Xây dựng mô hình tô màu hình ảnh tự động bằng GAN-Unet

Building an Automated Image Colorization Model using GAN-Unet

Nguyen Dinh Thanh San, Nguyen Huynh Chi Khang, Nguyen Phan Duc Thanh, Nguyen Phuoc Trinha,b,\*

Nguyen Dinh Thanh San, Nguyen Huynh Chi Khang, Nguyen Phan Duc Thanh, Nguyen Phuoc Trinha,b,\*

*aTrường Khoa học máy tính, Đại học Duy Tân, Đà Nẵng, Việt Nam*

*aSchool of Computer* *Science, Duy Tan University, 55000, Danang, Vietnam*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Tóm tắt**

Đề tài tự động tô màu hình ảnh là một đề tài đầy hấp dẫn và thú vị liên quan đến việc chuyển đổi hình ảnh trắng đen thang độ xám thành các hình ảnh đầy màu sắc tương ứng. Đề tài này là một đề tài được nghiên cứu khá nhiều, tuy nhiên nhiệm vụ này vốn không được thực hiện tốt do có nhiều cách tô màu hợp lý có thể được liên kết với một hình ảnh đen trắng. Với những tiến bộ gần đây Deep Learning đã dẫn đến sự phát triển các mô hình và phương tô màu hình ảnh tự động, trong đó việc kết hợp 2 mô hình GAN (Generative Adversarial Networks) và Unet nổi lên như một kỹ thuật đặc biệt hiệu quả trong bài toán tô màu hình ảnh. Trong đề tài này, chúng tôi xây dựng một mô hình tô màu hình ảnh tự động dựa trên sự kết hợp giữa GAN và mạng U-Net, được gọi là GAN-Unet. Unet là một mạng CNN phổ biến sử dụng trong các bài toàn xử lý hình ảnh, với kiến trúc đặc biệt của mình, Unet có khả năng trích xuất đặc trưng và tái tạo hình ảnh từ hình ảnh đầu vào rất tốt, qua đó giữ lại được tính toàn vẹn của hình ảnh đầu vào – yếu tố rất quan trọng trong bài tô màu hình ảnh. Trong khi đó, mô hình GAN là một mô hình có khả năng tạo ra được những hình ảnh rất chất lượng, với cấu trúc gồm 2 phần là Generator và Discriminator được huấn luyện bằng quy trình huấn luyện đối nghịch, trong đó Generator và Discriminator được cập nhật đồng thời để cải thiện hiệu suất của chúng. Việc tìm hiểu, nghiên cứu để kết hợp 2 mô hình này lại với nhau đã mang đến một mô hình tô màu cho hình ảnh đen trắng hết sức chất lượng.

*Từ khoá:* Deep Learning; GAN; Unet; Generator; Discriminator.

\* *Corresponding Author:* Nguyen Dinh Thanh San, Nguyen Huynh Chi Khang, Nguyen Phan Duc Thanh, Nguyen Phuoc Trinh, *School of Information Technology, Duy Tan University, 55000, Danang, Vietnam*

*Email:* [sannguyen0907@gmail.com](mailto:sannguyen0907@gmail.com) or [nguyendthanhsan@gmail.com](mailto:nguyendthanhsan@gmail.com) or nguyendthanhsan@dtu.edu.vn

**Abstract**

Automatic image colorization is a captivating and intriguing topic that involves transforming grayscale images into their corresponding colorful counterparts. This task has attracted significant research attention, yet it remains challenging due to the inherent multiplicity of plausible colorizations associated with a single grayscale image. Recent advancements in deep learning have spurred the development of promising approaches for automatic image colorization, with Generative Adversarial Networks (GANs) emerging as a particularly effective technique. In this work, we propose an automated image colorization model based on the integration of GANs and the U-Net architecture, known as GAN-Unet. The GAN-Unet model comprises two primary components: a generator and a discriminator. The generator takes a grayscale image as input and produces a colorized image as output. The discriminator, on the other hand, distinguishes between real color images and color images generated by the generator. The GAN-Unet model is trained using an adversarial training process, where the generator and discriminator are simultaneously updated to enhance their performance. The generator is trained to generate colorized images that are indistinguishable from real color images, while the discriminator is trained to accurately classify real and fake color images.

**1. Mở đầu**

Vào thời điểm cuối năm 2022, công ty OpenAI đã cho ra mắt chatbot thông minh ChatGPT, một mô hình hỗ trợ trong nhiều nhiệm vụ như trả lời câu hỏi, viết văn bản, dịch thuật, và thậm chí tham gia vào các cuộc trò chuyện tương tác trong rất nhiều lĩnh vực khác nhau. Nhờ việc sử dụng chat GPT, nhiều người nhận ra rằng việc sử dụng các công cụ AI không những có thể giảm thiểu khối lượng công việc hằng mà còn nâng cao chất lượng và năng suất công việc. Các công cụ AI từ đó cũng được nhiều người quan tâm và biết đến nhiều hơn, và nó đã tạo một động lực to lớn để các đội ngũ nghiên cứu, sáng tạo cho ra mắt nhiều công cụ AI hiện đại, tiến tiến và đáp ứng nhiều nhu cầu của người dùng. Đặc biệt là vào thời điểm nửa cuối năm 2023, một loạt công cụ xử lý hình ảnh và sinh ảnh theo nhiều nội dung và hình thức khác nhau xuất hiện như Dall-E, Midjourney, …Với một niềm đam mê chung với lĩnh vực Computer Vision cũng như xu hướng của AI thời điểm bấy giờ, nhóm đã quyết định làm một đề tài về lĩnh vực này. Qua quá trình tìm hiểu, phân tích những thuận lợi và khó khăn trong quá trình thực hiện một dự án AI và nguồn lực sẵn có, nhóm đã quyết định chọn đề tài: “Xây dựng mô hình tô màu hình ảnh tự động bằng GAN-Unet”. Đề tài này tập trung vào việc xây dựng một công cụ AI sử dụng mô hình Generative Adversarial Network-Unet, hoặc viết tắt là Gan-Unet, để tự động tô màu cho các hình ảnh đen trắng, giúp tái tạo lại những bức ảnh cũ, làm mới nội dung để sử dụng trong marketing và quảng cáo, hoặc thậm chí giúp nghệ sĩ thể hiện sự sáng tạo của họ một cách nhanh chóng và hiệu quả.

**2. Mục đích nghiên cứu**

Xây dựng được công cụ AI có khả năng tô màu tự động cho hình ảnh đen trắng.

Tạo ra các hình ảnh có chất lượng cao, màu sắc chân thực, sống động.

Tìm hiểu các kỹ thuật xử lý ảnh.

Tìm hiểu các phương pháp thu thập dữ liệu và xử lý dữ liệu trước khi tiến hành huấn luyện.

Khám phá và thử nghiệm công cụ trong nhiều lĩnh vực khác nhau.

Tiến hành đánh giá sự hiệu quả và độ chính xác của công cụ và thực hiện các điều chỉnh và tối ưu hóa cần thiết để cải thiện kết quả.

**3. Phương pháp nghiên cứu**

Tìm hiểu về các mô hình GAN và Unet đã tồn tại, cũng như các nghiên cứu liên quan đến việc tô màu hình ảnh. Điều này bao gồm việc đọc các bài báo nghiên cứu, tài liệu học tập và dự án tương tự đã được triển khai.

Thu thập tập dữ liệu hình ảnh mặt người đen trắng cùng với hình ảnh màu tương ứng để sử dụng cho việc đào tạo và kiểm tra mô hình

**4. Đối tượng nghiên cứu**

Tìm hiểu về các mô hình GAN và Unet đã tồn tại, cũng như các nghiên cứu liên quan đến việc tô màu hình ảnh. Điều này bao gồm việc đọc các bài báo nghiên cứu, tài liệu học tập và dự án tương tự đã được triển khai.

Thu thập tập dữ liệu hình ảnh mặt người đen trắng cùng với hình ảnh màu tương ứng

**5. Phạm vi nghiên cứu**

Giới hạn nội dung: Dự án sẽ tập trung vào việc phát triển mô hình GAN-Unet cho tô màu hình ảnh mặt người đen trắng. Với mục đích là xác định và cài đặt một kiến trúc mạng GAN-Unet cụ thể để thực hiện nhiệm vụ này. Sau đó sẽ thực hiện đào tạo mô hình và đánh giá hiệu suất.

Cơ sở dữ liệu hình ảnh mặt người được sử dụng để đào tạo và kiểm tra mô hình. Số lượng hình ảnh đã được sử dụng cho việc đào tạo mô hình là 2000 hình ảnh.

**6. Kết quả nghiên cứu**

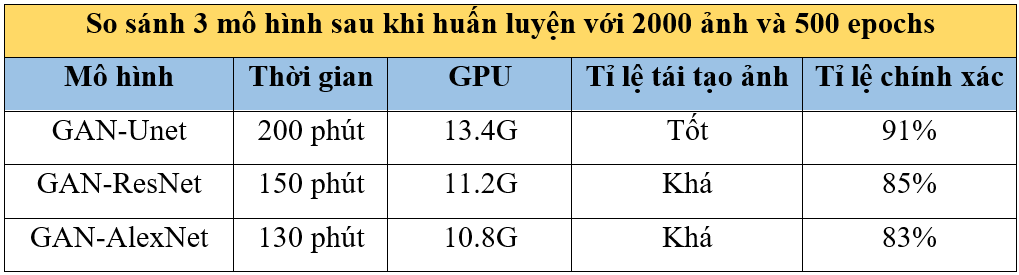
**6.1. Lí do chọn mô hình GAN-UNet**

Bài toán Xây dựng mô hình tô màu hình ảnh tự động là một nhiệm vụ quan trọng trong lĩnh vực xử lý hình ảnh. Mục tiêu là phát triển các mô hình trí tuệ nhân tạo (AI) có khả năng hiểu và tô màu hình ảnh một cách tự động và chính xác. Các mô hình AI phổ biến được sử dụng cho bài toán này bao gồm

• GAN-Unet: GAN (Generative Adversarial Network) kết hợp với Unet là một phương pháp phổ biến để giải quyết bài toán tô màu. GANs giúp tạo ra hình ảnh màu sắc thú vị và tự nhiên bằng cách đối đầu giữa một mô hình sinh (generator) và một mô hình phân biệt (discriminator). Unet sau đó được sử dụng để kết hợp thông tin chi tiết từ hình ảnh đen trắng và mô hình màu sắc được tạo ra bởi GAN.

• GAN-ResNet: Kiến trúc ResNet được kết hợp với mạng GAN để cải thiện khả năng học sâu và khả năng tạo ra hình ảnh chất lượng cao. ResNet là một kiến trúc mạng nơ-ron sâu nổi tiếng với khả năng xử lý vấn đề biến mất gradient. Việc sử dụng ResNet cũng giúp giảm thiểu hiện tượng vanishing gradient và cải thiện khả năng hội tụ của mô hình. Mô hình GAN-ResNet thường được áp dụng trong các nhiệm vụ như tạo ảnh, tô màu ảnh, và tái tạo hình ảnh với chất lượng cao.

• GAN-AlexNet: AlexNet là một trong những mô hình đầu tiên đạt được thành công lớn trong việc sử dụng mạng nơ-ron sâu cho bài toán nhận dạng hình ảnh. Khi kết hợp với GAN, AlexNet giúp cho mô hình có khả năng nhận diện và học các đặc trưng cấp cao trong hình ảnh. GAN-AlexNet thường được sử dụng trong các ứng dụng như tạo ảnh giả mạo, tái tạo hình ảnh, và tạo ra ảnh tự nhiên với chất lượng cao. Việc sử dụng AlexNet giúp cho mô hình GAN-AlexNet có thể học được các đặc trưng phức tạp trong hình ảnh và tạo ra các đầu ra có chất lượng và đa dạng.



1. Bảng so sánh 3 mô hình.

Sau quá trình huấn luyện và so sánh kết quả tô màu của 3 mô hình trên, nhóm chúng em đã thấy rằng GAN-Unet là mô hình có khả năng tái tạo hình ảnh một cách đầy đủ, sắc nét, tô màu chân thực và sống động nhất, đồng thời khi sử dụng GAN cũng sẽ tối ưu hóa tài nguyên một cách tốt nhất, vì vậy GAN-Unet đã được nhóm đặt niềm tin để sử dụng vào bài toán tô màu hình ảnh này.

**6.2. Mô hình Unet**

**6.2.1. Giới thiệu về Convolutional Neural Network**

CNN, viết tắt của Convolutional Neural Network (Mạng Neural Tích chập), là một kiến trúc mạng neural sâu đặc biệt phát triển cho việc xử lý hình ảnh và phân loại hình ảnh. CNN được thiết kế để tự động học và trích xuất các đặc trưng quan trọng từ dữ liệu hình ảnh thông qua các lớp tích chập.

Các lớp cơ bản của CNN:

• Convolutional layer

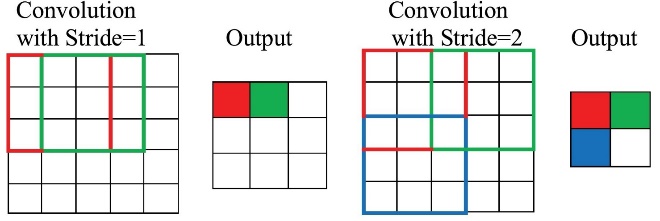
Đây chính là lớp đóng vai trò mấu chốt của CNN, khi layer này đảm nhiệm việc thực hiện mọi tính toán. Nó giúp mô hình có khả năng học được các đặc trưng từ dữ liệu đầu vào. Phép tích chập thường được thực hiện bằng cách trượt một bộ lọc (filter) qua toàn bộ đầu vào. Các filter map này được gọi là ma trận 3 chiều, bên trong chứa các parameter dưới dạng những con số. Tại lớp này, mô hình sẽ thực hiện phép tích chập từng phần tử của bộ lọc với các giá trị tương ứng trong phần nhỏ của đầu vào và sau đó cộng tổng để tạo ra giá trị cho một điểm trong feature map.



1. Phép tích chập trong CNN

Ngoài ra, Stride, Padding, Feature Map là các khái niệm quan trọng trong mạng neural tích chập (CNN)

+ Stride là khoảng cách giữa hai vị trí liên tiếp mà bộ lọc di chuyển qua đầu vào. Nếu Stride là 1, bộ lọc sẽ di chuyển một bước mỗi lần, nếu Stride là 2, bộ lọc sẽ di chuyển hai bước mỗi lần, và cứ như vậy. Stride giúp kiểm soát kích thước của feature map đầu ra.



1. Ví dụ về Stride trong CNN

+ Padding là quá trình thêm các giá trị 0 hoặc giá trị constant vào xung quanh đầu vào trước khi thực hiện phép tích chập. Padding giúp giữ nguyên kích thước của feature map đầu vào và tránh mất thông tin ở biên của hình ảnh. Có hai loại padding phổ biến là "valid" (không padding) và "same" (padding để giữ nguyên kích thước).

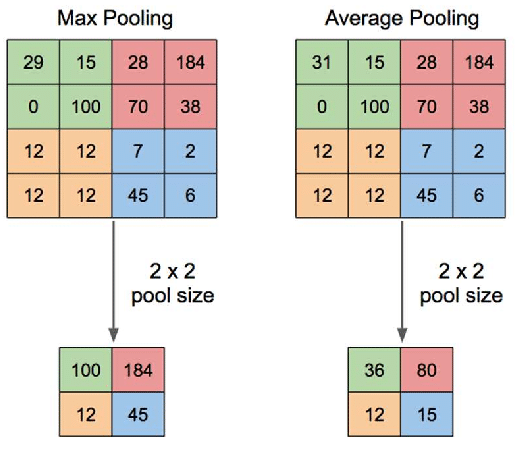
+ Feature map: Feature map là kết quả của quá trình tích chập. Nó là một ma trận chứa các giá trị biểu diễn đặc trưng của đầu vào sau khi đã được áp dụng bộ lọc và các tham số học của mô hình.

• Pooling layer

Khi nhận phải đầu vào quá lớn, các lớp pooling layer sẽ được xếp giữa những lớp Convolutional layer nhằm mục đích giảm parameter. Pooling layer được chia thành 2 loại phổ biến là max pooling và average.

• Pooling layer

Khi 2 lớp convolutional layer và pooling layer nhận được ảnh truyền, lớp này sẽ có nhiệm vụ xuất kết quả. Khi ta nhận được kết quả là model đọc được thông tin ảnh, ta cần phải tạo sự liên kết để cho ra nhiều output hơn. Đây chính là lúc các lập trình viên sử dụng fully connected layer. Hơn nữa, nếu fully connected layer có dữ liệu về hình ảnh thì chúng sẽ chuyển thành mục chưa được phân chia chất lượng.



1. Các loại pooling phổ biến

• Actiovation function

Activation function là một thành phần quan trọng trong mạng neural tích chập (CNN). Nó được sử dụng để giới hạn giá trị đầu ra của một neuron trong mạng. Một số hàm kích hoạt phổ biến trong CNN bao gồm: ReLU, Sigmoid, Tanh, LeakyReLU, ELU, và Softmax. Mỗi hàm kích hoạt có những ưu điểm và nhược điểm riêng, và lựa chọn hàm kích hoạt phù hợp sẽ giúp cải thiện hiệu suất của mô hình.



1. Một số hàm kích hoạt phổ biến

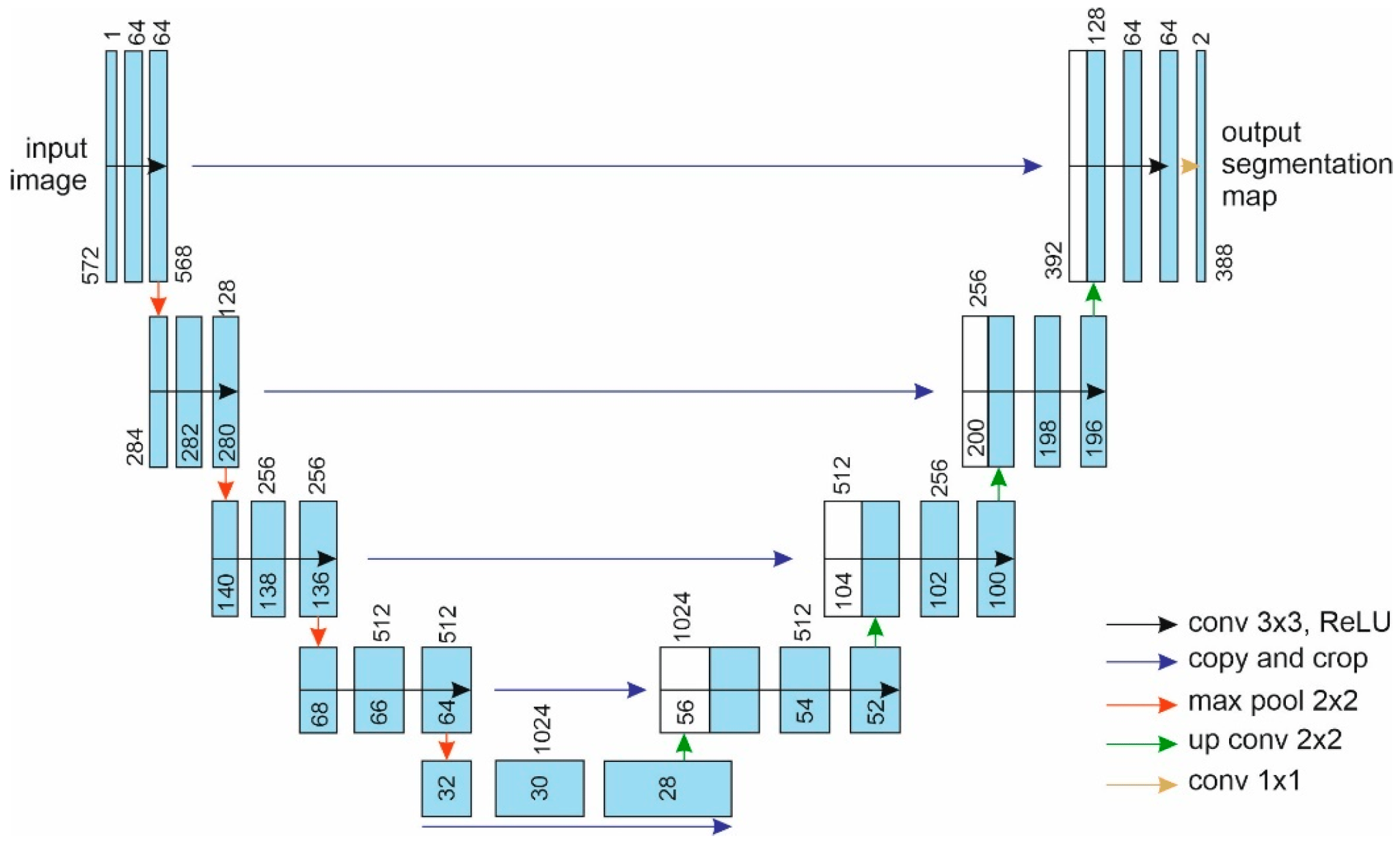
**6.2.2. Giới thiệu về mô hình Unet**

Mô hình UNET có các thành phần chính sau:

• Nhánh encoder: Phần này chịu trách nhiệm trích xuất các đặc trưng quan trọng từ hình ảnh đầu vào. Nhánh encoder bao gồm một loạt các lớp convolution và pooling. Các lớp convolution được sử dụng để trích xuất các đặc trưng từ hình ảnh đầu vào. Các lớp pooling được sử dụng để giảm kích thước của hình ảnh, giúp mô hình có thể học hỏi các đặc trưng tổng quát hơn. Các lớp này giúp nhận dạng các đặc trưng như cạnh, góc và các đặc điểm quan trọng khác của hình ảnh.

• Nhánh decoder: Phần này thực hiện việc tái tạo hình ảnh đầu ra từ các đặc trưng đã được trích xuất bởi encoder. Cấu Nhánh decoder bao gồm một loạt các lớp convolution và upsampling. Các lớp convolution được sử dụng để xây dựng lại hình ảnh đầu ra từ các đặc trưng đã được trích xuất. Các lớp upsampling được sử dụng để tăng kích thước của hình ảnh, giúp mô hình có thể khôi phục các chi tiết nhỏ.

• Bottleneck: Đây là một phần trung gian giữa encoder và decoder, và thường chỉ bao gồm một lớp tích chập. Nó giúp kết nối các đặc trưng cấp thấp và cấp cao giữa hai phần này.



1. Kiến trúc của mô hình Unet

• Một phần quan trọng của Unet là Skip connection. Skip connection là một kỹ thuật kết nối các lớp ở các mức độ giảm kích thước (downsampling) với các lớp ở các mức độ tăng kích thước (upsampling). Mục tiêu là giúp thông tin từ các lớp giảm kích thước trực tiếp truyền đến các lớp tăng kích thước, giúp mô hình duy trì thông tin về cấu trúc và đặc trưng của đầu vào. Cụ thể, trong U-Net, các skip connection thường được thực hiện bằng cách nối đầu ra của một lớp trong đường đi giảm kích thước với đầu vào của một lớp trong đường đi tăng kích thước. Điều này giúp mô hình kết hợp cả thông tin chi tiết từ các lớp giảm kích thước và thông tin bối cảnh từ các lớp tăng kích thước, tạo ra một đầu ra chi tiết và đồng thời bảo toàn cấu trúc của hình ảnh.

Skip connection giúp U-Net giảm thiểu vấn đề mất thông tin trong quá trình upsampling và cung cấp một cách linh hoạt để tích hợp thông tin từ nhiều mức độ giảm kích thước.

**6.2.3. Nguyên lý hoạt động của mạng UNET**

Skip connection giúp U-Net giảm thiểu vấn đề mất thông tin trong quá trình upsampling và cung cấp một cách linh hoạt để tích hợp thông tin từ nhiều mức độ giảm kích thước

Mô hình UNET hoạt động theo nguyên lý sau:

• Nhánh encoder trích xuất các đặc trưng từ hình ảnh đầu vào.

• Các đặc trưng được truyền qua cổng nối đến nhánh decoder.

• Nhánh decoder xây dựng lại hình ảnh đầu ra từ các đặc trưng đã được trích xuất.

Mô hình UNET có nhiều ưu điểm sau:

• Có khả năng học hỏi các mối quan hệ giữa các đặc trưng ở các cấp độ khác nhau.

• Có thể tái tạo các chi tiết nhỏ trong hình ảnh.

• Có thể được áp dụng cho nhiều loại hình ảnh khác nhau.

Tuy nhiên, mô hình UNET cũng có một số nhược điểm sau:

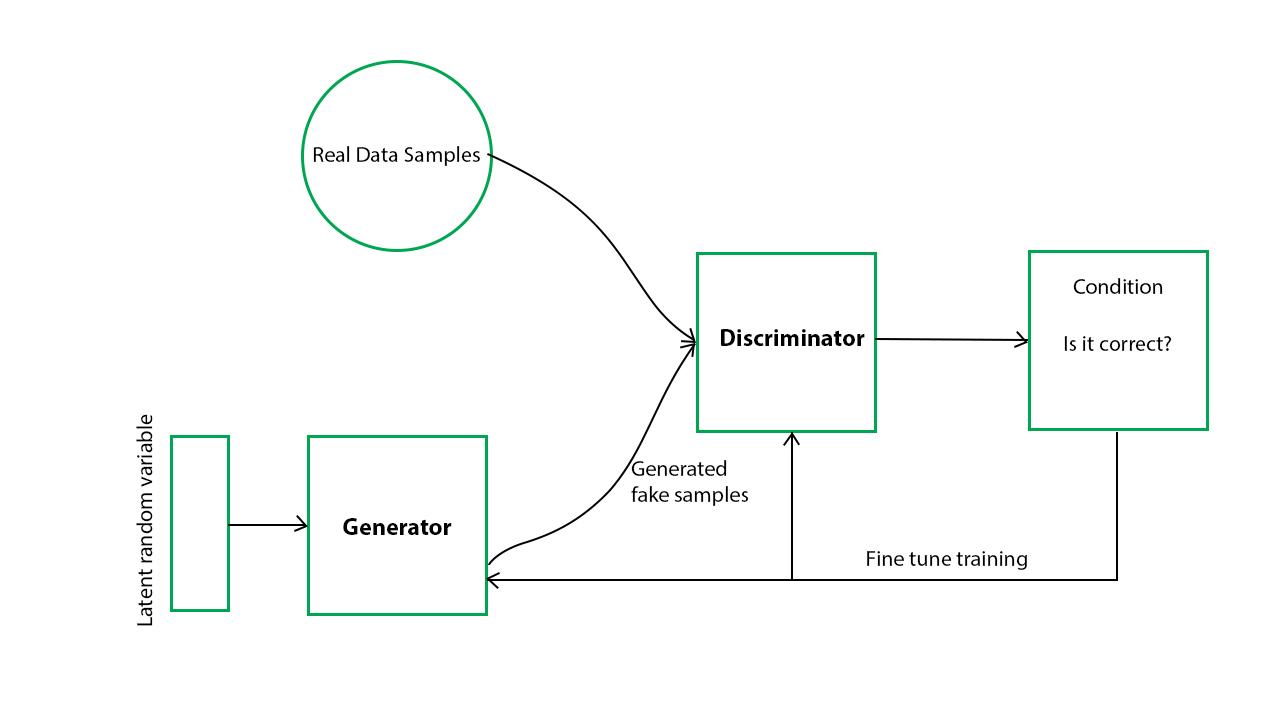
• Có thể mất nhiều thời gian để đào tạo.

• Có thể không phù hợp với các ứng dụng yêu cầu tốc độ xử lý cao

**6.3. Tìm hiểu về Generative Adversarial Networks (GAN)**

**6.3.1. Cấu trúc mạng Generative Adversarial Networks (GAN)**

Mạng GAN (Generative Adversarial Network) bao gồm hai thành phần chính: Generator và Discriminator. Hai thành phần này hoạt động song song và đối đầu với nhau trong quá trình huấn luyện. Dưới đây là mô tả cụ thể về cấu trúc của GAN:



1. Mô hình Generative Adversarial Network

• Generator

+ Input: Generator thường nhận một vector ngẫu nhiên gọi là vector nhiễu (latent vector) làm đầu vào. Vector nhiễu này thường có kích thước cố định và được lấy ngẫu nhiên từ một phân phối chuẩn (ví dụ: phân phối Gaussian).

+ Architecture: Generator thường bao gồm một mạng neural, thường là mạng neural convolutional (CNN) hoặc mạng neural hồi quy (RNN), tùy thuộc vào loại dữ liệu đang được tạo ra. Mạng này có nhiệm vụ biến đổi vector nhiễu thành một hình ảnh hoặc dữ liệu mới tương ứng với loại nhiệm vụ (ví dụ: ảnh mặt người, dữ liệu văn bản, âm thanh, ...).

+ Output: Generator sẽ tạo ra một dữ liệu mới dưới dạng đầu ra, thường là một hình ảnh hoặc dữ liệu theo định dạng của loại nhiệm vụ cụ thể.

• Discriminator

+ Input: Discriminator nhận đầu vào là dữ liệu từ hai nguồn: (a) dữ liệu thật từ tập dữ liệu huấn luyện và (b) dữ liệu được tạo ra bởi Generator.

+ Architecture: Discriminator thường là một mạng neural, thường là CNN, với kiến trúc dùng để phân biệt giữa dữ liệu thật và dữ liệu giả mạo.

+ Output: Discriminator đưa ra một đánh giá về xác suất mà đầu vào là dữ liệu thật. Nếu đầu vào là dữ liệu thật, xác suất này gần với 1. Nếu đầu vào là dữ liệu giả mạo, xác suất này gần với 0.

**6.3.2. Nguyên lý hoạt động của mạng Generative Adversarial Networks (GAN)**

Quá trình huấn luyện của GAN bao gồm việc cải thiện cả Generator và Discriminator thông qua đối đầu. Generator cố gắng tạo ra dữ liệu giống hơn với dữ liệu thật để đánh lừa Discriminator, trong khi Discriminator cố gắng phân biệt dữ liệu thật và giả mạo một cách chính xác hơn.

Hai mạng neural này được huấn luyện xen kẽ nhau. Discriminator được huấn luyện trước và sau đó, Generator được huấn luyện để cải thiện khả năng tạo dữ liệu giả mạo. Quá trình này tiếp tục đến khi Generator tạo ra dữ liệu giả mạo khó phân biệt với dữ liệu thật và Discriminator không thể phân biệt nổi.

Cụ thể:

Huấn luyện Generator: Ban đầu, generator tạo ra các dữ liệu giả một cách ngẫu nhiên từ không gian ẩn. Các dữ liệu này sau đó được đưa vào discriminator để đánh giá.

Huấn luyện Discriminator: Discriminator cố gắng phân biệt giữa dữ liệu thật và giả. Nó được huấn luyện bằng cách sử dụng dữ liệu thật từ bộ huấn luyện cùng với dữ liệu giả từ generator.

Đối đầu và cải thiện: Quá trình đối đầu này tiếp tục lặp lại. Generator cố gắng cải thiện khả năng tạo ra dữ liệu giả sao cho chúng khó phân biệt với dữ liệu thật. Discriminator cố gắng cải thiện khả năng phân biệt giữa dữ liệu thật và giả. Quá trình này liên tục lặp lại cho đến khi generator tạo ra dữ liệu giả đủ tốt để đánh lừa discriminator.

Khi quá trình huấn luyện hoàn tất, generator có khả năng tạo ra dữ liệu mới có tính chất và phân phối giống với dữ liệu thật. GANs đã được sử dụng rộng rãi trong nhiều ứng dụng như tạo hình ảnh nghệ thuật, tạo ảnh khuôn mặt giả, cải thiện chất lượng hình ảnh, và nhiều lĩnh vực khác.

**6.4. Mô hình GAN-Unet**

Kết hợp giữa GAN (Generative Adversarial Network) và UNet là một ứng dụng phổ biến trong lĩnh vực xử lý hình ảnh y tế và phân đoạn hình ảnh. Trong mô hình này, UNet được sử dụng như một thành phần Generator trong GAN để tạo ra dữ liệu thật sáng tạo và cải thiện chất lượng phân đoạn hình ảnh.

• Vai trò của UNet trong GAN:

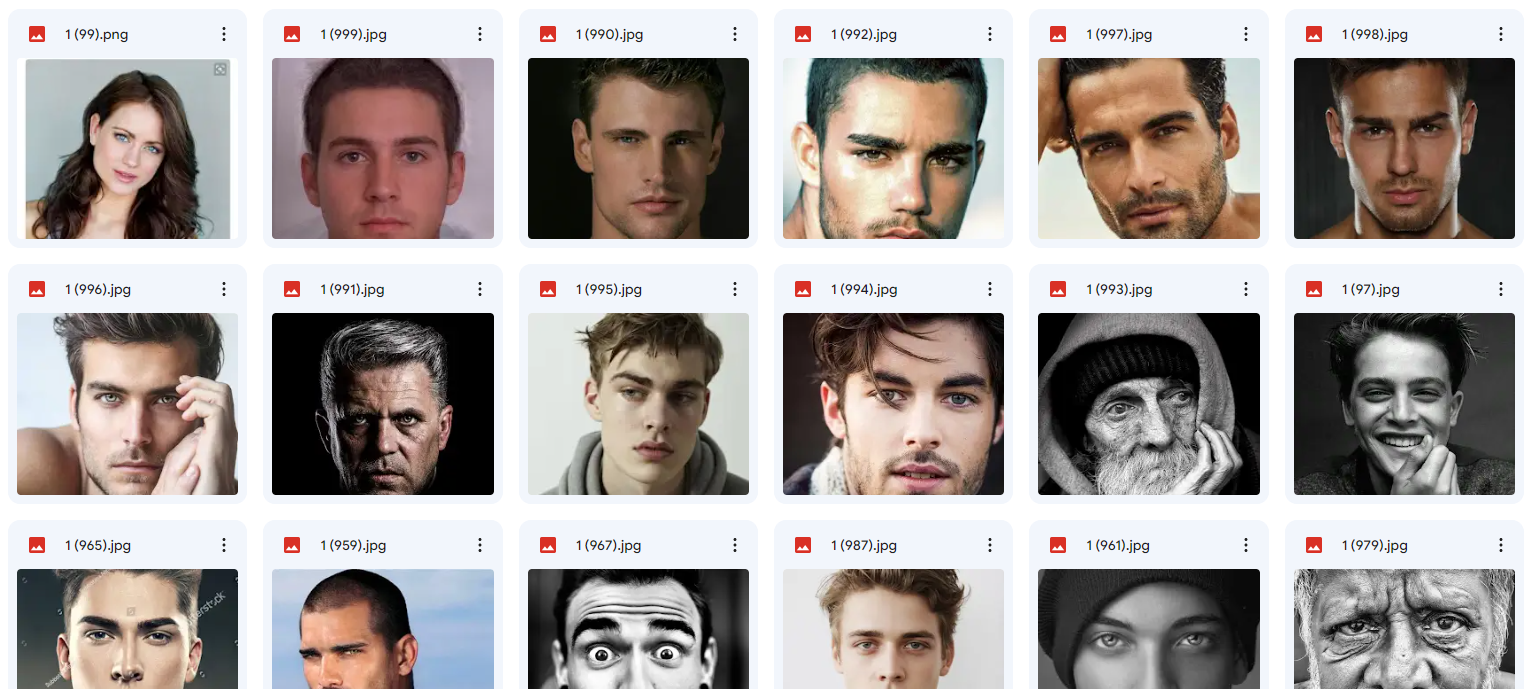
+ Generator (UNet): UNet thường được sử dụng làm Generator trong mô hình GAN. UNet đã được phát triển ban đầu để phân đoạn hình ảnh và có khả năng tạo ra các mask hoặc hình ảnh phân đoạn chất lượng cao. Trong mô hình GAN, UNet sẽ tạo ra dữ liệu giả mạo dựa trên dữ liệu đầu vào, thường là hình ảnh chưa được phân đoạn hoặc dữ liệu y tế, chẳng hạn như hình ảnh MRI hoặc CT.

+ Chất lượng và chính xác: UNet được chọn làm Generator vì khả năng của nó trong việc tạo ra các phân đoạn chính xác và chất lượng cao. GAN cố gắng cải thiện chất lượng của những gì được tạo ra bởi UNet và đồng thời đánh lừa Discriminator.

**7. Triển khai đề tài**

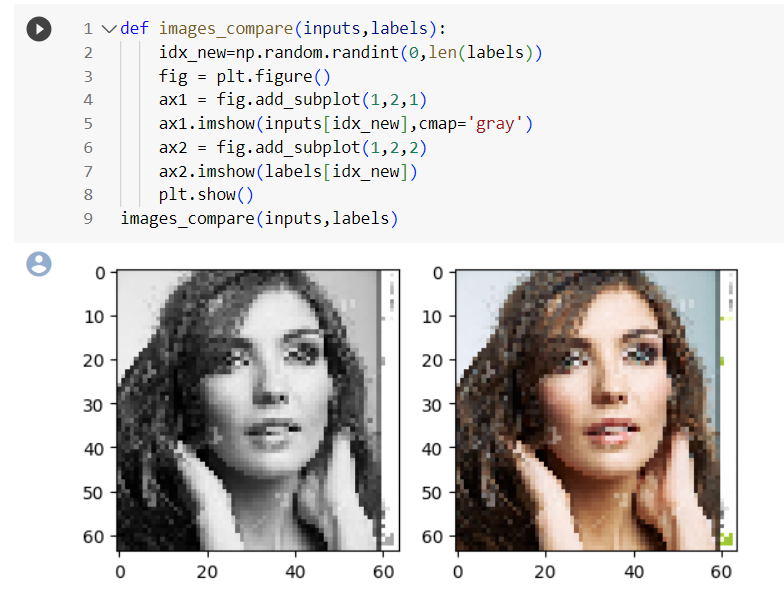
**7.1. Thu thập và xử lý dữ liệu**

Bộ dữ liệu ban đầu là tập hợp các ảnh mặt người được lấy từ bộ Human Faces trên Kaggle với số lượng 5000 tấm ảnh với những khuôn mặt khác nhau, đa dạng về giới tính, độ tuổi và màu da.



1. Một số hình ảnh trong bộ dữ liệu mặt người

Sau đó các hình ảnh này được tải lên Google drive để thuận lợi cho việc đưa vào tiền xử lý và huấn luyện mô hình. Bộ dữ liệu sẽ được chia thành 2 tập Inputs và Labels. Trong đó, tập Inputs sẽ chứa các hình ảnh đen trắng và tập Labels sẽ chứa các hình ảnh gốc (hình ảnh màu) làm nhãn. Mỗi điểm dữ liệu trong tập dữ liệu huấn luyện sẽ bao gồm 1 hình ảnh đen trắng trong tập Inputs và hình ảnh màu tương ứng trong tập Labels.



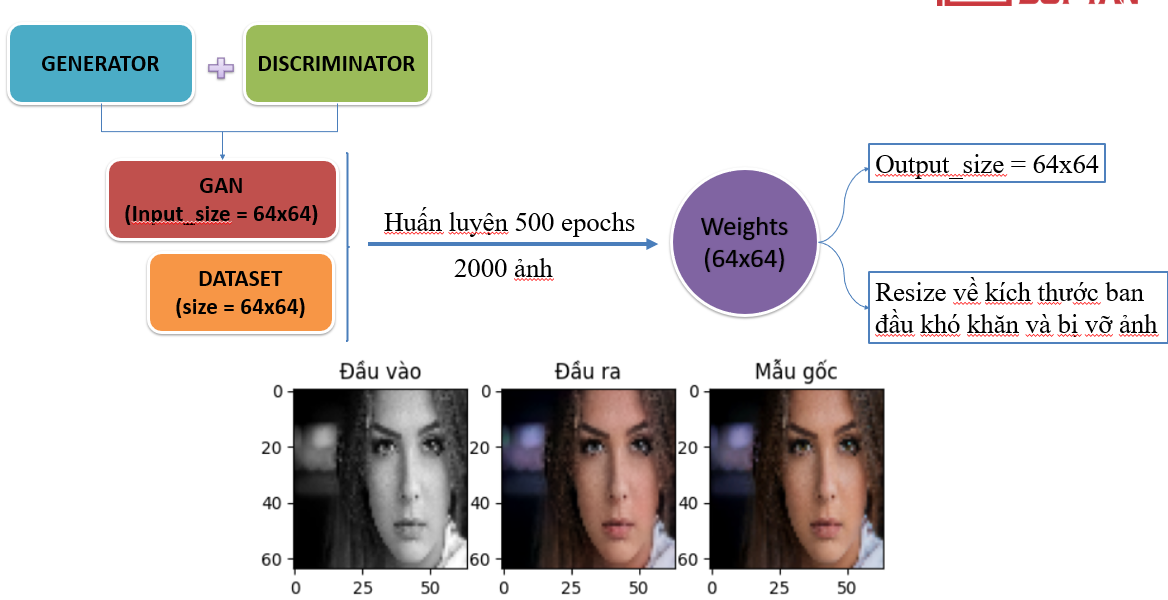
1. Một điểm dữ liệu trong bộ dữ liệu huấn luyện

**7.2. Xây dựng và huấn luyện mô hình GAN-Unet**

Mô hình Generator của mô hình GAN trong dự án này sẽ là một mô hình Unet cơ bản gồm 23 lớp. Mô hình này sẽ có nhiệm vụ sinh ra các ảnh tô màu từ ảnh đen trắng đầu vào.

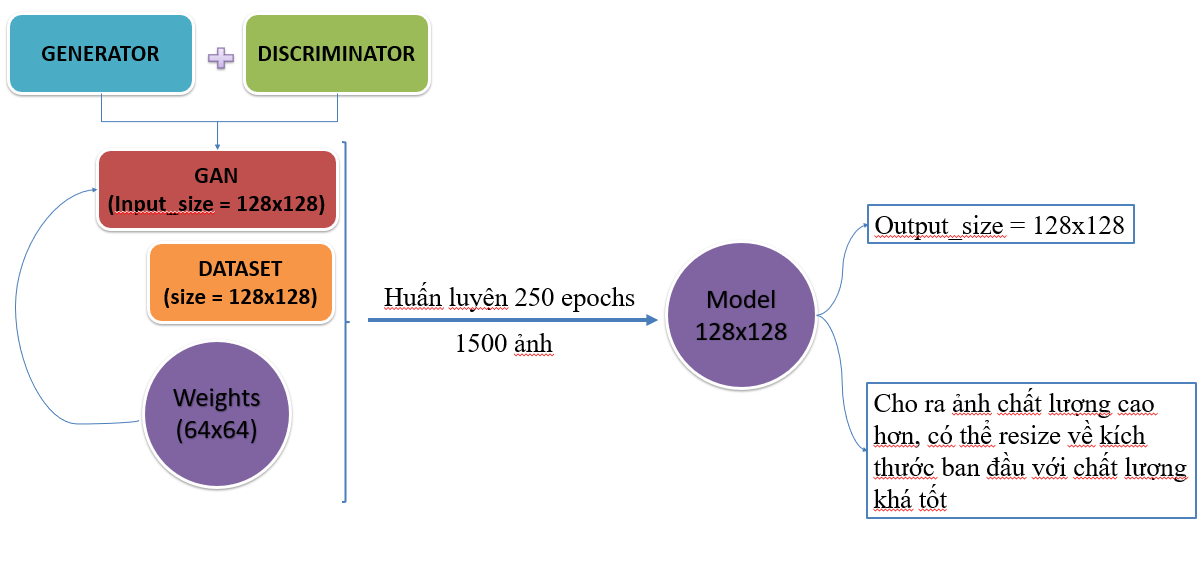
Phần Discriminator sẽ là một mô hình CNN cơ bản có nhiệm vụ phân biệt ảnh thật và ảnh giả sinh ra từ phần Generator

Quá trình huấn luyện mô hình GAN-Unet hoàn toàn diễn ra ở môi trường ảo Google Colab với số lượng hình ảnh huấn luyện là 2000 ảnh, 500 epochs với kích thước đầu vào là 64x64. Lí do số lượng hình ảnh, số epochs và kích thước đầu vào như vậy là do giới hạn về tài nguyên của Google Colab, thông thường một tài khoản Google sẽ được cấp một lượng GPU nhất định, sau khi hết GPU thì quá trình huấn luyện sẽ tự động ngắt, việc này có thể ảnh hưởng đến chất lượng huấn luyện. Vì vậy, việc lựa chọn số lượng dữ liệu và điều chỉnh các thông số để có thể tối ưu hóa tài nguyên là rất quan trọng.



1. Mô hình với đầu vào 64x64

Với kích thước đầu vào chỉ là 64x64 nên đầu ra của mô hình cũng sẽ là 64x64. Đây là một kích thước được coi là rất nhỏ, nó không phù hợp với yêu cầu của bài toán tô màu hình ảnh là phải cho ra một hình ảnh đầu ra chất lượng, sắc nét. Từ kích thước 64x64, chúng tôi đã sử dụng các phương pháp để phục hồi lại kích thước hình ảnh ban đầu. Ví dụ, khi đưa vào mô hình một hình ảnh 1920x1080, nó sẽ được chuyển về kích thước 64x64 để phù hợp với đầu vào của mô hình, sau khi tô màu, hình ảnh kết quả sẽ có kích thước 64x64. Sau đó, hình ảnh màu này sẽ được phục hồi lại kích thước ban đầu, nhưng từ kích thước 64x64 phục hồi về 1920x1080 thì sẽ bị vỡ ảnh, ảnh không được sắc nét, nên chúng tôi đã có một phương pháp khác đó là chúng tôi sẽ nâng kích thước đầu vào của mô hình từ 64x64 lên 128x128, đây là kích thước tối đa để huấn luyện phù hợp với yêu cầu tài nguyên ở Google Colab, đồng thời trọng số của mô hình 64x64 có vai trò trở thành mô hình pre-trained của mô hình mới.

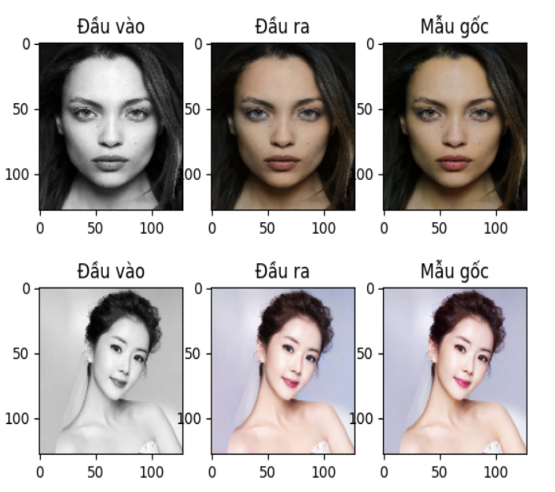


1. Mô hình với đầu vào 128x128

Với việc nâng kích thước đầu vào, chúng tôi buộc phải giảm số lượng hình ảnh huấn luyện và epochs lần lượt xuống còn 1500 và 250 để phù hợp với yêu cầu tài nguyên của Google Colab.

Tới đây, sẽ có một câu hỏi được đặt ra là tại sao ban đầu chúng ta không đặt kích thước là 128x128 và huấn luyện với 1500 hình ảnh qua 250 epochs. Lí do đơn giản là bởi vì như vậy thì kết quả tô màu hình ảnh sẽ không được tối ưu, việc huấn luyện chỉ được 250 epochs thì khả năng tô màu của mô hình sẽ không được tốt, có thể nó vẫn chưa học đủ để có thể thực hiện nhiệm vụ tô màu, ảnh hưởng tới kết quả đầu ra. Cho nên cách làm này sẽ có kết quả không tốt bằng cách làm trên.

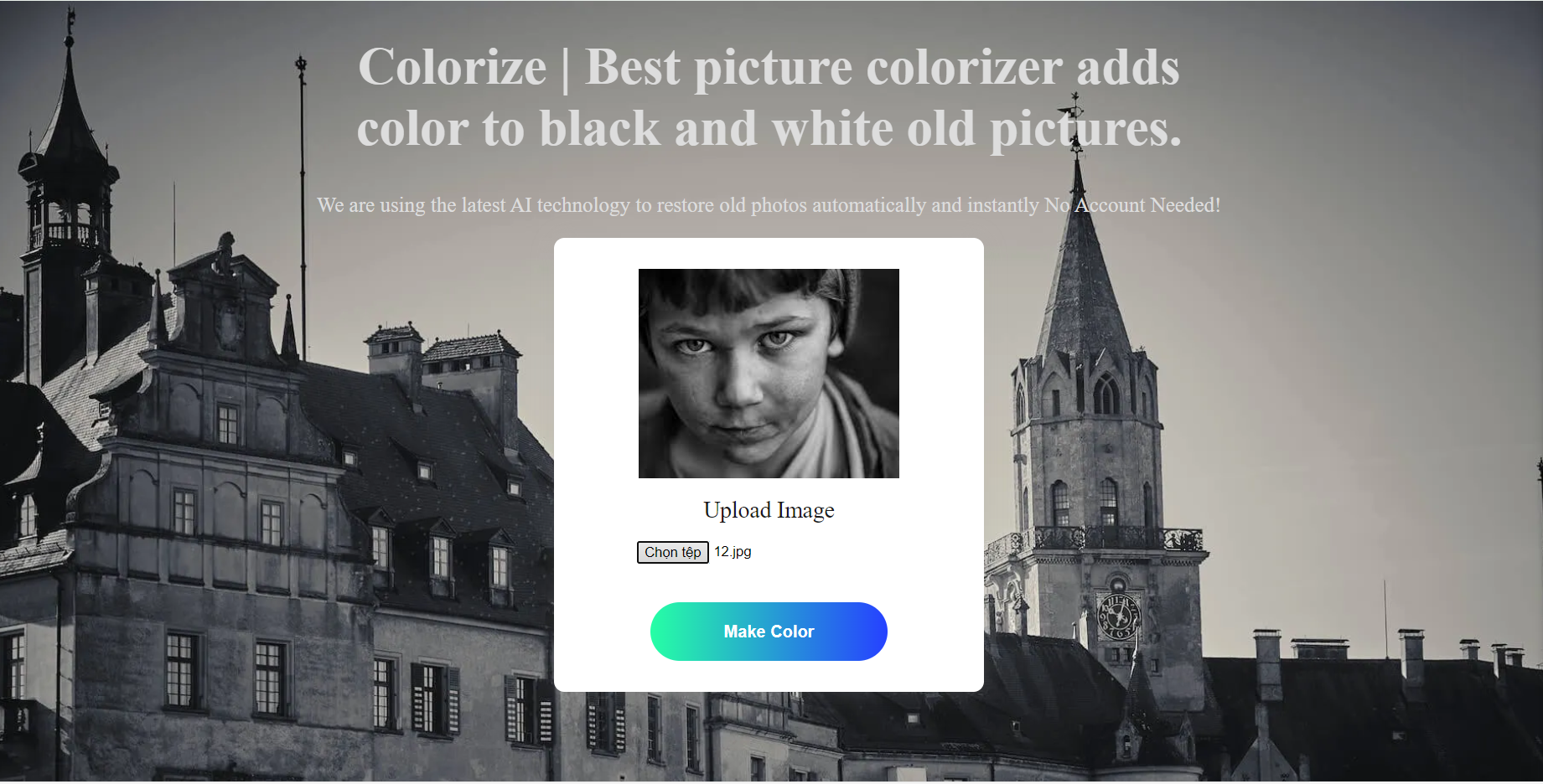
Với kích thước đầu vào và đầu ra là 128x128, việc khôi phục lại về kích thước ban đầu sẽ tốt hơn rất nhiều, hình ảnh sẽ không bị vỡ và không mất nhiều chi tiết, phù hợp với yêu cầu đặt ra của đề tài.



1. Hình ảnh kết quả của mô hình 128x128

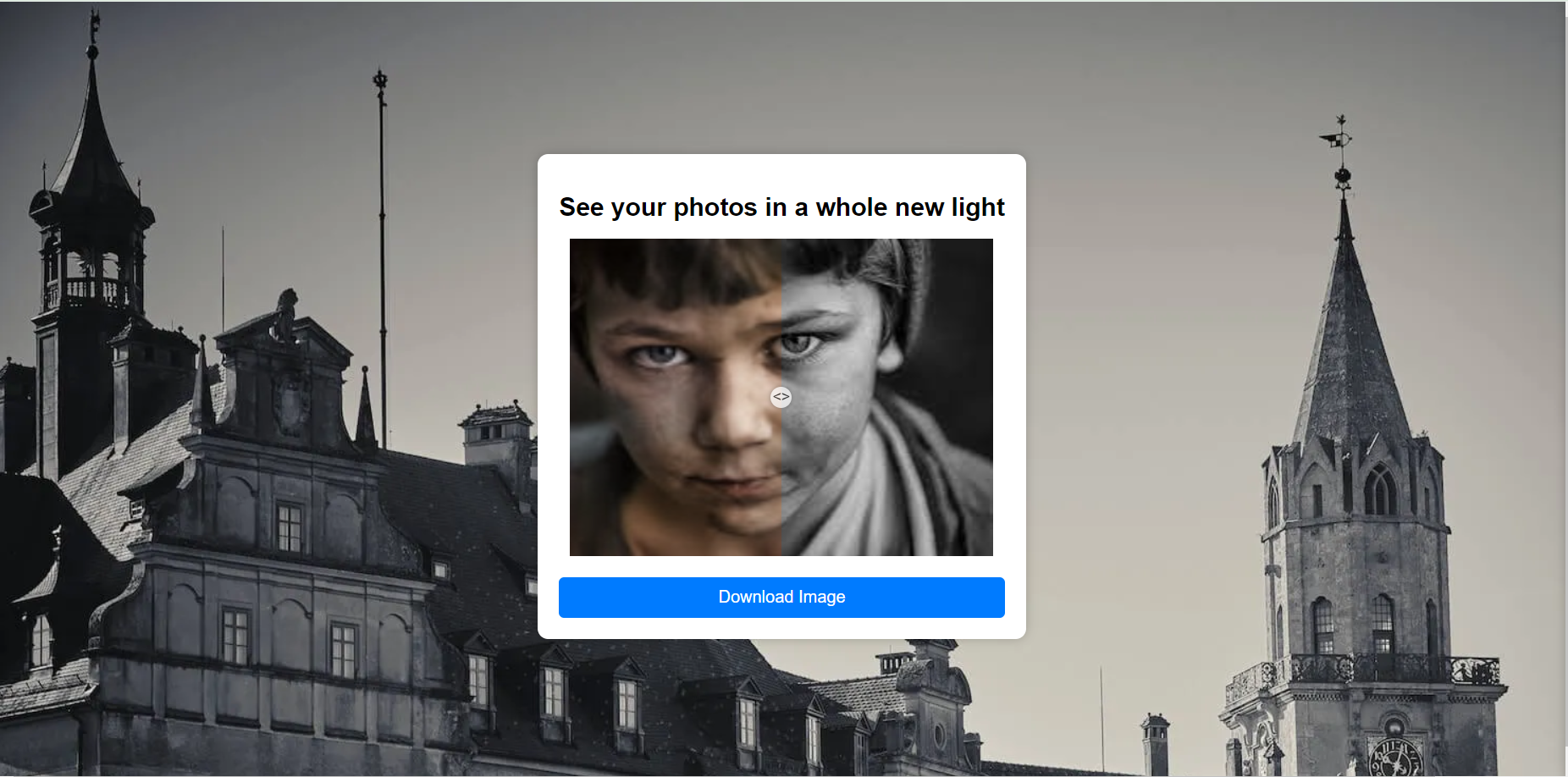
**7.3. Lưu mô hình và nhúng vào website bằng Flask**

Sau khi huấn luyện xong, mô hình được lưu lại dưới dạng Model Keras (model.h5) để có thể tải về máy và sử dụng cho việc nhúng vào giao diện. Sau khi tải về, chúng tôi sẽ sử dụng thư viện Flask để xây dựng một giao diện cơ bản và nhúng model.h5 đã được huấn luyện vào giao diện vừa tạo.



1. Giao diện trang chủ để upload hình ảnh đen trắng cần tô màu

Tại đây, mô hình sẽ tiến hành xử lý và cho ra hình ảnh kết quả dựa vào hình ảnh đầu vào và chuyển sang trang kết quả. Website sẽ cung cấp một thanh trượt để có thể so sánh hình ảnh trước và sau khi tô màu để người dùng có cái nhìn trực quan hơn về hình ảnh kết quả.



1. Giao diện trang kết quả

**8. Kết luận và kiến nghị**

**8.1. Kết luận**

**8.1.1. Mô tả kết quả nghiên cứu**

Thực hiện các kỹ năng tìm kiếm, thu thập và tiền xử lý dữ liệu.

Hiểu được mô hình Unet, cấu trúc mô hình Unet, cách Unet hoạt động và các ứng dụng phổ biến của Unet.

Đã khái quát hóa về mô hình GAN, xây dựng mô hình GAN bao gồm generator và discriminator, tìm hiểu cách mô hình GAN hoạt động và quá trình huấn luyện GAN.

Kết hợp GAN và Unet để phục vụ cho quá trình huấn luyện và tô màu cho hình ảnh.

Xây dựng được website cơ bản có chức năng upload hình ảnh, tô màu hình ảnh, so sánh kết quả và tải hình ảnh kết quả về máy.

**8.1.2. Ứng dụng của kết quả nghiên cứu**

Nghiên cứu này giúp tạo ra công cụ tự động tô màu hình ảnh, giúp tiết kiệm thời gian và công sức cho việc tô màu thủ công, ứng dụng trong việc phục dựng hình ảnh cũ, tạo ra ảnh mô phỏng, và nhiều ứng dụng trong lĩnh vực nghệ thuật và công nghiệp.

Tại Việt Nam, điều kiện áp dụng phụ thuộc vào sự phát triển công nghệ và tài trợ. Các tổ chức nghiên cứu, trường đại học, và các nhiếp ảnh gia có thể áp dụng công trình này để tạo ra sản phẩm và dịch vụ tô màu hình ảnh tự động, hỗ trợ trong lĩnh vực nghệ thuật và truyền thông. Nó cũng có thể được ứng dụng trong việc phục dựng hình ảnh lịch sử và trong lĩnh vực công nghiệp như quảng cáo và thiết kế đồ họa.

**8.1.3. Hạn chế của đề tài**

Vì tài nguyên Google Colab nên bị giới hạn về RAM và GPU, nên chỉ huấn luyện trên một bộ dữ liệu tối đa 2000 ảnh và phải giảm chiều đầu vào về 64x64 hoặc 128x128 để có thể huấn luyện. Đồng thời chỉ huấn luyện được 200-500 epochs tùy vào độ lơn của bộ dữ liệu trong thời gian 2-3 tiếng.

Một số hình ảnh đầu ra vẫn chưa tô màu được tốt.

Mô hình không sử dụng latent variable nên kết quả không đa dạng, chỉ được một mẫu duy nhất.

Kích thước ảnh bị giới hạn, việc resize ảnh về ban đầu vẫn chưa đạt được hiệu quả tối đa.

Việc làm quen với thư viện Flask trong một thời gian ngắn nên website chỉ mới phát triển trên localhost và vẫn chưa có nhiều chức năng.

**8.2. Phần kiến nghị**

Huấn luyện trên bộ dữ liệu lớn và đa dạng hơn. Mở rộng đề tài hình ảnh, không chỉ giới hạn bởi hình ảnh mặt người.

Điều chỉnh quá trình huấn luyện để máy học tốt hơn, cho ra nhiều ảnh đầu ra chất lượng, chân thực và giống với ảnh gốc hơn.

Sử dụng nhiều model khác nhau để tô màu ảnh, qua đó tăng sự lựa chọn đầu ra cho người dùng.

Hoàn thiện trang web và tích hợp nhiều công cụ xử lý ảnh: text2img, img2img, …

**9. Tài liệu tham khảo**

[1] Image Colorization Algorithm Based on Deep Learning - 2022 - Na Wang, Guo-Dong Chen, Ying Tian. (https://www.mdpi.com/2073-8994/14/11/2295)

[2] Convolutional Neural Networks: A Practical Approach - 2019 - Hamed Habibi Aghdam, Elnaz Jahani Heravi. (https://link.springer.com/book/10.1007/978-3-319-57550-6)

[3] Hands-On Image Processing with Tensorflow - 2020 - Soon Yau Cheong. (https://books.google.com.vn/books?id=tGcREAAAQBAJ&pg=PA83&hl=vi&source=gbs\_selected\_pages&cad=1#v=onepage&q&f=false)

[4] Deploy Machine Learning Model using Flask – Geeksforgeeks (https://www.geeksforgeeks.org/deploy-machine-learning-model-using-flask/)

[5] How To Deploy A Machine Learning Model On AWS EC2 | AUG 2021 Updated | ML Model To Flask Website – Youtube Video – Seen on 20/10/2023