

|  | **TRƯỜNG ĐẠI HỌC BÁCH KHOA**  **KHOA CÔNG NGHỆ THÔNG TIN** |  |
| --- | --- | --- |

**BÁO CÁO**

**PBL 5 - ĐỒ ÁN KỸ THUẬT MÁY TÍNH**

**TÊN ĐỀ TÀI**

**CHUYỂN HÀNH ĐỘNG THÀNH VĂN BẢN**

GV Hướng dẫn: TS. Ninh Khánh Duy

Nhóm đồ án: 28

| HỌ VÀ TÊN SINH VIÊN | LỚP HỌC PHẦN | ĐIỂM BẢO VỆ |
| --- | --- | --- |
| Trương Bích Quỳnh | 20.12A |  |
| Nguyễn Thị Thanh Hiền | 20.12A |  |
| Nguyễn Phước Nhâm | 20.12A |  |

ĐÀ NẴNG, 05/2023

**MỤC LỤC**

[**TÓM TẮT ĐỒ ÁN 4**](#_heading=h.pcrah5awt97d)

[**BẢNG PHÂN CÔNG NHIỆM VỤ 5**](#_heading=h.s1apbmcrsoh9)

[**MỤC LỤC HÌNH ẢNH 6**](#_heading=h.6f6jb5p0f60s)

[1. Giới thiệu 8](#_heading=h.sr2zng4se3mn)

[1.1. Tổng quan 8](#_heading=h.30j0zll)

[1.2. Mục tiêu 8](#_heading=h.1fob9te)

[2. Giải pháp 8](#_heading=h.3znysh7)

[2.1. Phần cứng 8](#_heading=h.fxxv2w3whpao)

[2.2. Phần mềm 8](#_heading=h.1aas3ixdzlwu)

[2.3. Cấu trúc hệ thống phần mềm 8](#_heading=h.v16wp9ecos44)

[2.4. IoT 9](#_heading=h.yn8jgmpwyq7y)

[2.5. AI 9](#_heading=h.qvnymoazonie)

[2.5.1. Bài toán nhận dạng chuỗi hành động 9](#_heading=h.7enypoudcau7)

[2.5.2. Tổng quan kiến trúc của một hệ thống nhận dạng chuỗi hành động 9](#_heading=h.j2m3zgcaa686)

[2.5.3. Hạn chế, tồn tại của các phương pháp 10](#_heading=h.20mu8qyir3ej)

[2.6. Một số mô hình nhận dạng hành động 11](#_heading=h.8nal4fb5pfbb)

[2.7. Tổng quan dữ liệu 11](#_heading=h.yuwok6nmwhwn)

[2.8. Quá trình huấn luyện và trích xuất đặc trưng 14](#_heading=h.nr0mioa11z51)

[2.8.1. Lần 1 14](#_heading=h.p1nwz7u4u58k)

[2.8.2. Lần 2: 19](#_heading=h.okg5lq4akffl)

[2.8.3. Lần 3: 25](#_heading=h.vu19bsq5lpx8)

[Viết mã nguồn 31](#_heading=h.vsu4fw6so1bw)

[2.9. Server 32](#_heading=h.rly0pw46akug)

[2.10. Nguyên lý hoạt động của phần cứng 32](#_heading=h.4i0hj1yipi0b)

[2.11. Nguyên lý hoạt động hệ thống nhận dạng ngôn ngữ ký hiệu 32](#_heading=h.4ytzjq94kta1)

[2.12. Các linh kiện sử dụng 33](#_heading=h.q7z2u59xpex1)

[Tham số kỹ thuật đầu vào-ra và nguyên tắc hoạt động 33](#_heading=h.zekjr1f0mf7v)

[Bảng kê chi phí linh kiện 34](#_heading=h.rip10yce9yrg)

[3. Kết quả 34](#_heading=h.w625v6op2ev)

[3.1. Dữ liệu đã sử dụng 34](#_heading=h.mqct7iw4jbll)

[3.2. Các công cụ và framework đã dùng 35](#_heading=h.1ksv4uv)

[3.3. Các điều kiện tiến hành thực nghiệm 35](#_heading=h.44sinio)

[3.4. Các độ đo (metrics) cụ thể dùng để đánh giá hiệu suất của giải pháp 35](#_heading=h.2jxsxqh)

[3.5. Quy trình và điều kiện kiểm thử hệ thống, và các kết quả đánh giá hệ thống theo các tiêu chí (đã nêu) 35](#_heading=h.z337ya)

[Quy trình 35](#_heading=h.nrtcpx6blmtg)

[Kết quả 36](#_heading=h.1y810tw)

[3.6. Đánh giá 36](#_heading=h.inev1jq41tqr)

[4. Kết luận 37](#_heading=h.b4219qni3yru)

[4.1. Đánh giá 37](#_heading=h.5fa8b17md5m1)

[4.2. Kết luận 37](#_heading=h.f31rfs3ixpm)

[4.3. Hướng phát triển 37](#_heading=h.1pxezwc)

[TÀI LIỆU THAM KHẢO 38](#_heading=h.xfnohfvkj705)

# TÓM TẮT ĐỒ ÁN

Ngôn ngữ ký hiệu tay đã trở thành một phương pháp giao tiếp quan trọng và phổ biến trong cộng đồng người khiếm thính, mang lại sự tiếp cận thông tin và truyền đạt ý nghĩa cho những người có khả năng nghe kém. Tuy nhiên, hiện tại, các hệ thống nhận dạng ngôn ngữ ký hiệu tay hiện có vẫn chưa đáp ứng đầy đủ tiện ích và hiệu quả cho người sử dụng. Vì vậy, nhóm chúng em đã phát triển một hệ thống nhận dạng ngôn ngữ ký hiệu tay sử dụng camera để chuyển đổi hành động thành văn bản, nhằm giúp người khiếm thính giao tiếp một cách tự nhiên hơn. Để đạt được mục tiêu này, chúng em đã tiến hành thu thập dữ liệu video cho từng cử chỉ, trích xuất đặc trưng và xây dựng mô hình học máy để nhận dạng các hành động riêng lẻ. Sau đó, chúng em kết hợp các hành động để tạo chuỗi và chuyển đổi chúng thành văn bản. Hệ thống của chúng em được thiết kế để mang lại sự thuận tiện, linh hoạt và tiếp cận thông tin cho người khiếm thính, góp phần tạo điều kiện giao tiếp tốt hơn và tăng cường sự kết nối trong cộng đồng người khiếm thính. Chúng em hy vọng rằng công nghệ này sẽ giúp người khiếm thính truyền đạt ý kiến, suy nghĩ và cảm xúc một cách dễ dàng và tự tin hơn, đồng thời tạo ra một môi trường giao tiếp bình đẳng và thuận lợi cho tất cả mọi người.

# BẢNG PHÂN CÔNG NHIỆM VỤ

| Sinh viên thực hiện | Các nhiệm vụ | Tự đánh giá theo 3 mức |
| --- | --- | --- |
| Nguyễn Phước Nhâm | * AI * Thử nghiệm, train model * Xây dựng API Server * Viết báo cáo | * Đã hoàn thành |
| Nguyễn Thị Thanh Hiền | * Thu thập dữ liệu * Web FE * Sử dụng ESP cam gửi video lên server * Viết báo cáo | * Đã hoàn thành |
| Trương Bích Quỳnh | * Thu thập dữ liệu * Triển khai API vào hệ thống * Sử dụng ESP cam gửi video lên server * Viết báo cáo | * Đã hoàn thành |

# MỤC LỤC HÌNH ẢNH

[Hình 1:](#_heading=h.tyjcwt)Esp32 Web Server 8

[Hình 2:](#_heading=h.tyjcwt) Hệ thống nhận dạng mặt người 9

[Hình 3:](#_heading=h.1t3h5sf) Các bước trong hệ thống nhận dạng 10

[Hình 4: Workflow của MTCNN](#_heading=h.4d34og8) 13

[Hình 5:](#_heading=h.2s8eyo1) Mô hình Siam Network 14

[Hình 6:](#_heading=h.26in1rg) Thuật toán SVM 15

[Hình 7: Tìm hình khuôn mặt trong ảnh](#_heading=h.lnxbz9) 16

[Hình 8:](#_heading=h.tyjcwt) Nhận dạng khuôn mặt 16

[Hình 9:](#_heading=h.26in1rg) Sơ đồ ngữ cảnh hệ thống 18

[Hình 10:](#_heading=h.lnxbz9) Sơ đồ chương trình nhận diện 19

[Hình 11:](#_heading=h.tyjcwt) Cấu trúc thư mục Dataset 19

[Hình 12:](#_heading=h.26in1rg) Ảnh được sinh viên upload 20

[Hình 13: Ảnh khuôn mặt sau khi xử lý](#_heading=h.lnxbz9) 20

[Hình 14:](#_heading=h.tyjcwt) Ảnh khuôn mặt sau khi chuyển thành Vector 20

[Hình 15:](#_heading=h.26in1rg) Kết quả nhận diện 21

[Hình 16: Mô hình phần cứng](#_heading=h.lnxbz9) 22

[Hình 17:](#_heading=h.tyjcwt) Hệ thống Arduino và Kết nối IoT 23

[Hình 18:](#_heading=h.1t3h5sf)  [Sơ đồ hệ thống điểm danh](#_heading=h.4d34og8) 24

[Hình 19: Nguyên lý hoạt động của Servo motor](#_heading=h.2s8eyo1) 25

[Hình 20: Database User](#_heading=h.26in1rg) 26

[Hình 21: Database Courses](#_heading=h.lnxbz9) 26

[Hình 22:](#_heading=h.1t3h5sf) [Database Departments](#_heading=h.35nkun2) 27

[Hình 23: Trang đăng nhập 14](#_heading=h.4i7ojhp)

[Hình 24: Trang đăng ký tài khoản 1](#_heading=h.2xcytpi)5

Hình 25: Chọn học phần điểm danh (Student) 15

Hình 26: Điểm danh (Teacher) 16

Hình 27: Thêm học phần (Admin) 16

Hình 28: Thêm khoa (Admin) 17

Hình 29: Thêm giảng viên (Admin) 17

Hình 30: Thêm sinh viên (Admin) 18

Hình 31: Xem danh sách giảng viên (Admin) 18

Hình 32: Xem danh sách giảng viên (Admin) 19

Hình 33: Xem danh sách học phần (Admin) 19

\

## Giới thiệu

### Tổng quan

Ngôn ngữ ký hiệu tay đã trở thành một phương pháp giao tiếp quan trọng và phổ biến trong cộng đồng người khiếm thính . Tuy nhiên, hiện tại, các hệ thống nhận dạng ngôn ngữ ký hiệu tay vẫn chưa đáp ứng đầy đủ nhu cầu và tiện ích của người sử dụng.Vì vậy, nhóm chúng chúng em đã đề xuất và phát triển một hệ thống nhận dạng ngôn ngữ ký hiệu tay sử dụng camera, nhằm giúp người khiếm thính giao tiếp một cách tự nhiên và thuận tiện hơn. Chúng em tập trung vào việc nhận dạng và chuyển đổi các cử chỉ ngôn ngữ ký hiệu tay thành văn bản, nhằm mang lại sự linh hoạt và tiện lợi cho người sử dụng.

### Mục tiêu

Mục tiêu của dự án này là tạo ra một thiết bị có thể nhận dạng các cử chỉ ngôn ngữ ký hiệu của tay được camera để ghi lại và chuyển đổi chúng thành văn bản.

## Giải pháp

### Phần cứng

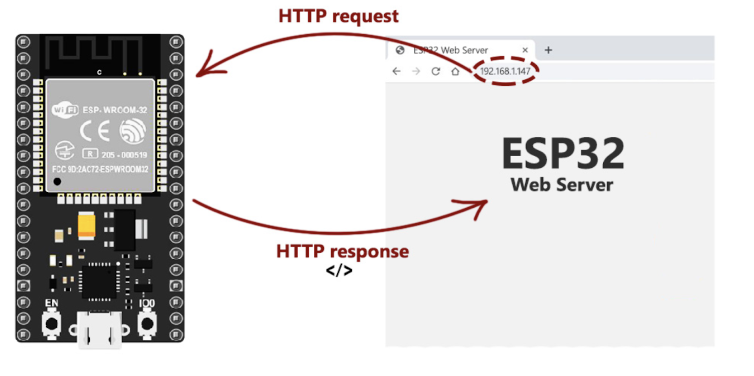
Sử dụng camera ESP32 để stream, quay và ghi video các cử chỉ ngôn ngữ ký hiệu tay.

### Phần mềm

Nhận yêu cầu từ người dùng: bắt đầu quay, kết thúc quay, gửi video nhận được lên server.

### Cấu trúc hệ thống phần mềm

* Web FE*:* 
  + Sử dụng thư viện DNS server để tạo một DNS server, cho phép client sử dụng tên miền là địa chỉ IP của WiFi để kết nối đến ESP32. Khi ESP32 kết nối đến WiFi, nó sẽ được cấp một địa chỉ IP, và đó sẽ là địa chỉ IP mà DNS server sẽ sử dụng để tạo tên miền, cho phép client sử dụng tên miền là địa chỉ ip của wifi để kết nối đến ESP32.
  + Tạo trang HTML để gửi yêu cầu http đến server của ESP32



*Hình 1. Esp32 Web Server*

* Gửi video lên server:

Khi người dùng click record/stop trên Web FE, sẽ gửi yêu cầu HTTP (POST request) đến ESP32, được xử lý bởi recorder\_handler/stop\_handler. Sau khi ghi xong, ESP32 sẽ gửi file video lên DNS server với URL là ip/stop. Trang Web FE sẽ lấy file video từ DNS server và gửi lên API, sau đó nhận kết quả về và hiển thị lên trang.

### IoT

Sử dụng camera ESP32 để thu thập video và gửi video lên server, nhận kết quả từ server hiển thị ra màn hình.

### AI

#### Bài toán nhận dạng chuỗi hành động

- Để giải quyết bài toán nhận dạng chuỗi hành động ta cần nhận dạng được 1 hành động riêng lẻ:

+ Cần thu thập dữ liệu là video của từng hành động riêng lẻ.

+ Xử lý các video thu thập được, trích xuất các đặc trưng từ video.

+ Tạo mô hình huấn luyện dựa trên những video đã thu thập.

- Áp dụng mô hình huấn luyện cho nhận dạng chuỗi hành động:

+ Đầu vào là 1 video gồm nhiều hành động.

+ Đọc video, lấy từng đoạn video gồm 1 số frame nhất định (số lượng frame được chọn dựa trên số lượng frame cho mỗi hành động riêng lẻ đã thu thập), sau đó đưa vào mô hình để dự đoán trả về kết quả cho 1 hành động. từ đó ta có được kết quả cho chuỗi hành động

#### Tổng quan kiến trúc của một hệ thống nhận dạng chuỗi hành động

**Bước 1:** Đọc từng frame của video:

Đầu tiên đọc lần lượt từng frame của video đầu vào.

**Bước 2:** Trích xuất keypoint từ frame:

+ Sử dụng thư viện Mediapipe để trích xuất các keypoint từ frame hiện tại. Điều này bao gồm trích xuất các điểm vị trí của tay và pose (tư thế cơ thể).

+ Thêm các keypoint đã trích xuất vào một danh sách tạm thời.

**Bước 3:** Dự đoán hành động:

+ Kiểm tra xem danh sách tạm thời đã đủ số lượng frame cần thiết hay chưa để đưa vào mô hình dự đoán. Nếu đã đủ, tiến hành dự đoán hành động.

+ Đưa danh sách keypoint vào mô hình LSTM đã được huấn luyện để dự đoán hành động tương ứng.

+ Kiểm tra kết quả dự đoán và xác định độ chính xác bằng một ngưỡng xác định trước. Nếu kết quả đạt ngưỡng xác định, thêm hành động tương ứng vào danh sách kết quả.

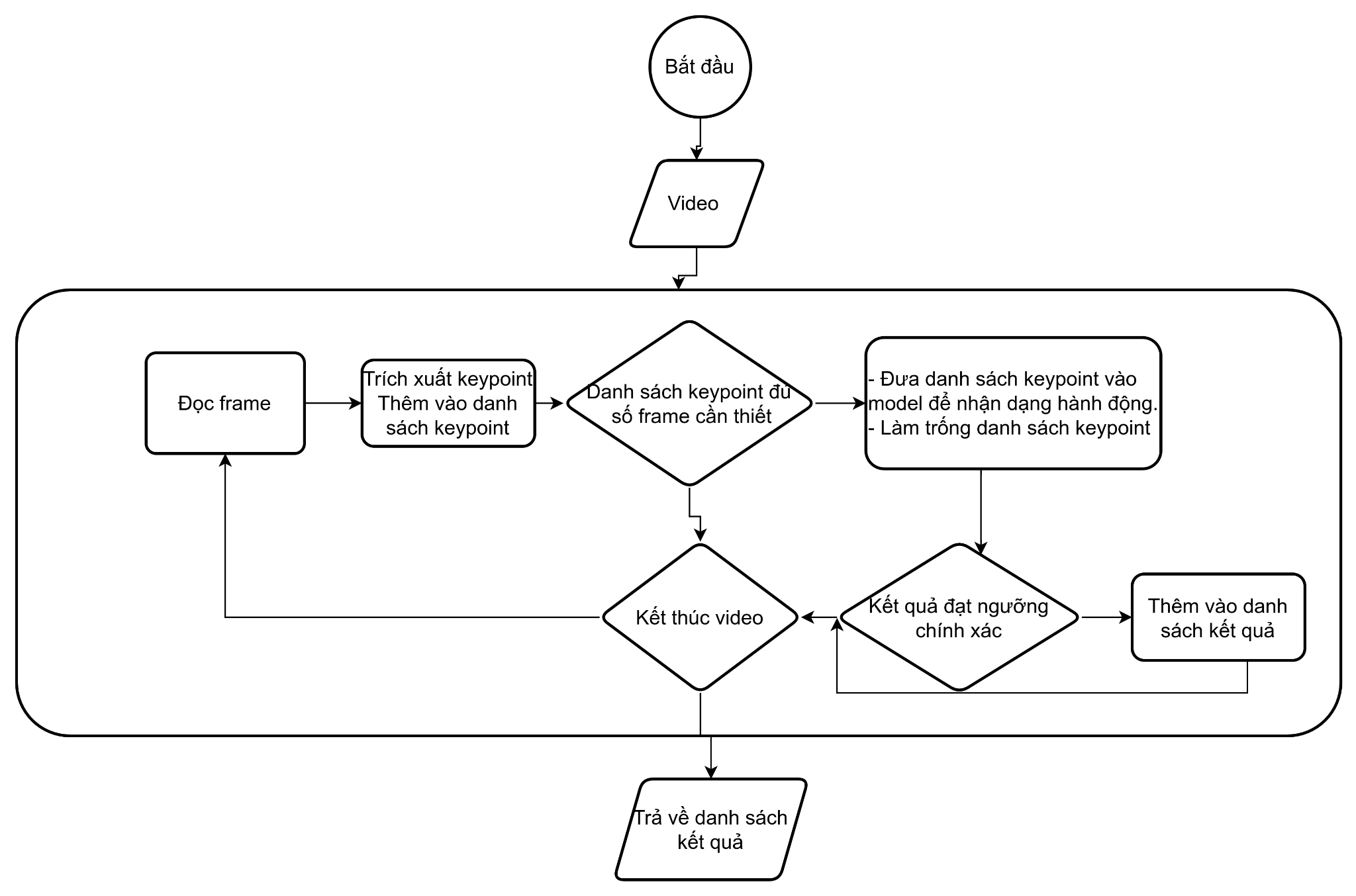
+ Sau đó, làm trống danh sách tạm thời để chuẩn bị cho frame tiếp theo.

**Bước 4:** Lặp lại cho đến khi kết thúc video:

+ Lặp lại các bước 2 và 3 cho đến khi đã xử lý hết tất cả các frame trong video đầu vào.

**Kết quả:**

Sau quá trình xử lý, danh sách kết quả sẽ chứa các hành động có trong video theo thứ tự xuất hiện.



*Hình 2: Các bước trong hệ thống nhận dạng*

#### Hạn chế, tồn tại của các phương pháp

- Độ chính xác: Độ chính xác khi nhận dạng không cao khi nhận dạng chuỗi hành động, có thể trả về nhiều kết quả không mong muốn.

- Đa dạng hành động: Nhận dạng các hành động cử chỉ của tay phức tạp và đa dạng có thể là một thách thức. Các hành động có thể có nhiều biến thể và tương tự nhau, gây khó khăn trong việc phân loại chính xác.

- Dữ liệu đầu vào: Để xây dựng mô hình nhận dạng hành động, cần có một lượng lớn dữ liệu huấn luyện đại diện cho các hành động khác nhau. Việc thu thập, gắn nhãn và chuẩn bị dữ liệu đầu vào có thể tốn thời gian và công sức.

- Phụ thuộc vào đặc trưng: Các phương pháp nhận dạng cử chỉ của tay thường dựa vào việc trích xuất đặc trưng từ keypoint. Đôi khi, việc trích xuất đặc trưng không thể hiện đầy đủ thông tin về hành động và có thể gây mất mát thông tin quan trọng.

- Thời gian xử lý: Các phương pháp nhận dạng cử chỉ của tay có thể đòi hỏi thời gian xử lý tương đối lớn, đặc biệt khi áp dụng trong thời gian thực. Điều này có thể làm giảm khả năng ứng dụng trong các hệ thống yêu cầu phản hồi nhanh.

#### Một số mô hình nhận dạng hành động

- Mạng nơ-ron hồi quy (RNN - Recurrent Neural Network): RNN là một kiến trúc mạng nơ-ron được sử dụng để mô hình hóa dữ liệu chuỗi như chuỗi hành động. Nó có khả năng lưu trữ thông tin trạng thái trước đó và sử dụng thông tin đó để dự đoán các hành động tiếp theo trong chuỗi.

- Mạng nơ-ron tích chập (CNN - Convolutional Neural Network): CNN là một kiến trúc mạng nơ-ron được sử dụng rộng rãi trong xử lý ảnh và cũng có thể được áp dụng cho nhận dạng hành động. CNN có khả năng trích xuất các đặc trưng không gian từ ảnh và dùng chúng để phân loại các hành động.

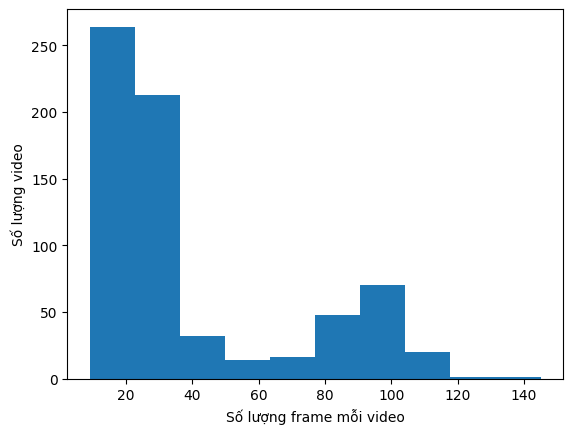
- Mạng nơ-ron tái phát (LSTM - Long Short-Term Memory): LSTM là một loại RNN đặc biệt được thiết kế để giải quyết vấn đề mất mát thông tin lâu dài khi xử lý dữ liệu chuỗi. Nó có khả năng ghi nhớ thông tin quan trọng từ quá khứ và sử dụng nó để dự đoán các hành động tiếp theo.

\* Trong báo cáo sử dụng mạng nơ ron tái phát LSTM

#### Tổng quan dữ liệu

Dữ liệu gồm 679 video của 14 hành động là ("Bye", "Eat", "Fine", "Good", "Hello", "I", "You", "Love", "None", "Read", "What", "Name", "Sleep", "Your").

Số lượng frame mỗi video:



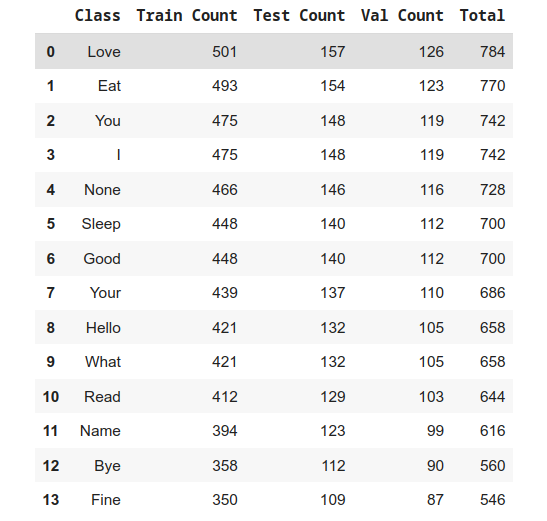
*Hình 3: Số lượng frame mỗi video*

Phần lớn video có số lượng frame từ 10 đến 20.

Tổng số dữ liệu sau khi làm giàu:

| **Hành động** | **Số lượng** |
| --- | --- |
| **Bye** | **560** |
| **Eat** | **770** |
| **Fine** | **546** |
| **Good** | **700** |
| **Hello** | **658** |
| **I** | **742** |
| **You** | **742** |
| **Love** | **784** |
| **None** | **728** |
| **Read** | **644** |
| **What** | **658** |
| **Name** | **616** |
| **Sleep** | **700** |
| **Your** | **686** |

Phân chia dữ liệu train/val/test:



#### Quá trình huấn luyện và trích xuất đặc trưng

##### Lần 1

**Dữ liệu**

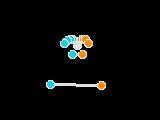
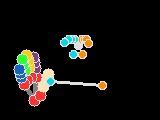
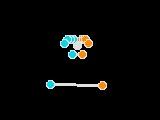
Gồm các hành động: ['Your', 'Love', 'Sleep', 'Name', 'What', 'Read', 'I', 'Bye', 'You', 'Eat', 'Hello']

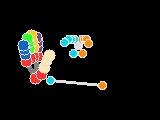
Mỗi hành động gồm 17 đến 20 video được quay bằng esp32 cam.

| Hành động | Số lượng |
| --- | --- |
| Your | 17 |
| Love | 15 |
| Sleep | 19 |
| Name | 20 |
| What | 18 |
| Read | 19 |
| I | 20 |
| You | 18 |
| Eat | 19 |
| Hello | 20 |

**Trích xuất đặc trưng:**

Dùng thư viện Mediapipe trích xuất các điểm pose, hand tạo thành 1 video chỉ gồm các điểm keypoint.







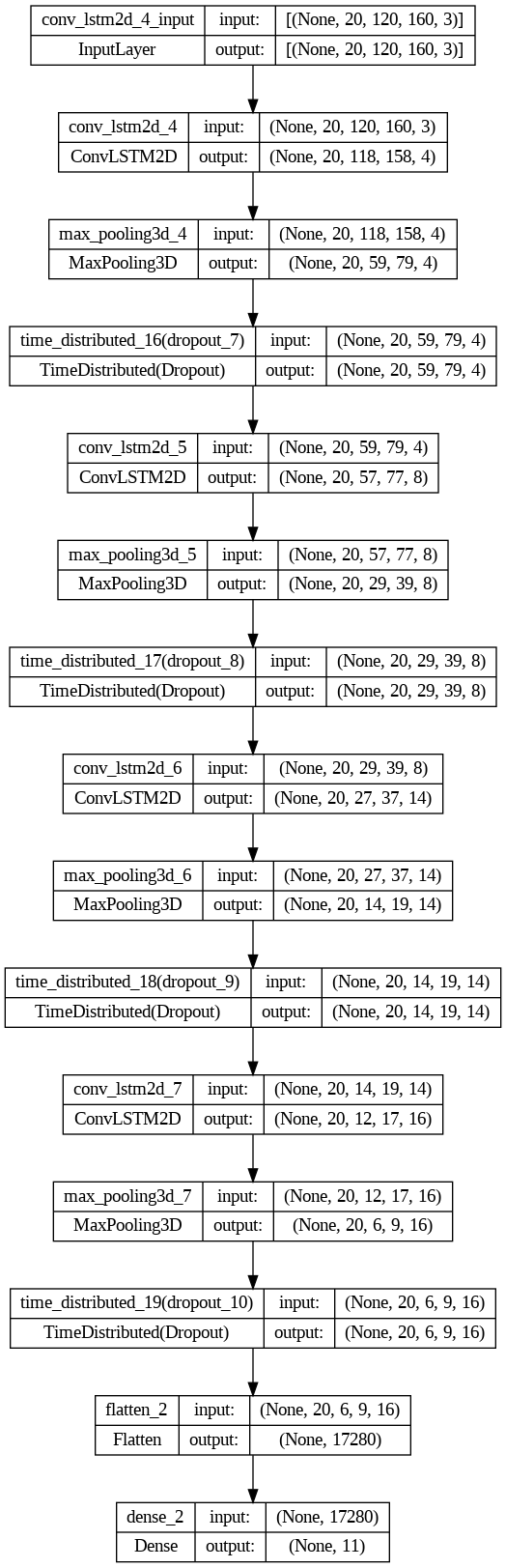




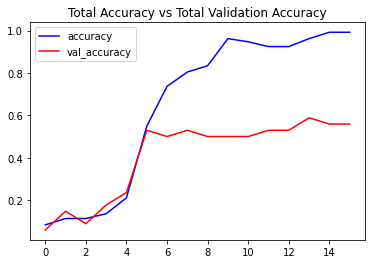
*Hình 4: Trích xuất đặc trưng*

**Train:**

Sử dụng ConvLSTM:

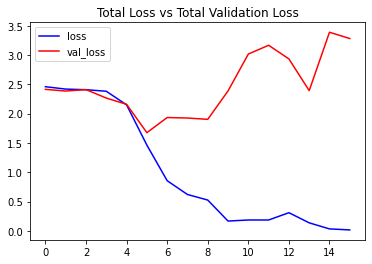


Accuracy:



*Hình 5: Total Accuracy and total Validation Accuracy*

Loss

:

*Hình 6: Total Loss and total validation Loss*

**Nhận xét:**

Độ chính xác trên tập validation khoảng 60% nhưng khi đưa vào inference để kiểm tra thực tế, mô hình nhận dạng với độ chính xác rất thấp.

- Nguyên nhân:

+ Với hình ảnh chỉ gồm các điểm keypoint, rất dễ bị nhầm lẫn giữa các hành động do không có thông tin về chiều sâu của vị trí cánh tay, ngón tay. Các hành động để thể hiện từ, "Tôi", "Bạn" rất dễ bị nhầm lẫn do vị trí tay khi trích xuất ra video chỉ có keypoint rất giống nhau.

+ Số lượng video cho mỗi hành động nhận dạng còn ít.

##### Lần 2:

**Dữ liệu:**

Gồm các hành động: ['Bye', 'Fine', 'Good', 'Hello', 'I', 'You', 'Love', 'None']

Không sử dụng những hành động dễ gây nhầm lẫn mơ hồ, thu thập thêm dữ liệu hành động mới.

Video được quay ở nhiều góc độ và độ xa khác nhau.

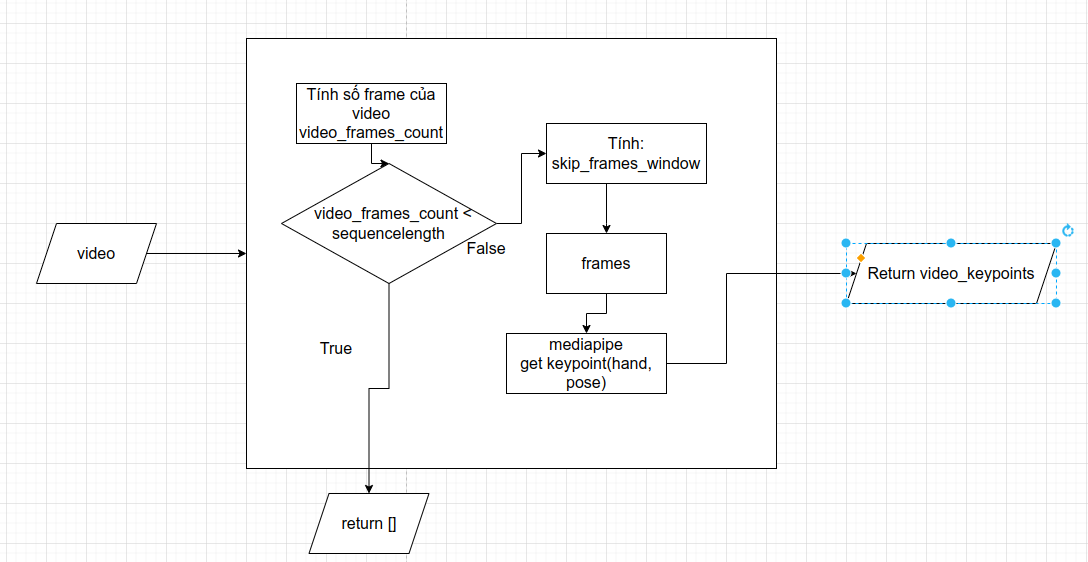


Mỗi hành động gồm 40 đến 53 video:

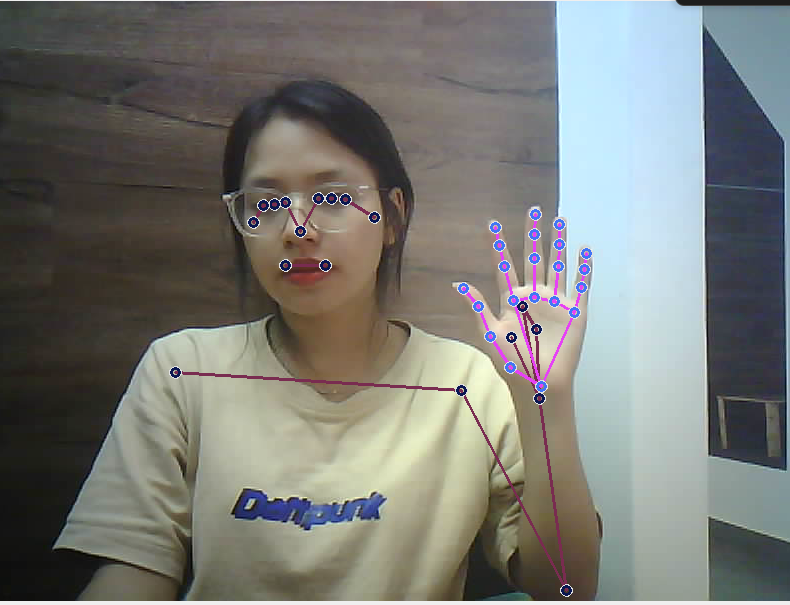
| Hành động | Số lượng |
| --- | --- |
| Bye | 40 |
| Fine | 41 |
| Good | 51 |
| Hello | 47 |
| I | 45 |
| You | 53 |
| Love | 43 |
| None | 52 |

**Trích xuất đặc trưng:**

**S**ử dụng thư viện Mediapipe trích xuất các keypoint, sao đó lưu các thông số keypoint đó vào ma trận, sử dụng để train.



*Hình 7: Sơ đồ trích xuất đặc trưng*



*Hình 8: Hình ảnh các đặc trưng được trích xuất*

Mỗi keypoint: gồm các thông số [x, y, z, visibility] của hand và pose

+ x, y, z: Đây là ba giá trị tọa độ để định vị vị trí của đối tượng trong không gian ba chiều. Giá trị x đại diện cho hoành độ, y đại diện cho tung độ và z đại diện cho hoành độ.

+ visibility: Thuộc tính này đo lường mức độ mà đối tượng có thể được nhìn thấy hoặc phát hiện trong một hệ thống nhận dạng hoặc theo dõi. Giá trị của visibility có thể là một số thực trong khoảng từ 0 đến 1, trong đó 0 có nghĩa là không nhìn thấy hoặc không phát hiện được, và 1 có nghĩa là hoàn toàn nhìn thấy hoặc phát hiện được

=> Tổng cộng 75 keypoint

=>Shape Input cho model: (10, 300)

**Phân chia dữ liệu train/val/test:**

Dùng StratifiedShuffleSplit của Sklearn, chia dữ liệu theo tỷ lệ train:val:test là 81:9:10

**Model sử dụng:**

Sử dụng kiến trúc LSTM (Long Short-Term Memory) trong thư viện Keras.

- Mô hình bao gồm các lớp tuần tự (Sequential) của các lớp LSTM và Dense.

+ Lớp đầu tiên là một lớp LSTM với 64 đơn vị LSTM, với tham số "return\_sequences=True" để trả về chuỗi kết quả từ LSTM và kích hoạt "relu" (hàm kích hoạt ReLU).

+ Lớp thứ hai là một lớp LSTM với 128 đơn vị LSTM, cũng có "return\_sequences=True" và kích hoạt "relu".

+ Lớp thứ ba là một lớp LSTM cuối cùng với 64 đơn vị LSTM, có "return\_sequences=False" để chỉ trả về kết quả cuối cùng từ LSTM và kích hoạt "relu".

+ Tiếp theo là ba lớp Dense, mỗi lớp Dense có kích thước đầu ra khác nhau và kích hoạt "relu".

+ Lớp cuối cùng là một lớp Dense với kích thước đầu ra bằng số lượng hành động có sẵn (actions.shape[0]), và kích hoạt "softmax" để tạo ra xác suất dự đoán cho các hành động.

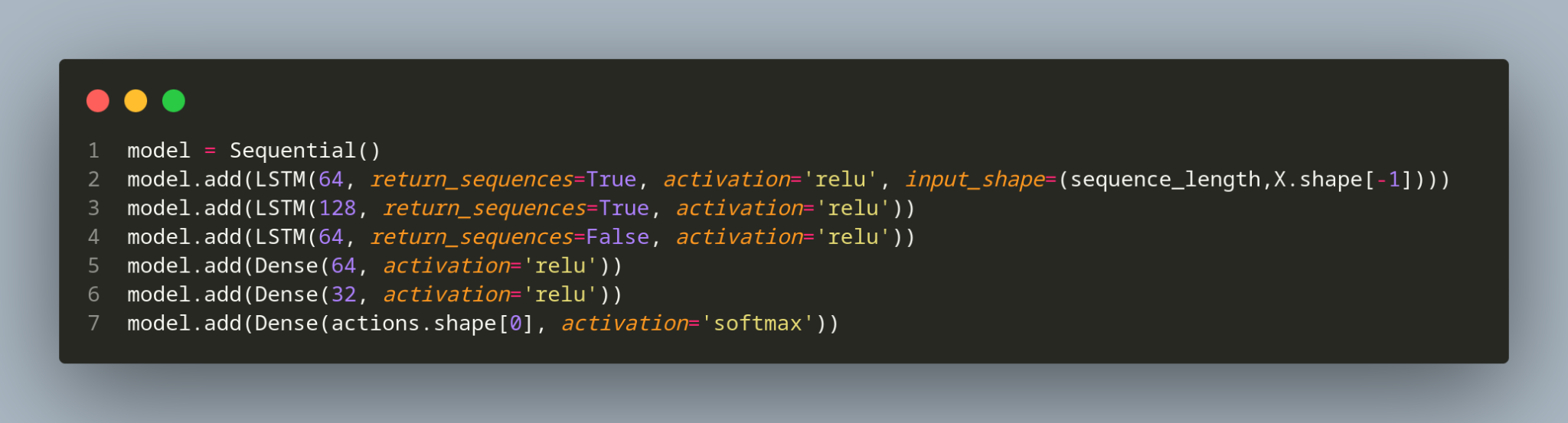
\*\* return\_sequences , kết quả trả về có dạng chuỗi hay không, nếu return\_sequences = True, thì kết quả trả về sẽ có kích thước là:

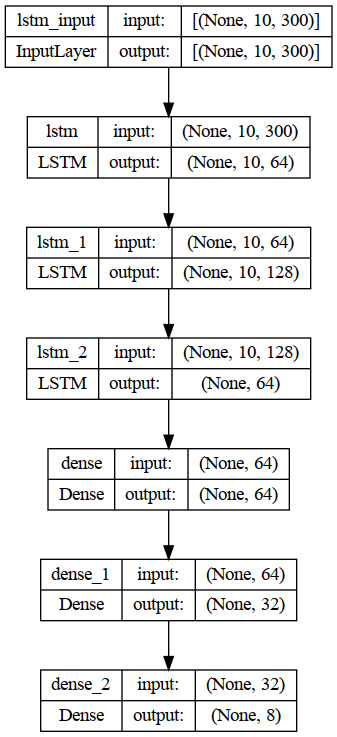
(batch\_size, sequence\_length, units), return\_sequences = False, thì kết quả trả về có kích thước là (batch\_size, units)

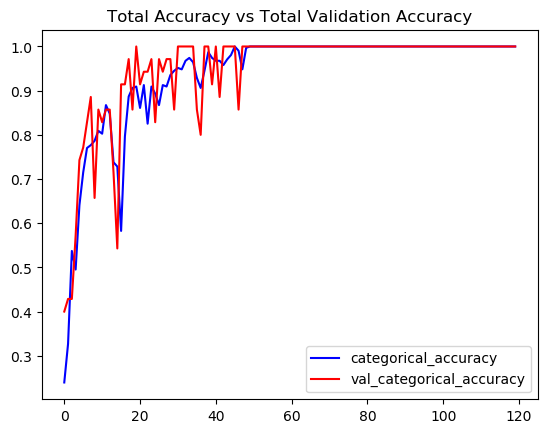
model.compile(*optimizer*=Adam(*learning\_rate*=0.001), *loss*='categorical\_crossentropy', *metrics*=['categorical\_accuracy'])

- Sử dụng hàm tối ưu: Adam với learning rate là 0,001

- Hàm loss là categorica\_crossentropy.

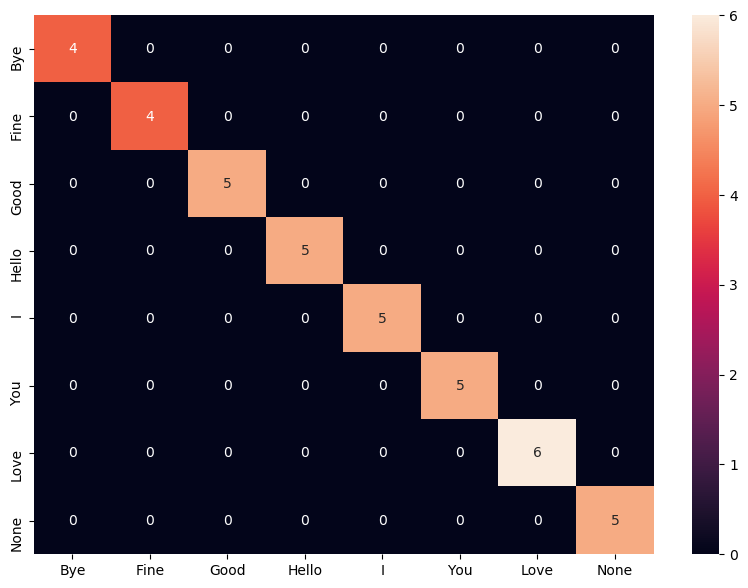








Ma trận nhầm lẫn:



*Hình 9: Ma trận nhầm lẫn*

**Nhận xét:** Độ chính xác đạt xấp xỉ 100%. Tuy nhiên khi dự đoán cho chuỗi hành động, đưa ra những kết quả không mong muốn. Ví dụ chưa thực hiện hành động nào vẫn đưa ra dự đoán, làm cho kết quả trả về không được chính xác. **Nguyên nhân:** dữ liệu không có sự mơ hồ giữa các hành động nên dự đoán chính xác với những hành động riêng lẻ nhưng với chuỗi hành động thì do lúc đọc đủ số frame cần thiết để dự đoán, nhưng vẫn chưa có hành động cụ thể dễ đưa ra dự đoán ngẫu nhiên, do gần như tất cả hành động vị trí tay ban đầu đều giống nhau.

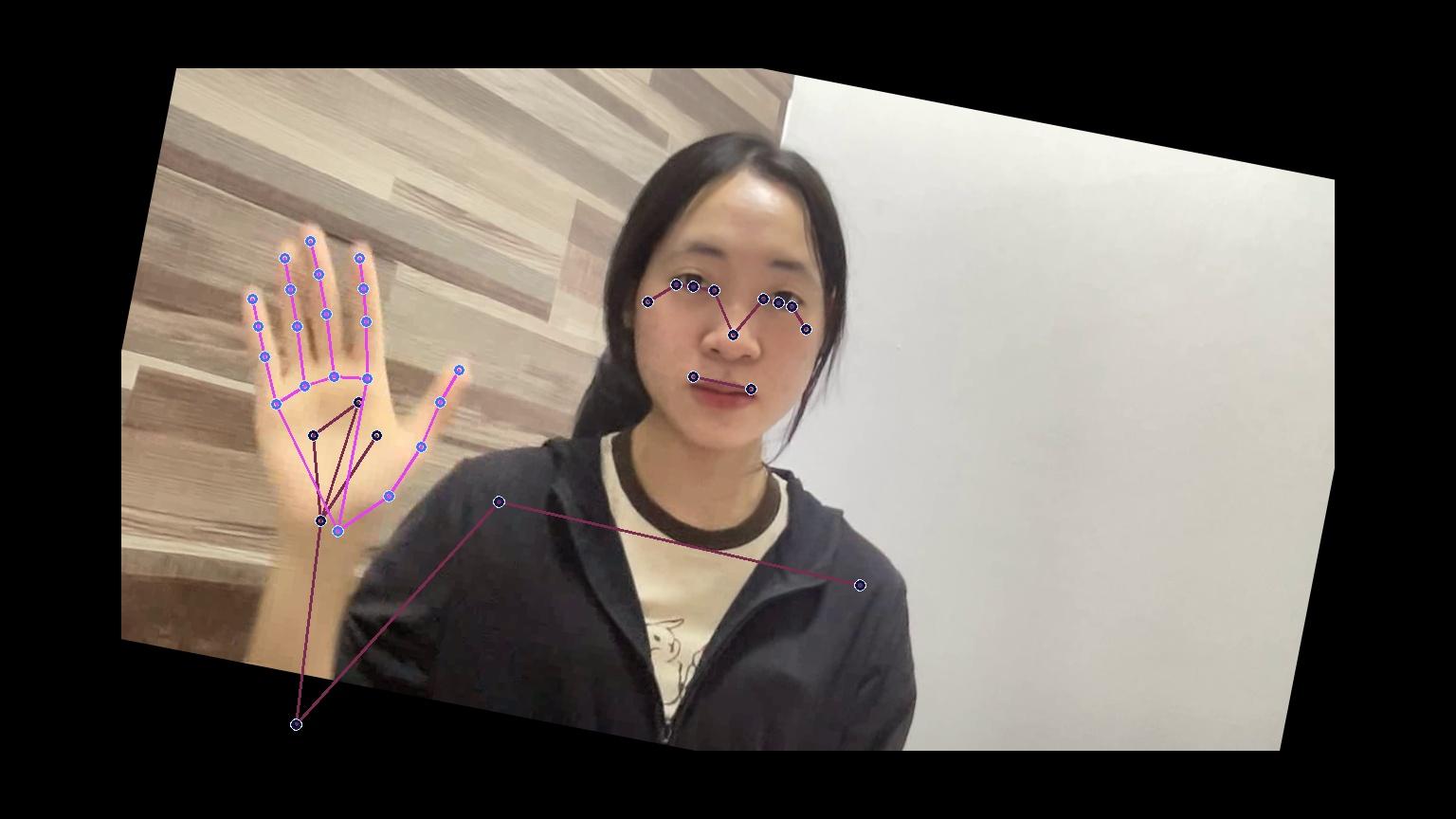
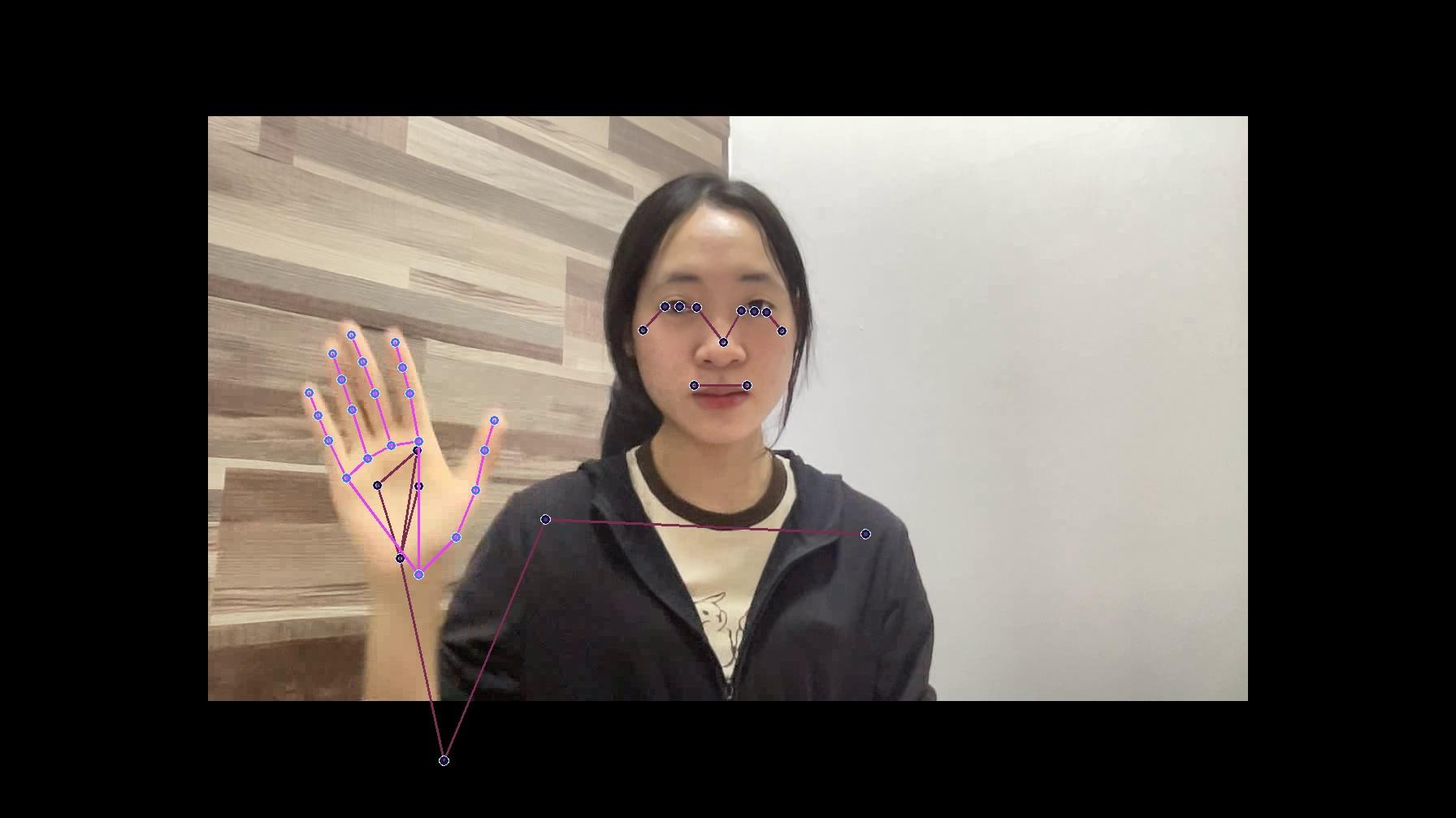
**Hạn chế:**  Các hành động trong dữ liệu sử dụng 1 tay cố định để tránh nhầm lẫn dẫn đến lúc đưa vào thực tế nếu sử dụng tay khác dễ đưa ra dự đoán sai.

##### Lần 3:

**Dữ liệu:**

Sử dụng tất cả các video đã thu thập, bao gồm các hành động dễ nhầm lần như Sleep và Eat.

Làm giàu dữ liệu bằng cách xoay, flip, và thêm padding vào cho video. Từ 1 video gốc, sau khi làm giàu ta có được 14 video khác nhau.

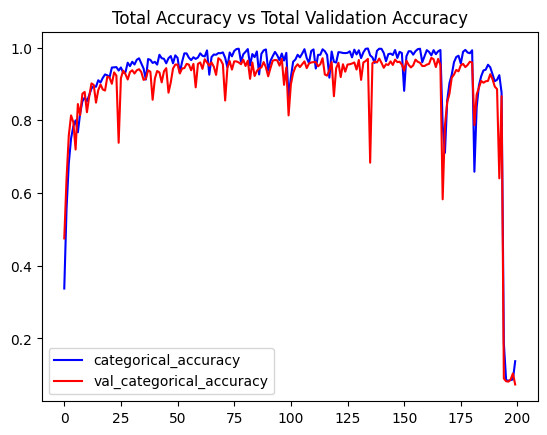
****

| **Hành động** | **Số lượng** |
| --- | --- |
| **Bye** | **560** |
| **Eat** | **770** |
| **Fine** | **546** |
| **Good** | **700** |
| **Hello** | **658** |
| **I** | **742** |
| **You** | **742** |
| **Love** | **784** |
| **None** | **728** |
| **Read** | **644** |
| **What** | **658** |
| **Name** | **616** |
| **Sleep** | **700** |
| **Your** | **686** |

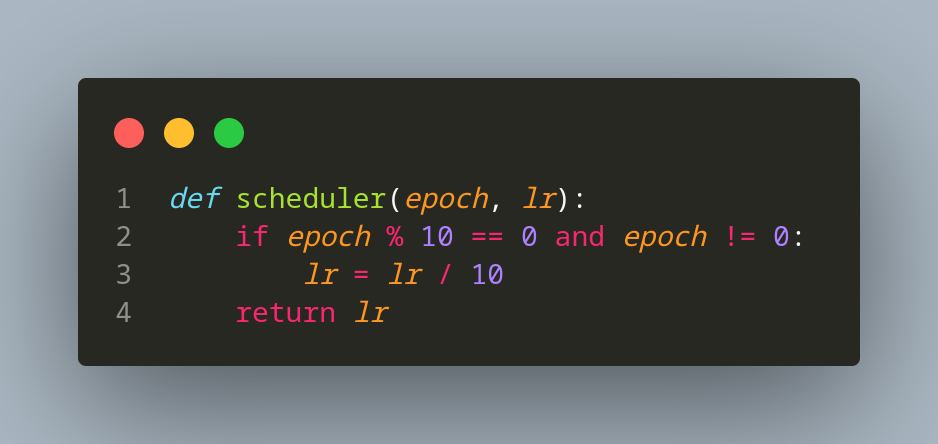
**Trích xuất đặc trưng:** Như lần train thứ 2

**Model sử dụng:** Như lần train thứ 2 như sử dụng thêm giá trị mặc định cho trọng số ban đầu(he\_normal) để tránh sự mất mát hoặc biến mất gradient trong quá trình huấn luyện, đồng thời tăng khả năng hội tụ và cải thiện hiệu suất của mô hình mạng neural.

Nếu không có khởi tạo giá trị ban đầu cho trọng số, có thể xảy ra các trường hợp thể này:



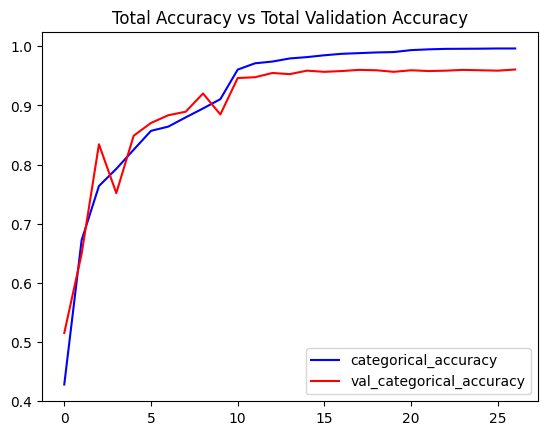
Ngoài ra còn sử dụng callback, giảm learning rate 10 lần sau mỗi 10 epoch giúp mô hình dừng lại hoặc di chuyển chậm hơn trong không gian tìm kiếm, nhằm tìm ra điểm cực tiểu toàn cục tốt hơn. Điều này có thể giúp cải thiện khả năng hội tụ và chất lượng của mô hình huấn luyện.



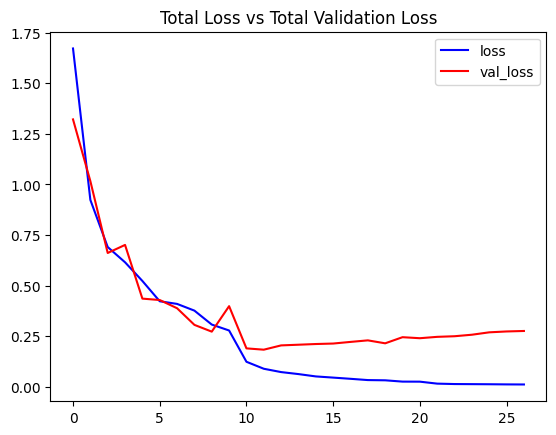


**Kết quả huấn luyện**

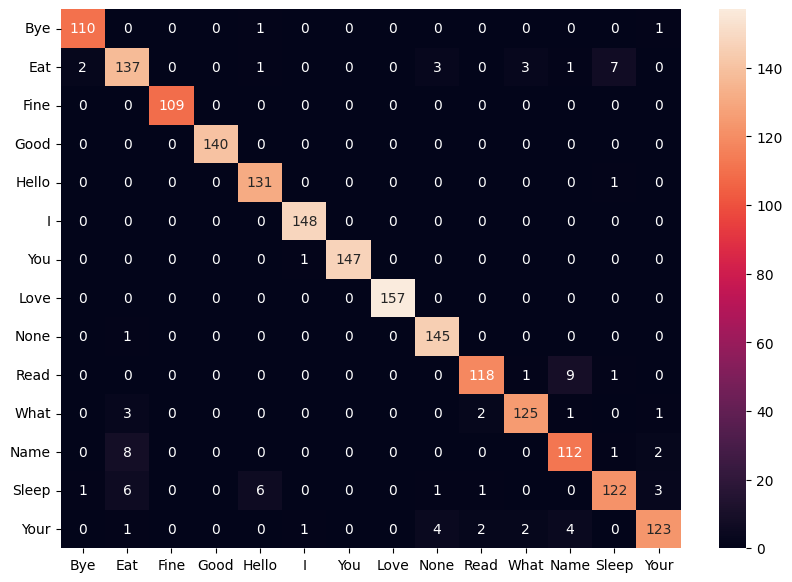
Độ chính xác:

****

Loss:

****

Ma trận nhầm lẫn ở tập test:



Độ chính xác ở tập test đạt 95.6%

**Nhận xét:**

Kết quả train và test đạt độ chính xác cao, nhưng độ chính xác khi đưa vào thực tế vẫn chưa cao, do các hành động có độ nhầm lẫn cao.

#### Viết mã nguồn

Bao gồm

- Đọc video

* Trích xuất đặc trưng và tạo file data.npz chứa dữ liệu train
* Đọc dữ liệu và train
* Tạo inference để test

### Server

Sử dụng Framework Flask để tạo các endpoint API. Gồm 2 route để xử lý các yêu cầu:

+ LSTM\_model (/LSTM): để xử lí các video của chuỗi hành động trả về chuỗi các từ ngữ nhận dạng được tương ứng.

+ LSTM\_model\_single(/LSTM/single): Để xử lí video gồm 1 hành động và trả về dự đoán của hành động trong video.

### Nguyên lý hoạt động của phần cứng

* Gửi video lên server: ESP32-CAM được sử dụng để ghi video từ camera và gửi nó lên server. Khi người dùng nhấn nút “Record/Stop” trên giao diện web, ESP32-CAM sẽ nhận yêu cầu HTTP(POST request) để bắt đầu hoặc dừng quá trình ghi video. Sau khi ghi xong, ESP32-CAM sẽ gửi file video lên server thông qua địa chỉ IP của DNS server.
* Tạo DNS server: ESP32-CAM sử dụng thư viện DNS server để tạo ra một DNS server. Điều này cho phép client sử dụng tên miền là địa chỉ IP của wifi để kết nối đến ESP32. Khi ESP32-CAM kết nối đến wifi, nó được cung cấp một địa chỉ IP và DNS server sẽ sử dụng địa chỉ này để tạo ra tên miền, cho phép client sử dụng tên miền là địa chỉ IP của wifi để kết nối đến ESP32
* Tạo trang HTML: ESP32-CAM tạo một trang HTML để tạo giao diện web cho người dùng, trang HTML này cho phép người dùng gửi yêu cầu HTTP đến server của ESP32-CAM, chẳng hạn như yêu cầu ghi video. Giao diện web này có thể chứa các nút và các phần tử tương tác khác để điều khiển ESP32-CAM và thực hiện các chức năng khác khi video và dừng ghi.

### Nguyên lý hoạt động hệ thống nhận dạng ngôn ngữ ký hiệu

* Thu thập dữ liệu: Đầu tiên, cần thu thập dữ liệu bằng cách ghi lại video của từng hành động riêng lẻ mà hệ thống muốn nhận dạng. Các video này sẽ được dùng để nhận dạng mô hình.
* Xử lý và trích xuất đặc trưng: Sau khi thu thập video, các video này sẽ được xử lý để trích xuất đặc trưng từ video.
* Huấn luyện mô hình: Các đặc trưng trích xuất từ video được sử dụng để huấn luyện mô hình nhận dạng chuỗi hành động. Quá trình huấn luyện này dựa trên dữ liệu đã thu thập từ các video và mục tiêu là để mô hình học cách phân loại các hành động khác nhau.
* Nhận dạng chuỗi hành động: Khi có một video chứa một hoặc nhiều hành động, quá trình nhận dạng hành động sẽ được áp dụng. Đầu tiên video sẽ được đọc và xử lý frame từng frame. Từ mỗi frame, các keypoint và đặc trưng liên quan sẽ được trích xuất. Sau đó, danh sách keypoint sẽ được đưa vào mô hình đã được huấn luyện để dự đoán hành động tương ứng. Kết quả dự đoán sẽ được kiểm tra và xác định xem có đạt độ chính xác mong muốn hay không. Khi một hành động được dự đoán đạt độ chính xác cao, nó sẽ được đưa vào danh sách kết quả. Quá trình này sẽ được lặp lại cho đến khi xử lý hết tất cả các frame trong video.

Cuối cùng, sau khi quá trình nhận dạng hoàn tất, danh sách kết quả chứa các hành động được nhận dạng trong video hiển thị trên màn hình web.

### Các linh kiện sử dụng

- ESP32 cam

### Tham số kỹ thuật đầu vào-ra và nguyên tắc hoạt động

ESP32 cam:

* Tham số kỹ thuật đầu vào:
* Khởi tạo Serial communication với tốc độ baud rate là 115200.
* Thiết lập chế độ debug output cho Serial.
* In ra các thông báo và thông tin về phiên bản code, phiên bản cơ bản và tên của ESP32-CAM.
* Kiểm tra xem có PSRAM (External Pseudo Static RAM) được phát hiện hay không. Nếu không có, thông báo lỗi và dừng chương trình.
* Kiểm tra xem có cấu hình chi tiết wifi hay không. Nếu không có, thông báo lỗi và dừng chương trình.
* Nếu có đèn LED (LED\_PIN được định nghĩa), thiết lập chế độ OUTPUT cho chân đèn LED và bật đèn LED.
* Khởi động hệ thống tệp SPIFFS (SPI Flash File System) để chuẩn bị cho việc sử dụng hệ thống tệp.
* Khởi động và cấu hình camera sử dụng hàm StartCamera().
* Nếu có hệ thống tệp SPIFFS, tải và áp dụng các thiết lập đã được lưu trữ trước đó.
* Tiếp tục khởi tạo phần cứng khác (nếu có).
* Nguyên tắc hoạt động:
* Kết nối với mạng Wifi hoặc tạo điểm truy cập riêng nếu chưa kết nối.
* Nếu OTA (Over-the-Air) được kích hoạt, cấu hình và khởi động OTA (cập nhật qua mạng không dây) để có thể cập nhật firmware từ xa.
* Nếu OTA không được kích hoạt, khởi tạo MDNS (Multicast DNS) để có thể truy cập thông qua tên miền local.
* Thêm dịch vụ HTTP vào máy chủ MDNS.
* Nếu có cấu hình, thiết lập thời gian từ máy chủ NTP (Network Time Protocol) để đồng bộ thời gian.
* Khởi tạo và thiết lập đèn (lamp) nếu có.
* Khởi động máy chủ camera để cung cấp giao diện web và luồng video trực tiếp.
* In ra các URL và thông điệp hoạt động của ESP32-CAM.
* Kiểm tra và xóa bộ đệm nhận dữ liệu trên cổng Serial để loại bỏ các dữ liệu không hợp lệ nhận được trong quá trình

### Bảng kê chi phí linh kiện

| Linh kiện | Chi phí |
| --- | --- |
| ESP32 cam | 160.000 VND |

## Kết quả

#### Dữ liệu đã sử dụng

| **Hành động** | **Số lượng** |
| --- | --- |
| **Bye** | **560** |
| **Eat** | **770** |
| **Fine** | **546** |
| **Good** | **700** |
| **Hello** | **658** |
| **I** | **742** |
| **You** | **742** |
| **Love** | **784** |
| **None** | **728** |
| **Read** | **644** |
| **What** | **658** |
| **Name** | **616** |
| **Sleep** | **700** |
| **Your** | **686** |

#### Các công cụ và framework đã dùng

* Ngôn ngữ Python, Javascript, C
* Frontend: HTML, CSS, Javascript
* Backend: Python
* Xử lý ảnh số – mã nguồn mở OpenCV.
* Phần cứng Camera, camera esp32.

#### Các điều kiện tiến hành thực nghiệm

* Điều kiện ánh sáng tốt, camera không đặt quá xa và quay hướng phù hợp để thu được hình ảnh hành động chính diện

#### Các độ đo (metrics) cụ thể dùng để đánh giá hiệu suất của giải pháp

* Độ chính xác: 85-95%
* Tốc độ thực thi: 100 ms
* Độ ổn định: Cao
* Tính bảo mật: Cao
* Khả năng mở rộng: Dễ dàng

#### Quy trình và điều kiện kiểm thử hệ thống, và các kết quả đánh giá hệ thống theo các tiêu chí (đã nêu)

#### Quy trình

* B1: Click Single để bắt đầu quay video hành động, sau khi click, Single button sẽ chuyển thành Stop button có chức năng dừng ghi hình và bắt đầu gửi video hành động lên server xử lý.
* B2: Chờ và nhận kết quả ngay dưới màn hình quay video.

#### Kết quả

Giao diện quay và nhân diện hành động:



*Giao diện người dùng*

### 

#### Đánh giá

* Trong môi trường ánh sáng tốt, hướng mặt chính diện(0o), không bị che khuất thì tỷ lệ phát hiện khuôn mặt cao hơn. Vì vậy trong môi trường bình thường nếu điều kiện không tốt như ngước sáng hay góc mặt bị che khuất hay không chính diện tỷ lệ phát hiện bị giảm đáng kể.
* Cơ sở dữ liệu là ảnh được training lúc đăng nhập lần đầu tiên, training càng nhiều tỷ lệ phát hiện và độ chính xác khi nhận diện càng cao.
* Sau khi giảng viên upload ảnh có khuôn mặt sinh viên trong lớp thì dữ liệu đó sẽ được so sánh với dữ liệu có sẵn trong Database, nếu trùng khớp thì điểm danh hoàn tất.

## 

## Kết luận

### Đánh giá

* Ưu điểm:
  + Cung cấp các chức năng cơ bản nhất theo yêu cầu người dùng

Hoạt động tốt, xử lý tương đối nhanh.

* Giao diện thân thiện và dễ sử dụng
* Nhận dạng chính xác những video có 1 hành động

- Khuyết điểm:

+ UI/UX còn khá đơn điệu.

+ Những video chuỗi hành động, nhận dạng chưa được chính xác

### Kết luận

* Hệ thống nhận diện ngôn ngữ ký hiệu hoạt động tương đối hiệu quả và ổn định. Tuy nhiên độ chính xác khi nhận diện động còn chưa cao nếu gặp các điều kiện bất lợi như mặt không chính diện, thiếu sáng, …
* Hiệu quả của hệ thống ngoài phụ thuộc những yếu tố về chuyên môn còn bị ảnh hưởng bởi điều kiện môi trường.

#### 

### Hướng phát triển

* Cải thiện độ chính xác của thuật toán để nhận diện nhanh và hiệu quả hơn.
* Cải thiện dữ liệu để nhận diện được ngay cả khi hành động bị khuất hay điều kiện môi trường không thích hợp.
* Cải thiện chương trình để có thể nhận diện được chuỗi hành động liên tục với độ chính xác cao.
* Nâng cấp giao diện đẹp, thân thiện hơn với người dùng.
* Hoàn thiện sản phẩm tiến tới phiên bản dùng thử và phát triển thành một sản phẩm hoàn thiện có tính ứng dụng cao, được sử dụng trong thực tế.

# TÀI LIỆU THAM KHẢO

- Freecodecamp-How to use transfer learning for sign language <<https://www.freecodecamp.org/news/asl-recognition-using-transfer-learning-918ba054c004/>>

- MediaPipe Solutions guide. <<https://developers.google.com/mediapipe/solutions/guide>>

- Taha Anwar ( BleedAI.com, 2021)-Introduction to Video Classification and Human Activity Recognition <<https://learnopencv.com/introduction-to-video-classification-and-human-activity-recognition/>>

- How to use LSTM <<https://keras.io/api/layers/recurrent_layers/lstm/>>