

GAN and Its Applications

Võ Thành Nam, Phạm Minh Đức, Trần Quốc Hưng

February 2024

Tóm tắt nội dung

Mạng đối nghịch tạo sinh (GANs) trình bày một cách để tìm hiểu các biểu diễn sâu mà không cần dữ liệu đào tạo được chú thích rộng rãi. Các mạng này đạt được việc học thông qua việc lấy các tín hiệu truyền ngược thông qua một quá trình cạnh tranh liên quan đến một cặp mạng. Các biểu diễn mà GANs có thể học được có thể được sử dụng trong một số ứng dụng. GAN đã có những tiến bộ đáng kể và hiệu suất vượt trội trong nhiều ứng dụng. Các ứng dụng thiết yếu bao gồm chỉnh sửa hình ảnh ngữ nghĩa, chuyển kiểu, tổng hợp hình ảnh, siêu phân giải và phân loại hình ảnh. Bài viết này nhằm mục đích trình bày tổng quan về GAN, các biến thể khác nhau của nó và ứng dụng tiềm năng trong các lĩnh vực khác nhau.

Bài viết cố gắng xác định những ưu điểm, nhược điểm và những thách thức đáng kể của GAN đối với việc triển khai thành công GAN trong các lĩnh vực ứng dụng khác nhau. Mục tiêu của nghiên cứu này là thực hiện đánh giá toàn diện mô hình GAN trong Thị Giác Máy Tính bằng các tài liệu liên quan và trình bày tóm về tổng quan kiến thức có được từ GAN, bao gồm lý thuyết đằng sau nó, mục đích sử dụng của nó, những đột phá mới nhất trong lĩnh vực này. Bài báo này sẽ giúp bạn có được cái nhìn toàn diện về GAN và cung cấp tổng quan về GAN và các loại mô hình khác nhau của nó, cũng như các triển khai phổ biến, đề xuất các thông số đo lường và các ứng dụng GAN trong xử lý ảnh. Ngoài ra, bài báo cũng sẽ đề cập đến một số ứng dụng của GAN trong xử lý ảnh, cùng với những lợi ích và hạn chế của chúng, cũng như phạm vi tiềm năng của chúng đối với Thị Giác Máy Tính trong tương lai.

1 Giới thiệu chung về bài toán GANs

(learning) được giới thiệu bởi Ian Goodfellow và đồng nghiệp vào năm 2014, hướng tới việc sinh ra các tập

1.1 Tổng quan

GANs(1) là một dạng mạng nơ-ron 1 đặc biệt trong lĩnh vực học sâu (deep

dữ liệu mới sau quá trình học. GANs có thể tự tạo sinh ra hình ảnh, khuôn mặt mới, một nhân vật mới hay là đoạn văn, chữ viết, bài nhạc và những thứ tương tự.

GANs viết tắt cho Generative Adversarial Networks, với generative có nghĩa là tạo sinh, networks nghĩa là kiến trúc mạng (mô hình) còn Adversarial có nghĩa là đảo nghịch, sở dĩ GANs có cái tên này là do cấu tạo bên trong GANs được cấu tạo từ 2 mạng có tên là Discriminator và Generator, 2 mạng này được xem là đối thủ truyền kiếp của nhau, luôn đối đầu nhau trong quá trình huấn luyện mạng GAN

1.2 Động lực về mặt khoa học nghiên cứu

Việc nghiên cứu về GANs là một lĩnh vực nghiên cứu quan trọng trong khoa học máy tính và trí tuệ nhân tạo, tầm quan trọng trong việc nghiên cứu rất lớn và đa chiều, với ảnh hưởng sâu rộng đến nhiều lĩnh vực khác nhau trong khoa học và ứng dụng. Nó liên quan đến việc tạo sinh dữ liệu mới từ những dữ liệu đã có, đó là vấn đề tuy nghe khá là đơn giản nhưng để giải quyết nó thật sự là phức tạp. Các nhà nghiên cứu trong lĩnh vực này không ngừng tìm tòi và phát triển các phương pháp mới để cải thiện độ hiệu quả, chất lượng và độ chính xác của hệ thống tạo sinh này. Họ cũng đã và đang tìm hiểu cách để ứng dụng các công nghệ mới vào nghiên cứu.

1.3 Động lực về mặt thực tiễn

Mạng đối thủ tạo sinh (GAN) đang thu hút sự quan tâm ngày càng tăng trong cộng đồng học sâu. GAN đã được áp dụng cho nhiều lĩnh vực khác nhau như thị giác máy tính, xử lý ngôn ngữ tự nhiên, tổng hợp chuỗi thời gian, phân đoạn ngữ nghĩa v.v. GAN thuộc họ các mô hình tổng quát trong học máy. So với các mô hình tổng quát khác, ví dụ: bộ mã hóa tự động biến thiên, GAN mang lại những ưu điểm như khả năng xử lý các hàm mật độ ước tính sắc nét, tạo ra các mẫu mong muốn một cách hiệu quả, loại bỏ sai lệch xác định và có khả năng tương thích tốt với kiến trúc thần kinh bên trong. Những đặc tính này đã cho phép GAN đạt được thành công lớn, đặc biệt là trong lĩnh vực thị giác máy tính, ví dụ: tạo hình ảnh hợp lý, dịch từ hình ảnh sang hình ảnh, hình ảnh siêu phân giải và hoàn thiện hình ảnh.

1.4 Thách thức của GANs

Tuy nhiên, GAN không phải là không có vấn đề. Hai điều quan trọng nhất là chúng khó huấn luyện và khó đánh giá. Xét về độ khó huấn luyện, nếu bộ phân biệt làm việc quá tốt thì việc đào tạo cho bộ tạo sinh có thể không thành công do độ dốc biến mất (vanishing gradient) là hiện tượng mà gradient truyền ngược từ phần dưới của mạng không thể truyền qua các lớp trên một cách hiệu quả, dẫn đến việc

các lớp phía trước không nhận được đủ thông tin từ gradient để cập nhật các tham số. Điều này có thể xảy ra khi các giá trị gradient trở nên rất nhỏ hoặc gần như biến mất khi di chuyển từ lớp cuối cùng của mạng ngược lại các lớp đầu tiên. việc bộ phân biệt và bộ tạo đạt được trạng thái cân bằng Nash trong quá trình đào tạo là điều không hề nhỏ và việc bộ tạo không thể học tốt, phân phối đầy đủ các tập dữ liệu là điều thường thấy. Khía cạnh đầu tiên liên quan trực tiếp đến hiệu suất của GAN, ví dụ: chất lượng hình ảnh, tính đa dạng của hình ảnh và quá trình huấn luyện ổn định. Phần lớn nghiên cứu GAN hiện tại có thể được xem xét theo hai mục tiêu sau: Cải thiện đào tạo và triển khai GAN cho các ứng dụng trong thế giới thực. Cái trước tìm cách cải thiện hiệu suất của GAN và do đó là nền tảng cho cái sau. Tức là đã xuất hiện nhiều biến thể GAN đã được đề xuất trong tài liệu để cải thiện hiệu suất. Thông thường, GAN tạo ra nhiều loại kết quả. Ví dụ: muốn có một khuôn mặt khác cho mỗi đầu vào ngẫu nhiên vào trình tạo khuôn mặt. Tuy nhiên, nếu trình tạo tạo ra kết quả đặc biệt hợp lý, thì trình tạo có thể tìm hiểu để chỉ tạo kết quả đó. Trên thực tế, trình tạo luôn cố gắng tìm một đầu ra có vẻ hợp lý nhất với bộ phân biệt. Nếu trình tạo bắt đầu tạo ra nhiều đầu ra (hoặc một tập hợp nhỏ các đầu ra) nhiều lần, thì chiến lược phân biệt tốt nhất là học cách luôn từ chối đầu ra đó. Nhưng nếu thế hệ bộ phân biệt tiếp theo bị kẹt ở mức tối thiểu cục bộ và không tìm thấy chiến lược tốt

nhất, thì việc lặp lại bộ tạo tiếp theo sẽ trở nên quá dễ dàng để tìm ra đầu ra hợp lý nhất cho bộ phân biệt hiện tại. Mỗi lần lặp lại của trình tạo sẽ tối ưu hóa quá mức cho một bộ phân biệt cụ thể và bộ phân biệt không bao giờ tìm cách thoát khỏi bãy. Do đó, trình tạo sẽ xoay vòng qua một tập hợp nhỏ các loại đầu ra. Quá trình trên được gọi là thu gọn chế độ(mode collapse).

Ngoài ra, việc đánh giá chất lượng của các mô hình GANs cũng là một thách thức. Trong nhiều trường hợp, việc đánh giá đối với các hình ảnh được tạo ra bởi GANs là một quá trình mở, không có một phương pháp chuẩn xác và toàn diện. Điều này làm cho việc so sánh giữa các mô hình và đánh giá sự đa dạng và chất lượng của các ảnh trở nên khó khăn.

Đánh giá về không gian tiềm ẩn của GANs cũng vô cùng phức tạp. Đánh giá này yêu cầu đo lường sự phân tán, tính ngẫu nhiên và khả năng kiểm soát của các chiều trong không gian tiềm ẩn.

2 Phát biểu bài toán

Như đã giới thiệu ở trên. Mô hình GAN ban đầu tồn tại 2 bộ phận chính là Bộ tạo sinh (Generator) và Bộ phân biệt (Descrimiator) (Hình 1)

- Generator: Ở dạng cơ bản nhất, bộ tạo sinh nhận đầu vào là một tập hợp các vector nhiễu 'z' được khởi tạo ngẫu nhiên theo phân phối chuẩn, tập hợp các vector nhiễu z còn có tên gọi

khác là không gian tiềm ẩn (latent space). Ở một số mô hình GAN tiên tiến, cải tiến hơn, đầu vào của bộ tạo sinh có thể là một dữ liệu có sẵn như là bức ảnh, đoạn văn hoặc âm thanh được dùng để tạo ra dữ liệu mới cùng tính chất với dữ liệu đầu vào. Với mục đích làm quen với nắm rõ được mô hình GANs là gì chúng tôi sẽ nói chi tiết với đầu vào giả sử là không gian tiềm ẩn trước rồi mới đi nói về các mô hình có đầu vào được liệt kê như trên

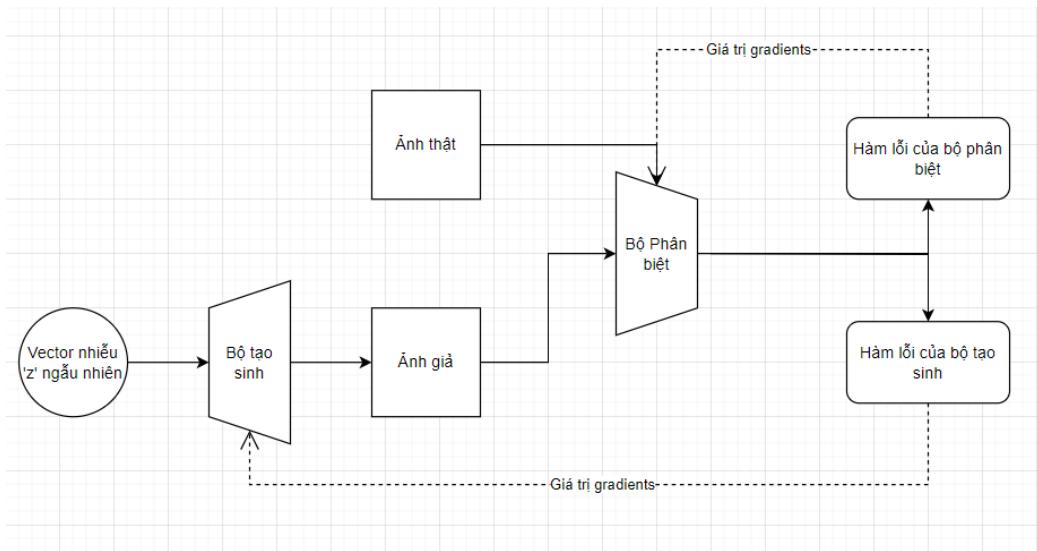
- Discriminator: Bộ phân biệt trong GANs đơn giản là một thuật toán phân loại. Bộ phân biệt này nhận dữ liệu từ Generator và dữ liệu thật, sau đó cố gắng phân biệt giữa chúng. Nhiệm vụ của Discriminator là phân biệt dữ liệu được tạo ra từ Generator và dữ liệu thật.

Quá trình huấn luyện GANs diễn ra thông qua việc cải thiện cùng một lúc cả Generator và Discriminator. Generator cố gắng tạo ra dữ liệu sao cho có thể đánh lừa Discriminator, trong khi Discriminator cố gắng phân biệt đúng giữa dữ liệu giả tạo và dữ liệu thật. Quá trình này tiếp tục cho đến khi Generator tạo ra dữ liệu không thể phân biệt được với dữ liệu thật.

2.1 Các bài toán thành phần của GANs

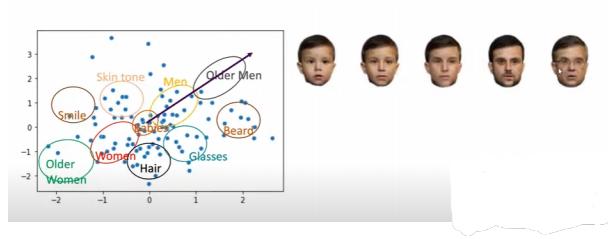
2.1.1 Mô hình tạo sinh Generator

Mục tiêu của mô hình sinh là tạo ra hoặc sản xuất ra các mẫu mới và tổng hợp của một đầu vào nhất định, có thể là một tập hợp ngẫu nhiên các giá trị hoặc nhiễu. Ví dụ, nếu bạn huấn luyện nó với lớp của một con mèo, mô hình sinh sẽ thực hiện một loạt các tính toán, sau đó tạo ra một hình ảnh của một con mèo, mặc dù không thật nhưng trông rất giống thật. Để có thể đánh lừa được Discriminator thì đòi hỏi mô hình sinh ra output phải thực sự tốt. Do đó chất lượng ảnh phải càng như thật càng tốt. Lý tưởng nhất, đầu ra sẽ không phải là con mèo giống nhau trong mỗi lần chạy, và để đảm bảo rằng mô hình sinh tạo ra các mẫu khác nhau mỗi lần, đầu vào sẽ là một tập hợp ngẫu nhiên các giá trị, được gọi là vector nhiễu. Xem xét thực tế rằng nó đang 'cạnh tranh' với bộ phân biệt, mô hình sinh cố gắng hết sức để tạo ra một hình ảnh giả mới với hy vọng rằng bộ phân biệt sẽ coi hình ảnh đó là thật. Mô hình sinh về bản chất là một mô hình nhận đầu vào là một tập hợp các vec tơ nhiễu trong không gian tiềm ẩn được khởi tạo ngẫu nhiên theo phân phối Gaussian(2). Không gian tiềm ẩn có thể được hiểu là mỗi điểm trong không gian đó có thể biểu diễn tiềm ẩn của một mẫu dữ liệu nào đó chẳng hạn như râu, tóc hoặc là màu da. Một không gian tiềm ẩn tốt sẽ có những



Hình 1: kiến trúc mạng GAN cơ bản

tính chất mà các vectơ đầu vào khác nhau tạo ra các mẫu dữ liệu khác nhau và đa dạng. Điều này cho phép mô hình GAN tạo ra các mẫu dữ liệu mới một cách sáng tạo và đa dạng dựa trên các biểu diễn trừu tượng trong không gian tiềm ẩn. Hình 2 mô tả không gian tiềm ẩn, nếu không gian chứa những điểm biểu diễn tiềm ẩn về phụ nữ, tóc, râu thì khi ta lấy một vector ngẫu nhiên chứa những điểm dữ liệu liên quan thì mô hình tạo sinh sau quá trình huấn luyện sẽ tạo ra ảnh có liên quan. Từ vector đầu vào được lấy trong không gian tiềm ẩn, mô hình generator là một mạng học sâu có tác dụng biến đổi ra bức ảnh giả ở output. Bức ảnh giả này sẽ được sử dụng làm đầu vào cho kiến trúc Discriminator.



2.1.2 Mô hình phân biệt Discriminator

Mô hình phân biệt được sử dụng để phân biệt xem ảnh đầu vào là thật hay giả. Khi đưa vào ảnh, nếu nó được lấy từ tập huấn luyện thì nhãn của mô hình sẽ là "thật", và nếu nó được lấy từ kết quả của mô hình Generator thì nhãn sẽ là "giả". Cơ bản, đây là một bài toán phân loại nhị phân, và chúng ta sử dụng hàm sigmoid để tính toán phân phối xác suất cho đầu ra của mô hình phân biệt.

3 Kiến trúc của GANs

3.1 Kiến trúc cơ bản

Kiến trúc cơ bản của GANs đã được mô tả qua hình 1 kết hợp từ hai model là Bộ tạo sinh và Bộ phân biệt. Hai model này có vai trò như là cảnh sát với tội phạm trong trò chơi minimax: trò chơi khi mà tội phạm là G (Bộ tạo sinh) sẽ tạo ra tiền giả để cảnh sát là D (Bộ phân biệt) sẽ cố gắng phân biệt xem đâu là tiền giả, đâu là tiền thật. Cảnh sát D càng phân biệt được tiền giả thì tội phạm G sẽ dựa vào feedback từ D để cải thiện chất lượng tiền giả của mình, cố gắng để cho cảnh sát D không thể phân biệt được đâu là tiền thật, đâu ra tiền giả, về thuật toán theo lưu đồ của hình 1 sẽ như sau:

- Từ một nhiễu z ngẫu nhiên, G sẽ sinh ra ảnh giả $G(z)$ có kích thước như ảnh thật. Tại lần tạo sinh đầu tiên thì $G(z)$ hoàn toàn là ảnh nhiễu và không chứa thông tin gì liên quan đến ảnh thật.
- Ảnh thật và ảnh giả $G(z)$ sẽ cùng đưa đến bộ phân biệt kèm nhãn để nhận biết đúng sai, từ đây sẽ cố gắng huấn luyện Bộ phân biệt để học được khả năng phân biệt được ảnh giả và ảnh thật.
- Sau khi đưa ảnh giả và ảnh thật, bộ phân biệt sẽ tính toán độ lỗi bằng cách phân loại.

- Lan truyền ngược qua cả mạng phân biệt và bộ tạo để thu được các độ dốc.
- Sử dụng các độ dốc để chỉ thay đổi các trọng số của bộ tạo.
- Quá trình trên sẽ lặp lại cho đến khi nào Bộ phân biệt không phân biệt được đâu là ảnh giả, đâu là ảnh thật, khi đó quá trình sẽ dừng lại

3.2 Loss Function

Bằng cách đào tạo song song hình ảnh trong thực tế và hình ảnh được tạo trong bộ tạo sinh, GANs xây dựng bộ phân biệt để điều chỉnh cho bộ tạo sinh tạo ra các vector đặc trưng sao cho giống với hình ảnh trong thực tế nhất có thể. Sau đó, bộ phân biệt sẽ phản hồi về cho trình tạo biết hình ảnh mà trình tạo đã tạo ra đã đáp ứng được yêu cầu là hình ảnh được tạo đã giống với hình ảnh trong thực tế nhất có thể hay chưa?

Cách để bộ phân biệt phản hồi cho bộ tạo sinh biết được hình ảnh được tạo ra bởi bộ tạo sinh đã đạt yêu cầu hay chưa bằng cách so sánh hình ảnh thực và hình ảnh tạo ra có tương đồng nhau hay không? Tức là bộ phân biệt nó phải nhận biết được xem đâu là bức ảnh thực và đâu là bức ảnh giả được tạo ra bởi bộ tạo sinh. Đầu ra của bộ phân biệt là giá trị $D(x)$ là xác suất thể hiện ảnh đầu vào X là ảnh thực trong thực tế. Chúng ta sẽ đào tạo bộ phân biệt này như giống như một trình phân loại trong Deep Learning thông thường. Nếu đầu vào

là một hình ảnh thực thì chúng ta mong muốn đào tạo sao cho đầu ra $D(x) = 1$, còn trong trường hợp đó là hình ảnh giả do bộ tạo sinh tạo ra thì $D(x) = 0$. Thông qua quá trình đào tạo này, thì bộ phân biệt có thể xác định được sự đóng góp của các đặc trưng z được tạo ra trong hình ảnh X đầu vào có gần giống với hình ảnh trong thế giới thực hay chưa. Với mục đích ban đầu của chúng ta là mong muốn bộ tạo sinh tạo ra hình ảnh chân thực nhất, tức là ta cần phải làm sao để đánh lừa bộ phân biệt rằng hình ảnh đấy là hình ảnh thực, tức là $D(x) = 1$, nếu làm được như vậy tức là bộ tạo sinh của chúng ta có khả năng tạo được hình ảnh chân thực nhất có thể/

Vì vậy, chúng ta sẽ phải đào tạo trình tạo bằng cách lan truyền ngược các giá trị mục tiêu này quay trở lại trình tạo, tức là chúng ta huấn luyện trình tạo để tạo ra hình ảnh theo hướng mà trình so sánh nghĩ là hình ảnh được tạo là có khả năng có thật.

Tiếp điền ta sẽ đi vào kỹ thuật lan truyền ngược. Mục tiêu của chúng ta đối với bộ phân biệt là tối đa hóa cơ hội nhận ra hình ảnh thật là hình ảnh thật và được hình ảnh được tạo ra bởi bộ tạo sinh là hình giả. Cách để tối ưu điều này là sử dụng cross-entropy được ứng dụng hầu hết trong các ứng dụng Deep Learning thông thường là $p \log(q)$. Đối với hình ảnh thực thì $p = 1$ và với hình ảnh giả được tạo ra thì sẽ đảo ngược nhau ($p = -1$). Vì vậy hàm mục tiêu của hàm D sẽ là

Discrimination Loss:

$$\max_D L(D) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (1)$$

Đối với bộ tạo sinh, mục tiêu của chúng ta mong muốn bộ tạo sinh tạo ra các hình ảnh chân thực nhất, tức là giá trị $D(x)$ đầu ra với đầu vào là hình ảnh được tạo x sẽ gần với giá trị 1 nhất có thể, tức là hàm mục tiêu của hàm G sẽ là như sau :

Generator Loss

$$\min_G L(G) = \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (2)$$

Và đối với bộ phân biệt, vì mục tiêu là phân biệt được ảnh giả, ảnh thực nên nó sẽ cố gắng làm sao cho $D(x)$ càng nhỏ càng tốt đối với đầu vào là ảnh giả x . Việc này giống như một trò chơi mini max, trong đó, ta muốn giá trị D của trình tạo G càng lớn càng tốt, trong khi lại muốn tối thiểu giá trị tương ứng D của trình so sánh.

Total Loss

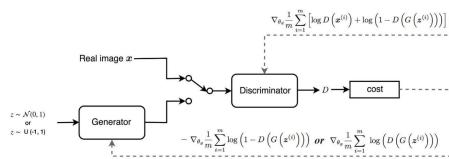
$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (3)$$

Trong đó:

- $p_{\text{data}}(x)$ là phân phối của dữ liệu thực.
- $p_z(z)$ là phân phối của dữ liệu nhiễu.

- $D(x)$ là xác suất mà một mẫu dữ liệu thực x được xem xét là thật bởi bộ phân biệt (discriminator).
- $G(z)$ là một mẫu dữ liệu được tạo ra bởi mô hình sinh (generator) từ dữ liệu nhiễu z .

Khi xác định được hàm mục tiêu, mô hình sẽ học được cách giảm độ dốc xen kẽ. Chúng ta cập nhật các tham số của trình tạo và thực hiện một bước lặp duy nhất của gradient descent trên trình so sánh bằng cách sử dụng hình ảnh thực và hình ảnh được tạo. Sau đó, chúng ta lại quay về trình tạo. Cập nhật các tham số của trình so sánh và huân luyện trình tạo cho một lần lặp khác. Chúng tôi đào tạo cả hai mạng theo các bước xen kẽ cho đến khi trình tạo tạo ra hình ảnh chất lượng tốt. Sau đây tóm tắt luồng dữ liệu và gradient được sử dụng cho quá trình lan truyền ngược :



Hình 2: Chú thích của ảnh

4 Các thành phần dữ liệu của bài toán GAN

4.1 Cơ sở dữ liệu

Cơ sở dữ liệu trong bài toán GANs là tập hợp các hình ảnh gồm nhiều

đối tượng khác nhau thuộc nhiều lĩnh vực khác nhau như là ảnh con người, động vật, ảnh phong cảnh ... hoặc là các đoạn văn, chữ viết, bài nhạc tùy thuộc vào ứng dụng GANs vào những mục đích gì để từ đó xây dựng cơ sở dữ liệu tương ứng.

Xây dựng dữ liệu là một trong những thành phần quan trọng trong công việc giải quyết bài toán GANs, việc này giúp đảm bảo hệ thống có thể hoạt động một cách trơn tru, hiệu quả và chính xác nhất có thể.

Xây dựng cơ sở dữ liệu hoàn chỉnh cần phải trải các bước bao gồm: Thu nhập dữ liệu có sẵn ở trên mạng được cho phép hoặc các nguồn dữ liệu nội bộ hoặc có thể tự thu thập thủ công; Trích xuất đặc trưng của dữ liệu thông qua các phần mềm, chương trình và lưu trữ dưới dạng vector số học; Gán nhãn và phân loại dữ liệu; Lưu trữ và bảo quản dữ liệu và cuối cùng là kiểm tra và đánh giá xem cơ sở dữ liệu có phù hợp với hệ thống hay không nhằm hoàn thiện cơ sở dữ liệu tốt hơn

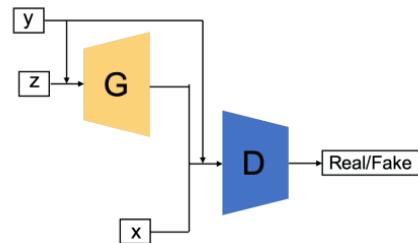
4.2 Input

Dữ liệu đầu vào của bài toán GANs thường có nhiều dạng tùy thuộc vào việc mình ứng dụng GANs vào mục đích gì, thông thường sẽ là hình ảnh, văn bản hoặc âm thanh hoặc là cả hình ảnh và văn bản kèm theo

4.3 Output

Dữ liệu đầu ra của bài toán GANs có cơ sở dữ liệu tương đồng với dữ liệu

đầu vào và tùy thuộc vào mục đích và nhu cầu của bài toán, nếu đầu vào là tập hình ảnh thì đầu ra là tập hình ảnh được phát sinh có tính chất tương đồng nhưng được cải tiến theo yêu cầu của bài toán



Hình 3: CGAN Architecture

5 Những phương pháp được ứng dụng trong bài toán GANs

5.1 cGAN

Conditional GAN hay cGAN là một mô hình phát triển từ DCGAN. Trong bản gốc của Generative Adversarial Networks (GANs), mô hình học sâu của nó thường được xây dựng dựa trên Neural Networks. Mô hình GAN gốc bao gồm hai mạng chính là Generator và Discriminator sử dụng Neural Networks để học biểu diễn dữ liệu và thực hiện các nhiệm vụ tạo dữ liệu mới và phân biệt dữ liệu. Do đó thay vì sử dụng Neural Networks(NN) thì DCGAN đã sử dụng Convolutional Neural Networks (CNNs) để xây dựng 2 mạng Generator (G) và Discriminator (D). CNNs tốt hơn hiệu quả và nhanh hơn trong các tác vụ xử lý ảnh. Tuy nhiên DCGAN lại không thể phân loại ảnh được sẽ được tạo sinh từ các "category" từ bộ dữ liệu, nên cGAN đã ra đời. Conditional GAN tập trung vào khả năng kiểm soát generator sinh ra ảnh theo một "category" cụ thể.

5.1.1 Mô hình CGAN

Bên cạnh noise vector z như GAN thông thường, ta sẽ thêm vào nhãn y (label), mỗi thể loại trong dữ liệu Fashion-MNIST(3) sẽ được encode về 1 label. Mình mong muốn là z với nhãn y nào thì sẽ cho ra dữ liệu tương ứng thể loại đấy.

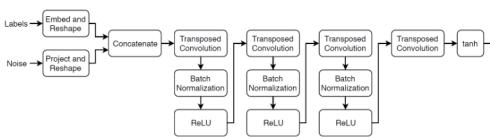
- Mô hình Generator cGAN

Input là vector nhiễu z, thông thường được đưa qua một lớp dense để chuyển đổi về kích thước $7 \times 7 \times 128$. (y1)

Input y là một số hoặc một loại thông tin cụ thể khác, được đưa qua một lớp embedding để ánh xạ thành một vector 50×1 , sau đó thông qua một lớp dense với đầu ra 49 nút, cuối cùng được reshape về dạng 3D tensor kích thước $7 \times 7 \times 1$. (y2)

Hai đầu ra từ bước trước (y1 và y2) được xếp chồng lên nhau để tạo thành một tensor 3D kích thước $7 \times 7 \times 129$. Tensor từ bước trước được đưa qua các lớp transposed convolution để tăng kích thước lên 14×14 và 28×28 , cuối cùng cho ra output

28x28x1, tức là hình ảnh được tạo ra (ảnh fake).

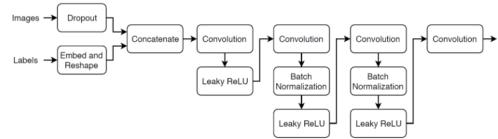


Hình 4: CGAN Generator Architecture

- Mô hình Discriminator

Input y là một số hoặc một loại thông tin cụ thể khác (giống với y trong bộ G), được đưa qua một lớp embedding để ảnh xạ thành một vector 50x1, sau đó thông qua một lớp dense với đầu ra 28x28 nút, cuối cùng được reshape về dạng 3D tensor kích thước 28x28x1. Hai đầu ra từ bước trước (y_1) và ảnh input được xếp chồng lên nhau để tạo thành một tensor 3D kích thước 28x28x2.

Tensor từ bước trước được đưa qua các lớp convolution với stride = 2 để giảm kích thước ảnh từ 28x28 xuống 14x14 và sau đó xuống 7x7. Trong quá trình giảm kích thước, độ sâu của tensor tăng dần. Cuối cùng, tensor có kích thước 2x2x512 được reshape về dạng vector 2048 và đi qua một lớp fully connected để chuyển từ 2048 chiều về 1 chiều, đại diện cho xác suất mẫu đến từ dữ liệu huấn luyện hoặc từ Generator.



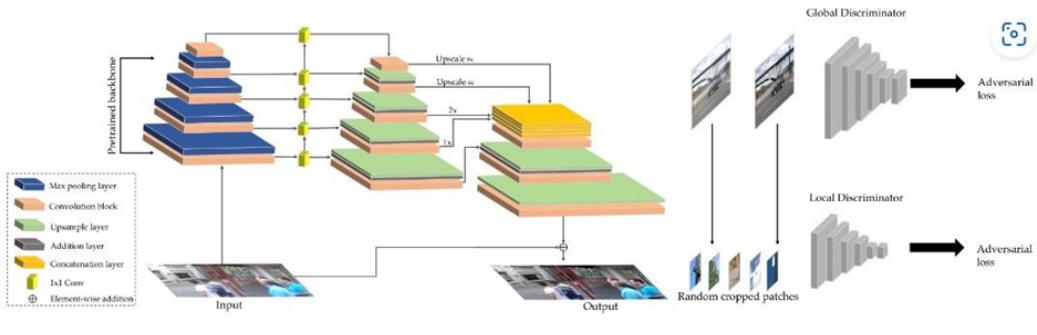
Hình 5: CGAN Discriminator Architecture

5.2 Phương pháp Deblur GAN-v2: Deblurring trong bài toán khử mờ hình ảnh

Tổng quan kiến trúc

5.2.1 DeblurGAN

DeblurGAN sử dụng mô hình kết hợp các biến thể của GAN bao gồm conditional Generative Adversarial Network (cGAN) (4) và Wasserstein GAN (wGAN) (5), để tạo thành conditional Wasserstein GAN (cWGAN). cGAN được sử dụng để tạo ra ảnh rõ nét dựa trên điều kiện của ảnh ban đầu và các thông tin cung cấp (ví dụ: ảnh mờ). wGAN được sử dụng để cải thiện hiệu suất huấn luyện và ổn định hơn trong quá trình huấn luyện bằng cách sử dụng hàm mất mát Wasserstein, thay vì sử dụng hàm mất mát binary cross-entropy như trong GAN thông thường, điều này có nghĩa là trong hàm lỗi (loss function) của 2 mạng học thì khả năng phân phối thật và phân phối sinh . Kết hợp giữa cGAN và wGAN tạo ra cWGAN, giúp DeblurGAN đạt được kết quả tốt hơn và nhanh hơn trong việc tạo ra ảnh mới.



Hình 6: kiến trúc của DeblurGANv2

5.2.2 DeblurGANv2

A) RCGAN

Có sự khác biệt trong mô hình của DeblurGANv2 (6), đó là nó sử dụng mô hình Relativistic Conditional GAN (RCGAN), là kết hợp của RGAN + LSGAN (7) (8), một loại GAN có điều kiện tương đối, để cải thiện hiệu suất và chất lượng của quá trình deblurring. Điều này giúp tăng tính ổn định và hiệu quả của mô hình so với DeblurGAN. Mặc dù quá trình đào tạo của WGAN tốt hơn với loss function của nó (giúp ảnh thật hơn) nhưng trong trường hợp ảnh bị mờ, hay nói cách khác là mất quá nhiều ngữ nghĩa thì việc tìm kiếm sự tương đồng cho bộ đào tạo rất mất thời gian. Do đó RGAN được xem như biện pháp thay thế khi nó giúp tạo ra ảnh thật hơn cả thật làm cho quá trình huấn luyện nhanh hơn.

B) Double-scale Discriminator

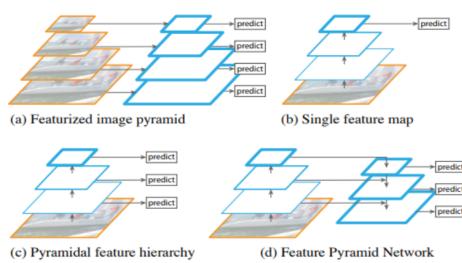
DeblurGAN-v2 áp dụng double-scale discriminator để đánh giá

chất lượng của ảnh tái tạo ở cả hai tỷ lệ, giúp cải thiện khả năng đánh giá và cải thiện chất lượng của ảnh tái tạo so với DeblurGAN. Double-scale discriminator sử dụng hai bộ lọc quét khác nhau (ở hai tỷ lệ hoặc scale khác nhau) để phân tích đặc trưng của ảnh cùng lúc. Bằng cách này, nó có thể xem xét cả chi tiết cục bộ và tổng thể của ảnh. Ví dụ, một bộ lọc có thể tập trung vào các đặc trưng cục bộ, trong khi bộ lọc khác tập trung vào các đặc trưng tổng thể của ảnh.

C) Feature Pyramid Network (FPN)

DeblurGAN-v2 sử dụng FPN (9) trong generator, một phương pháp mới được giới thiệu lần đầu trong lĩnh vực deblurring, để xử lý đa tỷ lệ và cải thiện hiệu suất mà không tăng đáng kể về độ phức tạp. Điều này giúp DeblurGAN-v2 linh hoạt hơn và có khả năng làm việc tốt với nhiều backbone khác nhau. FPN phân thành nhiều tầng với các

độ phân giải khác nhau giúp giữ được nhiều nghĩa của đặc trưng ảnh nhất có thể. Các đặc trưng ở các tỷ lệ cao hơn thường chứa thông tin tổng thể và phân loại đối tượng, trong khi các đặc trưng ở các tỷ lệ thấp hơn thường chứa thông tin chi tiết và định vị đối tượng.



Hình 7: Some pyramids model Architecture

D) BackBone

DeblurGAN-v2 linh hoạt trong việc lựa chọn backbones để ảnh hưởng đến chất lượng và hiệu suất của quá trình khử mờ ảnh. Có nghĩa là nó cho phép sử dụng các loại backbone khác nhau tùy trường hợp để đạt hiệu quả tốt nhất. Sự chọn backbones ảnh hưởng đến chất lượng và hiệu suất của quá trình khử mờ ảnh. Sử dụng Inception-ResNet-v2 (10) giúp đạt hiệu suất khử mờ tốt nhất, trong khi sử dụng MobileNet và MobileNet-DSC (11) giúp tăng hiệu suất và giảm kích thước mô hình, dẫn đến tốc độ xử lý nhanh hơn. Backbone sẽ được sử dụng chủ yếu trong quá

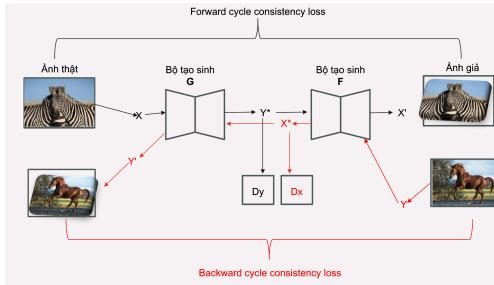
trình huấn luyện và quá trình khử mờ.

5.3 Cycle-Consistent Adversarial Networks trong bài toán Unpaired Image-to-Image Translation

5.3.1 Động lực

Thay vì các cặp input - output được gắn nhãn trong các bài toán dịch ảnh(image to image translation) trước đây sẽ tốn kém chi phí và không phù hợp với một số trường hợp nhất định ví dụ như để chuyển đổi một phong cảnh từ mùa hè sang mùa đông phải đợi đủ 2 điều kiện là ảnh cảnh đó lúc mùa hè và ảnh cảnh đó lúc mùa đông đề mà gán cặp dữ liệu, hay là khi chuyển đổi một con ngựa sang ngựa vằn, ta cần ảnh một con ngựa và ảnh của con ngựa đó mà có vằn để gán cặp dữ liệu là không khả thi, hơn nữa là với tác vụ chuyển đổi ảnh chụp từ máy ảnh sang ảnh phong cách nghệ thuật của các họa sĩ thì việc kiểm ra được một tấm vẽ về ảnh chụp từ máy ảnh có phong cách như các họa sĩ đó là khó để nhận định được và rất tốn kém. Nên ý tưởng là dùng đến dữ liệu unpaired tức là miền ảnh vào vào miền ảnh đích là hoàn toàn độc lập không được gắn nhãn cặp lại với nhau, đã được đề xuất là CycleGAN(12).

5.3.2 Kiến trúc



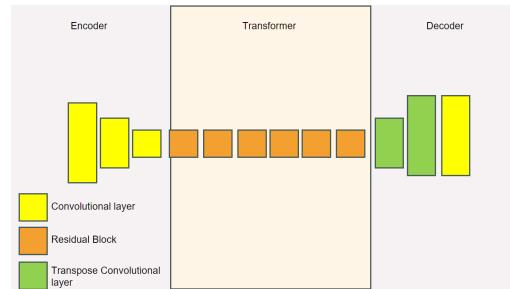
Hình 8: Kiến trúc

CycleGAN được thiết kế dựa trên GAN với một số tinh chỉnh cho phù hợp với bài toán như là sẽ có cả 2 bộ tạo sinh và 2 bộ phân biệt. Xem như ta đang giải bài toán chuyển đổi từ ảnh ngựa thường thành ngựa vằn thì với bộ tạo sinh G của cycleGAN sẽ được nhận ảnh từ một X(ngựa thường), thay vì vector trong latent space như các bộ mô hình G của GAN truyền thống khác. Bộ tạo sinh G này sẽ góp vai trò chuyển đổi từ miền X sang Y(ngựa vằn), bộ tạo sinh F sẽ làm việc ngược lại là từ Y về X. Khi đó việc chuyển đổi về ngược lại ảnh được đưa vào G sẽ gọi là forward cycleconsistency và ngược lại với khi ta đưa ảnh y vào F và tái tạo lại ảnh y có được backward cycleconsistency. Khi đó sẽ là bài toán cố gắng tái tạo lại ảnh sau khi được đưa qua một bộ tạo sinh và cố gắng phục hồi lại tấm ảnh đó là các Cycleconsistency loss đóng vai trò như làm một người giám sát cho mô hình cycleGAN khi thiếu vắng đi tập dữ liệu được gắn nhãn cặp. Về hai bộ phân biệt D sẽ có vai trò tương tự nhau và chúng như các

GAN truyền thống khác đã đề cập trước đó, ví dụ như với Dy sẽ phân biệt ra ảnh của con ngựa vằn được tạo ra bởi G ($G(X)$) và ảnh con ngựa vằn trong Y.

Bộ tạo sinh là một bộ mã hóa tự động lấy hình ảnh đầu vào, trích xuất các tính năng từ nó và tạo ra một hình ảnh khác với ba giai đoạn:

- Mã hóa (encoder)
- Chuyển đổi (transformer)
- Giải mã (decoder)



Hình 9: Bộ tạo sinh của CycleGAN

Giai đoạn mã hóa bao gồm ba lớp tích chập 2D, theo sau là lớp chuẩn hóa phiên bản và hàm kích hoạt (ReLU). Bộ mã hóa trích xuất các đặc điểm từ hình ảnh đầu vào bằng cách sử dụng phép tích chập, giảm mức biểu diễn xuống 25% kích thước hình ảnh đầu vào. Sau đó output được cho qua phần transformer với 6 residual block và cuối cùng được cho qua phần decoder để về ảnh có kích thước giống ban đầu. Bộ phân biệt dùng để phân biệt ảnh sinh ra bởi generator hay ảnh thật trong dataset. Ý tưởng dựa theo

patchGAN(13) sẽ thực hiện kiểm tra trên từng vùng ảnh nhỏ ($70*70$) trên ảnh ($224*224$) thay vì làm toàn bộ trên ảnh. Và thay vì trả về là ảnh là sai hay đúng thì sẽ là từng giá trị đúng sai của từng vùng, với cách này sẽ khiến cho mô hình dễ phát hiện các lỗi xảy ra sự sai lệch một cách hiệu quả hơn hết.

5.3.3 Loss function

Tương tự với GAN, CycleGAN trong quá trình tạo sinh đối nghịch Generator và Discriminator vẫn được áp dụng đến Adversarial loss nhưng chỉ thế thì chưa đủ tốt cho mô hình và không đảm bảo việc G sẽ tạo ra ảnh X đến được miền Y do thiếu sót việc giám sát với dữ liệu được gán nhãn ghép cặp.

Cycle consistency loss được xây dựng nên để cải thiện vấn đề đó, trong bài báo họ đặt yêu cầu bài toán khi dịch như sau: $G:X \rightarrow Y$ và từ $F:Y \rightarrow X$ phải là như nhau (cycle consistency). Áp dụng giả định đó bằng cách huấn luyện đồng thời cả ảnh xạ G và F, do khi train nếu chỉ có adversarial loss nhưng mô hình dùng unpaired data, generator sẽ có xu hướng tạo ra được output bất kỳ trong tập kết quả. Ví dụ với bài toán chuyển ngựa vằn thành ngựa thường như trên thì generator có thể biến con ngựa vằn thành 1 con ngựa thường rất đẹp nhưng lại không có đặc điểm nào liên quan tới con ngựa vằn ban đầu. Nên ý nghĩa của cycle-consistency loss là để đảm bảo khi chuyển từ a sang b được a' thì a' khi chuyển ngược lại phải là a.

Với Adversarial loss thì vẫn tương tự như các mô hình GAN truyền thống đã đề cập.

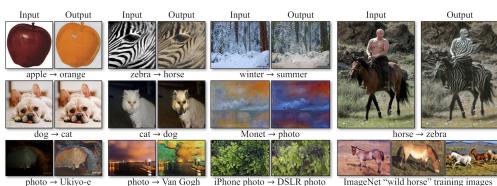
Theo nghiên cứu thì họ sẽ chọn một tham số vào để hàm mất mát cuối cùng như sau: $\text{Finalloss} = \text{Adversarialloss}(G) + \text{Adversarialloss}(F) + 10^8 \text{Cycle consistencyloss}(G,F)$. Việc Cycle loss được thêm tiền tố là 10 do trong quá trình huấn luyện họ đưa ra các con số và nhận thấy rằng hiệu quả khi tăng lên 10 lần, cũng như thấy được việc hàm mất mát để đảm bảo quá trình song ánh của G và F là cần thiết và ảnh hưởng đến quá trình học rất nhiều.

5.3.4 Thực nghiệm

Đối với các tác vụ không ảnh hưởng về mặt hình học thì khá thành công, như đổi quả táo sang quả cam, ngựa sang ngựa vằn, tranh phong cảnh sang tranh phong cách họa sĩ nổi tiếng,... Khi ảnh hưởng đến thay đổi hình học như là tác vụ chuyển đổi con mèo thành chó thì sẽ không hiệu quả. Chủ yếu là do pathGAN chưa hiệu quả với path size, về việc dữ liệu khi huấn luyện cũng rất quan trọng do là trên cơ sở là dữ liệu ở CycleGAN thiếu đi sự giám sát khi học bởi cặp nhãn gắn sẵn ở hai miền, thay vào đó là học dựa trên quá trình song ánh giữa hai miền, và với điều đó sẽ xảy ra vấn đề như khi học một tập ảnh toàn là ngựa thường hoang (tức là không có người cưỡi) khi áp tác vụ chuyển đổi sang ngựa sang ngựa vằn nhưng với đầu vào là một người cưỡi ngựa sẽ

cho ta được kết quả là người lẩn ngựa đều có sọc vẫn do tập dữ liệu miền nguồn không được đa dạng khiến mô hình học sẽ huấn luyện một cách thô sơ và đơn giản khi gặp được đầu vào thử thách sẽ không xử lý được.

Là một mô hình có tiềm năng dựa trên việc nguồn dữ liệu chưa gắn nhãn là vô cùng dồi dào trong thực tế, nên các đánh đổi về hiệu quả là đáng được cân nhắc. Các tác giả nhận thấy bộ tạo sinh của họ cũng như việc dùng patchGAN không đạt hiệu quả trong các tác vụ biến đổi hình học.



Hình 10: Một vài lỗi thông thường của cycleGAN từ bài báo của tác giả

6 A Style-Based Generator Architecture for Generative Adversarial Networks trong bài toán tạo sinh ảnh

6.1 Tổng quan

A Style-Based Generator Architecture for Generative Adversarial Networks hay có tên gọi rút gọn hơn là StyleGAN (14) được giới thiệu bởi NVIDIA vào năm 2018. Được truyền

cảm hứng từ các tài liệu về chuyển đổi phong cách, StyleGAN làm lại kiến trúc của bộ sinh ảnh với thiết kế hoàn toàn mới để mở ra những cách tiếp cận mới để kiểm soát quá trình tổng hợp ảnh. Bộ sinh ảnh của StyleGAN bắt đầu từ một đầu vào hằng số đã học được và điều chỉnh "phong cách" của ảnh tại mỗi tầng convolution dựa trên mã ẩn, do đó điều chỉnh trực tiếp sức mạnh của các đặc điểm hình ảnh ở các tỷ lệ khác nhau. Kết hợp với nhiều được tiêm trực tiếp vào mạng, các cải tiến cho phép ta có thể điều chỉnh được mức độ chi tiết của khuôn mặt trong một bức ảnh, từ các chi tiết nhỏ, đơn giản ví dụ là màu da, màu mắt, các phụ kiện cho đến các chi tiết lớn hơn như là độ dày và kiểu của tóc, hình dáng của mắt, tai, khuôn mặt. Việc lấy cảm hứng từ StyleGAN đã tạo ra các tiến bộ đáng kể trong lĩnh vực sinh ảnh tự động và điều khiển phong cách hình ảnh. StyleGAN đã mở ra cánh cửa cho việc tạo ra các mô hình mới như StyleGAN2(15) và JojoGAN (16), mỗi một trong số chúng mang lại những cải tiến và ứng dụng độc đáo.

Trước khi đi sâu và kiến trúc của styleGAN thì trước tiên chúng ta cần hiểu về tầm quan trọng của không gian tiềm ẩn (latent space) và lý do tại sao nó lại được coi là cốt lõi của bộ tạo sinh nói chung và GAN nói riêng. Không gian tiềm ẩn chúng ta có thể coi nó là một không gian trong đó mỗi hình ảnh được biểu thị bằng một vectơ có N chiều. Mục tiêu là để có được thông tin duy nhất từ mỗi chiều. Bằng cách này, không gian tiềm ẩn

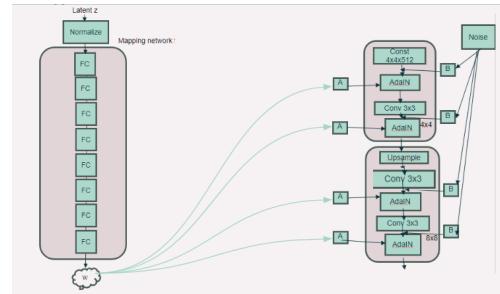
sẽ được phân tách và trình tạo sẽ có thể thực hiện bất kỳ chỉnh sửa mong muốn nào trên hình ảnh. Để hiểu rõ hơn về mối quan hệ giữa việc chỉnh sửa hình ảnh và sự tách rời không gian tiềm ẩn, hãy tưởng tượng rằng bạn muốn hình dung con mèo của bạn sẽ trông như thế nào nếu nó có 'lông dài'. Vì vậy, bạn chỉ muốn thay đổi kích thước chứa thông tin về độ dài của tóc. Trong trường hợp không gian tiềm ẩn bị vướng víu, sự thay đổi của chiều này có thể biến con mèo của bạn thành một con chó bông nếu loại động vật và độ dài lông của nó được mã hóa trong cùng một chiều. Đó chính là vấn đề vướng víu mà không gian tiềm ẩn gặp phải, việc thay đổi một thuộc tính có thể dễ dàng dẫn đến những thay đổi không mong muốn cùng với các thuộc tính khác. Do đó, mục tiêu chính của kiến trúc GAN là thu được một không gian tiềm ẩn có cấu trúc rõ ràng, có khả năng kiểm soát, tạo hình ảnh thực tế, thao tác ngữ nghĩa, chỉnh sửa cục bộ, v.v.

6.2 Kiến trúc

StyleGan được thiết kế dựa trên GAN và vì tính chất bài toán được StyleGAN tập trung nghiên cứu là bài toán tạo sinh ảnh nên công việc chủ yếu ở đây xây dựng mô hình tạo sinh mà không đụng đến mô hình phân biệt hoặc hàm lỗi. Nên về cơ bản, mô hình phân biệt của GAN vs StyleGAN là như nhau.

Về kiến trúc của mô hình tạo sinh của StyleGAN khác biệt khá nhiều so với mô hình tạo sinh gốc của GAN cũ

thể qua hình dưới đây:



Hình 11: Kiến trúc bộ tạo sinh style-GAN

Theo truyền thống, mã tiềm ẩn được cung cấp cho bộ tạo thông qua lớp đầu vào, tức là lớp đầu tiên của mạng chuyển tiếp. Tác giả bắt đầu từ thiết kế này bằng cách bỏ qua hoàn toàn lớp đầu vào và thay vào đó bắt đầu từ một hằng số đã học. Cho một mã tiềm ẩn z trong không gian tiềm ẩn đầu vào Z , mạng ánh xạ phi tuyến tính $f: Z \rightarrow W$ trước tiên tạo ra $w \in W$ (Hình 1b, bên trái). Để đơn giản, tác giả đặt chiều của Tác giả bắt đầu từ thiết kế này bằng cách bỏ qua hoàn toàn lớp đầu vào và thay vào đó bắt đầu từ một hằng số đã học (Hình 1b, bên phải). Cho một mã tiềm ẩn z trong không gian tiềm ẩn đầu vào Z , mạng ánh xạ phi tuyến tính $f: Z \rightarrow W$ trước tiên tạo ra $w \in W$ (Hình 1b, bên trái). Để đơn giản, tác giả đặt chiều của cả hai không gian đến 512 và ánh xạ f được triển khai bằng MLP 8 lớp, quyết định mà tác giả sẽ phân tích sau. Sau đó, các phép biến đổi affine đã học sẽ chuyên môn hóa w thành các kiểu y (y, ys) điều khiển

các hoạt động chuẩn hóa phiên bản thích ứng (AdaIN) [26, 16, 20, 15] sau mỗi lớp tích chập của mạng tổng hợp g. Hoạt động AdaIN được định nghĩa là

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

Hình 12: Enter Caption

Công thức với đầu vào là tensor \mathbf{x}_i và vector phong cách \mathbf{y} , công thức sẽ tính giá trị trung bình và độ lệch chuẩn của \mathbf{x}_i , sau đó ta sẽ chuẩn hóa \mathbf{x}_i và áp vào vector phong cách \mathbf{y} lên \mathbf{x}_i bằng cách thay đổi giá trị trung bình và độ lệch chuẩn. So sánh cách tiếp cận của tác giả với việc chuyển kiểu, tác giả tính toán kiểu \mathbf{y} bất biến về mặt không gian từ vectơ \mathbf{w} thay vì một hình ảnh mẫu. Tác giả chọn sử dụng lại từ "kiểu" cho \mathbf{y} vì các kiến trúc mạng tương tự đã được sử dụng để chuyển kiểu tiếp nối, dịch từ ảnh sang ảnh không giám sát và kết hợp miền. So với các phép biến đổi đặc trưng tổng quát hơn, AdaIN đặc biệt phù hợp với mục đích của tác giả do tính hiệu quả và biểu diễn nhỏ gọn của nó. Cuối cùng, tác giả cung cấp cho trình tạo của mình một phương tiện trực tiếp để tạo ra chi tiết ngẫu nhiên bằng cách đưa ra các đầu vào nhiều rõ ràng. Đây là những hình ảnh một kênh bao gồm nhiều Gaussian không liên quan đến nhau và tác giả cung cấp hình ảnh nhiều chuyên dụng cho từng lớp của mạng tổng hợp. Hình ảnh nhiều được phát triển tất cả các bản đồ đối tượng bằng

cách sử dụng các hệ số tỷ lệ cho mỗi đối tượng đã học và sau đó được thêm vào đầu ra của tích chập tương ứng Thông thường đối với mô hình tạo sinh của GAN, đầu vào sẽ là vector đặc trưng \mathbf{z} (thường sẽ được khởi tạo ngẫu nhiên theo phân phối chuẩn) và sẽ được đưa vào mạng tạo sinh để tạo sinh ra ảnh giả, Đối với Style-GAN, vector đặc trưng \mathbf{z} trước khi được đưa vào mạng tạo sinh thì phải qua quá trình ánh xạ (mapping) là một mạng Multilayer Perceptron gồm 8 lớp để mã hóa vector \mathbf{z} thành vector phong cách \mathbf{w} với cùng số chiều. Vector \mathbf{w} được sử dụng để điều khiển phong cách của hình ảnh thông qua lớp AdaIN(Adaptive Instance Normalization). Một phép biến đổi affine (được học thông qua một lớp fully connected - được ký hiệu là \mathbf{A} trong hình dưới) được áp dụng lên \mathbf{w} trước khi đưa vào mạng generator. Ngoài ra còn có noise được thêm vào từ \mathbf{B} đóng vai trò quan trọng trong việc tạo ra động lượng ngẫu nhiên và tính ngẫu nhiên trong quá trình sinh ảnh. Noise thường là dưới dạng các vector ngẫu nhiên, được kết hợp với các biểu diễn của các lớp trong quá trình lan truyền tiến. Việc này giúp làm cho mỗi lần sinh ảnh sẽ có một lượng ngẫu nhiên khác nhau được đưa vào, dẫn đến việc tạo ra các ảnh mới và đa dạng. Mỗi lớp trong mô hình genertor Stylgan sẽ kiểm soát các đặc trưng chi tiết khác nhau trong ảnh, Các lớp đầu ($4x4, 8x8, 16x16..$) sẽ kiểm soát các đặc trưng thô lớn như là khuôn mặt, kiểu tóc và các lớp cuối cùng có độ phân giải $512x512, 1024x1024$)

sẽ kiểm soát các đặc trưng kết cấu khuôn mặt như màu da, màu tóc, màu mắt...

Nhưng tại sao họ lại thêm một không gian trung gian? Chính khả năng giải rối tốt hơn của không gian W đã khiến nó trở thành đặc điểm chính trong kiến trúc này. Điều đó có nghĩa là 512 chiều của một vectơ w đã cho chứa từng thông tin duy nhất về hình ảnh. Bạn có thể tự hỏi làm sao chúng ta biết liệu không gian W có thực sự ít vướng víu hơn không gian Z hay không. Để trả lời câu hỏi này, các tác giả đề xuất hai thước đo mới để định lượng mức độ tháo gỡ:

- Perceptual path length : để đo mức độ thay đổi được thực hiện trên ảnh khi thực hiện phép nội suy. Những thay đổi mượt mà cho kết quả tốt hơn. Đó là lý do tại sao chúng ta phải giảm thiểu số liệu này.
- Khả năng phân tách tuyến tính : để tính toán mức độ chúng ta có thể phân tách các điểm không gian tiềm ẩn tương ứng với hai lớp hình ảnh thông qua một siêu phẳng. Điều đó được thực hiện bằng cách tính toán entropy có điều kiện để có thể cho biết cần bao nhiêu thông tin để phân loại chính xác những điểm tiềm ẩn này.

6.3 Trộn phong cách của StyleGAN

Với cấu tạo của mạng generator của StyleGAN, các nhà nghiên cứu đã nghĩ ra ý tưởng nếu ta có một mô hình tạo sinh với đầu vào là vector latent w_1 được mã hóa từ vector latent z_1 và ta lấy một vector latent space w được mã hóa từ vector latent z_2 khác rồi đưa vào một lớp bất kì trong mô hình tạo sinh ban đầu thì kết quả ta thu được sẽ là gì? Ta sẽ thu được bức ảnh được tạo sinh có phong cách, vóc dáng nhìn na ná với một bức ảnh khác nếu ta dùng vector latent z_2 để tạo sinh bức ảnh đó. Đây là quá trình trộn phong cách từ 2 vector latent z khác nhau và nếu ta lấy hai vector đó để tạo sinh ra hai bức ảnh khác và so với bức ảnh được trộn từ hai vector latent z ta có thể thấy được ảnh đó là sự kết hợp tương quan giữa hai bức ảnh trên. Quá trình trộn phong cách của StyleGAN có thể điều chỉnh thông qua việc thêm vector w sau khi được mã hóa vào lớp nào của mô hình tạo sinh

Có nhiều khía cạnh trong chân dung con người có thể được coi là ngẫu nhiên, chẳng hạn như vị trí chính xác của các sợi tóc, râu, tàn nhang hoặc lỗ chân lông trên da. Bất kỳ điều nào trong số này có thể được chọn ngẫu nhiên mà không ảnh hưởng đến nhận thức của chúng ta về hình ảnh miễn là chúng tuân theo sự phân bố chính xác. Chúng ta hãy xem xét cách một máy phát điện truyền thống thực hiện biến đổi stochastic. Cho rằng đầu vào duy nhất của mạng

là thông qua lớp đầu vào, mạng cần phát minh ra cách tạo ra các số giả ngẫu nhiên thay đổi theo không gian từ các kích hoạt trước đó bất cứ khi nào chúng cần tính toán dung lượng mạng và che giấu tính chu kỳ của tín hiệu được tạo ra là rất khó và không phải lúc nào cũng thành công, bằng chứng là các mẫu lặp đi lặp lại thường thấy trong các hình ảnh được tạo ra. Kiến trúc của chúng tôi hoàn toàn tránh được những vấn đề này bằng cách thêm nhiễu trên mỗi pixel sau mỗi lần chập.

Chúng ta có thể thấy rằng nhiều chỉ ảnh hưởng đến các khía cạnh ngẫu nhiên, giữ nguyên thành phần tổng thể và các khía cạnh cấp cao như danh tính. Hình minh họa thêm tác dụng của việc áp dụng biến đổi ngẫu nhiên cho các tập hợp con khác nhau của các lớp. Vì những hiệu ứng này được thấy rõ nhất trong hoạt ảnh, vui lòng tham khảo video đi kèm để biết cách thay đổi đầu vào nhiễu của một lớp dẫn đến biến đổi ngẫu nhiên ở tỷ lệ phù hợp. Chúng tôi thấy thú vị khi hiệu ứng của tiếng ồn xuất hiện được định vị chặt chẽ trong mạng. Chúng tôi đưa ra giả thuyết rằng bất cứ lúc nào điểm trong trình tạo, có áp lực phải giới thiệu nội dung mới càng sớm càng tốt và cách dễ nhất cho mạng của chúng tôi để tạo ra sự thay đổi ngẫu nhiên là phải dựa vào nhiễu được cung cấp. Một tập hợp tiếng ồn mới có sẵn cho mỗi lớp và do đó có không có động lực để tạo ra các hiệu ứng ngẫu nhiên từ trước đó kích hoạt, dẫn đến hiệu quả cục bộ. Hai bộ ảnh được tạo ra từ các mã ẩn

tương ứng của chúng (nguồn A và B); phần còn lại của các hình ảnh được tạo ra bằng cách sao chép một phần cụ thể của các phong cách từ nguồn B và lấy phần còn lại từ nguồn A. Việc sao chép các phong cách tương ứng với độ phân giải không gian thô (4×4 đến 8×8) đem lại các khía cạnh cấp cao như tư thế, kiểu tóc tổng quát, hình dáng khuôn mặt và kính áp tròng từ nguồn B, trong khi tất cả các màu sắc (mắt, tóc, ánh sáng) và các đặc trưng tinh tế của khuôn mặt giống như nguồn A. Nếu thay vào đó chúng ta sao chép các phong cách ở độ phân giải trung bình (16×16 đến 32×32) từ B, chúng ta sẽ thừa kế các đặc trưng khuôn mặt với tỷ lệ nhỏ hơn, kiểu tóc, mắt mở/đóng từ B, trong khi tư thế, hình dáng khuôn mặt tổng quát và kính áp tròng từ A được bảo tồn. Cuối cùng, việc sao chép các phong cách tinh tế (64×64 đến 1024×1024) từ B chủ yếu đem lại khác biệt về màu xa, màu tóc .

6.4 Thủ thuật cắt ngắn trong StyleGAN

Các hình ảnh được trình bày kém trong tập dữ liệu thường rất khó được GAN tạo ra. Vì trình tạo không nhìn thấy một lượng đáng kể các hình ảnh này trong khi đào tạo nên nó không thể học cách tạo chúng một cách chính xác, điều này sẽ ảnh hưởng đến chất lượng của các hình ảnh được tạo ra. Để giải quyết vấn đề này, có một kỹ thuật gọi là “thủ thuật cắt ngắn” giúp tránh các vùng có mờ

độ xác suất thấp để cải thiện chất lượng hình ảnh được tạo ra. Nhưng vì chúng ta đang bỏ qua một phần của phân phối nên chúng ta sẽ có ít biến thể về kiểu dáng hơn. Kỹ thuật này được biết đến là một cách tốt để cải thiện hiệu suất của GAN và nó đã được áp dụng cho Z-space. StyleGAN cũng cung cấp khả năng thực hiện thủ thuật này trên W-space. Điều thú vị là thủ thuật cắt ngắn trong w-space cho phép chúng ta kiểm soát các kiểu. Như thể hiện trong hình dưới đây, khi chúng ta đưa tham số về 0, chúng ta sẽ thu được hình ảnh trung bình. Mặt khác, khi so sánh kết quả thu được với 1 và -1, chúng ta có thể thấy chúng tương ứng đối lập nhau (về tư thế, mái tóc, độ tuổi, giới tính..). Điều này một lần nữa nêu bật những điểm mạnh của không gian W.

7 Đánh giá khả năng các mô hình, thực nghiệm

7.1 DeBlurGAN

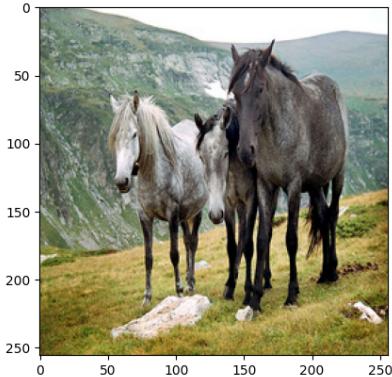
Để có cái nhìn tốt hơn về DeblurGAN nhóm thực hiện khảo sát đánh giá 100 cặp ảnh khác nhau của 2 bộ dataset là BlurDetection Dataset từ bài báo(17) và StereoBlur Dataset (18).

Bảng 1: So sánh kết quả khử mờ

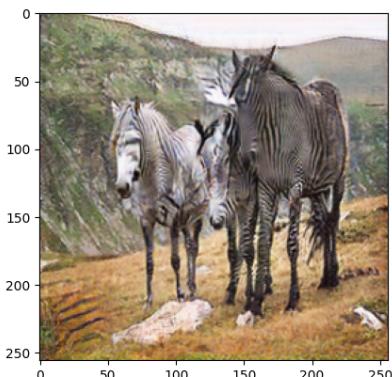
	Tập dữ liệu 1	Tập dữ liệu 2
PSNR	36.05	0.82 (24.3)
SSIM	0.96	0.87 (26.9)

7.2 CycleGAN

Với bài toán dịch ảnh với dữ liệu được gán cặp sẵn và ngược lại là cycleGAN trong bài toán dữ liệu giữa hai miền không được ghép cặp, với dữ liệu đầu vào và ra của cả hai đều là đầu vào có một vài đặc điểm được xem như là cùng một số đặc tính giữa hai miền. Nhưng với cycleGAN việc mapping từ một miền ảnh sang một miền ảnh khác nó không phải là điều dễ dàng, đây cũng là một thách thức của cycleGAN với việc dữ liệu không được ghép cặp hiện nay là vô cùng lớn và cũng như là nhu cầu của việc dịch ảnh được áp dụng trong một số bài toán. Và trên thực tế thì cycleGAN đưa ra các kết quả chuyển đổi sắc thái, phong cách và trạng thái của ảnh. Nhưng với việc thay đổi về cấu trúc hình học của đối tượng trong ảnh cycleGAN không cho ra được kết quả tốt trong các phương pháp dịch ảnh với tập dữ liệu không ghép cặp được ra mắt vào cùng thời điểm. CycleGAN được xem là phương pháp tiên phong mở đầu xu hướng mới cho bài toán dịch ảnh. CycleGAN còn là tiền đề cho GANILLA(Generative Adversarial Networks for Image to Illustration Translation) phát triển dựa trên cơ sở với hai miền dữ liệu không được ghép cặp.



Hình 13: Real image



Hình 14: Fake image

Với tác vụ chuyển đổi từ ngựa sang ngựa vằn, các tác giả dùng dữ liệu từ ImageNet với lớp ngựa(wild horse) là 939 ảnh, ngựa vằn(zebra) là 1177 với tỷ lệ là 256x256 pixels. Huấn luyện với tỷ lệ học là 0.0002 họ giữ nguyên giá trị đó trong 100 epochs và giảm dần về 0 trong 100 epochs tiếp

theo, trọng số khởi tạo từ một phân phối Gaussian $N(0, 0.02)$.

7.3 StyleGAN

Mã nguồn Image Generation using Stylegan pre-trained model được sử dụng để tạo sinh ảnh với hai bộ kiến trúc GAN truyền thống và StyleGAN, mã nguồn thể hiện sự khác biệt khi tạo sinh ảnh khi trộn phong cách bằng phương pháp styleGAN bằng cách trộn hai vector latent w sau khi mapping hai vector z và tạo sinh ảnh bằng cách trộn 2 vector latent z lại với nhau. Kết quả cho thấy được sự khác biệt nếu trộn phong cách bằng kiến trúc styleGAN so với trộn phong cách theo kiểu trộn hai vector latent z rồi tạo sinh theo cách truyền thống. Cách tạo sinh theo kiểu trộn half $z_1 + \text{half } z_2$ có thể được hiểu là mình sẽ tạo ra một vector latentz mới để bỏ vào kiến trúc tạo sinh, vector latentz này có thể một số điểm dữ liệu có tương đồng với một số điểm dữ liệu của half z_1 và half z_2 nên ảnh sau khi tạo ra có thể giống một số nét cơ bản chứ thật chất là mình đang tạo sinh ảnh bằng một vector latent z mới. Với cách trộn phong cách của StyleGAN ta có thể thấy được sự dung hợp của kết quả là từ phong cách của hai bức ảnh được tạo ra bằng z_1 và z_2 , cho thấy được kiến trúc bộ tạo sinh StyleGAN linh hoạt hơn trong việc tạo sinh ảnh

Tài liệu

- [1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” 2014.
- [2] Wikipedia, “Hàm gauss — wikipedia, bách khoa toàn thư mở,” 2024. [Trực tuyến; truy cập 1-01-2024].
- [3] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” 2017.
- [4] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” 2014.
- [5] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” 2017.
- [6] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, “Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better,” 2019.
- [7] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “Least squares generative adversarial networks,” 2017.
- [8] A. Jolicoeur-Martineau, “The relativistic discriminator: a key element missing from standard gan,” 2018.
- [9] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” 2017.
- [10] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, Feb. 2017.
- [11] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilnetv2: Inverted residuals and linear bottlenecks,” 2019.
- [12] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” 2020.
- [13] M. Assens, X. G. i Nieto, K. McGuinness, and N. E. O’Connor, “Pathgan: Visual scanpath prediction with generative adversarial networks,” 2018.

- [14] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” 2019.
- [15] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” 2020.
- [16] M. J. Chong and D. Forsyth, “Jojogan: One shot face stylization,” 2022.
- [17] J. Shi, L. Xu, and J. Jia, “Discriminative blur detection features,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2965–2972, 2014.
- [18] S. Zhou, J. Zhang, W. Zuo, H. Xie, J. Pan, and J. Ren, “Davanet: Stereo deblurring with view aggregation,” in *CVPR*, 2019.