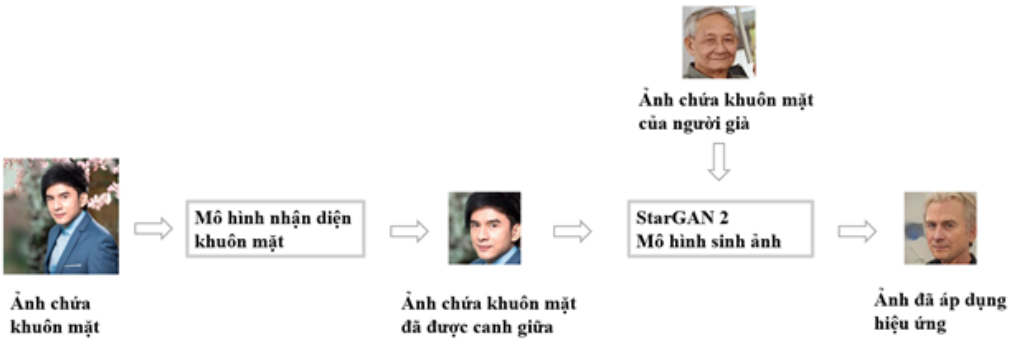
Chương 3

Giải pháp đề tài

3.1 Tổng quan giải pháp kiến trúc mô hình

Nhóm sinh viên đề xuất sử dụng mô hình StarGANv2 kết hợp cùng với mô hình nhận diện khuôn mặt dựa trên thuật toán HOG và SVM của thư viện Dlib để tạo hệ thống mô hình chỉnh sửa ảnh đầu cuối (end-to-end). Mô hình StarGANv2 được nhóm Clova AI công bố vào tháng 04/2020, có nhiều cải tiến so với phiên bản tiền nhiệm và có tiếp thu sự cải tiến từ mô hình tân tiến nhất Style GAN.

Kiến trúc tổng quan của hệ thống được minh họa cụ thể ở hình 3.1. Trong đó, nhóm sinh viên sẽ tập trung chính vào phần mô hình sinh ảnh StarGANv2.



Hình 3.1: Tổng quan kiến trúc mô hình

Tiếp theo đây, nhóm sẽ trình bày các giải pháp giải quyết vấn đề cho từng thành phần trong hệ thống chỉnh sửa ảnh, hướng xây dựng máy chủ và ứng dụng áp dụng trên nền tảng di động.

3.2 Giải pháp xây dựng mô hình nhận diện khuôn mặt

3.2.1 Giới thiệu

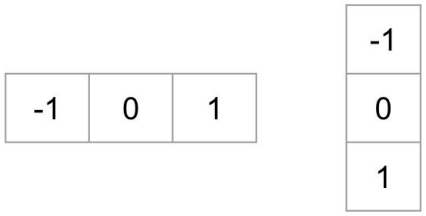
Phát hiện con người trong ảnh số là một nhiệm vụ đầy thách thức do ngoại hình có thể thay đổi và nhiều tư thế được chụp. Nhu cầu đầu tiên là một bộ đặc trưng mạnh mẽ cho phép hình dạng con người được phân biệt rõ ràng, ngay cả trong điều kiện nền hỗn tạp dưới môi trường thiếu sáng. Từ vấn đề của các bộ tính năng để phát hiện con người được đề cập trên, một nghiên cứu cho thấy rằng Biểu đồ chuẩn hóa cục bộ của bộ mô tả Hướng Gradient (HOG) cung cấp hiệu suất tuyệt vời so với các bộ đặc trưng hiện có khác bao gồm wavelets. Các bộ mô tả được đề xuất gợi nhớ đến biểu đồ hướng cạnh, bộ mô tả SIFT và ngữ cảnh hình dạng. Nhưng chúng được tính toán trên một lưới dày đặc gồm các ô có khoảng cách đều nhau và chúng sử dụng các chuẩn hóa tương phản cục bộ chồng chéo để cải thiện hiệu suất. Để đơn giản và nhanh chóng, SVM tuyến tính được sử dụng làm bộ phân loại.

Hình 3.2: Tổng quan về dây chuyền trích xuất đặc trưng và dò tìm đối tượng. (Nguồn: [0])

3.2.2 Chi tiết phần cài đặt

- Tính toán gradient hình ảnh

Gradient hình ảnh có thể được tính thông qua bộ lọc ảnh có nhân tương ứng sau:



Hình 3.3: Nhân của bộ lọc dùng để tính gradient hình ảnh.

Sau đó, chúng ta sẽ tiến hành tính độ lớn và hướng của gradient theo công thức sau, với g_x, g_y lần lượt là gradient hình ảnh theo trục O_x, O_y.

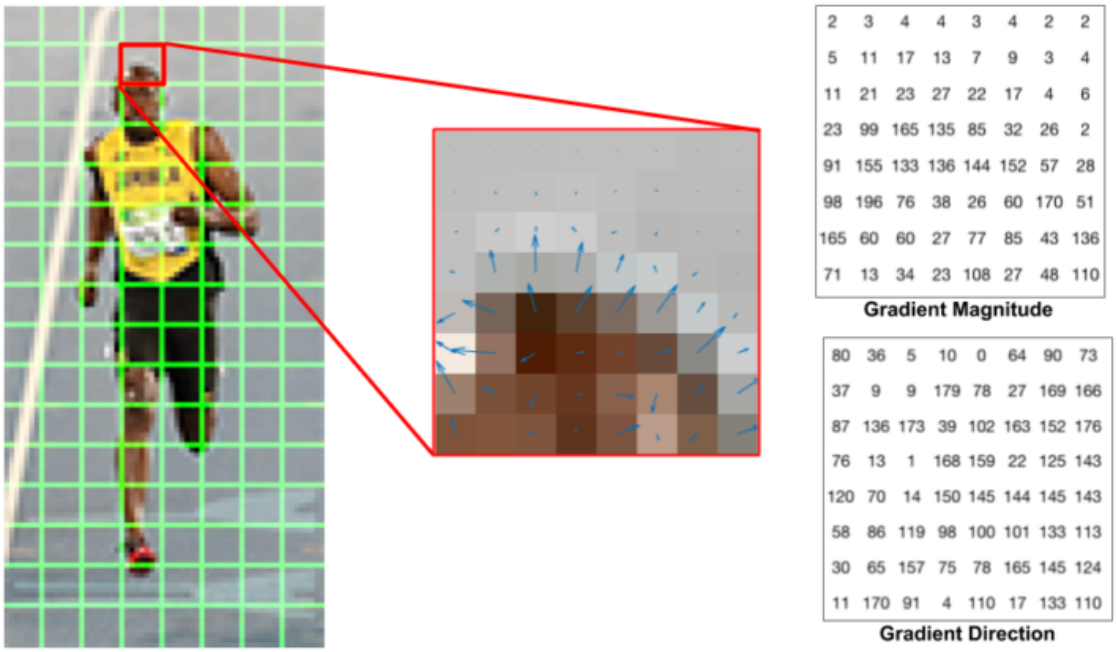
$$g = \sqrt{g_x^2 + g_y^2} \tag{3.1}$$

$$\theta = \arctan \frac{g_y}{g_x} \tag{3.2}$$

Với mỗi pixel trong ảnh, gradient mà ta tính được sẽ có 2 giá trị: độ lớn g và hướng θ. Trong trường hợp ảnh đầu vào là ảnh màu thì độ lớn của gradient ở một ô pixel sẽ là giá trị lớn nhất của độ lớn gradient trong các kênh màu. Hướng của gradient cũng sẽ là hướng tương ứng với độ lớn có giá trị lớn nhất.

- Đặt không gian/hướng vào các ngăn của biểu đồ gradient

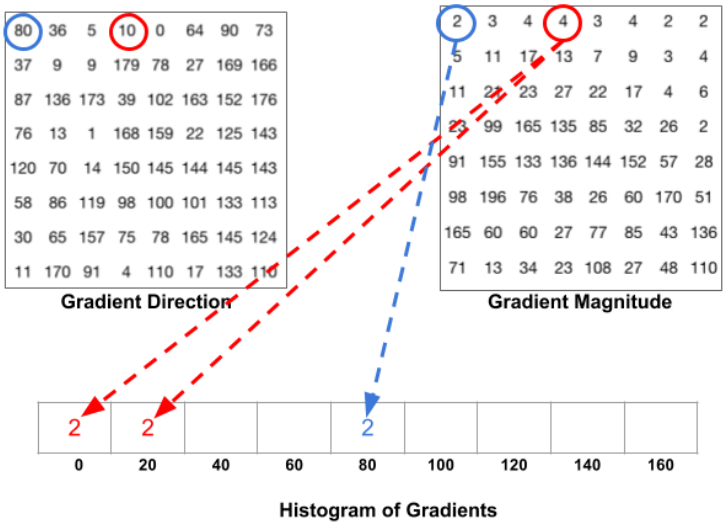
  Hình ảnh sẽ được chia thành các ô có kích thước 8 x 8 và với mỗi ô, các pixel trong ô đó sẽ được tính gradient hình ảnh. Chúng ta có tổng cộng 64 pixel trong một ô nên sẽ có 64 giá trị độ lớn của gradient và 64 giá trị hướng của gradient.



Hình 3.4: Minh họa việc tính gradient hình ảnh trong 1 ô có kích thước 8x8

Bước tiếp theo sẽ là đặt các 2 giá trị của gradient: độ lớn và hướng vào vào biểu đồ gradient gồm 9 ngăn đại diện cho các góc (hướng): 0, 20, 40 ... 160

Hình sau đây sẽ minh họa quá trình đặt các giá trị gradient vào biểu đồ gradient. Với mỗi ô pixel trong ô có kích thước 8 x 8, ta tiến hành đặt độ lớn của gradient của pixel hiện tại vào 1 trong 9 ngăn hướng phù hợp.



Hình 3.5: Đặt độ lớn của gradient hình ảnh vào một trong 9 ngăn ứng với hướng của gradient đó.

Trong hình trên, pixel được khoanh tròn màu xanh có độ lớn gradient là 2, hướng là 80 nên ta sẽ đặt giá trị 2 vào ngăn thứ 5 (đại diện cho 80°) trong biểu đồ gradient. Tương tự, với pixel được khoanh tròn màu đỏ, ta cũng sẽ giá trị 4 vào ngăn trong biểu đồ gradient. Tuy nhiên, do hướng của gradient có giá trị 10 không thuộc về bất kỳ ngăn nào trong 9 ngăn nên ta sẽ chia đôi giá trị độ lớn và đặt vào 2 ngăn kề cạnh 10°.

Toàn bộ các pixel trong ô kích thước 8x8 sẽ được tính gradient và thêm vào lần lượt một trong 9 ngăn của biểu đồ gradient.

- Chuẩn hóa và các khối bộ mô tả

  Sau khi đã có được biểu đồ gradient ở bước trên, ta sẽ tiến hành chuẩn hóa dữ liệu này. Gradient hình ảnh rất nhạy cảm với ánh sáng môi trường. Nếu như chúng ta chia toàn bộ giá trị pixel cho 2, độ lớn gradinet của các pixel cũng sẽ giảm đi một nửa, và dẫn đến biểu đồ gradient cũng thay đổi giá trị. Chính vì thế, việc chuẩn hóa các giá trị của biểu đồ gradient sẽ giúp chúng ta không bị tác động bởi các điều kiện ánh sáng khác nhau. Việc chuẩn hóa sẽ dựa trên phương

pháp chuẩn hóa L2 (còn được gọi là chuẩn hóa Euclid).

- Cửa sổ dò tìm

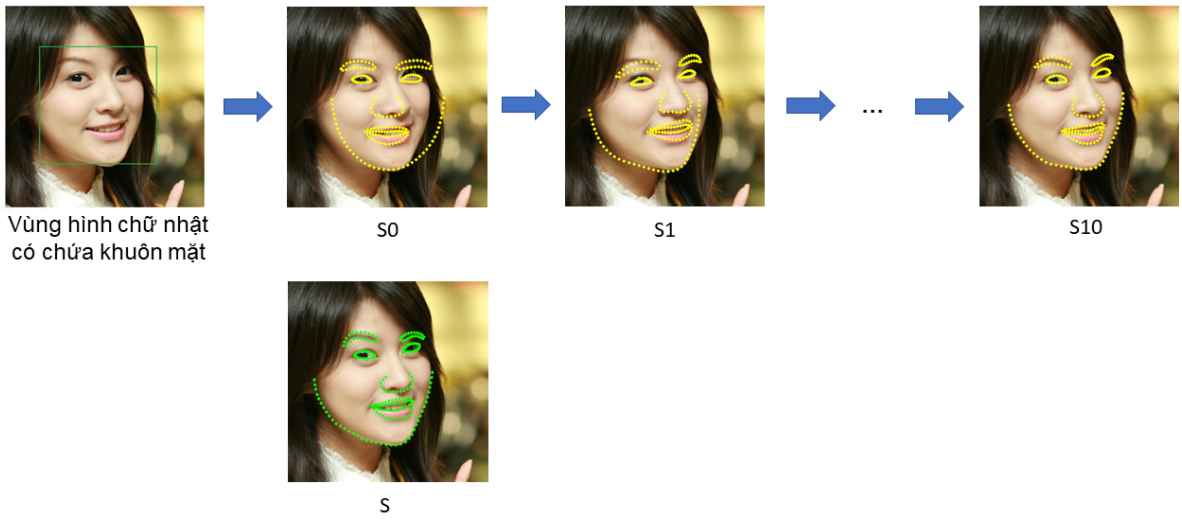
  Trong quá trình nghiên cứu, tác giả bài báo đã thử nghiệm với nhiều kích thước khác nhau cho cửa sổ dò tìm để đưa ra được con số tốt nhất. Kết quả là họ kết hợp 4 ô lại với nhau (mỗi ô có kích thước 8x8 pixel) tạo thành 1 khối (kích thước 2x2 ô). Khối này sẽ lần lượt duyệt toàn bộ bức ảnh với sải bước là 1 ô (tương ứng với 8 pixel). Với mỗi lần duyệt, bốn ô trong một khối sẽ cho ra 4 vector của 4 biểu đồ gradient có kích thước mỗi vector là 9. Bốn vector này sẽ được nối lại với nhau tạo thành 1 vector $\mathbf{v}$ có kích thước là 9 x 4 = 36. Sau đó vector $\mathbf{v}$ này sẽ được chuẩn hóa theo phương pháp đã đề cập ở mục trên. Vector đặc trưng cuối cùng sẽ được tạo thành từ các vector $\mathbf{v}$ đã được chuẩn hóa nối lại.

- Bộ phân loại

  Một mô hình SVM tuyến tính được huấn luyện bằng SVMLight được sử dụng với tham số C = 0.001

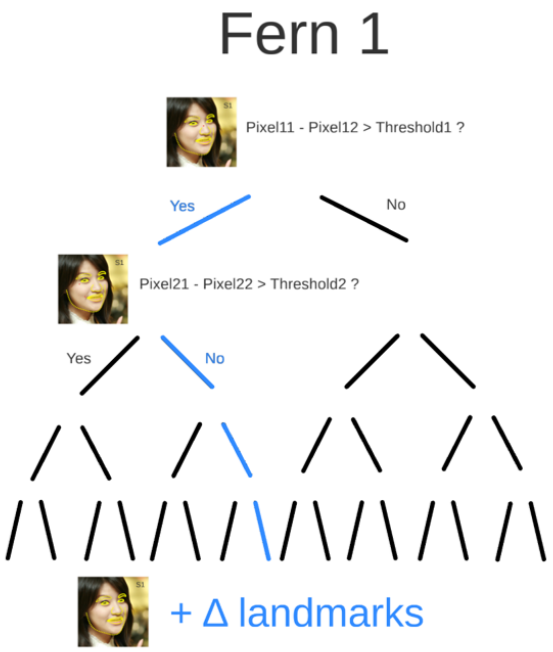
3.3 Giải pháp căn chỉnh khuôn mặt (Face Alignment)

Trong luận văn có sử dụng đến thư viện Dlib để căn chỉnh khuôn mặt sau khi đã được nhận diện từ giải pháp 3.2. Do đây là một bước phụ cần phải thực hiện trước khi có thể đưa ảnh đầu vào vào mô hình GAN nên nhóm sẽ trình bày một cách khái quát, tóm tắt giải pháp mà thư viện Dlib sử dụng.



Hình 3.6: Tổng quan các bước trong giải pháp căn chỉnh khuôn mặt

Giải pháp mà nhóm sinh viên lựa chọn trong việc căn chỉnh khuôn mặt, còn được biết đến là xác định các cột mốc trên khuôn mặt là một thuật toán có độ chính xác tương đối cao cùng với số lượng phép tính sử dụng hiệu quả dựa trên *phương pháp Gradient Boosting*. Trong hình 3.6, chúng ta sẽ nhận đầu vào là 1 tấm ảnh kèm theo tọa độ vùng hình chữ nhật có chứa khuôn mặt. Sau đó, ta sẽ tiến hành khởi tạo $S^{(0)}$ - các điểm cột mốc trên khuôn mặt ban đầu bằng cách lấy trung bình toàn bộ các điểm cột mốc trên tập dữ liệu hình được huấn luyện. So đó ta sẽ tiến hành lần lượt cập nhật $S^{(0)}$ thông qua một bộ học mạnh r_0 cho ra kết quả các cột mốc $S^{(1)}$. Và cứ thế lặp lại cho đến bộ học mạnh cuối cùng r_{10} cho ra kết quả của mô hình là $S^{(10)}$. Mỗi bộ học mạnh r_t sẽ có cụ thể 500 bộ học yếu g_k bên dưới, với mỗi bộ học yếu là một fern. Hình 3.7 sẽ minh họa cho cấu tạo của một fern. Với một bộ học yếu, ta sẽ cập nhật các cột mốc hiện tại bằng cách cộng cho $\Delta landmarks$ ở dưới mỗi nút lá của cây. Tùy vào kết quả của phép so sánh giá trị pixel ở mỗi nút mà sẽ quyết định nút lá chứa $\Delta landmarks$ nào được duyệt đến.

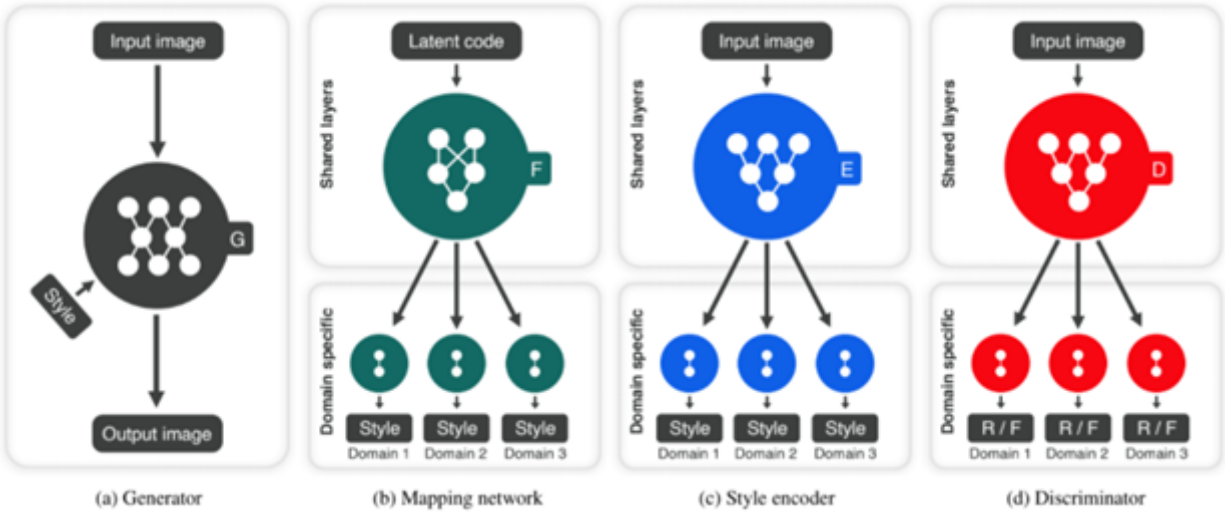


Hình 3.7: Minh họa cho fern

3.4 Giải pháp xây dựng mô hình sinh ảnh

3.4.1 Khung chương trình đề xuất

Gọi $\mathcal{X}$ và $\mathcal{Y}$ lần lượt là tập ảnh và các miền. Cho một ảnh $\mathbf{x} \in \mathcal{X}$ và một miền tùy ý $y \in \mathcal{Y}$, mục tiêu của chúng ta là huấn luyện một bộ sinh G duy nhất có thể tạo ra các ảnh đa dạng của mỗi miền y tương ứng với ảnh $\mathbf{x}$. Chúng ta tạo vectơ kiểu cụ thể cho miền trong không gian kiểu đã học của mỗi miền và huấn luyện G để phản ánh các vectơ kiểu. Hình 3.8 minh họa tổng quan về khung mô hình, bao gồm bốn mô-đun được mô tả bên dưới.



Hình 3.8: Tổng quan mô hình StarGANv2. (Nguồn [0])

Bộ sinh (Hình 3.8a). Bộ sinh G có nhiệm vụ chuyển hình ảnh đầu vào $\mathbf{x}$ thành hình ảnh đầu ra $G(\mathbf{x}, \mathbf{s})$ phản ánh mã kiểu thuộc miền cụ thể $\mathbf{s}$. Mã kiểu $\mathbf{s}$ sẽ được cung cấp bởi mạng ánh xạ F hoặc bởi bộ mã hóa kiểu E. Sau đó, chuẩn hóa phiên bản thích ứng (AdaIN) được sử dụng để truyền $\mathbf{s}$ vào G. Chúng ta có thể thấy rằng $\mathbf{s}$ được thiết kế để đại diện cho kiểu của một miền cụ thể y. Điều này loại bỏ sự cần thiết của việc cung cấp y cho bộ sinh G và cho phép G tổng hợp ảnh của tất cả các miền.

Mạng ánh xạ (Hình 3.8b). Cho một mã tiềm ẩn $\mathbf{z}$ và một miền y, mạng ánh xạ F cho ra một mã kiểu $\mathbf{s} = F_y(\mathbf{z})$, trong đó $F_y(.)$ biểu thị một đầu ra của F tương ứng với miền y. F bao gồm một MLP với nhiều nhánh đầu ra để cung cấp mã kiểu cho tất cả các miền có sẵn. F có thể tạo ra các mã kiểu đa dạng bằng cách lấy mẫu vectơ tiềm ẩn $\mathbf{z} \in \mathcal{Z}$ và miền $y \in \mathcal{Y}$ một cách ngẫu nhiên. Kiến trúc đa tác vụ cho phép F học cách biểu diễn kiểu của tất cả các miền một cách hiệu quả.

Bộ mã hóa kiểu (Hình 3.8c). Cho một hình ảnh $\mathbf{x}$ và miền tương ứng của nó là y, bộ mã hóa E có nhiệm vụ trích xuất mã kiểu $\mathbf{s} = E_y(\mathbf{x})$ của $\mathbf{x}$. Ở đây, $E_y(.)$ biểu thị đầu ra của E tương ứng với miền y. Tương tự như F, bộ mã hóa kiểu E cũng được hưởng lợi từ việc thiết lập học đa tác vụ. E có thể tạo ra các mã kiểu đa dạng sử dụng các hình ảnh tham khảo

khác nhau. Điều này cho phép G tổng hợp một hình ảnh đầu ra phản ánh kiểu dáng của một hình ảnh tham chiếu $\mathbf{x}$.

Bộ phân biệt (Hình 3.8d). Bộ phân biệt D chính là bộ phân biệt đa tác vụ. Nó bao gồm nhiều nhánh đầu ra. Mỗi nhánh Dy học cách phân loại xem ảnh $\mathbf{x}$ là ảnh thực của miền y hay ảnh giả $G(\mathbf{x}, \mathbf{s})$ do bộ sinh G tạo ra.

3.4.2 Các mục tiêu huấn luyện mô hình

Cho một hình ảnh $\mathbf{x} \in \mathcal{X}$ với miền gốc của nó là $y \in \mathcal{Y}$, khung mô hình được đào tạo bằng cách sử dụng các mục tiêu sau.

Mục tiêu đối kháng. Trong quá trình đào tạo, ta lấy mẫu một mã tiềm ẩn $\mathbf{z} \in \mathcal{Z}$ và miền đích $\tilde{y} \in \mathcal{Y}$ một cách ngẫu nhiên, và tạo mã kiểu đích $\tilde{\mathbf{s}} = F_{\tilde{y}}(\mathbf{z})$. Bộ sinh G lấy hình ảnh $\mathbf{x}$ và $\tilde{\mathbf{s}}$ làm đầu vào và học cách tạo đầu ra hình ảnh $G(\mathbf{x}, \tilde{\mathbf{s}})$ dựa trên hàm mất mát đối nghịch.

$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{x},y}[\log D_y(\mathbf{x})] + \mathbb{E}_{\mathbf{x},\tilde{y},\mathbf{z}}[\log(1 - D_{\tilde{y}}(G(\mathbf{x}, \tilde{\mathbf{s}})))] \tag{3.3}$$

trong đó $D_y(.)$ biểu thị đầu ra của D tương ứng với miền y. Mạng ánh xạ F học cách tạo các mã kiểu $\tilde{\mathbf{s}}$ có thể có trong miền đích $\tilde{y}$ và G học cách sử dụng các $\tilde{\mathbf{s}}$ và tạo ra hình ảnh $G(\mathbf{x}, \tilde{\mathbf{s}})$ không thể phân biệt được so với hình ảnh thực của miền $\tilde{y}$.

Tái tạo kiểu. Để bắt buộc hàm sinh G sử dụng mã kiểu $\tilde{\mathbf{s}}$ khi tạo hình ảnh $G(\mathbf{x}, \tilde{\mathbf{s}})$, một hàm mất mát tái tạo kiểu được sử dụng.

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{x},\tilde{y},\mathbf{z}}[\|\tilde{\mathbf{s}} - E_{\tilde{y}}(G(\mathbf{x}, \tilde{\mathbf{s}}))\|_1] \tag{3.4}$$

Mục tiêu này tương tự như các cách tiếp cận trước đây, sử dụng nhiều bộ mã hóa để học cách ánh xạ từ một hình ảnh sang mã tiềm ẩn của nó. Sự khác biệt đáng chú ý là chúng ta đào tạo một bộ mã hóa E duy nhất để khuyến khích các đầu ra đa dạng hóa cho nhiều miền. Tại thời điểm thử nghiệm, bộ mã hóa E đã được huấn luyện cho phép G biến đổi hình

ảnh đầu vào và phản ánh kiểu của hình ảnh tham chiếu.

Đa dạng hóa kiểu. Để cho phép bộ sinh G tạo ra các hình ảnh đa dạng hơn nữa, chúng ta chính quy hóa tường minh G bằng hàm mất mát đa dạng nhạy cảm.

$$\mathcal{L}_{ds} = \mathbb{E}_{\mathbf{x},\tilde{y},\mathbf{z}_1,\mathbf{z}_2}[\|G(\mathbf{x},\tilde{\mathbf{s}}_1) - G(\mathbf{x},\tilde{\mathbf{s}}_2)\|_1] \tag{3.5}$$

trong đó mã kiểu đích $\tilde{\mathbf{s}}_1$ và $\tilde{\mathbf{s}}_2$ được tạo ra bởi F có điều kiện dựa trên hai mã tiềm ẩn ngẫu nhiên $\mathbf{z}_1$ và $\mathbf{z}_2$ ($\tilde{\mathbf{s}}_i = F_{\tilde{y}}(\mathbf{z}_i), i \in \{1,2\}$). Việc tối đa hóa sự chính quy buộc G phải khám phá không gian hình ảnh và khám phá các đặc trưng kiểu có ý nghĩa để tạo ra các hình ảnh một cách đa dạng. Lưu ý rằng ở giai đoạn ban đầu, sự chênh lệch nhỏ của $\|\mathbf{z}_1 - \mathbf{z}_2\|_1$ ở mẫu số làm tăng tổn thất đáng kể. Điều này dẫn đến việc huấn luyện không ổn định do có gradient lớn. Do đó, chúng ta loại bỏ phần mẫu số và đưa ra một phương trình mới để luyện tập ổn định.

Bảo toàn đặc tính nguồn. Để đảm bảo rằng hình ảnh được tạo $G(\mathbf{x},\tilde{\mathbf{s}})$ duy trì đúng các đặc điểm bất biến thuộc miền (ví dụ: tư thế) của hình ảnh đầu vào $\mathbf{x}$, chúng ta sử dụng hàm mất mát nhất quán của chu trình

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{x},y,\tilde{y},\mathbf{z}}[\|\mathbf{x} - G(G(\mathbf{x},\tilde{\mathbf{s}}),\hat{\mathbf{s}})\|_1] \tag{3.6}$$

trong đó $\hat{\mathbf{s}} = E_y(\mathbf{x})$ là mã kiểu ước tính của ảnh đầu vào $\mathbf{x}$, và y là miền gốc của $\mathbf{x}$. Bằng cách khuyến khích bộ sinh G tái tạo lại hình ảnh đầu vào $\mathbf{x}$ với mã kiểu ước tính $\hat{\mathbf{s}}$, G học cách bảo toàn các đặc điểm ban đầu của $\mathbf{x}$ trong khi thay đổi kiểu của nó một cách trung thực.

Mục tiêu đầy đủ. Hàm mục tiêu đầy đủ được tóm tắt như sau:

$$\min_{G,F,E} \max_{D} \quad \mathcal{L}_{adv} + \lambda_{sty}\mathcal{L}_{sty} - \lambda_{ds}\mathcal{L}_{ds} + \lambda_{cyc}\mathcal{L}_{cyc} \tag{3.7}$$

trong đó λ_{sty}, λ_{ds} và λ_{cyc} là các siêu tham số cho mỗi số hạng. Một mô hình khác với cùng mục tiêu kể trên cũng được huấn luyện với sự thay đổi

nhỏ là sử dụng hình ảnh tham chiếu thay thế cho vector tiềm ẩn khi tạo mã kiểu.

3.4.3 Kiến trúc mạng nơ-ron

Ở mục này, nhóm sinh viên mô tả chi tiết kiến trúc của mô hình StarGAN v2, bao gồm bốn mô-đun sau

Bộ sinh (Generator) (Bảng 3.1): Với tập dữ liệu AFHQ, mạng này gồm bốn khối giảm mẫu (downsampling), bốn khối trung gian ở giữa, bốn khối tăng mẫu (upsample), tất cả đều sử dụng chung đơn vị tiền kích hoạt phần dư (pre-activation residual units). Nhóm sử dụng Instance normalization (IN) và Adaptive Instance Normalization (AdaIN) cho lần lượt các khối giảm mẫu và tăng mẫu tương ứng. Một mã kiểu được truyền vào toàn bộ các lớp AdaIN, cung cấp các vector tỉ lệ và vector tịnh tiến thông qua các biến đổi affine (Affine transformation) học được. Với tập dữ liệu CelebA-HQ, số lượng của các tầng giảm mẫu và tầng tăng mẫu đều được tăng từng đôi một. Toàn bộ các lối tắt (shortcut) trong khối phần dư tăng mẫu được xóa bỏ và thêm vào skip connection với adaptive wing dựa trên heatmap.

Bảng 3.1: Kiến trúc bộ sinh

Tầng	Tái lấy mẫu	Chuẩn hóa	Chiều đầu ra
Hình ảnh x	-	-	256 x 256 x 3
Conv1x1	-	-	256 x 256 x 64
ResBlk	AvgPool	IN	128 x 128 x 128
ResBlk	AvgPool	IN	64 x 64 x 256
ResBlk	AvgPool	IN	32 x 32 x 512
ResBlk	AvgPool	IN	16 x 16 x 512
ResBlk	-	IN	16 x 16 x 512
ResBlk	-	IN	16 x 16 x 512
ResBlk	-	AdaIN	16 x 16 x 512
ResBlk	-	AdaIN	16 x 16 x 512
ResBlk	Upsample	AdaIN	32 x 32 x 512
ResBlk	Upsample	AdaIN	64 x 64 x 256
ResBlk	Upsample	AdaIN	128 x 128 x 128
ResBlk	Upsample	AdaIN	256 x 256 x 64
Conv1x1	-	-	256 x 256 x 3

Mạng ánh xạ (Mapping network) (Bảng 3.2): Mạng ánh xạ bao gồm một mạng MLP với K nhánh đầu ra, với K đại diện cho số lượng miền. Bốn tầng kết nối đầy đủ (fully-connected layer) được chia sẻ với toàn bộ các miền, theo sau là bốn tầng kết nối đầy đủ riêng biệt cho mỗi miền. Chiều cho các mã tiềm ẩn, tầng ẩn (hidden layer), mã kiểu lần lượt là 16, 512, 64. Mã tiềm ẩn được lấy mẫu từ phân phối Gaussian. Pixel Normalization không được áp dụng với mã tiềm ẩn, bởi vì nó đã được quan sát rằng không cải thiện hiệu suất mô hình cho công việc trên. Chuẩn hóa đặc trưng (feature normalization) cũng đã được thử nghiệm nhưng nó cũng làm giảm hiệu suất.

Bảng 3.2: Kiến trúc mạng ánh xạ

Loại	Tầng	Kích hoạt	Chiều đầu ra
Được chia sẻ	z tiềm ẩn	-	16
Được chia sẻ	Tuyến tính	ReLU	512
Được chia sẻ	Tuyến tính	ReLU	512
Được chia sẻ	Tuyến tính	ReLU	512
Được chia sẻ	Tuyến tính	ReLU	512
Không chia sẻ	Tuyến tính	ReLU	512
Không chia sẻ	Tuyến tính	ReLU	512
Không chia sẻ	Tuyến tính	ReLU	512
Không chia sẻ	Tuyến tính	ReLU	64

Bộ mã hóa kiểu (Style encoder) (Bảng 3.3): bộ mã hóa bao gồm một mạng nơ-ron tính chập (CNN) với K nhánh đầu ra, trong đó K là số lượng các miền. Sáu khối tiền kích hoạt phần dư (pre-activation residual block) được chia sẻ với toàn bộ các miền, theo sau bởi một tầng kết nối đầy đủ (fully connected layer) cho mỗi miền. Gộp trung bình toàn cục (Global average pooling) không được sử dụng để trích xuất kiểu toàn diện (fine stye) đặc trưng cho tấm ảnh tham chiếu. Chiều đầu ra của D được gán bằng 64, chính là chiều của mã kiểu..

Bảng 3.3: Kiến trúc bộ mã hóa kiểu và bộ phân biệt. D và K đại diện cho chiều đầu ra và số lượng miền.

Tầng	Tái lấy mẫu	Chuẩn hóa	Chiều đầu ra
Hình ảnh x	-	-	256 x 256 x 3
Conv1x1	-	-	256 x 256 x 64
ResBlk	AvgPool	-	128 x 128 x 128
ResBlk	AvgPool	-	64 x 64 x 256
ResBlk	AvgPool	-	32 x 32 x 512
ResBlk	AvgPool	-	16 x 16 x 512
ResBlk	AvgPool	-	8 x 8 x 512
ResBlk	AvgPool	-	4 x 4 x 512
LReLU	-	-	4 x 4 x 512
Conv4x4	-	-	1 x 1 x 512
LReLU	-	-	1 x 1 x 512
Reshape	-	-	512
Linear * K	-	-	D * K

Bộ phân biệt (Discriminator) (Bảng 3.3). Bộ phân biệt đề xuất là một bộ phân biệt đa tác vụ, chứa nhiều nhánh đầu ra tuyến tính. Bộ phân biệt gồm sáu khối tiền kích hoạt phần dư với đơn vị tuyến tính chỉnh lưu rò rỉ (Leaky ReLU). K tầng kết nối đầy đủ được sử dụng cho sự phân loại thật/giả của mỗi miền, trong đó K chính là số lượng các miền. Chiều của đầu ra "D" được gán bằng 1 cho sự phân loại thật/giả. Các kỹ thuật chuẩn hóa đặc trưng hoặc PatchGAN đều không được sử dụng vì tất cả đã được quan sát rằng không cải thiện chất lượng đầu ra. Theo như quan sát được trong phần cài đặt, bộ phân biệt đa tác vụ cho kết quả tốt hơn những dạng khác của bộ phân biệt có điều kiện.

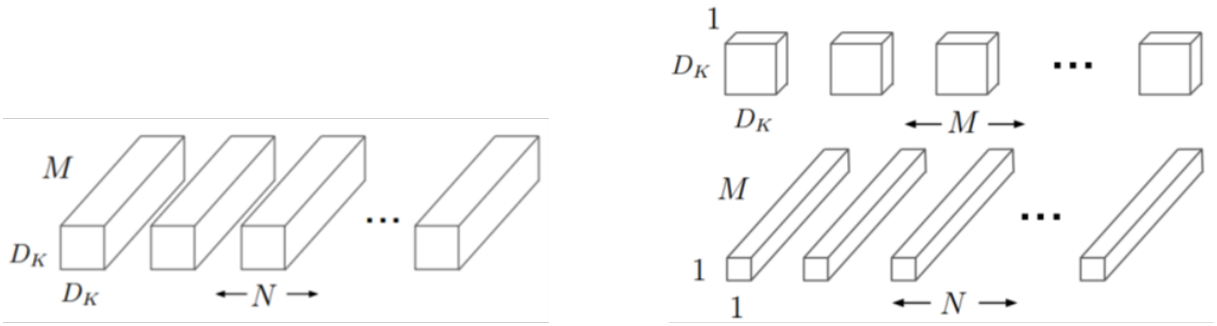
3.5 Giải pháp xây dựng mô hình phân loại giới tính

Bài toán phân loại hình ảnh không phải là bài toán quá mới mẻ hiện nay. Các bài toán như nhận dạng vật thể và phân loại hình ảnh đều có

điểm chung giữa các giải pháp là đều dựa trên các mạng nơ-ron tích chập. Tuy nhiên, các mô hình tân tiến nhất giúp giải quyết các bài toán này đa phần có kiến trúc rất lớn, với số lượng tham số lên đến hàng triệu. Điều này dẫn đến khó triển khai lên các thiết bị có phần cứng hạn chế như thiết bị nhúng, thiết bị di động. Do đó, nhóm nghiên cứu B-IT-BOTS đã đề xuất mô hình CNN có thể chạy theo thời gian thực trên các thiết bị có cấu hình thấp.

Mô hình được nhóm tác giả tạo với mục đích cân bằng giữa 2 yếu tố độ chính xác và số lượng tham số. Để có thể giảm số lượng tham số của mô hình, nhóm tác giả đã bỏ hoàn toàn lớp kết nối đầy đủ và thay vào đó là lớp tích chập và lớp gộp trung bình toàn cục (Global Average Pooling). Sau đó, nhóm tác giả tiếp tục sử dụng tích chập có thể chia tách theo chiều sâu (depth-wise separable convolution) để có thể tiếp tục giảm số lượng tham số và tăng tốc độ tính toán.

- Tích chập có thể phân tách theo chiều sâu

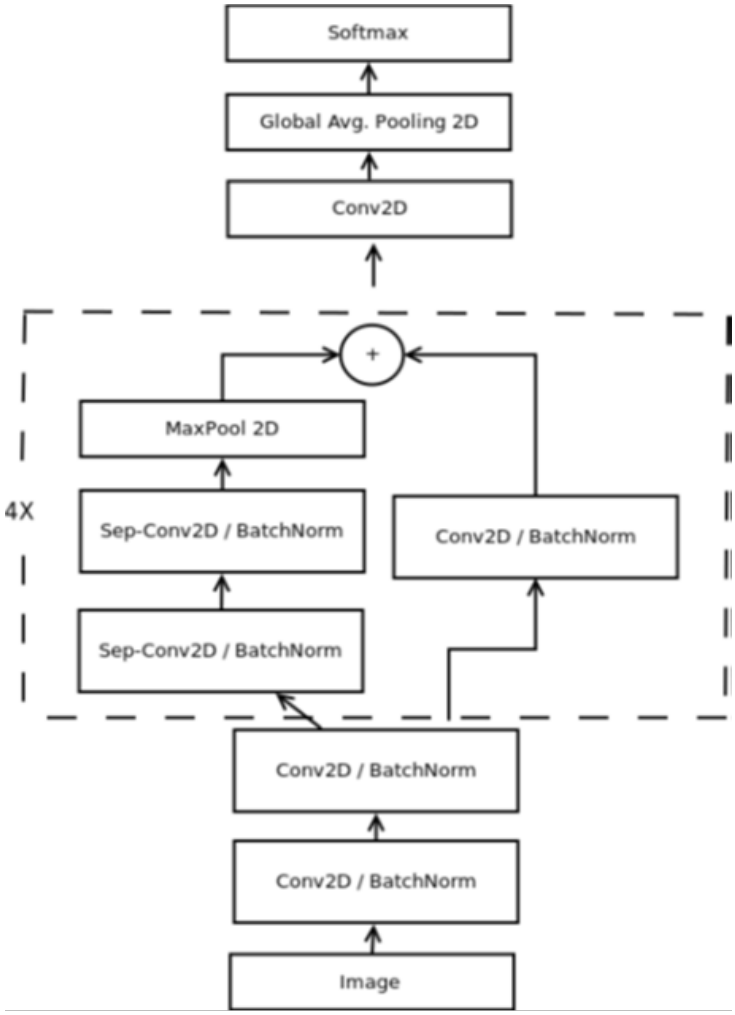


(a) (b)

Hình 3.9: Sự khác nhau giữa tích chập chuẩn (a) và tích chập có thể chia tách theo chiều sâu (b)

Tích chập có thể phân tách theo chiều sâu bao gồm hai lớp khác nhau: phần tích chập theo chiều sâu và phần tích chập theo điểm. Mục đích chính của các lớp này là tách các mối tương quan chéo không gian khỏi mối tương quan chéo kênh. Điều này được thực hiện bằng cách áp dụng bộ lọc $D \times D$ trước tiên trên mỗi M kênh đầu

vào và sau đó áp dụng N bộ lọc tích chập $1 \times 1 \times M$ để kết hợp M kênh đầu vào thành N kênh đầu ra. Việc áp dụng $1 \times 1 \times M$ tích chập giúp kết hợp từng giá trị trong ánh xạ đặc trưng mà không cần xem xét mối quan hệ không gian của chúng trong kênh.



Hình 3.10: Kiến trúc mô hình phân loại giới tính. (Nguồn ?)

Kiến trúc sau cùng được đề xuất là một mạng nơ-ron tích chập chứa 4 bộ tích chập có thể chia tách theo chiều sâu dư. Với mỗi phép tích chập, theo sau là chuẩn hóa batch và hàm kích hoạt ReLU. Tầng cuối cùng áp dụng gộp trung bình toàn cục và hàm kích hoạt soft-max để thực hiện phân loại giới tính. Mô hình này chứa khoảng 60.000 tham số. Nó được

huấn luyện trên tập dữ liệu 460,723 ảnh RGB kèm theo nhãn cho 2 lớp giới tính nam, nữ. Độ chính xác của mô hình đạt 95% trên tập dữ liệu IMDB.

3.6 Giải pháp xây dựng máy chủ

Trong phạm vi luận văn, nhóm sinh viên tiến hành xây dựng máy chủ với mục đích: cung cấp API của mô hình StarGANv2, cho phép nhận ảnh đầu vào là khuôn mặt của một người và id của hiệu ứng, trả về kết quả là khuôn mặt với hiệu ứng có id đã được áp dụng.

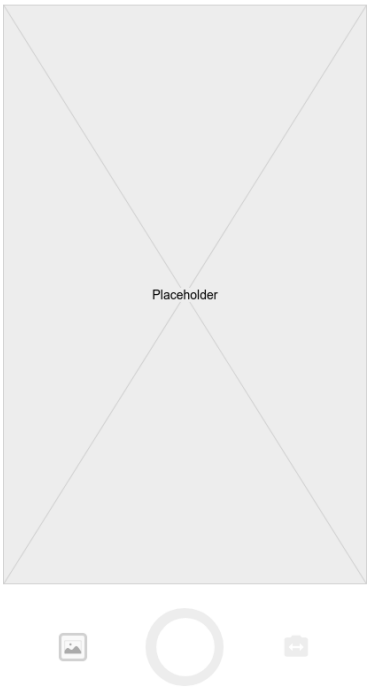
3.7 Giải pháp xây dựng ứng dụng

ứng dụng Android sẽ minh họa việc sử dụng API của mô hình. Ứng dụng có các chức năng chính là cho phép chụp hoặc chọn hình có sẵn sau đó chọn một trong các hiệu ứng làm già khuôn mặt, hoán đổi giới tính, mọc râu,... và cuối cùng hiện thị ảnh kết quả. Đồng thời, ứng dụng cũng cho phép người dùng lưu ảnh kết quả vào thư viện hoặc chia sẻ với bạn bè trên các mạng xã hội.

3.7.1 Thiết kế giao diện

Giao diện ứng dụng gồm một số màn hình chính sau:

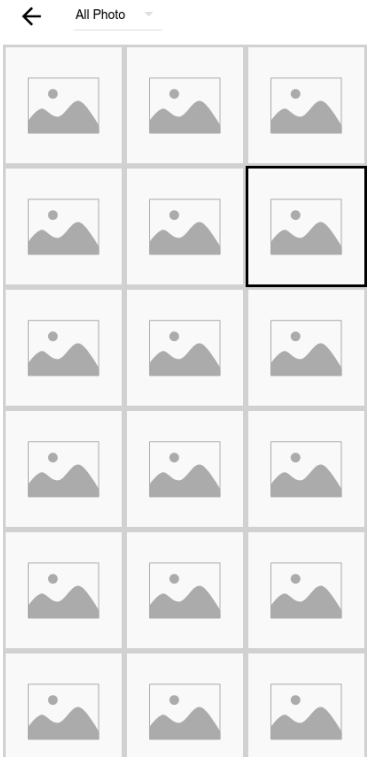
- Giao diện màn hình chụp ảnh



Hình 3.11: Màn hình chụp ảnh

Ở tại màn hình này, người dùng có thể chụp ảnh, chuyển đổi giữa cam trước và cam sau trên điện thoại hoặc đi đến thư viện ảnh để chọn ảnh có sẵn trên máy. Phần màn hình chụp ảnh sẽ được thiết kế toàn màn hình nhằm giúp đem lại trải nghiệm chụp ảnh tốt nhất cho người dùng.

- Giao diện màn hình thư viện ảnh



Hình 3.12: Màn hình thư viện ảnh

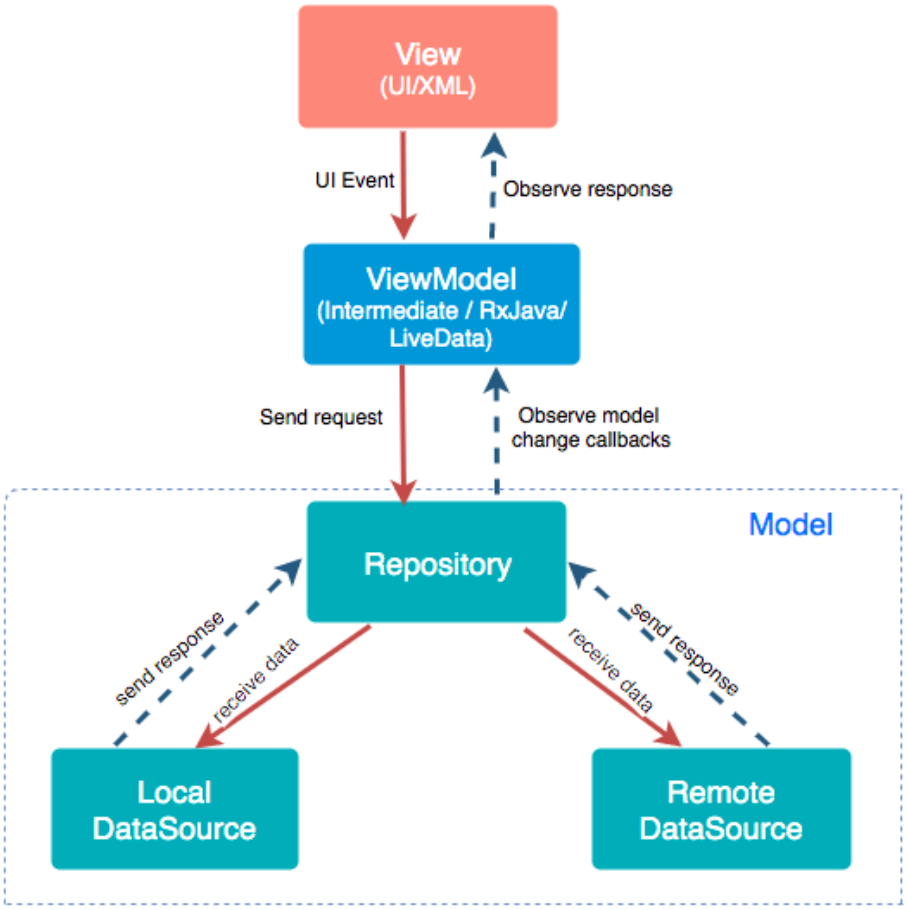
Tại đây, người dùng có thể chọn hình ảnh mà mình mong muốn từ thư viện chính. Bên cạnh đó, người dùng cũng có thế lọc các ảnh trong cùng thư viện để giúp nhanh chóng tìm được bức ảnh mong muốn thông qua nút dropbox ở phía bên trái trên cùng.

- Giao diện màn hình các hiệu ứng

Hình 3.13: Màn hình các hiệu ứng

Đây là màn hình hiển thị danh sách các hiệu ứng. Các hiệu ứng trong danh sách này đã được lọc theo giới tính của khuôn mặt trong bức ảnh được chọn/chụp.

3.7.2 Thiết kế kiến trúc



Hình 3.14: Kiến trúc MVVM trên Android

Có thể thấy rằng, với phần thiết kế giao diện trên, ứng dụng chụp ảnh giải trí cần xây dựng có kiến trúc rất đơn giản, không có gì phức tạp chuyên sâu. Bên cạnh đó, ở thời điểm hiện tại, kiến trúc thông dụng mà được Google đề xuất khi phát triển ứng dụng Android là kiến trúc Model-View-ViewModel (MVVM). Chính vì thế, nhóm sinh viên quyết định chọn kiến trúc MVVM như hình 3.14.