ỨNG DỤNG MẠNG GAN TRONG BÀI TOÁN SINH DỮ LIỆU ĐA PHƯƠNG TIỆN

Trần Quý Nam
Đại học Đại Nam

Tóm tắt: Ảnh các nhân vật Pokemon có tính chất nhân tạo, mang tính hoạt hình nên khác với các bài toán xử lý ảnh thông thường. Nghiên cứu này thử nghiệm, đánh giá khả năng đáp ứng của mô hình mạng học sâu GAN (Generative Adversarial Networks) đối với tập ảnh các nhân vật Pokemon. Mô hình sử dụng mạng nơ ron tích chập cho phần phân biệt và mạng nơ ron giải chập cho phần sinh dữ liệu ảnh nhân vật Pokemon. Tập dữ liệu thử nghiệm là bộ ảnh Veekun. Kết quả thử nghiệm cho thấy, dù là hình ảnh trò chơi nhưng mô hình GAN vẫn có khả năng áp dụng khá phù hợp, có tiềm năng ứng dụng cho bài toán sinh dữ liệu đa phương tiện.

Từ khóa: GAN, Pokemon, mô hình sinh, mô hình phân biệt, ảnh thật, ảnh giả.

I. GIỚI THIỆU

Ngành công nghệ đa phương tiện hiện nay đang phát triển đa dạng với sự phát triển sản phẩm Games, thực tế ảo (VR), thực tế ảo tăng cường (AR), thực tế hỗn hợp (MR), thực tế ảo mở rộng Extended Reality (XR)… Metaverse xuất hiện, thể hiện một xu thế phát triển các công nghệ ảo hóa và mở ra không gian lớn cho hoạt động nghiên cứu và ứng dụng. Trò chơi Pokemon Go là một trong các ví dụ sử dụng các công nghệ ảo hóa này.

Hiện nay, trò chơi Pokemon có hữu hạn nhân vật (khoảng 800) hình ảnh đồ họa (Hình 1). Quá trình nâng cấp trò chơi, phát triển kịch bản, cần các nhân vật mới. Trong khi đó, việc vẽ thủ công, dùng đồ họa máy tính thông thường bị giới hạn khả năng tăng số lượng nhân vật và tính đa dạng của nhân vật, chưa tự động kế thừa các đặc tính của nhân vật cũ. Đồng thời, nhà phát triển games muốn thăm dò thị hiếu người chơi với dự kiến hình ảnh nhân vật mới thì việc tự động hóa, tốn ít chi phí trong tạo hình nhân vật đóng một vai trò nhất định. Để phát triển mở rộng trò chơi Pokemon, việc tự động tạo thêm các nhân vật trò chơi ít tốn kém chi phí thông qua mạng GAN sẽ đóng góp một hướng nghiên cứu tiềm năng trong công nghiệp trò chơi.

Mô hình GAN (Generative Adversarial Networks) do Goodfellow và cộng sự đưa ra năm 2014 [1]. GAN là mô hình sinh dữ liệu, nghĩa là mô hình có khả năng sinh ra dữ liệu mới từ tập dữ liệu đầu vào có các đặc điểm tương tự.

Hình 1: Nhân vật Pokemon

Những năm gần đây, có nhiều thử nghiệm với ứng dụng thay đổi tuổi của khuôn mặt, thay đổi độ tuổi của khuôn mặt người nào đó. Dựa trên khuôn mặt của con người hiện tại, GAN sẽ sinh ra các biến thể theo từng độ tuổi của con người. Trên thực tế, có thể thử nghiệm các ứng dụng này trên các mạng xã hội như Instagram, TikTok... Các ứng dụng đó tạo ra những ảnh mặt người già nua đi sau vài năm tới và hình ảnh tiến hóa này có thể được sinh ra bởi GAN, không phải mặt người thật. Dữ liệu sinh ra nhìn như thật nhưng không phải là ảnh thật. GAN chứa hai mạng nơ-ron riêng biệt, một mạng nơ-ron đóng vai trò sinh dữ liệu (Generator) và một mạng nơ-ron khác đóng vai trò phân biệt (Discriminator). Đầu tiên, bộ phận sinh dữ liệu tạo ra các hình ảnh ngẫu nhiên và bộ phận phân biệt sẽ đánh giá những hình ảnh đó và cho bộ phận sinh đã tạo ra dữ liệu đó biết mức độ chân thực của các hình ảnh được tạo ra. Discriminator sẽ là đối thủ của Generator được cung cấp cùng lúc với cả hình ảnh được sinh ra cũng như loại hình ảnh gốc cho phép Discriminator phân biệt. Sau khi đạt đến một điểm nhất định, bộ phận phân biệt (Discriminator) sẽ không thể biết được hình ảnh tạo ra bởi bộ phận sinh (Generator) là ảnh thật hay ảnh giả, khi đó mô hình tốt, sinh dữ liệu giả giống như thật. Trên thực tế, chất lượng của những ứng dụng GAN áp dụng trên khuôn mặt ngày càng tốt hơn qua từng năm.

Mặc dù có phạm vi ứng dụng hẹp, nhưng bài toán sinh ảnh (generating images) vẫn có ý nghĩa thực tiễn nhất định. Trong ngành công nghiệp phim hoạt hình Nhật Bản, đã có một số bài báo trình bày tiềm năng ứng dụng mạng GAN để tạo nhân vật hoạt hình cho nhân vật hoạt hình có tên là Anime [10]. Thay vì sử dụng các họa sỹ, sử dụng nhân lực thủ công và tốn nhiều chi phí, tốn công sức để

Tác giả liên hệ: Trần Quý Nam,
Email: namtq.dn@gmail.com
Đến tòa soạn: 12/2022, chỉnh sửa: 02/2023, chấp nhận đăng: 03/2023.

thuê nghệ sĩ vẽ tranh, mạng GAN có thể hỗ trợ cách làm việc hiệu quả hơn nghệ sĩ vẽ tranh trong lĩnh vực này.

Trong công nghiệp thời trang, để thăm dò thị trường, thử thị hiếu người tiêu dùng trước đối với các sáng tạo mẫu mã quần áo mới, GAN được áp dụng để sinh ra hình ảnh mẫu mã mới mà không cần làm mẫu sản phẩm trước, giúp tiết kiệm chi phí và có nhiều mẫu mã thăm dò mới. Sohn và cộng sự [13] đã xem xét đánh giá của người tiêu dùng về giá trị tiêu dùng sản phẩm, ý định mua hàng và mức độ sẵn sàng chi trả cho các sản phẩm thời trang được thiết kế bằng cách sử dụng mạng GAN. Nghiên cứu này khám phá sự khác biệt giữa đánh giá của người tiêu dùng về sản phẩm tạo ra bởi GAN và sản phẩm không tạo ra bởi GAN và kiểm tra xem việc sử dụng công nghệ GAN có ảnh hưởng đến đánh giá của người tiêu dùng hay không.

GAN là một trong những xu hướng nghiên cứu thu hút được đông đảo các nhà khoa học, có tính ứng dụng cao và phát triển mạnh mẽ trong những năm gần đây trong ứng dụng kỹ thuật học sâu. Mô hình GAN gần đây đã đạt được một số kết quả ấn tượng cho nhiều ứng dụng trong thế giới thực và nhiều biến thể GAN đã xuất hiện với những cải tiến về chất lượng mẫu và độ ổn định đào tạo. GAN được ứng dụng rộng rãi trong các bài toán thực tế, cho cả bài toán phân loại và hồi quy, cho kết quả tốt. Bên cạnh kiến trúc GAN đầu tiên được Ian GoodFellow giới thiệu vào năm 2014, trong những năm qua, một số kiến trúc GAN nâng cao hơn đã ra đời, đưa lại nhiều lợi ích cho các ứng dụng thực tế như CycleGAN, StyleGAN...

II. CÁC NGHIÊN CỨU ĐÃ ỨNG DỤNG GAN

Thực tế đã có rất nhiều nghiên cứu sử dụng mô hình GAN trên thế giới. Một trong số đó là nghiên cứu do Islam và cộng sự [2] thực hiện đã cho thấy mô hình GAN làm việc hiệu quả với bộ dữ liệu hình ảnh y tế với tập dữ liệu hạn chế và số lượng nhỏ các mẫu. Điều này chỉ ra tiềm năng sử dụng GAN để phát triển một mô hình chẩn đoán bệnh thông qua các hình ảnh y tế tổng hợp bằng cách sử dụng mạng GAN. Năm 2016, Olof đã ứng dụng mạng GAN để đào tạo hiệu quả các mạng nơron học sâu cho âm nhạc [3]. Đây là một mô hình đối nghịch hoạt động dựa trên dữ liệu tuần tự liên tục và áp dụng GAN bằng cách đào tạo trên một bộ dữ liệu âm nhạc cổ điển. Nghiên cứu kết luận rằng GAN tạo ra âm thanh ngày càng hay hơn khi mô hình được đào tạo và cho phép người nghe đánh giá chất lượng bằng cách tải xuống các bài hát đã tạo.

Tero và cộng sự [4] đã tạo ra khuôn mặt người thông qua mô hình GAN. Trong đó có khả năng tạo ra những khuôn mặt nhân tạo mà rất khó phân biệt với người thật. Nghiên cứu của Antipov [5] đã sử dụng mạng GAN để tạo ra hình ảnh tổng hợp có độ trung thực cao. Trong nghiên cứu đó đã đề xuất phương pháp dựa trên GAN để tự động hóa quá trình lão hóa khuôn mặt [5]. Kết quả nghiên cứu đã đánh giá khách quan các hình ảnh khuôn mặt trẻ hóa và già nua thu được bằng các giải pháp ước tính độ tuổi và nhận dạng khuôn mặt cho thấy tiềm năng cao của phương pháp được đề xuất dựa trên mô hình GAN. David và cộng sự [6] đã nghiên cứu sâu mạng GAN để hình dung hoặc hiểu rõ xử lý các thể hiện thế giới hình ảnh bên trong và nguyên nhân tạo ra kết quả mạng GAN, các lựa chọn kiến trúc ảnh hưởng đến việc học GAN. Từ đó giúp phát triển những hiểu biết sâu sắc mới và các mô hình GAN tốt hơn. Trong nghiên cứu này, các tác giả đã trình bày một khung phân tích để hình dung và hiểu các tác động GAN ở cấp độ đơn vị, đối tượng và cảnh, xác định một nhóm các đơn vị có thể diễn giải có liên quan chặt chẽ đến các khái niệm đối tượng bằng cách sử dụng phương pháp phân tích mạng dựa trên phân đoạn. Sau đó, phân tích định lượng tác động nhân quả của các đơn vị có thể diễn giải bằng cách đo lường khả năng can thiệp kiểm soát các đối tượng trong đầu ra. Kết quả nghiên cứu hiển thị một số ứng dụng thực tế được hỗ trợ bởi khung công tác trên các đơn vị, từ việc so sánh các biểu diễn nội bộ trên các lớp, mô hình và bộ dữ liệu khác nhau, đến cải thiện GAN bằng cách định vị và loại bỏ các đơn vị gây nhiễu, đến thao tác tương tác các đối tượng trong một khung ảnh. Jianmin và cộng sự [7] đã ứng dụng GAN để tổng hợp hình ảnh khuôn mặt của một người hoặc các đối tượng cụ thể trong một danh mục và sử dụng đối sánh đặc điểm theo từng cặp để giữ cấu trúc của hình ảnh được tạo. Các tác giả thử nghiệm với các hình ảnh tự nhiên về khuôn mặt, hoa và chim, và chứng minh rằng các mô hình được đề xuất có khả năng tạo ra các mẫu thực tế và đa dạng với các nhãn danh mục chi tiết. Kết quả nghiên cứu còn cho thấy rằng các mô hình GAN có thể được áp dụng cho các tác vụ, chẳng hạn như in hình ảnh, độ phân giải siêu cao và dữ liệu tăng cường đào tạo mô hình nhận dạng khuôn mặt tốt hơn.

Marra và cộng sự [8] đã sử dụng GAN để giải quyết bài toán lan truyền của các hình ảnh và video giả trên mạng xã hội. Các tác giả đã sinh hình ảnh nhờ các hình ảnh khác, dựa trên kiến trúc mạng đối nghịch GAN. Trong nghiên cứu đó, các tác giả đã nghiên cứu hiệu suất của một số bộ phát hiện giả mạo hình ảnh chống lại quá trình biến đổi từ ảnh sang ảnh, cả trong điều kiện lý tưởng và khi ảnh bị nén, được thực hiện thường xuyên khi tải lên mạng xã hội. Nghiên cứu, được thực hiện trên tập dữ liệu gồm 36.302 hình ảnh, cho thấy độ chính xác phát hiện lên đến 95% có thể đạt được bằng cả kỹ thuật học máy truyền thống và học sâu, nhưng chỉ kỹ thuật học sâu mới cung cấp độ chính xác cao, khoảng 89% trên dữ liệu ảnh bị nén.

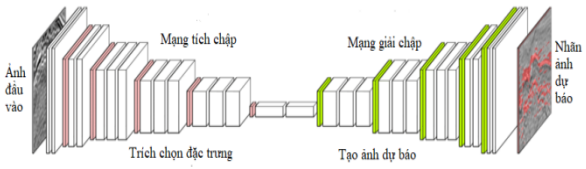
Nghiên cứu trên các bức ảnh chụp phổi bệnh nhân Covid-19 sử dụng mô hình GAN đã cho thấy tiềm năng chẩn đoán hình ảnh X-Quang bệnh nhân nhiễm Covid-19. Nghiên cứu [9] mở ra hướng mới trong ứng dụng trí tuệ

nhân tạo trong chẩn đoán hình ảnh chụp phổi các bệnh nhân Covid-19.

III. MÔ HÌNH LÝ THUYẾT MẠNG GAN

Dựa trên cơ sở lý thuyết mô hình GAN của Goodfellow và cộng sự đưa ra năm 2014 [1], bài toán này áp dụng cho tập dữ liệu hình ảnh Pokemon. Vì vậy, mô hình GAN áp dụng thử nghiệm sử dụng mạng nơ ron tích chập (Convolutional Neural Network) cho Discriminator và giải tích chập (Deconvolutional Neural Network) cho Generator (Hình 2). Mô hình GAN sẽ có hai bộ phận chính: Generator (G) và Discriminator (D). Trong đó, G có chức năng sinh ra ảnh Pokemon giả và D làm nhiệm vụ phân biệt một bức ảnh Pokemon là ảnh thật hay ảnh giả. Về bản chất, Generator học cách sinh ra dữ liệu giả để lừa mô hình Discriminator. Để có thể đánh lừa được Discriminator thì đòi hỏi mô hình sinh ra các bức ảnh Pokemon phải thực sự tốt. Do đó chất lượng ảnh Pokemon phải càng như thật càng tốt. Trong khi đó, Discriminator sẽ học cách phân biệt giữa dữ liệu ảnh Pokemon giả được sinh từ mô hình Generator với dữ liệu thật. Discriminator như một bộ lọc giám sát đánh giá kết quả của Generator xem liệu mạng nơ ron này đã sinh dữ liệu đã đạt chất lượng tốt để qua đánh lừa được Discriminator chưa và nếu chưa thì Generator cần tiếp tục phải học để tạo ra ảnh Pokemon giống ảnh thật hơn. Đồng thời, Discriminator cũng phải cải thiện khả năng phân biệt của mình vì chất lượng ảnh được tạo ra từ Generator càng ngày càng giống thật hơn. Thông qua quá trình huấn luyện thì cả Generator và Discriminator cùng cải thiện được khả năng của mình.

Generator và Discriminator tương tự như hai người chơi trong bài toán trò chơi tổng bằng không trong lý thuyết trò chơi. Ở trò chơi này thì hai người chơi xung đột lợi ích, thiệt hại của người này chính là lợi ích của người kia. Mô hình Generator tạo ra dữ liệu giả tốt hơn sẽ làm cho Discriminator phân biệt khó hơn và khi Discriminator phân biệt tốt hơn thì Generator cần phải tạo ra ảnh giống thật hơn để Discriminator không phát hiện được. Trong trò chơi tổng bằng không, mỗi người chơi sẽ có chiến lược riêng của mình, đối với Generator thì đó là sinh ra ảnh giống thật và Discriminator là phân loại chính xác ảnh thật (real) và ảnh giả (fake). Sau các bước ra quyết định của mỗi người chơi thì trò chơi tổng bằng không sẽ đạt được cân bằng Nash tại điểm cân bằng (Nash Equilibrium). Quá trình hoạt động của mạng là một quá trình huấn luyện các trọng số cho mô hình 2 lớp trong một trò chơi có tổng bằng không [12]. Trong quá trình truyền nghịch, một lớp được dùng để tạo ra dữ liệu giả giống hệt như dữ liệu thật. Trong khi đó, lớp còn lại là lớp kiểm tra đóng vai trò đánh giá để phân biệt dữ liệu thật và dữ liệu giả được tạo ra. Hình 2 minh họa kiến trúc của mạng GAN sử dụng mạng tích chập cho Discriminator và mạng giải chập cho Generator [14].



Hình 2: Mô hình mạng GAN [14]

Generator về bản chất là một mô hình sinh nhận đầu vào là một tập hợp các véc tơ nhiễu z được khởi tạo ngẫu nhiên theo phân phối Gaussian [1]. Từ tập véc tơ đầu vào z ngẫu nhiên, mô hình Generator là một mạng học sâu có tác dụng biến đổi ra bức ảnh giả ở đầu ra. Bức ảnh giả này sẽ được sử dụng làm đầu vào cho kiến trúc Discriminator.

Để xác định cách phân phối P_g của Generator trên dữ liệu x, cần xác định biến đầu vào $p_z(z)$, sau đó biểu diễn một ánh xạ tới không gian dữ liệu dưới dạng $G(z; \theta_g)$, trong đó G là một hàm khả vi được đại diện bởi một mạng nơ ron nhiều lớp (mạng giải chập) với các tham số θ_g. Đồng thời cũng xác định một mạng nơ ron nhiều lớp thứ hai $D(x; \theta_d)$ với đầu ra là một đại lượng vô hướng duy nhất. D(x) đại diện cho xác suất x đến từ dữ liệu, không phải P_g. Thực hiện huấn luyện D để tối đa hóa xác suất gán đúng nhãn cho cả dữ liệu đào tạo và mẫu từ G. Đồng thời đào tạo G để giảm thiểu lỗi:

$$L = \log (1 - D(G(z))) \quad (1)$$

Nói cách khác, D và G đóng vai trò là 2 người chơi trò chơi đối nghịch nhau, cả G và D đều cố gắng tối ưu giá trị của hàm. Định nghĩa sự tối ưu giá trị của V (G, D) như sau:

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log (1 - D(G(z)))] \quad (2)$$

Mô hình Discriminator sẽ có tác dụng phân biệt ảnh đầu vào là thật hay giả. Nhãn của mô hình sẽ là thật nếu ảnh đầu vào của Discriminator được lấy tập mẫu huấn luyện và là giả nếu được lấy từ đầu ra của mô hình Generator. Về bản chất đây là một bài toán phân loại nhị phân (binary classification) thông thường nên để tính xác suất cho đầu ra cho Discriminator sẽ sử dụng hàm Sigmoid. Mục tiêu của pha huấn luyện Discriminator này là huấn luyện một mô hình Discriminator sao cho khả năng phân loại là tốt nhất. Ở pha này tạm thời coi các giá trị trọng số của G không đổi và chỉ quan tâm đến về maxDV(D, G). Đây là chính là đối của hàm cross entropy đối với trường hợp phân loại nhị phân. Mục tiêu của hồi quy logistic đối với bài toán phân loại nhị phân là tối thiểu hóa một hàm cross entropy như sau:

$$\mathcal{L}(W; X, y) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log p(y_i/x_i) + (1 - y_i)\log (1 - p(y_i/x_i))] \quad (3)$$

Trong đó $p(y_i/x_i)$ là xác suất dự báo nhãn y_i từ mô hình logistic, và D(x) cũng đóng vai trò như $p(y_i/x_i)$. Nói cách khác 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 (X_{real}) thì $y_i = 1$ và $1 - y_i = 0$ và do đó hàm mất mát tương ứng với: $y_i \log(p(y_i/x_i))) = \log(p(y_i/x_i))$ ở công thức (3). Giá trị này được coi như là logD(x) ở công thức (2). Ký hiệu $x \sim p_{data}(x)$ ở công thức (2) thể hiện là phân phối xác suất của các điểm dữ liệu đầu vào. Trong trường hợp ở công thức (3) thì các quan sát có vai trò như nhau nên có chung giá trị phân phối là $\frac{1}{N}$. Trường hợp ảnh đầu vào là giả (X_{fake}) thì $y_i = 0$ và $1 - y_i = 1$, khi đó đóng góp vào hàm mất mát chỉ còn thành phần $(1 - y_i) \log (1 - p(y_i/x_i)) = \log (1 - p(y_i/x_i))$ ở công thức (3). Giá trị này được coi như là $\log (1 - D(G(z)))$ ở công thức (2).

Tại pha huấn luyện Generator, mục tiêu của pha này là tăng cường khả năng tạo ảnh của Generator sao cho ảnh sinh ra là giống với thật nhất. Ở pha này coi D có giá trị không đổi và chỉ quan tâm đến G(z) sao cho giá trị dự báo xác suất từ D đối với ảnh 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. Áp dụng đảo dấu trong $E_{z \sim p_z(z)}[\log (1 - D(G(z)))]$ suy ra mục tiêu cần tối ưu là tối thiểu hóa giá trị $\min_G V(D,G)$.

Quá trình huấn luyện thực hiện theo thuật toán 1 của mô hình GAN của Goodfellow [1]. Theo đó, trong quá trình huấn luyện sẽ kết hợp một cách xen kẽ giữa hai pha, sử dụng phương pháp mini-batch stochastic giảm độ dốc để huấn luyện GAN. Số lượng các bước (steps) được áp dụng cho Discriminator, với k là siêu tham số. Quá trình huấn luyện bắt đầu với k = 1 để có chi phí lựa chọn thấp nhất trong mô hình.

for number of training iterations **do**

for k steps **do**

*) Lấy mẫu minibatch của m mẫu nhiễu $\{z^{(1)}; z^{(2)};\ldots; z^{(m)}\}$.

*) Lấy mẫu minibatch của m mẫu dữ liệu thật $\{x^{(1)}; x^{(2)};\ldots; x^{(m)}\}$ từ phân bố sinh $p_{data}(x)$.

*) Cập nhật Discriminator theo phương pháp mini-batch giảm độ dốc:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log (1 - D(G(z^{(i)})))] \qquad (4)$$

end for

Lấy mẫu minibatch của m mẫu nhiễu $\{z^{(1)}; z^{(2)};\ldots; z^{(m)}\}$.

Cập nhật generator bằng cách giảm độ dốc stochastic:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log (1 - D(G(z^{(i)}))) \qquad (5)$$

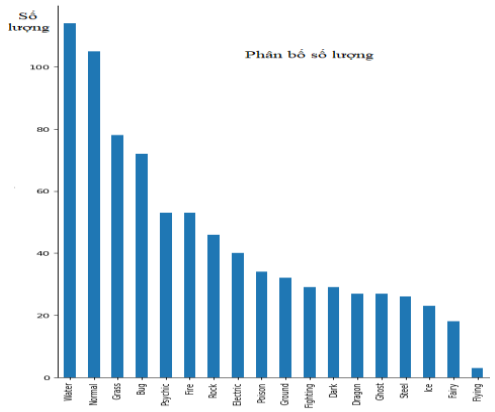
end for

Phương pháp huấn luyện là sử dụng k batch đầu tiên để huấn luyện Discriminator. Khi huấn luyện Discriminator, thực hiện lấy mẫu một mini-batch kích thước m là các nhiễu $z^{(1)}, z^{(2)},\ldots, z^{(m)}$ và là đầu vào của Generator. Đồng thời lấy mẫu một mini-batch khác kích thước m là những điểm dữ liệu thật $x^{(1)}, x^{(2)},\ldots, x^{(m)}$. Những dữ liệu này sẽ được sử dụng để cập nhật giảm độ dốc theo phương pháp mini-batch giảm độ dốc. Do là huấn luyện trên mô hình Discriminator nên chỉ được cập nhật các hệ số trên mô hình Discrimator là θ. Thuật toán cập nhật bộ tham số dựa trên giảm độ dốc có thể sử dụng bất kỳ quy luật học nào, kết hợp sử dụng đà (momentum). Các hệ số của Generator được đóng băng, không thay đổi. Sau khi kết thúc k batch huấn luyện trên Discriminator, sẽ tiếp tục huấn luyện trên Generator. Khi đó, một mini-batch kích thước m được lựa chọn ra từ các nhiễu là $z^{(1)}, z^{(2)}, \ldots, z^{(m)}$ được sử dụng như đầu vào huấn luyện. Giá trị giảm độ dốc sẽ được tính trên m dữ liệu này. Quá trình cập nhật giảm độ dốc chỉ được áp dụng trên các hệ số của Generator là θ_G. Tiếp tục quá trình này cho tới khi tổng số lượt huấn luyện là đủ lớn hoặc giá trị mất mát của mô hình tiệm cận về 0.

IV. THỰC NGHIỆM VÀ KẾT QUẢ

1. Mô tả dữ liệu sử dụng cho thử nghiệm

Bài báo này sử dụng bộ dữ liệu Pokemon gồm 801 hình ảnh chia ra 18 loại các nhân vật Pokemon được tải về từ https://veekun.com/dex/downloads. Phân bố thống kê các loại nhân vật Pokemon như hình 3.



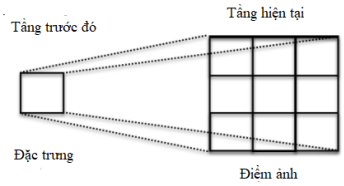
Hình 4: Phân bố dữ liệu Pokemon

Hình 3: Dữ liệu Pokemon thực nghiệm

Dữ liệu ví dụ hình ảnh của một số nhân vật Pokemon điển hình được hiển thị trong hình 4.

2. Mô tả quá trình thử nghiệm

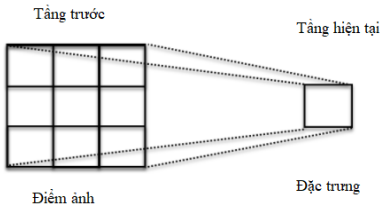
Nghiên cứu này không đề xuất mô hình hay tham số mới, chủ yếu áp dụng tập dữ liệu Pokemon cập nhật của Veekun vào mô hình GAN sử dụng thư viện PyTorch. Để làm rõ hơn các nội dung mạng GAN khi áp dụng bộ dữ liệu Pokemon, bài toán này sử dụng mô hình mạng giải chập (Deconvolutional Neural Network) cho phần Generator. Trong đó, các lớp mạng có kích thước giảm dần qua các tầng để cuối cùng thu được những đặc trưng bậc cao, để sinh hình ảnh Pokemon từ các đặc trưng này. Một mạng giải chập sẽ có kiến trúc chung là kích thước các tầng sẽ tăng dần qua các lớp [Hình 2]. Qua từng lớp mạng sẽ giải mã các khối đặc trưng thành những thông tin không gian của từng điểm ảnh và cuối tạo thành một bức ảnh mới ở đầu ra của mô hình [Hình 5].



Hình 5: Mạng giải chập

Để lập trình, qúa trình tăng kích thước tại các tầng sử dụng tích chập chuyển vị (Transposed Convolution 2D) trong thư viện PyTorch. Đầu vào cho bộ Generator là một ma trận các số ngẫu nhiên (được gọi là latent tensor) có latent_size = 128. Bộ Generator sẽ chuyển đổi một latent tensor có kích thước (128, 1, 1) thành một tensor hình ảnh có kích thước 3 x 28 x 28. Để đạt được điều này, sử dụng lớp ConvTranspose2d trong thư viện PyTorch với các siêu tham số kernel_size = 4, stride = 1, padding = 0, bias = False. Bài toán này thử nghiệm Generator có 5 lớp mạng, với hàm kích hoạt ReLU cho 4 lớp đầu và hàm Tanh cho lớp cuối cùng. Về bản chất, quá trình giải chập thực hiện biến đổi ngược, giả sử từ ma trận đầu vào có kích thước (w_1, h_1) sau khi áp dụng phép tích chập thông thường thu được kích thước (w_2, h_2). Tích chập chuyển vị sẽ biến đổi từ một ma trận có kích thước (w_2, h_2) của kết qủa đầu ra sang ma trận có kích thước (w_1, h_1) của giá trị đầu vào trong khi vẫn duy trì được các kiểu kết nối phù hợp với tích chập. Bộ Generator sử dụng hàm kích hoạt ReLU cho các đầu ra của các tầng để khử tính tuyến tính. Hàm ReLU giúp lọc các giá trị âm và cho tốc độ hội tụ và tính toán nhanh hơn. Trong khi đó, ở tầng đầu ra sử dụng hàm Tanh để chuyển đầu vào thành một giá trị trong khoảng (-1, 1). Qua chạy dữ liệu thử nghiệm, kết quả cho thấy việc sử dụng hàm kích hoạt Tanh có giới hạn cho phép mô hình học nhanh hơn.

Trong bộ Discriminator, sử dụng mạng tích chập (Convolutional Neural Network) có đầu vào là một hình ảnh và nhiệm vụ của Discriminator là xác định hình ảnh đó là thật (real images) hoặc ảnh giả được tạo ra bởi Generator (ảnh fake images). Mạng tích chập giúp giảm dần kích thước qua các tầng để kết xuất các đặc trưng [Hình 6].



Hình 6: Mạng tích chập

Để tính giá trị tích chập, di chuyển các điểm ảnh của ma trận đầu vào từ trái qua phải và từ trên xuống dưới. Sau đó lấy giá trị của điểm ảnh nhân với ma trận bộ lọc sẽ thu được ma trận đầu ra. Trong bài toán nhận diện ảnh Pokemon này, nghiên cứu thử nghiệm sử dụng thư viện PyPorch với lớp Conv2d để áp dụng cho cá ảnh Pokemon. Bài toán này thử nghiệm Discriminator có 5 lớp mạng, với hàm kích hoạt LeakyReLU có độ dốc bằng 0.2 cho 4 lớp đầu và hàm Sigmoid cho lớp cuối cùng. Các siêu tham số được lựa chọn gồm kernel_size = 4, stride = 2, padding = 1, bias = False. Chọn bước nhảy (stride = 2) và lát thêm 1 (padding = 1) để mở rộng ma trận sao cho quá trình tích chập không bị dư thừa. Quá trình di chuyển kết quả của mỗi lần nhân điểm ảnh với bộ lọc sang 2 đơn vị mỗi lần tính. Sau cùng thực hiện tính tổng các vị trí tương ứng của các ma trận kết quả để thu được kết quả cuối cùng. Trong trường hợp này, giá trị bước nhảy không bằng kích thước ma trận bộ lọc nên ma trận kết quả tích chập sẽ ghi chồng lên nhau, do đó giá trị được cộng dồn. Thử nghiệm này có sử dụng BatchNorm2d để chuẩn hóa các tính năng dọc theo thứ nguyên của lô để nhận được kết quả phù hợp. Discriminator thử nghiệm với hàm kích hoạt LeakyReLU với độ dốc bằng 0,2. Mục đích sử dụng hàm LeakyReLU để thay vì trả về giá trị 0 với các đầu vào âm như hàm ReLU thì LeakyReLU tạo ra một đường xiên có độ dốc nhỏ (ở đây chọn độ dốc bằng 0,2) để hàm kích hoạt hoạt động tốt đối với mô hình có độ phân giải cao. Kết quả là làm cho các độ dốc từ bộ Discriminator truyền tốt hơn vào bộ Generator. Thay vì vượt qua một độ dốc bằng 0 sẽ vượt qua một độ dốc âm giá trị nhỏ. Ở tầng đầu ra, sử dụng hàm Sigmoid cho giá trị ở giữa 0 và 1. Bởi vì Discriminator là mô hình phân loại nhị phân, là xác suất của hình ảnh đầu vào là thật (real), tức là được chọn từ tập dữ liệu thật ban đầu hoặc là ảnh giả (fake) được tạo ra bởi Generator.

Về quá trình huấn luyện, Discriminator hoạt động để phân biệt đâu là ảnh thật và đâu là ảnh giả, giá trị đầu ra của mô hình qua hàm sigmoid có giá trị trong khoảng (0, 1). Vì vậy, Discriminator sẽ được huấn luyện để nếu đầu vào ảnh là dữ liệu thật thì đầu ra ở gần giá trị 1, còn nếu đầu vào ảnh là dữ liệu giả là ảnh sinh ra từ Generator thì giá trị đầu ra gần 0. Nói cách khác, giá trị D(x) tiến về 1

còn giá trị D(G(z)) tiến về 0. Trong quá trình huấn luyện, trước tiên, chuyển một loạt hình ảnh thực và tính toán giá trị mất mát, đặt giá trị nhãn mục tiêu thành 1. Sau đó, chuyển các ảnh giả (được tạo bằng Generator) đưa vào Discriminator và tính giá trị mất mát, đặt giá trị nhãn mục tiêu thành thành 0. Cuối cùng, cộng hai giá trị mất mát và sử dụng giá trị mất mát chung (overall loss) để thực hiện giảm độ dốc để điều chỉnh trọng số (weights) của Discriminator. Quá trình này cần đóng băng, không thay đổi các trọng số (weights) của Generator trong quá trình huấn luyện Discriminator.

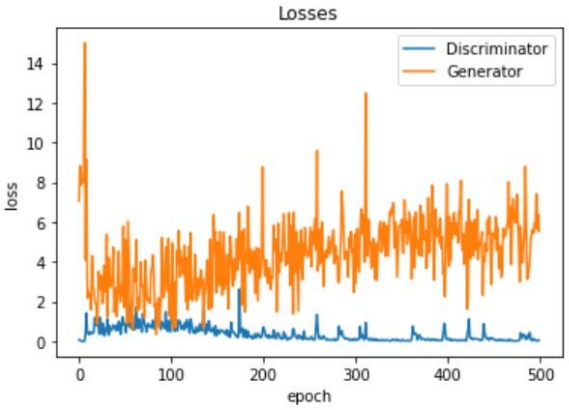
Đối với huấn luyện Generator, vì đầu ra của Generator là hình ảnh, nên sử dụng Discriminator như một phần của hàm mất mát. Theo đó, trước hết cần tạo ra các hình ảnh giả bằng cách sử dụng Generator, chuyển vào Discriminator. Sau đó, thực hiện tính toán giá trị mất mát bằng cách đặt các giá trị nhãn mục tiêu thành 1, tức là ảnh thật (real), bởi vì mục tiêu của Generator là đánh lừa Discriminator, ảnh giả trông như ảnh thật. Sau đó sử dụng giá trị mất mát để thực hiện giảm độ dốc, tức là thay đổi trọng số (weights) của Generator. Do đó, sẽ tăng khả năng tạo ra hình ảnh giống như thật để có thể đánh lừa Discriminator.

Do quá trình huấn luyện mô hình Generator và mô hình Discriminator đối nghịch nhau, trong khi Discriminator cố gắng tối đa hóa mát mát thì Generator cố gắng tối thiểu hóa. Quá trình huấn luyện mạng GAN kết thúc khi mô hình GAN đạt đến trạng thái cân bằng của 2 mô hình, gọi là cân bằng Nash (Nash Equilibrium). Tiêu chí khả năng nhận dạng đúng ảnh thật hay giả để đo lường khả năng dự đoán do bộ Discriminator đưa ra, vì vậy sử dụng BCE Loss (Binary Cross Entropy Loss).

Thử nghiệm này sử dụng công cụ tối ưu hóa Adam từ mô-đun tối ưu của thư viện PyTorch (*torch.optim.Adam*), là công cụ tối ưu phù hợp cho bài toán. Bởi vì Adam là sự kết hợp của Momentum và RMSprop, cho kết quả hội tụ nhanh hơn.

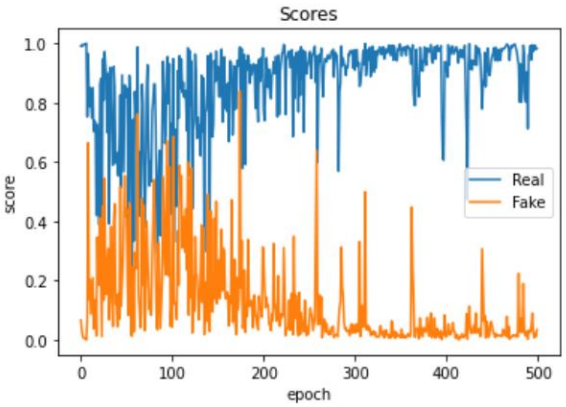
Thử nghiệm này chọn tốc độ học learning rate = 0.0002, momentum = 0.5, số lần lặp epochs = 500, và batch_size = 64.

Kết qủa thử nghiệm cho thấy mô hình GAN cho kết quả khá trên bộ dữ liệu Pokemon. Giá trị mất mát của Discriminator giảm dần, còn của Generator ổn định như hình 7 bên dưới.



Hình 7: Đồ thị hàm mất mát

Để xác định mức độ ổn định của mô hình, hình 8 bên dưới đã cho thấy, đối với ảnh thật thì giá trị D(x) tiến về 1 còn giá trị D(G(z)) tiến về 0 theo yêu cầu bài toán đã đặt ra.



Hình 8: Đồ thị giá trị D(x) và D(G(z))

Hình 9 dưới đây mô tả một số hình ảnh giả (fake) được tạo ra bởi mô hình GAN dựa trên bộ dữ liệu Pokemon. Tuy chất lượng ảnh Pokemon không được tốt như ảnh thật ở Hình 2 phía trên, nhưng điều này cũng cho thấy tiềm năng ứng của ứng dụng mạng GAN cho sinh ảnh nhân vật Pokemon.

Hình 9: Hình ảnh Pokemon sinh bởi GAN

Để có tham số định lượng đánh giá xem chất lượng ảnh sinh ra đã tốt chưa, mà không bị phụ thuộc cảm nhận mắt thường để đánh giá, mạng GAN sử dụng chỉ số Inception Score (IS) [15]. Theo đó, IS được tính bằng cách sử dụng mạng Inception V3 của Google đã được huấn luyện trên tập dữ liệu ảnh ImageNet. Sử dụng Inception nguyên bản, không tinh chỉnh tham số fine-tune. Các bức ảnh Pokemon sinh ra bởi GAN là ảnh đầu vào của mạng Inception và đầu ra đưa qua hàm softmax để kết xuất được giá trị xác suất ảnh đấy thuộc lớp nào trong 18 lớp ảnh Pokemon.

IS đánh giá 2 thuộc tính của tập hợp các hình ảnh được tạo ra: Chất lượng hình ảnh, có giống một đối tượng cụ thể không? Đa dạng hình ảnh, các ảnh được tạo ra có độ phủ rộng, đa dạng? IS dùng Kullback–Leibler Divergence (KLD) để đánh giá và nếu mạng GAN sinh ra ảnh tốt, tức chất lượng ảnh tốt và đa dạng thì giá trị KLD sẽ cao và ngược lại.

Kết quả đo IS trên tập ảnh Pokemon sinh ra bởi mạng GAN sử dụng thư viện *pytorch_gan_metrics* cho giá trị IS = 2.2936 và độ lệch chuẩn IS_std = 0.3265. Giá trị này là thấp so với các kết qủa thử nghiệm khác (ví dụ kết quả thử nghiệm trên tập ảnh CIFAR-10 dataset là IS = 8.09). Để giải thích kết quả IS thấp này, nghiên cứu này có thử nghiệm chỉ số đo IS với bộ dữ liệu Pokemon nguyên bản gốc (không phải ảnh sinh ra bởi GAN) thì giá trị IS = 3.2631 và IS_std = 0.4095.

Như vậy, ngay cả đối với bộ dữ liệu Pokemon gốc cũng không cho giá trị IS cao như mong muốn. Điều này có thể lý giải là do Inception V3 được huấn luyện trên tập dữ liệu ảnh ImageNet và tập dữ liệu ảnh này chủ yếu gồm các bức ảnh tự nhiên, ảnh con người, có thể không chứa các bức ảnh nhân vật hoạt hình Pokemon nên khả năng nhận dạng đối với dữ liệu Pokemon chưa cao.

Tuy vậy, kết quả nghiên cứu cho thấy có tiềm năng trong sử dụng mạng GAN phục vụ bài toán sinh dữ liệu ảnh Pokemon nói riêng và các dữ liệu đa phương tiện khác nói chung trong quá trình phát triển các ứng dụng Games, thực tế ảo (VR), thực tế ảo tăng cường (AR), thực tế hỗn hợp (MR), thực tế ảo mở rộng Extended Reality (XR)…

V. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Nghiên cứu này đã thử nghiệm mô hình GAN (Generative Adversarial Networks) cho bài toán sinh dữ liệu ảnh đa phương tiện sử dụng bộ dữ liệu hình ảnh trò chơi Pokemon của Nhật Bản. Mô hình thử nghiệm sử dụng mạng nơ ron tích chập (Convolutional Neural Network) cho phần phân biệt (Discriminator) và mạng nơ ron giải chập (Deconvolutional Neural Network) cho phần sinh (Generator). Kết quả thử nghiệm cho thấy, mô hình GAN cho kết quả có tiềm năng, gợi ý một hướng giải quyết bài toán sinh dữ liệu đa phương tiện để ứng dụng cho các bài toán Games, hoạt hình, đồ họa,…

Hướng phát triển các nghiên cứu tiếp theo có thể thử nghiệm các tập dữ liệu đa phương tiện khác (âm thanh, video, tập ảnh khác…) và sử dụng các mô hình biến thể GAN khác như CycleGAN, StyleGAN, BigGAN, Wasserstein GAN, StarGAN, SRGAN,…

TÀI LIỆU THAM KHẢO

[1] Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., and Bengio Y., “Generative Adversarial Nets”, Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

[2] Islam, J., Zhang, Y. GAN-based synthetic brain PET image generation. Brain Inf. 7, 3 (2020). https://doi.org /10.1186 /s40708-020-00104-2.

[3] Olof M. (2016) “C-RNN-GAN: Continuous recurrent neural networks with adversarial training”, Constructive Machine Learning Workshop (CML) at NIPS 2016 in Barcelona, Spain. https://doi.org/10.48550/arXiv.1611.09904

[4] Tero K., Timo A., Samuli L., Jaakko L. (2017) “Progressive Growing of GANs for Improved Quality, Stability, and Variation”, Neural and Evolutionary Computing, https:// doi.org /10.48550 /arXiv.1710.10196

[5] G. Antipov, M. Baccouche and J. Dugelay, “Face aging with conditional generative adversarial networks”, 2017 IEEE International Conference on Image Processing (ICIP), 2017, pp. 2089-2093, doi: 10.1109/ICIP.2017.8296650.

[6] David B., Jun-Yan Z., Hendrik S., Bolei Z., Joshua B. T., William T. F., Antonio T. (2018) “GAN Dissection: Visualizing and Understanding Generative Adversarial Networks”, Computer Vision and Pattern Recognition, https:/ /doi.org/10.48550/arXiv.1811.10597

[7] Jianmin B., Dong C., Fang W., Houqiang L., Gang H. (2017) “CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training”, 2018 IEEE Conference on Computer Vision and Pattern Recognition(CVPR 2018), https://doi.org /10.48550/arXiv.1803.11182.

[8] F. Marra, D. Gragnaniello, D. Cozzolino and L. Verdoliva, "Detection of GAN-Generated Fake Images over Social Networks," 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018, pp. 384-389, doi: 10.1109/MIPR.2018.00084.

[9] A. Waheed, M. Goyal, D. Gupta, A. Khanna, F. Al-Turjman and P. R. Pinheiro, "CovidGAN: Data Augmentation Using Auxiliary Classifier GAN for Improved Covid-19 Detection," in IEEE Access, vol. 8, pp. 91916-91923, 2020, doi: 10.1109/ACCESS.2020.2994762.

[10] Yanghua J., Jiakai Z., Minjun L., Yingtao T., Huachun Z., Zhihao F., “Towards the Automatic Anime Characters Creation with Generative Adversarial Networks”, Computer Vision and Pattern Recognition, https://doi.org/10.48550 /arXiv.1708.05509

[11] Vincent D., Francesco V., “A guide to convolution arithmetic for deep learning”, Machine Learning, 2018, https://doi.org /10.48550/arXiv.1603.07285

[12] Matthew D. Z., Rob F., “Visualizing and Understanding Convolutional Networks”, 2013, Computer Vision and Pattern Recognition, https://doi.org/10.48550/arXiv.1311.2901

[13] Sohn, K., Sung, C.E., Koo, G. and Kwon, O. (2021), "Artificial intelligence in the fashion industry: consumer responses to generative adversarial network (GAN) technology", International Journal of Retail & Distribution Management, Vol. 49 No. 1, pp. 61-80. https://doi.org/10.1108/IJRDM-03-2020-0091

[14] Alaudah, Yazeed & Michałowicz, Patrycja & Alfarraj, Motaz & Alregib, Ghassan: A Machine Learning Benchmark for Facies Classification. Interpretation, Volume 7, Issue 3, pp. 1A-T725 (2019), https://doi.org/10.1190/INT-2018-0249.1

[15] Tim S., Goodfellow I., Wojciech Z., Vicki C., Alec R., Xi C., "Improved Techniques for Training GANs", Machine Learning, Computer Vision and Pattern Recognition, Neural and Evolutionary Computing, 2016, https://doi.org/10.48550 /arXiv.1606.03498.

APPLIED GAN (GENERATIVE ADVERSARIAL NETWORKS) MODEL INTO MULTIMEDIA DATA GENERATION

Abstract: The characters of Pokemon images are artificial, animated, so they are different from normal image processing problems. This study tests and evaluates the responsiveness of the GAN (Generative Adversarial Networks) model to the image set of Pokemon characters. The model uses a convolutional neural network for the discriminator and a deconvolutional neural network for the Pokemon character image data generator. The test dataset is the Veekun image set. Experimental results show that, even though it is a game image, the GAN model is still quite suitable and has potential for application to the problem of multimedia data generation.

Keywords: GAN, Pokemon, generator, discriminator, real image, fake image.

Trần Quý Nam tốt nghiệp Kỹ sư (năm 1999) và Thạc sỹ (năm 2005) ngành công nghệ thông tin hệ chính quy tại Đại học Bách Khoa Hà Nội, nhận bằng Tiến sỹ quản lý công nghệ thông tin tại Đại học Quốc gia Xê-Un Hàn Quốc năm 2010. Lĩnh vực nghiên cứu: học máy, học sâu, chính phủ điện tử, chuyển đổi số, hiệu năng hệ thống thông tin, ứng dụng trí tuệ nhân tạo, phát triển ứng dụng đa phương tiện.