôi tuy phức tạp khi huấn luyện nhưng lại **tối giản khi suy luận** (vì bỏ đi một nhánh), tối ưu cho ứng dụng thực tế. - **Huấn luyện hiệu quả trên dữ liệu không ghép cặp:** DTGAN vẫn duy trì khả năng train end-to-end với dữ liệu không cần ghép cặp như các phiên bản AnimeGAN trước [30]. Nhờ các hàm loss đặc thù (bàn bên dưới) và kiến trúc hỗ trợ, mô hình học tốt quan hệ giữa ảnh người và ảnh anime mà không cần ảnh tương ứng

trực tiếp. - **Bảo toàn nội dung và danh tính:** Qua hai giai đoạn sinh ảnh, mô hình có điều kiện để giữ lại bố cục khuôn mặt và nét nhận dạng, sau đó mới áp phong cách. Điều này giúp **giảm nguy cơ mất đặc điểm** so với việc biến đổi một bước mạnh.

Nhược điểm: Bên cạnh ưu điểm, DTGAN cũng có vài hạn chế: - **Độ phức tạp huấn luyện:** Việc huấn luyện hai nhánh đồng thời đòi hỏi thiết kế hàm loss và lịch trình tinh tế. Nếu không cẩn thận, nhánh chính có thể phụ thuộc quá mức vào nhánh hỗ trợ hoặc ngược lại, dẫn đến hội tụ kém. Quá trình tinh chỉnh các siêu tham số (tỷ trọng các loss, tốc độ học của mỗi nhánh) trở nên phức tạp hơn so với GAN thông thường. - **Phụ thuộc vào mô hình tiền huấn luyện:** Để đạt kết quả tốt, thường cần sử dụng thêm các mô hình pre-trained hỗ trợ tính toán loss. Chẳng hạn, nhóm tác giả sử dụng **VGG network** pre-trained để tính **perceptual loss** hoặc **edge detector** để tính loss làm mịn vùng. Sự phụ thuộc này có thể gây khó khi triển khai độc lập, và cũng làm tăng thời gian tính toán trong quá trình huấn luyện. - **Hạn chế về tính tổng quát phong cách:** Mặc dù hỗ trợ nhiều style anime, mô hình vẫn chỉ giới hạn trong phạm vi phong cách đã huấn luyện. Nếu áp dụng cho phong cách hoạt hình khác (ví dụ phong cách cartoon phương Tây, 3D Pixar), mô hình cần được huấn luyện hoặc điều chỉnh lại. Khả năng **đa phong cách** không linh hoạt như hướng StyleGAN đa miền (với các tail chuyên biệt cho mỗi style như một số phiên bản AnimeGANv3 mở rộng hỗ trợ multi-style [35]). Nói cách khác, DTGAN rất hiệu quả cho phong cách anime 2D truyền thống, nhưng chưa chắc hiệu quả nếu phong cách đích thay đổi quá nhiều.

Tổng quan, Double-Tail GAN đánh dấu một hướng tiếp cận mới đầy hứa hẹn, kết hợp ưu điểm của các phương pháp trước (nhẹ như AnimeGAN, chất lượng cao như StyleGAN) để chuyên giải quyết bài toán chuyển ảnh chân dung người sang phong cách anime.

Chương 3: Phương pháp Nghiên cứu Đề xuất

3.1. Lựa chọn Kiến trúc Mô hình Cơ sở

Dựa trên tổng quan ở chương 2, luận văn này lựa chọn kiến trúc **AnimeGAN kiểu Double-Tail (DTGAN)** làm nền tảng cho mô hình đề xuất. Lý do lựa chọn là bởi DTGAN đã được chứng minh có hiệu suất xuất sắc trong nhiệm vụ chuyển ảnh thực sang ảnh anime: chất lượng ảnh đầu ra cao và mô hình gọn nhẹ hơn nhiều so với các đối thủ. Cụ thể, nghiên cứu của Liu *et al.* (2024) cho thấy AnimeGANv3 (DTGAN) có thể tạo ảnh anime chất lượng vượt trội và **vượt qua các mô hình state-of-the-art** trước đó trong các phép đo định lượng lẫn định tính [34]. Bên cạnh đó, kiến trúc Double-Tail rất phù hợp với mục tiêu của đề tài là **giữ được đặc điểm khuôn mặt**: cấu trúc hai nhánh (coarse-to-fine) giúp mô hình trước tiên nắm bắt bố cục gương mặt, sau đó mới áp phong cách và tinh chỉnh, do đó giảm thiểu nguy cơ mất mát danh tính so với chuyển đổi một lượt.

Tóm lại, **mô hình cơ sở** được chọn là **AnimeGANv3 (DTGAN)**, với một số điều chỉnh và mở rộng phù hợp để giải quyết bài toán cụ thể của luận văn. Phần tiếp theo sẽ mô tả chi tiết cấu trúc mô hình và các cải tiến kỹ thuật được đề xuất.

3.2. Cấu trúc Mô hình Đề xuất

3.2.1. Kiến trúc Generator “Double-Tail” (Hai đuôi)

Như đã trình bày, Generator của mô hình có hai nhánh đầu ra song song: - **Nhánh Hỗ trợ:** gồm một chuỗi các tầng tích chập và ResNet nhằm biến ảnh thật thành ảnh anime thô. Nhánh này chia sẻ chung phần

Encoder với nhánh chính ở giai đoạn đầu, sau đó tách ra một **Decoder phụ**. Ảnh anime thô ở đây giữ được bố cục và nội dung chính, nhưng chi tiết còn chưa sắc nét. - **Nhánh Chính:** nhận đầu vào là ảnh anime thô (từ nhánh hỗ trợ) cùng các đặc trưng trung gian, qua một **Decoder chính** để tạo ảnh anime hoàn thiện. Nhánh chính cũng có cấu trúc ResNet decoder tương tự nhưng tập trung vào hiệu chỉnh chi tiết và màu sắc.

Trong quá trình huấn luyện, cả hai nhánh đều hoạt động: nhánh phụ học tái hiện phong cách chung, nhánh chính học sửa lỗi. Tuy nhiên, khi suy luận, **chỉ nhánh chính được dùng**. Lúc này, ta cung cấp ảnh gốc vào encoder, đồng thời **bỏ qua nhánh phụ** bằng cách đưa trực tiếp đặc trưng vào decoder chính (có thể nối tắt hoặc nhân một trọng số rất nhỏ cho đầu ra nhánh phụ). Thủ thuật này khả thi do nhánh chính đã học cách dựa vào đầu ra nhánh phụ; trong triển khai, ta giả lập đầu ra phụ bằng chính đầu ra của encoder hoặc một module trung gian. Kết quả là ta có một mô hình suy luận chỉ với **một nửa cấu trúc** so với khi train, giúp giảm đáng kể tham số và độ trễ.

Về kiến trúc cụ thể, **Encoder** được xây dựng bằng các lớp conv 3x3 xen kẽ pooling để giảm dần kích thước đặc trưng, đi kèm các **ResNet blocks** để giữ lại thông tin chi tiết. **Decoder phụ** và **Decoder chính** đều bao gồm các khối **Residual Upsampling** (kết hợp upsample + conv + skip connection) để tăng dần độ phân giải lên lại như ảnh đầu vào. Các **skip-connection** giữa encoder và decoder chính được giữ lại (theo kiểu U-Net) để đảm bảo nhánh chính vẫn nhận được thông tin gốc, phòng trường hợp nhánh phụ làm sai lệch một vài chi tiết. Nhờ kiến trúc này, Generator “Double-Tail” có khả năng học hiệu quả cả **phần thô** lẫn **phần tinh** của ảnh anime.

3.2.2. Kỹ thuật Chuẩn hóa LADE (Linearly Adaptive Denormalization)

Một đóng góp quan trọng trong mô hình đề xuất là tích hợp cơ chế **LADE** vào các tầng của generator. Đây là kỹ thuật chuẩn hóa được điều chỉnh một cách **thích ứng tuyến tính**, lần đầu xuất hiện trong AnimeGANv3 [31]. Về bản chất, LADE cho phép mô hình học cách **pha trộn** giữa hai cực trị: (1) đặc trưng đã được chuẩn hoá (ví dụ bằng Instance Norm hoặc Layer Norm) và (2) đặc trưng gốc chưa chuẩn hoá, thông qua các hệ số học được.

Cụ thể, giả sử h là vector đặc trưng trước chuẩn hóa, μ và σ là trung bình và độ lệch chuẩn trên batch (hoặc layer). Thay vì xuất $h' = \frac{h - \mu}{\sigma}$ (chuẩn hóa thông thường) rồi áp affine (như AdaIN), LADE tính $h' = \alpha * \frac{h - \mu}{\sigma} + \beta * h$ với α, β là các hệ số học được (có thể phụ thuộc vào điều kiện style). Khi $\alpha=1, \beta=0$ ta được Instance Norm thuần túy, khi $ \alpha=0, \beta=1$ là giữ nguyên đặc trưng thô; các giá trị ở giữa cho phép một sự **kết hợp tuyến tính** giữa hai dạng. Ý tưởng LADE gần gũi với AdaLIN của U-GAT-IT [36], nhưng thay vì kết hợp layer-norm và instance-norm, LADE kết hợp normalization với identity mapping của chính đặc trưng đó.

Việc áp dụng LADE giúp mô hình giải quyết tốt hơn bài toán **artifact xuất hiện do quá chuẩn hóa**. Trước đây, người ta nhận thấy Instance Norm tuy loại bỏ thống kê không mong muốn nhưng dễ làm mất các **thông tin quan trọng (ví dụ màu da, ánh sáng)**, còn giữ nguyên thì lại bị ảnh hưởng bởi những biến động cục bộ. LADE cân bằng hai yếu tố này, cho phép mô hình **tự học mức độ chuẩn hóa phù hợp cho từng kênh đặc trưng**. Kết quả là ảnh sinh ra có vùng màu sắc đồng đều hơn (khử nhiễu màu lốm đốm), nhưng vẫn giữ được chuyển tiếp tự nhiên ở ranh giới các vùng – rất quan trọng để ảnh anime trông mượt mà mà không bị “phẳng lì” thiếu chiều sâu [37].

Trong triển khai, LADE được đặt ngay sau các ResNet block trong decoder. Hệ số α, β có thể được tính bởi một mạng con nhỏ (ví dụ MLP) dựa trên một vector điều khiển phong cách. Tuy nhiên, ở chế độ

không ghép cặp, ta để α, β như tham số tự do học được cho mỗi kênh. Ngoài ra, để ổn định, ta ràng buộc $\alpha + \beta = 1$ (hoặc giới hạn trong [0,1]) để đảm bảo giữ tỷ lệ năng lượng đặc trưng.

Như kết quả đã công bố, với LADE, mô hình **giảm hẳn hiện tượng artifacts** trong vùng nền trời, da mặt – những nơi trước kia hay xuất hiện nhiễu hạt khi dùng Instance Norm thuần túy [33] . Điều này góp phần lớn giúp ảnh anime đầu ra đạt chất lượng chuyên nghiệp.

3.2.3. Hệ thống Hàm Mất mát (Loss Functions) chuyên biệt

Để huấn luyện mô hình chuyển đổi phong cách, ta kết hợp nhiều thành phần hàm loss khác nhau, bao gồm cả các loss truyền thống của GAN lẫn các loss chuyên dụng cho phong cách anime: - **Adversarial Loss:** Đây là thành phần không thể thiếu, bao gồm loss cho Discriminator và Generator như mô tả ở mục 2.2. Ta sử dụng dạng **non-saturating GAN loss** cho ổn định [12] . Cụ thể, Discriminator D được huấn luyện tối đa hóa $L_D = \mathbb{E}*{x\sim P}*[\ln D(x)] + \mathbb{E}*{y\sim P}*[\ln(1 - D(y))]$. Generator G thì tối thiểu hóa $L_G = \mathbb{E}*{y\sim P$ là phân phối ảnh anime do generator tạo ra từ ảnh thực. - }}[-\ln D(y)]$. Ở đây $P_{gen**Content Preservation Loss:** Để đảm bảo ảnh output giữ lại nội dung và bố cục gốc, ta dùng **Identity loss/cycle loss/perceptual loss**. Vì ta không có ảnh anime "thật" tương ứng từng ảnh người, nên không dùng được pixel-loss trực tiếp. Thay vào đó, ta áp dụng **Cycle-Consistency Loss** một chiều: chuyển ảnh anime ngược về ảnh thật (sử dụng một mạng generator phụ F ngược hoặc đơn giản dùng chính output overlay với input) để phạt nếu mất nội dung. Tuy nhiên, cách hiệu quả hơn là dùng **Perceptual Loss**: cho ảnh gốc và ảnh sinh qua một mạng pretrained (VGG-19) và tính MSE giữa các đặc trưng layer cao. Loss này buộc ảnh anime phải giống ảnh thật ở các đặc trưng trừu tượng (như vị trí mắt, mũi, miệng). Ngoài ra, ta cũng áp dụng **Identity loss**: đưa ảnh anime gốc (nếu có) qua generator và yêu cầu thu được chính nó (không thay đổi nhiều ảnh đã đúng style) [19] – cách này giúp generator không dịch sai màu khi ảnh đầu vào vốn đã giống ảnh đích. - **Style Loss:** Đây là thành phần lấy cảm hứng từ **Neural Style Transfer**. Ta muốn ảnh đầu ra có texture, màu sắc giống phong cách anime. Một cách làm là dùng **Gram-matrix loss** trên các đặc trưng của ảnh output so với ảnh anime mẫu. Tuy nhiên, trong thiết lập không cặp, ta áp dụng style loss theo kiểu **Statistic Matching**: buộc phân phối màu sắc tổng thể của ảnh output gần với phân phối ảnh anime (có thể qua Regularization trên mean/var các kênh màu). AnimeGAN v1 từng đề xuất **Grayscale style loss** – tính loss giữa phiên bản xám của ảnh output và ảnh anime để tập trung vào texture hơn là màu [4] . Chúng tôi cũng thử nghiệm áp dụng ý tưởng tương tự. - **Loss chuyên biệt cho anime:** Điểm mới trong mô hình là hai hàm mất mát được điều chỉnh riêng cho ảnh anime, dựa theo đề xuất của Liu *et al.* (2024): - **Region Smoothing Loss:** hàm mất mát làm mịn vùng ảnh [33] . Ta nhận thấy tranh anime thường có mảng màu phẳng, ít chi tiết texture như ảnh thật. Do đó, loss này khuyến khích output giảm bớt các chi tiết texture nhỏ không cần thiết. Cách thực hiện: áp dụng bộ lọc làm mịn (ví dụ Gaussian) lên ảnh output, và phạt sự khác biệt giữa ảnh gốc và ảnh đã làm mịn ở những vùng nhất định (như da, bầu trời). Kết quả là output có xu hướng **đơn giản hóa các hoa văn, texture** trong vùng đồng nhất, tạo hiệu ứng *"đổ màu phẳng"* đặc trưng của anime [38] . - **Fine-Grained Revision Loss:** hàm mất mát sửa chi tiết [38] . Trái ngược với loss trên, loss này tập trung vào **bảo vệ và làm sắc nét các chi tiết ranh giới** (ví dụ viền nét vẽ, cạnh tóc, mắt). Ta có thể sử dụng kỹ thuật **edge loss**: trích biên (edge) của ảnh output bằng Sobel hoặc Canny, đồng thời trích biên của ảnh input (hoặc ảnh anime tham chiếu), sau đó tính loss L1 giữa hai map biên. Thêm nữa, để loại bỏ hoàn toàn nhiễu, ta đào tạo một mạng phân biệt phụ cho tác vụ "phát hiện artifact": mạng này cố gắng phân biệt vùng ảnh output nào bị nhiễu (so với ảnh anime thật mượt mà). Generator sẽ nhận thêm tín hiệu để loại bỏ những điểm bất thường đó. Tổng hợp lại, fine-grained loss giúp **giữ cạnh rõ nét, nền vùng trơn tru không đốm nhiễu** [38] . - **Regularization Loss khác:** Bao gồm các thành phần như **TV-loss (Total Variation)** để ảnh đầu ra mượt liên tục, **GAN feature matching loss** (cho generator học thống kê trung gian của ảnh thật ở discriminator để ổn định hơn), v.v.

Tổng hàm mất mát của Generator là tổ hợp có trọng số của các thành phần trên. Các trọng số được chọn thông qua thử nghiệm, đảm bảo một sự cân bằng giữa **bảo toàn nội dung** và **áp phong cách anime**. Nhờ các hàm mất mát chuyên biệt (đặc biệt hai hàm smoothing và revision nói trên), mô hình học tốt hơn các đặc trưng riêng của ảnh anime: những vùng lớn cần phẳng màu thì phẳng hẳn, còn nét vẽ quan trọng thì được nhấn mạnh rõ nét. Kết quả là ảnh chuyển đổi trông **hài hòa và ít lỗi** hơn hẳn so với chỉ dùng các loss thông thường của GAN.

3.4. Tập dữ liệu và Tiền xử lý

Tập dữ liệu nguồn (ảnh chân dung người thật): Chúng tôi sử dụng tập ảnh chân dung khuôn mặt người đa dạng về tuổi tác, giới tính, sắc tộc để mô hình học được các đặc trưng phổ quát. Các ảnh nguồn được thu thập từ các dataset công khai (VD: **CelebA-HQ**, **FFHQ**) cũng như ảnh selfie từ Internet. Để tăng cường tính đa dạng, chúng tôi bổ sung một số ảnh chụp ngoài trời, trong nhà, với các biểu cảm khác nhau. Tổng số ảnh thật sử dụng khoảng vài nghìn. Chẳng hạn, một phần dữ liệu đến từ bộ **Selfie2Anime (UGATIT)** gồm ~3,400 ảnh chân dung phụ nữ và 3,400 ảnh nhân vật anime tương ứng không cặp [39] .

Tập dữ liệu đích (ảnh anime): Bao gồm các hình ảnh nhân vật anime 2D được sưu tầm từ phim hoạt hình và truyện tranh. Để phù hợp với ảnh chân dung, ta chọn các ảnh anime tập trung vào khuôn mặt nhân vật (các cảnh cận mặt). Nguồn ảnh anime có thể lấy từ các bộ phim nổi tiếng (Studio Ghibli, Shinkai Makoto) hoặc dataset có sẵn (VD: **Danbooru2018** về tranh anime). Nhằm đảm bảo chất lượng, ta ưu tiên ảnh anime độ phân giải cao, nét vẽ rõ ràng. Theo kinh nghiệm từ AnimeGANv2, việc huấn luyện trên các frame phim anime chất lượng cao (như *Your Name*, *Paprika*) giúp cải thiện đáng kể kết quả [32] – do đó dữ liệu đích của ta cũng bao gồm những frame chọn lọc từ các phim này. Tổng số ảnh anime sử dụng tương đương tập ảnh người (vài nghìn ảnh).

Tiền xử lý: Trước khi đưa vào huấn luyện, cả ảnh thật và ảnh anime đều được tiền xử lý qua các bước: - **Căn chỉnh và cắt khuôn mặt:** Áp dụng thuật toán phát hiện khuôn mặt (MTCNN, dlib) để xác định các điểm mặt (mắt, mũi, miệng), từ đó xoay và căn giữa ảnh cho thẳng mặt. Sau đó cắt vùng khuôn mặt theo một khung cố định (ví dụ vùng chứa cả tóc và cằm). Bước này nhằm chuẩn hóa đầu vào, giúp mô hình tập trung vào vùng mặt chính, loại bỏ nền quá thừa. - **Chuẩn hóa kích thước:** Tất cả ảnh được resize về cùng kích thước (thí nghiệm dùng 256×256 pixel). Điều này đồng nhất độ phân giải, tránh scale khác ảnh hưởng việc học. Ảnh anime thường có tỷ lệ khung hình khác, ta cắt hoặc padding phù hợp trước khi resize. - **Loại bỏ nhiễu và điều chỉnh màu:** Với ảnh thật, có thể áp dụng một số bước làm đẹp (beautify) nhẹ như cân bằng sáng tối, loại mắt đỏ... Với ảnh anime, do thu thập từ nhiều nguồn có thể có ảnh nền phức tạp, ta loại các ảnh có nền text, sub hoặc logo. Đôi khi cũng chuyển đổi ảnh anime sang không gian màu phù hợp (sRGB) để thống nhất. - **Data Augmentation:** Để tăng dữ liệu hiệu quả, ta áp dụng các phép biến đổi ngẫu nhiên: lật ngang ảnh (flip), thay đổi độ sáng, tương phản một chút, xoay nhẹ (±5 độ), phóng to/thu nhỏ nhẹ. Đặc biệt, phép **flip ngang** rất hữu ích vì ảnh mặt người hay anime khi lật vẫn hợp lý và gần như tăng gấp đôi dữ liệu [40] . Ta tránh các phép biến đổi quá mạnh (như xoay 90°, lật dọc) vì sẽ làm sai lệch cấu trúc khuôn mặt. - **Chuẩn hóa đầu vào:** Cuối cùng, ảnh được chuẩn hóa pixel (ví dụ trừ trung bình 0.5, chia độ lệch 0.5 nếu dùng range -1..1) trước khi đưa vào mạng.

Sau tiền xử lý, chúng tôi thu được hai tập ảnh đã cân bằng và tương đồng về phân bố tổng quát (ví dụ độ sáng, vị trí khuôn mặt). Quá trình huấn luyện sẽ lấy ngẫu nhiên một batch gồm ảnh thật và một batch ảnh anime rồi đưa qua mô hình GAN không ghép cặp (theo kiến trúc đã chọn). Việc chuẩn bị dữ liệu kỹ lưỡng như trên giúp mô hình học **ổn định hơn** và kết quả chuyển đổi **đạt chất lượng cao**, hạn chế các lỗi do khác biệt dữ liệu gây ra.

(Các phần tiếp theo của luận văn sẽ trình bày về quá trình huấn luyện mô hình trên cơ sở kiến trúc và dữ liệu đã nêu, cùng với việc phân tích kết quả đạt được.)

---

1 2 Anime Market Size Hit Around USD 203.68 Billion by 2034
https://www.precedenceresearch.com/anime-market

3 4 8 AnimeGAN: A Novel Lightweight GAN for Photo Animation | SpringerLink
https://link.springer.com/chapter/10.1007/978-981-15-5577-0_18

5 11 12 Understanding GAN Loss Functions
https://neptune.ai/blog/gan-loss-functions

6 7 20 21 22 26 28 32 35 36 Human To Anime. Human-to-Anime Image Conversion with... | by CodeMass | Medium
https://medium.com/@codemass/human-to-anime-baafbf0e7490

9 10 13 14 15 16 Generative adversarial network - Wikipedia
https://en.wikipedia.org/wiki/Generative_adversarial_network

17 18 19 Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks
https://openaccess.thecvf.com/content_ICCV_2017/papers/Zhu_Unpaired_Image-To-Image_Translation_ICCV_2017_paper.pdf

23 24 25 27 Analyzing and Improving the Image Quality of StyleGAN
https://openaccess.thecvf.com/content_CVPR_2020/papers/Karras_Analyzing_and_Improving_the_Image_Quality_of_StyleGAN_CVPR_2020_paper.pdf

29 30 31 33 34 37 38 A Novel Double-Tail Generative Adversarial Network for Fast Photo Animation
https://www.jstage.jst.go.jp/article/transinf/E107.D/1/E107.D_2023EDP7061/_article/-char/ja

39 40 selfie2anime - Kaggle
https://www.kaggle.com/datasets/arnaud58/selfie2anime