TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ISO 9001 : 2008

ĐỒ ÁN HỌC PHẦN

Phần Mềm Đếm Số Ngón Tay: từ 5-9

GIẢNG VIÊN HƯỚNG DẪN: ThS. Tôn Quang Toại

SINH VIÊN THỰC HIỆN: Trà Nguyễn Gia Khánh – 21DH110837

Nguyễn Hoàng Gia Huy – 21DH110660

Nguyễn Hoàng Khương – 21DH110915

Phạm Gia Bảo – 21DH110160

TP. HỒ CHÍ MINH – 03/2024

PHIẾU CHẤM ĐIỂM

Họ tên sinh viên 1 (SV1):_____ Mã SV: _____

Họ tên sinh viên 2 (SV2):_____ Mã SV: _____

Họ tên sinh viên 3 (SV3):_____ Mã SV: _____

Họ tên sinh viên 4 (SV4):_____ Mã SV: _____

CLO	Nội dung/Chuẩn đầu ra	ĐIỂM CỦA SV 1	ĐIỂM CỦA SV 2	ĐIỂM CỦA SV 3	ĐIỂM CỦA SV 4
1	Xây dựng và huấn luyện mạng neuron (Bài toán phân lớp ảnh, huấn luyện mô hình phân lớp…)				
2	Tinh chỉnh mô hình và thuật toán huấn luyện (Bài toán phân đoạn, tìm gốc quay, phát hiện vị trí, …)				
3	Vận dụng mạng neuron tích chập và mạng hồi quy (Triển khai mô hình trên web, desktop, mobile, …)				
4	Có khả năng giải quyết một số vấn đề thực tế (Thu thập dữ liệu, xử lý dữ liệu, …)				
5	Có năng lực trình bày giải pháp kỹ thuật (Thuyết trình, trình bày báo cáo, …)				
Tổng					

Họ tên GV 1: _____ Ký tên: _____

Họ tên GV 2: _____ Ký tên: _____

TÓM TẮT ĐỒ ÁN

Bài toán xây dựng phần mềm đếm số ngón tay: từ 5 - 9 (Deep learning)

- a. Thu thập dữ liệu
- b. Phân lớp các đối tượng trên
- c. Phân đoạn (segmentation) đối tượng trong ảnh
- d. Triển khai mô hình

Phương pháp đề xuất: Sử dụng 4 mô hình CNN, MLP, VGG16, ResNet50 tiến hành phân lớp hình ảnh và sử dụng mô hình U-NET để phân đoạn đối tượng trong ảnh.

Kết quả: Dự đoán được số ngón tay khi đưa hình ảnh vào và phân đoạn được hình ảnh của đối tượng ở đây là bàn tay.

MỤC LỤC

MỤC LỤC HÌNH ẢNH

Chương 1. Giới thiệu bài toán

1.1 Câu hỏi nghiên cứu

Bài toán này nhằm xây dựng một phần mềm có khả năng nhận diện và đếm số lượng ngón tay có trong các hình ảnh chứa bàn tay, với số lượng ngón từ 5 đến 9. Mục tiêu là tạo ra một hệ thống tự động có khả năng phân loại và đếm số ngón tay chính xác từ các hình ảnh đầu vào.

Trong đó input là các hình ảnh chứa các bàn tay với số ngón từ 5 đến 9 khoảng 5352 hình ảnh được chụp bằng webcam. Những hình ảnh này được chia thành các thư mục tương ứng với số ngón tay có trong hình ảnh và tập dữ liệu được chia thành ba phần: 80% cho train, 10% cho test và 10% cho validation.

Output của bài toán là đối với mỗi hình ảnh đầu vào, mô hình sẽ dự đoán số lượng ngón tay có trong hình ảnh từ 5 đến 9.

Ví dụ minh họa:

Ví dụ 1:

Input: Hình ảnh của một bàn tay có 7 ngón.

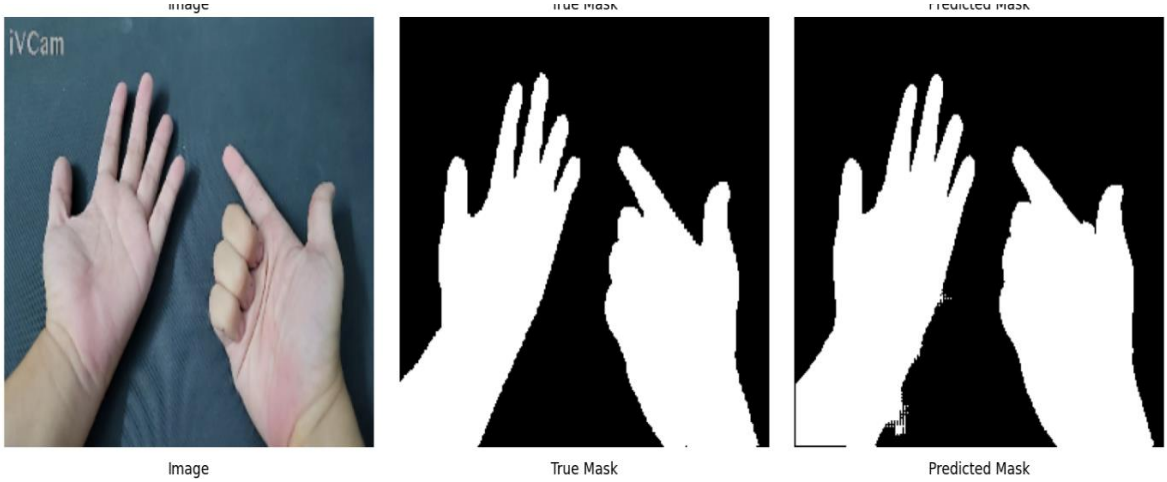
Output: Mô hình dự đoán đây là một hình ảnh của một bàn tay có 7 ngón

Hình 1: Ví dụ phân lớp

Ví dụ 2:

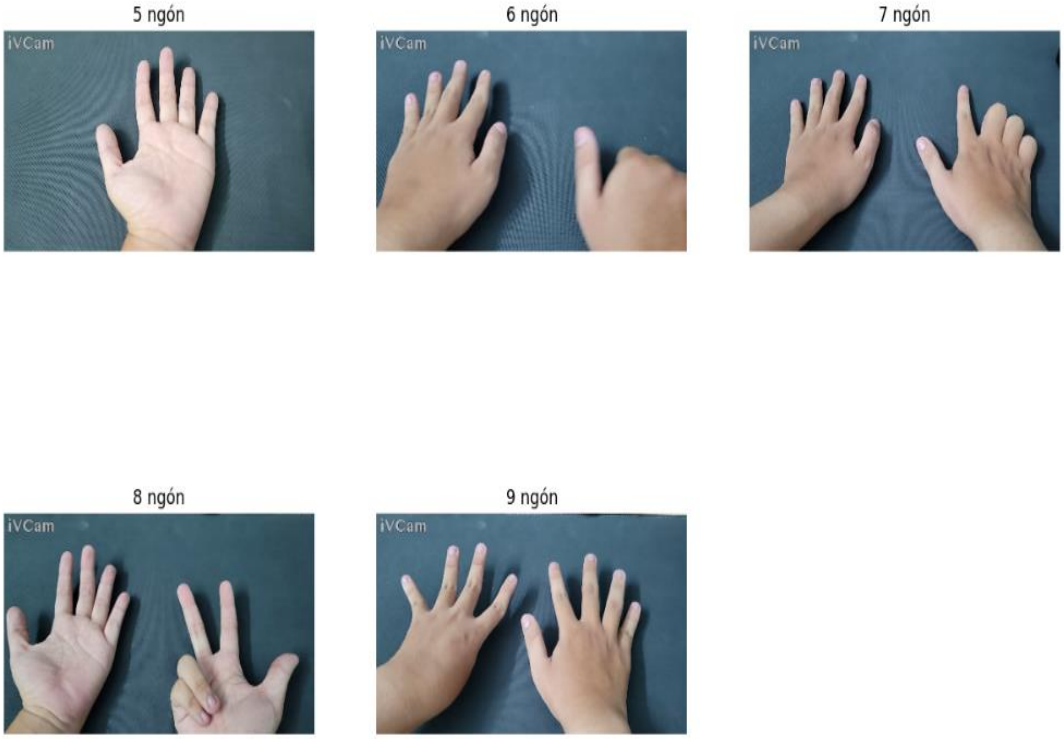
Input: Hình ảnh của một bàn tay có 7 ngón.

Output: Mô hình dự đoán hình ảnh phân đoạn của bàn tay đó

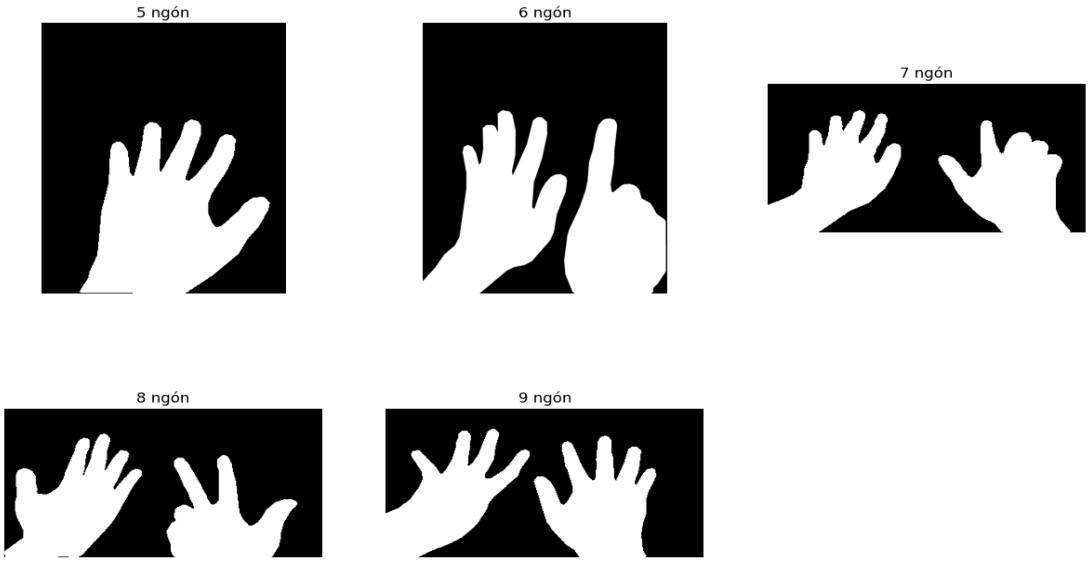


Hình 2: Ví dụ phân đoạn

Giới thiệu Dataset được sử dụng để giải quyết bài toán:



Hình 3: Ảnh gốc



Hình 4: Ảnh Mask

Dataset này bao gồm 5352 hình ảnh chứa các bàn tay với số ngón từ 5 đến 9 và có số lượng ảnh mask tương tự với ảnh gốc.

Dữ liệu đã được chia thành các thư mục tương ứng với số ngón tay có trong hình ảnh.

Tập dữ liệu được chia thành ba phần: 80% cho huấn luyện, 10% cho kiểm tra và 10% cho validation và cũng chia tương tự so với ảnh mask với các ảnh có cùng tên và label

1.2 Giới hạn nghiên cứu

Dữ liệu chưa đa dạng và hạn chế về góc độ và nền:

- Dữ liệu trong tập dataset chưa đủ đa dạng về góc độ chụp và nền phông, có thể dẫn đến hiệu suất của mô hình không cao khi áp dụng vào các tình huống thực tế.
- Hạn chế về số lượng người tham gia chụp hình có thể làm giảm tính đại diện của dữ liệu và làm giảm khả năng tổng quát hóa của mô hình.

Chỉ dự đoán được số ngón từ 5 đến 9:

- Mô hình chỉ được huấn luyện và dự đoán số lượng ngón từ 5 đến 9, không thể áp dụng cho trường hợp số ngón ít hơn hoặc nhiều hơn.
- Không có khả năng phân biệt giữa tay trái và tay phải trong hình ảnh.

Hạn chế về phạm vi áp dụng:

- Mô hình được thiết kế để áp dụng cho ảnh chứa bàn tay với số ngón từ 5 đến 9 và không thể mở rộng để áp dụng cho các tình huống khác như các hình dạng bàn tay khác, hoặc số ngón tay ngoài phạm vi từ 5 đến 9.

Ràng buộc về phần cứng và tài nguyên:

- Hiệu suất của mô hình có thể phụ thuộc vào tài nguyên phần cứng và thời gian huấn luyện, đặc biệt là khi sử dụng các mạng nơ-ron sâu và phức tạp.

1.3 Bố cục đồ án

Báo cáo này được chia thành ba chương chính, bao gồm một phần tóm tắt cùng với kết luận cuối cùng.

❖ **Chương 1: Giới thiệu bài toán**

➢ Câu hỏi nghiên cứu

♦ Trong đồ án này, nhóm em tập trung giải quyết một câu hỏi đó là: "Làm thế nào để phát triển một phần mềm có khả năng đếm số ngón tay trong một ảnh từ 5 đến 9 sử dụng công nghệ học sâu? và phân lớp và phân đoạn các đối tượng trong ảnh" Để giải quyết câu hỏi này, nhóm em sẽ khám phá các phương pháp trong môn học sâu để thực hiện công việc phân đoạn ảnh và phân lớp đối tượng, đồng thời xây dựng và huấn luyện một mô hình mạng neuron tích chập (CNN),(VGG16),(MLP),(RESNET50) cho phần phân lớp các đối tượng và xây dựng mô hình mạng U-NET cho phần phân đoạn các đối tượng trong ảnh để thực hiện nhiệm vụ này một cách chính xác.

➢ Giới hạn nghiên cứu

♦ Đồ án này tập trung vào việc phát triển một hệ thống có thể đếm số ngón tay từ 5 đến 9 trong các ảnh chụp. Chúng em giới hạn phạm vi của đồ án này trong các điều kiện ảnh chụp tiêu chuẩn, không bao gồm các trường hợp ảnh bị nhiễu, mờ hoặc có độ phân giải thấp. Ngoài ra, đồ án không xem xét việc phát hiện hoặc đếm số ngón tay trong video hoặc trong điều kiện ánh sáng yếu và chưa xây dựng được demo cho

các mô hình mà chỉ dừng ở mức triển khai và thử nghiệm với tập dữ liệu test.

- Do các ảnh của tụi em cũng chỉ sử dụng webcam và số lượng tay cũng chỉ từ 1-3 người và các góc độ chụp vào cũng không đa dạng nên đây là phần giới hạn về tập dữ liệu. Tập dữ liệu sẽ thiếu tính đa dạng.

- Bố cục đồ án
  - Chương 1: Giới thiệu bài toán - Phần này giới thiệu bối cảnh, giới thiệu tập dữ liệu và đặt ra câu hỏi nghiên cứu.
  - Chương 2: Giải pháp đề xuất

Chương 2: Giải pháp đề xuất

- 2.1 Phân tích dữ liệu
  - Trong nghiên cứu này, chúng em đã tiến hành phân tích dữ liệu tỉ mỉ để hiểu rõ các đặc điểm và tính chất của dữ liệu ngón tay. Quá trình thu thập dữ liệu bao gồm việc lựa chọn các ảnh ngón tay từ 5 đến 9 từ webcam và một vài ảnh là chụp từ điện thoại, kích thước và góc độ đa số là giống nhau. Dữ liệu sau đó được tiền xử lý, bao gồm việc chuyển đổi kích thước, chuẩn hóa và áp dụng các kỹ thuật tăng cường dữ liệu để cải thiện khả năng tổng quát hóa của mô hình.
- 2.2 Mô hình
  - Mô hình được đề xuất trong đồ án này là một hệ thống phức hợp, tận dụng sức mạnh của mạng neuron tích chập sâu (CNN) cùng với kiến trúc nổi tiếng như VGG16, ResNet50 và U-Net để giải quyết bài toán đếm số ngón tay và phân đoạn ảnh. Chúng em áp dụng một cách tiếp cận đa mô-đun, nơi mỗi kiến trúc mạng đóng góp một chức năng riêng biệt như sau:
  - VGG16 và ResNet50 (Transfer Learning): Chúng em sử dụng cả hai mô hình này như là phần của chiến lược transfer learning để tận dụng kiến thức đã học từ tập dữ liệu lớn (ImageNet) và áp dụng nó vào nhiệm vụ phân lớp số ngón tay của chúng tôi. Điều này giúp mô hình

nhanh chóng thích nghi và đạt hiệu suất cao hơn mà không cần huấn luyện từ đầu.

- MLP và CNN cơ bản: Đây là những mô hình được xây dựng và huấn luyện từ đầu, phục vụ mục đích so sánh hiệu suất với các mô hình transfer learning. MLP đại diện cho một mạng nơ-ron truyền thẳng, trong khi CNN cơ bản thể hiện sức mạnh của việc trích xuất đặc trưng trong không gian hai chiều.
- U-Net (Phân đoạn ảnh): Đối với nhiệm vụ phân đoạn ảnh, chúng tôi lựa chọn mô hình U-Net vì kiến trúc đặc biệt phù hợp với loại bài toán phân đoạn y tế, nơi mà sự chính xác về không gian là cực kỳ quan trọng. U-Net được thiết kế để làm việc tốt ngay cả với số lượng dữ liệu giới hạn, và khả năng học các đặc trưng ở mức độ chi tiết cao làm cho nó trở thành lựa chọn lý tưởng cho việc phân đoạn các ngón tay trong ảnh.
- Toàn bộ hệ thống được huấn luyện và tối ưu hóa sử dụng TensorFlow và Keras trên Google Colab, tận dụng lợi thế của việc tính toán trên GPU để tăng tốc quá trình huấn luyện. Chúng em cũng áp dụng các phương pháp như tiền xử lý dữ liệu, tăng cường ảnh và chia tập dữ liệu thành các phần huấn luyện và validation để đảm bảo mô hình có khả năng tổng quát hóa tốt trên dữ liệu chưa thấy.

❖ **Chương 3: Thực nghiệm**

- Chương này chi tiết quá trình thực nghiệm, bao gồm cách thức thiết lập thí nghiệm, huấn luyện mô hình, và đánh giá hiệu suất qua các chỉ số quan trọng như độ chính xác và loss function. Kết quả thực nghiệm cho thấy mô hình có khả năng đếm số ngón tay với độ chính xác cao, thể hiện qua các trường hợp thử nghiệm đa dạng. Phân tích kết quả cũng nêu bật những điểm mạnh cũng như hạn chế của mô hình, từ đó rút ra các bài học quan trọng.

❖ **Kết luận**

- Phần này tóm tắt những phát hiện chính và đóng góp của đồ án tới lĩnh vực thị giác máy tính và học sâu. Kết quả của đồ án không chỉ chứng minh khả năng ứng dụng của học sâu trong việc giải quyết các bài toán thực tiễn mà còn mở ra hướng phát triển mới cho việc nghiên cứu và cải thiện các mô hình tương tự. Cuối cùng, chúng em đề xuất một số hướng nghiên cứu và phát triển tương lai dựa trên nền tảng đã xây dựng, với hy vọng đóng góp vào sự tiến bộ của ngành công nghệ thông tin và thị giác máy tính.

Chương 2. Giải pháp đề xuất

2.1 Phân tích dữ liệu

Số lượng mẫu:

- Có tổng cộng 1028 hình ảnh chứa bàn tay có 5 ngón.
- Có tổng cộng 1087 hình ảnh chứa bàn tay có 6 ngón.
- Có tổng cộng 1083 hình ảnh chứa bàn tay có 7 ngón.
- Có tổng cộng 1094 hình ảnh chứa bàn tay có 8 ngón.
- Có tổng cộng 1065 hình ảnh chứa bàn tay có 9 ngón.
- Được chia thành 3 tập train,val,test theo tỷ lệ (8,1,1) và folder mask ảnh theo 3 tập train,val,test có số ảnh và tỷ lệ tương đương với folder gốc

Dung lượng tổng:

- Tổng dung lượng của tập dataset là 925 MB.

Kích thước của ảnh:

- Đa số kích thước của các hình ảnh trong dataset là 384x216.
- Các hình ảnh được chụp bằng webcam, điều này có thể ảnh hưởng đến chất lượng và độ phân giải của hình ảnh.

Nội dung của hình ảnh:

- Hình ảnh chứa bàn tay với số lượng ngón từ 5 đến 9.
- Các hình ảnh có vài hình có độ phân giải và chất lượng khác nhau, có thể cần được chuẩn hóa trước khi sử dụng cho việc huấn luyện mô hình.

Nhận xét:

Phân bố số lượng mẫu:

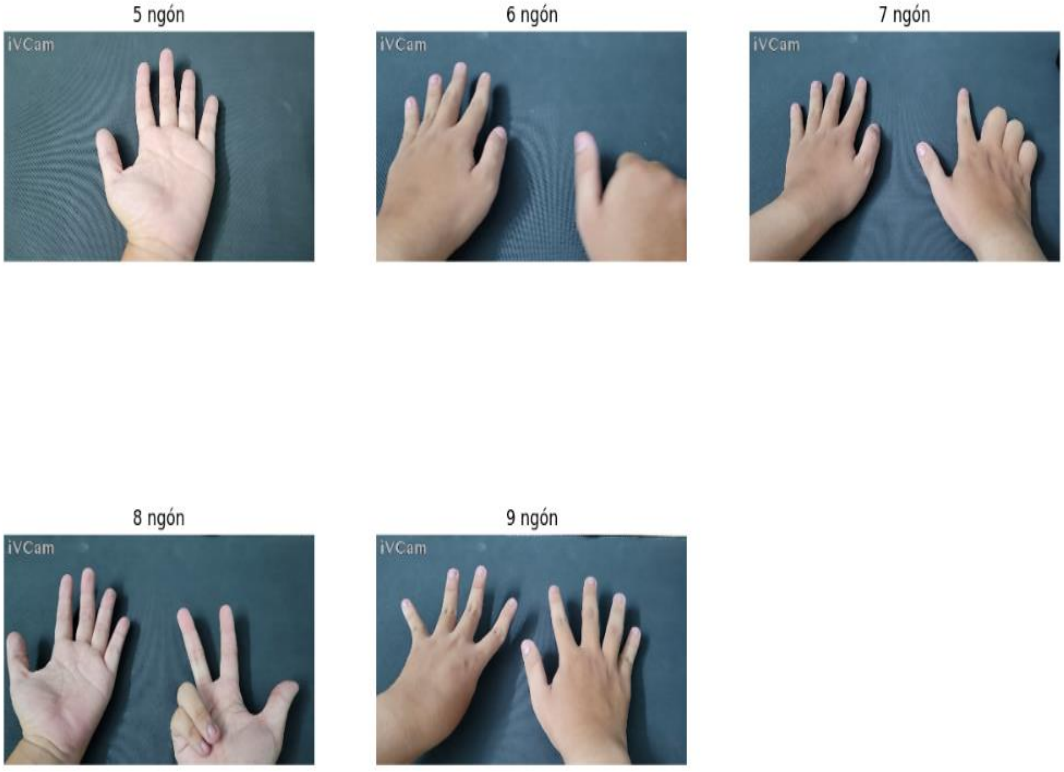
- Số lượng mẫu trong mỗi nhóm (5 đến 9 ngón) không có sự chênh lệch lớn, điều này có thể giúp đảm bảo tính cân bằng giữa các lớp trong quá trình huấn luyện mô hình.

Dung lượng tập dataset:

- Dung lượng tổng của dataset là khá lớn (925 MB), điều này cần xem xét để đảm bảo có đủ tài nguyên lưu trữ và xử lý khi làm việc với dữ liệu này.

Kích thước của ảnh:

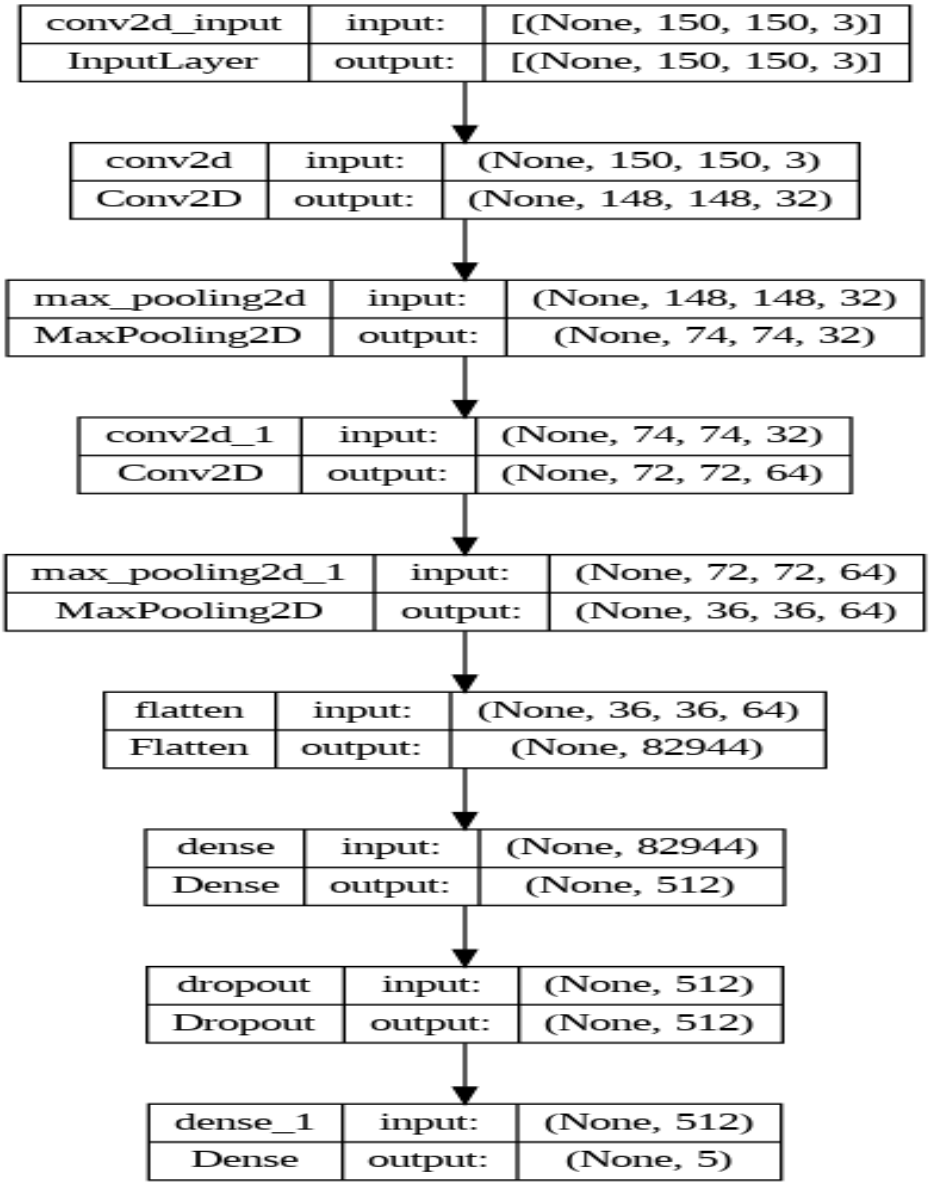
- Kích thước của hình ảnh là 384x216, điều này có thể ảnh hưởng đến hiệu suất của mô hình nếu không được xử lý một cách phù hợp.



Hình 5: Giới thiệu dữ liệu

2.2 Mô hình

- Mô hình CNN
  - Vẽ hình kiến trúc



Hình 6: Kiến trúc mô hình CNN

Bảng mô tả chi tiết kiến trúc

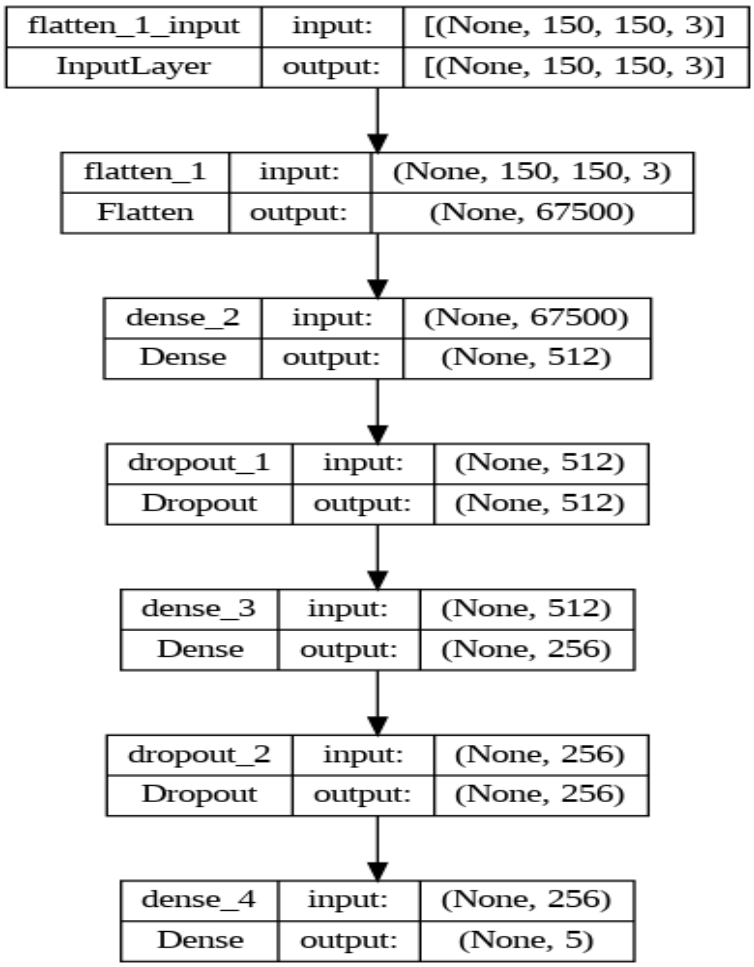
Tầng	Kích thước bộ lộc	Số lượng bộ lọc	Kích thước đầu ra	Hàm kích hoạt
Conv2D	3x3	32	(148, 148, 32)	ReLU
MaxPooling2D	2x2	-	(74, 74, 32)	-
Conv2D	3x3	64	(72, 72, 64)	ReLU
MaxPooling2D	2x2	-	(36, 36, 64)	-
Flatten	-	-	82944	-
Dense	-	512	512	ReLU
Dropout	-	-	-	-
Dense	-	5	5	-

- Hàm activation của tầng cuối
    - Tại tầng cuối của mô hình, hàm activation được sử dụng là softmax. Hàm softmax chuyển đổi vector logit (hay còn gọi là vector điểm số trước khi softmax) thành một vector xác suất bằng cách lấy exponents của mỗi output và sau đó chuẩn hóa những số này bằng cách chia cho tổng của tất cả exponents. Điều này giúp đảm bảo rằng tổng xác suất của tất cả các lớp là 1 và mỗi giá trị xác suất nằm trong khoảng từ 0 đến 1. Điều này rất hữu ích trong các tác vụ phân loại đa lớp.
- Hàm loss và Optimizer
    - Hàm Loss: Hàm mất mát được sử dụng trong mô hình chính là hàm categorical_crossentropy. Hàm mất mát categorical_crossentropy thường được sử dụng trong các tác vụ phân loại đa lớp khi các nhãn được mã hóa one-hot. Hàm này đo lường sự khác biệt giữa phân phối xác suất dự đoán và phân phối.

- Optimizer: Optimizer được chọn là Adam với learning rate là 0.001. Adam (Adaptive Moment Estimation) là một phương pháp tối ưu hóa được sử dụng để cập nhật trọng số mạng nơ-ron dựa trên tốc độ học thích ứng. Nó kết hợp hiệu quả của hai phương pháp tối ưu hóa khác là AdaGrad và RMSProp, đồng thời tính đến trung bình động của gradient và trung bình động của bình phương gradient.

- Mô hình MLP

- Vẽ hình kiến trúc



Hình 7: Kiến trúc mô hình MLP

Bảng mô tả chi kiến trúc mô hình

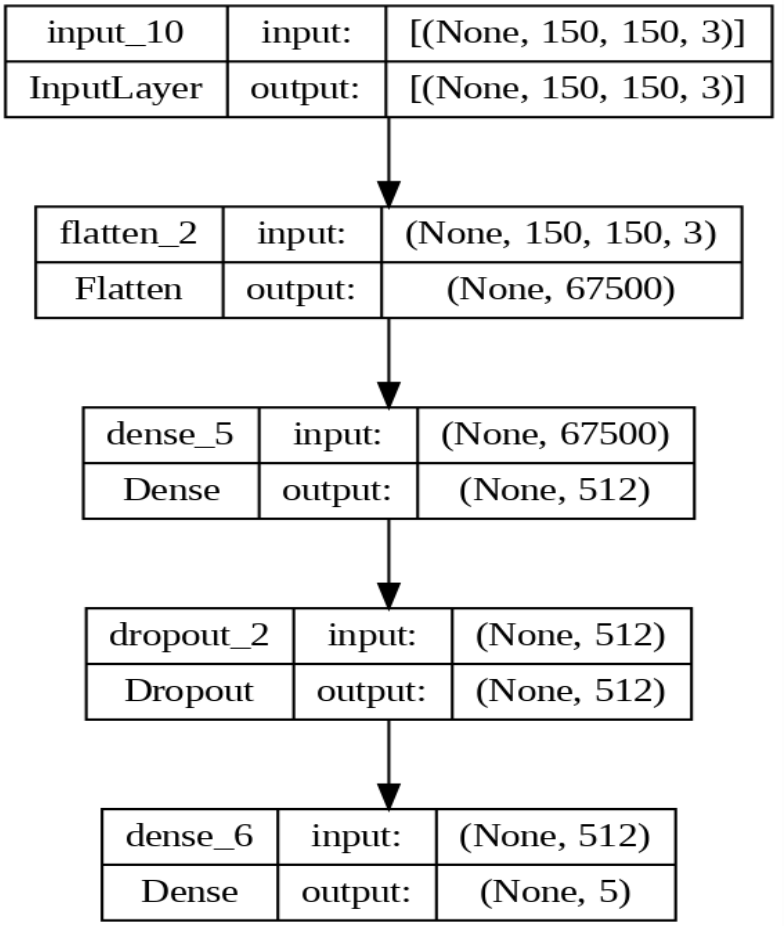
Tầng	Kích thước đầu vào	Kích thước đầu ra	Hàm kích hoạt	Ghi chú
Flatten	150x150x3	67500	-	Chuyển đổi đầu vào thành vector một chiều
Dense	67500	512	Relu	
Dropout	512	512	-	Tỉ lệ dropout 0.5
Dense	512	256	Relu	-
Dropout	256	256	-	Tỉ lệ dropout 0.5
Dense	256	5	Softmax	Tầng đầu ra cho 5-9 ngón

- Hàm Activation của Tầng Cuối
  - Tại tầng cuối cùng, hàm activation được sử dụng là softmax. Hàm này phù hợp cho các bài toán phân loại đa lớp vì nó chuyển đổi vector đầu ra của mô hình thành một phân phối xác suất trên các lớp đầu ra, giúp xác định lớp nào có khả năng cao nhất.
- Hàm Loss và Optimizer
  - Hàm Loss: categorical_crossentropy. Đây là lựa chọn phổ biến cho các bài toán phân loại đa lớp. Hàm mất mát này đo lường sự khác biệt giữa phân phối xác suất dự đoán và phân phối xác suất thực tế của nhãn, với mục tiêu là giảm thiểu sự khác biệt này trong quá trình huấn luyện.

- Optimizer: Adam với learning rate là 0.001. Adam là một phương pháp tối ưu hóa mạnh mẽ và hiệu quả.

- Mô hình VGG16

  - Vẽ hình kiến trúc



Hình 8: Kiến trúc mô hình VGG16

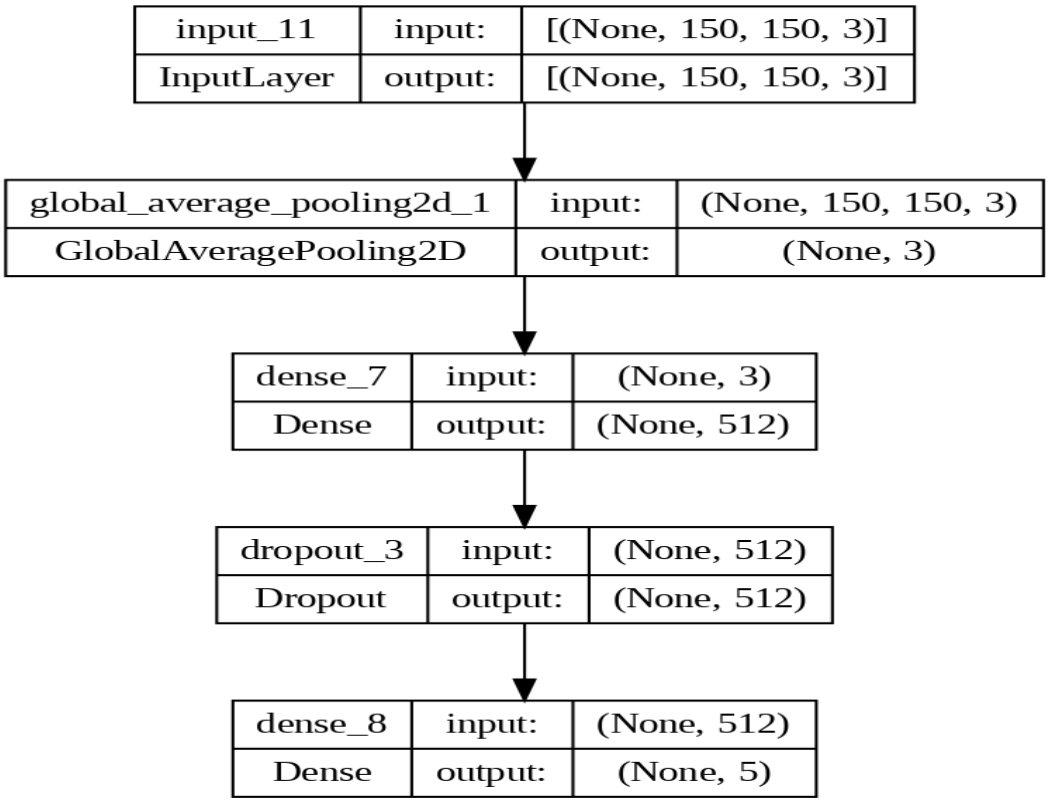
Bảng mô tả chi kiến trúc mô hình:

Tầng	Kích thước bộ lọc	Số lượng bộ lọc	Kích thước đầu ra	Hàm kích hoạt
VGG16 Convolutional Base	-	-	4.5x4.5x512	ReLu
Flatten	-	-	204800	-
Dense	-	512	512	ReLu
Dropout	-	-	-	-
Dense(Preditctions)	-	5	5	Softmax

- Hàm Activation của Tầng Cuối
  - Hàm activation tại tầng cuối là softmax, giúp chuyển đổi đầu ra của mô hình thành phân phối xác suất giữa 5 lớp tương ứng với số ngón tay từ 5 đến 9.
- Hàm Loss và Optimizer
  - Hàm Loss: categorical_crossentropy được sử dụng cho bài toán phân loại đa lớp với nhãn được mã hóa dưới dạng one-hot. Hàm này đo lường sự khác biệt giữa phân phối xác suất dự đoán và phân phối xác suất thực tế của nhãn.
  - Optimizer: Sử dụng Adam với learning rate 0.0001. Adam là một phương pháp tối ưu hóa mạnh mẽ, thích ứng tốc độ học cho từng tham số, giúp tối ưu hóa mô hình hiệu quả hơn.

- Mô hình ResNet50

  - Vẽ hình kiến trúc



Hình 9: Kiến trúc mô hình ResNet50

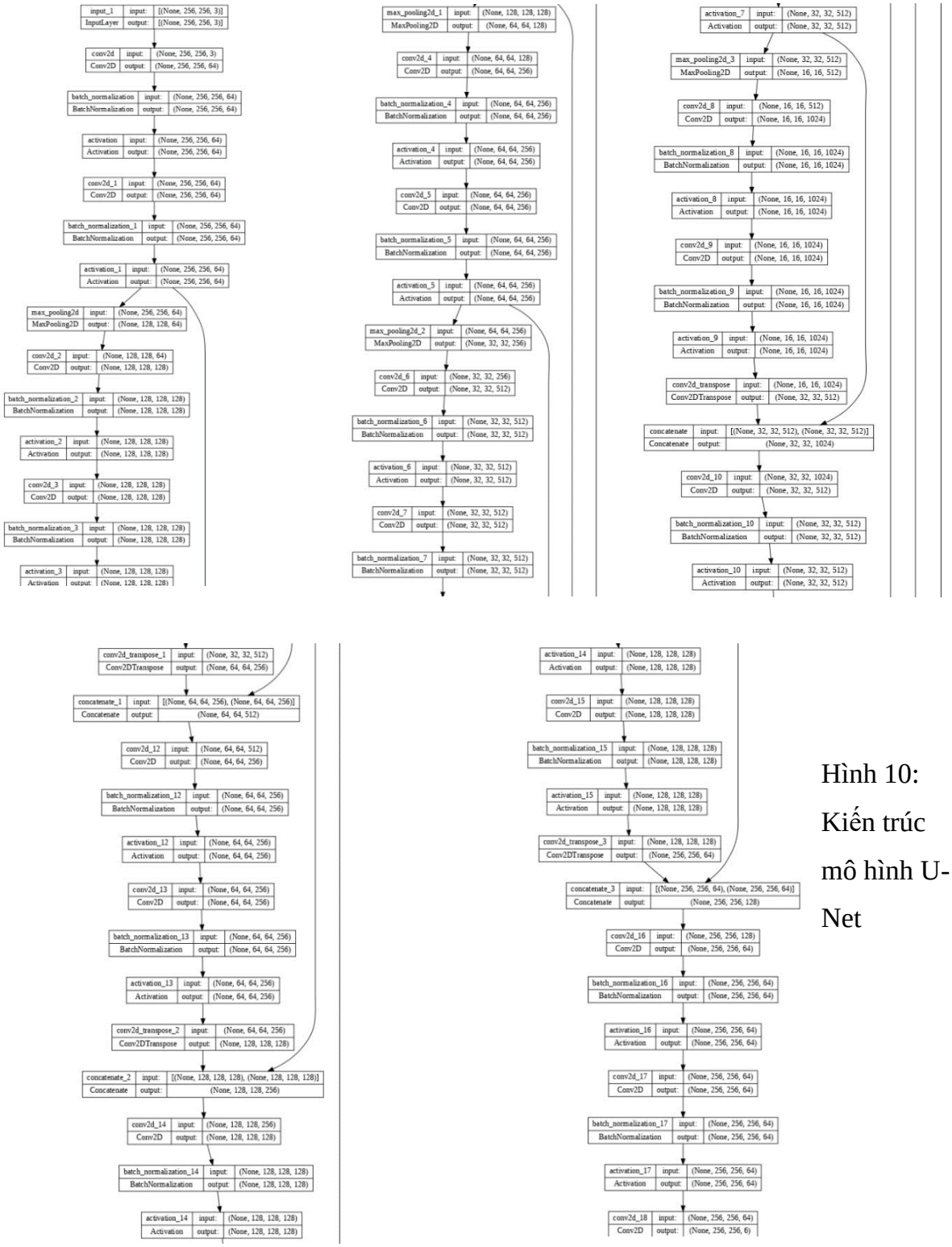
Bảng mô tả chi kiến trúc mô hình:

Tầng	Kích thước bộ lọc	Số lượng bộ lọc	Kích thước đầu ra	Hàm kích hoạt
ResNet50 Convolutional Base	-	-	Kích thước sau khi pooling	ReLu
GlobalAverage Pooling2D	-	-	Tùy thuộc vào kích thước đầu ra của ResNet50	-
Dense	-	512	512	ReLu
Dropout	-	-	-	-
Dense(Preditctions)	-	5	5	Softmax

- Hàm Activation của Tầng Cuối
  - Hàm activation: Là softmax tại tầng cuối, giúp chuyển đổi đầu ra của mô hình thành phân phối xác suất giữa 5 lớp tương ứng với số ngón tay từ 5 đến 9.
- Hàm Loss và Optimizer
  - Hàm Loss: Sử dụng categorical_crossentropy, lựa chọn tiêu chuẩn cho bài toán phân loại đa lớp khi nhãn được mã hóa dưới dạng one-hot.
  - Optimizer: Sử dụng Adam với learning rate 0.0001. Adam là một phương pháp tối ưu hóa mạnh mẽ, thích ứng tốc độ học cho từng tham số, giúp tối ưu hóa mô hình hiệu quả hơn.

- Mô hình U-NET

o Vẽ hình kiến trúc



Hình 10: Kiến trúc mô hình U-Net

Bảng mô tả chi kiến trúc mô hình:

Tầng	Loại lớp	Số lượng bộ lọc	Kích thước bộ lọc	Hàm kích hoạt	Ghi chú
Input	Input	N/A	N/A	N/A	(256, 256, 3)
Conv1	Conv2D	64	(3,3)	ReLu	padding=same
Conv2	Conv2D	64	(3,3)	ReLu	padding=same
Pool1	MaxPooling2D	N/A	(2,2)	N/A	
…	…	…	…	…	…
ConvTrans1	Conv2DTranspose	512	(2,2)	N/A	strides=(2, 2), padding=same
Concat1	Concatenate	N/A	N/A	N/A	
ConvLast1	Conv2D	64	(3,3)	ReLu	padding=same
ConvLast2	Conv2D	64	(3,3)	ReLu	padding=same
Output	Conv2D	6	(1,1)	Softmax	Tầng đầu ra, 6 lớp

- Hàm Activation của Tầng Cuối
  - Softmax: Được sử dụng ở tầng cuối cùng của mô hình để chuyển đổi các giá trị đầu ra thành các xác suất cho mỗi một trong 6 lớp đầu ra (từ 5 đến 9 ngón và lớp nền hoặc khác).
- Hàm Loss và Optimizer
  - Hàm Loss: chúng em đã chọn sử dụng Sparse Categorical Crossentropy làm hàm loss. Lựa chọn này dựa trên hai lý do chính:

- Số Lớp Đầu Ra: Mô hình U-Net của chúng em được thiết kế để phân biệt giữa nhiều lớp đối tượng khác nhau (từ 5 đến 9 ngón và một lớp cho nền hoặc khác). Sparse Categorical Crossentropy phù hợp cho các bài toán phân loại nhiều lớp như vậy.
- Biểu Diễn Nhãn: Khác với Categorical Crossentropy, hàm Sparse Categorical Crossentropy cho phép sử dụng nhãn dưới dạng chỉ số nguyên của lớp, thay vì phải biểu diễn nhãn dưới dạng one-hot encoding. Điều này giúp giảm bộ nhớ sử dụng và đơn giản hóa quy trình xử lý dữ liệu.

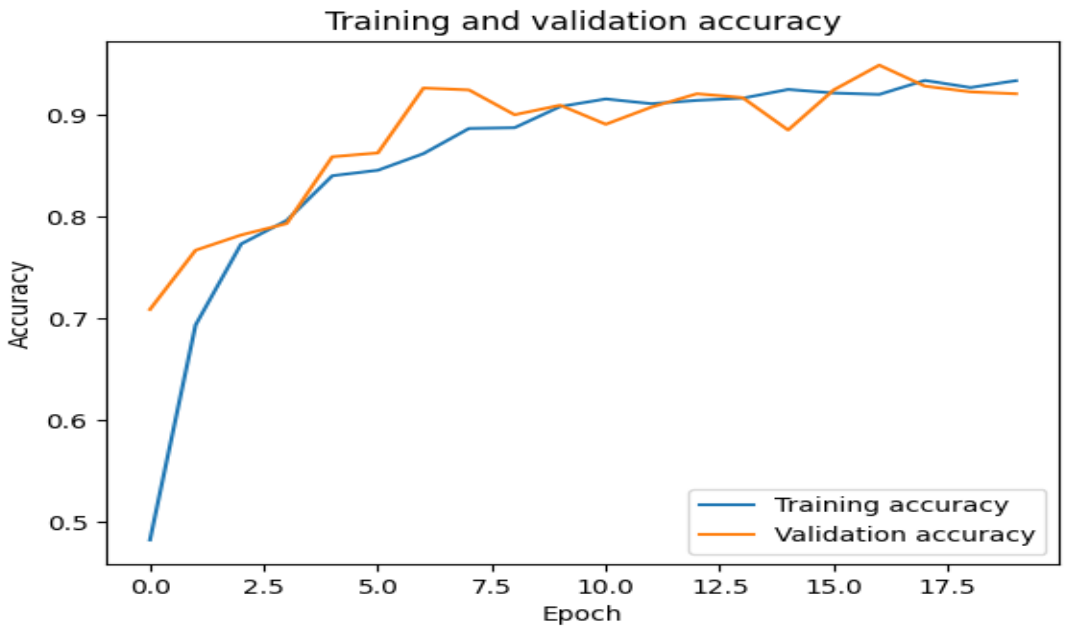
▪ Optimizer: Sử dụng Adam với learning rate 0.0001. Adam là một phương pháp tối ưu hóa mạnh mẽ, thích ứng tốc độ học cho từng tham số, giúp tối ưu hóa mô hình hiệu quả hơn.

Chương 3. Thực nghiệm

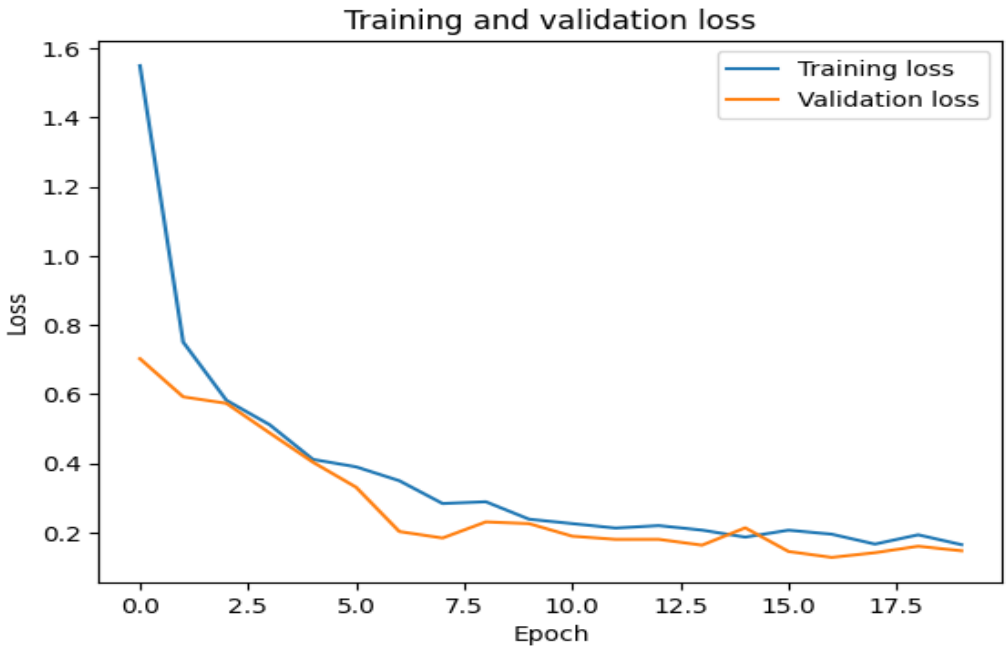
Mô hình CNN:

- Thiết lập siêu tham số:
    - Kích thước Batch (Batch Size): 32. Kích thước này giúp cân bằng giữa hiệu suất tính toán và độ chính xác của mô hình.
    - Tốc Độ Học (Learning Rate): 0.001. Được sử dụng cho optimizer Adam, tốc độ học này giúp cải thiện quá trình học mà không gây ra sự biến động lớn trong gradient.
    - Số Epochs: 20. Số lần lặp qua toàn bộ tập dữ liệu huấn luyện, đủ để mô hình đạt được sự ổn định trong quá trình học mà không gây overfitting.
    - Hàm Loss: Categorical Crossentropy. Phù hợp cho bài toán phân loại đa lớp.
    - Metrics: Accuracy. Độ chính xác được sử dụng như một chỉ số đánh giá hiệu suất mô hình.

Biểu đồ:



Hình 11: Đồ thị Accuracy CNN



Hình 12: Đồ thị Loss CNN

Bảng biểu:

Metric	Training	Validation	Test
Accuracy	0.96	0.94	0.93
Loss	0.12	0.15	0.18

Hình ảnh khi model predict

Hình 13: Test mô hình 5 ngón CNN

Hình 14: Test mô hình 6 ngón CNN

Hình 15: Test mô hình 7 ngón CNN

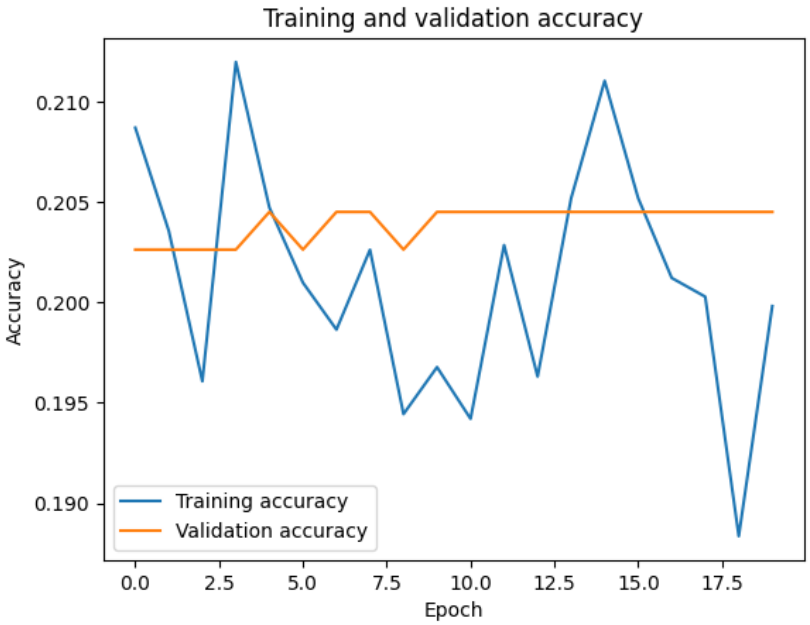
Hình 16: Test mô hình 8 ngón CNN

Hình 17: Test mô hình 9 ngón CNN

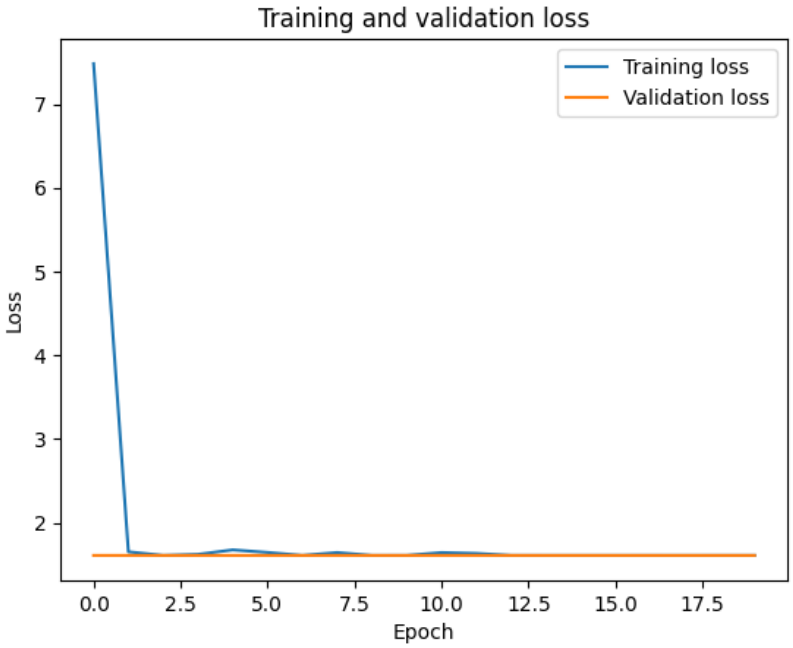
Mô hình MLP:

- Thiết lập siêu tham số:
    - Kích thước Batch (Batch Size): 32. Điều này cân nhắc giữa hiệu quả tính toán và độ chính xác.
    - Tốc Độ Học (Learning Rate): 0.001. Được sử dụng cho optimizer Adam, tốc độ học này giúp cải thiện quá trình học mà không gây ra sự biến động lớn trong gradient.
    - Số Epochs: 20. Số lần lặp qua toàn bộ tập dữ liệu, nhằm đảm bảo mô hình học đủ từ dữ liệu mà không bị overfitting.
    - Hàm Loss: Categorical Crossentropy phù hợp cho bài toán phân loại đa lớp.
    - Metrics: Accuracy. Đo lường độ chính xác của mô hình trên tập dữ liệu.

Biểu đồ:



Hình 18: Đồ thị Accuracy MLP



Hình 19: Đồ thị Loss MLP

Bảng biểu:

Metric	Training	Validation	Test
Accuracy	0.2	0.2	0.2
Loss	1.61	1.62	1.61

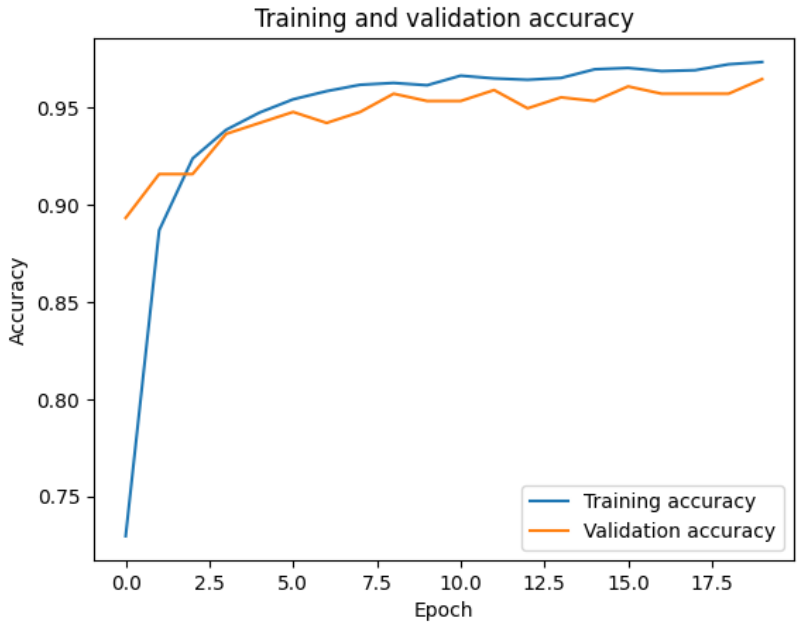
Hình ảnh khi model thực hiện predict

Hình 20: Test mô hình 5 ngón MLP

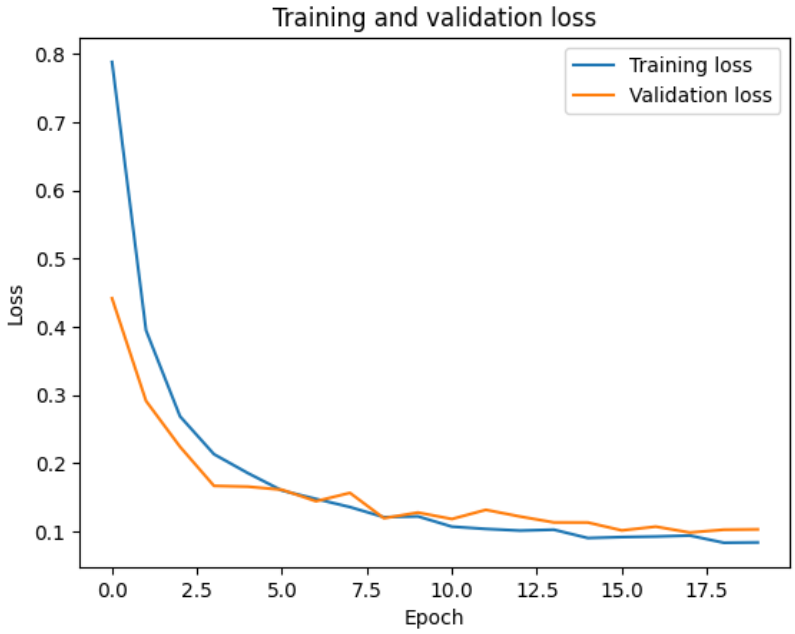
Mô hình VGG16:

- Thiết lập siêu tham số:
    - Kích thước Batch (Batch Size): 32. Điều này cân nhắc giữa hiệu quả tính toán và độ chính xác.
    - Tốc Độ Học (Learning Rate): 0.001. Được sử dụng cho optimizer Adam, tốc độ học này giúp cải thiện quá trình học mà không gây ra sự biến động lớn trong gradient.
    - Số Epochs: 20. Số lần lặp qua toàn bộ tập dữ liệu, nhằm đảm bảo mô hình học đủ từ dữ liệu mà không bị overfitting.
    - Hàm Loss: Categorical Crossentropy phù hợp cho bài toán phân loại đa lớp.
    - Metrics: Accuracy. Đo lường độ chính xác của mô hình trên tập dữ liệu.

Biểu đồ:



Hình 21: Đồ thị Accuracy VGG16



Hình 22: Đồ thị Loss VGG16

Bảng biểu:

Metric	Training	Validation	Test
Accuracy	0.97	0.96	0.96
Loss	0.08	0.1	0.08

Hình ảnh khi model thực hiện predict

Hình 23: Test mô hình 5 ngón VGG16


```
1/1 [==============================] - 0s 281ms/step
Mô hình dự đoán:  6 ngón
```

Hình 24: Test mô hình 6 ngón VGG16


```
1/1 [==============================] - 0s 404ms/step
Mô hình dự đoán:  7 ngón
```

Hình 25: Test mô hình 7 ngón VGG16


```
1/1 [==============================] - 0s 277ms/step
Mô hình dự đoán:  8 ngón
```

Hình 26: Test mô hình 8 ngón VGG16

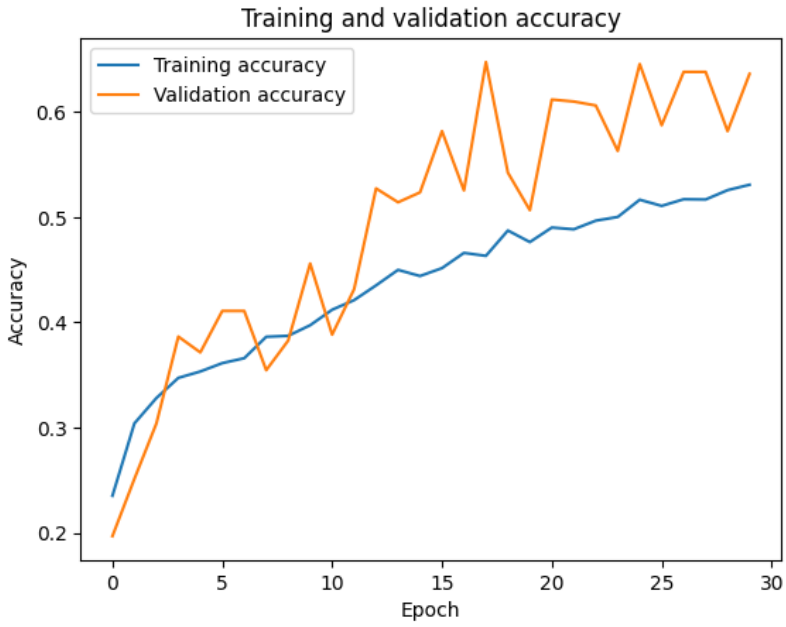

```
1/1 [==============================] - 0s 193ms/step
Mô hình dự đoán:  9 ngón
```

Hình 27: Test mô hình 9 ngón VGG16

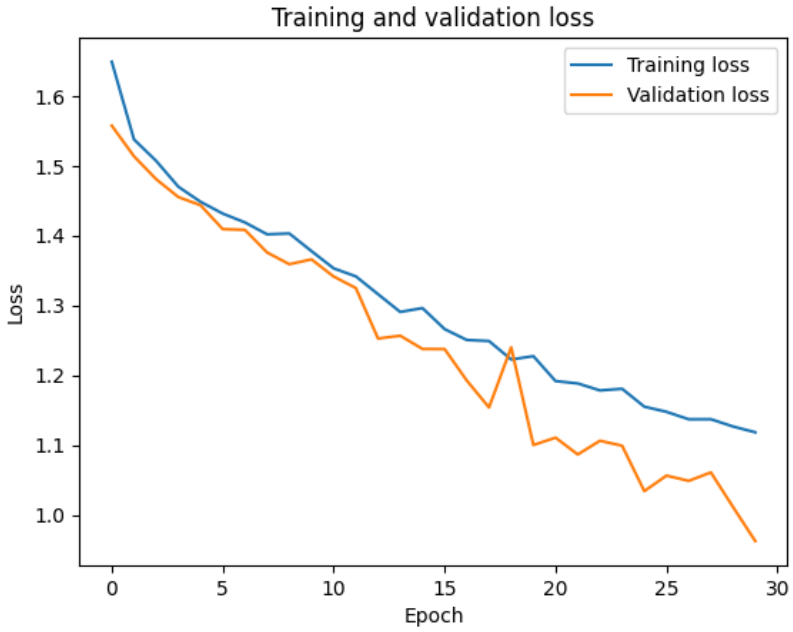
Mô hình ResNet50:

- Thiết lập siêu tham số:
    - Kích thước Batch (Batch Size): 32. Điều này cân nhắc giữa hiệu quả tính toán và độ chính xác.
    - Tốc Độ Học (Learning Rate): 0.001. Được sử dụng cho optimizer Adam, tốc độ học này giúp cải thiện quá trình học mà không gây ra sự biến động lớn trong gradient.
    - Số Epochs: 30. Số lần lặp qua toàn bộ tập dữ liệu, nhằm đảm bảo mô hình học đủ từ dữ liệu mà không bị overfitting.
    - Hàm Loss: Categorical Crossentropy phù hợp cho bài toán phân loại đa lớp.
    - Metrics: Accuracy. Đo lường độ chính xác của mô hình trên tập dữ liệu.

Biểu đồ:



Hình 28: Đồ thị Accuracy ResNet50



Hình 29: Đồ thị Loss ResNet50

Bảng biểu:

Metric	Training	Validation	Test
Accuracy	0.64	0.63	0.65
Loss	0.96	0.96	0.93

Hình ảnh khi mô hình thực hiện predict

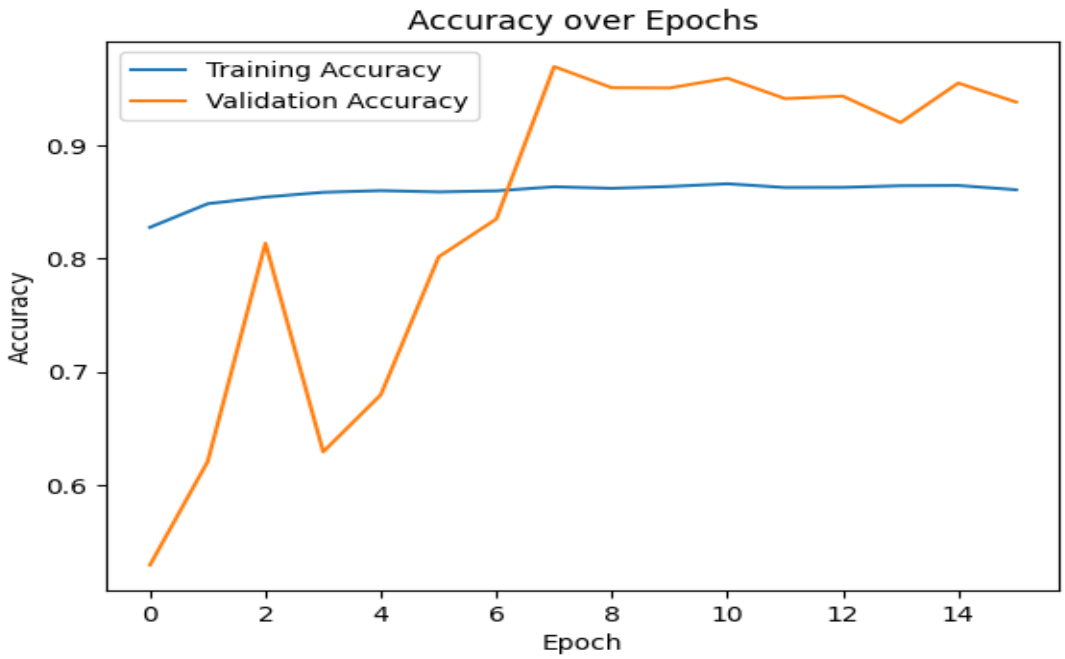
Hình 30: Test mô hình 5 ngón ResNet50

Hình 31: Test mô hình 9 ngón ResNet50

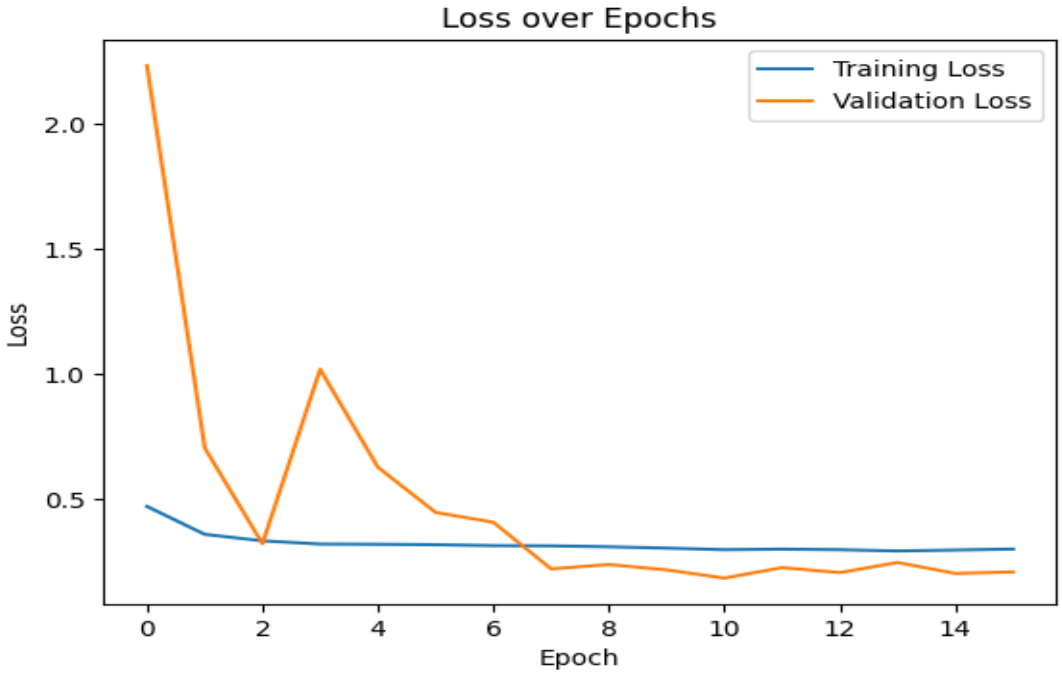
Mô hình U-NET:

- Thiết lập siêu tham số:
    - Kích thước batch (Batch Size): 32. Kích thước batch phù hợp giúp cân bằng giữa hiệu quả huấn luyện và yêu cầu về bộ nhớ.
    - Tốc độ học (Learning Rate): 0.001 cho optimizer Adam. Tốc độ học này giúp thuật toán tối ưu hóa tìm kiếm giá trị tối ưu một cách hiệu quả.
    - Số lần lặp qua dữ liệu (Epochs): 25. Đủ số lượng epochs để mô hình có thể học được các đặc trưng quan trọng từ tập dữ liệu mà không bị overfitting.
    - Hàm Loss: Sparse Categorical Crossentropy, được chọn do khả năng xử lý dữ liệu phân loại đa lớp với nhãn dưới dạng chỉ số.

Biểu đồ:



Hình 32: Đồ thị Accuracy U-NET

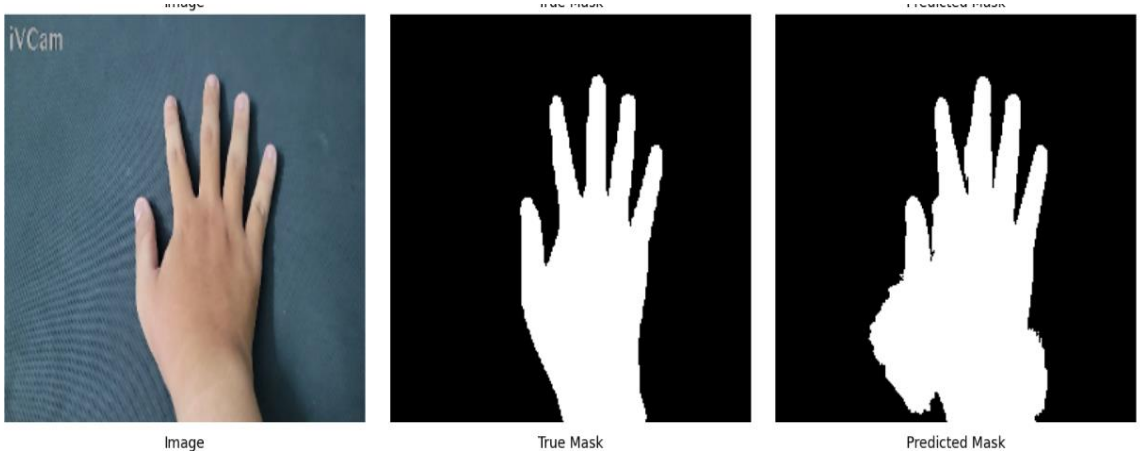


Hình 33: Đồ thị Loss U-NET

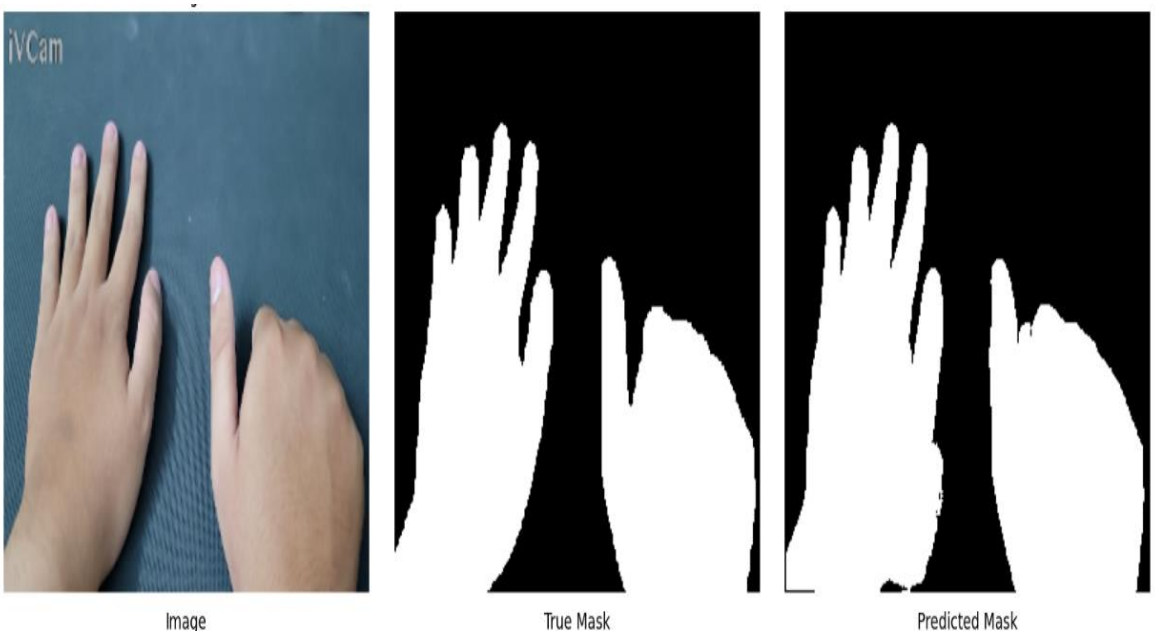
Bảng biểu:

Metric	Training	Validation	Test
Accuracy	0.88	0.93	0.94
Loss	0.28	0.21	0.18

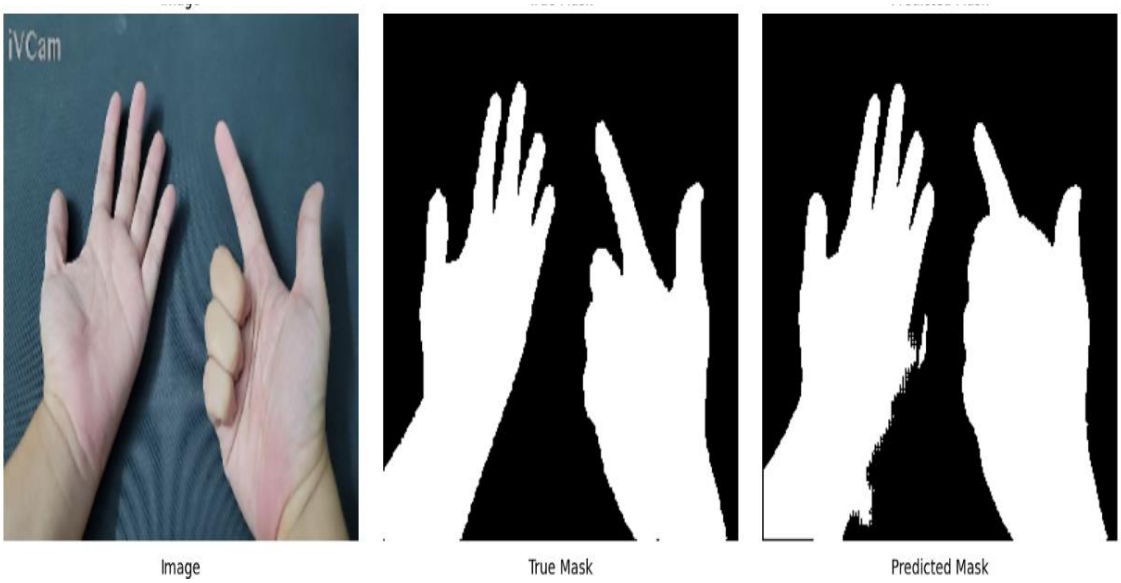
Hình ảnh khi mô hình thực hiện predict:



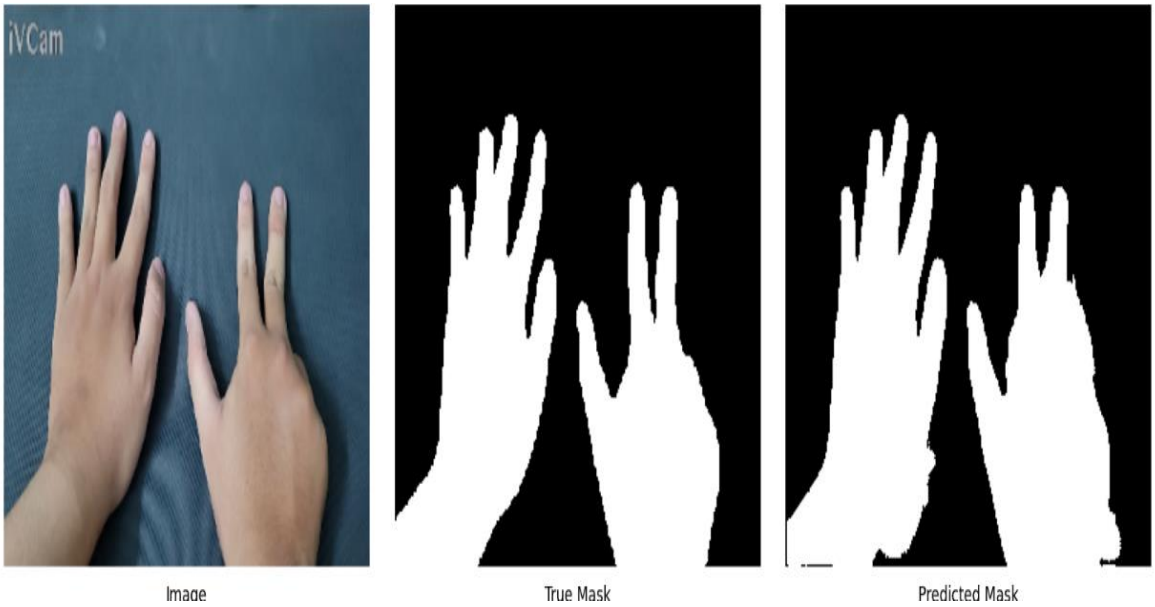
Hình 34: Test mô hình 5 ngón U-Net



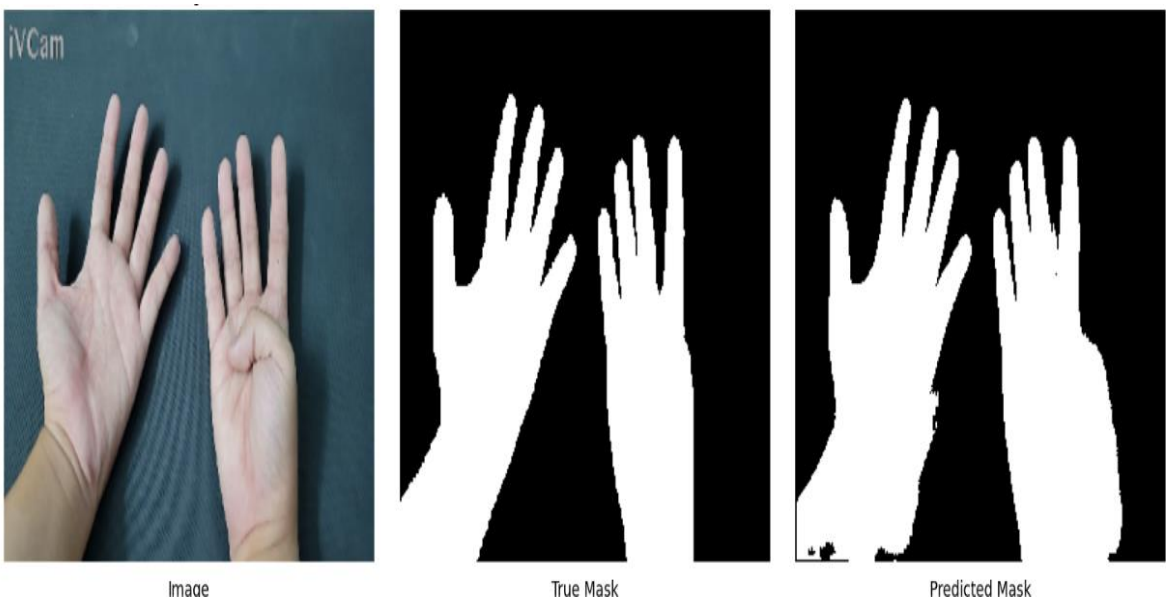
Hình 35: Test mô hình 6 ngón U-Net



Hình 36: Test mô hình 7 ngón U-Net

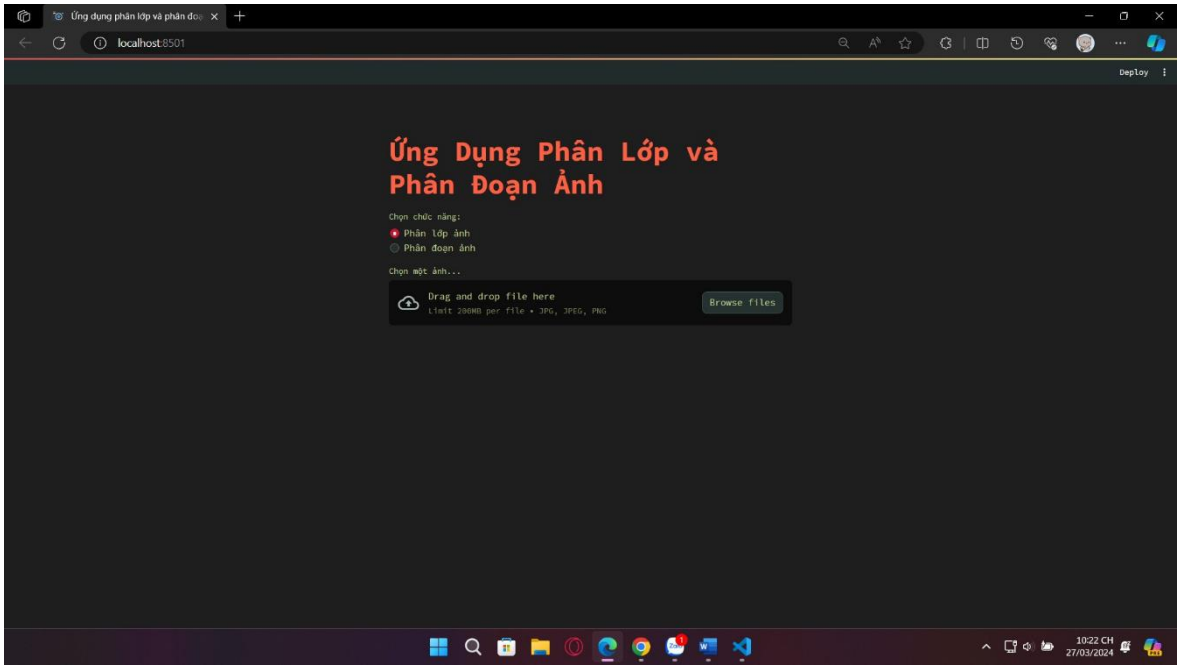


Hình 37: Test mô hình 8 ngón U-Net

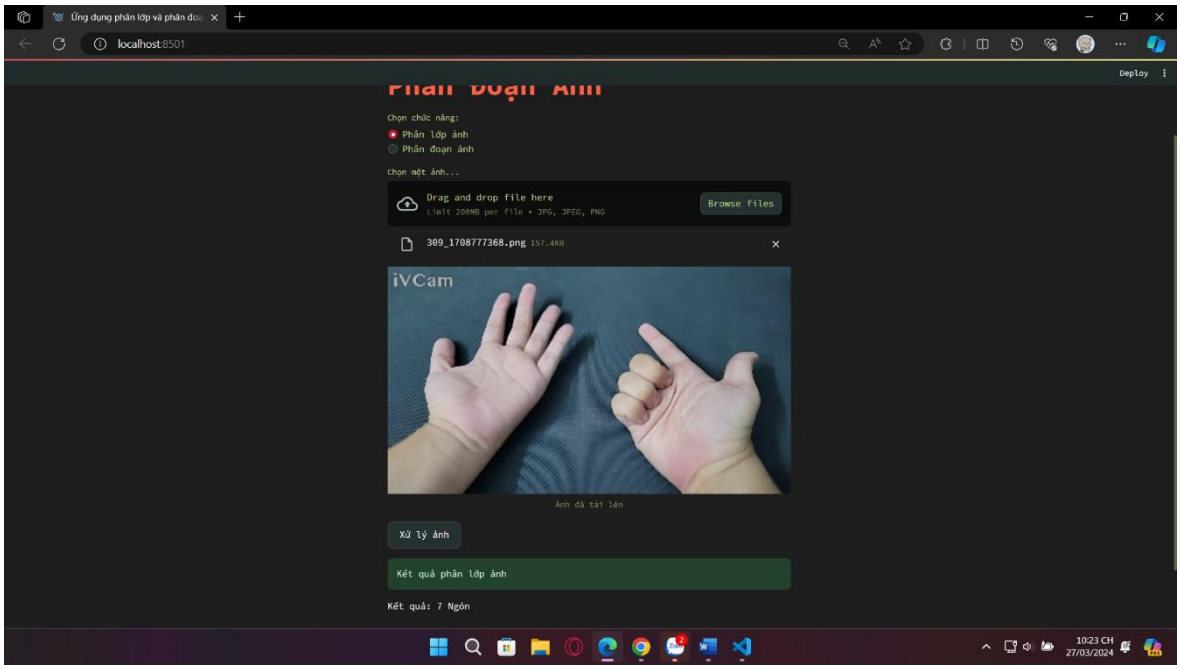


Hình 38: Test mô hình 9 ngón U-Net

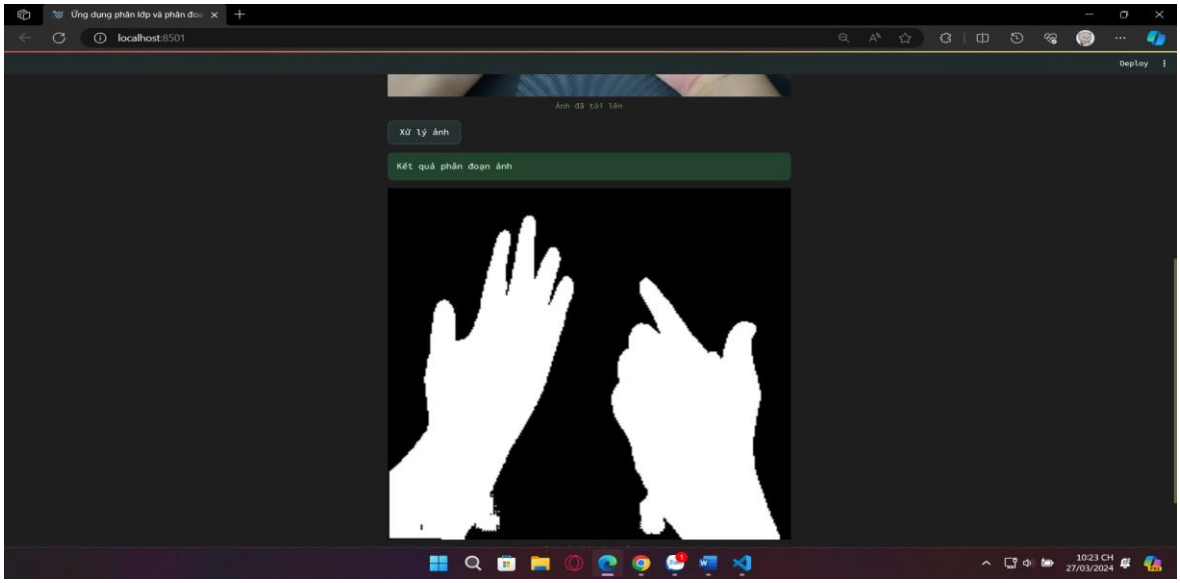
Hình ảnh khi triển khai mô hình trên website:



Hình 39: Hình ảnh khi triển khai mô hình trên website



Hình 40: Thực hiện phân lớp



Hình 41: Thực hiện phân đoạn

KẾT LUẬN

Những điều đã làm được

1. Thu thập dữ liệu khoảng hơn 1000 tấm cho mỗi class ảnh tương đương với số ngón tay từ 5-9.
2. Thực hiện các bước tăng cường dữ liệu và xử lý dữ liệu và chia dữ liệu theo tỷ lệ (9,1,1) cho tập train, val, test.
3. Xây dựng được 5 mô hình trong đó có 4 mô hình được sử dụng để phân lớp ảnh số ngón tay từ 5-9 là (CNN, MLP, VGG16,ResNet50) và một mô hình phân đoạn ảnh là (U-NET).
4. Đã thực hiện phân lớp khá chính xác (>90%) trừ MLP là 20% khi dự đoán được đúng số ngón tay khi đưa hình ảnh vào với mô hình và phân đoạn hình ảnh cũng khá chính xác khi phân đoạn được ngón tay khỏi nền với độ chính xác khoảng 90% tuy có vài hình ảnh chưa ổn khi so với tập test nhưng nhìn chung là ổn.
5. Xây dựng được website dựa trên streamlit để người dùng có thể sử dụng dễ dàng hơn việc phân đoạn và phân lớp hình ảnh.

Những điều chưa làm được

1. Chưa thu thập dữ liệu đa dạng đa số là từ webcam và một vài hình ảnh được chụp từ điện thoại số lượng người chụp không cao chỉ từ 2-3 người dữ liệu khá thiếu sự đa dạng dù đã tăng cường dữ liệu nhưng vẫn cần thêm hình ảnh các kiểu bàn tay khác và nền khác.
2. Phân đoạn độ chính xác cao nhưng hình ảnh khi predict chưa chính xác lắm cần cải thiện.
3. Giao diện website chưa được đẹp khá đơn giản và chưa được thực hiện kỹ càng những vẫn ở mức đủ dùng và thể hiện được chức năng.
4. Dù độ chính xác cao nhưng vẫn có trường hợp dự đoán hình ảnh sai nhưng không nhiều có thể do bộ dữ liệu không đủ đa dạng.

Hướng phát triển

1. Thu thập dữ liệu kỹ càng hơn và từ nhiều nguồn và tăng cương thêm hình ảnh để có được kết quả train tốt nhất và phù hợp với hiện thực để mô hình tránh bị overfitting.
2. Cải thiện mô hình phân đoạn để có kết quả chính xác hơn.
3. Kết hợp thêm các mô hình khác và có thể mở rộng sang thêm hướng xác định vị trí của vật thể để giải quyết thêm các bài toán khác.
4. Tìm hiểu thêm các mô hình phân đoạn khác để tích hợp vào như Mask R-CNN.
5. Mở rộng hơn không chỉ dừng lại ở 5-9 ngón mà là cả 2 bàn tay.

TÀI LIỆU THAM KHẢO

1. OpenCV Python Image Preprocessing for VGG16 Model- https://stackoverflow.com/questions/62370995/opencv-python-image-preprocessing-for-vgg16-model - Ngày truy cập gần nhất: 15/03/2024
2. Image_classification using resnet50 model with imagenet db with my custom labels - https://stackoverflow.com/questions/57782824/image-classification-using-resnet50-model-with-imagenet-db-with-my-custom-labels - Ngày truy cập gần nhất: 15/03/2024
3. Xây dựng mạng Unet cho bài toán Segmentation - https://viblo.asia/p/xay-dung-mang-unet-cho-bai-toan-segmentation-gDVK2QOX5Lj - Ngày truy cập gần nhất: 15/03/2024
4. AnyLabeling v0.2.22 - Segment Anything Demo - https://www.youtube.com/watch?v=5qVJiYNX5Kk - Ngày truy cập gần nhất: 15/03/2024
5. Phamdinhkhanh - 20 Jun 2020 - Bài 42 - Thực hành Unet - https://phamdinhkhanh.github.io/2020/06/20/Unet.html - Ngày truy cập gần nhất: 15/03/2024
6. Nttuan8 - May 12, 2019 - Bài 12: Image segmentation với U-Net - https://nttuan8.com/bai-12-image-segmentation-voi-u-net/ - Ngày truy cập gần nhất: 15/03/2024