ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

BÁO CÁO ĐỒ ÁN
XỬ LÍ ẢNH VÀ ỨNG DỤNG

IMAGE COLORIZATION

USING GAN MODELS & CNN

GV hướng dẫn: Cáp Phạm Đình Thăng

Nhóm thực hiện:

1. Võ Đức Dương – 21521992
2. Nguyễn Hồng Thái – 21520445
3. Nguyễn Lưu Trọng Tấn – 21520103

Tp.HCM, ngày 04 tháng 01 năm 2025

Mục lục

I. GIỚI THIỆU ĐỀ TÀI

1. Tổng quan về Image Colorization

Image colorization (tô màu ảnh) là tác vụ đã được nghiên cứu từ nhiều năm. Một cách nôm na, đây là quá trình khôi phục hoặc thêm màu sắc cho các bức ảnh không màu hoặc mất màu.

Quá trình tô màu ảnh, mở rộng ra là video không chỉ nâng cao tính thẩm mỹ của ảnh và video mà còn mang lại nhiều lợi ích thiết thực trong các lĩnh vực khác nhau, có thể kể đến như:

- **Bảo Tồn Lịch Sử**: Phục hồi các ảnh, bộ phim đen trắng, tư liệu lịch sử, giúp lưu giữ và truyền tải giá trị văn hóa cho các thế hệ sau.
- **Giáo Dục**: Tô màu các tài liệu ảnh, video giúp tăng cường sự hấp dẫn và sinh động trong giảng dạy, giúp học sinh dễ dàng tiếp thu thông tin hơn.
- **Khoa Học Y Tế**: Hỗ trợ quan sát và phân tích dữ liệu hình ảnh y tế, cải thiện độ chính xác trong chẩn đoán và nghiên cứu.
- **Giải Trí**: Cải thiện trải nghiệm người xem trong các bộ phim, chương trình truyền hình bằng cách thêm màu sắc tự nhiên cho các nội dung cũ.

2. Lý do lựa chọn đề tài

Trước đây, các chuyên gia hoặc nghệ sĩ sẽ tô màu thủ công cho các ảnh, quá trình này tốn rất nhiều thời gian và công sức.

Sự phát triển của máy tính với các phần mềm đồ họa chuyên dụng đã hỗ trợ rất nhiều trong việc tô màu ảnh bằng các ứng dụng và công cụ mạnh mẽ, chẳng hạn Adobe Photoshop, Corel Painter, …

Bên cạnh đó, không thể không kể đến đó là sự giúp ích của các thuật toán máy học và trí tuệ nhân tạo trong tác vụ tô màu ảnh và video. Các thuật toán được nghiên cứu và cải thiện dần dần, từ các thuật toán được lập trình thủ công dựa trên các quy tắc được định nghĩa trước, dự đoán màu sắc dựa trên đặc trưng ảnh bằng các thuật toán máy học cơ bản, ứng dụng các mạng học sâu như CNN. Đỉnh điểm là sự xuất hiện của các mô hình AI tạo sinh và mô hình Transformer. Các mô hình ngày càng được phát triển, tự động hóa quá trình tô màu ảnh nói riêng và các tác vụ khác trong thị giác máy tính nói chung.

Cùng với sự giúp đỡ của thầy Cáp Phạm Đình Thăng, với đồ án môn học, nhóm chúng em tập tập trung vào việc sử dụng mô hình GAN cùng với CNN để thực hiện tác vụ tô màu ảnh – Image Colorization. Đồ án này là cơ hội để chúng em nghiên cứu cách thức hoạt động của GAN – một trong những mô hình GenAI cổ điển, ôn tập lại cách thức hoạt động của CNN – một trong những loại mạng học sâu quen thuộc, xem xét ứng dụng và tính hiệu quả của chúng vào tác vụ tô màu ảnh.

II. BÀI TOÁN VÀ DỮ LIỆU

1. Mô tả bài toán

Với một ảnh số ở dạng grayscale, yêu cầu của bài toán là thực hiện tô màu cho ảnh nói trên.

Input:

- Dataset: $D = \{I_m; (L_m; a_m; b_m)\},\ m = 1,2,3,\dots,N$
  $I_m,\ m = 1,2,3,\dots,N$ là các ảnh số ở dạng grayscale (ảnh xám)
  $L_m; a_m; b_m$ lần lượt là giá trị biểu diễn trong hệ màu Lab của ảnh màu (ảnh gốc)
- Image là ảnh số (grayscale) cần được tô màu

Output:

- $\text{Image}_{color}^{256\times256}$là ảnh đã được tô màu

Dataset sẽ tiến hành tiền xử lý dữ liệu trước khi đưa vào thực nghiệm. Đồng thời các ảnh output luôn được đưa về cùng kích thước là 256 x 256.

2. Tập Dữ Liệu

Để huấn luyện và kiểm thử mô hình, nhóm sử dụng các bộ dữ liệu sau:

- **ImageNet 1000 (mini)**: Một phiên bản rút gọn của ImageNet với 1,000 lớp, cung cấp đa dạng các đối tượng và cảnh vật.
- **100 Sports Image Classification** : Bộ dữ liệu chứa các hình ảnh thể thao, giúp mô hình học cách tô màu cho các hoạt động động.
- **CelebA-HQ resized (256x256)**: Bộ dữ liệu hình ảnh khuôn mặt được nâng cấp với độ phân giải cao (256x256), giúp mô hình học cách tô màu các khuôn mặt một cách tự nhiên.

Các bộ dữ liệu này được kết hợp và xử lý để tạo thành tập huấn luyện phong phú và đa dạng, đảm bảo mô hình có khả năng general hóa cao.

3. Tiền Xử Lý Dữ liệu

Dữ liệu trước khi đưa vào thực nghiệm sẽ được tiền xử lý, các bước bao gồm:

+ Kiểm tra và loại bỏ các ảnh grayscale.
+ Chọn ra các ảnh có đuôi .jpeg và .jpg, sau đó resize các ảnh đã chọn về cùng kích thước 256×256, biến đổi không gian màu RGB sang Lab và chuẩn hóa khoảng giá trị về [-1; 1]
+ Tăng cường dữ liệu tập train bằng cách lật ngang ảnh ngẫu nhiên sau khi đã resize.

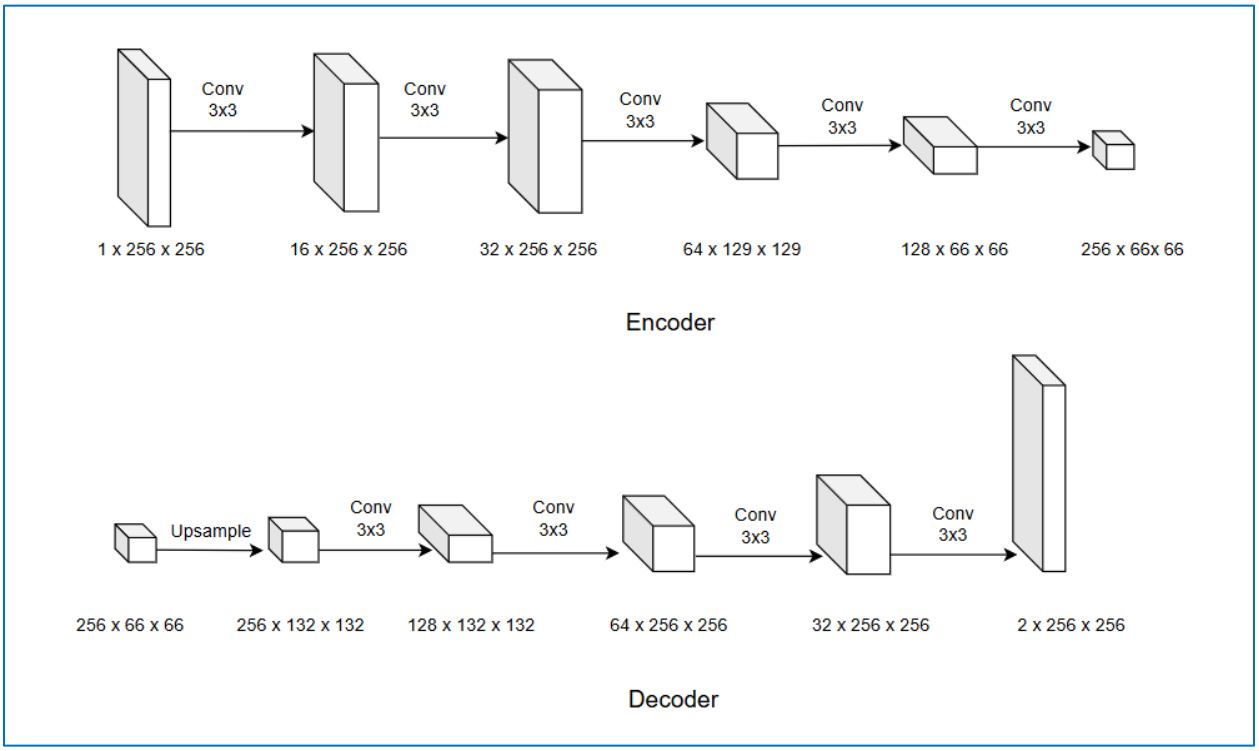
III. PHƯƠNG PHÁP TIẾP CẬN

1. Mạng Tích Chập - CNN

1.1. Giới thiệu về CNN

Convolutional Neural Network (CNN) là một loại mạng nơ-ron sâu được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc lưới như hình ảnh. Trong nhiệm vụ colorization, CNN được sử dụng để học ánh xạ giữa các khung hình đen trắng và màu sắc tương ứng, thông qua việc học các đặc trưng không gian và ngữ cảnh của hình ảnh.

1.2. Kiến trúc CNN cho tác vụ Colorization



Hình 1. Kiến trúc Encoder và Decoder sử dụng CNN cho tác vụ tô màu ảnh

Đối với tác vụ Image Colorization, kiến trúc mạng được tổ chức thành Encoder và Decoder, trong đó các lớp của cả 2 thành phần này đều là các lớp CNN, ngoài ra trong kiến trúc còn sử dụng thêm các lớp Batch Normalization, Max Pooling và Upsampling.

Encoder nhận input là tensor độ sáng *L*, Decoder nhận input là output của Encoder.

Output của Decoder sau đó được đưa qua lớp Concatenation và Output layer để dự đoán *ab*.

Kiến trúc cụ thể của từng khối như sau:

- **Encoder**:

+ Sử dụng 5 lớp *Convolution*
+ 4 lớp *BatchNorm* được sử dụng sau mỗi lớp tích chập, bỏ qua lớp đầu tiên
+ Sử dụng 2 lớp *Max Pooling*.
+ Hàm kích hoạt được sử dụng giữa các lớp đều là *Leaky ReLU*.

- **Decoder**:
  + Sử dụng 2 lớp *Convolution.*
  + Lớp *Upsampling* được sử dụng trước mỗi lớp tích chập và
  + Lớp *Batch Normalization* được sử dụng sau lớp tích chập đầu tiên
  + Hàm kích hoạt được sử dụng giữa các lớp là *Leaky ReLU*.

Sau khi đưa qua Encoder và Decoder, output tại lớp đầu ra của Decoder được resize về kích thước của tensor độ sáng *L* trở thành tensor *x*.

- **Concatenation**:
  + Nối tensor *x* và tensor *L*
  + Sau khi nối, tensor này được đưa qua 1 lớp tích chập và lớp Batch Normalization
  + Hàm kích hoạt được sử dụng là hàm *Leaky ReLU*
- **Output:**
  + Sử dụng 2 lớp tích chập
  + Hàm kích hoạt được sử dụng sau lớp đầu tiên là hàm *Leaky ReLU*
  + Hàm kích hoạt được sử dụng cho lớp cuối cùng là hàm *Tanh*

1.3. Quy trình huấn luyện của CNN

- Encoder trích xuất đặc trưng
  + Ở giai đoạn Encoder, mô hình sử dụng các lớp tích chập để trích xuất đặc trưng cục bộ.
  + Sử dụng Batch Normalization sau mỗi lớp tích chập giúp tăng tốc độ và ổn định quá trình huấn luyện
  + Max Pooling được sử dụng để giảm chiều không gian của đặc trưng, giúp tinh gọn dữ liệu, giảm chi phí tính toán cũng như hạn chế overfitting.
  + Hàm Leaky ReLU được sử dụng để tránh tình trạng gradient biến mất khi giá trị bé hơn 0.
- Decoder khôi phục kích thước:
  + Ở giai đoạn decoder, các lớp Upsampling được sử dụng để phóng to đặc trưng trở về kích thước ban đầu.
- Ghép kênh (Concatenation) và tiếp tục tích chập:

+ Mô hình ghép (concatenate) output của Decoder với ảnh xám gốc (L channel) trước khi đi qua các lớp tích chập tiếp theo.
+ Cuối cùng, mô hình dùng lớp tích chập đầu ra để dự đoán 2 kênh màu a,b.

2. Mạng Đối Kháng Tạo Sinh – GAN

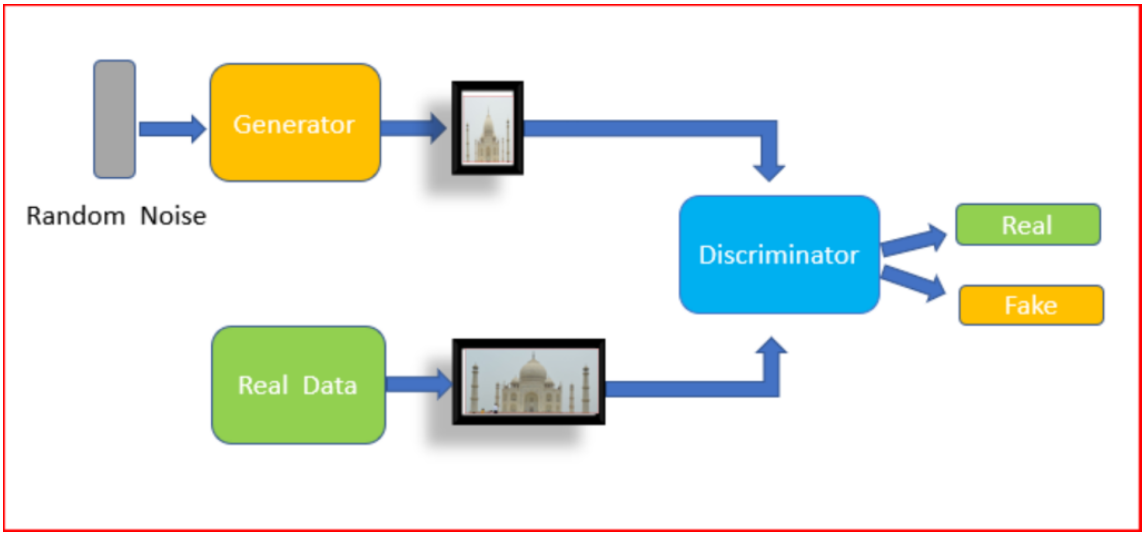
1.1. Mạng GAN

1.1.1. Giới thiệu về GAN

Generative Adversarial Network (GAN) được giới thiệu lần đầu vào năm 2014, là một trong những đột phá của lĩnh vực học sâu, đặc biệt trong việc tạo ra dữ liệu giống với dữ liệu thực.

Ứng dụng đầu tiên của mạng là tạo ra các bức ảnh giả giống với các ảnh thật. Sau này, kiến trúc của GAN dần được cải thiện và khắc phục được các nhược điểm của mô hình GAN truyền thống. Ví dụ: DCGAN, StackedGAN, WGAN, CycleGAN, BigGAN, StyleGAN, …

1.1.2. Các thành phần của GAN



Hình 2. Cấu trúc và hoạt động cơ bản của mạng GAN

Mạng GAN gồm 2 thành phần, bao gồm Generator và Discriminator.

Ý tưởng của GAN giống với trò đối kháng zero-sum. Trong đó, Discriminator sẽ cố gắng phân biệt dữ liệu giả được sinh ra từ Generator. Generator dựa trên kết quả phân biệt từ Discriminator để cố gắng cải thiện khả năng sinh dữ liệu.

1.1.3. Quy trình huấn luyện của GAN

Mô hình GAN và các biến thể về cơ bản có chung một quy trình huấn luyện như sau:

- Bước 1: Phát sinh nhiễu

Input của Generator là một vector nhiễu (noise) z theo một phân phối xác suất (thường là phân phối Gaussian hoặc Uniform). Vector nhiễu đóng vai trò tạo sự đa dạng cho các dữ liệu mà Generator sinh ra

- Bước 2: Generator phát sinh dữ liệu giả

Nhiễu z sau khi đi qua các layer của Generator sẽ tạo ra dữ liệu giả G(z). Ví dụ: tạo ra một bức ảnh giả dựa trên nhiễu z

- Bước 3: Discriminator phân biệt dữ liệu

Discriminator nhận đồng thời cả dữ liệu thật và dữ liệu giả được phát sinh bởi Generator. Nhiệm vụ của Discriminator là gán nhãn cho 2 loại dữ liệu này: 'real' đối với dữ liệu thật và 'fake' đối với dữ liệu giả.

- Bước 4: Học đối kháng

Discriminator: Học cách phân biệt giữa dữ liệu thật và giả sao cho nó gán nhãn chính xác nhất cho cả hai loại dữ liệu.

Generator: Học cách "đánh lừa" Discriminator bằng cách tạo ra dữ liệu giả ngày càng giống dữ liệu thật. Mục tiêu của Generator là làm cho Discriminator không thể phân biệt được giữa dữ liệu thật và giả.

- Bước 5: Lặp lại quá trình

Generator và Discriminator sẽ tiếp tục cải thiện khả năng thông qua quá trình huấn luyện đối kháng. Theo lý thuyết, quá trình huấn luyện sẽ dừng lại (tức mô hình GAN hội tụ) khi cả Discriminator và Generator đạt tới trạng thái cân bằng Nash và không bên nào có thể cải thiện hiệu suất của mình mà không làm giảm hiệu suất của bên kia.

Có nghĩa rằng Generator tạo ra dữ liệu giả mà Discriminator không thể phân biệt được với dữ liệu thật. Còn Discriminator đưa ra dự đoán với xác suất gần như ngẫu nhiên (~50%) cho cả dữ liệu thật và giả.

1.2. Biến thể của GAN - DCGAN

1.2.1. Giới thiệu về DCGAN

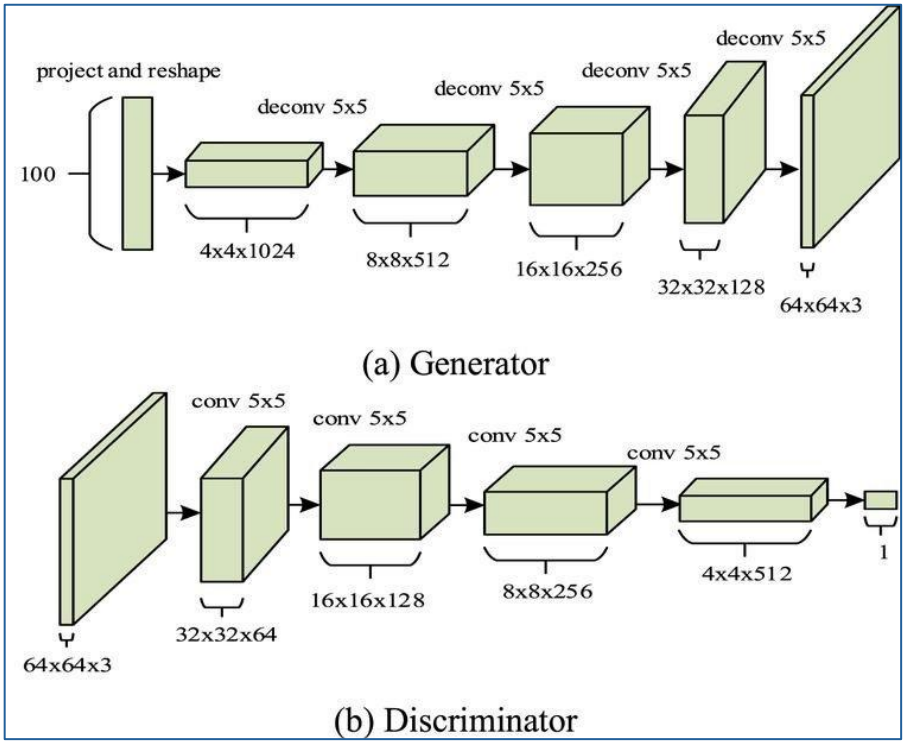
Trong kiến trúc được giới thiệu lần đầu của GAN, cả Generator và Discriminator đều sử dụng fully-connected (multi-layer perceptron).

Ban đầu, mô hình chủ yếu được thiết kế để làm việc với dữ liệu có độ phức tạp thấp (như MNIST với ảnh nhỏ, xám 28x28), do đó các lớp fully connected hoạt động hiệu quả. Khi làm việc với dữ liệu phức tạp hơn (ví dụ, hình ảnh có độ phân giải cao), các lớp fully connected sẽ không còn phù hợp do kích thước tham số lớn và khó khăn trong việc học các tính chất không gian của dữ liệu.

Dựa trên kiến trúc của GAN, DCGAN - một biến thể được giới thiệu, chuyên sử dụng để xử lí dữ liệu dạng hình ảnh.

1.2.2. Kiến trúc của DCGAN

Thay vì sử dụng các lớp fully-connected cho cả Generator và Discriminator như mạng GAN truyền thống, các lớp của Generator là các lớp ***deconvolution***, còn các lớp của Discriminator là các lớp ***convolution***, điều này giúp DCGAN xử lý tốt hơn đối với các dữ liệu dạng hình ảnh.



Hình 2. *Minh họa kiến trúc mạng DCGAN*

Kiến trúc cụ thể của Generator và Discriminator như sau:

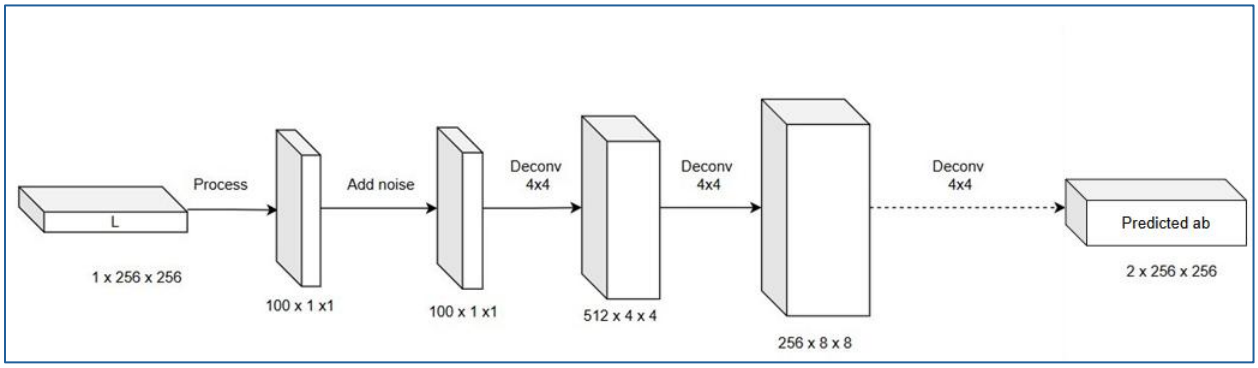
- **Generator:**
  + Sử dụng các lớp *Convolutional Transpose* (*Deconvolution*) để tăng chiều không gian từ vector ngẫu nhiên thành kích thước của ảnh đầu ra
  + *Batch Normalization* được sử dụng sau mỗi lớp Deconvolution để ổn định quá trình huấn luyện
  + Hàm kích hoạt giữa các lớp được sử dụng là hàm *ReLU*.
  + Hàm kích hoạt được sử dụng ở lớp đầu ra là hàm *Tanh*
- **Discriminator:**
  + Sử dụng các lớp *Convolution* để trích xuất đặc trưng từ ảnh
  + Sau mỗi lớp cũng sử dụng *Batch Normalization*
  + Hàm kích hoạt giữa các lớp được sử dụng là hàm *Leaky ReLU* thay vì hàm ReLU để tránh tình trạng gradient bị triệt tiêu khi giá trị bé hơn 0.

+ Hàm kích hoạt được sử dụng ở lớp đầu ra là hàm *Sigmoid*

1.2.3. Kiến trúc DCGAN cho tác vụ Image Colorization

Đối với tác vụ Image Colorization trong hệ màu Lab, input cũng như kiến trúc của Discriminator và Generator có một số thay đổi so với DCGAN ban đầu.

a, Kiến trúc của Generator



Hình 3. Kiến trúc Generator được sử dụng cho tác vụ tô màu ảnh

Kích thước của vector ngẫu nhiên được chọn là 100. Do đó Tensor độ sáng L sẽ được xử lý bằng một processor để đưa về kích thước $100 \times 1 \times 1$, sau đó thêm nhiễu để làm Input cho DCGAN.

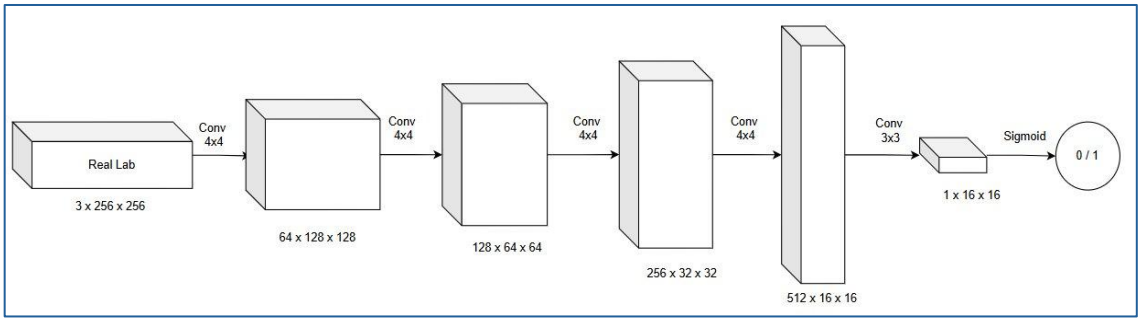
Output của Generator là một tensor chứa thông tin về giá trị màu a, b được dự đoán.

Hàm lỗi được sử dụng cho Generator là: $L_G = -\alpha . L_1(G(z), ab_{real}) - \beta . E\left[log\left(1 - D\left(x_{fake}\right)\right)\right]$

Trong đó:

+ $z = L_{transformed} + noise$ là input của Generator
+ ab_{real} là giá trị màu ab thực tế của ảnh
+ $G(z)$ là giá trị ab được tạo (dự đoán) bởi Generator
+ x_{fake} là ảnh giả, được tạo bởi L và giá trị ab sinh ra từ Generator
+ $D\left(x_{fake}\right)$ là xác suất dự đoán ảnh tạo bởi Generator là ảnh giả của Discriminator
+ Hàm L_1 đo lường sự khác biệt giữa giá trị ab dự đoán và giá trị thực tế
+ α và β là các siêu tham số, được điều chỉnh trong quá trình huấn luyện

b, Kiến trúc của Discriminator



Hình 4. Kiến trúc Discriminator được sử dụng cho tác vụ tô màu ảnh

Discriminator nhận Input là 2 ảnh, bao gồm 1 ảnh được tạo bởi *L* với giá trị *ab* thực và 1 ảnh được tạo bởi *L* với giá trị *ab* giả được sinh ra bởi Generator.

Output của Discriminator là các nhãn dự đoán tương ứng với ảnh thật và ảnh được tô bởi Generator.

Hàm lỗi được sử dụng cho Discriminator là: $L_D = -E[log(D(x_{real}))] - E\left[log\left(1 - D(x_{fake})\right)\right]$

Trong đó:

+ x_{real} và x_{fake} lần lượt là ảnh thật và ảnh giả
+ $D(x_{fake})$ là xác suất dự đoán ảnh tạo bởi Generator là ảnh giả của Discriminator
+ $D(x_{real})$ là xác suất dự đoán ảnh gốc là ảnh thật của Discriminator

IV. TIÊU CHÍ ĐÁNH GIÁ VÀ KẾT QUẢ THỰC NGHIỆM

1. Các độ đo đánh giá

Các metric được sử dụng để đánh giá hiệu quả của quá trình colorization bao gồm:

1.1. PSNR

Peak Signal-to-Noise Ratio - PSNR: Đánh giá độ chính xác của màu sắc trong từng khung hình so với khung hình gốc, đo lường sự khác biệt giữa các pixel.

Công thức tính của PSNR là: $PSNR = 10\,log_{10}\left(\frac{MAX^2}{MSE}\right)$

Trong đó:

+ MAX là giá trị lớn nhất của pixel trong hệ màu 8-bit
+ MSE là sai số giữa ảnh được tô với ảnh gốc

Giá trị PSNR phản ánh độ sai khác của ảnh được tô so với ảnh gốc:

+ $PSNR \geq 40$: Ảnh được tô rất giống ảnh gốc

+ $30 \leq PSNR < 40$: Ảnh được tô có chất lượng cao, không thể phân biệt được với ảnh gốc bằng mắt thường.
+ $20 \leq PSNR < 30$: Ảnh được tô có sự khác biệt với ảnh gốc nhưng có thể chấp nhận.
+ $PSNR < 20$: Ảnh chưa đạt yêu cầu

1.2. Delta-E

Giá trị ΔE đánh giá sự sai khác của hai ảnh trong hệ màu Lab.

Công thức tính của ΔE là: $\Delta E = \sqrt{(L_1 - L_2)^2 + (a_1 - a_2)^2 + (b_1 - b_2)^2}$

Trong đó:

+ $L_1; a_1; b_1$ lần lượt là các giá trị L,a,b của ảnh đã được tô màu
+ $L_2; a_2; b_2$ lần lượt là các giá trị L,a,b của ảnh gốc (ảnh màu)

Đối với tác vụ tô màu ảnh, do $L_1 = L_2 = L$ nên giá trị ΔE phản ảnh độ lệch về màu sắc của ảnh được tô so với ảnh gốc. Cụ thể

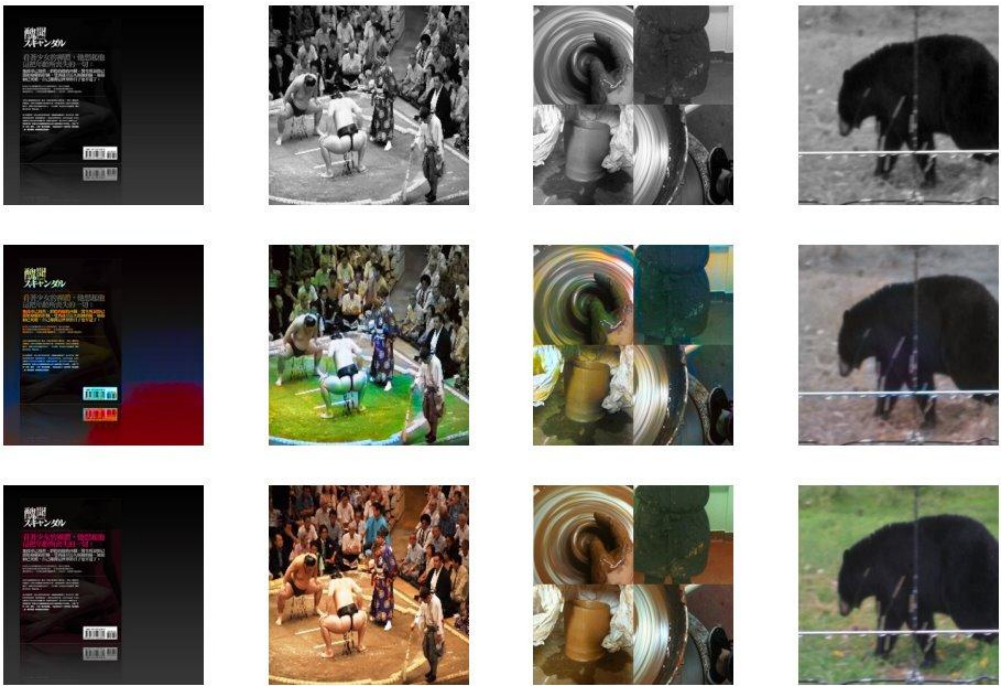
+ $\Delta E \leq 1$: Ảnh được tô rất giống ảnh gốc
+ $1 < \Delta E < 2$: Khác biệt giữa 2 ảnh rất nhỏ.
+ $2 \leq \Delta E < 5$: 2 ảnh có sự khác biệt nhưng vẫn có thể chấp nhận.
+ $5 \leq \Delta E < 10$: Khác biệt rõ ràng, ảnh chưa đạt yêu cầu.
+ $\Delta E \geq 10$: Khác biệt 2 ảnh rất lớn, không thể ứng dụng

2. Kết quả thực nghiệm

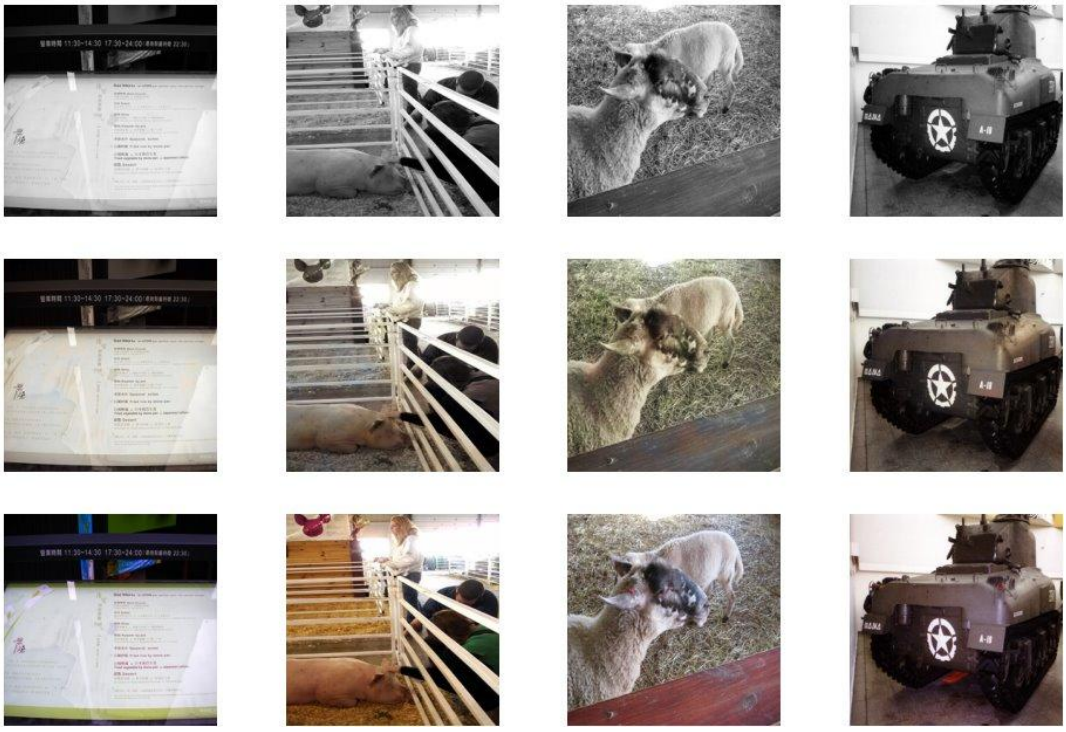
Các giá trị độ đo tương ứng của hai phương pháp là:

Metric	CNN	DCGAN
PSNR	**23.24**	20.04
ΔE	**15.12**	19.94

Một số ảnh kết quả:



Hình 5. Kết quả tô màu sử dụng mô hình GAN



Hình 6: Kết quả tô màu sử dụng mô hình CNN

V. NHẬN XÉT VÀ KẾT LUẬN

1. Nhận xét

Mô hình CNN cho kết quả tốt hơn trên cả hai độ đo

Các ảnh sinh ra vẫn có sự sai khác so với ảnh gốc, nhưng nhiều ảnh được tô vẫn cho kết quả khá chân thực và không thể nhận ra nếu được đặt riêng.

Mặc dù đạt được những kết quả khả quan, dự án vẫn gặp phải một số hạn chế sau:

- Hạn chế về tài nguyên tính toán: Nhóm thực hiện huấn luyện trên GPU miễn phí của Kaggle. Do lượng thời gian sử dụng có hạn nên nhóm chưa khai thác triệt để các mô hình, các siêu tham số được điều chỉnh chưa đạt tối ưu.
- Mô hình GAN gặp rất nhiều khó khăn khi huấn luyện, nhất là vấn đề Mode Collapse, mặc dù được tinh chỉnh nhiều để cải thiện khả năng tô màu ảnh nhưng kết quả thu được vẫn chưa đủ tốt.
- Giới hạn về số lượng ảnh trong bộ Dataset: số mẫu trong tập dữ liệu có thể chưa đủ lớn để các mô hình đưa ra kết quả tốt
- Hạn chế trong đánh giá mô hình: Hai độ đo được sử dụng đang đo lường sự khác biệt giữa ảnh được tô với ảnh gốc. Mặc dù vậy, mục tiêu cuối cùng của mô hình không phải là tái tạo ảnh gốc mà là tô màu ảnh sao cho ảnh có tính chân thực và có thể sử dụng được.
- Hạn chế về mô hình sử dụng: Kiến trúc trong các mô hình có thể chưa đủ sâu để mô hình học được các đặc trưng sâu hơn.

2. Kết luận

Image colorization sử dụng GAN và CNN là hướng nghiên cứu đầy hứa hẹn. Mặc dù kết quả thực nghiệm chưa đủ tốt, nhưng các mô hình cũng đã cho thấy khả năng của mình để đưa ra những màu sắc chân thực. Dự án đã chứng minh được tiềm năng của GAN cũng như CNN trong việc tái tạo màu sắc tự nhiên, đồng thời cung cấp cái nhìn về mô hình GAN – một trong những GenAI model cổ điển.

Mặc dù vậy, dự án vẫn còn nhiều thách thức cần phải khắc phục, bao gồm tăng tốc độ xử lý, nâng cao tính nhất quán màu sắc và phạm vi ứng dụng. Nhìn chung, dù còn nhiều thách thức, Image Colorization sử GAN và CNN vẫn là một chủ đề thu hút trong Thị giác máy tính nói riêng và học sâu nói chung.

VI. Hướng Phát Triển

Để khắc phục các hạn chế trên và nâng cao hiệu quả của mô hình, nhóm đã đề xuất các hướng phát triển sau:

- Tối ưu hóa tốc độ xử lý: sử dụng các mô hình nhẹ hơn, tối ưu hóa kiến trúc mạng và tận dụng các kỹ thuật phân tán để tăng tốc quá trình huấn luyện và colorization.
- Mở Rộng Bộ Dữ Liệu: thu thập và tạo dựng thêm các bộ dữ liệu đa dạng hơn, bao gồm các loại video đặc thù để mô hình có khả năng học tập và general hóa tốt hơn.
- Cải thiện kỹ thuật đánh giá: thay đổi hàm loss khi thực nghiệm (ví dụ Hinge Loss, …), sử dụng các độ đo khác để đo lường tính chân thực của ảnh được tô màu, …
- Áp Dụng Các Mô Hình GAN Tiên Tiến: Khai thác các mô hình GAN mới như StyleGAN, BigGAN hoặc các biến thể khác để cải thiện chất lượng và tính nhất quán của colorization.
- Sử dụng kĩ thuật và mô hình tiên tiến hơn: Ví dụ cơ chế Attention, mô hình Vision Transformer, …
- Sử Dụng Phương Pháp Học Nâng Cao: kỹ thuật học chuyển giao (Transfer Learning), học không giám sát (Unsupervised Learning), học tăng cường (Reinforcement Learning)

VII. Tài Liệu Tham Khảo

1. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672-2680).
2. Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).
3. Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2223-2232).
4. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., ... & Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4681-4690).
5. Wang, X., Yu, K., Dong, C., Loy, C. C., & Tang, X. (2018). ESRGAN: Enhanced super-resolution generative adversarial networks. *arXiv preprint arXiv:1809.00219*.

6. Cho, M., Lee, J. Y., Lee, Y., & Kweon, I. S. (2018). Deep feature consistent generative adversarial networks for video colorization. In *Proceedings of the European Conference on Computer Vision* (pp. 86-103).
7. Larsson, G., Maire, M., & Shakhnarovich, G. (2016). Learning representations for automatic colorization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4496-4504).
8. Moein Shariatnia, et al. (n.d.). Image Colorization with U-Net and GAN Tutorial. *Google Colab Notebook*.
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. *arXiv preprint arXiv:1406.2661*.
10. Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv preprint arXiv:1511.06434.