ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

VŨ TUẤN HẢI

KHÓA LUẬN TỐT NGHIỆP

PHỤC HỒI ẢNH RĂNG TỪ ẢNH NIỀNG RĂNG CÓ MẮC CÀI SỬ DỤNG GAN

Recover teeth photo from braces photo using GAN

KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM

TP. HỒ CHÍ MINH, 2021

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA CÔNG NGHỆ PHẦN MỀM

VŨ TUẤN HẢI – 17520433

KHÓA LUẬN TỐT NGHIỆP

PHỤC HỒI ẢNH RĂNG TỪ ẢNH NIỀNG RĂNG CÓ MẮC CÀI SỬ DỤNG GAN

Recover teeth photo from braces photo using GAN

KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM

GIẢNG VIÊN HƯỚNG DẪN

PGS.TS. PHẠM THẾ BẢO

ThS. HUỲNH HỒ THỊ MỘNG TRINH

TP. HỒ CHÍ MINH, 2021

THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …… ngày …….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. Chủ tịch.

2. Thư ký.

3. Ủy viên.

4. Ủy viên.

LỜI CẢM ƠN

Lời đầu tiên, tôi xin gửi lời cảm ơn đến quý Thầy Cô khoa Công Nghệ Phần Mềm đã tận tình giảng dạy, truyền đạt những kiến thức quý báu cho tôi trong thời gian học đại học và tạo điều kiện cho tôi thực hiện luận văn này.

PGS.TS. Phạm Thế Bảo, người đã tạo ra một môi trường học thuật thuận lợi, giúp tôi có thể trình bày kết quả nghiên cứu cũng như giải đáp các thắc mắc liên quan đến chuyên môn thứ 7 mỗi tuần.

ThS. Huỳnh Hồ Thị Mộng Trinh, người đã trực tiếp hướng dẫn khóa luận. Trong quá trình thực hiện, cô đã tận tình hướng dẫn, giúp tôi giải quyết các vấn đề nảy sinh trong quá trình làm đề tài.

Cuối cùng, tôi xin được gửi lời cảm ơn tới gia đình là những người động viên và hỗ trợ giúp tôi những lúc khó khăn. Mặc dù tôi đã nỗ lực hết sức để hoàn thành luận văn, song vẫn không thể tránh khỏi những thiếu sót. Vì vậy, tôi rất mong nhận được những đóng góp quý báu của quý Thầy Cô và các bạn.

Xin chân thành cảm ơn.

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của tôi. Các số liệu và kết quả nêu bên trong luận văn là trung thực và chưa từng được công bố trong bất kỳ công trình nào khác, ngoại trừ các tài liệu tham khảo.

TP. Hồ Chí Minh, tháng … năm 2021

Vũ Tuấn Hải

MỤC LỤC

DANH MỤC TỪ TIẾNG ANH

Từ tiếng Anh	Diễn giải
Deep learning	Học sâu, là một phương pháp Machine learning.
Superpixel	Siêu điểm ảnh, là pixel đại diện cho n pixel xung quanh nó .
Mapping	Ánh xạ từ $A^* \rightarrow B^*$ với $f: A \rightarrow B$.
Histogram	Đồ thị biểu thị phân phối màu của ảnh.
GAN	Generative Adversarial Network, mạng đối nghịch tạo sinh.
CNN	Convolutional Neural Network. mạng neural tích chập.
Model collapse	Mô hình sụp đổ. Fake image do Generator sinh ra giống hệt nhau khi generator tìm ra một điểm dữ liệu đặc biệt mà tại điểm đó discriminator không thể phân biệt được. Toàn bộ mô hình không phát triển mặc dù tiếp tục training.
Loss function	Hàm tính độ lỗi của mô hình.
Metric	Số liệu đánh giá mô hình.
SUP	Least upper bound, cận trên nhỏ nhất.
Fine - tuning	Quy trình đánh giả, thử nghiệm và tìm kiếm và lặp lại cho đến khi tìm được hyperparameter tối ưu.

DANH MỤC BẢNG, HÌNH VẼ

TÓM TẮT KHÓA LUẬN

Hình 0.1. Mô hình chuyển đổi từ ảnh niềng răng có mắc cài sang ảnh không còn niềng răng

Những nghiên cứu gần đây về những mô hình học sâu đã cho thấy kết quả đáng mong đợi trong tác vụ loại bỏ vật thể trong ảnh, hay là việc thay thế các đối tượng không mong muốn bằng các pixel thích hợp với ngữ cảnh đã biết. Loại bỏ vật thể dựa trên học sâu đã được giải quyết thông qua những bài toán như Inpainting hay Img2Img (Image to Image translation). Thay vì tìm hiểu trong một lĩnh vực trừu tượng và rộng lớn như vậy, khóa luận này tập trung vào những vấn đề phát sinh khi bài toán áp dụng vào một chuyên ngành cụ thể - nha khoa, đó là xóa bỏ mắc cài trên răng (braces2teeth). Đặc biệt, tôi phân tích hướng giải quyết theo ba hoàn cảnh cụ thể tương ứng với ba loại dataset.

Đầu tiên, tôi sử dụng mô hình CycleGAN với dataset, bao gồm hai tập con là ảnh răng có mắc cài và ảnh răng không có mắc cài. Trong trường hợp thứ hai là mô hình Pix2Pix với dataset là dãy các cặp {ảnh răng có mắc cài, ảnh răng không có mắc cài}. Trong trường hợp cuối cùng, tôi sử dụng mô hình Inpainting (GraphCut kết hợp Contextual Attention) để cho phép người dùng thao tác xóa mắc cài một cách nhanh chóng. Ngoài ra, tôi nâng cao chất lượng ảnh được xử lý bằng cách áp dụng một số phương pháp tiền xử lý ảnh trên.

Theo hiểu biết của tôi, khóa luận này là một trong những nghiên cứu đầu tiên quan tâm, giải quyết bài toán braces2teeth (braces to teeth) bằng các kỹ thuật học sâu.

CHƯƠNG 1. TỔNG QUAN

Nội dung chương 1 trình bày tổng quan về đề tài, bao gồm: động lực nghiên cứu (1.1), mục tiêu đề tài (1.2), đối tượng và phạm vi nghiên cứu (1.3), các đóng góp (1.4) và cuối cùng là bố cục của khóa luận (1.5).

1.1. Động lực nghiên cứu

Xử lý ảnh hay rộng hơn là thị giác máy tính là một trong những lĩnh vực khoa học máy tính có rất nhiều ứng dụng hữu ích hiện nay. Đặc biệt vào khoảng thời gian từ 2010 đến nay, những mô hình học sâu có cơ hội phát triển và được liên tục cải tiến. Số lượng ngành nghề trong đời sống và nhu cầu cần áp dụng những mô hình này ngày càng gia tăng.

Y tế là một trong những ngành cần đến sự trợ giúp của máy tính nhiều nhất, do số lượng dân số gia tăng và tình hình bệnh tật rất phức tạp, trong khi số lượng bác sĩ, y tá không tăng theo nên không thể đáp ứng nhu cầu xã hội. Khi xử lý ảnh được ứng dụng, chúng đã giải phóng sức lao động cho bác sĩ trong nhiều công việc như kê khai đơn thuốc, chẩn đoán bệnh, ...

Nha khoa là lĩnh vực có bệnh nhân thường xuyên dồi dào, vì mỗi người trong đời đều phải gặp nha sĩ nhiều lần để nhổ răng, trám răng hay niềng răng. Đặc biệt, do nhu cầu thẩm mỹ, số lượng người đang niềng răng là rất lớn, nhưng hạn chế của niềng răng là thời gian đeo mắc cài khá lâu, ít nhất là vài tháng đến vài năm. Điều đó khiến nhu cầu giao tiếp, thể hiện bản thân, ... của người đeo mắc cài bị hạn chế. Trong khi nhiều ứng dụng xử lý ảnh hiện tại trong nha khoa chỉ quan tâm đến vấn đề chẩn đoán và chữa trị các bệnh lý răng như sâu răng, nhổ răng, ... bao gồm chẩn đoán, tái tạo hàm với hệ thống CAD/CAM, … mà không để ý đến vấn đề thẩm mỹ của người bệnh. Từ thực tế như vậy,

tôi đã chọn “Phục hồi ảnh răng từ ảnh niềng răng có mắc cài sử dụng GAN” làm chủ để nghiên cứu.

1.2. Mục tiêu đề tài

Mục tiêu đề tài là tạo mô hình xử lý ảnh có đầu ra và đầu vào như sau.

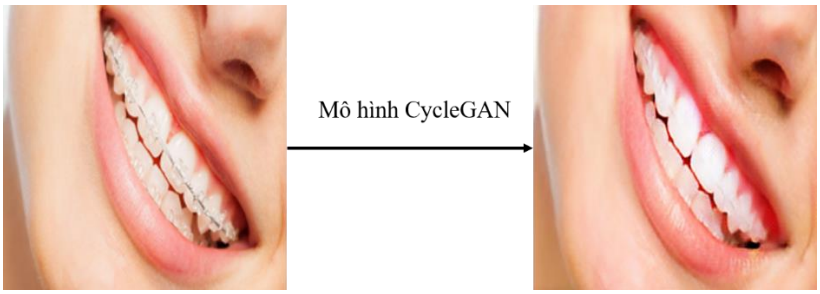
Đầu vào: ảnh niềng răng có mắc cài, mắc cài có thể có màu sắc, hình dạng và chủng loại khác nhau. Mắc cài có thể ở mọi góc độ và mọi vị trí trong ảnh.

Hình 1.1. Ảnh răng có mắc cài. Nguồn: google images

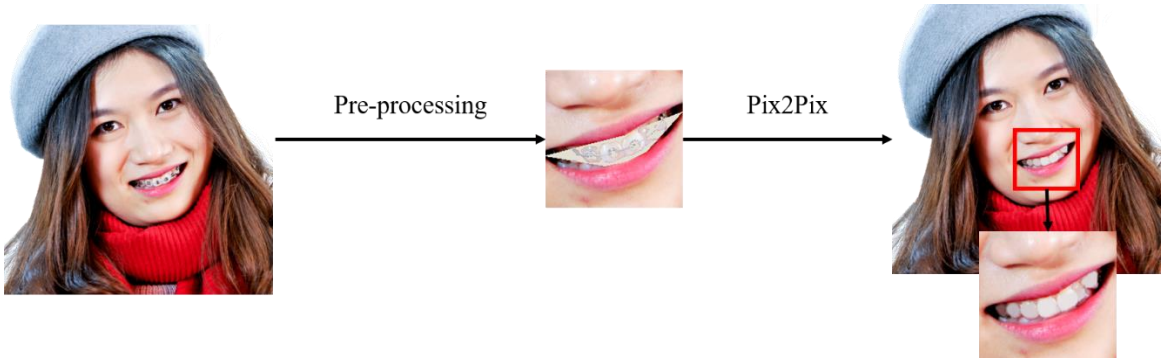
Đầu ra: ảnh răng không còn mắc cài, những vùng mắc cài bị xóa được khôi phục sao cho ảnh chân thật nhất.

Hình 1.2. Ảnh răng không có mắc cài. Nguồn: google images

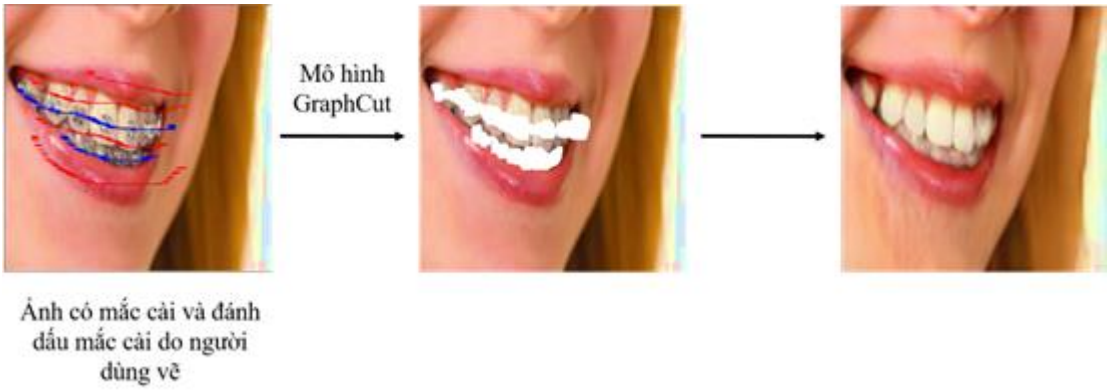
Pipeline: phân tích, đánh giá độ khả thi bằng ba mô hình, mô hình CycleGAN, mô hình Pix2Pix và mô hình Inpainting.



Hình 1.3. Quá trình xử lý của mô hình CycleGAN



Hình 1.4. Quá trình xử lý của mô hình Pix2Pix



Hình 1.5. Quá trình xử lý của mô hình Inpainting

Tuy việc xóa mắc cài hoàn hảo là rất khó nhưng trong giới hạn đề tài khoá luận này, tôi mong rằng ứng dụng có thể giải quyết được những vấn đề đang hiện hữu trong thực tế ở mọi nơi, mọi đối tượng. Từ đó, đáp ứng nhu cầu thẩm mỹ cho người đang niềng răng.

1.3. Đối tượng và phạm vi nghiên cứu

Những ứng dụng làm đẹp như B612, Facebook Filter, … đều ứng dụng các phương pháp chỉnh sửa ảnh tự động, đã xuất hiện trên thị trường rất lâu và cũng có mức độ ảnh hưởng trong đời sống của mỗi người. Tuy nhiên tất cả những ứng dụng, hay nói chính xác hơn là những phương pháp chỉnh sửa ảnh trên tuy giúp người dùng đạt được mục đích, nhưng đều khiến ảnh không còn được chân thật như ban đầu do tính chọn lọc (xác định vùng cần sửa đổi và nội dung cần sửa đổi đã chuẩn bị sẵn) và thay thế. Do đó chúng chỉ đáp ứng được nhu cầu trong trường hợp giải trí mà chưa thể sử dụng trong trường hợp mang tính chất phi giải trí như hội họp, công việc, …

Hình 1.6. Ví dụ về người dùng đang trong quá trình đeo mắc cài (góc dưới bên trái) do đang trong quá trình niềng răng và sử dụng các filter như hiện tại để che giấu nhưng giải pháp này tỏ ra không phù hợp trong nhiều trường hợp. Nguồn: Microsoft Team

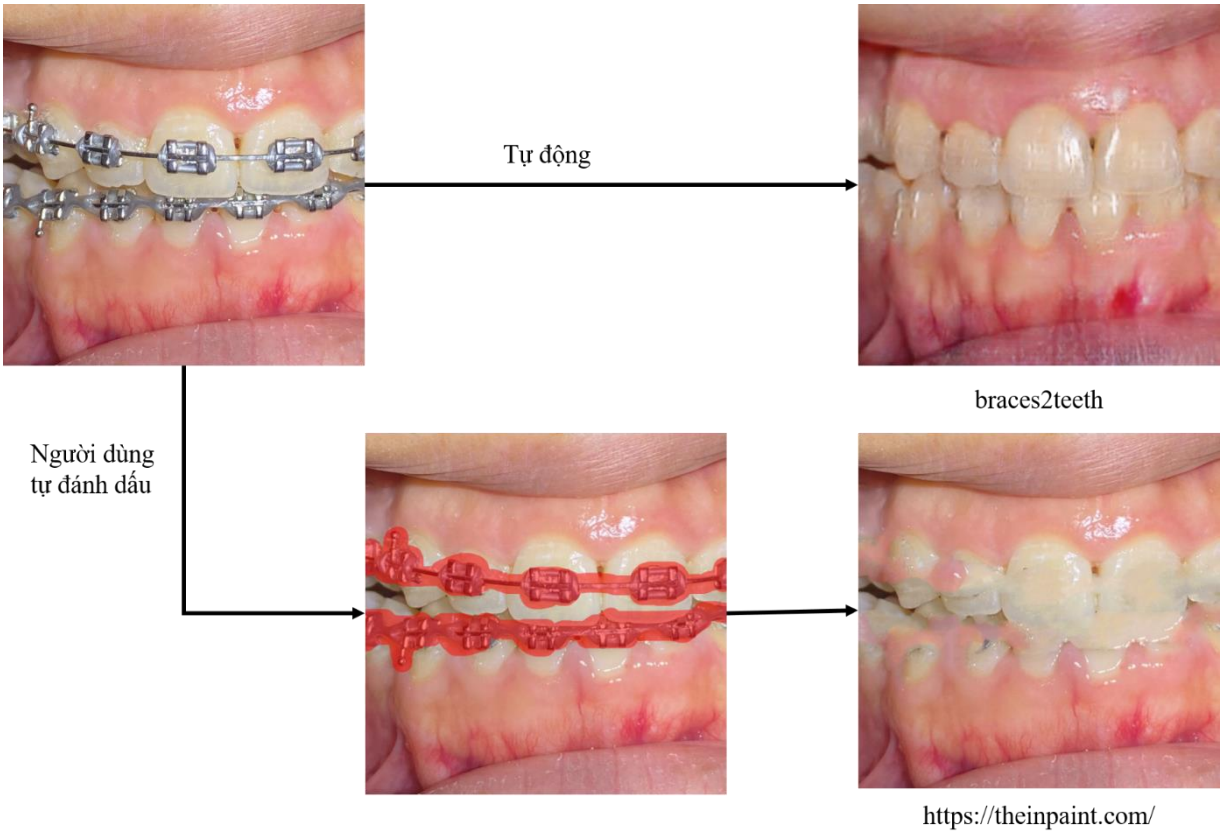
Hình 1.7. Ví dụ khác về những tình huống không thể sử dụng filter, nhưng người dùng vẫn muốn nụ cười của mình hoàn hảo. Nguồn: Facebook

Giải pháp hiện tại để giải quyết nhu cầu là sử dụng những phần mềm chỉnh sửa ảnh như Photoshop để chỉnh sửa trực tiếp, tuy nhiên điều này khá bất tiện với đa số người dùng và cũng không khả thi khi xử lý định dạng video hay trực tuyến.

Do vậy, phương pháp của tôi đề xuất phải có khả năng loại bỏ hoàn toàn mắc cài và tái tạo lại nơi vừa xóa với chất lượng tương đương những thợ chỉnh sửa ảnh chuyên nghiệp.

Trong những năm gần đây, đã có nhiều nghiên cứu liên quan đến tác vụ xóa vật thể không mong muốn ra khỏi ảnh và khôi phục lại vùng đã bị xóa như [1], [2] và [3]. Tuy nhiên, những phương pháp này còn đang gặp những hạn chế như phải trích xuất đặc trưng thủ công. Một trong số đó là PatchMatch (2009) đã có ý tưởng sáng tạo về việc tìm kiếm những vùng ảnh có sẵn trong ảnh để điền vào chỗ trống, mô hình này khá tốt với những ảnh đơn giản tuy nhiên vẫn không xử lý được những ảnh có kết cấu màu sắc phức tạp như mắc cài. Một số phương pháp sử dụng GAN như [4] và [14], … đã đạt được nhiều kết quả tốt trên các bộ dữ liệu phổ biến như phong cảnh, xe cộ, nhà cửa và có khả năng có kết quả tương tự trên bộ dữ liệu mắc cài. Do đó, trong khóa luận này, tôi

sẽ đánh giá việc sử dụng mô hình GAN trong bài toán braces2teeth kết hợp với một số phương pháp khác.



Hình 1.8. So sánh kết quả của mô hình được trình bày trong khóa luận (trên) và dịch vụ xóa mắc cài có sẵn (dưới).

1.4. Đóng góp khóa luận

Khóa luận đã đạt được nội dung sau:

- Đây là nghiên cứu đầu tiên trình bày phương pháp giải quyết bài toán braces2teeth.
- Thu thập bộ dữ liệu braces2teeth bao gồm 1704 ảnh về răng và 2165 hình ảnh về mắc cài. Ngoài ra còn có hai bộ dữ liệu khác là braces2teeth Augmented và teeth2.
- Xây dựng mô hình chuyển đổi từ ảnh có mắc cài sang ảnh răng bằng mô hình Pix2Pix (cho tập dữ liệu cặp) và CycleGAN (cho tập dữ liệu không theo cặp).
- Đề xuất mô hình cho phép chỉnh sửa ảnh nhanh chóng bằng Inpainting.

- Xây dựng ứng dụng web sử dụng các mô hình trên.
- Tạo ra một mô hình mở để cộng đồng có thể sử dụng kết quả nghiên cứu và phát triển ứng dụng khác theo nhu cầu.
- Công bố khoa học:
  1. Kỷ yếu Hội nghị AI gặp gỡ lãnh đạo Thành phố Hồ Chí Minh, Việt Nam, 6/11/2020 (Phụ lục 1).
  2. Tái tạo ảnh răng từ ảnh niềng răng có mắc cài sử dụng GAN, Hội nghị khoa học Trẻ & nghiên cứu sinh UIT, Việt Nam, 18/11/2020 (Phụ lục 1).
  3. Kỷ yếu chung kết Eureka lĩnh vực Công nghệ thông tin 2020, Việt Nam, 27/11/2020 (Phụ lục 2).
  4. Reconstructed teeth images from braces using GAN, Biomedical Engineering: Applications, Basis and Communications (BME) [Q4], Singapore, 2021 (Phụ lục 3).

1.5. Bố cục khóa luận

Khóa luận bao gồm 7 chương:

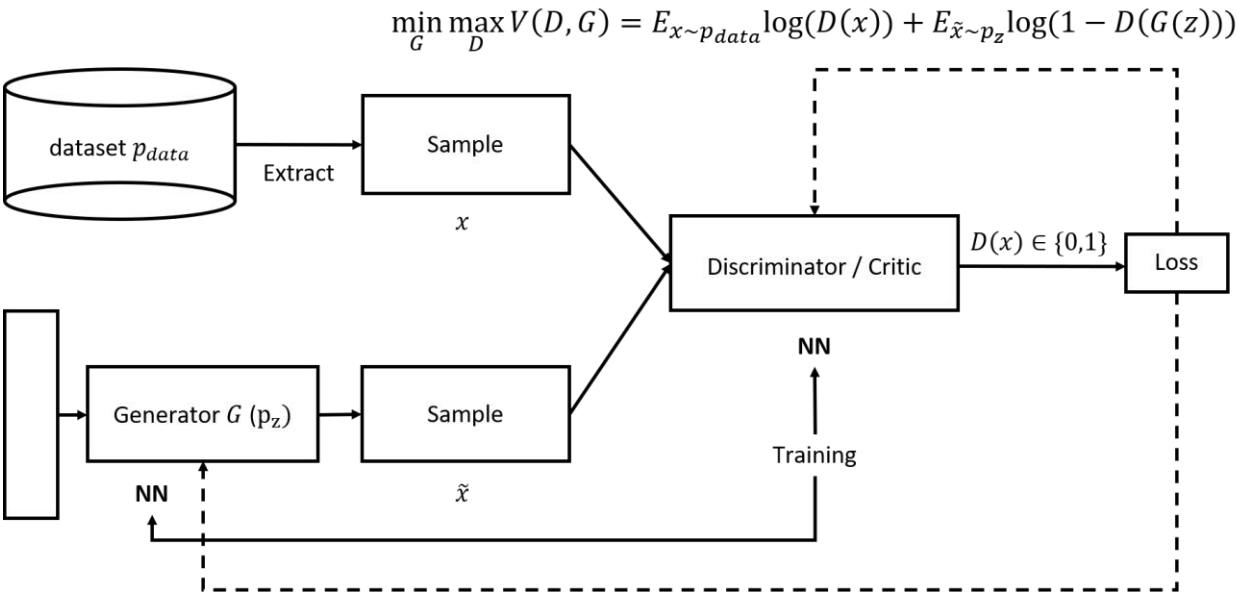
- Chương 1: Tổng quan
- Chương 2: Cơ sở lý thuyết
- Chương 3: Bài toán braces2teeth
- Chương 4: Thực nghiệm
- Chương 5: Đánh giá
- Chương 6: Ứng dụng minh họa
- Chương 7: Kết luận.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Tôi xác định cơ sở lý thuyết theo ba khía cạnh. Đầu tiên là một số phương pháp giải quyết bài toán Img2Img (image to image) cơ bản (2.1). Thứ hai là các nghiên cứu đã được cải tiến từ những mô hình cơ bản để sinh ra ảnh có chất lượng cao trong một số ngữ cảnh phổ biến về mặt người, phong cảnh, ... (2.2). Cuối cùng là các công trình được nghiên cứu để giải quyết bài toán Img2Img theo cách gián tiếp bằng cách loại bỏ đối tượng không mong muốn và khôi phục lại ảnh (2.3).

2.1. Bài toán Img2Img

2.1.1. GAN



Hình 2.1. Sơ đồ tổng quát của mô hình GAN. Nguồn: [7]

GAN [7] là một mô hình cổ điển được giới thiệu bởi Ian J. Goodfellow vào năm 2014 đã đạt được rất nhiều thành công lớn trong deep learning nói riêng và AI nói chung. Một số ứng dụng nổi bật của GAN như deep fake, image editing, generate realistic photographs, super resolution, text to image, …

GAN là học một ánh xạ từ vectơ noise ngẫu nhiên z đến ảnh y, kí hiệu $G: z \rightarrow y$ [7]. Ngược lại, conditional GAN (cGAN) [16] học một ánh xạ từ ảnh x và vectơ noise ngẫu nhiên z tới y, $G: \{x, z\} \rightarrow y$. Generator G được huấn luyện để tạo ra đầu ra không thể phân biệt được với dữ liệu thực bởi Discriminator D, là một mạng classification được huấn luyện để phát hiện dữ liệu giả do G sinh ra. Cả 2 mạng đều được huấn luyện song song hoặc tuần tự cho đến khi đạt trạng thái cân bằng Nash (Nash equilibrium).

Giả sử p là phân phối xác suất tập ảnh thật và q là phân phối xác suất tập ảnh giả. Để đánh giá mô hình tốt hay không (mức độ chân thật của ảnh sinh ra), tôi sử dụng một số metric là phân kỳ Kullback - Leibler (KL divergence):

$$D_{KL}(P||Q) = \sum_{x=1}^{N} P(x) \log \left(\frac{P(x)}{Q(x)}\right) \tag{2.1}$$

hoặc phân kỳ Jensen - Shannon (JS divergence):

$$D_{JS}(P||Q) = \frac{1}{2} D_{KL}\left(P \middle| \frac{P+Q}{2}\right) + \frac{1}{2} D_{KL}\left(Q \middle| \frac{P+Q}{2}\right) \tag{2.2}$$

với mục tiêu là $D_{KL}(P||Q)$ hoặc $D_{JS}(P||Q)$ xấp xỉ 0. Tuy nhiên, khi kỳ vọng (mean) của q tăng (một nửa mô hình GAN đang hội tụ) thì giá trị phân kỳ tăng, đạo hàm giảm. G cập nhật rất ít hoặc hầu như không cập nhật từ gradient descent, hiện tượng này được gọi generator diminished.

Một vấn đề khác, loss function nguyên thủy (trong khóa luận đầu tiên về GAN) làm mất cân bằng quá trình training (D thường ổn định hơn với G ở giai đoạn đầu). Khi q quá xa so với p:

$$-\nabla_\theta \log\left(1 - D\left(G(z^i)\right)\right) \rightarrow 0 \ (log 1 = 0). \tag{2.3}$$

Khiến G sẽ khó hội tụ, đây cũng là một trong những nguyên nhân làm giảm chất lượng ảnh. Nói chung, GAN vẫn là một mô hình cơ bản nên trong giai đoạn từ năm 2014 đến nay, đã có nhiều nghiên cứu cải thiện chất lượng ảnh sinh ra lên rất nhiều.

2.1.2. DCGAN

Với nhiệm vụ liên quan tới xử lý ảnh, mô hình CNN được kết hợp với GAN để chất lượng ảnh được sinh ra tốt hơn. Mô hình này được gọi là DCGAN [13]. Một số đặc điểm DCGAN khác với CNN truyền thống như sử dụng stride - 2 conv layer thay cho max pooling layer, sử dụng global average pooling thay cho fully connected layer.

2.1.3. WGAN

Vấn đề trong quá trình training GAN là generator diminishes và mô hình collapse (ngược lại với generator diminishes) khiến mô hình khó hội tụ. Để khắc phục một phần vấn đề này, [9] đã đề xuất loss function mới trong mô hình Wasserstein GAN (WGAN) [14].

Wasserstein W (hay Kantorovich – Rubinstein), giống như phân kỳ KL và phân kỳ JS là một metric đánh giá sự khác biệt giữa 2 phân phối xác suất P_r (tập ảnh thật) và P_g (tập ảnh giả) nhưng có một đặc điểm quan trọng là đạo hàm mượt tại mọi điểm. Trong loss function mới, mục tiêu cuối cùng của mạng là cố gắng sao cho $W(P_r, P_g)$ thấp nhất có thể (hay generator sinh ra ảnh có chất lượng gần như ảnh thật nhất).

Để tính toán W theo công thức gốc sẽ phát sinh nhiều chi phí nên tác giả đã đề xuất sử dụng đối ngẫu Kantorovich - Rubinstein để đơn giản hóa công thức gốc thành:

$$W(P_r, P_g) = \sup_{D \in \mathcal{D}} E_{x \sim P_r}[D(x)] - E_{\tilde{x} \sim P_g}[D(\tilde{x})] \tag{2.4}$$

Trong đó sup là cận trên nhỏ nhất và $\mathcal{D}$ là tập 1 - Lipschitz function (thỏa điều kiện $|f(x_1) - f(x_2)| \leq |x_1 - x_2|$). Như vậy mục tiêu mới của tôi là tìm (mô phỏng) f bằng deep neural network F nào đó với layer cuối có đầu ra là vô hướng, tượng trưng cho việc đánh giá chất lượng ảnh x.

Để F hội tụ, tôi giới hạn weight của F (F_θ) trong miền compat $(-c, c)$ với c là hyperparameter.

$$w \leftarrow clip(w, -c, c) \tag{2.5}$$

Thực nghiệm cho thấy c nằm ở khoảng gần với 0.01. Tuy nhiên với mô hình khác hoặc dataset khác c sẽ phải thay đổi. Như vậy, tôi phải fine – tuning để có được c. Nhược điểm này sẽ được khắc phục trong WGAN - GP.

2.1.4. WGAN - GP

GP là từ viết tắt của gradient penalty. Lấy $\hat{x} = \varepsilon x + (1 - \varepsilon)\tilde{x}$ là phân phối mẫu, tôi ép đạo hàm của 1 - Lipschitz function D nhỏ hơn 1 bằng cách cộng vào $W(P_r, P_g)$ lượng:

$$\lambda \left(\left|\left| \nabla_{\hat{x}} D_w(\hat{x}) \right|\right|_2 \odot (1 - m) - 1 \right)^2, \lambda = 10 \text{ (m = 0 nếu là mask)} \tag{2.6}$$

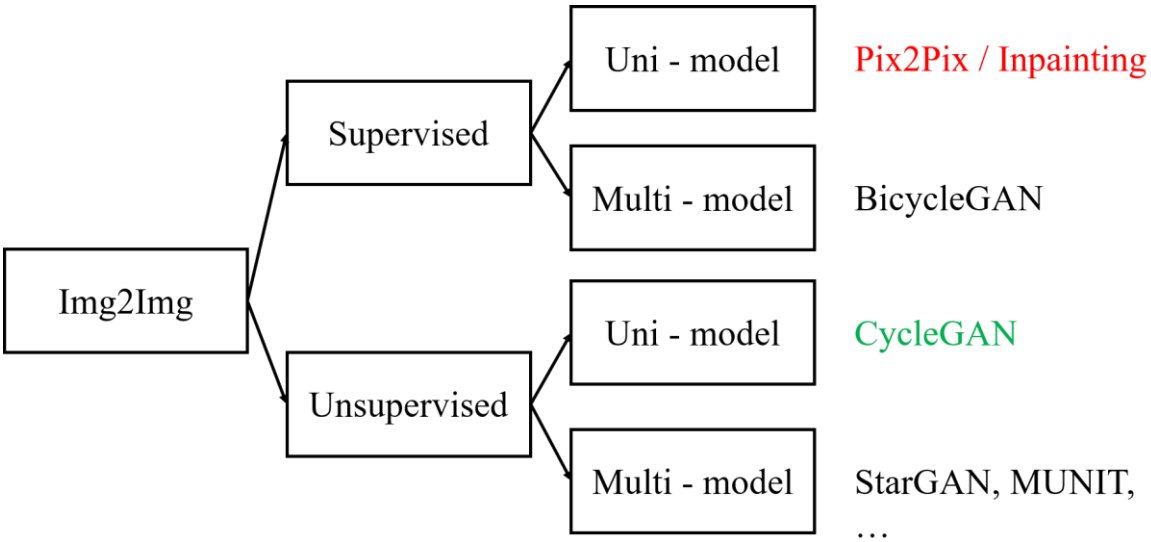
Giải thuật cuối cùng của mô hình:

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.
while θ has not converged **do**
  for $t = 1, ..., n_{\text{critic}}$ **do**
    for $i = 1, ..., m$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, a random number $\epsilon \sim U[0, 1]$.
      $\tilde{\boldsymbol{x}} \leftarrow G_\theta(\boldsymbol{z})$
      $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1 - \epsilon)\tilde{\boldsymbol{x}}$
      $L^{(i)} \leftarrow D_w(\tilde{\boldsymbol{x}}) - D_w(\boldsymbol{x}) + \lambda(\|\nabla_{\hat{\boldsymbol{x}}} D_w(\hat{\boldsymbol{x}})\|_2 - 1)^2$
    end for
    $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$
  end for
  Sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^m \sim p(\boldsymbol{z})$.
  $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^m -D_w(G_\theta(\boldsymbol{z})), \theta, \alpha, \beta_1, \beta_2)$
end while

Hình 2.2. Mã giả của mô hình WGAN – GP. Nguồn: [14]

2.1.5. Các mô hình hiện đại giải quyết bài toán Img2Img



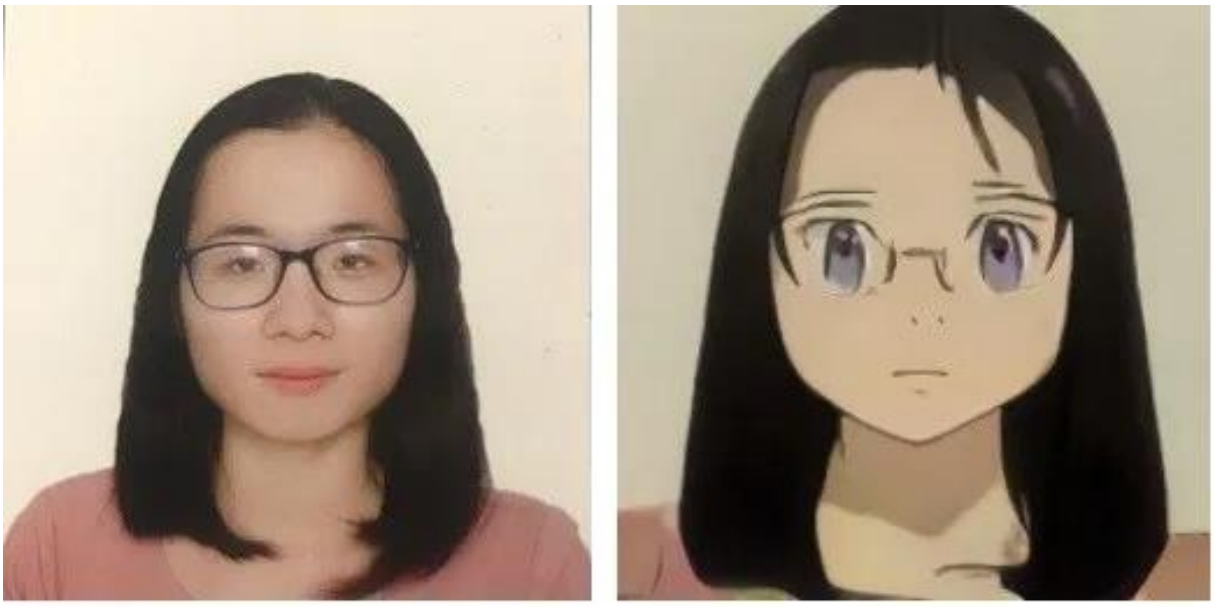
Hình 2.3. Các biến thể của GAN. Nguồn: https://nttuan8.com/

Về mặt tổng quát, bài toán Img2Img là bài toán nếu cho đầu vào là một ảnh thì mô hình sẽ cho ra một hoặc nhiều ảnh tương ứng.

Với yêu cầu một đầu ra tương ứng (uni – model) thì có lớp các mô hình giám sát (Pix2Pix) và không giám sát (CycleGAN), một số bài toán tiêu biểu của uni - model là chuyển từ ảnh xám sang ảnh màu, ảnh chụp sang tranh vẽ, hay ảnh chụp sang ảnh hoạt hình.

Hình 2.4. Bài toán chuyển từ ảnh xám sang ảnh màu. Nguồn: https://nttuan8.com/

Hình 2.5. Bài toán chuyển từ ảnh thật sang tranh vẽ. Nguồn: [14]

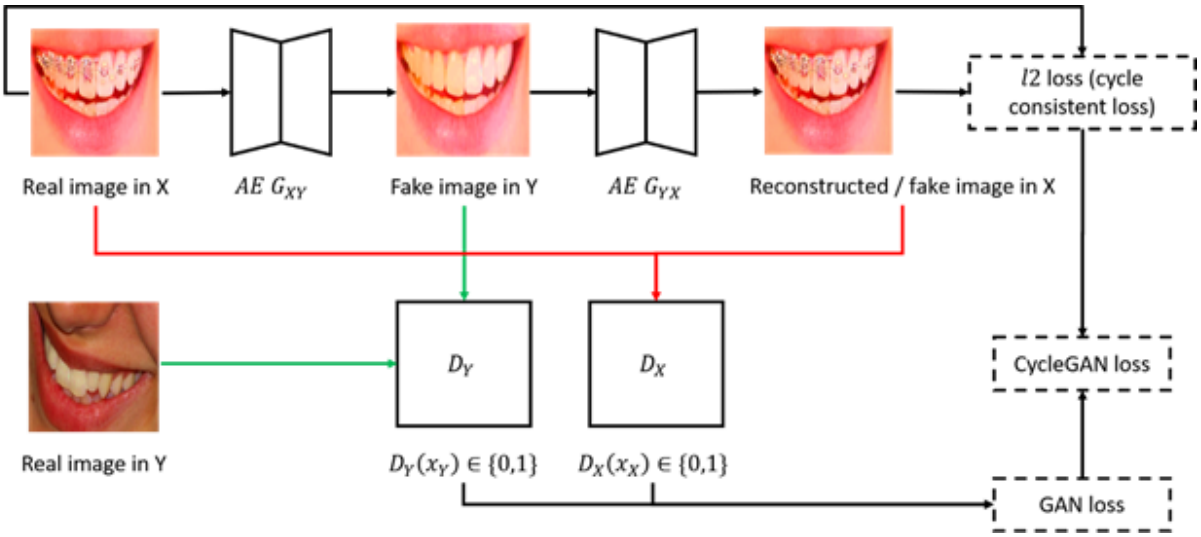


Hình 2.6. . Bài toán chuyển từ ảnh thật sang ảnh hoạt hình. Nguồn: https://nttuan8.com

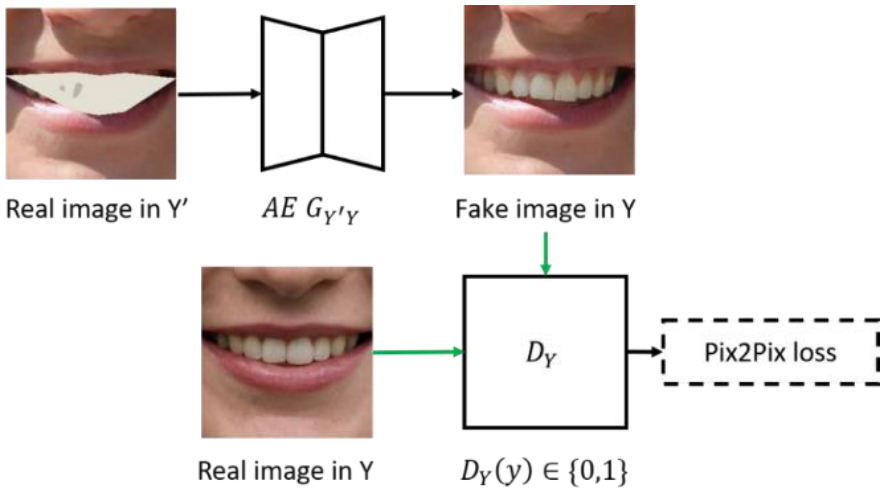
Với yêu cầu nhiều đầu ra theo các điều kiện khác nhau (multi - model), cũng có lớp các mô hình giám sát (BicycleGAN) và không giám sát (StarGAN, MUNIT, …), bài toán tiêu biểu của multi - mô hình là nhận vào một ảnh nhưng cho ra nhiều ảnh theo nhiều ngữ cảnh / điều kiện khác nhau.

Về bài toán braces2teeth, mục tiêu của chúng ta là học một ánh xạ f chuyển đổi giữa hai domain (uni – model), ảnh răng mắc cài (X) và ảnh răng không có mắc cài (Y) với các

mẫu cho trước $x_{i=1}^{n}$ ($x_i \in X$) và $y_{j=1}^{m}$ ($y_j \in Y$) dùng để huấn luyện nên tôi tập trung đến mô hình Pix2Pix và CycleGAN.



Hình 2.7. Mô hình tổng quát của CycleGAN, bao gồm hai Generator G_XY,G_YX và hai Discriminator D_X,D_Y. Nguồn: [14]



Hình 2.8. Mô hình tổng quát của Pix2Pix, bao gồm một Generator $G_{Y'Y}$ và một Discriminator D_Y, $G_{YY'}$ là ánh xạ Y'→Y trong khi đó D_Y là mạng classification giữa Y và Y_{fake}. Nguồn: [15]

Trong mô hình Pix2Pix, generator được huấn luyện để tạo ra ảnh trong domain đích tương ứng với các ảnh đầu vào trong domain nguồn. Tuy nhiên, Pix2Pix phải được cung cấp các mẫu dữ liệu theo từng cặp đầu ra – đầu vào tương ứng. Để giải quyết hạn chế

này, CycleGAN [14] bằng cách sử dụng cycle consistency loss, được sinh ra từ ý tưởng dựa trên thực tế là ảnh phải được tái tạo chính xác sau khi thực hiện hai phép biến đổi liên tiếp với phép biến đổi thứ hai ngược với phép biến đổi thứ nhất. Mô hình này giúp các mô hình img2img có thể được huấn luyện bằng hai dataset riêng biệt (không cần các cặp tương ứng). Tuy nhiên, sự thật là kết quả của phương pháp này không tốt bằng Pix2Pix.

2.1.5.1. CycleGAN

CycleGAN [14] là một lựa chọn tốt để giải quyết bài toán này vì f được thiết kế để có hiệu suất tốt nhất đối với tác vụ thay đổi về bề ngoài, không phải đối với các thay đổi về hình dạng. Vì có cùng mục tiêu là xóa bỏ mắc cài trên răng (không phải thay đổi hình dạng răng) nên tôi chọn CycleGAN làm mô hình đầu tiên để thử nghiệm.

CycleGAN có hai ánh xạ (generator) $G: X \rightarrow Y$ và $F: Y \rightarrow X$ và hai discriminator D_X, D_Y , với D_X dùng để phân biệt ảnh thật x và ảnh giả $F(y)$; tương tự, D_Y dùng để phân biệt ảnh thật y và ảnh giả $G(x)$. Mục tiêu của CycleGAN bao gồm hai phần: adversarial loss [7] đánh giá mức độ trùng khớp giữa hai phân phối của ảnh thật trong domain đích và ảnh giả do generator sinh ra; và cycle consistency losses dùng để ngăn hai ánh xạ G và F mâu thuẫn với nhau.

CycleGAN sử dụng adversarial losses [7] cho cả hai ánh xạ. Đối với ánh xạ $G: X \rightarrow Y$ và discriminator tương ứng D_Y, ta có hàm loss như sau:

$$\begin{aligned} L_{GAN}(G, D_Y, X, Y) \\ &= E_{y \sim p_{data}(y)} [log D_Y(y)] \\ &+ E_{x \sim p_{data}(x)} [log(1 - D_y(G(x))] \end{aligned} \tag{2.7}$$

Trong đó G sinh ra ảnh răng không có mắc cài giả $G(x)$ giống như ảnh thật y trong domain Y và D_Y phân biệt có nhiệm vụ phân biệt $G(x)$ và y. G có nhiệm vụ tối thiểu hóa

ảnh giả bị phát hiện trong khi D_Y lại cố gắng tối đa hóa số trường hợp này. CycleGAN cũng có hàm adversarial loss tương tự đối với ánh xạ $F: Y \rightarrow X$ và discriminator D_X.

Hai ánh xạ G và F có thể tạo ra đầu ra có phân phối giống hệt như trong domain đích Y và X tương ứng. Tuy nhiên, khi số lượng ảnh dùng để huấn luyện đủ lớn, có rất nhiều ánh xạ thỏa mãn adversarial loss là sinh ra phân phối trùng với mẫu cho trước mà không thể đảm bảo rằng ánh xạ đã học có thể biến đổi một đầu vào đơn lẻ x_i tới đầu ra mong muốn y_i. Để giảm thêm không gian của các ánh xạ khả dĩ, các tác giả của CycleGAN lập luận rằng các ánh xạ phải nhất quán với chu trình sau, với mỗi phần tử x từ domain X, chu kỳ chuyển đổi có thể đưa x về ban đầu, $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$. Tương tự, với mới phần tử y từ domain Y, G và F cũng phải thỏa mãn tính nhất quán của chu trình: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. Ta có cycle consistency loss là:

$$Lcyc(G,F) = E_{x\sim pdata(x)}[F(G(x)) - x] + E_{y\sim pdata(y)}[G(F(y)) - y]. \quad (2.8)$$

Hàm loss đầy đủ của CycleGAN:

$$L(G,F,D_X,D_Y,X,Y) = L_{GAN}(G,D_Y,X,Y) + L_{GAN}(F,D_X,X,Y) + \lambda L_{cyc}(G,F) \quad (2.9)$$

Với hệ số λ = 10 là hệ số tối ưu đã được thực nghiệm trong [14].

2.1.5.2. Pix2Pix

Pix2Pix [15] đã được chứng minh là có chất lượng ảnh sinh ra tốt hơn CycleGAN trong [14]. Tuy nhiên, việc chuẩn bị dataset bao gồm các cặp như hình 1 là điều rất khó khăn và hết sức tốn kém. Ảnh chụp trong quá trình niềng răng và sau khi tháo niềng / trước khi đeo niềng thường không khác nhau do khác góc độ chụp hoặc do răng của người niềng đã có nhiều thay đổi.

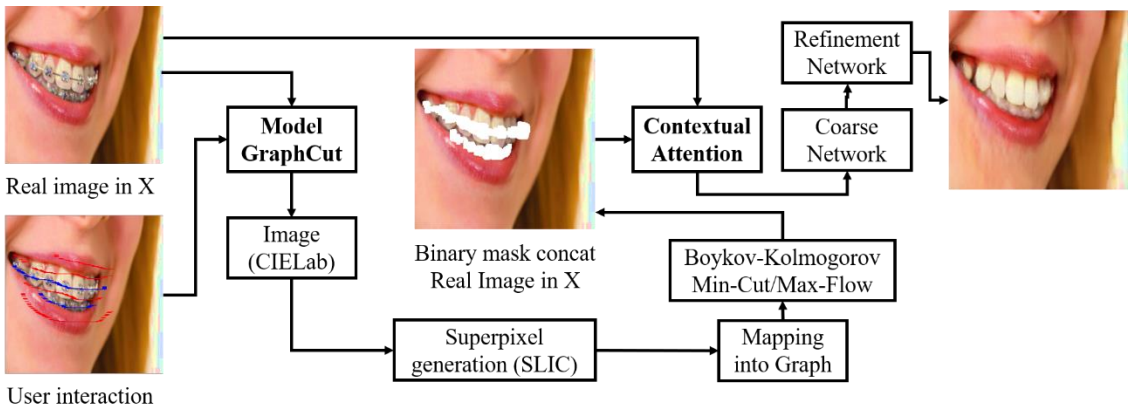
2.2. Kỹ thuật tăng cường ảnh cho bài toán Img2Img

Các công trình gần đây đã cố gắng cải thiện kết quả của các mô hình trong một số bài toán như ariel2map, ... [23] tập trung vào tác động của các lớp latent và sửa đổi kiến trúc của mô hình để khiến chất lượng ảnh sinh ra được cải tiến. Những nghiên cứu khác như [24] đề xuất hướng bổ sung các thành phần phụ trợ như attention layer vào mô hình. Ngoài ra còn rất nhiều kỹ thuật khác được tôi lấy cảm hứng để cải thiện kết quả trong bài toán braces2teeth sẽ được trình bày trong chương 3.

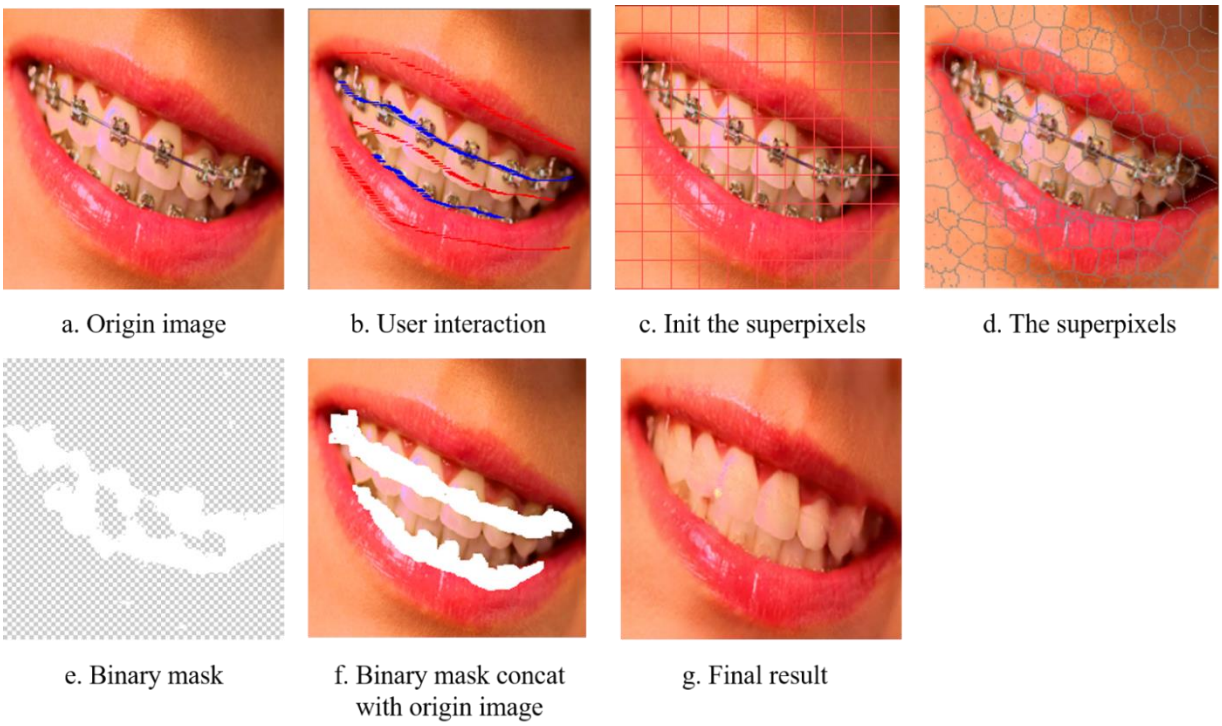
2.3. Xóa vật thể không mong muốn và khôi phục lại ảnh

[25] là nghiên cứu về một bài toán có nhiều nét tương đồng với braces2teeth. Mô hình của họ có thể tự động xóa khẩu trang và tái tạo lại khu vực đã xóa. Hai tác vụ này được thực thi trên hai module riêng biệt và kết quả của module xóa khẩu trang (1) sẽ là đầu vào cho module khôi phục vùng bị xóa (2). Mặc dù đã đạt được kết quả tốt trong đa số trường hợp nhưng toàn bộ mô hình còn khá cồng kềnh và không thể xóa những khẩu trang không có trong dataset.

Từ những khuyết điểm này, tôi đề xuất thay đổi hướng tiếp cận của module (1) để có thể xử lý tất cả mắc cài có kích thước, kiểu dáng, màu sắc khác nhau. Ý tưởng được trình bày chi tiết như sau.



Hình 2.9. Sơ đồ tổng quan của mô hình Inpainting, bao gồm hai mô hình GraphCut và Contextual Attention nối tiếp nhau



Hình 2.10. Ảnh trong các giai đoạn xử lý của mô hình Inpainting

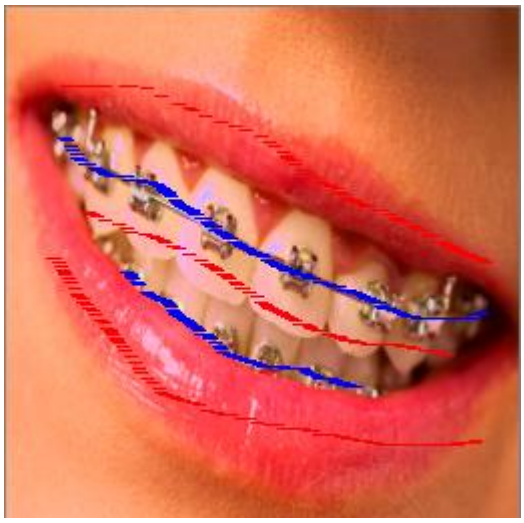
Mô hình GraphCut sẽ nhận vào ảnh gốc (hình 2.10.a) và user interaction (hình 2.10.b) đánh dấu vị trí mắc cài (màu đỏ đánh dấu không phải mắc cài và màu xanh đánh dấu mắc cài). Mô hình GraphCut ánh xạ ảnh sang tập superpixel (hình 2.10.c và 2.10.d) và sử dụng thêm thông tin từ user interaction để ánh xạ tiếp sang đồ thị ba chiều. Cuối cùng đưa đồ thị ba chiều vào thuật toán của Boykov-Kolmogorov Min-cut/Max-follow để thu được 2 đồ thị con S là tập các superpixel được đánh dấu là braces và T là tập các superpixel được đánh dấu là teeth. Tôi ánh xạ ngược lại S và T về ảnh gốc và có được một binary mask với giá trị 1 đánh dấu mắc cài và giá trị 0 đánh dấu không phải mắc cài (hinh 2.10.e).

2.3.1. Xóa vật thể với mô hình GraphCut

Hiện nay có nhiều mô hình deep learning như U – net, Fast – RCNN đã xử lý tốt tác vụ xóa vật thể tuy nhiên yêu cầu thời gian và quá trình training. Để đơn giản chúng, tôi sử

dụng GraphCut, một phương pháp thuần xử lý ảnh có độ chính xác kém hơn nhưng thời gian phản hồi thấp và không cần giai đoạn training.

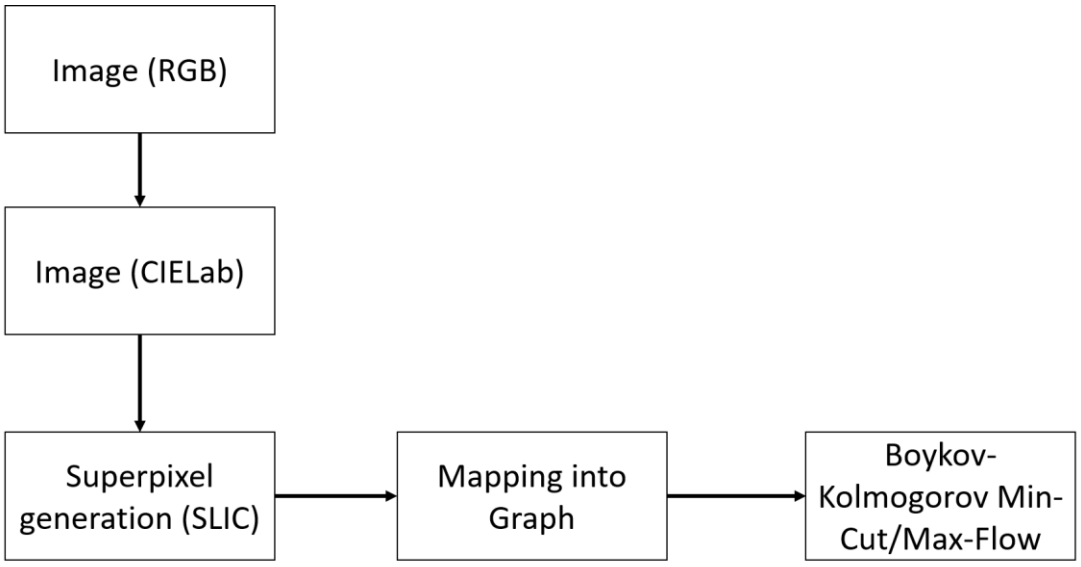
Đầu vào: ảnh màu M (không gian màu RGB), tập sink node và tập source node.



Hình 2.11. Đầu vào của mô hình GraphCut, chấm xanh dương đánh dấu vị trí sink node và chấm đỏ đánh dấu vị trí source node

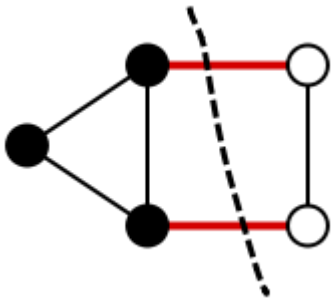
Đầu ra: những pixel thuộc node s được giữ nguyên, còn node t thì chuyển về giá trị RGB (255,255,255).

Hình 2.12. Đầu ra của mô hình GraphCut



Hình 2.13. Quy trình xử lý của mô hình GraphCut

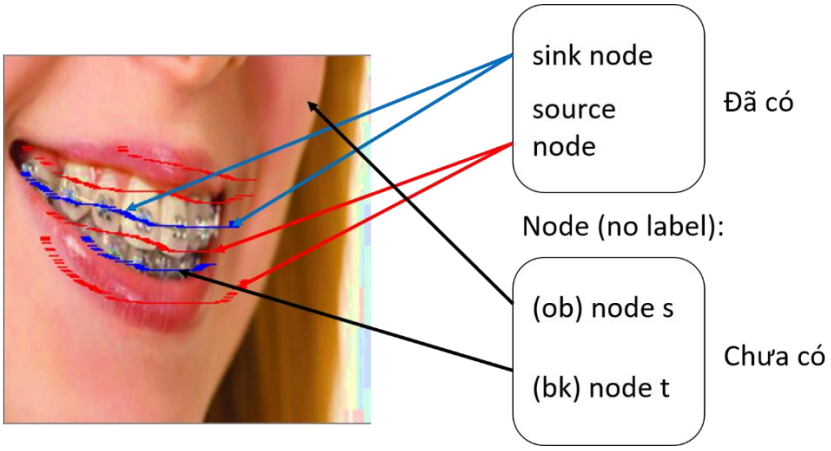
GraphCut [1] là một phương pháp áp dụng lý thuyết đồ thị. Cho đồ thị $G(E,V)$, cut là đường cắt chia G thành 2 đồ thị con với min-cut là đường cut với tổng số lượng edge cắt qua là nhỏ nhất và ngược lại với max-cut. Ví dụ: với đồ thị dưới min-cut là 2, và không có cut nào có giá trị 1 vì đồ thị không có cầu.



Hình 2.14. Graph G với 5 đỉnh, nét đứt chính là cut. Nguồn: [1]

Trong mạng dòng chảy (flow network), cho $G(E,V)$ hữu hạn và mỗi edge (u,v) có trọng số c là giá trị thực không âm. Giả sử có hai node, sink node (đỉnh thu) và source node (đỉnh phát) đã được xác định. s - t cut là đường cut chia G thành hai tập S và T sao cho $\forall s \in S, s$ là node chứa pixel là foreground và $\forall t \in T, t$ là node chứa pixel là background.

Tôi gọi s - t cut là max-flow khi tổng trọng số của các edge đường cut đi qua là cực đại và ngược lại với min-flow.



Hình 2.15. Hình 2.15. 4 loại node trong đồ thị, sink node là node chứa background và source node chứa foreground

Ứng dụng trong việc xóa vật thể (phân đoạn ảnh), đường phân đoạn foreground và background là đường cut thỏa mãn điều kiện min-cut lẫn max-flow. Tôi sử dụng thuật toán Boykov - Kolmogorov để tìm ra đường cut này. Mỗi pixel là node và khoảng cách (độ lệch màu) giữa 2 pixel là trọng số $c(u, v)$ của edge nối liền chúng, ta có thể ánh xạ từ ảnh bất kì sang đồ thị $G(E, V)$ tương ứng.

2.3.1.1. Giai đoạn chuyển đổi sang không gian CIELab

Để tính toán khoảng cách màu giữa 2 pixel, công thức CIDE2000 phát triển bởi CIE (International Commission on Illumination) được khuyến khích sử dụng. Để sử dụng CIDE2000, ta chuyển ảnh RGB về không gian màu CIELab.

Việc chuyển đổi bao gồm 2 bước:

Bước 1: Chuyển từ RGB sang không gian chuẩn XYZ

```
//sR, sG and sB (Standard RGB) input range = 0 ÷ 255
//X, Y and Z output refer to a D65/2° standard illuminant.

var_R = ( sR / 255 )
var_G = ( sG / 255 )
var_B = ( sB / 255 )

if ( var_R > 0.04045 ) var_R = ( ( var_R + 0.055 ) / 1.055 ) ^ 2.4
else                   var_R = var_R / 12.92
if ( var_G > 0.04045 ) var_G = ( ( var_G + 0.055 ) / 1.055 ) ^ 2.4
else                   var_G = var_G / 12.92
if ( var_B > 0.04045 ) var_B = ( ( var_B + 0.055 ) / 1.055 ) ^ 2.4
else                   var_B = var_B / 12.92

var_R = var_R * 100
var_G = var_G * 100
var_B = var_B * 100

X = var_R * 0.4124 + var_G * 0.3576 + var_B * 0.1805
Y = var_R * 0.2126 + var_G * 0.7152 + var_B * 0.0722
Z = var_R * 0.0193 + var_G * 0.1192 + var_B * 0.9505
```

Hình 2.16. Mã giả quá trình chuyển đổi từ không gian màu RGB→XYZ. Nguồn: [19]

Bước 2: Chuyển từ không gian chuẩn XYZ sang không gian CIELab

```
//Reference-X, Y and Z refer to specific illuminants and observers.
//Common reference values are available below in this same page.

var_X = X / Reference-X
var_Y = Y / Reference-Y
var_Z = Z / Reference-Z

if ( var_X > 0.008856 ) var_X = var_X ^ ( 1/3 )
else                    var_X = ( 7.787 * var_X ) + ( 16 / 116 )
if ( var_Y > 0.008856 ) var_Y = var_Y ^ ( 1/3 )
else                    var_Y = ( 7.787 * var_Y ) + ( 16 / 116 )
if ( var_Z > 0.008856 ) var_Z = var_Z ^ ( 1/3 )
else                    var_Z = ( 7.787 * var_Z ) + ( 16 / 116 )

CIE-L* = ( 116 * var_Y ) - 16
CIE-a* = 500 * ( var_X - var_Y )
CIE-b* = 200 * ( var_Y - var_Z )
```

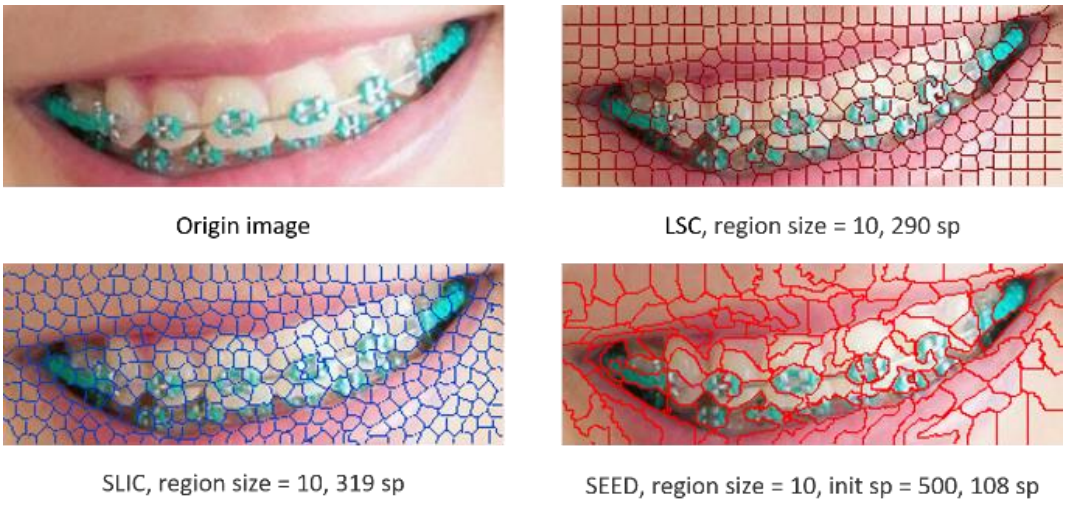
Hình 2.17. Mã giả quá trình chuyển đổi từ không gian màu XYZ→CIELab. Nguồn: [19]

2.3.1.2. Giai đoạn chuyển đổi SLIC

Tuy nhiên, khi thực hiện theo cách truyền thống với ví dụ ảnh 256 x 256, giả sử chỉ có 1 sink node và 1 source node, chúng ta có số lượng node là 256 x 256 = 65536, số lượng edge (256/4*4 + 256/8*2)*255 + 256*2 = 82112. Vì kích thước của đồ thị sẽ tăng theo cấp số nhân kích ảnh nên để giảm bớt khối lượng tính toán, tôi sử dụng phương pháp Superpixel generation, là phương pháp ánh xạ từ ảnh có kích thước bất kì sang ma trận những pixel lớn (pixel chứa những pixel gần giống nhau).

Khái niệm Superpixel được sử dụng để ám chỉ ý tưởng về những pixel gần nhau thì có giá trị màu gần như tương tự nhau, do đó chúng ta có thể nhóm những pixel chung đặc điểm thành một pixel duy nhất có giá trị màu có thể bằng trung bình các pixel cấu thành, pixel mới được tạo được gọi là superpixel (viết tắt là SP). Mỗi SP sẽ có thông tin bao gồm pixel trung tâm (center) và biên với superpixel khác (boundary).

Hiện tại có 3 thuật toán Superpixel generator phổ biến: SEEDS [19], SLIC [20] và LCS [18]. Kết quả và ý tưởng của ba phương pháp trên ảnh răng có mắc cài là tương đối giống nhau nên tôi sẽ trình bày SLIC là mô hình tượng trưng.

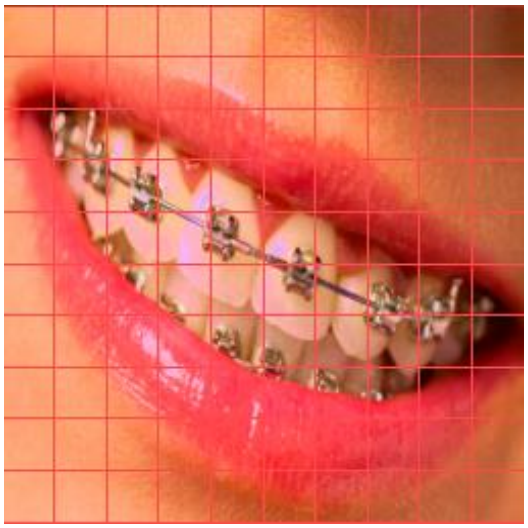


Hình 2.18. Superpixel generation bằng những phương pháp khác nhau, LSC và SLIC yêu cầu tính toán lại region size để đạt kết quả tốt nhất. Trong khi đối với SEED, kích thước mắc cài sẽ thay đổi theo kích thước ảnh nên không cần thay đổi tham số là số lượng superpi

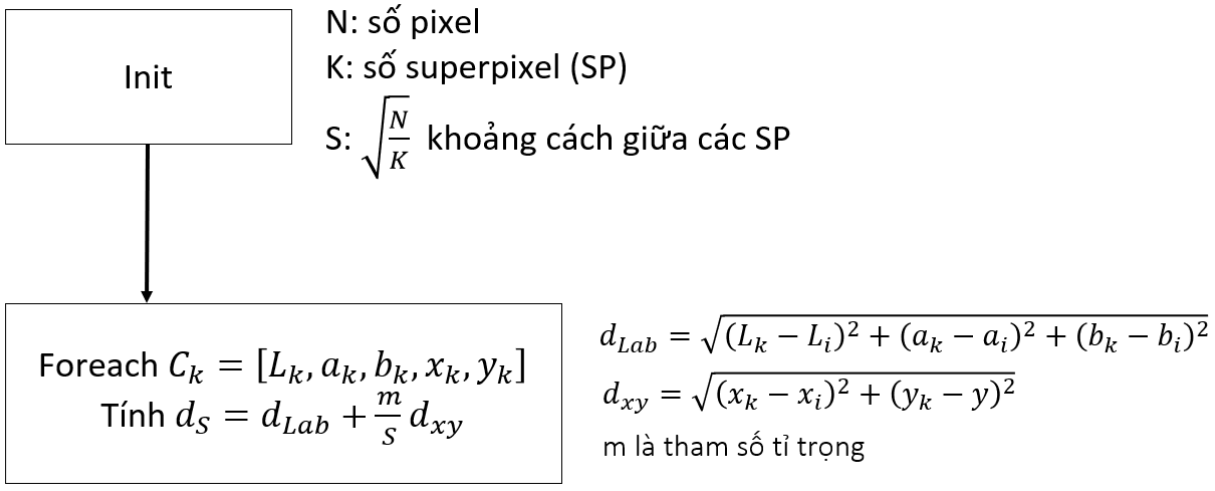
2.3.1.2.1. SLIC

SLIC (viết tắt simple linear Iterative cluster) là một phương pháp ánh xạ ảnh bất kì sang $\{SP\}, \forall sp \in SP, sp = \{center, boundary\}$. Phương pháp này chia thành 2 bước.

Bước 1: Khởi tạo với mỗi center $C_k = [L_k, a_k, b_k, x_k, y_k]$ trong đó L_k, a_k, b_k là giá trị màu trong không gian CIELab, x_k, y_k là tọa độ trong ảnh.

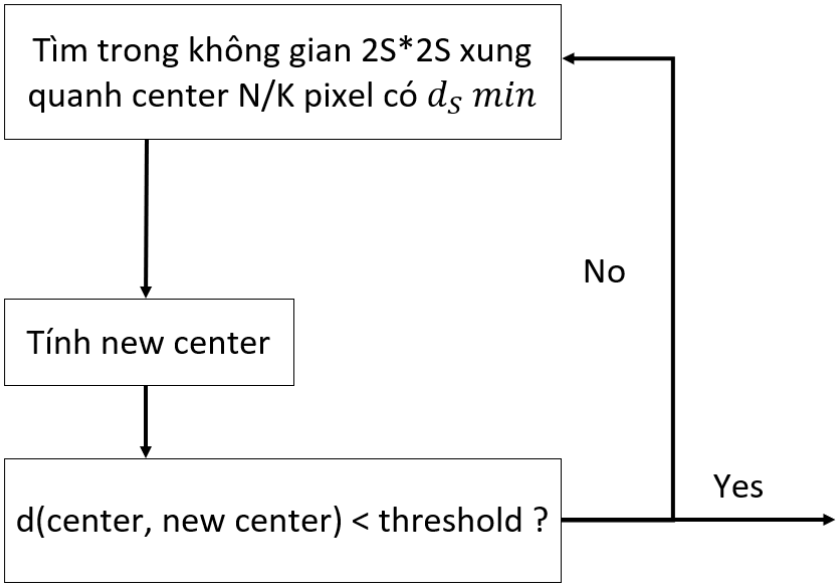


Hình 2.19. Các SP khi khởi tạo có center cách đều nhau nên các boundary sẽ song song và cách đều nhau



Hình 2.20. Quy trình khởi tạo với mỗi SP. Nguồn: [19]

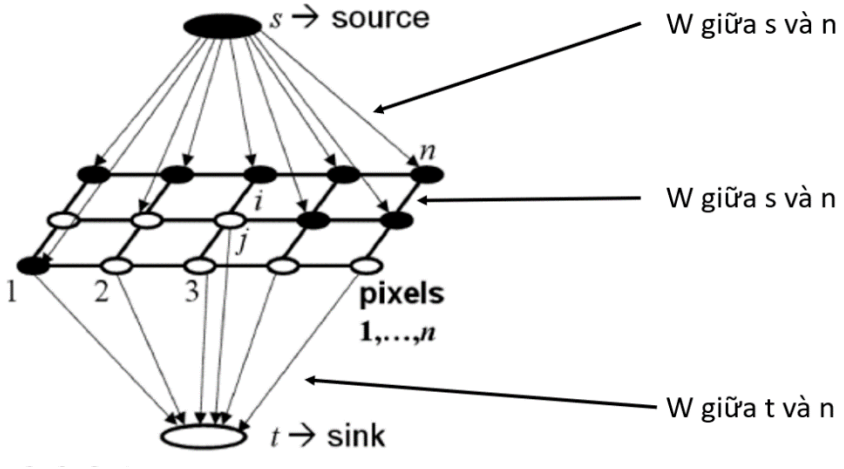
Bước 2: Tìm center mới cho mỗi SP.



Hình 2.21. Quy trình xử lý với mỗi SP. Nguồn: [19]

Hình 2.22. Các boundary mới sau khi tính lại center cho tất cả SP

2.3.1.3. Giai đoạn Mapping



Hình 2.23. Đồ thị mẫu sau khi Mapping. Nguồn: [19]

Sau khi có tập SP, ta ánh xạ chúng sang đồ thị để áp dụng thuật toán Boykov - Kolmogorov. Có hai bước tính toán chính:

Bước 1: Khởi tạo trọng số cho các node bằng với giá trị màu mỗi superpixel với việc gán một SP tương ứng với một node.

Bước 2: Khởi tạo trọng số cho các edge $w(u, v)$ bao gồm 3 loại giữa 2 node không label, không label - có label và 2 node có label.

Với 2 node u $(center,\ hist)$ và v $(center,\ hist)$ (với hist là histogram):

$$similarity(u, v) = e^{\left(-\frac{cv2.compareHist(u.hist,\ v.hist)^2}{2*\sigma^2}\right)} * \frac{1}{distance(u.center,\ v.center)} \quad (2.10)$$

với:

$$cv2.compareHist(H1, H2) \vdots\equiv d(H1, H2) = \sum_I \min\left(H_1(I), H_2(I)\right) \quad (2.11)$$

Ta có được:

$$w(u,v) = 1 + sim(u,v) \quad (2.12)$$

Với node $s(center,\ hist), t(center,\ hist)$ và $u(center,\ hist)$ (không label):

$$pr_{ob} \equiv \frac{hist_{ob}(u)}{hist_{ob}sum}, pr_{bk} \equiv \frac{hist_{bk}(u)}{hist_{bk}sum} \quad (2.13)$$

$$\text{Nếu } pr_{bk} > 0 \ \rightarrow w(t,u) = 100000,\ w(s,u) = 0.9 * -\log(pr_{ob}) \quad (2.14)$$

$$\text{Nếu } pr_{ob} > 0 \ \rightarrow w(s,u) = 100000,\ w(t,u) = 0.9 * -\log(pr_{ob}) \quad (2.15)$$

Với node $s(center, hist), t(center, hist)$ $(center, hist)$ và $u(center, hist)$ (có label):

$$u\ (label\ Source\ node),\ w(s,u) = 1 + sim,\ w(t,u) = 0 \quad (2.16)$$

$$u\ (label\ Sink\ node),\ w(t,u) = 1 + sim,\ w(s,u) = 0 \quad (2.17)$$

2.3.1.4. Giai đoạn chuyển đổi Boykov - Kolmogorov

Sau khi có đồ thị $G(E,V)$, tôi áp dụng hàm Boykov - Kolmogorov: $G \rightarrow [S,T]$ để thu được đồ thị S và T.

2.3.2. Khôi phục ảnh bằng Generative Image Inpainting with Contextual Attention

Đầu vào: ảnh RGB và binary mask, với binary mask được ánh xạ từ đầu ra của mô hình GraphCut.

Hình 2.24. Ảnh gốc (Ground truth, viết tắt là GT)

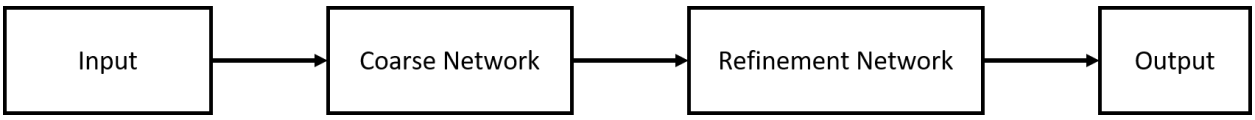
Hình 2.25. Mặt nạ nhị phân (Binary mask), pixel màu trắng thể hiện vùng cần xóa

Hình 2.26. GT và binary mask sau khi thực hiện phép concat

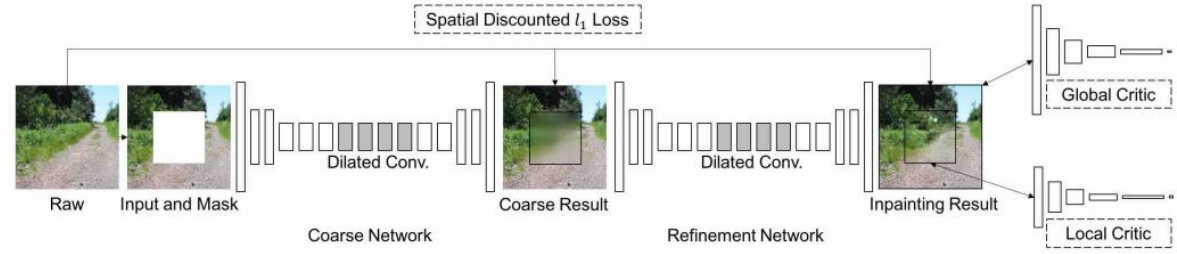
Hình 2.27. Ảnh sau khi được phục hồi

Đầu ra: ảnh được phục hồi.

Pipeline:



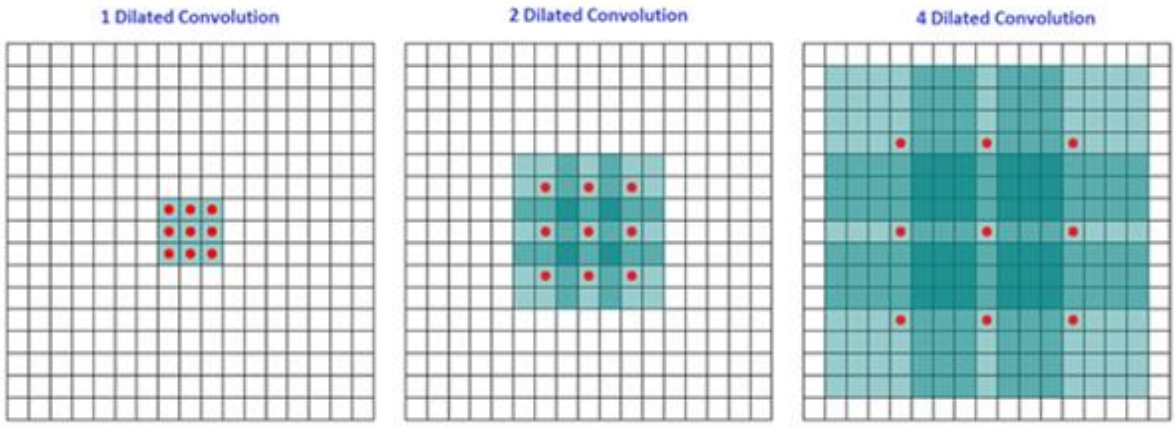
Hình 2.28. Quy trình xử lý của mô hình Inpainting



Hình 2.29. Kiến trúc của toàn bộ mô hình Contextual Attention. Nguồn: [4]

Kiến trúc mạng chia làm 2 giai đoạn:

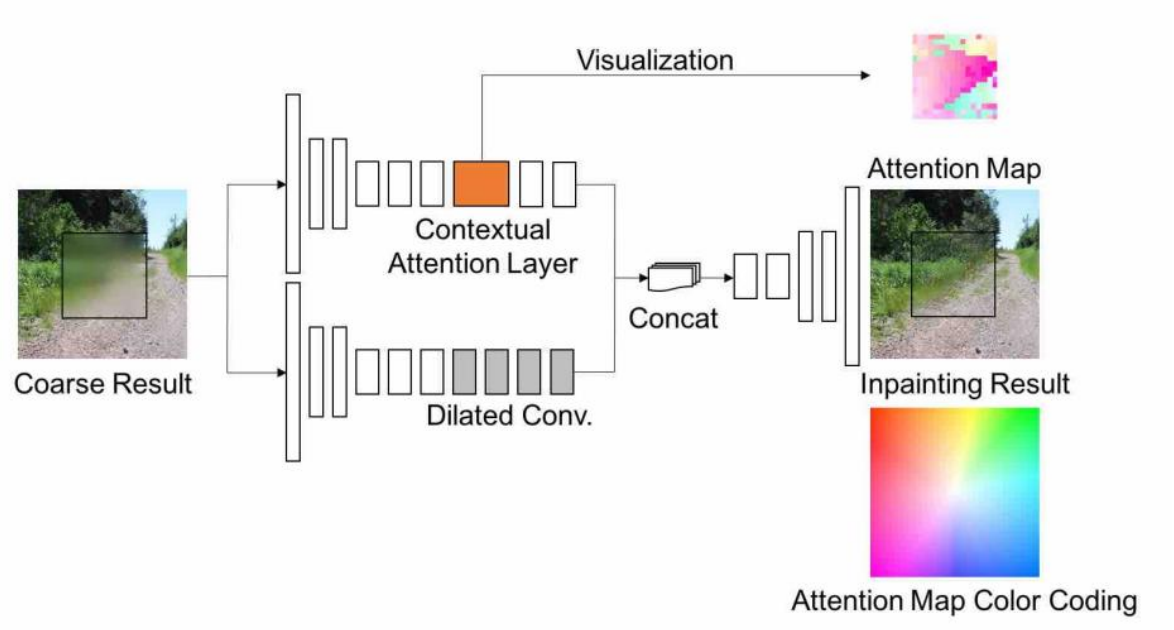
Giai đoạn 1 (Coarse network): giai đoạn làm thô, từ missing region ảnh được khôi phục sao cho nội dung bên trong chỉ cần tương thích với background. Giai đoạn này mô hình sử dụng thêm Spatial discounted reconstruction loss để đánh giá chất lượng theo khoảng cách từ boundary của missing region đến các missing pixel bên trong, khoảng cách càng nhỏ thì pixel phục hồi càng phải khớp với background (γ^i với i là khoảng cách đến pixel bên ngoài boundary gần nhất và $\gamma = 0.99$). Các lớp convolutional sử dụng filter là dilated layer.



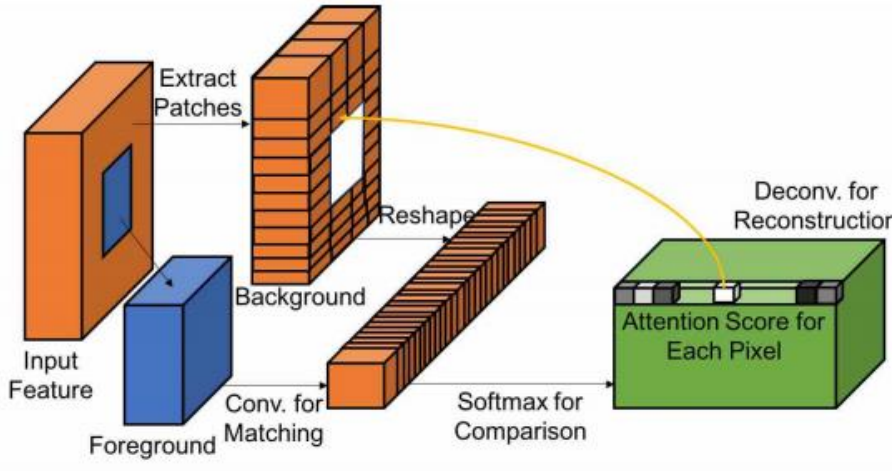
Hình 2.30. Ma trận dilated với padding và độ giãn nở khác nhau. Nguồn: [4]

Dilated layer sử dụng kernel có các phần tử cách đều nhau và padding bởi phần tử 0. Khoảng cách cách đều và độ dày của padding tùy thuộc vào kích thước ảnh muốn giãn nở. Khi đi qua nhiều dilated layer, kích thước ảnh được tăng về ban đầu.

Giai đoạn 2 (Refinement network): giai đoạn làm mịn.



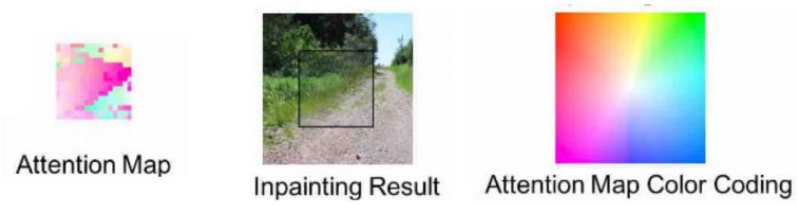
Hình 2.31. Kiến trúc của mạng Coarse. Nguồn: [4]



Hình 2.32. Kiến trúc của Contextual Attention Layer. Nguồn: [4]

Refinement network sử dụng kết quả từ hai mạng độc lập Contextual Attention và Dilated Convolutional sau đó concat với nhau. Mạng Contextual Attention tìm những pixel phù hợp trong background và khôi phục vào missing region với 3 bước như sau:

– Bước 1: Rút trích các mảnh (patch) $3x3$ từ background và reshape để tạo thành filter.
– Bước 2: Missing region (foreground) được đưa qua mạng convolutional với filter từ bước 1 và layer cuối cùng sử dụng hàm softmax.
– Bước 3: Tôi thu được ma trận Attention Score có nghĩa sau. Dựa vào Attention Score, tôi xác định được chính xác giá trị màu cần phục hồi dựa vào những pixel khác trong ảnh.



Hình 2.33. Attention thể hiện độ tương đồng pixel cần khôi phục khi so với pixel khác trong background. Nguồn: [4]

Mạng dilated mở rộng receptive field với mục đích tổng quát hóa và có kiến trúc tương tự với coarse network. Lý do cần sử dụng hai mạng với cùng mục đích là để đạt được cả hai yếu tố: chất lượng (resolution) - đặc trưng chi tiết và độ hợp lý (consistency) - đặc trưng tổng thể. Khi sử dụng chỉ một trong hai, tôi gặp vấn đề sau:

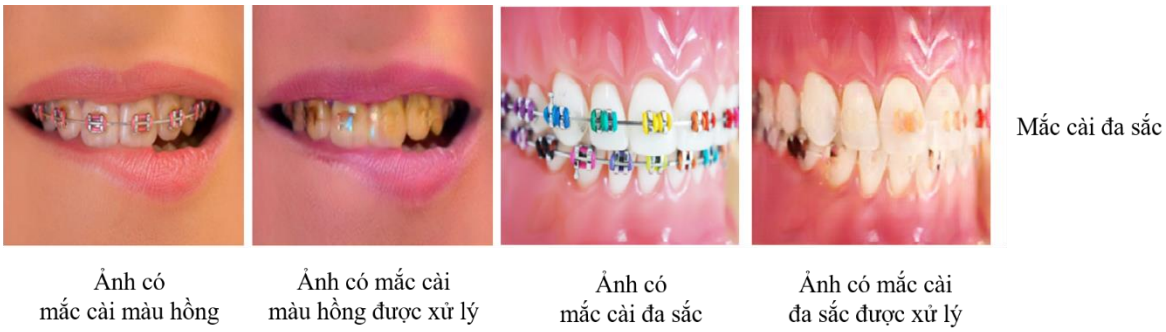
Hình 2.34. Các vị trí tại biên (đánh dấu đỏ) có hiện tượng inconsistency khi không kết hợp với dilated network. Nguồn: [4]

CHƯƠNG 3. BÀI TOÁN BRACES2TEETH

Như đã trình bày, CycleGAN có thể đạt được kết quả tốt trong nhiều trường hợp. Nhưng đối với một số loại mắc cài đặc biệt như mắc cài đa sắc, mắc cài sứ, mắc cài vô hình thì mô hình gặp tình trạng xóa không hết hoặc không thể xóa được. Nội dung của chương 3 bài gồm hai phần, phân tích các trường hợp khó xử lý xảy ra khi sử dụng mô hình CycleGAN và các kỹ thuật hạn chế lỗi (3.1), một số kỹ thuật tạo dataset cho mô hình Pix2Pix (3.2).

3.1. Xử lý ngoại lệ

3.1.1. Tăng cường ảnh



Hình 3.1. Trường hợp mắc cài đa sắc

Hình 3.2. Trường hợp mắc cài sứ

Vì mắc cài có thể có nhiều màu sắc khác nhau nên mô hình vẫn không thể xóa được những trường hợp đặc biệt như màu trùng với những thành phần của răng như hồng (lợi), răng (trắng – mắc cài sứ). Tôi giải quyết vấn đề này bằng một số phương pháp tăng cường ảnh nhằm tăng cường số lượng màu sắc thể hiện trong mắc cài.

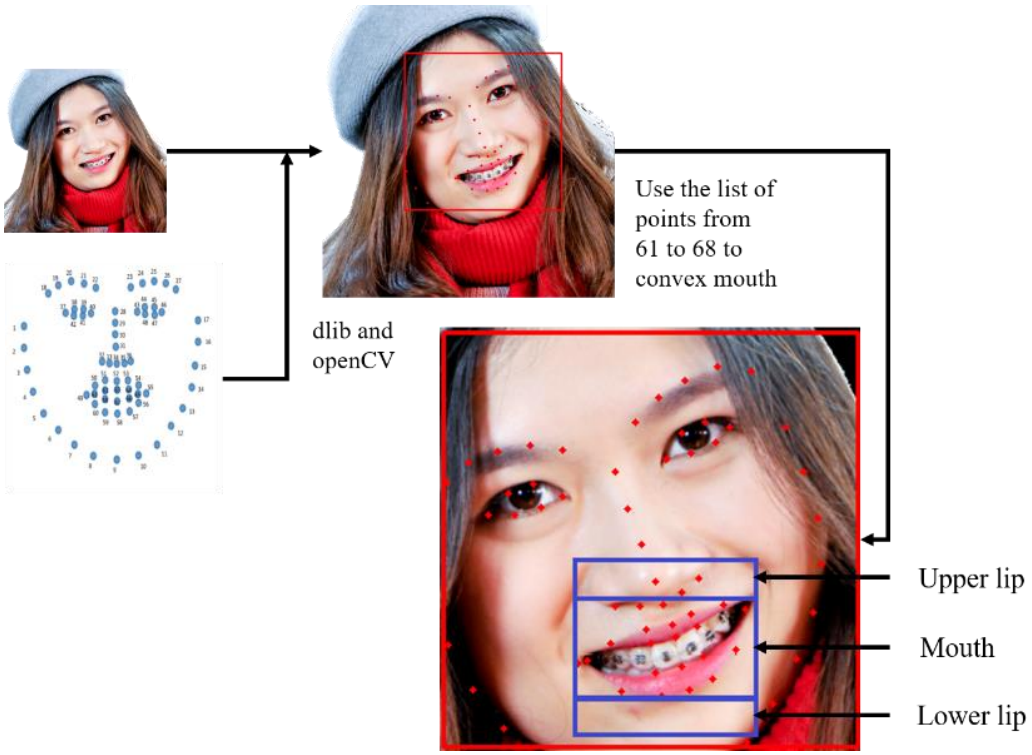
Đối với mỗi ảnh trong dataset, tôi tạo ngẫu nhiên các phiên bản tăng cường bằng cách thay đổi các kênh màu S và V trong không gian HSV để răng và niềng răng có tông màu khác nhau. Sau khi thử nghiệm, những mắc cài có màu sắc sặc sỡ đã được xóa, tuy nhiên mô hình cũng thay đổi sắc thái của ảnh khá nhiều.

Hình 3.3. Ảnh gốc (góc trên bên trái) và những phiên bản khác sau khi áp dụng tăng cường ảnh

Hình 3.4. Ảnh gốc và ảnh do mô hình CycleGAN sau khi áp dụng tăng cường ảnh.

3.1.2. Trích xuất vùng miệng

Hình 3.5. Trường hợp mắc cài có độ phân giải thấp



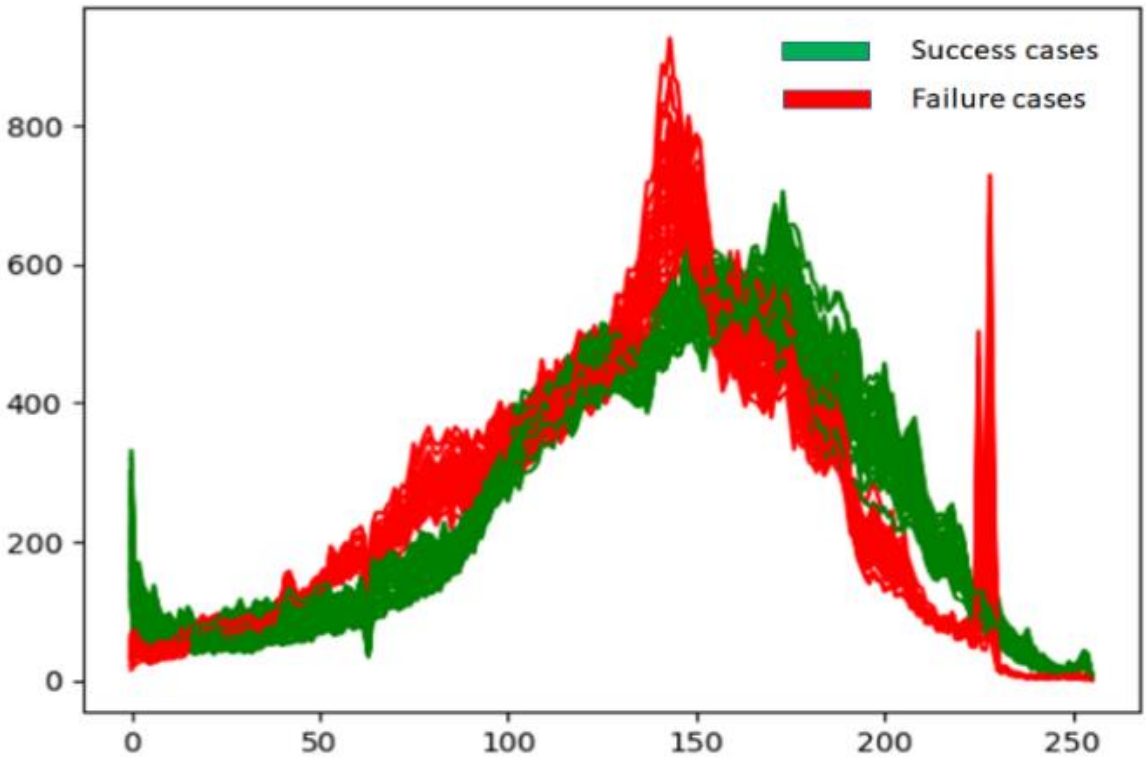
Hình 3.6. Phương pháp cắt ra vùng miệng, bằng dlib và OpenCV để tạo facial landmark detection. Phần mouth được đánh dấu bởi các point trên landmark có index từ 61 đến 68

Những ảnh có chứa mắc cài mà trong đó mắc cài quá nhỏ và cấu trúc của nó không rõ ràng, tôi cắt vùng miệng, đưa nó vào mô hình và lấy kết quả thay thế vùng được trích xuất trong ảnh gốc. Để giữ tỷ lệ chiều rộng và chiều cao của vùng miệng được chiết xuất,

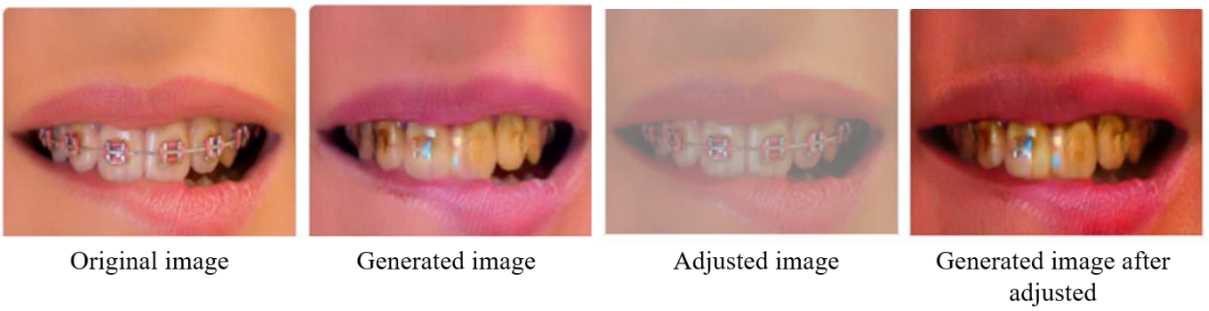
tôi xác định lại vùng trích xuất bằng cách mở rộng về phía môi trên và môi dưới để ảnh không bị méo khi đưa vào mô hình.

3.1.3. Cân bằng Histogram

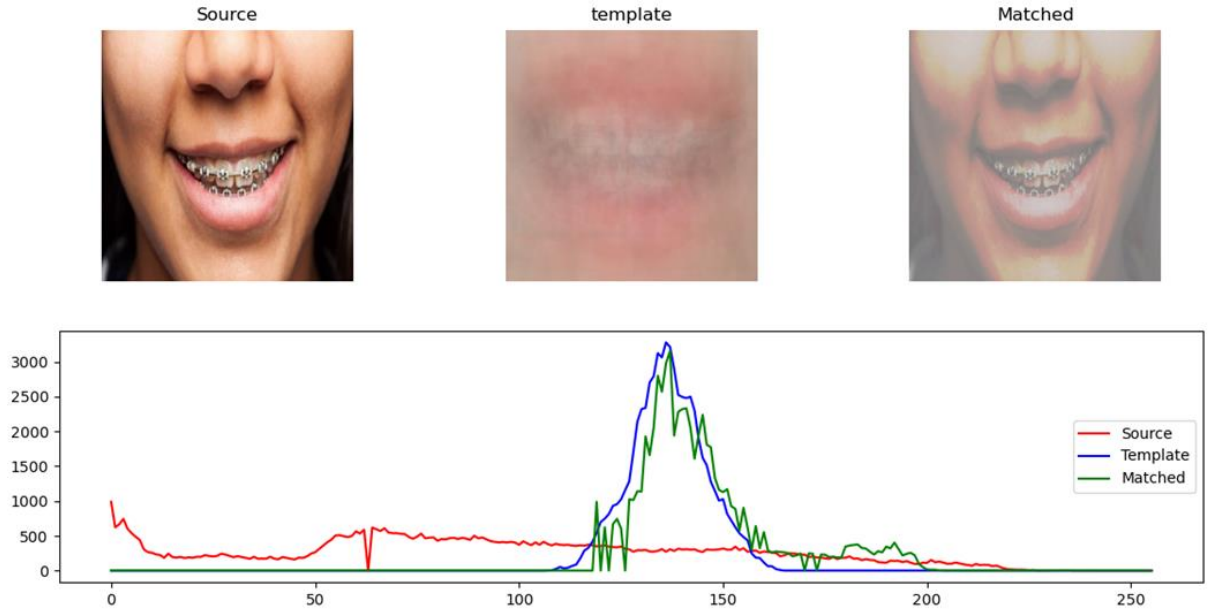
Tôi tiến hành phân tích histogram của trung bình RGB trong tập 200 ảnh test. Dễ thấy những trường hợp failure có 2 đỉnh ở khoảng 130 – 140 và 220 – 230 vì elastic tie thường có màu sắc sặc sỡ (sáng). Tôi thử nghiệm việc cân bằng histogram của những trường hợp failure với histogram trung bình của những trường hợp success bằng phương pháp nội suy tuyến tính tuy nhiên phương pháp này tỏ ra không hiệu quả.



Hình 3.7. Histogram trung bình của những điểm dữ liệu thuộc tập test mà mô hình CycleGAN đã xóa thành công (màu xanh) và những điểm dữ liệu thuộc tập test mà mô hình CycleGAN đã không xử lý được (màu đỏ).

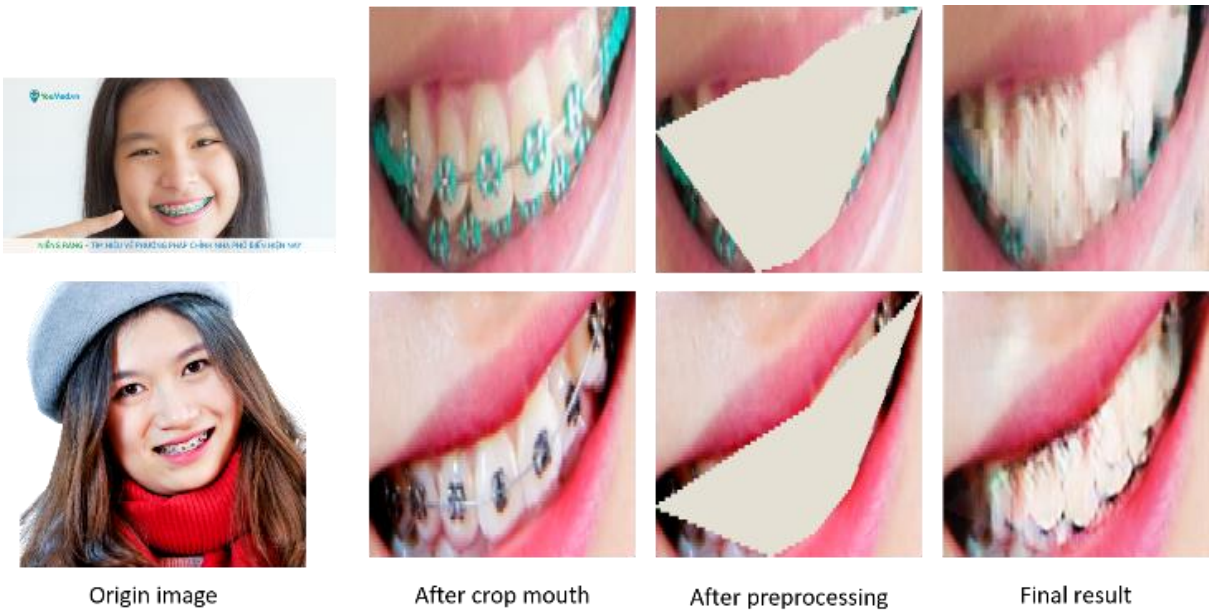


Hình 3.8. Kết quả thử nghiệm vẫn không tốt sau khi áp dụng phương pháp cân bằng histogram.



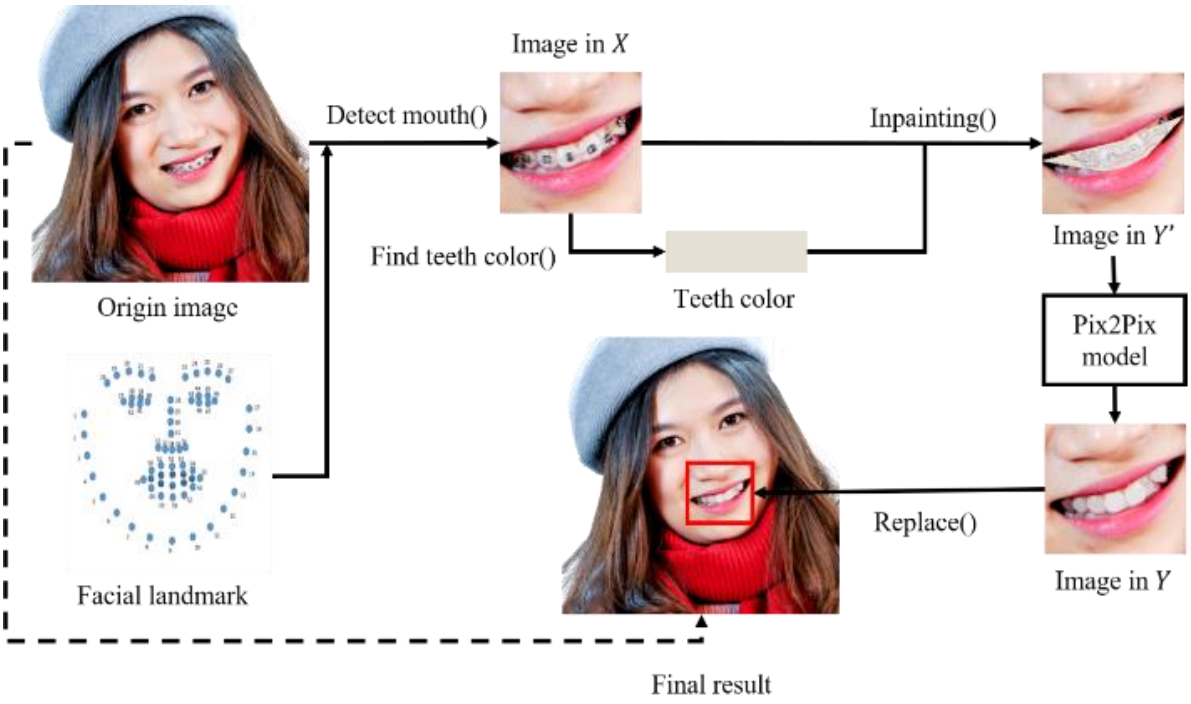
Hình 3.9. Quá trình cân bằng histogram bằng phương pháp nội suy tuyến tính, trong đó source là điểm dữ liệu cần điều chỉnh (histogram đỏ), template là histogram mẫu hướng đến (histogram màu xanh nước biển) và matched (histogram màu xanh lá) là histogram đã được

3.2. Mô hình Pix2Pix



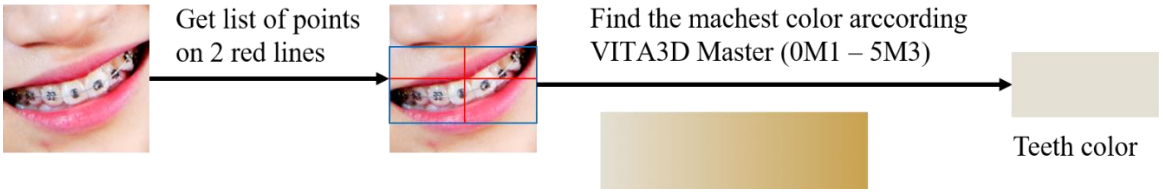
Hình 3.10. Kết quả của mô hình Pix2Pix với phương pháp xóa mắc cài A, lần lượt từ trái sang phải, ảnh gốc, phần mouth đã được cắt, selection zone được inpainting bằng màu răng và ảnh do mô hình Pix2Pix trả về

Pix2Pix [15] như đã giới thiệu là phương pháp sinh ảnh có chất lượng tốt hơn CycleGAN nhưng lại yêu cầu dataset là các cặp. Nên nội dung chủ yếu của phần này là một framework để tạo ra dataset theo cặp. Đầu tiên, chúng ta chỉ sử dụng ảnh răng không có mắc cài $y_{j=1}^{m}$, với mỗi y_j chúng ta trích xuất vùng miệng (bằng phương pháp đã được trình bày trong phần 3.1.2) và lấp khu vực trong khoang miệng bằng màu răng. Sau khi lập đầy, y_i biến đổi y'_i là phiên bản trung gian giữa x_i và y_i. y'_i là y_i được biến đổi nhưng chúng ta sẽ tạm xem chúng là ảnh được xử lý từ x_i. Hiện tại chúng ta có các cặp $\{y'_i, y_i\}_{i=1}^{N}$ có thể sử dụng chúng làm dữ liệu huấn luyện cho mô hình Pix2Pix.



Hình 3.11. Tổng quan về toàn bộ quá trình xử lý, sau khi có được kết quả từ giai đoạn crop mouth như được mô tả trong phần 3.4. Tôi tìm màu răng và inpainting trên selection zone (là vùng nằm trong môi), cuối cùng là đưa vào mô hình Pix2Pix

3.2.1. Tìm màu răng



Hình 3.12. Quy trình tìm màu răng

Tôi sử dụng bảng màu dựa trên VITA3D Master để tham chiếu trực tiếp đến những điểm ảnh lấy được có khả năng là răng. Sau đó tìm màu răng bằng lấy tập pixel thông qua 2 đường trung trực của ảnh và so sánh pixel có màu gần nhất với màu nào đó trên bảng mẫu.

Hình 3.13. Dải màu răng VITA3D Master từ 0M1 đến 5M3

3.2.2. Xóa mắc cài

Hình 3.14. Kết quả khi của mô hình Pix2Pix với phương pháp xóa mắc cài B, lần lượt từ trái sang phải, ảnh gốc, phần mouth đã được cắt, selection zone được lấp đầy bằng màu răng và ảnh được mô hình Pix2Pix trả về

Tôi sử dụng hàm convexPolygon trong OpenCV các point có index từ 61 – 68 trên facial landmark để xác định selection zone. Tôi thử nghiệm 3 phương pháp khác nhau để so sánh:

Phương pháp A (hình 3.10): Những pixel nằm trong selection zone được lấp đầy lại bằng màu răng.

Phương pháp B (hình 3.12): Những pixel nằm trong selection zone và có khoảng cách với màu răng lớn hơn ngưỡng cho trước sẽ được lấp đầy lại bằng màu răng. Tại đây tôi chuyển ảnh sang không gian màu CIELab và sử dụng khoảng cách CIE2000.

Phương pháp A khá đơn giản nhưng sẽ khiến ảnh sinh ra sai so với ground truth và không có thông tin về cấu trúc răng khi đưa vào mô hình. Phương pháp B có thể giữ lại nhiều thông tin quan trọng, nhưng kết quả có thể không đạt như mong muốn như vài trường hợp được mô tả ở hình 3.14.

CHƯƠNG 4. THỰC NGHIỆM

Để đánh giá tính khả thi của các mô hình được đề xuất, tôi đã tiến hành thu thập dữ liệu (4.1) và huấn luyện các mô hình đã trình bày (4.2).

4.1. Dataset

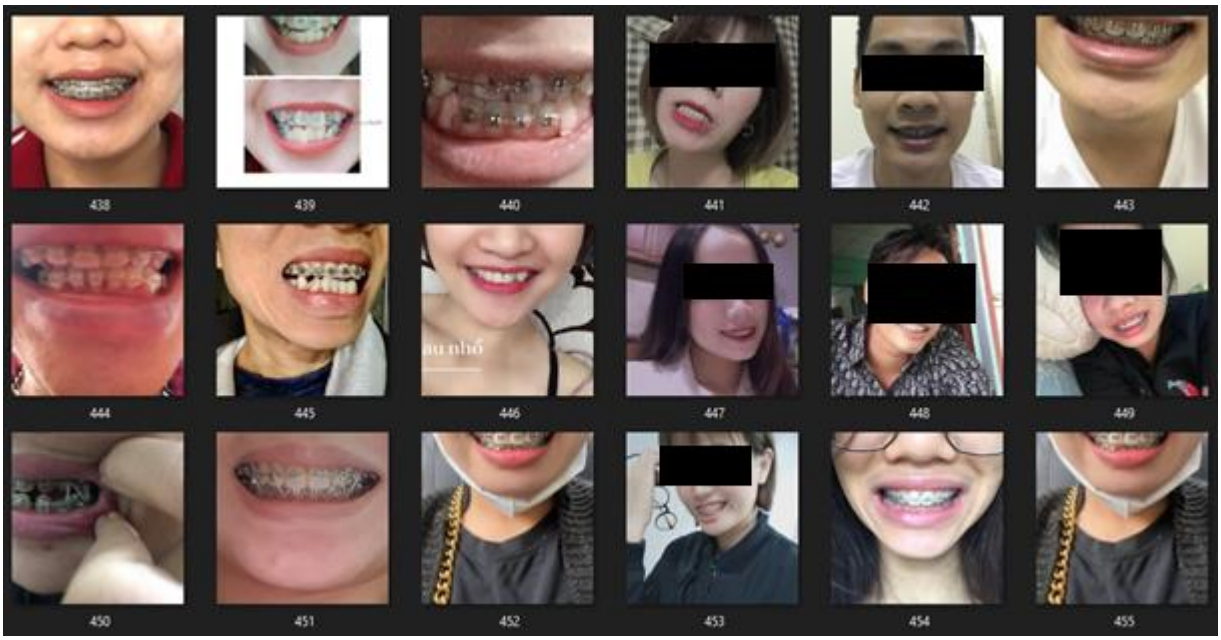
Dataset bao gồm hai domain là ảnh răng có mắc cài và ảnh răng không có mắc cài. Vì bài toán braces2teeth chưa tồn tại dataset sẵn có nên tôi đã thu thập dữ liệu từ các nguồn: google image, facebook và StyleGAN2, ngoài ra còn có số lượng nhỏ đến từ các phòng khám nha khoa.

Tôi lấy ảnh từ google images và facebook với công cụ Downloader được cung cấp bởi Yabin Zheng bằng cách sử dụng những từ khóa trong nhiều ngôn ngữ khác nhau liên quan tới mắc cài và răng. Kết quả từ nguồn này tạm chấp nhận được với 0 – 100 ảnh trên mỗi từ khóa. StyleGAN2 là một mô hình lớn đến từ kết quả nghiên cứu của NVIDIA, nó cung cấp ảnh khuôn mặt người công khai và tôi sử dụng nguồn này bằng công nghệ cào ảnh tự động Selenium web driver, sau đó sử dụng những module như đã giới thiệu ở phần 3.2 để lấy phần miệng. Ảnh từ StyleGAN2 không giới hạn số lượng, tuy nhiên răng ở tất cả các ảnh đều khá giống nhau (đều trắng, sáng và đẹp) nên tôi chỉ sử dụng chúng với tỉ lệ nhỏ trong dataset. Nguồn ảnh từ các phòng khám nha khoa rất có giá trị, nhưng đa số đều không công khai và tôi chỉ được cung cấp một số lượng nhỏ trong đó.

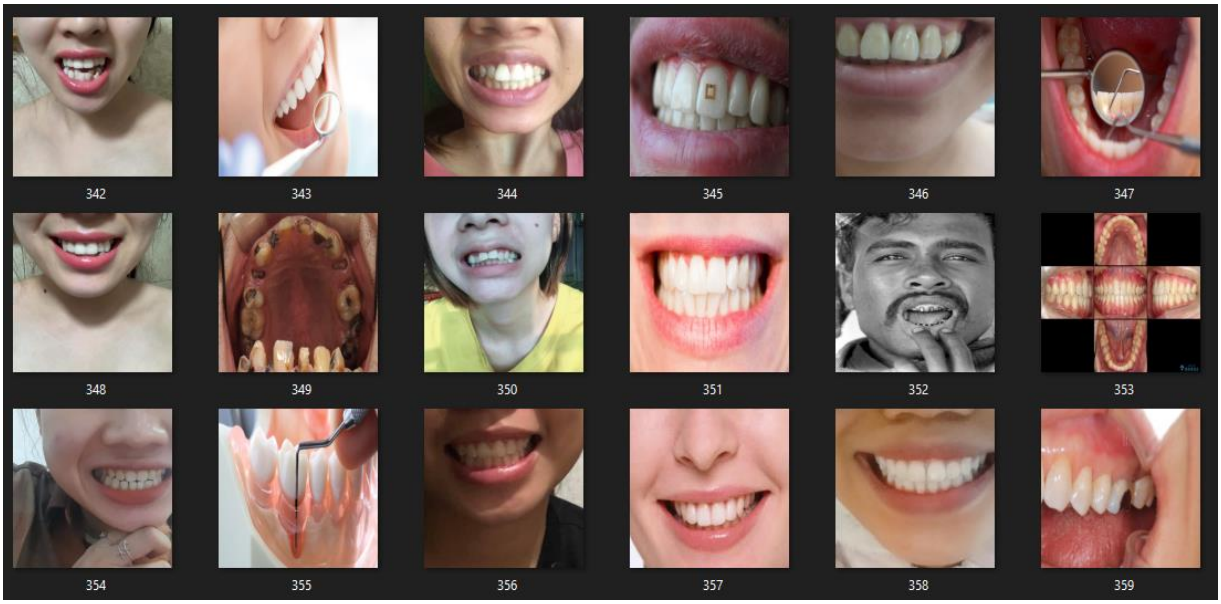
Tất cả ảnh trong dataset đều được xử lý trước khi đưa vào training. Tác vụ này bao gồm thay đổi kích thước về 256 x 256 x 3 sử dụng nội suy lưỡng cực (bicubic interpolation), giảm nhiễu bằng cách loại bỏ những ảnh không liên quan như ảnh hoạt hình, tranh vẽ, quảng cáo, …

Bảng 4.1. Chi tiết về ba bộ dataset

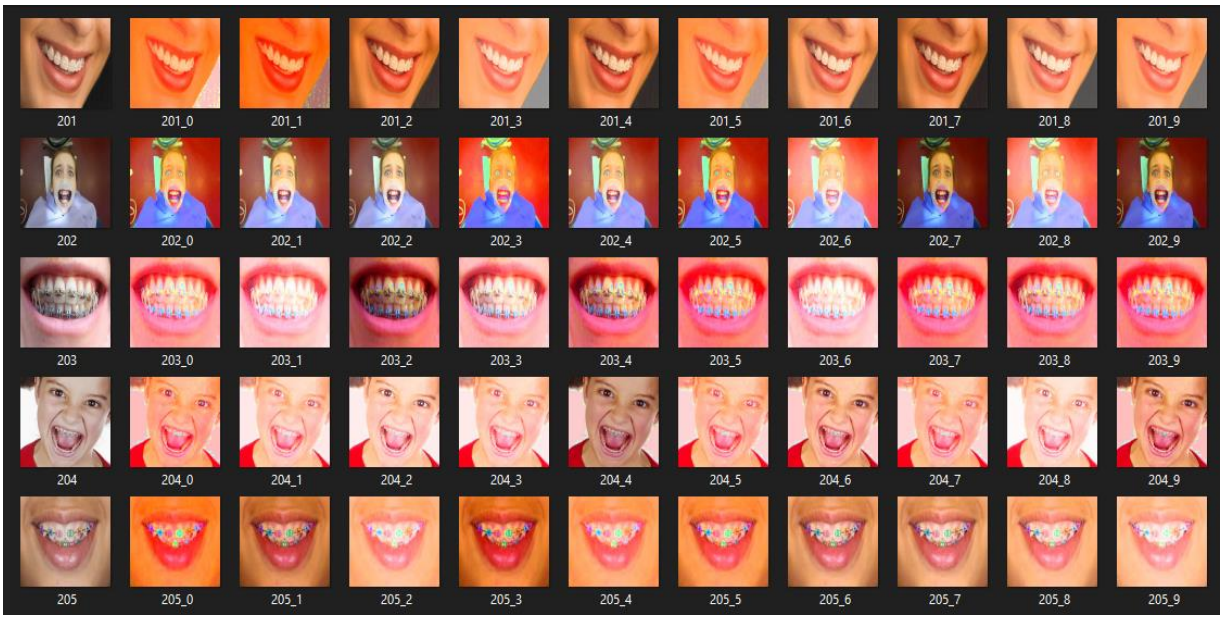
Tên	Số lượng ảnh	Chi tiết	Nguồn
braces2teeth	3869	testX: 437 trainX: 1728 testY: 340 trainY: 1364	Google image, Facebook, phòng khám nha khoa.
braces2teeth Augmented	38690	testX: 4370 trainX: 17280 testY: 3400 trainY: 13640	braces2teeth sau khi áp dụng phép tăng cường ảnh được mô tả trong phần 3.1.1.
teeth2	7000	train: 5000 val: 1000 test: 1000	braces2teeth/teeth, StyleGAN2.



Hình 4.1. Lớp braces trong dataset braces2teeth



Hình 4.2. . Lớp teeth trong dataset braces2teeth



Hình 4.3. Lớp braces trong dataset braces2teethAugmented

Hình 4.4. Lớp teeth trong dataset braces2teethAugmented

Hình 4.5. Dataset teeth2

4.2. Cài đặt thực nghiệm

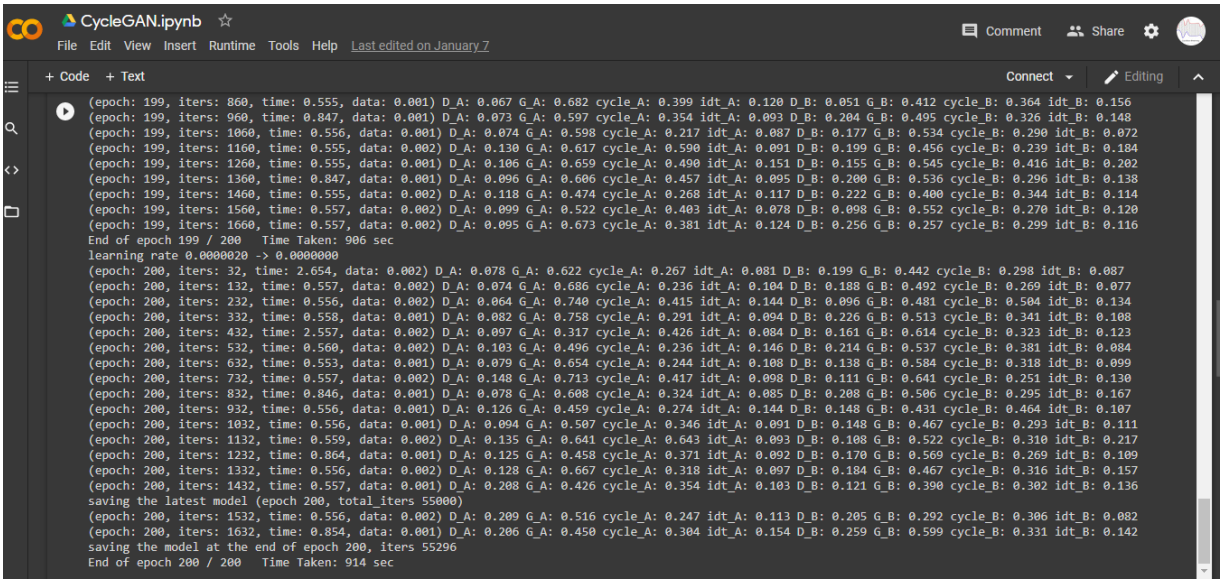
Theo nghiên cứu từ [14], tôi sử dụng lại mô hình ổn định và không thay đổi nhiều về kiến trúc và loss functions.

Tôi cũng sử dụng tối ưu Adam và thử nghiệm với batch size 1, 16 và 32 với không có khác biệt đáng kể (batch size 2 hoặc 4 không tốt)

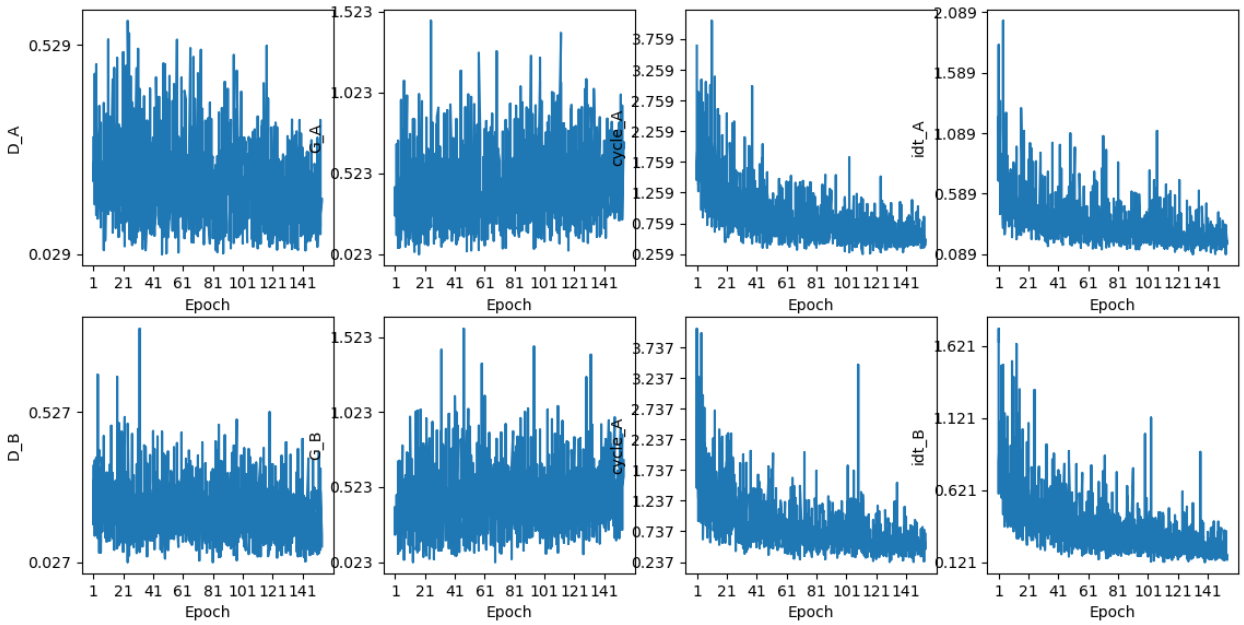
Tôi huấn luyện mô hình lần lượt theo các chiến lược: 50 epoch (learning rate 0.0002), 100 epoch (learning rate 0.0002), 200 epoch (learning rate 0.0002) và 200 epoch (learning rate được giữ trong 100 epoch đầu tiên và giảm dần tuyến tính tới xấp xỉ 0 trong 100 epoch kế tiếp). Chiến lược cuối cùng cho ra chất lượng hình ảnh tốt nhất.

Bảng 4.2. Chi tiết về quá trình thực nghiệm

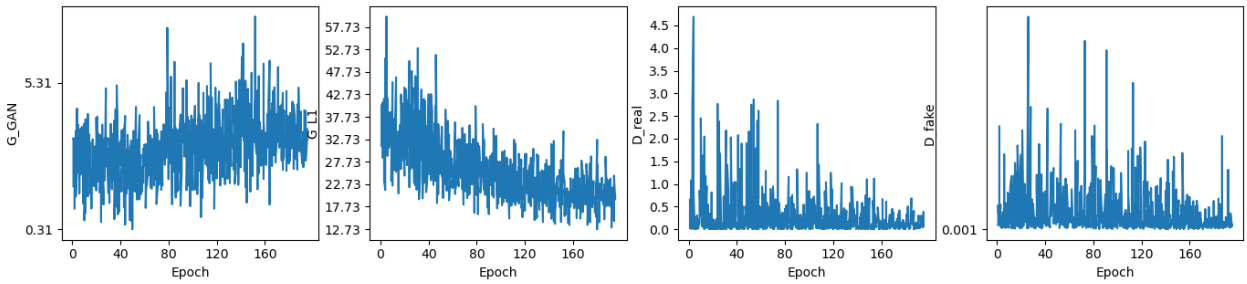
Mô hình	Dataset	Kiến trúc	Số lượng tham số	Môi trường
CycleGAN	braces2teeth braces2teeth Augmentation	Resnet: generator PatchGANs: discriminator	28.286 M (G_X: 11.378 M, G_Y: 11.378 M, D_X: 2.765 M, D_Y: 2.765 M).	Google Colab GPU (Pytorch)
Pix2Pix	teeth2	Unet: generator PatchGANs: discriminator	57.183 M ($G_{Y'Y}$: 54.414 M, D_Y: 2.769 M).	Google Colab GPU (Pytorch)



Hình 4.6. Quá trình huấn luyện trên môi trường Google Colab



Hình 4.7. CycleGAN loss trong quá trình huấn luyện



Hình 4.8. Pix2Pix loss trong quá trình huấn luyện

CHƯƠNG 5. ĐÁNH GIÁ

Thật sự rất khó để đánh giá ảnh do mô hình sinh ra có tốt hay không. Vậy nên, tôi sử dụng phương pháp đánh giá (5.2) giống như [15] đã đề xuất, tôi so sánh phương pháp của mình với một số phương pháp cơ sở (5.1) bao gồm COGAN [21], SimGAN [22] and BiGAN / ALI [26]. Cuối cùng là một số mẫu thực tế được mô hình xử lý.

5.1. Các phương pháp cơ sở

Các mô hình được trình phía dưới cũng là những mô hình được cải tiến từ GAN, tuy nhiên chúng vẫn có những nhược điểm mà các mô hình do tôi đề xuất sẽ khắc phục.

CoGAN [22] mô hình này bao gồm 2 Generator domain X và domain Y cộng thêm một số trọng số ràng buộc tại lớp latent. Việc chuyển đổi ảnh từ domain X sang domain Y có thể thực hiện bằng cách tìm latent phù hợp sinh ra x và chuyển đổi nó sang domain Y.

BiGAN/ALI [23] bao gồm một generator $G: Z \rightarrow X,$ với z là random noise. Mô hình này có thêm một ánh xạ nghịch đảo $F: X \rightarrow Z$.

SimGAN [24] tương tự CycleGAN, mô hình này sử dụng hàm adversarial loss để chuyển đổi từ X sang Y.

5.2. Chỉ số AMT

Chỉ số đánh giá truyền thống như per-pixel mean-squared trong một vài phương pháp trước đây không phù hợp để đánh giá chất lượng ảnh sinh ra nên được thay đổi chỉ số được xác định bằng người - Amazon Mechanical Turk (AMT) - độ linh hoạt về màu sắc và cấu trúc của hình ảnh đối với người quan sát hợp lý đến mức nào.

Để thiết lập cho quá trình thử nghiệm của mình, tôi sử dụng cùng phương thức giống nhau mô tả của tác giả Isola [15] nhưng có một ít thay đổi để phù hợp với điều kiện thực tế. Người quan sát (Turker) được chiếu một danh sách các cặp bao gồm một ảnh thật và

một ảnh giả được tạo ra bởi mô hình; ảnh giả nằm ở bên trái hoặc bên phải một cách ngẫu nhiên. Trong quá trình thử nghiệm, mỗi cặp ảnh sẽ được cho xuất hiện khoảng 1 - 2 giây; ngay sau đó, Turkers sẽ trả lời là bên nào là ảnh giả trong khoảng 5 – 10 giây (không có thông tin phản hồi về tính chính xác của đáp án sau mỗi câu hỏi). Mỗi phiên chỉ kiểm thử một mô hình và người tham gia chỉ được phép tham gia một phiên duy nhất. Do đó, chỉ số này chỉ nên được sử dụng để so sánh phương pháp hiện tại của tôi với các phương pháp cơ sở (được huấn luyện trong các điều kiện giống nhau).

Hình 5.1. Biểu mẫu đánh giá: Mỗi cặp ảnh giả, ảnh thật sắp xếp ngẫu nhiên bên trái hoặc bên phải

Ước lượng khoảng trung bình:

– Độ tin cậy $(1 - \alpha)$: 95%.

– Số lượng mẫu (n): 50.

Bảng 5.1. Chỉ số AMT trên mô hình do tôi đề xuất và các phương pháp cơ sở. Mặc dù có sử dụng dataset khác nhau để huấn luyện nhưng chỉ số vẫn dựa trên cùng một tập kiểm thử

Mô hình	Dataset	% Turker labeled real
CoGAN	braces2teeth	0.7% ± 7.5%
BiGAN/ALI	braces2teeth	2.1% ± 6.5%
SimGAN	braces2teeth	1.1% ± 7.7%
CycleGAN	braces2teeth	**28.7% ± 2.7%**
CycleGAN	braces2teeth Augmented	**32.8% ± 4.1%**
Pix2Pix	teeth2	**36.4% ± 4.2%**
Inpainting	Braces2teeth/teeth2	**1.7% ± 6.3%**

5.3. Thử nghiệm

Hình 5.2. Từ trái sang phải: ảnh gốc và ảnh do mô hình CycleGAN sinh ra

Hình 5.3. Từ trái sang phải: ảnh gốc, ảnh do mô hình Pix2Pix sinh ra

CHƯƠNG 6. ỨNG DỤNG MINH HỌA

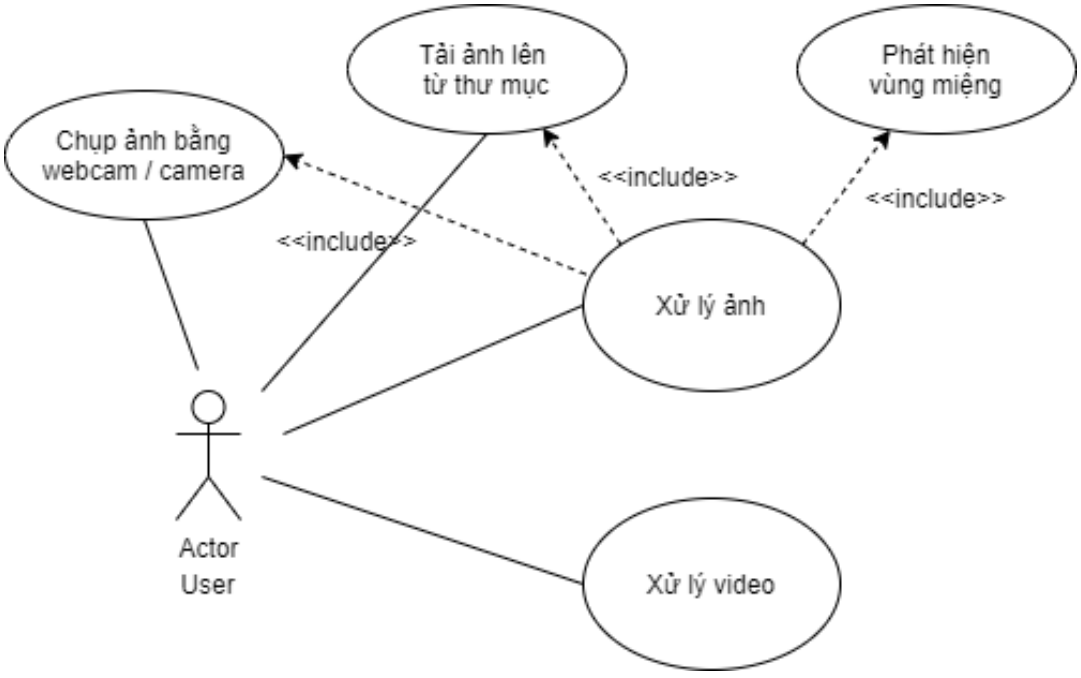
Nhằm biểu diễn kết quả về mô hình đã nghiên cứu, tôi đã tiến hành xây dựng ứng dụng minh trên nền tảng web cho phép nhập và lấy kết quả trả về từ mô hình một cách trực quan và thân thiện với người dùng hơn. Nội dung chương 6 bao gồm Phân tích, thiết kế ứng dụng (6.1) và Thực hiện (6.2).

6.1. Phân tích, thiết kế ứng dụng

6.1.1. Mô tả nghiệp vụ

Ứng dụng có thể tiếp nhận định dạng ảnh hoặc video, sau đó xử lý định dạng tương ứng và hiển thị kết quả trả về của mô hình một cách trực quan. Ứng dụng có thể chạy trên nhiều thiết bị khác nhau từ laptop, tablet đến smartphone.

6.1.2. Use case



Hình 6.1. Sơ đồ Use case

Mô tả chức năng:

- Xử lý ảnh: người dùng có thể tải ảnh răng có mắc cài hoặc chụp bằng webcam, sau đó ứng dụng trả về ảnh răng với mắc cài đã được xóa.
- Xử lý video: người dùng có thể tải video có sẵn, sau đó ứng dụng trả về video đã được xóa mắc cài.

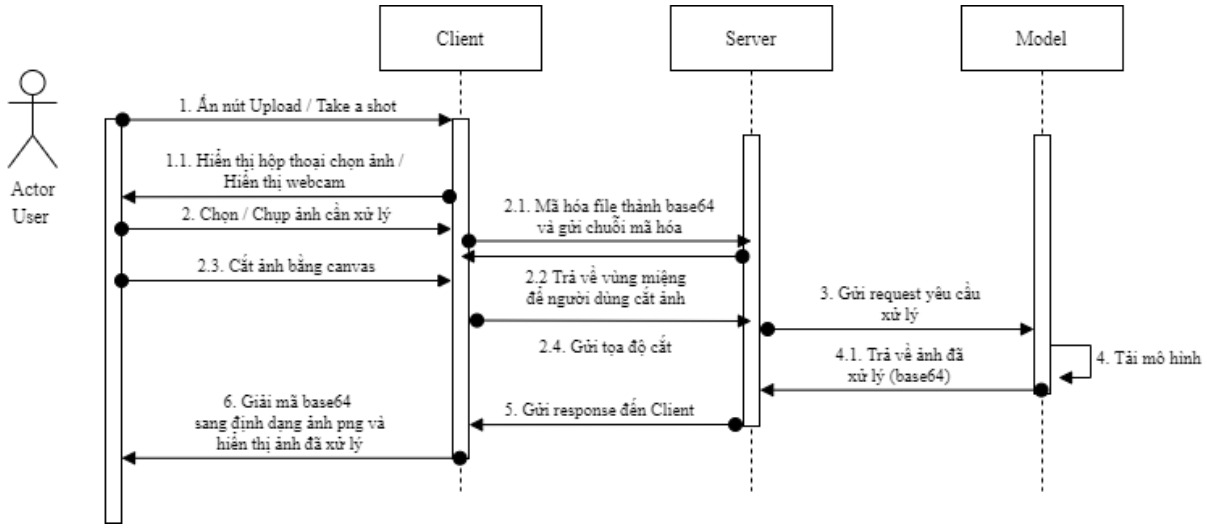
Bảng 6.1. Danh sách Actor

Actor	Ý nghĩa / Ghi chú
User	Là người tham gia quá trình sử dụng ứng dụng

Bảng 6.2. Danh sách Use case

Use case	Ý nghĩa / Ghi chú
Xử lý ảnh	Người dùng chọn và tải ảnh từ thư mục lên hoặc cho phép ứng dụng mở webcam và chụp ảnh từ webcam, mô hình sẽ tiếp nhận ảnh, trả về kết quả và ứng dụng sẽ hiển thị kết quả.
Xử lý video	Người dùng chọn và tải video từ thư mục lên, mô hình sẽ tiếp nhận video, trả về kết quả và ứng dụng sẽ hiển thị kết quả.

6.1.3. Sơ đồ tuần tự



Hình 6.2. Sơ đồ tuần tự Use case "Xử lý ảnh"

Mô tả:

(1) Người dùng ấn nút Upload, Client hiển thị hộp thoại chọn ảnh nếu đang ở chế độ tải ảnh lên từ thư mục. Hoặc người dùng ấn nút "Take a shot", Client yêu cầu người dùng cấp quyền sử dụng phần cứng nếu đang ở chế độ chụp ảnh bằng webcam / camera.

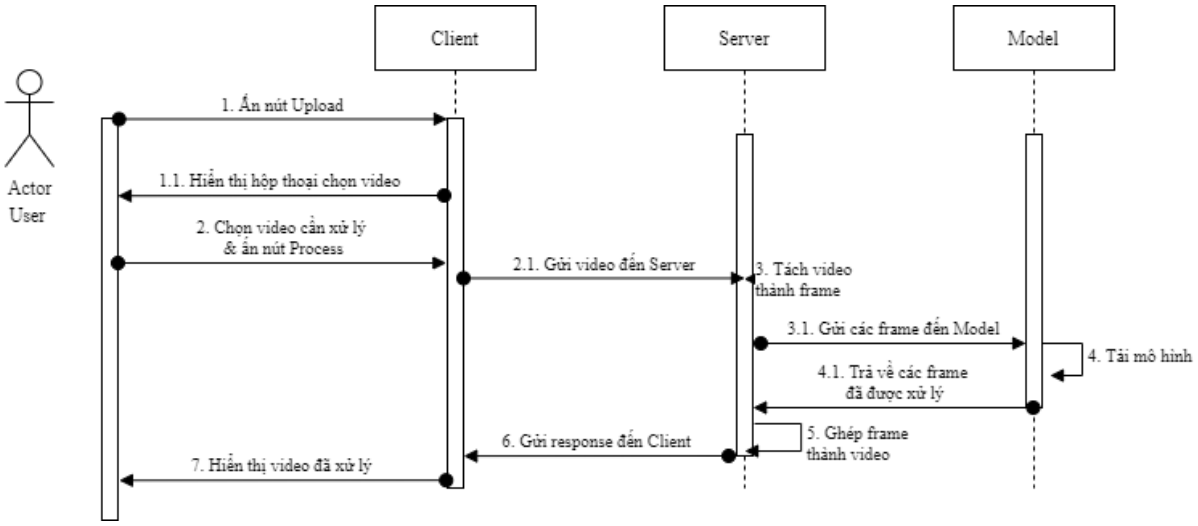
(2) Người dùng chọn / chụp ảnh và nhấn nút Process sau khi tải / chụp ảnh. Đồng thời, Client mã hóa ảnh thành chuỗi base64 và

(3) Client gửi request đến server.

(4) Model tải mô hình lên RAM, xử lý và tiến hành trả ảnh được xử lý.

(5) Server gửi response đến Client.

(6) Client nhận response và giải mã base64 sang định dạng ảnh, sau đó hiển thị ảnh cho người dùng.



Hình 6.3. Sơ đồ tuần tự Use case "Xử lý video"

Mô tả:

(1) Người dùng ấn nút Upload để tải video.

(2) Người dùng chọn video cần xử lý và ấn nút Process. Client gửi request đến server.

(3) Server tách video thành các frame và gửi đến Model.

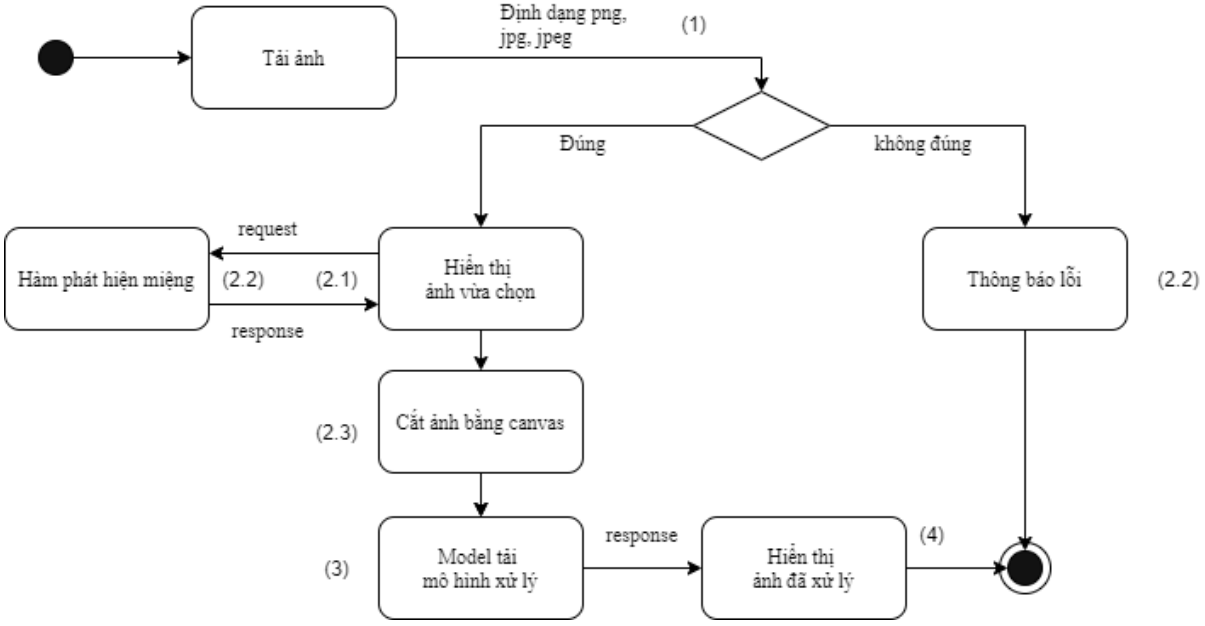
(4) Model tải mô hình lên RAM, xử lý và tiến hành trả ảnh được xử lý.

(5) Server ghép các frame thành video.

(6) Server gửi response đến Client.

(7) Client nhận response và hiển thị video cho người dùng.

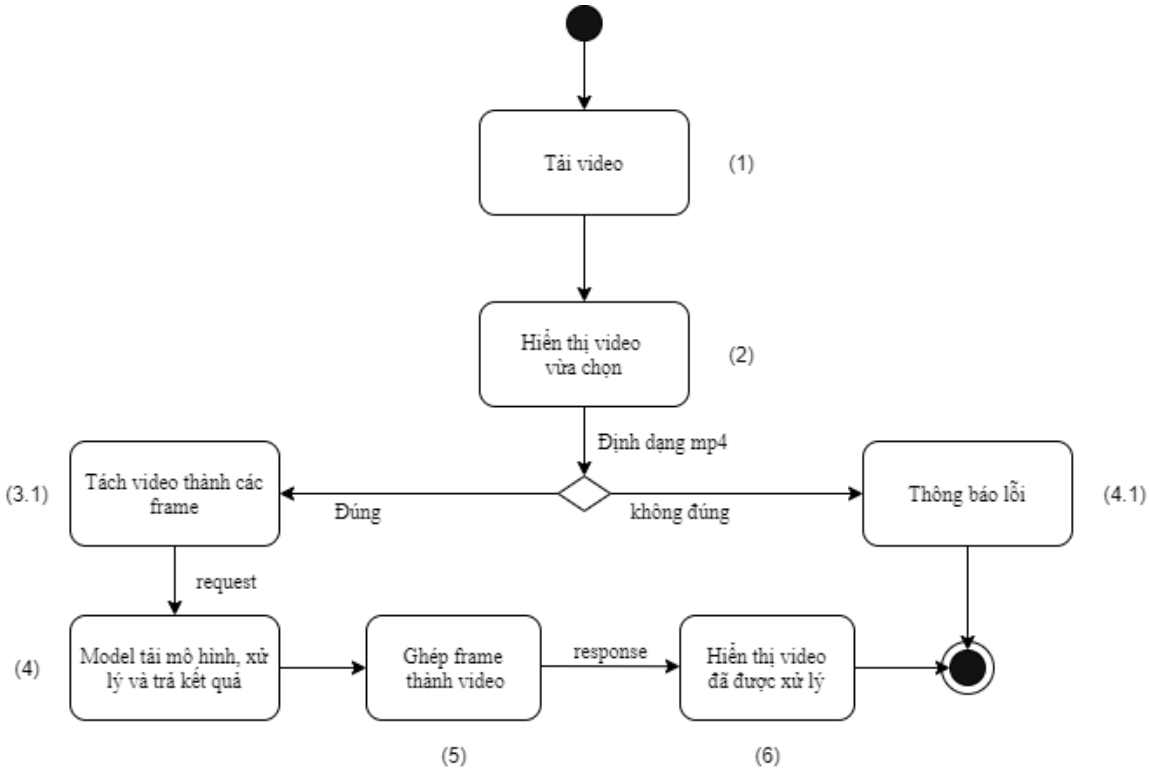
6.1.4. Sơ đồ trạng thái



Hình 6.4. Sơ đồ trạng thái Use case "Xử lý ảnh"

Mô tả:

- Người dùng chọn và tải ảnh từ thư mục lên hoặc người dùng cho phép ứng dụng mở webcam và chụp ảnh từ webcam (1). Client hiển thị ảnh vừa chọn (2.1). Server trả về canvas có vùng miệng được cắt sẵn, người dùng xác nhận và Server sẽ tiếp nhận ảnh cộng với tọa độ cắt, giao cho Model xử lý và trả về kết quả (3). Client sẽ hiển thị kết quả (4).
- Nếu ảnh tải lên không nằm trong các định dạng (png, jpeg, jpg, …) hoặc người dùng không cho phép ứng dụng truy cập webcam / camera thì Client sẽ thông báo lỗi (2.2).

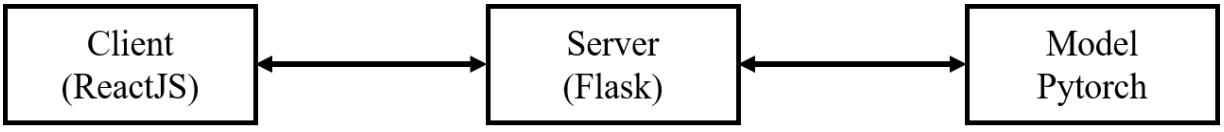


Hình 6.5. Sơ đồ trạng thái Use case "Xử lý video"

Mô tả:

- Người dùng chọn và tải video từ thư mục lên (1). Client hiển thị video vừa chọn (2). Server sẽ tiếp nhận video và tách video thành các frame (3.1). Model tiếp nhận các frame, xử lý và trả về kết quả (4). Server ghép các frame đã xử lý thành video và trả về Client (5). Client sẽ hiển thị video kết quả.
- Nếu video không nằm trong các định dạng mp4 thì Client sẽ thông báo lỗi (3.2).

6.1.5. Sơ đồ thành phần

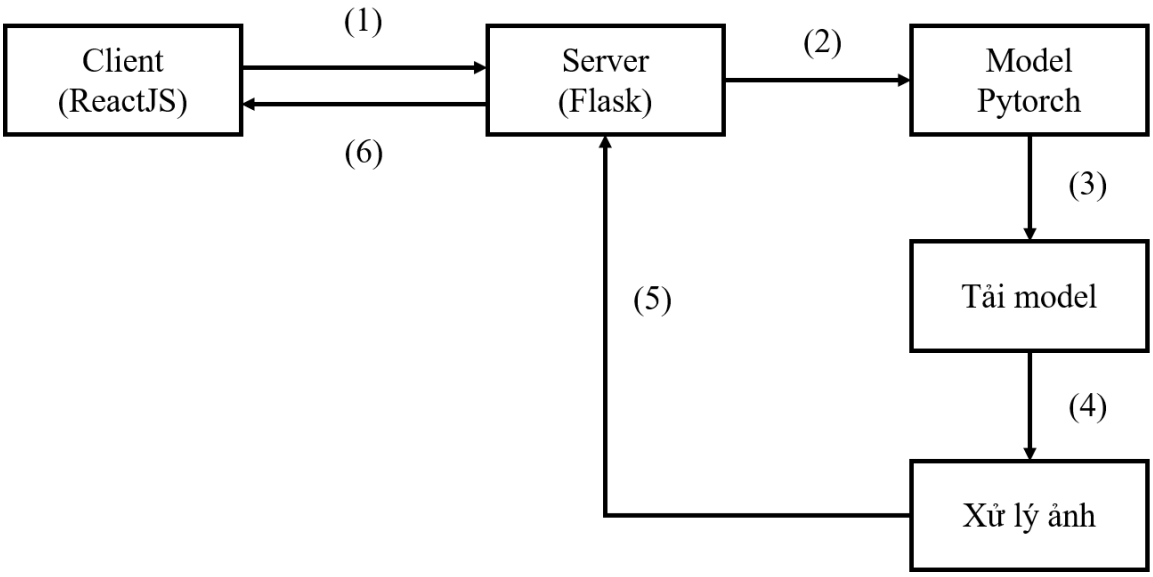


Hình 6.6. Sơ đồ các thành phần trong ứng dụng

Mô tả:

– Client: chịu trách nhiệm hiển thị UI, xử lý các sự kiện, gửi request và nhận response.
– Server: triển khai API và gọi thành phần Model phía dưới để xử lý. Xử lý đầu ra của Model. Xử lý request và gửi response đến Client
– Model: là thành phần xử lý ảnh đầu vào của ứng dụng, chứa những file thông tin về mô hình (số lượng lớp, chi tiết số lượng trọng số từng lớp, kích thước đầu ra, đầu vào) và file lưu trọng số đã được huấn luyện trước đó (latest_net_G.pth).

6.1.6. Sơ đồ hoạt động



Hình 6.7. Sơ đồ hoạt động của ứng dụng

Luồng dữ liệu được xử lý như sau:

(1) Đầu tiên, file định dạng ảnh / video xử được gửi dưới method POST.

(2) Server Flask tiếp nhận request, làm các thao tác tiền xử lý (giải mã base64 đối với ảnh và tách video thành frame đối với video). Đồng thời yêu cầu lớp Model xử lý các file này.

(3) Model tải generator Y (latest_net_G.pth) vào RAM.

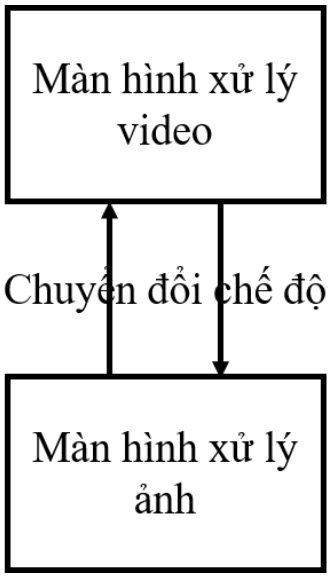
(4) Generator Y nhận file các ảnh chứa mắc cài và trả về các ảnh được xóa mắc cài.

(5) Server xử lý kết quả của Model (mã hóa ảnh sang base64 đối với đầu vào ảnh hoặc ghép frame thành video đối với đầu vào video).

(6) Server trả về status code và file, Client tiếp nhận và hiển thị kết quả.

6.1.7. Thiết kế giao diện

6.1.7.1. Sơ đồ luồng màn hình



Hình 6.8. Sơ đồ luồng màn hình của ứng dụng

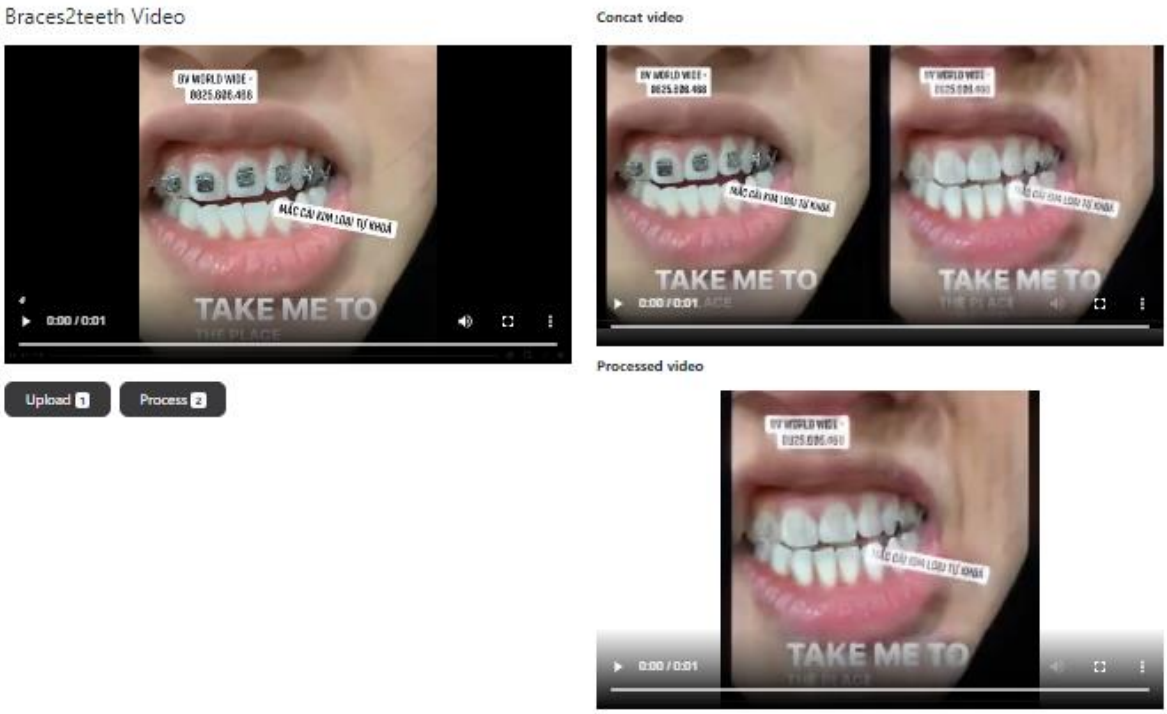
6.1.7.2. Mô tả chi tiết màn hình

Hình 6.9. Màn hình xử lý ảnh khi tải ảnh từ thư mục.

Hình 6.10. Màn hình xử lý ảnh khi chụp ảnh bằng webcam / camera

Bảng 6.3. Thành phần của màn hình trang chủ ở chế độ tải ảnh từ thư mục

Tên	Kiểu	Mô tả
ButtonUpload	Button	Cho phép người dùng upload ảnh từ thư mục.
ButtonProcess	Button	Gọi API /process từ server
Canvas	Canvas	Giao diện cho phép người dùng chọn phần ảnh phù hợp với tỉ lệ 1:1
Image	Image	Hiển thị ảnh vừa được chọn
ProcessedImage	Image	Hiển thị ảnh sau khi xử lý
Webcam	Webcam	Hiển thị đầu vào đang nhận từ webcam / camera.



Hình 6.11. Màn hình xử lý video

Bảng 6.4. Thành phần của màn hình trang chủ ở chế độ chụp ảnh bằng camera / webcam

Tên	Kiểu	Mô tả
ButtonCapture	Button	Cho phép người dùng lấy ảnh từ webcam / camera.
ButtonProcess	Button	Gọi API /process từ server
Image	Image	Hiển thị ảnh vừa được chọn
Webcam	Webcam	Hiển thị đầu vào đang nhận từ webcam / camera.

6.2. Thực hiện

6.2.1. Nghiên cứu và lựa chọn công nghệ

Dựa vào các lí thuyết đã nghiên cứu ở các chương trên về mô hình cũng như công nghệ, tôi thấy được sự thuận lợi và khó khăn khi nghiên cứu cũng như triển khai đề tài nên đã đưa ra lựa chọn sau để phù hợp với tính chất nghiên cứu.

- ReactJS (Client): là một library Javascript cho các ứng dụng Single Page. ReactJS cho phép nhúng Javascript vào HTML khiến cho việc xử lý sự kiện, tiếp nhận request và xử lý response đơn giản và nhanh chóng.
- Flask (Server): là một Web Framework rất nhẹ của Python, dễ dàng giúp người dùng tạo ra website nhỏ. Flask được kết hợp với Pytorch để triển khai API.
- Pytorch (Model): là một framework được xây dựng dựa trên python cung cấp nền tảng tính toán khoa học. Một platform học sâu có nhiệm vụ khởi tạo mô hình và sẵn sàng để xử lý ảnh đầu vào.
- Ngoài ra, ứng dụng còn sử dụng một số công nghệ hỗ trợ khác bao gồm: Torch Vision, PIL (lưu và đọc ảnh), Flask CORS (tránh chính sách bảo mật CORS) , React Bootstrap (tạo UI đẹp mắt hơn), Redux (quản lý trạng thái của UI), React Webcam (cho phép truy truy cập Webcam / Camera), …

6.2.2. Kiểm thử và triển khai

Về phía server Flask, tôi triển khai trên môi trường Google Colab (host) kết hợp với domain miễn phí tại Freedom (http://test.braces2teeth.tk/) thông qua service của Clondflare.

Về phía client ReactJS, tôi triển khai trên service miễn phí của Vercel (https://braces2teeth.vercel.app/)

CHƯƠNG 7. KẾT LUẬN

7.1. Kết quả đạt được

Qua khóa luận này tôi đã nghiên cứu những giải pháp cho bài toán braces2teeth. Bằng việc phân tích các yêu cầu liên quan và các mô hình khác nhau, tôi đã chọn được mô hình CycleGAN (với dataset không theo cặp), Pix2Pix (với dataset theo cặp) và Inpainting (với yêu cầu chỉnh sửa kết quả đến từ người dùng).

Tôi đã công bố công trình nghiên cứu trên nhiều hội nghị, kỷ yếu và tạp chí khác nhau và cũng đã xây dựng được một ứng dụng minh họa mô hình đơn giản.

Hạn chế:

Mặc dù CycleGAN, Pix2Pix và Inpainting sau khi áp dụng thêm một số phương pháp tiền xử lý đã xử lý tốt nhiều trường hợp nhưng vẫn còn những ngoại lệ chưa thể xử lý hoàn hảo như đã trình bày trong phần 3.1, bao gồm:

- Mắc cài sứ, mắc cài đa sắc: nguyên nhân cơ bản vẫn là do phân phối của dataset chưa đủ để thể hiện tất cả các trường hợp của bệnh nhân đeo niềng răng, ví dụ mô hình không thể xử lý loại niềng răng vô hình hoặc mắc cài màu sắc độc lạ như màu tím vì trong dataset không có những trường hợp đó. Tôi cũng cảm nhận thấy sự khác biệt về mặt kết quả nếu có thể áp dụng triệt để hướng giám sát so với không giám sát. Tuy nhiên, như đã đề cập đến, việc chuẩn bị các cặp dữ liệu bao gồm ảnh đang trong quá trình niềng răng và ảnh sau khi tháo niềng răng mà không cần qua quá trình tiền xử lý gặp rất nhiều khó khăn.
- Mắc cài trong ảnh không rõ ràng (khu vực chứa mắc cài mờ): có thể xử lý bằng một mô hình nội suy ảnh khác để giảm độ mờ của ảnh, tuy nhiên việc thiếu thông tin về

kết cấu răng và mắc cài có thể mang đến cho người dùng cảm giác không chính xác.

- Mắc cài trong ảnh chiếm tỉ lệ nhỏ (khu vực chứa mắc cài có diện tích nhỏ so với diện tích toàn bộ ảnh) khiến mô hình gặp nhiễu: như đã trình bày, tình trạng này có thể khắc phục bằng các mô hình phân đoạn / phát hiện đối tượng trong ảnh (cụ thể là miệng). Sau đó trích xuất vùng miệng, đưa vào mô hình và thay thể vùng đã trích xuất bằng ảnh đã xử lý. Tuy nhiên, tại các biên (vùng tiếp giáp của ảnh mới và ảnh củ) có hiện tượng inconsistency (mô tả ở hình 2.34) do phân phối của hai ảnh này là hoàn toàn khác nhau, vấn đề này đã được khắc phục ở mô hình Inpainting nhưng chi phí huấn luyện mô hình này vẫn còn quá đắt đỏ và kém hiệu quả.

7.2. Thuận lợi và khó khăn

7.2.1. Thuận lợi

- Giảng viên hướng dẫn tận tâm, có kiến thức sâu rộng, hỗ trợ tối đa từ việc định hướng đề tài, cung cấp thiết bị, dữ liệu cho đến hướng dẫn viết báo cáo.
- Thường xuyên trao đổi, thuyết trình đề tài nghiên cứu với các nhóm khác nên được học hỏi và trao đổi rất nhiều.
- Các mô hình cơ sở công khai miễn phí bài báo khoa học và các số liệu liên quan.

7.2.2. Khó khăn

- Kiến thức mới lạ, không đúng chuyên ngành nên khó khăn trong việc tìm hiểu cũng như ứng dụng.
- Nguồn dữ liệu còn tương đối hạn chế.
- Điều kiện thực nghiệm khó khăn, yêu cầu phần cứng mạnh.
- Một số trường hợp chưa thể xóa triệt để mắc cài.

7.3. Hướng phát triển

- Tìm kiếm thêm những trường hợp dataset vẫn đang còn thiếu về những mắc cài hiếm như mắc cài đa sắc, mắc cài sứ và mắc cài vô hình.
- Nghiên cứu những mô hình mới và ưu việt hơn, bao gồm: Inpainting Error Maximization, Image Matting, Flow-edge Guided Video Completion, ...
- Nghiên cứu thêm những bài toán tương tư về loại bỏ các vật thể khác trên khuôn mặt như sẹo, hình xăm, kính, ...

TÀI LIỆU THAM KHẢO

Bài báo khoa học

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2009), 2009.

[2] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 341–346. ACM, 2001.

[3] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on, volume 2, pages 1033–1038. IEEE, 1999.

[4] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5505– 5514, 2018.

[5] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. ACM Transactions on Graphics (TOG), 36(4):107, 2017.

[6] Chao Yang, Yuhang Song, Xiaofeng Liu, Qingming Tang, and C-C Jay Kuo. Image inpainting using block-wise procedural training with annealed adversarial counterpart. arXiv preprint arXiv:1803.08943, 2018.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.

[8] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. In ICLR, 2017.

[9] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In ICML, 2016.

[10] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by Inpainting. CVPR, 2016

[11] S. Vicente, V. Kolmogorov and C/ Rother.. Graph cut based image segmentation with connectivity priors. In IEEE conference on computer vision and pattern recognition (pp. 1-8). IEEE, 2008.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028, 2017.

[13] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

[14] Zhu, Jun-Yan, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE international conference on computer vision. 2017.

[15] Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[16] Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[17] Li, Zhengqin, and Jiansheng Chen. "Superpixel segmentation using linear spectral clustering." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[18] Van den Bergh, Michael, et al. "Seeds: Superpixels extracted via energy-driven sampling." European conference on computer vision. Springer, Berlin, Heidelberg, 2012.

[19] Achanta, Radhakrishna, et al. "SLIC superpixels compared to state-of-the-art superpixel methods." IEEE transactions on pattern analysis and machine intelligence 34.11 (2012): 2274-2282.

[20] NAIK, D., et al. Fast interactive superpixel based image region generation. 2019.

[21] Liu, Ming-Yu, and Oncel Tuzel. "Coupled generative adversarial networks." Advances in neural information processing systems. 2016.

[22] Shrivastava, Ashish, et al. "Learning from simulated and unsupervised images through adversarial training." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[23] Liu, M. Y., Breuel, T., & Kautz, J. (2017). Unsupervised image-to-image translation networks. Advances in neural information processing systems, 30, 700-708.

[24] Mejjati, Y. A., Richardt, C., Tompkin, J., Cosker, D., & Kim, K. I. (2018). Unsupervised attention-guided image-to-image translation. In Advances in Neural Information Processing Systems (pp. 3693-3703).

[25] Din, N. U., Javed, K., Bae, S., & Yi, J. (2020). A novel GAN-based network for unmasking of masked face. IEEE Access, 8, 44276-44287.

[26] Dumoulin, Vincent, et al. "Adversarially learned inference." arXiv preprint arXiv:1606.00704 (2016).

Mã nguồn mở

[27] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix.

[28] https://github.com/JiahuiYu/generative_inpainting.

Tài liệu khác:

[29] https://machinelearningcoban.com/

[30] https://nttuan8.com/

PHỤ LỤC

Phụ lục 1: Bản sao bài báo Kỷ yếu Hội nghị AI gặp gỡ lãnh đạo Thành phố Hồ Chí Minh, Hội nghị khoa học Trẻ & nghiên cứu sinh UIT

Tái tạo ảnh răng từ ảnh niềng răng có mắc cài sử dụng GAN

Vũ Tuấn Hải[1]

[1] Đại học Công nghệ Thông tin - ĐHQG TP. HCM, TP. Hồ Chí Minh, Việt Nam

Hình 1: Với hai tập dữ liệu về braces và teeth, model có thể chuyển từ ảnh phía bên trái sang ảnh phía bên phải

Tổng quan. Xử lý ảnh đã được ứng dụng sâu rộng trong nhiều ngành nghề, đặc biệt trong y tế và chuyên ngành nha khoa. Trong lĩnh vực nha khoa, bệnh nhân niềng răng thường có một số vấn đề phát sinh do quá trình điều trị lâu dài, như việc bệnh nhân hay tự ti khi trò chuyện, chụp ảnh vì mắc cài khiến nụ cười không được đẹp. Do đó, trong bài báo này, tôi đề xuất một số phương pháp giải quyết bài toán xóa mắc cài bằng việc nghiên cứu và so sánh hai generative model: Pix2Pix và CycleGAN, model đầu tiên là ánh xạ $F: Y \to Y$, mode thứ hai là ánh xạ $G: X \to Y$ với X là phân phối ảnh răng chứa mắc cài (braces - Hình 1, bên trái mỗi cặp) và Y là phân phối ảnh răng không chứa mắc cài (teeth - Hình 1, bên phải mỗi cặp). Khi model được nhúng vào camera, người dùng có thể thoải mái trò chuyện, chụp ảnh mà không cần lo ngại về nụ cười của mình.

Từ khóa: braces2teeth, cycleGAN, generative model, graph – cut, image inpainting, WGAN - GP.

1 Giới thiệu

Bài toán Img2Img (chuyển đổi ảnh sang ảnh) là một bài toán rộng trong xử lý ảnh, model Img2Img có thể chuyển đổi ảnh x bất kì sang ảnh y trong ngữ cảnh nào đó cho trước. Một số ứng dụng nổi bật như chuyển từ ảnh xám sang ảnh màu, ghép ảnh hoặc tăng độ phân giải, … Inpainting là tập hợp của Img2Img, là quá trình sinh ra những phần bị thiếu hoặc cần thay thế trong ảnh sao cho thực tế và chính xác về mặt ngữ nghĩa. Quá trình này có thể ứng dụng để loại bỏ vật thể hoặc chỉnh sửa các vùng trong ảnh

theo mong muốn. Trong nội dụng bài báo này, bài toán cụ thể là xóa mắc cài ra khỏi ảnh.

Từ những năm 2000 đã tồn tại hai hướng giải quyết bài toán Inpainting bao gồm: PatchMatch [1] sử dụng cho ảnh có đặc trưng / độ phân giải thấp và generative model. Một số phương pháp tiếp cận trước đây [2, 3] có thể hồi phục các kết cấu tĩnh nhưng không ổn định với những vật thể phức tạp như khuôn mặt và đồ vật. Cách tiếp cận thứ hai [4, 5, 6] có thể khai thác những đặc trưng từ dataset cho trước và xử lý những vấn đề cụ thể liên quan đến dataset.

Trong bài báo này, tôi sẽ sử dụng hai phương pháp thuộc cách tiếp cận thứ hai, hai phương pháp này bao gồm supervised và unsupervised, tôi sẽ gọi hai model tương ứng là model A và model B. Trong cách tiếp cập này, chúng ta huấn luyện model A, trong dataset mỗi ảnh đầu vào x sẽ có ảnh đầu ra y tương ứng, dataset là tập hợp các cặp $\{x_i, y_i\}_{i=1}^{N}$. Trong quá trình training, tôi đề xuất việc tạo ra y_i từ chính x_i sau khi xóa ngẫu nhiên vùng nào đó, như vậy cả x_i lẫn y_i đều thuộc domain Y. Model A chỉ sử dụng lớp teeth trong dataset. Tôi sẽ hiện thực hóa ý tưởng này bằng model Pix2Pix, về kiến trúc cụ thể, tôi tham khảo phần lớn trong [4]. Tại giai đoạn kiểm thừ, để xóa chính xác phần mắc cài, tôi sử dụng model GraphCut vì tính đơn giản và nhanh chóng của nó.

Có một vấn đề là việc tạo ra dataset bao gồm những cặp như trên trong thực tế lại khó khăn và tốn nhiều chi phí. Trong bài toán của chúng ta, ảnh chụp trong quá trình niềng răng và sau khi niềng răng thường không liên quan nhiều với nhau có thể do khác góc độ chụp hoặc thể trạng bệnh nhân. Để tăng tính khả thi, phương pháp thứ hai sẽ huấn luyện model theo hướng unsupervised, model này có thể học được mối quan hệ giữa hai tập dataset X và Y mà không cần từng cặp mẫu. Tôi sẽ chọn CycleGAN vì nhiều điểm phù hợp so với yêu cầu trên.

2 Nghiên cứu liên quan

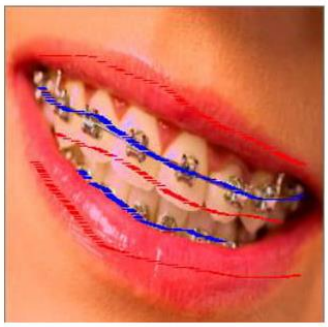
2.1 GAN

Generative Adversarial Networks (GAN) [7, 8] đã đạt được nhiều kết quả ấn tượng trong ứng dụng sinh ảnh và sửa ảnh. Những nghiên cứu gần đây về Conditional GAN đã cho ra nhiều ứng dụng như text2image [9], image inpainting [10], … Chìa khóa khiến GAN thành công là ý tưởng hàm adversarial loss giúp generator, theo lý thuyết sẽ cho ra kết quả khiến chuyên gia cũng không thể phân biệt thật giả.

2.2 GraphCut

GraphCut [11] là phương pháp tách vật thể cổ điển hiệu quả áp dụng lý thuyết đồ thị. Cho đồ thị $G(E, V)$, cut là đường cắt đi qua đồ thị, mincut là đường cut với tổng số lượng edge cắt qua là nhỏ nhất và ngược lại với maxcut.

Trong network dòng chảy (flow network), với mỗi edge (u, v) trong G có trọng số c là giá trị thực không âm. Giả sử có hai node, sink node (đỉnh thu) và source node (đỉnh phát) xác định. s - t cut là đường cut chia G thành hai đồ thị con S và T sao cho



$\forall s \in S, s$ là node chứa pixel là foreground và $\forall t \in T, t$ là node chứa pixel là background. Chúng ta gọi s - t cut là maxflow khi tổng trọng số của các edge mà đường cut đi qua là cực đại và ngược lại với minflow. Đường phân đoạn foreground và background chính là là đường cut thỏa mãn điều kiện mincut và maxflow. Để tìm đường phân đoạn này, tôi sử dụng thuật toán Boykov - Kolmogorov. Ngoài ra để giảm độ phức tạp tính toán khi ánh xạ ảnh sang đồ thị, tôi sử dụng kĩ thuật simple linear Iterative cluster (SLIC) để gom từng cụm pixel giống nhau thành từng superpixel.

Hình 2: GraphCut nhận ground truth và user interaction với user interaction gồm những điểm màu xanh đánh dấu foreground và điểm màu đỏ đánh dấu background.

2.3 WGAN - GP

WGAN - GP [12] là biến thể của GAN [7, 8] và cải tiến từ WGAN [13], sử dụng metric Wasserstein W đánh giá độ tương đồng giữa 2 phân phối xác suất P_r (tập ảnh thật) và P_g (tập ảnh giả). Wasserstein W khác với phân kỳ Kullback - Leibler và phân kỳ Jensen - Shannon ở đặc trưng đạo hàm mượt tại mọi điểm. Trong loss function của WGAN - GP, mục tiêu vẫn là tối thiểu hóa $W(P_r, P_g)$ (hay generator sinh ra ảnh gần như ảnh thật nhất). Để tính toán W theo công thức gốc sẽ tương đối phức tạp, do đó trong [13], Gulrajani và cộng sự đã sử dụng đối ngẫu Kantorovich - Rubinstein để đơn giản hóa việc tính toán:

$$W(P_r, P_g) = \sup_{D \in \mathcal{D}} E_{x \sim P_r}[D(x)] - E_{\tilde{x} \sim P_g}[D(\tilde{x})] \quad (1)$$

Trong đó $\mathcal{D}$ là tập hợp các hàm 1 - Lipschitz thỏa điều kiện:

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2|) \quad (2)$$

Như vậy mục tiêu mới của chúng ta là tìm (mô phỏng) f bằng một deep neural network F nào đó với layer cuối có đầu ra là vô hướng, tượng trưng cho chất lượng ảnh x.

Để F hội tụ, chúng ta giới hạn tập weight $\{w\}$ trong miền compact $(-c, c)$ với c là hyperparameter:

$$w \leftarrow clip(w, -c, c) \quad (3)$$

Thực nghiệm cho thấy c nằm ở khoảng 0.01. Tuy nhiên với model khác hoặc dataset khác, c có thể thay đổi. Yếu điểm này được khắc phục trong gradient penalty (GP) [12] như sau:

Lấy $\hat{x} = \varepsilon x + (1 - \varepsilon)\tilde{x}$ là phân phối mẫu, chúng ta cộng vào $W(P_r, P_g)$ đại lượng sau, khi tối thiểu hóa hàm loss, F sẽ hội tụ:

$$\lambda\left(\left|\left|\nabla_{\hat{x}}D_w(\hat{x})\right|\right|_2 \odot (1-m) - 1\right)^2, \lambda = 10 \text{ (m = 0 nếu là mask)} \quad (4)$$

Hàm loss của model WGAN - GP:

$$L = D_W(\tilde{x}) - D_W(x) + \lambda\left(\left|\left|\nabla_{\hat{x}}D_w(\hat{x})\right|\right|_2 \odot (1-m) - 1\right)^2 \quad (5)$$

2.4 CycleGAN

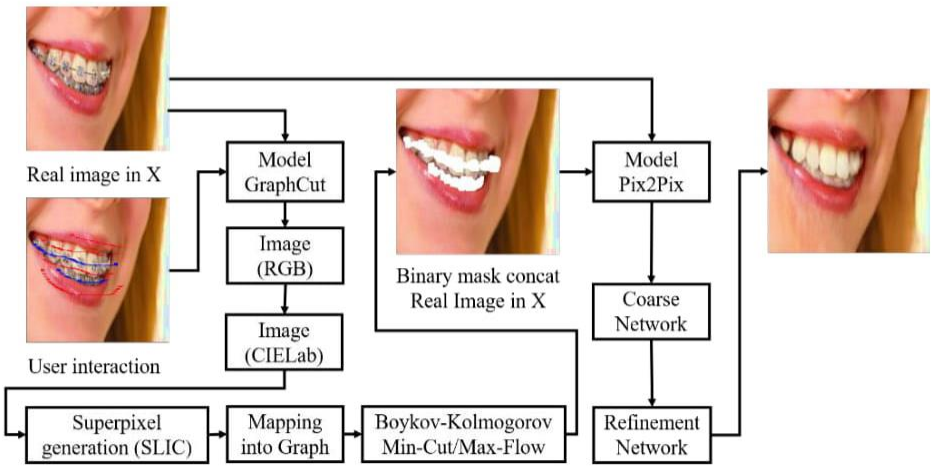
CycleGAN [14] là một biến thể của GAN [7, 8] với hàm loss được cộng thêm đại lượng cycle consistency loss $CCL = F\big(G(X)\big) - X$ với ánh xạ $G: X \to Y$ và ánh xạ ngược $G^{-1} = F = Y \to X$. CCL đánh giá chất lượng ảnh sinh ra và được tính toán cụ thể là:

$$L_{cyc} = E_{a\sim p_{data}(a)}[\|G_{BA}\big(G_{AB}(a)\big) - a\|]_2 + E_{b\sim p_{data}(b)}[\|G_{AB}\big(G_{BA}(b)\big) - b\|]_2 \quad (6)$$

Hàm loss của model CycleGAN (λ là hyperparameter, λ = 10):

$$L = L_{AB} + L_{BA} + \lambda L_{cyc} \quad (7)$$

3 Hướng tiếp cận



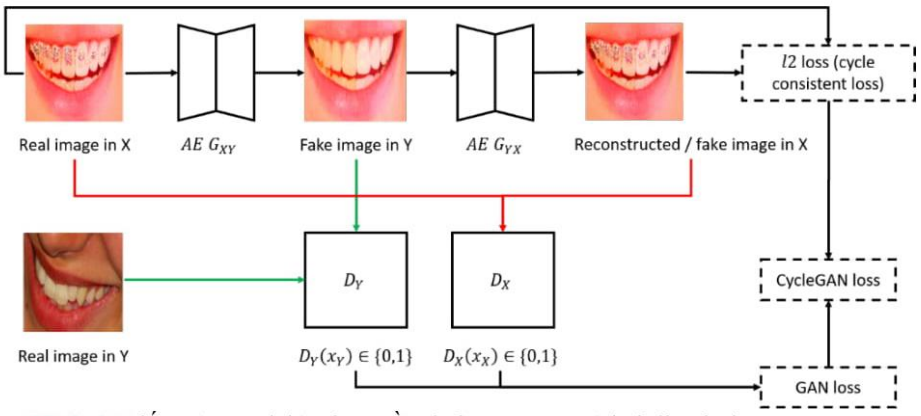
Hình 3: Kiến trúc model A. Đầu ra của GraphCut là binary mask. Pix2Pix nhận binary mask và ground truth để khôi phục những điểm ảnh có giá trị 0 trên binary mask màu trắng).

Ở model A, sau khi nhận ground truth, model GraphCut chuyển ảnh từ không gian màu RGB sang không gian màu CIELab, sau đó sử dụng SLIC để giảm độ phức tạp. Từ user interaction và tập superpixel, model ánh xạ sang đồ thị G. Sử dụng thuật toán

Boykov – Kolmogorov, G được cắt thành 2 đồ thị con S và T. Cuối cùng, model ánh xạ S sang binary mask là đầu vào của model Pix2Pix. Ở model Pix2Pix, ảnh sinh ra trải qua 2 giai đoạn:

Giai đoạn 1 (Coarse network): giai đoạn làm thô, missing region được khôi phục sao cho có phân phối giống với background. Giai đoạn này model sử dụng Spatial discounted reconstruction loss để cực tiểu hóa theo khoảng cách từ boundary đến từng missing pixel, khoảng cách càng nhỏ thì pixel phục hồi càng khớp với background. Các lớp convolutional sử dụng filter là Dilated layer.

Giai đoạn 2 (Refinement network): giai đoạn làm mịn. Sử dụng hai network độc lập Contextual Attention và Dilated Convolutional, sau đó concat kết quả với nhau. Contextual Attention Network thực hiện 3 bước: bước 1 - rút trích các mảnh (patch) 3 x 3 từ background và reshape thành filter, bước 2 - missing region được đưa qua convolutional network (với filter từ bước 1) và layer cuối cùng sử dụng hàm softmax, bước 3 - thu được ma trận Attention Score, dựa vào Attention Score, chúng ta xác định được chính xác giá trị pixel phục hồi dựa vào những pixel thuộc background. Dilated network mở rộng receptive field với mục đích tổng quát hóa và có kiến trúc tương tự với coarse network. Lý do cần sử dụng hai network với cùng mục đích là để ảnh phục hồi đạt được cả hai yếu tố: chất lượng (resolution) - đặc trưng chi tiết và tính hợp lý (consistency) - đặc trưng tổng thể.



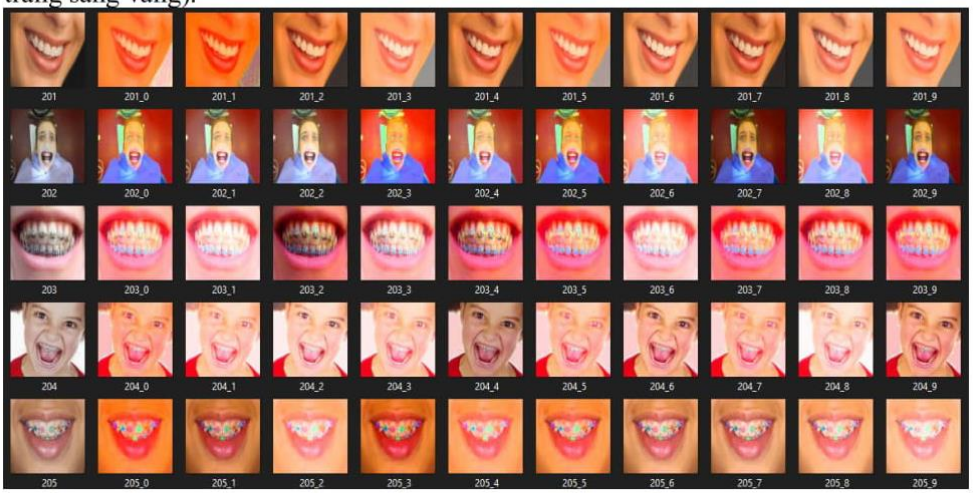
Hình 4: Kiến trúc model B, bao gồm hai generator và hai discriminator.

Ở model B, giai đoạn xóa và khôi phục được thực hiện đồng thời dựa vào ý tưởng autoencoder (AE). Ground truth x thuộc domain X qua AE network đầu tiên có được ảnh fake y' trong domain Y, y' tiếp tục đi qua AE network thứ hai (đối xứng với AE network đầu tiên) để phục hồi x (x'). $l2$ loss (x, x') chính là CCL. Model có D_X phân biệt x và x', D_Y phân biệt y và y'.

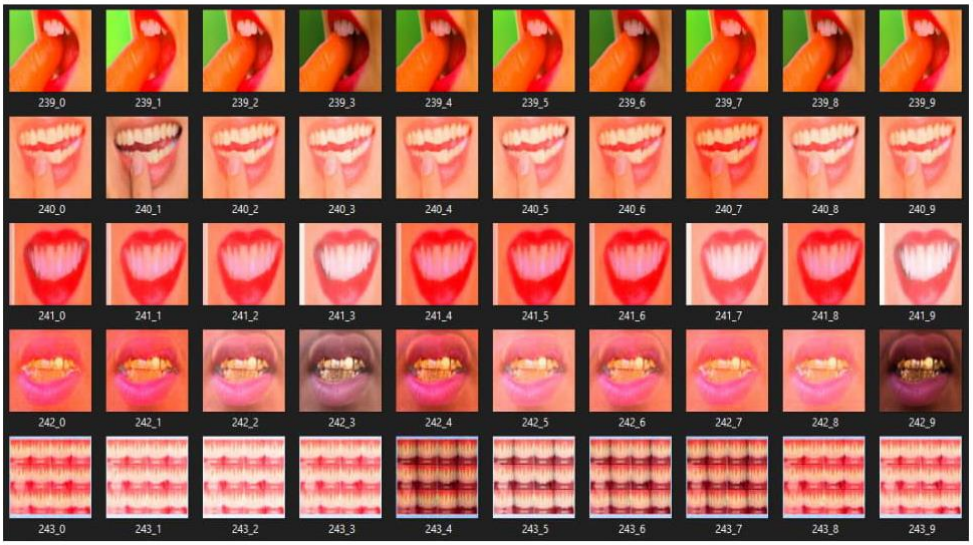
4 Thử nghiệm

Tôi đánh giá hai model của mình thông qua dataset braces2teeth với nguồn ảnh được crawl từ google image. Dataset có 22020 ảnh RGB bao gồm hai lớp là braces (Hình 5) và teeth (Hình 6), được xử lý về kích thước 256 x 256 và loại bỏ nhiễu. Dataset được

tăng cường bằng việc thay đổi ngẫu nhiên trong khoảng từ 0 - 50 trên hai kênh màu S và V (trong không gian HSV) để tạo sự đa dạng trong độ sáng ảnh và màu răng (từ trắng sang vàng).



Hình 5: Ảnh lớp braces trong dataset braces2teeth.



Hình 6: Ảnh lớp teeth trong dataset braces2teeth.

4.1 Model A

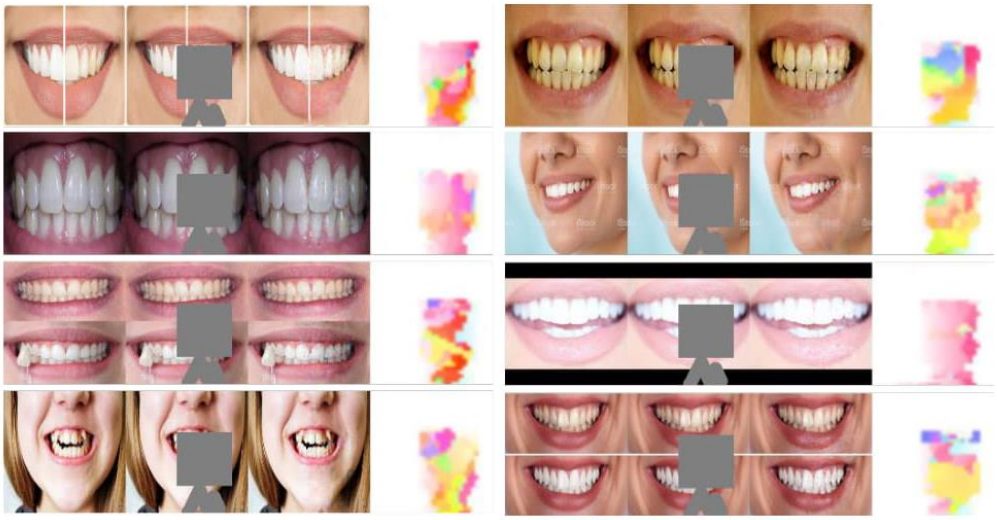
Model A chỉ sử dụng lớp teeth và được chia làm 2 tập bao gồm training (9490 ảnh) và validation (2390 ảnh). Model có tổng cộng 9999294 weight, được thực thi bằng Tensorflow 1.15, trên phần cứng CPU Intel Xeon E3 12xx v2 (Ivy Bridge) 3GHz (8

processor), không có GPU. Model thử nghiệm xử lý khoảng 4s trên CPU với ảnh 256 x 256. Thời gian training đến là 39 ngày, tổng số epoch hoàn thành là 4.

4.2 Model B

Model B sử dụng hai lớp ảnh và chia làm 4 tập con bao gồm trainA (braces, 8040 ảnh), testA (braces, 2010 ảnh), trainB (teeth, 9560 ảnh), testB (teeth, 2390 ảnh). Model có tổng cộng 28286000 weight (G_A: 11378000, G_B: 11378000, D_A: 2765000, D_B: 2765000) và được thực thi trên môi trường Google Colab có GPU. Model được training trong 200 epoch trong vòng 1 tuần.

5 Kết quả



Hình 7: Kết quả trong quá trình training model A, Lần lượt từ trái sang phải: ground truth, ground truth concat binary mask, ảnh sau phục hồi và ma trận Attention Score.

Hình 8: Kết quả sau quá trình training model A khi thử nghiệm với ảnh thực tế.

Hình 9: Kết quả trong quá trình training từ model B. Lần lượt từ trái sang phải: ground truth và ảnh sau phục hồi.

5.1 Đánh giá

Với model A, GraphCut gặp hạn chế ở ảnh có độ phân giải thấp và trung bình, thường xóa lan ra những vùng lân cận (Hình 8, phía bên trái). Ảnh sinh ra từ model Pix2Pix bị phụ thuộc vào dataset và số lượng epoch nên chưa đạt chất lượng tốt (Hình 8, phía bên phải).

Hình 10: Kết quả sau quá trình training từ model B. Lần lượt từ trái sang phải: ground truth và ảnh sau phục hồi.

Với model B đã xử lý tốt với những trường hợp thông thường nhưng chưa xử lý được một số ngoại lệ như mắc cài không rõ - bị che khuất (Hình 10, hàng 1 bên trái), mắc cài đa sắc (Hình 10, hàng 4 bên phải) hoặc mắc cái quá dày (Hình 10, hàng 1 bên phải).

5.2 Kết luận

Model A theo hướng supervised cần sử dụng nhiều tài nguyên tính toán và dataset lớn để đạt được độ chính xác cao hơn, do số lượng epoch hạn chế khiến model bị underfitting nên việc đánh giá model nào có kết quả tốt hơn chưa thực hiện được. Vì không có cặp $\{x_i, y_i\}$ nên ngoài việc sử dụng các metric như FCN score, cần có đánh giá của chuyên gia liên quan về những điểm chưa hợp lý của ảnh sinh ra (thực nghiệm).

5.3 Hướng phát triển

Model A có ưu điểm là có thể chỉ định những vùng cần thay đổi. Vì model A có tiềm năng giải quyết trọn vẹn bài toán, nên tôi đề xuất việc sử dụng phần cứng mạnh tiếp tục training để có kết quả so sánh khách quan với model B. Model B có thể khắc phục những trường hợp ngoại lệ bằng việc sử dụng thêm model detect mouth để giới hạn vùng xử lý chỉ trên một phần ảnh, hoặc chọn ngưỡng động để xóa được những mắc cài đa sắc.

Tài liệu tham khảo

1. C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2009), 2009.
2. Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 341–346. ACM, 2001.
3. Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on, volume 2, pages 1033–1038. IEEE, 1999.
4. Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5505– 5514, 2018.
5. Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. ACM Transactions on Graphics (TOG), 36(4):107, 2017.
6. Chao Yang, Yuhang Song, Xiaofeng Liu, Qingming Tang, and C-C Jay Kuo. Image inpainting using block-wise procedural training with annealed adversarial counterpart. arXiv preprint arXiv:1803.08943, 2018.
7. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.

10

8. J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. In ICLR, 2017.
9. S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In ICML, 2016.
10. D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. CVPR, 2016
11. S. Vicente, V. Kolmogorov and C/ Rother.. Graph cut based image segmentation with connectivity priors. In IEEE conference on computer vision and pattern recognition (pp. 1-8). IEEE, 2008.
12. I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028, 2017.
13. M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.
14. Zhu, Jun-Yan, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE international conference on computer vision. 2017.

Phụ lục 2: Bản sao bài báo Kỷ yếu chung kết Eureka lĩnh vực Công nghệ thông tin

TÁI TẠO ẢNH RĂNG TỪ NIỀNG RĂNG CÓ MẮC CÀI SỬ DỤNG GAN

Vũ Tuấn Hải
Trường Đại học Công nghệ thông tin - Đại học Quốc gia Thành phố Hồ Chí Minh
*Tác giả liên lạc: 3520433@gm.uit.edu.vn

TÓM TẮT

Xử lý ảnh đã được ứng dụng sâu rộng trong nhiều ngành nghề, đặc biệt trong y tế và nha khoa. Trong lĩnh vực nha khoa, bệnh nhân niềng răng thường có một số vấn đề phát sinh do quá trình điều trị lâu dài, như việc hay tự ti khi trò chuyện, chụp ảnh vì mắc cài khiến nụ cười không được đẹp. Do đó, trong bài báo này, chúng tôi đề xuất một số phương pháp giải quyết bài toán xóa mắc cài một cách triệt để bằng việc nghiên cứu và so sánh hai generative model: Pix2Pix và CycleGAN, model đầu tiên là ánh xạ F: Y→Y, model thứ hai là ánh xạ G:X→Y với X là phân phối ảnh niềng răng chứa mắc cài và Y là phân phối ảnh răng không chứa mắc cài.

***Từ khóa**: braces2teeth, CycleGAN, Pix2Pix, Inpainting.*

RECOVER TEETH PHOTO FROM BRACES PHOTO USING GAN

Vu Tuan Hai
University of Information Technology - VNU Ho Chi Minh City
*Corresponding author: 3520433@gm.uit.edu.vn

ABSTRACT

Image processing has been widely applied in many fields, especially in the medical and dental fields. In the dental field, patients with braces often have some problems that arise due to long-term treatment, such as the patient's self-esteem when talking and taking pictures because braces make the smile not beautiful. Therefore, in this article, we propose several methods to thoroughly solve the brace removal problem by studying and comparing two generative models: Pix2Pix and CycleGAN, the first model is the F mapping: Y → Y, the second model is the G: X → Y mapping where X is the braces image distribution containing braces and Y is the teeth image distribution.

***Từ khóa**: braces2teeth, CycleGAN, Pix2Pix, Inpainting.*

TỔNG QUAN

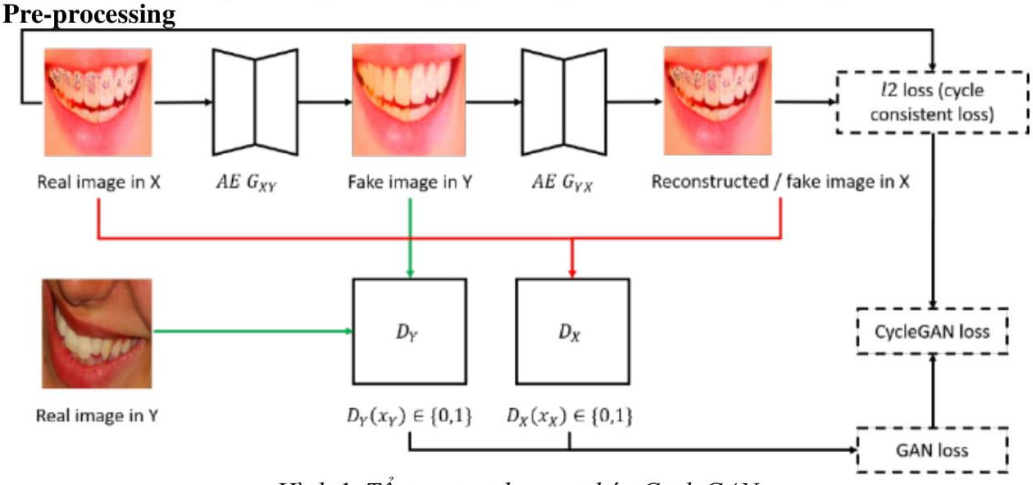
Img2Img (chuyển đổi ảnh sang ảnh) là một bài toán rộng trong xử lý ảnh, model Img2Img có thể chuyển đổi ảnh x sang ảnh y trong ngữ cảnh nào đó cho trước. Một số ứng dụng nổi bật như chuyển từ ảnh xám sang ảnh màu, ghép ảnh hoặc tăng độ phân giải, … Trong nội dung bài báo này, bài toán cụ thể là chuyển đổi ảnh niềng răng có mắc cài sang ảnh răng, hay nói cách khác là xóa mắc cài ra khỏi ảnh. Với tính khả quan khi khai thác những đặc trưng từ dataset cho trước và xử lý những vấn đề cụ thể liên quan đến dataset, chúng tôi lựa chọn sử dụng model deep learning kết hợp với nhiều phương pháp tiền xử lý khác. Chúng tôi đã thử nghiệm theo hai hướng là supervised và unsupervised.

VẬT LIỆU VÀ PHƯƠNG PHÁP NGHIÊN CỨU

CycleGAN

Ground truth x thuộc domain X qua AE (auto-encoder) network đầu tiên có được y_{fake} trong domain Y, y' tiếp tục đi qua AE network thứ hai (đối xứng với AE network đầu tiên) để phục hồi x (x'). $l2loss\ (x, x')$ được gọi là cycle – consistency loss. Model có D_X phân biệt x và x', D_Y phân biệt y và y'. Chúng tôi thử nghiệm với tập test 200 ảnh niềng răng có mắc cài.

Pre-processing



Hình 1. Tổng quan phương pháp CycleGAN

a. Data augmentation

Với mỗi điểm dữ liệu trong dataset braces2teeeth, chúng tôi thay đổi ngẫu nhiên kênh màu S và V trong không gian HSV để màu răng và màu mắc cài được thay đổi 10 lần khác nhau. Sau khi thử nghiệm, những mắc cài có màu sắc sặc sỡ đã được xóa, tuy nhiên model đã thay đổi sắc thái của ảnh khá nhiều.

b. Crop mouth

Phương pháp cắt ra vùng miệng, bằng dlib và OpenCV để tạo facial landmark detection. Phần mouth được đánh dấu bởi các point trên landmark có index từ 61 đến 68.

Chúng tôi sử dụng hàm OpenCV convex Rectangle để và lấy thêm nửa trên và nửa dưới để ảnh không bị biến dạng khi đưa vào model. Phương pháp này sẽ được áp dụng chủ yếu trong model Pix2Pix.

c. Cân bằng Histogram

Chúng tôi tiến hành phân tích histogram của trung bình RGB trong tập 200 ảnh test. Những trường hợp failure có 2 đỉnh ở khoảng 130 – 140 và 220 – 230 vì elastic tie thường có màu sắc sặc sỡ (sáng). Chúng tôi thử nghiệm việc cân bằng histogram của những trường hợp failure với histogram trung bình của những trường hợp success bằng phương pháp nội suy tuyến tính.

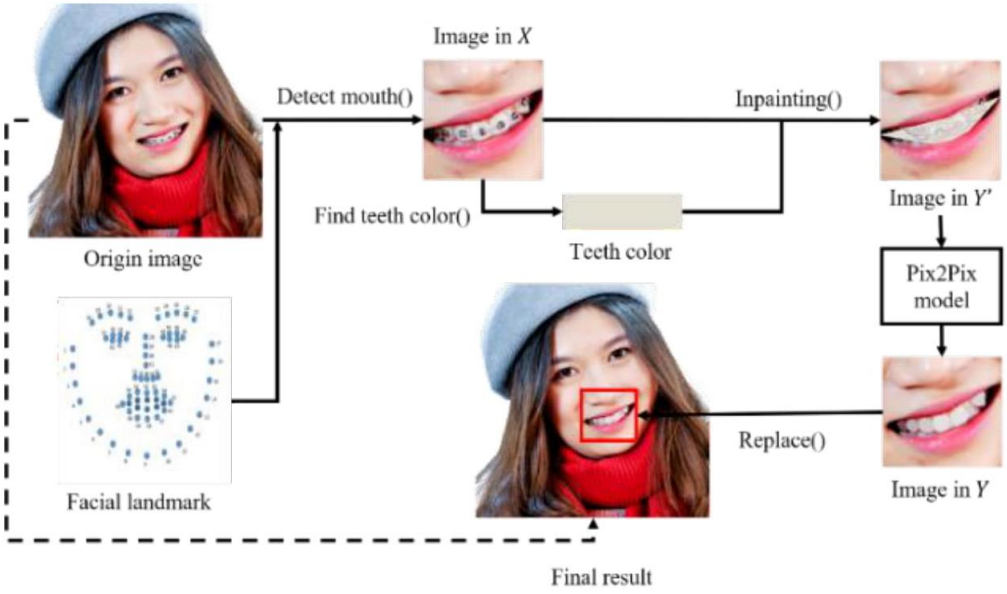
d. Tìm màu răng

Chúng tôi sử dụng bảng màu dựa trên VITA3D Master để tham chiếu trực tiếp đến những điểm ảnh lấy được có khả năng là răng. Sau đó tìm màu răng bằng lấy tập pixel thông qua 2 đường trung trực của ảnh và so sánh pixel có màu gần nhất với màu nào đó trên bảng mẫu.

e. Xóa mắc cài

Chúng tôi sử dụng hàm convexPolygon trong OpenCV các point có index từ 61 – 68 trên facial landmark để xác định selection zone. Chúng tôi thử nghiệm 3 phương pháp khác nhau để so sánh: a) Những pixel nằm trong selection zone được inpainting lại bằng màu răng. b) Những pixel nằm trong selection zone và có khoảng cách với màu răng lớn hơn ngưỡng cho trước sẽ được inpainting lại bằng màu răng. Tại đây chúng tôi chuyển ảnh sang không gian màu CIELab và sử dụng khoảng cách CIE2000. c): bao gồm 3 công đoạn: chuyển ảnh sang ma trận

superpixel - đánh nhãn những superpixel có density bé hơn threshold là “teeth”, giữ nguyên, ngược lại thì đánh dấu là “not teeth” - đối với những superpixel có label “not teeth”, chúng tôi tìm những pixel xung quanh thuộc



Hình 2. Tổng quan phương pháp Pix2Pix with pre-processing

superpixel được gán nhãn “teeth” và lấy trung bình màu của chúng, nếu không có neighbour teeth nào thì superpixel “not teeth” được inpaint bằng màu răng.

PixPix

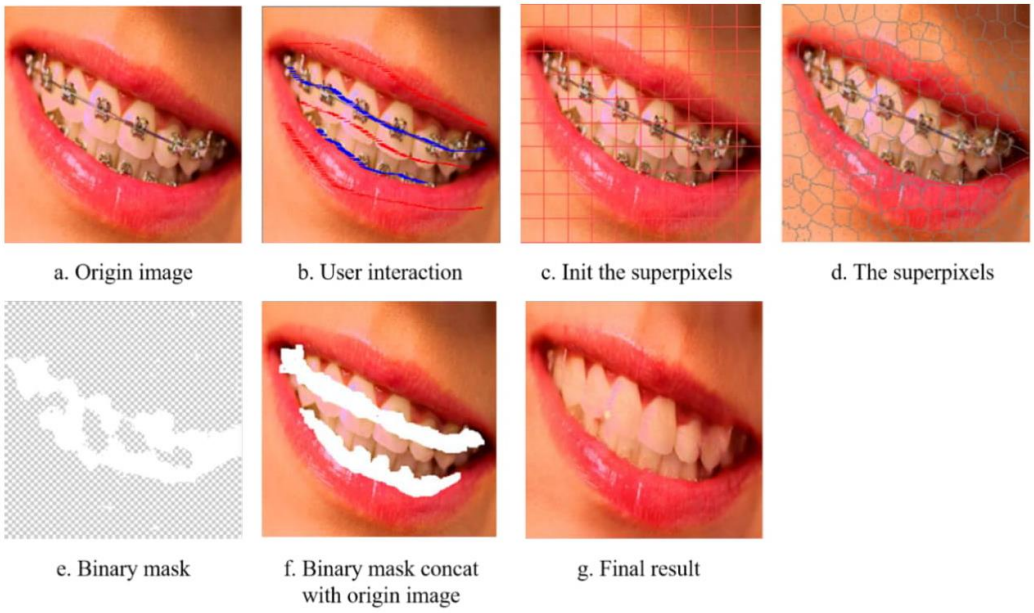
Chúng tôi vẫn train model Pix2Pix với dataset là tập các cặp $\{x_i, y_i\}_{i=1}^{N}$. Tuy nhiên, việc chuẩn bị dataset thành các cặp trong thực tế rất khó khăn và tốn nhiều chi phí. Ảnh chụp trong quá trình niềng răng và sau khi niềng răng thường không liên quan đến nhau do khác góc độ chụp hoặc thể trạng bệnh nhân đã thay đổi từ thời điểm niềng răng đến thời điểm tháo niềng. Do đó, mỗi điểm dữ liệu x_i sẽ tương ứng điểm dữ liệu y_i ($x_i = prePocessing(y_i)$), cả x_i lẫn y_i đều thuộc domain Y, để dễ phân biệt chúng tôi sẽ gọi x_i là y'_i. Cuối cùng, chúng tôi thử nghiệm thêm một model trong đó cho phép sự can thiệp từ người dùng nhằm điều chỉnh kết quả theo ý thích của họ.

GraphCut + Generative Image Inpainting with Contextual Attention (Inpainting)

Kết quả do hai model đã tương đối, tuy nhiên có một số trường hợp người dùng muốn can thiệp để đầu ra đạt được kết cấu theo như ý muốn. Do đó chúng tôi kết hợp hai model GraphCut và Generative Image Inpainting with Contextual Attention để hiện thực hóa nhu cầu trên (tôi sẽ gọi model này là Inpainting để ngắn gọn). Model GraphCut sẽ nhận vào ảnh gốc (hình . a) và user interaction (hình 3. b) đánh dấu vị trí mắc cài (màu đỏ đánh dấu không phải mắc cài và màu xanh đánh dấu mắc cài). Model GraphCut ánh xạ ảnh sang tập superpixel (hình 3. c và 3. d) và sử dụng thêm thông tin từ user interaction để ánh xạ tiếp sang đồ thị ba chiều. Cuối cùng đưa đồ thị ba chiều vào thuật toán của Boykov-Kolmogorov Min-cut/Max-follow để thu được 2 đồ thị con S là tập các superpixel được đánh dấu là braces và T là tập các superpixel được đánh dấu là teeth. Chúng tôi ánh xạ ngược lại S và T về ảnh gốc và có được một binary mask với giá trị 1 đánh dấu mắc cài và giá trị 0 đánh dấu không phải mắc cài (hình 3. e).

Model Inpainting nhận ảnh gốc và binary mask, tiến hành inpainting tại những vị trí đã được đánh dấu trên binary mask và cuối cùng trả về ảnh răng không chứa mắc cài.

KẾT QUẢ VÀ THẢO LUẬN



Hình 3. Tổng quan phương pháp Inpainting

Để xác định chất lượng ảnh sinh ra braces → teeth, chúng tôi sử dụng phương pháp Amazon Mechanical Turk (AMT). Những người tham gia được cho xem một chuỗi các cặp ảnh, một là ảnh thật và một là ảnh giả (được tạo bởi model) và yêu cầu chọn hình ảnh mà họ cho là thật. Mỗi người tham gia chỉ được phép thử nghiệm một lần duy nhất. Tất cả các model đều được đánh giá trên cùng một test dataset.

Model	**Training dataset**	**% Turker labeled real**
CoGAN	braces2teeth	0.7% ± 0.5%
BiGAN/ALI	braces2teeth	2.1% ± 0.5%
SimGAN	braces2teeth	1.1% ± 1%
Inpainting	braces2teeth/teeth	1.7% ± 0.3%
CycleGAN	**braces2teeth**	**28.3% ± 4.3%**
CycleGAN + Augmented data	**braces2teeth Augmented**	**32.8% ± 4.7%**
Pix2Pix + preprocessing	**teeth2**	**38% ± 5.2%**

Bảng 1. AMT score trên các model khác nhau, in đậm là các phương pháp chúng tôi tử nghiệm

KẾT LUẬN VÀ ĐỀ NGHỊ

Mặc dù CycleGAN và những model áp dụng thêm những phương pháp pre-processing đã xử lý tốt nhiều trường hợp những vẫn còn những ngoại lệ chưa thể xóa mắc cài hoàn hảo. Nguyên nhân cơ bản vẫn là do phân phối của dataset chưa đủ để thể hiện tất cả các trường hợp của bệnh nhân đeo niềng răng, ví dụ model không thể xử lý niềng răng vô hình, hoặc mắc cài màu sắc độc lạ như màu tím vì trong dataset không hề có những trường hợp đó. Chúng tôi cũng cảm nhận thấy sự khác biệt về mặt kết quả nếu có thể áp dụng triệt để hướng supervised so với unsupervised. Tuy nhiên, việc chuẩn bị các cặp điểm dữ liệu thật bao gồm ảnh đang trong quá trình niềng răng và ảnh sau khi tháo niềng răng mà không cần qua quá trình tiền xử lý gặp rất nhiều khó khăn như đã mô tả. Lý do model Inpainting đạt kết quả khá thấp vì còn phụ thuộc vào số lượng epoch hoàn thành và chất lượng tương tác đến từ người dùng. Tuy nhiên chúng tôi nghĩ rằng model vẫn có khả năng xử lý tốt hơn nếu được đầu tư về sức mạnh tính toán.

TÀI LIỆU THAM KHẢO

1. Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5505– 5514, 2018.
2. Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. ACM Transactions on Graphics (TOG), 36(4):107, 2017.
3. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.
4. J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. In ICLR, 2017.
5. S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In ICML, 2016.
6. S. Vicente, V. Kolmogorov and C/ Rother.. Graph cut based image segmentation with connectivity priors. In IEEE conference on computer vision and pattern recognition (pp. 1-8). IEEE, 2008.
7. I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028, 2017.
8. Zhu, Jun-Yan, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE international conference on computer vision. 2017.
9. ISOLA, Phillip, et al. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. p. 1125-1134.
10. Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
11. NAIK, D., et al. Fast interactive superpixel based image region generation. 2019.

Phụ lục 3: Bản sao bài báo Tạp chí BME

Reconstructed teeth image from braces with GAN

Vu Tuan Hai[1], Dang Thanh Vu[2], Huynh Ho Thi Mong Trinh[1], Pham The Bao[3]

[1] University of Information Technology – VNU, Vietnam
[2]Chonnam National University, Rep. of Korea
[3] Sai Gon University, Vietnam

Abstract

Recent advances in deep learning models have shown promising potential in object removal, which refers to the task of replacing undesired objects with appropriate pixel values using known context. Object removal-based deep learning can commonly be solved by modeling it as Inpainting or image to the image translation problem. Instead of dealing with a large context, this paper aims to a specific application of object removal, which is to erase braces trace our of an image of teeth with braces (brace2teeth). In particular, we divide the problem into three circumstances corresponding to different data sets. Firstly, we consider pairs of images with and without braces as a dataset to train the Pix2Pix model. In the second case, we use CycleGAN to deal with unpairs of the same kind of dataset. In the last case, we utilize graph cut combining inpainting model to build an interactive user interface. Moreover, we enhance the reconstructed image quality by preprocessing the training data and proposing an interpolation scheme to fill in the teeth' pixel values. To our best knowledge, this study is one of the first attempts to take the brace2teeth problem into account by using deep learning techniques. The experimental results are reported both in quantitive and quality.

Keywords: braces2teeth, CycleGAN, Pix2Pix.

1. Introduction

Image processing based deep learning model has been widely applied in various fields, especially in medical and dental images. In dentistry, braces patients often suffer some problems that arise due to long-term treatment, such as the patient's self-esteem because braces make smiles awkward when talking and taking pictures. As demand for braces removal in an image has arisen, this study addresses the brace2teeth problem by modeling it as the image to image translation task that could be solved by deep learning approaches such as generative models.

Brace2teeth problem is the task that fills plausible contents in where are obscured by braces with realistic teeth' texture details. Traditional approaches mostly develop techniques combined with multi-steps, segmentation to locate obstacles' areas where masks are marked and filled in by a hole-filling algorithm [1], [2], [3]. However, it is not trivial to do a complete segment without any good prior such as instance-level labels and advanced architectures [4]. Likewise, braces segmentation is a nuisance since traces are normally thin and hooks have the same color as teeth.

1.1. Related works

For a brevity of related studies, we review some of remarkable studies in this field below. We divided the current related work into three aspects. First is several basic img2img translation methods. Then, other studies modified the basic model to obtain good images but still for general usage. Recent studies tried to leverage img2img methods to remove unwanted objects in an image.

Img2Img translation: Img2Img translation is a task that aims to translate from an image to another with some particular purposes, such as changing the attribute of textures or semantic content. A variety of methods using deep convolutional neural networks have been developed for Img2Img translation. Noticeably, GANs-based methods have been proposed and attained remarkable results in generating desired images with high texture detail and resolution

GANs are generative models that learn to map the output image from a random noise vector z to y, $G: z \rightarrow y$ [7]. In the same year, based on the previous study, conditional GANs (cGAN) [16] enhanced mapping from an observed image x and random noise vector z, to y, $G: \{x, z\} \rightarrow y$. Generator G is trained to create performances that are impossible to discern from real images by an opponent trained discriminator, D, trained to differentiate the false image produced by G against the real image. GAN is an important model, so many research studies greatly improve the quality of the picture produced.

Pix2Pix [15] is a typical case of this work that adopts cGAN [16] to learn the mapping in a supervised way. In the Pix2Pix model, the generator is trained to generate realistic images in the target domain corresponding to the source domain's input images. This mechanism is restricted by an adverse loss deemed a sufficient structural constraint of high and low frequencies, respectively. Pix2Pix must, however, be focused on paired samples because of its supervised loss. CycleGAN [14] imposes cycle continuity to overcome these shortcomings, relying on a reasonable reconstruction of an image after two translations but the lack of ground truths makes this method's results are not as good as Pix2Pix. This principle enables the preparation of img2img models using unpaired datasets. These classic img2img translation models are the foundation of our process.

Augmented image for img2img translation: The latest analysis has aimed to enhance the results of models with a universal or particular function. [23] Focusing on the effect of latent features and changing the design of the proposed model to boost performance. [24] Recommended the inclusion of useful components to boost image generation, such as adding an attention layer to the model. Some approaches in the above study have enabled us to enhance the outcome of the teeth reconstruction mission.

Object removal [25]: is the study which dealt with the same problem as ours. Their model can automatically remove face masks in the image and recreating the erased part. Two operations are conducted in two different modules and the output of the mask deletion module will be entered in the removed zone restore module. While they have obtained a successful outcome in eliminating a particular object, their models tend to be heavy, and they cannot process which face masks are not in the dataset. Moreover, the idea of constructing a binary mask to delete an object from the Unet model is only useful for convex objects like masks, not braces. Their study has inspired us in creating paired datasets, and the way to deal with an object has various shapes, sizes, color and structure like face mask and braces.

1.2. Contributions

Apart from traditional approaches, this paper suggests using generative models to solve the brace2teeth problem. We design the problem as an image-to-image translation task where the input image is an image of teeth with braces, and the output image is a teeth image with braces already erased. The translation models used in this study is Pix2Pix and CycleGAN, which are alternatively applied on pair and unpair dataset. An advantage of our proposed modeling is that braces can be erased just using a unified model. The data sets are built only on the set of braces and without braces images, without any consideration to segmentation or the cost of labeling. We further aim towards a user application with the proposal of an interactive framework using Grapcut and Inpainting. The user can select the region of braces that should be removed and the region of teeth that should be retained. Since image-to-image translation models have suffered background inconsistency and incomplete fine-grained structures, we alleviate it by applying background removal and superpixel refinement to create nature-looking images.

To our knowledge, this paper presents the very first attempt to solve the brace2teeh problem using generative models. In summary, this paper has some contributions:

- We collect a dataset with 8704 images about teeth and 2165 images about braces. The dataset is available for practitioners who have an interest in this subject.
- To erase braces and construct a clean teeth image, we model the problem as an image-to-image translation that can be solved by generative models such as Pix2Pix (for pair dataset) and CycleGAN (for unpair dataset).
- We propose an image editing tool with Graph cut combining Inpainting to use useful prior knowledge to improve the model's performance.

This paper is structured as follows: Section 2 is Materials and Methods, it gives details of our approach and analyzes failure cases with proposed solutions. Section 3 presents our experimental results with plenty of settings. Section 4 discusses the metrics. Finally, section 5 is the debate and section 6 is the conclusion.

2. Materials and Methods

2.1. Materials

Datasets: to evaluate our proposed methods, we test various datasets, including two domains, braces and teeth. Because of our problem's specifics, the datasets about braces and teeth have not existed or public until now, so we have collected ourselves from google image and another GAN model (StyleGAN2). There is also a small amount offered from dental clinics.

We get google images with the Downloader tool provided by Yabin Zheng by using many keywords in different languages related to braces and teeth. The results from this source are only temporarily accepted with 0 – 100 images per keyword. StyleGAN2 is a huge model that comes from the researcher of NVIDIA, it produces human faces public on this with different contours, and we take this image resource by auto crawling using Selenium web driver (1024 x 1024 x 3) technology, using part of preprocessing module to get mouth part with click size 256 x 256 x 3. Image from StyleGAN2 is not limited, but the teeth of all images are quite similar (all white, bright, and beautiful), so we make them a minority of the datasets. The source from dental clinics are valuable, but most are kept private, and we are only given a small number of them.

All images in the three datasets are engineered before training. This task includes resizing 256 x 256 x 3 using bicubic interpolation, reducing noise by cropping doodle parts, which are not too relevant, and edit images (the braces that are not located correctly be adjusted near the center section manually).

We have three datasets for testing; therefore, CycleGAN uses the full dataset, and Pix2Pix only uses the teeth class. Below are the details; braces2teeth is collected from Google image, teeth2 and dental clinics, braces2teeth Augmentation is from braces2teeth by using augmentation technique, we change the H and S value randomly ten times to get more color of braces and make braces2teeth more diverse.

Training: details of training on each of these datasets are provided in the supplemental materials online. Qualitative results are shown in Figures 10 and Figure 11.

Experimental Settings: because of the result from [14], we reused their stable model and not changed much about the hyperparameters. The loss functions are the same, and we use Adam optimizer with batch size 1 too. All models were trained from scratch with a learning rate of 0.0002. We also keep the first 100 epochs' learning rate and linearly decay it to zero over the next 100 epochs. All the parameters in the training process are summarized in Table 2.

The detail of the dataset, training process and experiment are available at https://github.com/vutuanhai237/braces2teeth.

2.2. Methods

Our goal is to learn mapping functions f between two domains, braces (X) and teeth (Y) given training samples $x_{i=1}^{n}$ where $x_i \in X$ and $y_{j=1}^{m}$ where $y_j \in Y$. f can remove braces and inpainting automatically.

2.2.1. CycleGAN

CycleGAN [14] is a good choice for this task since the learned translation degenerates into making minor improvements to the input and its structures adapted to the better results of presentation changes, not geometric changes. Because of the same goal, we selected the CycleGAN as the first model to be evaluated.

CycleGAN includes two mappings $G: X \rightarrow Y$ and $F: Y \rightarrow X$. It has two adversarial discriminators D_X and D_Y, where D_X aims to distinguish between images x and translated images $F(y)$; in the same way, D_Y aims to discriminate between y and $G(x)$. Two components are included in the CycleGAN objective: adversarial loss [7] for matching the distribution of generated images to the distribution of data in the target domain and cycle consistency losses to escape the inconsistencies between the learned mappings G and F.

CycleGAN applies adversarial losses [7] to both mapping functions. For the mapping function $G: X \rightarrow Y$ and its discriminator D_Y, we express the objective as:

$$L_{GAN}(G, D_Y, X, Y) = E_{y \sim p_{data}(y)}[logD_Y(y)] + E_{x \sim p_{data}(x)}[log(1 - D_y(G(x))] \quad (1)$$

where G is attempting to produce $G(x)$ that look identical to teeth from domain Y, while D_Y is trying to differentiate between the interpreted $G(x)$ and the actual teeth y. The goal of G is to minimize this objective against the opponent D, which is attempting to maximize it. For the mapping function $F: Y \rightarrow X$ and its discriminator D_X, CycleGAN has a similar adversarial loss.

Adversarial training will learn G and F mappings that generate outputs distributed identically as target domains Y and X, respectively. However, with enough resources, the network can map the same set of input images to any random image permutation in the target domain, where any of the qualified mappings can induce the output distribution that matches the target distribution. Thus, adversarial defeats of their own cannot guarantee that the learned function will map an individual x i input to the desired y i output. In order to further minimize the space of possible mapping

functions, it is argued that the learned mapping functions should be loop-consistent, for each image x from domain X, The conversion period in the image should take x back to the original picture, $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$. Similarly, the backward period continuity of each Y, G and F image should be reached: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. So cycle consistency loss is:

$$Lcyc(G,F) = E_{x\sim pdata(x)}[F(G(x)) - x] + E_{y\sim pdata(y)}[G(F(y)) - y]. \quad (2)$$

CycleGAN's full objective is:

$$L(G,F,D_X,D_Y,X,Y) = L_{GAN}(G,D_Y,X,Y) + L_{GAN}(F,D_X,X,Y) + \lambda L_{cyc}(G,F) \quad (3)$$

Where λ = 10 is the most optimize was proved [14].

This old method can achieve compelling results in many cases. On translation that involves common braces like metal braces with fully archwire, bracket and elastic tie, the method often succeeds. But the results are unexpected in some special conditional.

2.2.2. Failure cases

As illustrated in Figure 2, the CycleGAN model still cannot erase the cases with identical colors that coincide with the colors of teeth components such as pink (gums), white (teeth). To address this problem, we applied some image enhancing methods. For braces with thin wires and small interest regions, we extract the mouth region before doing braces removal.

Data augmentation. For each sample in the braces2teeeth dataset, we randomly created ten different augmented versions by changing the S and V color channels in the HSV space so that the teeth and braces have different tones. Although the brightly colored braces have been removed, the reconstructed image has also been affected by color changes.

Extracting mouth region. Images in which the square of braces was too small, and its structure is unclear, we crop the mouth region, put it into the model, replace the mouth region with the result. To keep the ratio of width and height of the extracted mouth region, we re-defined the cropping region by expanding the interest region towards the upper and lower lip so that the image will not be distorted when put into the model. This method will be applied mainly in Pix2Pix models.

2.2.3. Pix2Pix with preprocessing

Pix2Pix [15] is proved that better than CycleGAN in [14] about the quality of the generated image. However, preparing the dataset consisting of the pairs shown in Figure 1 is very difficult and costly. Photographs taken during the braces and after the braces are usually unrelated due to different angles or changes in the patient's condition from braces to the time the braces were removed. So we introduce the below framework to generate a paired dataset. First, we only use the teeth image $y_{j=1}^{m}$, for each y_j we detect the teeth region (inner mouth) and are filled by teeth color. After filled, y_i became y'_i which is between x_i and y_i. y'_i is originally from y_i but we will temporarily treat it as a preprocessed image from x_i. Now we have a pair $\{y'_i, y_i\}_{i=1}^{N}$ and can put it into the Pix2Pix model.

$$\{x_i, y_i\}_{i=1}^{N} = \{y'_i, y_i\}_{i=1}^{N} = \{preprocessing(y_i), y_i\}_{i=1}^{N}. \quad (3)$$

The preprocessing method includes three sub-task. The first is to detect mouth was introduced in the CycleGAN model. The second is defining the teeth color, because the generator has a high chance of producing the same teeth zone if we filled the teeth region with the same color. Finally, the sub-task is the most important. We will color the inside of the mouth with the teeth color as if we are temporarily removing braces, even though they do not have braces on the teeth. We choose this method to completely resolve the coloring braces in the previous and furthermore is removed from any type of braces.

Teeth color inpainting. We used the VITA3D Master palette as the reference to find pixels that are probably teeth region. Then we take the pixels on two perpendicular lines of the image and comparing those pixels to color on the sample palette, the color of a tooth is the closet color.

Braces removal. We extracted the convex region covering landmark points with an index of 61 - 68 on the facial landmark to define the zone of interest. We conducted three different methods for comparison:

Method A: All pixels in the interest region are inpainted by a tooth color.

Method B: Pixels in the interest region and whose teeth color distance is greater than a given threshold will be inpainted with teeth color. Before that, we convert the image to CIELab color space and use CIE2000 spacing.

Method A was simple, but the result may be wrong according to the groud truth because there is no information about teeth' structure when put into the model. Method B can keep a part of teeth, but the result may be unreasonable as describe in Figure 6.

2.2.4. GraphCut combines Generative Image Inpainting with Contextual Attention

In practice, the user wants to manipulate the output to achieve the desired texture if the generated image can not remo braces altogether. This study combines GraphCut [20] and Generative Image Inpainting with Contextual Attention [4] implement an interactive model named the Inpainting model for brevity.

Graphcut is a method of image segmentation applied graph theory. If we see braces image like graph G (E, V), Graphcut c generate a cut that divides G into two subgraphs S as braces part and T as another part. We keep the only T to make a bin mask to put into the below generative model.

Generative Image Inpainting with Contextual Attention: takes the original image and binary mask, performs Inpainting at positions marked on the binary mask and finally returns the brace-free tooth image (Figure 7. f).

The overall process followed these steps. Firstly, the user would provide the original image (Figure 7. a) and desi interactions for Graphcut. The user interactions are marks of braces location (red mark without braces and blue mark brace Secondly, the GraphCut model maps an image to a superpixel set (Figures 7. c and Figure 7. d) and uses additio information from user interactions to map it to a three-dimensional graph. Then, we use the Boykov-Kolmogorov Min-c Max-follow algorithm on the three-dimensional graph to obtain two subgraphs, S is the superpixel sets marked as brac and T is the superpixel set marked as teeth. Next, we map S and T back to the original image and get a binary mask wit value of 1 marks bracket and a value of 0 marks no bracket. The Inpainting model receives the original image and bin mask, performs Inpainting at the marked position on the binary mask and finally returns a tooth image that does not cont braces (Figure 7. f).

3. Results

In this study, we used the same evaluation datasets and metrics as Pix2Pix [15]. We compared our method against several baselines, including COGAN [21], SimGAN [22] and BiGAN / ALI [26].

Traditional metrics such as per-pixel mean-squared in many previous method errors are not compatible to evaluate the results. We also run “real vs. fake” perceptual studies on Amazon Mechanical Turk (AMT). Plausibility about color and structure of image to a human observer is often the ultimate goal. Because of the specific dataset, we can not evaluate by IS score or FCN score.

AMT perceptual studies. For our AMT experiments, we follow the same perceptual study protocol from Isola et al. [15] but have a little change. Turkers were presented with a list of pairs include one real image and one fake image generated by our model; the fake image is randomly on the right or left side. On each trial, the pair appeared for 1 - 2 seconds; after that, the Turkers responded, which side was fake with 5 - 10 seconds. No feedback was provided after each trial. Each session only tested a single algorithm, and participants were only allowed to complete a single session. Therefore, our numbers should only be used to compare our current method against the baselines (which were run under identical conditions).

Like [15], we implement all the baselines using the same architecture and details as our method for far comparison, except for CoGAN [21]. CoGAN builds on generators that produce images from a shared latent representation, incompatible with our image-to-image network. We use the public implementation of CoGAN instead.

The results are shown in Table 3.

4. Discussion

Although CycleGAN and models that apply additional preprocessing methods have handled many cases well, some exceptions braces cannot be erased altogether, as shown in Section 2.2 and Figure 2, which includes:

Porcelain braces and multi-chromatic braces: the underlying cause is still due to the dataset's distribution insufficient to represent all cases of patients wearing braces, for example, the model's inability to handle some type of braces - invisible braces or color braces strangely like purple because in the dataset there are not many such cases. We also feel a difference in results if it is possible to fully apply the supervised versus unsupervised direction. As stated, however, it is very complicated and costs a lot to prepare data pairs like images taken before and after removing braces.

Braces in the image are not clear (the area containing blurred braces): can be treated with another image interpolation model to reduce image blurring, however lack of information about tooth structure and braces can give the user an inaccurate feeling.

The braces occupy a small proportion (the area containing the brace is small compared to the entire image), causing the model to experience noise as shown in Figure 2, this situation can be overcome by segmented models that can detect the subject in the image (namely the mouth). Then extract the mouth area, import it into the model, and replace the extracted area with the processed image. However, at the borders (contiguous areas of new areas and old images) there is inconsistency because the distribution of these two images is completely different, this problem has been fixed in the Inpainting model, but this model is still too expensive and ineffective. The Inpainting model has quite low results than expected because the model is not trained enough to converge to the feasible parameters, as well as this approach also requires proper initial braces region from users. Promisingly, the model will have better performance due to the improvement of training hardware.

5. Conclusions

This paper aims to erase braces trace using GAN techniques, given an image of teeth with braces. However, the paper address a limited domain, where its application is undoubtedly helpful to people during brace treatment. There are various GAN versions and image processing techniques presented in this paper. Through experiments, we can conclude that the proposed method works well in this specific domain. However, there are some cases that braces could not be erased completely due to a lack of resources. This paper presents the first study to help braces patients with better smile pictures through image processing techniques. The promising results also show that it can be applied to real scenes as braces filter. We will leave the challenge cases as future works.

References

[1] Barnes, Connelly, et al, PatchMatch: A randomized correspondence algorithm for structural image editing, *ACM Trans. Graph* **28**:3, 2009.

[2] Efros, Alexei A., Freeman, William T, Image quilting for texture synthesis and transfer, *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* 341, 2001.

[3] Alexei A Efros and Thomas K Leung, Texture synthesis by non-parametric sampling, *Proceedings of the Seventh IEEE International Conference on computer vision* **2**:1033., 1999.

[4] Yu, Jiahui, et al, Generative image inpainting with contextual attention, *Proceedings of the IEEE conference on computer vision and pattern recognition* 5505, 2018.

[5] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa, Globally and locally consistent image completion, *ACM Transactions on Graphics (TOG)*, **36**:107, 2017.

[6] Chao Yang, Yuhang Song, Xiaofeng Liu, Qingming Tang, and C-C Jay Kuo, Image inpainting using block-wise procedural training with annealed adversarial counterpart, *arXiv preprint arXiv:1803.08943*, 2018.

[7] Goodfellow, Ian J., et al, Generative adversarial networks, *arXiv preprint arXiv:1406.2661*, 2014.

[8] Zhao, Junbo, Michael Mathieu, and Yann LeCun, Energy-based generative adversarial network, *arXiv preprint arXiv:1609.03126*, 2016.

[9] Reed, Scott, et al, Generative adversarial text to image synthesis, *International Conference on Machine Learning*. PMLR 1060, 2016.

[10] Pathak, Deepak, et al, Context encoders: Feature learning by inpainting, *Proceedings of the IEEE conference on computer vision and pattern recognition* 2536, 2016.

[11] Vicente, Sara; Kolmogorov, Vladimir; Rother, Carsten, Graph cut based image segmentation with connectivity priors, *IEEE conference on computer vision and pattern recognition* 1, 2008.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

[13] M. Arjovsky, S. Chintala, and L. Bottou, Wasserstein gan, *arXiv preprint arXiv:1701.07875*, 2017.

[14] Zhu, Jun-Yan, et al, Unpaired image-to-image translation using cycle-consistent adversarial networks, *Proceedings of the IEEE international conference on computer vision* 2223, 2017.

[15] Isola, Phillip, et al, Image-to-image translation with conditional adversarial networks, *Proceedings of the IEEE conference on computer vision and pattern recognition* 1125, 2017.

[16] Mirza, Mehdi, and Simon Osindero, Conditional generative adversarial nets, *arXiv preprint 2014. arXiv preprint arXiv:1411.1784*, 2014.

[17] Li, Zhengqin, and Jiansheng Chen, Superpixel segmentation using linear spectral clustering, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1356, 2015.

[18] Van den Bergh, Michael, et al, Seeds: Superpixels extracted via energy-driven sampling, *European conference on computer vision. Springer* 13, 2012.

[19] Achanta, Radhakrishna, et al, SLIC superpixels compared to state-of-the-art superpixel methods, *IEEE transactions on pattern analysis and machine intelligence* **34**:2274, 2012.

[20] Naik, D., and Shameem Muhammed, Fast interactive superpixel based image region generation, 2019.

[21] Liu, Ming-Yu, and Oncel Tuzel, Coupled generative adversarial networks, *arXiv preprint arXiv:1606.07536* , 2016.

[22] Shrivastava, Ashish, et al, Learning from simulated and unsupervised images through adversarial training, *Proceedings of the IEEE conference on computer vision and pattern recognition* 2107, 2017.

[23] Liu, Ming-Yu, Thomas Breuel, and Jan Kautz, Unsupervised image-to-image translation networks, *arXiv preprint arXiv:1703.00848*, 2017.

[24] Mejjati, Youssef A., et al, Unsupervised attention-guided image to image translation, *arXiv preprint arXiv:1806.02311,* 2018.

[25] Din, Nizam Ud, et al, A novel GAN-based network for unmasking of masked face, *IEEE Access* **8**:44276, 2020.

[26] Dumoulin, Vincent, et al, Adversarially learned inference, *arXiv preprint arXiv:1606.00704,* 2016.

Figures and Tables

Figure 1: With two datasets of braces and teeth, our model could transition from braces image to image without braces (braces removed and replaced with teeth). The model results are better than [14, 15] because additional methods apply only to the characteristic data types depicted in Figure 1: (left per pair) braces layer data points and (right per pair) teeth class data point.

Figure 2: Pairs of images (left: the original image, right: generated image) result from CycleGAN. Several typical failure cases and their main reason. The first row shows braces with the same color of lip and gum (left) or multi-color braces (right). The second row shows braces with thin wires. The last row shows the braces placed in a small area of images.

Figure 3: Original images (upper left corner) and other versions after data augmentation have been applied.

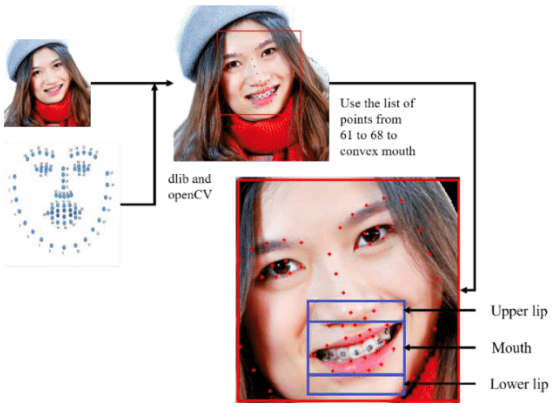


Figure 4: Extracting mouth region by facial landmarks. The mouth section is marked by Dlib landmarks with an index from 61 to 68.

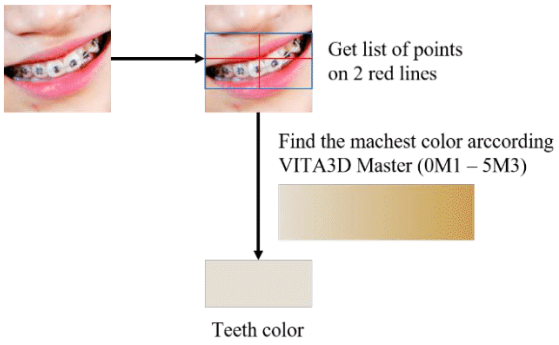


Figure 5:

Figure 6: Model produces unreasonable images

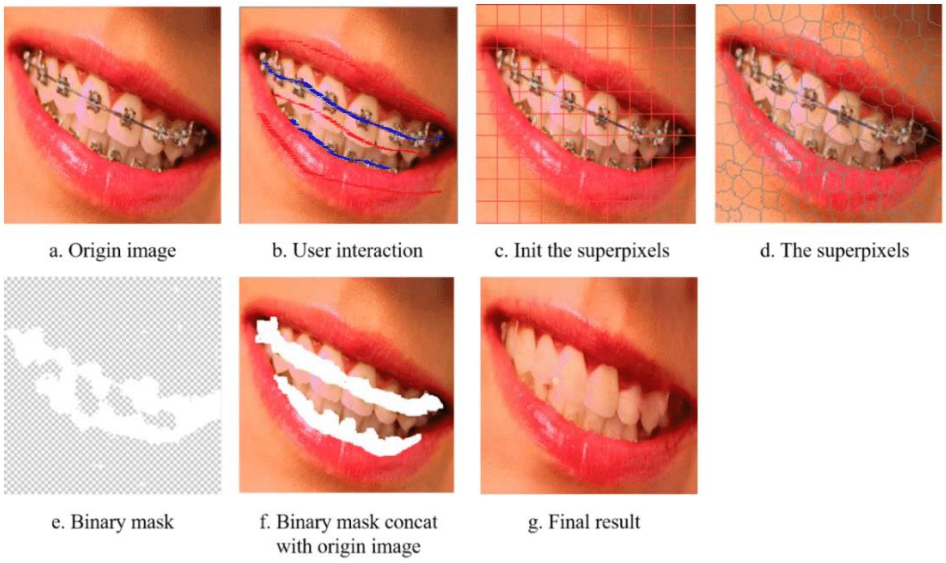


Figure 7: Results from each stage of the Inpainting model.

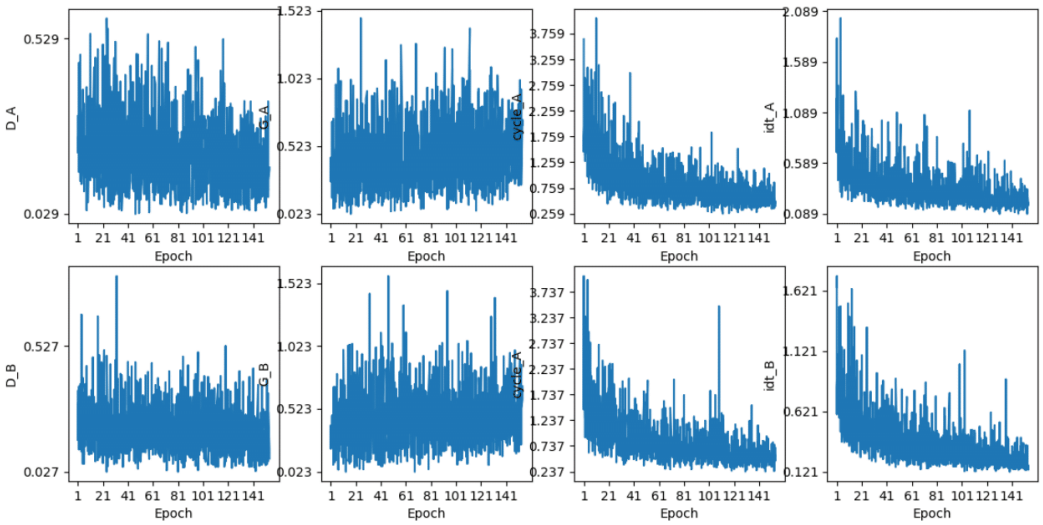


Figure 8: CycleGAN loss during training process

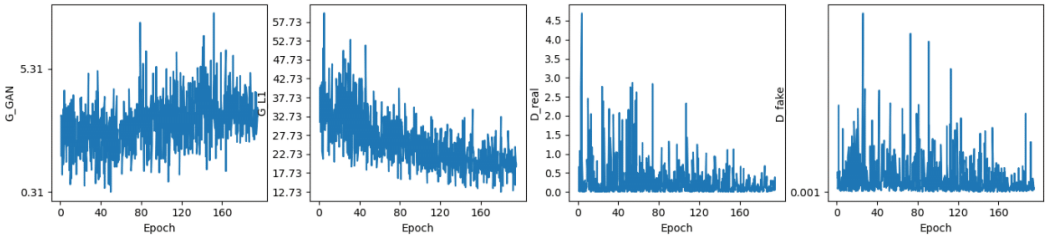


Figure 9: Pix2Pix loss during training process

Figure 10: Some test results. To the left of each pair (original photo), to the right of each pair (photo generated by model).

Figure 11: Some test results. To the left of each pair (original photo), to the right of each pair (photo generated by model).

Dataset	Total images	Details	Sources
braces2teeth	3869	testX: 437 trainX: 1728 testY: 340 trainY: 1364	Google image, dental clinics.
braces2teeth Augmentation	38690	testX: 4370 trainX: 17280 testY: 3400 trainY: 13640	braces2teeth
teeth2	7000	train: 5000 val: 1000 test: 1000	braces2teeth/teeth, StyleGAN2, dental clinics.

Table 1: Details of datasets used in our experiments.

		PatchGANs for discriminator.	2.765 M, D_Y: 2.765 M). lr: 0.0002. batch size: 1. optimizer: Adam. epoch: 200.		
Pix2Pix with preprocessing	braces2teeth Augmentation/teeth	Unet for generator. PatchGANs for discriminator.	Total weight: 57.183 M ($G_{Y'Y}$: 54.414 M, D_Y: 2.769 M). lr: 0.0002. batch size: 1. optimizer: Adam. epoch: 200.	Google Colab with GPU and Pytorch.	Under 1 second per image.

Table 2: Details of the experiment process.

Model	Training dataset	% Turker labeled real
CoGAN	braces2teeth	0.7% ± 7.5%
BiGAN/ALI	braces2teeth	2.1% ± 6.5%
SimGAN	braces2teeth	1.1% ± 7.7%
Inpainting	braces2teeth/teeth	1.7% ± 6.3%
CycleGAN	braces2teeth	**28.7% ± 2.7%**
CycleGAN + Augmented data	braces2teeth Augmented	**32.8% ± 4.1%**
Pix2Pix + preprocessing	teeth2	**36.4% ± 4.2%**

Table 3: AMT scores of our proposed models and baseline models. Although each model's training set is different, the scores were computed on the same test set.

Phụ lục 4: Danh mục các từ khóa trong dataset

Lớp	Từ khóa	Ngôn ngữ
Ảnh răng không có mắc cài	- teeth - hàm răng - 牙齿 - 歯 - зубы - tänder - δόντια - Zähne - दाँत,tanden - dientes - зуби - зубы - зуби - zęby - ធ្មេញ ...	- Anh - Việt - Trung - Nhật - Nga - Thụy điển - Hy lạp - Đức - Nepal - Na uy - Hà lan - Tây Ban Nha - Ukraina - Belarus - Serbia - Ba lan - Khơme
Ảnh niềng răng có mắc cài	- braces	- Anh

	- mắc cài - брекети - tandställning - σιδερακια ΔΟΝΤΙΩΝ - een beugel - un appareil dentaire - จัดฟัน - rovnátka - kawat gigi - брекеты - breketes - pendakap gigi - ಕಟ್ಟುಪಟ್ಟಿಗಳು - formatimin e teksteve - ब्रेसहरू - பிரேஸ்கள் - кашаалар - ধনুর্বন্ধনী	- Việt - Ukraina - Thụy điển - Hy lạp - Hà lan - Pháp - Thái - Séc - Indonesia - Belarus - Latvia - Mã lai - Kannada - Albania - Nepal - Tamil - Kyrgyz - Bangla

Phụ lục 5: Danh mục các tài nguyên liên quan

Mã nguồn:

- Client (ReactJS): https://github.com/vutuanhai237/Braces2TeethClient
- Server (Flask) & Model (Pytorch): https://github.com/vutuanhai237/Braces2TeethServer.
- Utility (Python): https://github.com/vutuanhai237/Braces2TeethUtilities
- Dataset: https://github.com/vutuanhai237/Braces2TeethDataset
- AMT Score: https://github.com/vutuanhai237/AMTScoreForBraces2Teeth