

Phục hồi hình ảnh chân dung sử dụng Generative Adversarial Networks

1st Trương Thị Cẩm Ly

Trường Đại học Công nghiệp TP.HCM
Hồ Chí Minh, Việt Nam
truongthicamly1412@gmail.com

2nd Hồ Quang Huy

Trường Đại học Công nghiệp TP.HCM
Hồ Chí Minh, Việt Nam
quanghuyho.iuh@gmail.com

3rd Trần Minh Hiếu

Trường Đại học Công nghiệp TP.HCM
Hồ Chí Minh, Việt Nam
hieuiuh2001@gmail.com

TÓM TẮT NỘI DUNG

Phục hồi hình ảnh chân dung (facial inpainting) là quá trình phục hồi các phần bị khuyết, hư hỏng trong hình ảnh chân dung, tạo ra các cấu trúc khuôn mặt hợp lý cho các pixel bị thiếu. Đây là một nhiệm vụ đầy thách thức trong đó các vùng bị thiếu phải được lắp đầy dựa trên dữ liệu trực quan (visual data) có sẵn. Các phương pháp trước đây chỉ trích xuất thông tin từ một hình ảnh duy nhất và thường tạo ra kết quả không đạt yêu cầu do thiếu bối cảnh cấp cao (high level context). Gần đây, phương pháp điều chỉnh dữ liệu đầu vào có sẵn được nghiên cứu và ứng dụng vào các bài toán khôi phục hình ảnh cho ra kết quả rất tốt. Để hiểu rõ hơn về phương pháp này, nhóm chúng tôi tiến hành hiện thực các mô hình dữ liệu có khả năng sinh ra dữ liệu mới (generative model). Quá trình thực hiện, chúng tôi tiến hành mã hóa hình ảnh đầu vào là những hình ảnh chưa được phục hồi bằng cách sử dụng ngữ cảnh hình ảnh và các tổn thất hình ảnh trước đó, mã hóa này sau đó được chuyển qua mô hình có khả năng sinh ra dữ liệu mới thông qua mạng đối nghịch (generative adversarial networks) để khôi phục phần ảnh bị thiếu.

Dựa trên bộ dữ liệu CelebA với hơn 200,000 hình ảnh chân dung khác nhau về tư thế và các biến thể nền. Trong bài nghiên cứu này chúng tôi kỳ vọng sẽ phục hồi được nội dung còn thiếu của hình ảnh, dự đoán thành công thông tin ở những vùng bị thiếu lớn và đạt được trạng thái ảnh hiện thực (photorealism).

Index Terms—Facial inpainting, Image inpainting, Deep generative model, Generative adversarial networks

I. GIỚI THIỆU

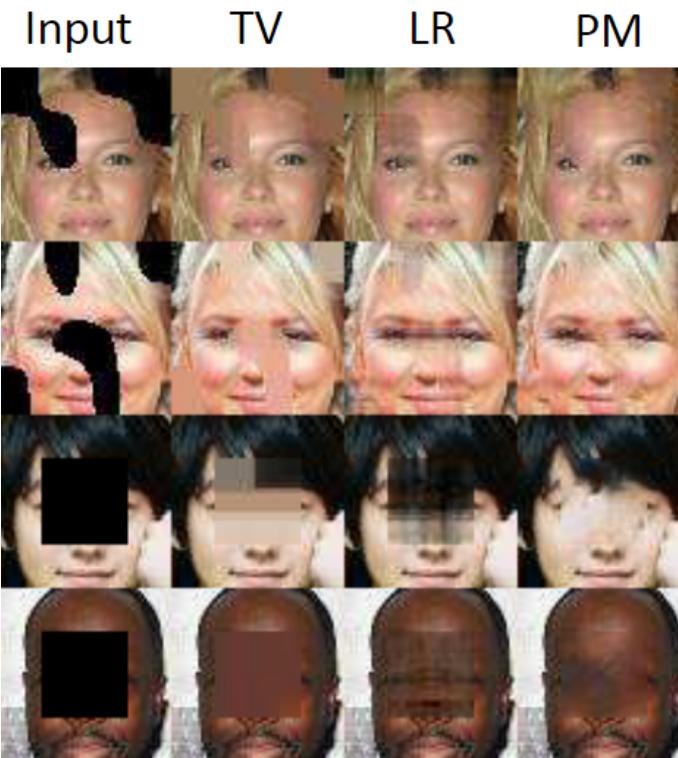
Phục hồi hình ảnh chân dung đề cập đến nhiệm vụ suy ra các vùng bị thiếu bất kỳ trong hình ảnh chân dung. Do dự đoán bối cảnh cấp cao là bắt buộc nên nhiệm vụ này khó hơn đáng kể so với việc hoàn thiện hình ảnh. Việc khôi phục hình ảnh được ứng dụng nhiều như khôi phục các bức tranh cũ bị hư hỏng hoặc chỉnh sửa hình ảnh. Tuy nhiên, việc khôi phục sẽ trở nên khó khăn nếu các vùng lớn bị thiếu hoặc nếu các cảnh phức tạp.

Các phương pháp phục hồi cổ điển thường dựa trên thông tin cục bộ hoặc không cục bộ để khôi phục ảnh. Hầu hết các phương pháp hiện có được thiết kế để vẽ một hình ảnh trong bức tranh. Do đó, chúng dựa trên thông tin có sẵn trong hình

ảnh đầu vào và khai thác hình ảnh trước để giải quyết tình trạng thiếu thông tin trong hình ảnh. Ví dụ: Phương pháp tiếp cận dựa trên total variation (TV) [4], tính đến thuộc tính độ mịn của hình ảnh tự nhiên, điều này rất hữu ích để lắp đầy các vùng bị thiếu nhỏ hoặc loại bỏ nhiễu giả. Các lỗ hổng trong ảnh có kết cấu có thể được lắp đầy bằng cách tìm một kết cấu tương tự từ cùng một hình ảnh trước đó, chẳng hạn như số liệu thống kê về patch offsets, planarity hoặc low rank (LR) [?] cũng có thể cải thiện đáng kể kết quả. PatchMatch (PM) [5] tìm kiếm các bản vá tương tự trong phần có sẵn của hình ảnh và nhanh chóng trở thành một trong những phương pháp vẽ thành công nhất nhờ chất lượng và hiệu quả cao. Tuy nhiên, tất cả các phương pháp vẽ ảnh đơn lẻ đều yêu cầu thông tin thích hợp được chứa trong ảnh đầu vào như giá trị pixel, cấu trúc thông tin hình ảnh, ... Giả định này khó thỏa mãn nếu vùng bị thiếu lớn và có thể có hình dạng tùy ý. Do đó, trong trường hợp này, các phương pháp này không thể khôi phục lại thông tin còn thiếu. Hình 1 cho thấy một số trường hợp thách thức với các vùng lớn bị thiếu, khi các phương pháp cổ điển không thể phục hồi được mũi và mắt.

Thay vì tập trung lắp đầy các phần khuyết trên hình ảnh, chúng tôi quan tâm đến việc dự đoán nội dung chi tiết của một vùng rộng lớn dựa trên bối cảnh của các pixel xung quanh. Một cách tiếp cận mới và đã được chứng minh hiệu quả trong vấn đề khôi phục hình ảnh mới nhất là phương pháp Context Encoder (CE). Phương pháp này đưa ra một mặt nạ để lắp đầy các vùng bị thiếu, một mạng lưới thần kinh được đào tạo để mã hóa thông tin ngữ cảnh và dự đoán nội dung không có sẵn. Tuy nhiên, CE chỉ tận dụng cấu trúc của các phần nhỏ trong quá trình đào tạo chứ không phải trong quá trình suy luận. Do đó, nó dẫn đến hình ảnh mờ hoặc không thực tế, đặc biệt là khi các vùng bị thiếu có hình dạng tùy ý.

Trong bài báo này, chúng tôi thực nghiệm một số phương pháp sử dụng mô hình tự sinh thông tin để phục hồi hình ảnh chân dung. Sau một mô hình tổng quát sâu, tức là trong trường hợp của chúng tôi, một mạng lưới đối nghịch (generative adversarial networks - GAN) được đào tạo, chúng tôi tìm kiếm một mã hóa hình ảnh gần nhất với hình ảnh gốc. Mã hóa sau đó được sử dụng để tái tạo lại hình ảnh bằng trình tạo (generative). So với CE, một trong những ưu điểm chính của phương pháp của chúng tôi là nó không yêu cầu mặt nạ để đào tạo và có thể được áp dụng cho các vùng bị thiếu có cấu trúc tùy ý trong quá trình suy luận. Chúng tôi đánh giá



Hình 1: Kết quả phục hồi hình ảnh chân dung từ các phương pháp cổ điển như TV, LR, PM. Các phần khuyết cần khôi phục được đánh dấu bằng màu đen.

phương pháp của mình trên bộ dữ liệu CelebA với các dạng vùng bị thiếu khác nhau. Quá trình xây dựng và thực nghiệm phương pháp được chúng tôi mô tả chi tiết trong bài báo này.

II. NGHIÊN CỨU LIÊN QUAN

Các phương pháp phục hồi cổ điển thường dựa trên thông tin cục bộ hoặc không cục bộ để khôi phục ảnh. Hầu hết các phương pháp hiện có được thiết kế để vẽ một hình ảnh trong bức tranh, chúng dựa trên thông tin có sẵn trong hình ảnh đầu vào và khai thác hình ảnh tiên nghiệm để giải quyết tình trạng thiếu thông tin trong hình ảnh.

Phương pháp tiếp cận dựa trên biến thể tổng (Total variation - TV) xét dựa trên các thông tin đầy đủ có trong hình ảnh, điều này rất hữu ích để lấp đầy các vùng bị thiếu nhỏ hoặc loại bỏ nhiễu giả. Các vùng ảnh bị thiếu được lấp đầy bằng cách tìm một kết cấu tương tự từ cùng một hình ảnh.

PatchMatch (PM) tìm kiếm và lấp đầy các vùng thiếu từ các thông tin tương tự trong phần có sẵn của hình ảnh. Phương pháp này nhanh chóng trở thành một trong những phương pháp phục hồi thành công nhất do chất lượng và hiệu suất của nó mang lại. Tuy nhiên, tất cả các phương pháp phục hồi hình ảnh yêu cầu thông tin thích hợp có trong hình ảnh gốc. Ví dụ, thông tin các pixel, cấu trúc hoặc các bản vá tương tự. Nhưng yêu cầu này khó thỏa mãn nếu vùng bị thiếu lớn và thông tin trong hình ảnh gốc không rõ ràng. Do đó, trong các trường hợp này các phương pháp kể trên không thể khôi phục được đầy đủ thông tin bị thiếu của hình ảnh.

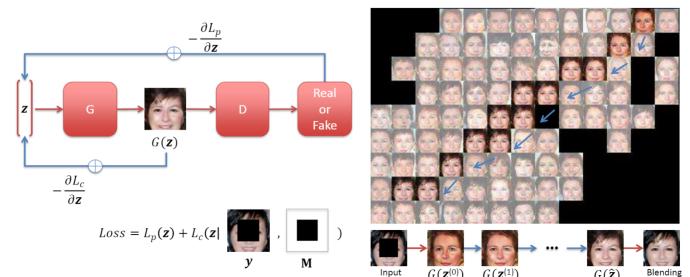
Generative Adversarial Networks (GANs) hình thành trên ý tưởng về sự cạnh tranh của hai mạng neural network. Discriminative network (D): có nhiệm vụ cỗ gắng phân biệt giữa dữ liệu thật và dữ liệu giả mạo. Generative network (G): sinh ra dữ liệu giả, và mục tiêu sinh ra được các dữ liệu giống với thật nhất làm Discriminator không thể phân biệt. Các mẫu được sinh ra dựa trên các mã ngầm (Latent code) z . Mạng phân biệt (Discriminator - Viết tắt là D) sẽ dựa trên những biến đầu vào \mathbf{X} để dự báo nhãn hoặc giá trị y . Về bản chất đây chính là một mô hình classification hoặc prediction. Mô hình sẽ dự báo đầu ra dựa trên các dấu hiệu là đầu vào đã biết và giá trị dự báo là một xác suất có điều kiện: $P(\mathbf{X}|y)$. Trong đó y là mục tiêu cần dự báo và \mathbf{X} được xem như điều kiện. Trái ngược lại so với mạng phân biệt (Discriminator) là mạng sinh (Generator - Viết tắt là G). Mạng sinh sẽ cố gắng dự báo $P(\mathbf{X}|y)$. Mô hình sẽ tập trung hơn vào việc tìm kiếm đặc trưng của dữ liệu như thế nào nếu như đã biết trước đầu ra của dữ liệu.

G ánh xạ một vectơ ngẫu nhiên z được khởi tạo ngẫu nhiên theo phân phối Gaussian vào không gian hình ảnh, từ tập vectơ đầu vào z ngẫu nhiên, G biến đổi ra bức ảnh giả ở output, bức ảnh giả này sẽ được sử dụng làm đầu vào cho D. D sẽ có tác dụng phân biệt ảnh input là thật hay giả. Nhãn của mô hình sẽ là thật nếu ảnh đầu vào của D được lấy tập mẫu huấn luyện và giả nếu được lấy từ output của mô hình G. Về bản chất đây là một bài toán phân loại nhị phân (binary classification) thông thường.

III. PHƯƠNG PHÁP

A. Kỹ thuật liên quan

1) **Deep convolutional GAN (DCGAN):** là mô hình GAN áp dụng trong các tác vụ của xử lý ảnh. Trong mô hình GANs thông thường, cả generator và discriminator đều được xây bằng mạng neural network thông thường với các fully connected layer thì ở DCGAN generator và discriminator được xây dựng bằng mô hình CNN với 2 layers chính là convolutional layer và transposed convolutional layer.



Hình 2: Khung mô hình dữ liệu để xuất cho phục hồi; Đưa ra mô hình DCGAN được đào tạo trên ảnh thực, chúng tôi lặp lại quá trình đào tạo này để cập nhật z từ đó tìm ánh xạ trên đa tạo hình ảnh tiềm ẩn (latent image) dựa trên các hàm mắt mong muốn; Sử dụng quá trình lan truyền ngược để cập nhật z , trong đó $z^{(0)}$ được khởi tạo ngẫu nhiên $z^{(k)}$ biểu thị kết quả trong lần lặp thứ k và \hat{z} biểu thị kết quả cuối cùng

Các mạng G và D được đào tạo bằng cách tối ưu hóa hàm mất mát (loss function):

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

trong đó, h là mẫu được lấy từ dữ liệu có phân phối p_{data} ; z là một mẫu ngẫu nhiên trong không gian tiềm ẩn (latent space). Thành phần $\log(1 - D(G(\mathbf{z}))$ đại diện cho loss của generator và $\log(D(\mathbf{h}))$ là loss của discriminator. Quá trình huấn luyện sẽ huấn luyện đồng thời G và D.

Pha huấn luyện Discriminator: Mục tiêu của pha này là huấn luyện một mô hình Discriminator sao cho khả năng phân loại là tốt nhất. Ở pha này tạm thời xem G không đổi và chỉ quan tâm đến $\max_D V(G, D)$. Đây là chính là đối của hàm cross entropy đối với trường hợp phân loại nhị phân sử dụng hồi quy logistic (logistic regression). Mục tiêu của hồi quy logistic đối với bài toán phân loại nhị phân là tối thiểu hóa mà cross entropy:

$$\mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = -\frac{1}{N} \sum_{i=1}^N [y_i \log p(y_i | \mathbf{x}_i)] + (1 - y_i) \log(1 - p(y_i | \mathbf{x}_i)) \quad (2)$$

Trong đó, $\log p(y_i | \mathbf{x}_i)$ là xác suất dự báo nhãn y_i từ mô hình logistic. Trong phương trình (1) thì $D(x)$ cũng như $\log p(y_i | \mathbf{x}_i)$. $D(x)$ đóng vai trò dự báo xác suất cho dữ liệu đầu vào. Có hai khả năng xảy ra:

- Nếu đầu vào là ảnh thật thì $y_i = 1$ và $1 - y_i = 0$ và do đó loss function tương ứng với $y_i \log p(y_i | \mathbf{x}_i) = \log p(y_i | \mathbf{x}_i)$ ở phương trình (2). Giá trị này được coi như là $\log D(x)$ ở phương trình (1). Kí hiệu $x \sim p_{\text{data}}(x)$ ở phương trình (1) là phân phối xác suất của các điểm dữ liệu đầu vào, trong trường hợp ở phương trình (2) thì các quan sát có vai trò như nhau nên chúng có chung giá trị phân phối là $\frac{1}{N}$.
- Trường hợp ảnh đầu vào là giả thì $y_i = 0$ và $1 - y_i = 1$. Khi đó đóng góp vào hàm loss function chỉ còn thành phần $(1 - y_i) \log(1 - p(y_i | \mathbf{x}_i)) = \log(1 - p(y_i | \mathbf{x}_i))$ ở phương trình (2). Giá trị này được coi như là $\log(1 - D(x))$ ở phương trình (1).

Pha huấn luyện Generator: Mục tiêu của pha này là cung cấp khả năng tạo ảnh của Generator sao cho ảnh sinh ra là giống với ảnh thật nhất. Ở pha này tạm thời xem D là không đổi và chỉ quan tâm đến $G(z)$ sao cho giá trị dự báo xác suất từ D đối với nó gần bằng 1 nhất, tức là ảnh giả được sinh ra giống ảnh thật nhất (xác suất càng gần 1 thì khả năng giống ảnh thật càng lớn). Như vậy $D(G(z))$ sẽ càng lớn càng tốt. Nếu đảo dấu trong $\mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$ ta suy ra mục tiêu cần tối ưu là tối thiểu hóa $\min_G V(D, G)$.

Quá trình huấn luyện mô hình sẽ kết hợp xen kẽ giữa 2 pha discriminator và generator. Với k batch dữ liệu đầu tiên mô hình sẽ huấn luyện discriminator trước:

- Huấn luyện discriminator: Lấy mẫu một mini-batch kích thước m là các nhiều $z^{(1)}, z^{(2)}, \dots, z^{(m)}$ và là

đầu vào của Generator. Đồng thời lấy mẫu một mini-batch khác kích thước m là những điểm dữ liệu thật $x^{(1)}, x^{(2)}, \dots, x^{(m)}$. Những dữ liệu này sẽ được sử dụng để cập nhật gradient descent theo phương pháp mini-batch gradient descent:

$$\frac{1}{m} \nabla_{\theta_D} \sum_{i=1}^m \log D(x^{(i)}) + \log(1 - D(G(z^{(i)}))) \quad (3)$$

Khi huấn luyện mô hình discriminator các hệ số trên mô hình discriminator là sẽ được cập nhật và các hệ số của mô hình generator được đóng băng.

• Huấn luyện generator: Sau khi kết thúc k batch huấn luyện trên discriminator chúng ta sẽ tiếp tục huấn luyện trên generator. một mini-batch kích thước m được lựa chọn ra từ các nhiều là $z^{(1)}, z^{(2)}, \dots, z^{(m)}$ được sử dụng như đầu vào huấn luyện. Gradient descent sẽ được tính trên m dữ liệu này theo công thức:

$$\frac{1}{m} \nabla_{\theta_G} \sum_{i=1}^m \log(1 - D(G(z^{(i)}))) \quad (4)$$

Cập nhật gradient descent chỉ được áp dụng trên các hệ số của Generator là θ_G . quá trình này sẽ được lặp lại cho tới khi tổng số lượt huấn luyện là đủ lớn hoặc loss của mô hình tiệm cận về 0.

2) Context Conditional GAN (CC-GAN): Context Conditional GAN là một mô hình GAN có điều kiện, trong đó:

- Generator (G) được đào tạo để điền vào bản vá hình ảnh bị thiếu (missing image patch) và
- Mô hình Generator và Discriminator được điều chỉnh dựa trên các pixel xung quanh.

CC-GAN còn làm một nhiệm vụ khác là Xác định xem một phần của hình ảnh là thật hay giả dựa trên bối cảnh xung quanh. Generator (G) nhận đầu vào là một hình ảnh với một phần bị che khuất ngẫu nhiên. Generator tiếp tục học và dự đoán các pixel trong phần ảnh bị thiếu dựa trên ngữ cảnh của toàn bộ hình ảnh. Ảnh sau khi đã được điền vào chỗ khuyết sau đó được đưa vào mô hình Discriminator (D) để nó nhận dạng được đây là ảnh thật hay ảnh giả.

Chính thức hơn, gọi $m \in \mathbb{R}^d$ biểu thị cho hình ảnh nhị phân được sử dụng để loại bỏ một phần trong ảnh (mask image). Generator nhận đầu vào là $m \odot x$ trong đó \odot biểu thị phép nhân từng phần tử. Generator sẽ tạo ra một ảnh $x_G = G(m \odot x, z) \in \mathbb{R}^d$ và hình vẽ bị khuyết được tạo ra là:

$$x_I = (1 - m) \odot X_G + m \odot x \quad (5)$$

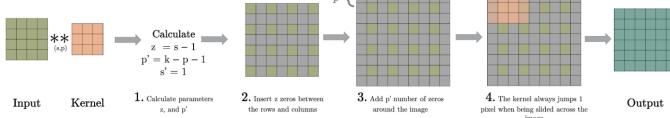
Mục tiêu của CC-GAN là:

$$\min_G \max_D \mathbb{E}_{x \sim X} [\log D(x)] + \mathbb{E}_{x \sim X, m \sim M} [\log(1 - D(x_I))] \quad (6)$$

Trong khi Generator tạo ra một hình ảnh đầy đủ thì Discriminator chỉ nhìn thấy một phần của nó (tương ứng với phần ảnh bị thiếu)

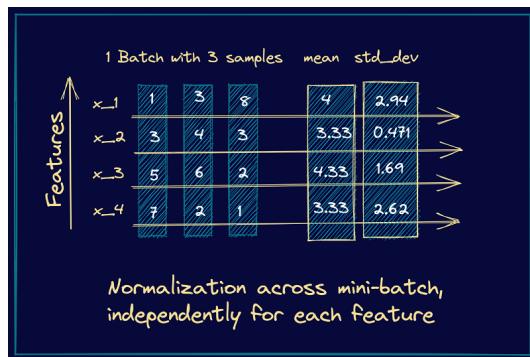
Trong đó, x là dữ liệu đầu vào và z là một vector ngẫu nhiên.

3) **Transposed convolutional**: hay deconvolution có thể coi là phép toán ngược của convolution. Nếu như convolution với stride > 1 giúp làm giảm kích thước của ảnh thì transposed convolution với stride > 1 sẽ làm tăng kích thước ảnh. Ví dụ stride = 2 và padding = ‘SAME’ sẽ giúp gấp đôi width, height kích thước của ảnh.



Hình 3: Các bước thực hiện transposed convolution

4) **Batch Norm**: Batch Normalization hay batch-norm có nghĩa là chuẩn hóa theo từng batch giá trị đầu vào của layer bất kì. Chuẩn hóa có nghĩa là đưa phân phối của layer về xấp xỉ phân phối chuẩn (trung bình xấp xỉ bằng 0 và phương sai xấp xỉ bằng 1). Trên thực tế, chúng ta chia tập dữ liệu huấn luyện thành từng batch (hay còn gọi là mini-batch) với kích thước 4, 8, 16, 32, 64..., batch-norm được tính toán trên các mini-batch đó.



Hình 4: Batch Normalization

Công thức tính trung bình của mini-batch:

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i \quad (7)$$

Công thức tính phương sai của mini-batch:

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 \quad (8)$$

Công thức chuẩn hóa từng batch:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (9)$$

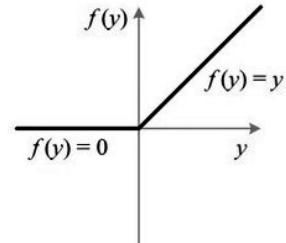
Trên thực tế, đôi khi mô hình lại hoạt động hiệu quả với một trung bình và phương sai khác, nên tác giả đã thêm 2 siêu tham số là gamma-scale và beta-shift để có tính tổng quát:

$$y_i = \gamma \hat{x}_i + \beta \quad (10)$$

Về mặt ý nghĩa, phân phối của dữ liệu đầu vào sẽ bị thay đổi dần do việc cập nhật trọng số của các layer trước đó, làm cho phân phối của đầu vào của các layer phía sau sẽ khác xa so với phân phối của data input. BN giúp cố định phân phối của dữ liệu về phân phối chuẩn, qua tất cả các lớp, dẫn tới tính chất phân phối của dữ liệu không thay đổi qua các lớp.

5) **ReLU (rectified linear unit)**: biến đổi đầu vào thành giá trị lớn nhất là 0 hoặc chính đầu vào đó.

$$f(x) = \max(0, x) \quad (11)$$



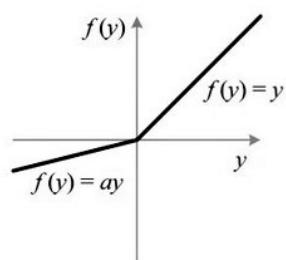
Hình 5: ReLU

ReLU được chứng minh giúp cho việc training các Deep Networks nhanh hơn rất nhiều so với sử dụng tanh. ReLU được tính toán gần như tức thời và gradient của nó cũng được tính cực nhanh với gradient bằng 1 nếu đầu vào lớn hơn 0, bằng 0 nếu đầu vào nhỏ hơn 0.

6) **Leaky ReLU**: Hàm Leaky ReLU có các điểm tốt của hàm ReLU và giải quyết được vấn đề Dying ReLU bằng cách xét một độ dốc nhỏ cho các giá trị âm thay vì để giá trị là 0.

$$f(x) = \max(0.1x, x) \quad (12)$$

Đồ thị hàm số của leaky ReLU:

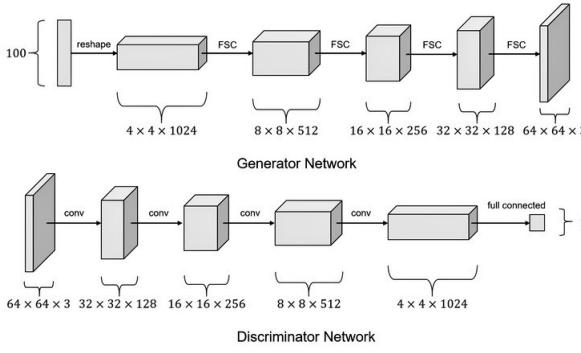


Hình 6: Leaky ReLU

B. Kiến trúc mô hình

1) **Kiến trúc mô hình DC-GAN**: Trong Mô hình DCGAN generator và discriminator được xây dựng bằng mô hình CNN với 2 layers chính là convolutional layer và transposed convolutional layer.

Mạng Generator nhằm mục đích sinh ảnh giả, đầu vào là noise vector kích thước 100 và đầu ra và ảnh giả cùng kích thước ảnh thật với các layer trong mạng:



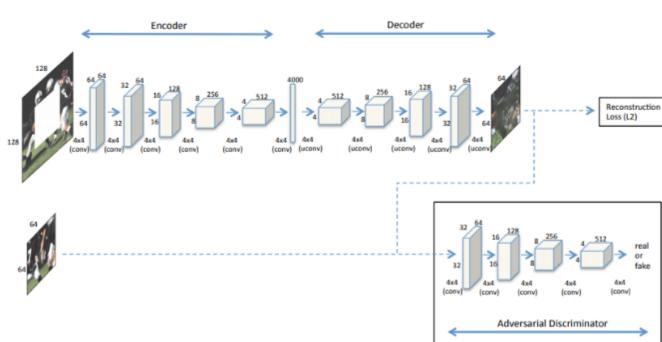
Hình 7: Kiến trúc mô hình DCGAN

- Flatten chuyển từ vector về dạng tensor 3d: $(100 \times 1) \rightarrow (4 \times 4 \times 1024)$
- Transposed convolutional layer: $(4 \times 4 \times 1024) \rightarrow (8 \times 8 \times 512)$
- Transposed convolutional layer: $(8 \times 8 \times 512) \rightarrow (16 \times 16 \times 256)$
- Transposed convolutional layer: $(32 \times 32 \times 128) \rightarrow (16 \times 16 \times 256)$
- Transposed convolutional layer: $(32 \times 32 \times 128) \rightarrow (64 \times 64 \times 3)$

Các lớp batch norm và ReLU theo sau mỗi lớp transposed convolutional.

Mạng Discriminator nhằm mục đích phân biệt ảnh thật từ dataset và ảnh giả do Generator sinh ra, đầu vào là ảnh kích thước $(64 \times 64 \times 3)$, đầu ra là giá trị phân biệt giữa ảnh thật và ảnh giả (binary classification). Mô hình discriminator đối xứng lại với mô hình generator. Ảnh đầu vào được đi qua convolution với stride = 2 để giảm kích thước ảnh từ $(64 \times 64 \times 3) \rightarrow (32 \times 32 \times 128) \rightarrow (16 \times 16 \times 256) \rightarrow (8 \times 8 \times 512) \rightarrow (4 \times 4 \times 1024)$. Khi giảm kích thước hình ảnh thì chiều sâu hình ảnh sẽ tăng dần. Cuối cùng thì tensor shape $(4 \times 4 \times 1024)$ thông qua hàm sigmoid để đưa ra xác suất cho mô hình. Các lớp batch norm và LeakyReLU theo sau mỗi lớp convolution.

2) *Kiến trúc mô hình CC-GAN:* Kiến trúc mô hình CC-GAN được chúng tôi sử dụng được mô tả như Hình 8



Hình 8: Kiến trúc mô hình CC-GAN

Input của mô hình là một ảnh đã bị cắt (hoặc ảnh nhiễu)

sau đó qua 6 lớp convolution để downsampling ảnh về một vector có kích thước $1 \times 1 \times 4000$. Sau đó vector này sẽ qua 5 lớp transpose convolution để upsampling ảnh về kích thước $64 \times 64 \times 3$. Ảnh này là ảnh Generator sinh ra ở phần ảnh bị khuyết. Sau đó ảnh này cũng với nhãn y được đưa vào mô hình Discriminator để nó phân biệt ảnh thật hay ảnh giả

IV. DỮ LIỆU

A. Tập dữ liệu

Chúng tôi sử dụng bộ dữ liệu CelebA với hơn 200,000 hình ảnh chân dung khác nhau về tư thế và các biến thể nền. Kích thước của ảnh là $218 \times 178 \times 3$. Mô tả về dữ liệu được biểu diễn trong Hình 9



Hình 9: CelebA dataset

B. Chuẩn bị dữ liệu

Chúng tôi tạo ra một trình tải dữ liệu bằng phương thức DataLoader của PyTorch để đưa trực tiếp từng tập dữ liệu nhỏ theo batch vào mô hình thay vì tải hết tất cả dữ liệu ngay từ đầu vì kích thước của tập dữ liệu đào tạo rất lớn dẫn đến việc không thể tải cùng lúc tất cả dữ liệu ngay từ đầu.

Đối với mô hình DCGAN, ảnh sẽ được chuyển về kích thước $(64 \times 64 \times 3)$ và sẽ được cắt random để đưa vào mô hình. Đối với mô hình CC-GAN, ảnh sẽ được chuyển về kích thước $(128 \times 128 \times 3)$ và mask image có kích thước là 64×64 được cắt random đối với data train, center đối với data test. Trong đó, tập dữ liệu huấn luyện có kích thước 198608 ảnh và tập dữ liệu thử nghiệm có 5344 ảnh.

V. KẾT QUẢ THỰC NGHIỆM

A. Chỉ số đánh giá

1) *BCEWithLogitLoss:* Chúng tôi sử dụng BCEWithLogitLoss cho Discriminator của mô hình DCGAN. Hàm mất mát này kết hợp một lớp Sigmoid và BCELoss trong một lớp duy nhất. Phiên bản này ổn định hơn về mặt số lượng so với việc sử dụng Sigmoid đơn giản theo sau là BCELoss, bằng cách kết hợp các hoạt động thành một lớp, chúng tôi tận dụng thủ thuật log-sum-exp để ổn định số lượng. Trong trường hợp phân loại nhiều nhãn, hàm mất mát có thể được mô tả như sau:

$$\begin{aligned} l_c(X, y) &= L_c = \{l_{1,c}, \dots, l_{N,c}\} \\ l_{n,c} &= -w_{n,c} [p_c y_{n,c} \cdot \log \sigma(X_{n,c}) \\ &\quad + (1 - y_{n,c}) \cdot \log(1 - \sigma(X_{n,c}))] \end{aligned}$$

Trong đó, c là số class, n là số batch, P_c là trọng số của mỗi class c

2) *BCELoss*: Chúng tôi sử dụng BCELoss cho Generator của mô hình DC-GAN. Hàm mất mát này tạo một tiêu chí để đo Binary Cross Entropy giữa mục tiêu và xác suất đầu vào. Hàm mất mát với reduction là 'none' có thể được mô tả là:

$$l(x, y) = L = \{l_1, \dots, l_N\}^T, \quad (13)$$

$$l_n = -w_n[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n)]$$

Trong đó, N là kích thước của mỗi batch. Nếu reduction khác 'none', khi đó:

$$l(x, y) = \begin{cases} \text{mean}(L), \text{if reduction = 'mean'} \\ \text{sum}(L), \text{if reduction = 'sum'} \end{cases} \quad (14)$$

Trong đó x và y là mảng tensor có kích thước n phần tử.

3) *L1Loss*: Chúng tôi sử dụng L1Loss cho Generator của mô hình CC-GAN. Nó tạo ra sai số tuyệt đối trung bình (MAE) giữa từng phần tử trong input x và output y. Hàm mất mát với reduction là 'none' có thể được mô tả là:

$$l(x, y) = L = \{l_1, \dots, l_N\}, l_n = |x_n - y_n| \quad (15)$$

Trong đó, N là kích thước của mỗi batch. Nếu reduction khác 'none', khi đó:

$$l(x, y) = \begin{cases} \text{mean}(L), \text{if reduction = 'mean'} \\ \text{sum}(L), \text{if reduction = 'sum'} \end{cases} \quad (16)$$

Trong đó x và y là mảng tensor có kích thước n phần tử.

4) *MSELoss*: Chúng tôi sử dụng MSELoss cho Discriminator đối với mô hình CC-GAN. Hàm loss này sẽ tạo ra sai số bình phương trung bình (bình phương chuẩn L2) giữa mỗi phần tử trong đầu vào x và mục tiêu y. Hàm mất mát với reduction là 'none' có thể được mô tả là:

$$l(x, y) = L = l_1, \dots, l_N, l_n = (x_n - y_n)^2 \quad (17)$$

Trong đó, N là kích thước của mỗi batch. Nếu reduction khác 'none', khi đó:

$$l(x, y) = \begin{cases} \text{mean}(L), \text{if reduction = 'mean'} \\ \text{sum}(L), \text{if reduction = 'sum'} \end{cases} \quad (18)$$

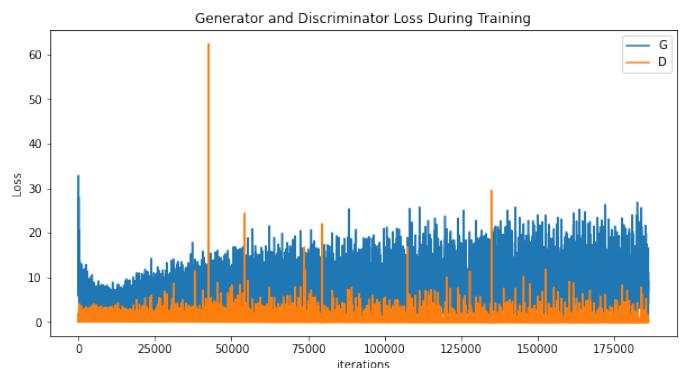
Trong đó, x, y là một tensor có kích thước là n phần tử.

B. Kết quả thực nghiệm

1) *DCGAN*: Mô hình đào tạo trên 180000 hình ảnh và được huấn luyện với 30 epoch. "Adam" được chọn làm hàm tối ưu hóa của mô hình với tốc độ học (learning rate) được đặt thành $2e - 4$ và các chỉ số beta1 và beta 2 lần lượt là 0.5 và 0.999.

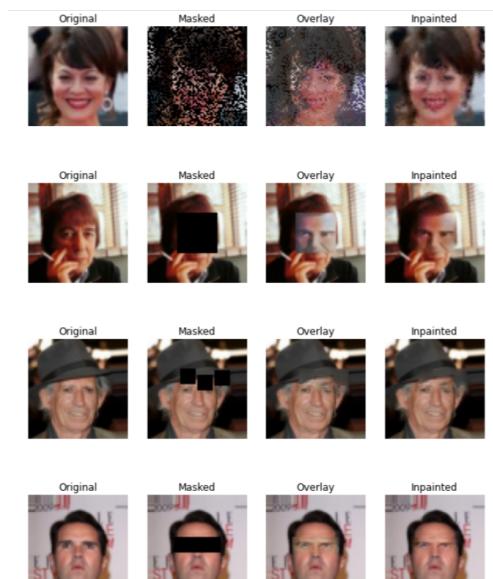
Mô hình sử dụng BCELoss đối với Generator và BCEWithLogitsLoss đối với Discriminator (như đã đề cập ở phần chỉ số đánh giá)

Hàm mất mát (loss function) của mô hình được thể hiện ở hình 10.



Hình 10: Loss của mô hình DCGAN.

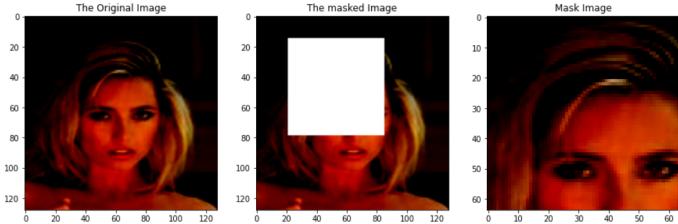
Kết quả output của mô hình DC-GAN được đưa ra ở hình 11. Trong đó, Original là ảnh gốc của mô hình, Masked là ảnh sau khi đã cắt để thêm các nhiễu làm đầu vào của mô hình, Overlay là ảnh được xếp chồng phần ảnh bị khuyết của ảnh gốc (hay phần ảnh đã bị cắt random) và ảnh sau khi được phục hồi. Và cuối cùng là phục hồi, đó là ảnh mà mô hình Generator tạo ra (hay ảnh kết quả của mô hình).



Hình 11: Kết quả của mô hình DC-GAN

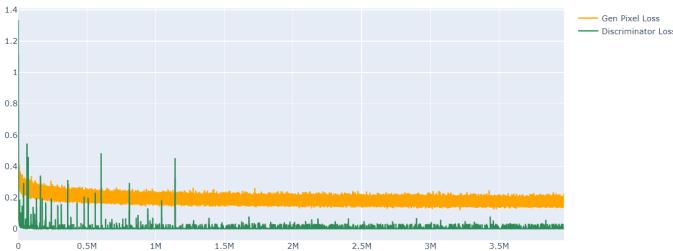
Hàm mất mát của discriminator có xu hướng hội tụ trong khi đó hàm mất mát của generator chưa hội tụ. Tuy nhiên, hàm mất mát của cả discriminator và generator giảm theo cùng một tốc độ.

2) *CC-GAN*: Mô hình này phức tạp hơn mô hình DC-GAN và có kết quả tốt hơn DC-GAN. Kiến trúc mô hình đã được chúng tôi đề cập ở Hình 8 với 11 lớp convolution ở mô hình Generator và 5 lớp convolution ở mô hình Discriminator. Ở mô hình này, chúng tôi sử dụng tập dữ liệu huấn luyện với mask image được cắt random, được mô tả như Hình 12. Sau đó thử nghiệm bằng tập dữ liệu được cắt ở trung tâm để tiện lợi hơn trong quá trình quan sát và đánh giá mô hình.



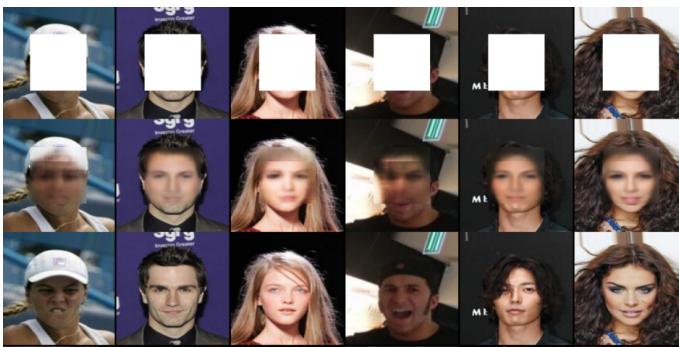
Hình 12: Ảnh thật và ảnh sau khi bị cắt đi một phần dữ liệu

Chúng tôi tiến hành huấn luyện mô hình với 20 epoch và chỉ số đánh giá là 2 hàm Loss ở mỗi mô hình Generator và Discriminator lần lượt là L1Loss và MSELoss. Cũng như DC-GAN, chúng tôi sử dụng hàm tối ưu Adam, với các tham số lần lượt là: learning rate = 0.00008, beta1 = 0.5, beta2 = 0.999 với kích thước Kết quả loss được biểu diễn trong Hình 13



Hình 13: Loss của mô hình CCGAN.

Sau khi huấn luyện với 20 epoch, kết quả chúng tôi thu được như Hình 15. Kết quả với loss của mô hình Generator là 0.169 và loss của mô hình Discriminator là 0.00032. Với kết quả loss này, đầu ra mô hình CCGAN của chúng tôi mang lại kết quả khá tốt. Cụ thể đầu ra của mô hình được biểu diễn trong hình sau:



Hình 14: Kết quả đầu ra của mô hình CC-GAN.

C. So sánh kết quả

Dựa vào kết quả thử nghiệm thu được ở (Hình 10) và (Hình 13) chúng ta có thể tóm tắt lại kết quả ở bảng sau:

Dự vào bảng kết quả trên, kết quả Loss mang lại ở cả 2 hình cho thấy loss của mô hình CC-GAN đang hội tụ tốt hơn so với DC-GAN.

Model	BCEWL	BCE	MSE	L2
D(DCGAN)	0.01	-	-	-
G(DCGAN)	-	6.5697	-	-
D(CC-GAN)	-	-	0.00032	-
G(CC-GAN)	-	-	-	0.169

Hình 15: Bảng so sánh 2 mô hình

VI. KẾT LUẬN

Trong bài báo này, chúng tôi đã đề xuất hai phương pháp mới để vẽ phục hồi ngữ nghĩa hình ảnh, cụ thể là phục hồi ảnh chân dung. Hai phương pháp đó là DC-GAN và CC-GAN. So với các phương pháp hiện có dựa trên các ảnh bị lỗi, hai phương pháp được đề xuất học cách biểu diễn dữ liệu huấn luyện và do đó có thể dự đoán nội dung có ý nghĩa cho hình ảnh bị hỏng. So với bài báo gốc, phương pháp của chúng tôi thường thu được hình ảnh có các cạnh sắc nét hơn, trông chân thực hơn nhiều. Kết quả thử nghiệm đã chứng minh hiệu suất vượt trội của nó trên các ví dụ về hình ảnh trong tranh đầy thách thức.

LỜI CẢM ƠN

Chúng tôi chân thành cảm ơn quý thầy bộ môn đã tận tình giúp đỡ và hướng dẫn chúng tôi hoàn thành đề tài. Cảm ơn các thành viên đã cùng nhau đóng góp và hoàn thiện đề tài. Do kinh nghiệm của các thành viên nhóm còn hạn chế nên không thể tránh khỏi sai sót, rất mong sẽ nhận được những lời nhận xét và góp ý đến từ các thầy và các bạn để bài có thể hoàn thiện hơn. Xin chân thành cảm ơn!

TÀI LIỆU

- [1] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo. An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems. IEEE TIP, 2011.
- [2] Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. ACM TOG, 2009.
- [3] . Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. IEEE PAMI, 2013.
- [4] J. Shen and T. F. Chan. Mathematical models for local non- texture inpaintings. SIAM Journal on Applied Mathematics, 2002.
- [5] Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. ACM TOG, 2009.
- [6] J. Hays and A. A. Efros. Scene completion using millions of photographs. ACM TOG, 2007.
- [7] K. He and J. Sun. Statistics of patch offsets for image completion. In ECCV. 2012.
- [8] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. IEEE PAMI, 2013.
- [9] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Image completion using planar structure guidance. ACM TOG, 2014.
- [10] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In ECCV, 2016.
- [11] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In ICLR, 2015.
- [12] D. Kingma and M. Welling. Auto-encoding variational bayes. In ICLR, 2014.