TRƯỜNG ĐẠI HỌC QUẢNG BÌNH

**KHOA KỸ THUẬT - CÔNG NGHỆ THÔNG TIN**

#### TÊN ĐỀ TÀI

#### NHẬN DẠNG PHÁT HIỆN HÀNH ĐỘNG CHO

#### NGÔN NGỮ KÝ HIỆU

Giảng viên hướng dẫn: TS. Hoàng Văn Thành

Họ và tên sinh viên: Hoàng Văn Thắng

Mã sinh viên: 19D115019

Lớp: ĐH Công nghệ thông tin K61

**Quảng Bình,ngày 12 tháng 3 năm 2022**

Giảng viên hướng dẫn: TS. Hoàng Văn Thành

Họ và tên sinh viên: Hoàng Văn Thắng

Lớp: ĐH Công nghệ thông tin K61

**Quảng Bình,tháng 12 năm 2021**

**NHẬN XÉT VÀ CHO ĐIỂM CỦA GIẢNG VIÊN**

|  |  |
| --- | --- |
| **Cán bộ chấm thi 1** | **Cán bộ chấm thi 2** |
|  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Điểm bằng số** |  | **Điểm bằng chữ** |  | **Chữ ký CBChT 1**  *(Ký và ghi rõ họ tên)*  …………………….……………………. |  | **Chữ ký CBChT 2**  *(Ký và ghi rõ họ tên)*  …………………….……………………. |

**MỤC LỤC**

[LỜI MỞ ĐẦU 1](#_Toc101575811)

[PHẦN MỞ ĐẦU 2](#_Toc101575812)

[1. Lý do chọn đề tài 2](#_Toc101575813)

[2. Mục tiêu nghiên cứu 2](#_Toc101575814)

[3. Đối tượng nghiên cứu 2](#_Toc101575815)

[4. Phương pháp nghiên cứu 2](#_Toc101575816)

[PHẦN NỘI DUNG 4](#_Toc101575817)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 4](#_Toc101575818)

[1.1. Ngôn ngữ lập trình Python 4](#_Toc101575819)

[1.1.1. Khái niệm 4](#_Toc101575820)

[1.1.2. Tính năng chính của Python 4](#_Toc101575821)

[1.1.3. Lý do học ngôn ngữ Python 5](#_Toc101575822)

[1.2. Thư viện OpenCV 6](#_Toc101575823)

[1.2.1. Thư viện OpenCV là gì? 6](#_Toc101575824)

[1.2.2. Các tính năng của OpenCV 6](#_Toc101575825)

[1.2.3. Ứng dụng của thư viện OpenCV 6](#_Toc101575826)

[1.2.4. Các module phổ biến của OpenCV 7](#_Toc101575827)

[1.3. Jupyter notebook 8](#_Toc101575828)

[1.3.1. Khái niệm 8](#_Toc101575829)

[1.3.2. Ưu điểm Jupyter notebook 8](#_Toc101575830)

[1.3.3. Cách cài đặt Jupyter notebook 8](#_Toc101575831)

[1.4. Train LSTM Neural Network 8](#_Toc101575832)

[1.4.1. Khái niệm 8](#_Toc101575833)

[1.4.2. Mô hình LSTM 9](#_Toc101575834)

[1.5. MediaPipe 9](#_Toc101575835)

[1.5.1. Khái niệm 9](#_Toc101575836)

[1.5.2. Ưu điểm của MediaPipe 9](#_Toc101575837)

[1.5.3. Các giải pháp trong MediaPipe 10](#_Toc101575838)

[1.5.3.1. Face Detection 10](#_Toc101575839)

[1.5.3.2. Face Mesh 10](#_Toc101575840)

[1.5.3.3. Hands Detection 10](#_Toc101575841)

[1.5.3.4. Human Pose Estimation 10](#_Toc101575842)

[CHƯƠNG 2. NHẬN DẠNG VÀ XỬ LÝ ẢNH 12](#_Toc101575843)

[2.1. Quá trình nhận dạng 12](#_Toc101575844)

[2.1.1. Tính chất và sự cần thiết của bài toán nhận dạng 12](#_Toc101575845)

[2.1.1.1. Tính chất 12](#_Toc101575846)

[2.1.1.2. Sự cần thiết 12](#_Toc101575847)

[2.1.2. Không gian biểu diễn đối tương , không gian miễn dịch. 12](#_Toc101575848)

[2.1.2.1. Không gian biểu diễn đối tương 12](#_Toc101575849)

[2.1.2.2. Không gian miễn dịch. 13](#_Toc101575850)

[2.1.3. Mô hình và bản chất của quá trình nhận dạng 13](#_Toc101575851)

[2.1.3.1. Mô hình 13](#_Toc101575852)

[2.1.3.2. Bản chất của quá trình nhận dạng 14](#_Toc101575853)

[2.2. Xử lý ảnh 15](#_Toc101575854)

[2.2.1. Tổng quan 15](#_Toc101575855)

[2.2.2. Quá trình xử lý ảnh 15](#_Toc101575861)

[2.2.3. Phạm vi ứng dụng của xử lý ảnh 16](#_Toc101575868)

[CHƯƠNG 3. NHẬN DẠNG PHÁT HIỆN HÀNH ĐỘNG CHO NGÔN NGỮ KÝ HIỆU 17](#_Toc101575870)

[3.1. Import and Install Dependencies 17](#_Toc101575871)

[3.2. Lấy những điểm chính 17](#_Toc101575872)

[3.3. Trích xuất các giá trị điểm 20](#_Toc101575873)

[3.4. Tạo thư mục cho dữ liệu 20](#_Toc101575874)

[3.5. Thu thập các giá trị để huấn luyện và kiểm tra 21](#_Toc101575875)

[3.6. Tiền xử lý dữ liệu và tạo nhãn và các đặc điểm 23](#_Toc101575876)

[3.7. Build and Train LSTM Neural Network 24](#_Toc101575877)

[3.8. Đưa ra dự đoán, lưu và load model 25](#_Toc101575878)

[3.9. Đánh giá ma trận nhầm lẫn và độ chính xác 25](#_Toc101575879)

[3.10. Chạy chương trình 26](#_Toc101575880)

[3.11. Kết quả chương trình 26](#_Toc101575881)

[PHẦN KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 29](#_Toc101575883)

[1. Đánh giá kết quả đạt được 29](#_Toc101575884)

[2. Hạn chế 29](#_Toc101575885)

[TÀI LIỆU THAM KHẢO 30](#_Toc101575886)

**DANH MỤC HÌNH ẢNH**

[Hình 1: Mô hình LSTM 9](#_Toc101576089)

[Hình 2: Các giai đoạn chính trong xử lý ảnh 15](#_Toc101576090)

[Hình 3: Mẫu thử 1 27](#_Toc101576091)

[Hình 4: Mẫu thử 2 27](#_Toc101576092)

[Hình 5: Mẫu thử 3 28](#_Toc101576093)

[Hình 6: Mẫu thử 4 28](#_Toc101576094)

LỜI MỞ ĐẦU

Xử lý và nhận dạng là một lĩnh vực từ lâu được nhiều người quan tâm. Nó đã được ứng dụng vào nhiều lĩnh vực như:

Trong y học, cải thiện ảnh X-quang và nhận dạng đường biên mạch máu từ ảnh chụp bằng tia X , ứng dụng vào các xét nghiệm lâm sang như phát hiện và nhận dạng u não, nội soi cắt lớp…

Trong thiên văn học, hệ thống chụp hình gắn trên tàu vũ trụ hạn chế về kích thước và trọng lượng, do đó chất lượng hình ảnh nhận được bị giảm chất lượng như bị mờ, méo hình học và nhiễu nền. Các hình ảnh đó được xử lý bằng máy tính.

Trong các lĩnh vực công nghiệp, người máy ngày càng đóng vai trò quan trọng. Chúng thực hiện các công việc nguy hiểm, đòi hỏi có tốc độ và độ chính xác cao vượt quá khả năng con người. Người máy sẽ trở nên tinh vi hơn và thị giác máy tính đóng vai trò quan trọng hơn. Người ta sẽ không chỉ đòi hỏi người máy phát hiện và nhận dạng các bộ phận công nghiệp mà còn phải “ hiểu ” được những gì chúng “ thấy ” và đưa ra hành động phù hợp. Xử lý ảnh sẽ tác động đến thị giác của máy tính.

Ngoài ra, xử lý và nhận dạng còn được ứng dụng trong lĩnh vực khác ít được nói đến hơn. Do khả năng nghe bị suy giảm nên khả năng giao tiếp bằng lời ở cộng đồng người khiếm thính bị hạn chế rất nhiều. Để thay thế cho khả năng giao tiếp bằng tiếng nói, ngôn ngữ ký hiệu, là ngôn ngữ sử dụng biểu diễn của bàn tay và cơ thể đã ra đời.

Trong bài tiểu luận này chúng tôi xin trình bày một ứng dụng của xử lý và nhận dạng: Phát hiện hành động cho ngôn ngữ ký hiệu.

# MỞ ĐẦU

1. Lý do chọn đề tài

Theo số liệu thống kê của WHO năm 2000, trên thế giới có khoảng 250 triệu người khiếm thính, chiếm 4,2 % dân số. Do khả năng nghe bị suy giảm nên khả năng giao tiếp bằng lời ở cộng đồng người khiếm thính bị hạn chế rất nhiều. Để thay thế cho khả năng giao tiếp bằng tiếng nói, ngôn ngữ ký hiệu, là ngôn ngữ sử dụng biểu diễn của bàn tay và cơ thể đã ra đời.

Trên thế giới, đã có nhiều nghiên cứu phát triển các dịch vụ thông dịch ngôn ngữ ký hiệu và các sản phẩm công nghệ nhằm hỗ trợ người khiếm thính trong giao tiếp xã hội. Hiện nay theo thống kê, Việt Nam có khoảng trên 2.5 triệu người khiếm thính. Với sự quan tâm đặc biệt của Đảng và Nhà nước, đã có nhiều trường học, trung tâm hỗ trợ dạy học và việc làm riêng cho người khiếm thính.

Việc phát triển sản phẩm ứng dụng công nghệ để phát huy ngôn ngữ ký hiệu nhằm nâng cao trình độ, tiếp nhận thông tin, khả năng giao tiếp cho người khiếm thính ở Việt Nam rất ít và kém hiệu quả. Vì vậy việc đề xuất một giải pháp để xây dựng một hệ thống nhận dạng ngôn ngữ kí hiệu Việt Nam là cần thiết.

Vì vậy, tôi chọn đề tài nghiên cứu “Nhận dạng phát hiện hành động cho ngôn ngữ ký hiệu” để hỗ trợ cho những người khiếm thính.

2. Mục tiêu nghiên cứu

+ Tìm hiểu về ngôn ngữ lập trình Python .

+ Tìm hiểu quá trình nhận dạng và xử lý ảnh.

+ Tìm hiểu về thư viện OpenCv qua đó ứng dụng nó trong nhận dạng phát hiện hành động cho ngôn ngữ ký hiệu một cách tương đối chính xác

3. Đối tượng nghiên cứu

Đối tượng nghiên cứu:

+ Thư viện Opencv và quá trình nhận dạng xử lý ảnh.

+ Ngôn ngữ lập trình Python

Phạm vi nghiên cứu:

+ Ứng dụng quá trình nhận dạng và xử lý ảnh để nhận dạng được đúng các hành động cho ngôn ngữ ký hiệu.

4. Phương pháp nghiên cứu

Phương pháp tham khảo tài liệu

+ Thu thập, đọc hiểu, phân tích thông tin, dữ liệu từ các tài liệu, giáo trình, sách liên quan đến nhận dạng xử lý ảnh .

+ Xem xét, lựa chọn phương pháp để giải quyết vấn đề.

+ Các tài liệu liên quan đến ngôn ngữ ký hiệu.

Phương pháp thực hành:

+ Tìm hiểu giải thuật nhận dạng và xử lý ảnh, sử dụng thư viện Opencv trong ngôn ngữ lập trình Python để lập trình xử lý hành động cho ngôn ngữ ký hiệu

# NỘI DUNG

## CƠ SỞ LÝ THUYẾT

### Ngôn ngữ lập trình Python

#### Khái niệm

Python[1]là một ngôn ngữ lập trình đa mục đích được tạo ra vào cuối những năm 1980 . Đây là một ngôn ngữ co mã nguồn mở với đặc điểm có cú pháp dễ đọc, dễ hiểu. Ngày nay Python trở thành một trong những ngôn ngữ lập trình phổ biến nhất thế giới bên cạnh C#, Java…. Đặc biệt trong lĩnh vực AI và Machine Learning thì Python càng thể hiện được tính ưu việt của nó.

#### Tính năng chính của Python

* Ngôn ngữ lập trình đơn giản, dễ học: Python có cú pháp rất đơn giản, rõ ràng. Nó dễ đọc và viết hơn rất nhiều khi so sánh với những ngôn ngữ lập trình khác như C++, Java, C#. Python làm cho việc lập trình trở nên thú vị, cho phép bạn tập trung vào những giải pháp chứ không phải cú pháp.
* Miễn phí, mã nguồn mở: Bạn có thể tự do sử dụng và phân phối Python, thậm chí là dùng cho mục đích thương mại. Vì là mã nguồn mở, bạn không những có thể sử dụng các phần mềm, chương trình được viết trong Python mà còn có thể thay đổi mã nguồn của nó. Python có một cộng đồng rộng lớn, không ngừng cải thiện nó mỗi lần cập nhật.
* Khả năng di chuyển: Các chương trình Python có thể di chuyển từ nền tảng này sang nền tảng khác và chạy nó mà không có bất kỳ thay đổi nào. Nó chạy liền mạch trên hầu hết tất cả các nền tảng như Windows, macOS, Linux.
* Khả năng mở rộng và có thể nhúng: Giả sử một ứng dụng đòi hỏi sự phức tạp rất lớn, bạn có thể dễ dàng kết hợp các phần code bằng C, C++ và những ngôn ngữ khác (có thể gọi được từ C) vào code Python. Điều này sẽ cung cấp cho ứng dụng của bạn những tính năng tốt hơn cũng như khả năng scripting mà những ngôn ngữ lập trình khác khó có thể làm được.
* Ngôn ngữ thông dịch cấp cao: Không giống như C/C++, với Python, bạn không phải lo lắng những nhiệm vụ khó khăn như quản lý bộ nhớ, dọn dẹp những dữ liệu vô nghĩa. Khi chạy code Python, nó sẽ tự động chuyển đổi code sang ngôn ngữ máy tính có thể hiểu. Bạn không cần lo lắng về bất kỳ hoạt động ở cấp thấp nào.
* Thư viện tiêu chuẩn lớn để giải quyết những tác vụ phổ biến: Python có một số lượng lớn thư viện tiêu chuẩn giúp cho công việc lập trình của bạn trở nên dễ thở hơn rất nhiều, đơn giản vì không phải tự viết tất cả code. Ví dụ: Bạn cần kết nối cơ sở dữ liệu MySQL trên Web server? Bạn có thể nhập thư viện MySQLdb và sử dụng nó. Những thư viện này được kiểm tra kỹ lưỡng và được sử dụng bởi hàng trăm người. Vì vậy, bạn có thể chắc chắn rằng nó sẽ không làm hỏng code hay ứng dụng của mình.
* Hướng đối tượng: Mọi thứ trong Python đều là hướng đối tượng. Lập trình hướng đối tượng (OOP) giúp giải quyết những vấn đề phức tạp một cách trực quan. Với OOP, bạn có thể phân chia những vấn đề phức tạp thành những tập nhỏ hơn bằng cách tạo ra các đối tượng.

#### Lý do học ngôn ngữ Python

* **Là ngôn ngữ lập trình phổ biến nhất:**

Python được coi là ngôn ngữ lập trình phổ biến hiện nay. Được xuất hiện vào đầu năm 199, trải qua sự phát triển và trưởng thành, Python đã trở thành ngôn ngữ lập trình cực kỳ thân thiện với người dùng. Hàng ngàn người đã sử dụng chúng trong kiểm thử vi mạch, trong các ứng dụng điện thoại, xây dựng các video game...

* **Rất nhiều "anh tài" sử dụng Python**

Có thể nói, hiếm có ngôn ngữ lập trình nào như Python được sử dụng bởi nhiều các tên tuổi lớn như vậy. Cụ thể, các thương hiệu sử dụng Python trong quá trình phát triển và thử nghiệm như: Uber, Google, Paypal, Facebook, Instagram, Netflix, Dropbox…

* **Sử dụng linh hoạt trong nhiều mảng lập trình**

Không chỉ nổi tiếng với mảng lập trình web, Python còn là ngôn ngữ được sử dụng rộng rãi trong lập trình robot và các hệ thống nhúng.

* **Là bước đệm để học thêm các ngôn ngữ lập trình khác**

Python thân thiện với người dùng và hòa đồng với cả những ngôn ngữ lập trình khác. Như bạn đã biết, Python là ngôn ngữ hướng đối tượng có ứng dụng đa dạng. Và nếu bạn nắm rõ Python thì sẽ dễ dàng hơn khi học thêm các ngôn ngữ lập trình khác.

* **Python được hỗ trợ rất tốt**

Với sự phổ biến của mình nên Python sở hữu cộng đồng hỗ trợ trực tuyến cực tốt. Có rất nhiều các trang web hỗ trợ người dùng Python cực tốt như: The Official Python Tutorial, Stack Overflow...

* **Cơ hội nghề nghiệp rộng mở với mức lương cao**

Nếu thành thạo Python thì các lập trình viên hoàn toàn có thể sở hữu những công việc cực tốt với mức lương siêu khủng. Và nếu có vốn ngoại ngữ tốt thì việc làm việc tại nước ngoài hay các công ty liên doanh, công ty nước ngoài là điều cực kỳ bình thường.

* **Python hoàn toàn miễn phí**

Ngôn ngữ lập trình nào cũng miễn phí. Sự khác biệt giữa Python với các ngôn ngữ khác chính là việc nó hoàn toàn tương thích với các phần mềm mã nguồn mở (Open Sourse và GPL) vì Python cũng là mã nguồn mở.

### Thư viện OpenCV

#### Thư viện OpenCV là gì?

OpenCV[3] là một thư viện đa nền tảng, giúp ta có thể phát triển cácứng dụng thị giác máy tính. Nó chủ yếu trọng tâm vào xử lý hình ảnh, quay video và phân tích bao gồm các tính năng như phát hiện khuôn mặt và phát hiện đối tượng.

#### Các tính năng của OpenCV

Sử dụng OpenCV có thể :

+ Đọc và ghi hình ảnh

+ Ghi hình và lưu video

+ Xử lý hình ảnh (lọc, chuyển đổi)

+ Thực hiện nhận dạng đặc điểm

+ Phát hiện các đối tượng xác định như khuôn mặt, mắt, xe trong video hoặc hình ảnh.

+ Phân tích video, ... ước lượng chuyển động của nó, trừ nền ra và theo dõi các đối tượng trong video.

OpenCV ban đầu đượ phát triển trong C++. Hơn thế nữa, Python và Java cũng được hỗ trợ.OpenCV chạy trên nhiều hệ điều nhau như Windows,

Linux, OS, FreeBSD, Net BSD, Open BSD, ..

#### Ứng dụng của thư viện OpenCV

Hiện nay OpenCV được ứng dụng khá rộng rãi với nhiều mục đích khác nhau cả trong coogn nghệ các và các lĩnh vực đời sống khác bao gồm:

+ Ứng dụng vào hình ảnh street view.

+ Có thể kiểm tra , giám sát tự động.

+ Ứng dụng robot và xe hơi tự lái.

+ Phân tích được các hình ảnh trong y học.

+ Tìm kiếm cũng như phục hồi được các hình ảnh, video.

+ Ứng dụng vào phim và các cấu trúc 3D chuyển động.

+ Ứng dụng vào nghệ thuật sắp đặt tương tác.

#### Các module phổ biến của OpenCV

Trong OpenCV sẽ có các cấu trúc module khác nhau và nó bao gồm cả một số thư viện liên kết tĩnh hoặc cũng có thể là liên kết động. Các module phổ biến nhất:

+ Core (core functionality) – được biết đến là một module nhỏ gọn, giúp xác định được các cấu trúc dữ liệu cơ bản nhất và nó bao gồm các mảng đa chiều dày đặc , các chức năng cơ bản. Core cũng được sử dụng rất nhiều các module khác trong OpenCV.

+ Imgproc ( Image processing ) – đây là module có chức năng xử lý các hình ảnh gồm có lọc hình ảnh tuyến tính, phi tuyến tính , cho phép biển đổi về hình học như là chỉnh size, ánh xạ lại trên cơ sở bảng chung , afin..), đồng thời moule này cũng có khả năng chuyển đổi không gian màu, các biểu đồ.

+ Video (video analysis) – module này cho phép phân tích các video có các chức năng ước tính chuyển động , tách nền cũng như các thuật toán theo dõi vât thể trong Opencv.

+ Cali 3D (Camera Calibration and 3Dreconstruction) – đây là thuật toán hình học đa chiều cơ bản nhất, có thể hiệu chuẩn máy ảnh single và stereo cũng như dự đoán được các đối tượng , thuật toán như là thư tín âm thanh nổi, các yếu tố giúp tái tạo 3D.

+ Objdetect (object detection )- module giúp phát hiện ra các đối tượng , đồng thời mô phỏng cho các hàm đã được định sẵn như là predefined classes. Ví dụ là con người , khuôn mặt , mắt , xe hơi,…

+ Highgui (high- level GUI) – đây là một giao diện rất dễ sử dụng nhằm thực hiện quá trình giao tiếp đơn giản trong OpenCV.

+ Videoio (video I/O)- đây cũng là một giao diện rất dễ sử dụng nhằm thu mã hóa các vudeo trong OpenCv.

+ GPU – đây chính là các thuật toán giúp tăng tốc GPU từ những module OpenCV khác.

+ Ngoài ra thì còn một số module hỗ trợ như kaf Google test Wrapper, Flann, Python binding,…

### Jupyter notebook

#### Khái niệm

Jupyter notebook là công cụ cho phép bạn đưa cả code Python và các thành phần văn bản phức tạp như hình ảnh, công thức, video, biểu thức... vào trong cùng một file giúp cho việc trình bày trở lên dễ hiểu, giống như một file trình chiếu nhưng lại có thể thực hiện chạy code tương tác trên đó.

#### Ưu điểm Jupyter notebook

* Code ngay trong trình duyệt, hỗ trợ syntax highlight, thụt lề,... như một code editor.
* Chạy được code ngay trong trình duyệt.
* Hiển thị kết quả tính toán bằng cách sử dụng biểu diễn đa phương tiện, như HTML, LaTeX, PNG, SVG...
* Hỗ trợ markdown, khiến cho việc comment cho code trở nên trực quan hơn nhiều so với việc dùng plain text.

#### Cách cài đặt Jupyter notebook

Cách cài đặt dễ nhất, cũng như được khuyến khích nhất, đó là bạn cài Anaconda, nó cài sẵn cho ta Python, Jupyter Notebook, và ti tỉ thứ khác - tất cả những thứ mà một người bắt đầu với data science cần.

Nếu bạn không muốn cài Anaconda thì Jupyter cũng có thể cài qua pip - trình quản lý package của Python. Tất nhiên là bạn cần cài Python trước, sau đó mở terminal lên mà nhập command sau:

pip install jupyter

Hoặc nếu dùng Linux, bạn có thể tải và cài đặt bằng command:

sudo apt-get install jupyter-notebook python-matplotlib

### Train LSTM Neural Network

#### Khái niệm

Viết tắt của Long-short term memory, là một kiến trúc mạng [nơ ron](https://trituenhantao.io/tu-dien-thuat-ngu/no-ron/) hồi quy nhân tạo được sử dụng trong [deep learning](https://trituenhantao.io/tu-dien-thuat-ngu/deep-learning/). Không giống như các mạng truyền thẳng tiêu chuẩn, [LSTM](https://trituenhantao.io/tu-dien-thuat-ngu/lstm/) có các kết nối phản hồi. Nó không chỉ có khả năng xử lý các điểm dữ liệu đơn lẻ mà còn xử lý toàn bộ chuỗi dữ liệu mà không gặp phải vấn đề [vanishing gradient](https://trituenhantao.io/tu-dien-thuat-ngu/vanishing-gradient/).

#### Mô hình LSTM



Hình 1: Mô hình LSTM

Mạng LSTM có thể bao gồm nhiều tế bào LSTM liên kết với nhau. Ý tưởng của LSTM là bổ sung trạng thái bên trong tế bào (cell internal state) st và 3 cổng sàn lọc thông tin đầu vào và đầu ra cho tế bào bao gồm cổng quên ft, cổng đầu vào it và cổng đầu ra ot. Các cổng được định nghĩa như sau:

* + Cổng quên: Có chức năng loại bỏ những thông tin không cần thiết nhận được khỏi trạng thái tế bào bên trong
  + Cổng đầu vào: Giúp sàn lọc những thông tin cần tiết để được thêm vào trạng thái tế bào bên trong.
  + Cổng đầu ra: Có chức năng xác định những thông tin nào từ các trạng thái tế bào bên trong được sử dụng như đầu ra

### MediaPipe

#### Khái niệm

MediaPipe[4] là tập hợp của một loạt các giải pháp Machine Learning đa nền tảng, có thể can thiệp được và cực kỳ lightweight.

#### Ưu điểm của MediaPipe

* Cung cấp một giải pháp inference nhanh chóng: Google khẳng định rằng bộ công cụ này có thể chạy ổn định trên hầu hết các cấu hình phần cứng thông dụng.
* Dễ dàng cài đặt và triển khai: Việc cài đặt cực kỳ dễ dàng và tiện lợi, có thể triển khai trên nhiều nền tảng khác nhau như Mobile (Android/iOS), Desktop/Cloud, Web và IoT devices.
* Mã nguồn mở và miễn phí: Toàn bộ source code được công khai trên [MediaPipe](https://github.com/google/mediapipe), người dùng hoàn toàn có thể sử dụng và tùy chỉnh trực tiếp để phù hợp với bài toán của mình.

#### Các giải pháp trong MediaPipe

##### Face Detection

Đây là một bài toán quen thuộc với tất cả mọi người. Với đầu vào là một ảnh hoặc một video, nhiệm vụ của chúng ta là tìm ra vị trí và đóng hộp (bounding box) những khuôn mặt con người xuất hiện trên đấy, cũng như đánh dấu các điểm quan trọng (MediaPipe sử dụng 5-landmarks) trên khuôn mặt đó. MediaPipe Face Detection sử dụng mạng BlazeFace làm nền tảng nhưng thay đổi backbones. Ngoài ra, thuật toán NMS (non-maximum suppression) cũng được thay thế bởi một chiến thuật khác, giúp thời gian xử lý giảm đáng kể.

##### Face Mesh

Thay vì tìm bounding box bao quanh mặt, Face Mesh là một bài toán nhận diện một loạt các điểm trên khuôn mặt, từ đó tạo thành 1 lưới (mesh) của mặt. Chiếc lưới này sẽ được áp dụng vào các bài toán chỉnh sửa ảnh mặt 3D hay các tác vụ liên quan tới 3D Alignment và Anti-spoofing. Công cụ trong MediaPipe sẽ sinh ra tổng cộng 468 điểm trên mặt và tạo thành lưới mà không đòi hỏi quá nhiều năng lực tính toán cũng như số camera (chỉ cần 1 camera chính diện).

##### Hands Detection

Hands Detection, hay còn gọi là nhận diện bàn tay, là bài toán mà chúng ta sẽ thử nghiệm trong bài này. Đầu ra của giải pháp này là một mô hình skeleton (khung xương) của bàn tay, gồm vị trí của các landmarks trên bàn tay và được nối với nhau thành một khung bàn tay hoàn chỉnh.

##### Human Pose Estimation

Mở rộng từ bài toán Hands Detection, Human Pose Estimation cung cấp một mô hình skeleton 3D cho cả cơ thể, với các khớp quan trọng được định nghĩa sẵn và được nối với nhau để tạo thành khung của người. Chiến thuật được đặt ra cho bài toán này tương tự như bài Hands Detection và Face Mesh. BlazeFace, một lần nữa, được sử dụng làm tư tưởng chính cho thuật toán xử lý bài này.

Ngoài ra, còn rất nhiều giải pháp nữa được Google cung cấp trong bộ cung cấp này, bao gồm như các bài toán Segmentation (Selfie, Hair, ...), Object Detection, Motion Tracking, 3D Object Detection, ... Tuy nhiên, mình sẽ không liệt kê hết trong bài viết này do quá dài. Còn bây giờ, mình sẽ giới thiệu một ứng dụng đơn giản để áp dụng bộ công cụ này vào thực tế.

## NHẬN DẠNG VÀ XỬ LÝ ẢNH

### Quá trình nhận dạng

#### Tính chất và sự cần thiết của bài toán nhận dạng

##### Tính chất

Nhận dạng là quá trình phân loại các đối tượng được biểu diễn theo một mô hình nào đó và gán cho chúng vào một lớp (gán đối tượng một tên gọi) dựa theo những quy luật và các mẫu chuẩn. Quá trình nhận dạng dựa vào những mẫu học biết trước gọi là học có giám sát ( supervised learning ); trong trường hợp ngược lại gọi là học không có giám sát (non supervised learning). Nhận dạng là một bài toán quan trọng trong ngành thị giác máy tính .

##### Sự cần thiết

Cùng với sự phát triển không ngừng của kinh kế và các ngành kỹ thuật hiện nay. Đòi hỏi sự quản lý và xử lý thông tin chính xác mà nó vượt quá sức của con người. Vì vậy, chúng ta cần có máy móc làm giảm hoặc thay thế công việc nặng nhọc, đòi hỏi sự chính xác cao và nhàm chán cho con người.

Một số ứng dụng của bài toán nhận dang

+ Nhận dạng dấu vân tay: ở Việt Nam đã được sử dụng trong việc chấm công, điểm danh và làm khóa an toàn cho các loại cửa sắt , máy tính xách tay..

+ Nhận dạng giọng nói: ứng dụng này được tích hợp trên nhiều sản phấm, bạn sẽ điều khiển các thiết bị qua giọng nói của mình thay vì thao tác trực tiếp bằng tay, như trong mô hình nhà thông minh, trên điện thoại....

#### Không gian biểu diễn đối tương , không gian miễn dịch.

##### Không gian biểu diễn đối tương

Các đối tượng khi quan sát hay thu nhập thường được biểu diễn bởi tập các đặc trưng. Như trong trường hợp xử lý ảnh, ảnh sau khi được tăng cường để nâng cao chất lượng,phân vùng và trích chọn đặc tính , được biểu diễn bởi các đặc trưng nhu biên, miền đồng nhất….. Người ta thường phân các đặc trưng này theo các loại như: đặc trung tô pô (tập điểm), đặc trưng hình học và đặc trưng chức năng

##### Không gian miễn dịch.

Không gian miễn dịch là tập các tên gội của đối tượng. Kết thúc quá trình nhận dạng ta xác định được tên gọi cho các đối tượng trong tập không gian đối tượng hay nói là đã nhận dạng đươc đối tượng.

Một hình thức gọi là Ω là tập tên đối tượng:

là tên các đối tượng

Quá trình nhận dạng đối tượng f là một ánh xạ f: A Ω với f là tập các quy luật để định một phần tử A ứng với một phần tử trong Ω.

#### Mô hình và bản chất của quá trình nhận dạng

##### Mô hình

Việc lựa chọn một quá trình nhận dạng có liên quan mật thiết đến kiểu mô tả mà người ta sử dụng để đặc tả đối tượng. Trong nhận dạng, người ta phân chia theo hai họ lớp:

+ Họ mô tả theo tham số.

+ Họ mô tả theo cấu trúc

a) Mô hình tham số

Mô hình tham số sử dụng một vector để đặc trưng đối tượng. Mỗi phần tử của vector mô tả một đặc tính của đối tượng. Ví dụ như trong các đặc trưng chức năng , người ta sử dụng các hàm cơ sở trực giao để biểu diện. Và như vậy ảnh sẽ được biểu diễn bởi một chuỗi các hàm trực giao. Giả sử C là đường bao của ảnh là điểm thứ i trên đường bao, i= 1,2,….,n ( đường bao gồm n điểm ).

b) Mô hình cấu trúc

Cách tiếp cận mô hình này dựa vào việc mô tả đối tượng nhờ một số khái niệm biểu thị các đối tượng cơ sở trong ngôn ngữ tự nhiên . Để mô tả đối tượng, người ta dùng một số dạng nguyên thủy như đoạn thẳng, cung …. Ngoài ra còn dùng một tập các luật sản xuất để mô tả cách xây dựng các đối tượng phù hợp trên các đối tượng đơn giản hơn hoặc đối tượng nguyên thủy. Trong cách tiếp cận này, ta chấp nhận một khẳng định là: cấu trúc là kết quả của việc áp dụng luật sản xuất theo những nguyên tắc xác định bắt đầu từ môt dạng gốc bắt đầu. Một cách hình thức, ta có thể coi mô hình này tương đương một văn phạm

với:

+ là bộ kí hiệu kết thúc.

+ là bộ kí hiệu không kết thúc

+ P là luật sản xuất

+ S là dạng (ký hiệu bắt đầu ).

##### Bản chất của quá trình nhận dạng

Quá trình nhận dạng gồm 3 giai đoạn chính :

+ Lựa chọn mô hình biểu diễn đối tượng.

+ Lựa chọn luật ra quyết định ( phương pháp nhận dạng ) và suy diễn quá trình học .

+ Học nhận dạng

Khi mô hình biểu diễn đối tượng đã được xác định , có thể là định lượng (mô hình tham số) hay định tính (mô hình cấu trúc), quá trình nhận dạng chuyển sang giai đoạn học. Học là giai đoạn quan trong. Thao tác học nhằm cải thiên, điều chỉnh việc phân họạch tập đối tượng thành lớp.

Việc nhận dạng chính là tìm ra quy luật và các thuật toán để có thể gán đối tượng vào một lớp hay nói một cách khác gán cho đối tương một tên.

1. Học có giám sát (supervised learning)

Kỹ thuật phân loại nhờ kiến thức biết trước gọi là học có giám sát. Đặc điểm cơ bản của kỹ thuật này là người ta có một thư viện các mẫu chuẩn.Mẫu cần nhận dạng sẽ được đem so sánh với mẫu chuẩn để xem nó thuộc loại nào . Vấn đề chủ yếu là thiết kế một hệ thống để có thể đối sánh đối tượng trong ảnh với mẫu chuẩn và quyết định gán cho chúng một lớp. Việc đối sánh nhờ vào các thủ tục ra quyết định dựa trên một công cụ gọi là hàm phân lớp hay hàm ra quyết định.

1. Học không có giám sát (non supervised learning)

Kỹ thuật này phải tự định ra các lớp khác nhau và xác định tham số đặc trưng cho từng lớp. Học không có giám sát khó khăn hơn.Một mặt, do số lớp không được biết trước, mặt khác những đặc trưng của lớp cũng không biết trước. Kỹ thuật này nhằm tiến hành mọi cách gộp nhóm có thể và lựa chọn cách tốt nhất. Bắt đầu từ tập dữ liệu, nhiều thủ tục xử lý khác nhau nhằm phân lớp và nâng cấp dần để đạt được môt phương án phân loại.

### Xử lý ảnh

#### Tổng quan

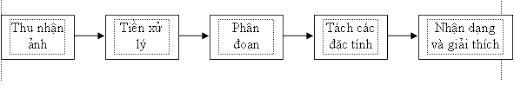
Xử lý ảnh (XLA) là đối tƣợng nghiên cứu của lĩnh vực thị giác máy, là quá trình biến đổi từ một ảnh ban đầu sang một ảnh mới với các đặc tính và tuân theo ý muốn của người sử dụng. Xử lý ảnh có thể gồm quá trình phân tích, phân lớp các đối tƣợng, làm tăng chất lượng, phân đoạn và tách cạnh, gán nhãn cho vùng hay quá trình biên dịch các thông tin hình ảnh của ảnh.

Cũng như xử lý dữ liệu bằng đồ hoạ, xử lý ảnh số là một lĩnh vực của tin học ứng dụng. Xử lý dữ liệu bằng đồ họa đề cập đến những ảnh nhân tạo, các ảnh này được xem xét như là một cấu trúc dữ liệu và được tạo bởi các chương trình. Xử lý ảnh số bao gồm các phương pháp và kỹ thuật biến đổi, để truyền tải hoặc mã hoá các ảnh tự nhiên. Mục đích của xử lý ảnh gồm:

* + Biến đổi ảnh làm tăng chất lƣợng ảnh.
  + Tự động nhận dạng ảnh, đoán nhận ảnh, đánh giá các nội dung của ảnh.

Nhận biết và đánh giá các nội dung của ảnh là sự phân tích một hình ảnh thành những phần có ý nghĩa để phân biệt đối tượng này với đối tượng khác, dựa vào đó ta có thể mô tả cấu trúc của hình ảnh ban đầu. Có thể liệt kê một số phương pháp nhận dạng cơ bản như nhận dạng ảnh của các đối tƣợng trên ảnh, tách cạnh, phân đoạn hình ảnh,… Kỹ thuật này được dùng nhiều trong y học (xử lý tế bào, nhiễm sắc thể), nhận dạng chữ trong văn bản.

#### Quá trình xử lý ảnh



Hình 2: Các giai đoạn chính trong xử lý ảnh

Thu nhận ảnh: Đây là công đoạn đầu tiên mang tính quyết định đối với quá trình XLA. Ảnh đầu vào sẽ được thu nhận qua các thiết bị như camera, sensor, máy scanner,v.v… và sau đó các tín hiệu này sẽ đƣợc số hóa. Việc lựa chọn các thiết bị thu nhận ảnh sẽ phụ thuộc vào đặc tính của các đối tƣợng cần xử lý. Các thông số quan trọng ở bước này là độ phân giải, chất lượng màu, dung lượng bộ nhớ và tốc độ thu nhận ảnh của các thiết bị.

Tiền xử lý: Ở bước này, ảnh sẽ được cải thiện về độ tương phản, khử nhiễu, khử bóng, khử độ lệch,v.v… với mục đích làm cho chất lượng ảnh trở lên tốt hơn nữa, chuẩn bị cho các bước xử lý phức tạp hơn về sau trong quá trình XLA. Quá trình này thường đƣợc thực hiện bởi các bộ lọc.

Phân đoạn ảnh: phân đoạn ảnh là bƣớc then chốt trong XLA. Giai đoạn này phân tích ảnh thành những thành phần có cùng tính chất nào đó dựa theo biên hay các vùng liên thông. Tiêu chuẩn để xác định các vùng liên thông có thể là cùng màu, cùng mức xám v.v… Mục đích của phân đoạn ảnh là để có một miêu tả tổng hợp về nhiều phần tử khác nhau cấu tạo lên ảnh thô. Vì lượng thông tin chứa trong ảnh rất lớn, trong khi đa số các ứng dụng chúng ta chỉ cần trích một vài đặc trưng nào đó, do vậy cần có một quá trình để giảm lượng thông tin khổng lồ đó. Quá trình này bao gồm phân vùng ảnh và trích chọn đặc tính chủ yếu.

Tách các đặc tính: Kết quả của bước phân đoạn ảnh thường đƣợc cho dưới dạng dữ liệu điểm ảnh thô, trong đó hàm chứa biên của một vùng ảnh, hoặc tập hợp tất cả các điểm ảnh thuộc về chính vùng ảnh đó. Trong cả hai trường hợp, sự chuyển đổi dữ liệu thô này thành một dạng thích hợp hơn cho việc xử lý trong máy tính là rất cần thiết. Để chuyển đổi chúng, câu hỏi đầu tiên cần phải trả lời là nên biểu diễn một vùng ảnh dƣới dạng biên hay dưới dạng một vùng hoàn chỉnh gồm tất cả những điểm ảnh thuộc về nó. Biểu diễn dạng biên cho một vùng phù hợp với những ứng dụng chỉ quan tâm chủ yếu đến các đặc trưng hình dạng bên ngoài của đối tƣợng, ví dụ như các góc cạnh và điểm uốn trên biên chẳng hạn. Biểu diễn dạng vùng lại thích hợp cho những ứng dụng khai thác các tính chất bên trong của đối tượng.

#### Phạm vi ứng dụng của xử lý ảnh

Xử lý ảnh đã đem lại nhiều ứng dụng trong nhiều lĩnh vực khác nhau: y học, khoa học hình hình sự, khí tƣợng thuỷ văn, quản lý, ... Quản lý là là một trong những ứng dụng quan trọng của xử lý ảnh. Cùng với sự bùng nổ của kinh tế thị trường. Khối lượng quản lý càng lớn, như quản lý hồ sơ, quản lý phiếu điều tra trong công tác thống kê, các câu hỏi trắc nghiệm. Để thực hiện các công việc trên một cách chính xác, nhanh chóng và hiệu quả. Xử lý ảnh và nhận dạng đã nghiên cứu và phát triển mạnh mẽ bài toán nhập liệu tự động.

## NHẬN DẠNG PHÁT HIỆN HÀNH ĐỘNG CHO NGÔN NGỮ KÝ HIỆU

### Import and Install Dependencies

!pip install tensorflow==2.4.1 tensorflow-gpu==2.4.1 opencv-python mediapipe sklearn matplotlibimport cv2

import cv2

import numpy as np

import os

from matplotlib import pyplot as plt

import time

import mediapipe as mp

**+** Cài các thư viện cần thiết và import các thư viện

### Lấy những điểm chính

mp\_holistic = mp.solutions.holistic # Holistic model

mp\_drawing = mp.solutions.drawing\_utils # Drawing utilities

mp\_drawing\_styles = mp.solutions.drawing\_styles

mphands = mp.solutions.hands

hands = mphands.Hands()

mp\_drawing = mp.solutions.drawing\_utils

def mediapipe\_detection(image, model):

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB) # COLOR CONVERSION BGR 2 RGB

image.flags.writeable = False # Image is no longer writeable

results = model.process(image) # Make prediction

image.flags.writeable = True # Image is now writeable

image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR) # COLOR COVERSION RGB 2 BGR

return image, results

def draw\_landmarks(image, results):

mp\_drawing.draw\_landmarks(image, results.left\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS) # Draw left hand connections

mp\_drawing.draw\_landmarks(image, results.right\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS) # Draw right hand connections

def draw\_styled\_landmarks(image, results):

# Draw left hand connections

mp\_drawing.draw\_landmarks(image, results.left\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(121,22,76), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(121,44,250), thickness=2, circle\_radius=2)

)

# Draw right hand connections

mp\_drawing.draw\_landmarks(image, results.right\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

# mp\_drawing.DrawingSpec(color=(245,117,66), thickness=2, circle\_radius=4),

# mp\_drawing.DrawingSpec(color=(245,66,230), thickness=2, circle\_radius=2)

mp\_drawing\_styles.get\_default\_hand\_landmarks\_style(),

mp\_drawing\_styles.get\_default\_hand\_connections\_style()

)

cap = cv2.VideoCapture(0)

# Set mediapipe model

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic:

while cap.isOpened():

# Read feed

ret, frame = cap.read()

# Make detections

image, results = mediapipe\_detection(frame, holistic)

print(results)

# Draw landmarks

draw\_styled\_landmarks(image, results)

# Show to screen

cv2.imshow('OpenCV Feed', image)

# Break gracefully

if cv2.waitKey(10) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

len(results.left\_hand\_landmarks.landmark)

results

draw\_landmarks(frame, results)

plt.imshow(cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB))

**+ def** mediapipe\_detection(image, model) : Phát hiện đối tượng, truyền vào hình ảnh và model của chúng ta sau đó chuyển đổi màu BGR sang RGB ,Trạng thái có thể ghi false, tiếp theo đưa ra dự đoán và chuyển đổi trạng thái có thể ghi và trả về hình ảnh và kết quả dự đoán

**+** def draw\_landmarks(image, results): Vẽ các điểm mốc của đối tượng.

+ def draw\_styled\_landmarks(image, results): Vẽ các điểm mốc theo styled

**+** Hàm cap = cv2.VideoCapture(0) dùng để lấy đối tượng từ máy ảnh

+ Thiết lập vòng while vô hạn và dùng read() để đọc các khung hình

+withmp\_holistic.Holistic(min\_detection\_confidence=0.5,min\_tracking\_confidence=0.5) as holistic: truy cập mô hình mediapipe với độ tin cậy phát hiện ban đầu = 0.5 và xác định độ tin cậy phát hiện = 0.5

+ image, results = mediapipe\_detection(frame, holistic) thực hiện phát hiện đối tượng

+ cv2.imshow() hiển thị các khung hình trong video

+ Ngắt vòng lặp khi nhấn vào phím ‘q’

+ cv2.destroyAllWindows() phá hủy tất cả các cửa sổ

+ len(results.left\_hand\_landmarks.landmark) in ra số điểm mốc trên đối tượng

### Trích xuất các giá trị điểm

**def** extract\_keypoints(results):

lh **=** np**.**array([[res**.**x, res**.**y, res**.**z] **for** res **in** results**.**left\_hand\_landmarks**.**landmark])**.**flatten() **if** results**.**left\_hand\_landmarks **else** np**.**zeros(21**\***3)

rh **=** np**.**array([[res**.**x, res**.**y, res**.**z] **for** res **in** results**.**right\_hand\_landmarks**.**landmark])**.**flatten() **if** results**.**right\_hand\_landmarks **else** np**.**zeros(21**\***3)

**return** np**.**concatenate([lh, rh])

result\_test = extract\_keypoints(results)

np.save('0', result\_test)

np.load('0.npy')

+ def extract\_keypoints(results): Trích xuất các giá trị điểm chính qua giá trị result. Xây dựng mảng nối các điểm x,y,z từ các điểm nếu không lấy được các điểm mốc thì sẽ tạo ra mảng có cùng hình dạng. Trả về kết quả nối các mảng lh,rh.

### Tạo thư mục cho dữ liệu

# Path for exported data, numpy arrays

DATA\_PATH = os.path.join('MP\_Data')

# Actions that we try to detect

actions = np.array(['ok', '1', '2','3', '4', '5','thank'])

# Thirty videos worth of data

no\_sequences = 40

# Videos are going to be 30 frames in length

sequence\_length = 40

for action in actions:

for sequence in range(no\_sequences):

try:

os.makedirs(os.path.join(DATA\_PATH, action, str(sequence)))

except:

pass

+ Tạo đường dẫn cho dữ liệu xuất ra,Các hành động mà ta cố gắng phát hiện ra, Chúng ta sẽ thu thập 40 khung hình dài và 40 khung hình nhỏ trong 40 khung hình dài

+ Tạo ra thư mục con của các hành động, và trong mỗi thư mục con từ 0 đến 39 cho đến khi khi kết thúc

### Thu thập các giá trị để huấn luyện và kiểm tra

cap = cv2.VideoCapture(0)

# Set mediapipe model

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic:

for action in actions:

# Loop through sequences aka videos

for sequence in range(no\_sequences):

# Loop through video length aka sequence length

for frame\_num in range(sequence\_length):

# Read feed

ret, frame = cap.read()

# Make detections

image, results = mediapipe\_detection(frame, holistic)

draw\_styled\_landmarks(image, results)

# NEW Apply wait logic

if frame\_num == 0:

cv2.putText(image, 'STARTING COLLECTION', (120,200),

cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0,255, 0), 4, cv2.LINE\_AA)

cv2.putText(image, 'Collecting frames for {} Video Number {}'.format(action, sequence), (15,12),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 0, 255), 1, cv2.LINE\_AA)

# Show to screen

cv2.imshow('OpenCV Feed', image)

cv2.waitKey(500)

else:

cv2.putText(image, 'Collecting frames for {} Video Number {}'.format(action, sequence), (15,12),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 0, 255), 1, cv2.LINE\_AA)

# Show to screen

cv2.imshow('OpenCV Feed', image)

# NEW Export keypoints

keypoints = extract\_keypoints(results)

npy\_path = os.path.join(DATA\_PATH, action, str(sequence), str(frame\_num))

np.save(npy\_path, keypoints)

if cv2.waitKey(10) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

+ Set model và thực hiện lặp các hành động, từng video sẽ lặp 40 lần để lấy và quay 40 video cho mỗi hành động, thu thập 40 điểm chính mỗi video để trong mỗi hành động

+ Thực hiện phát hiện đối tượng và vẽ đối tượng, trích xuất các điểm chính và lưu chúng vào các thư mục.

### Tiền xử lý dữ liệu và tạo nhãn và các đặc điểm

from sklearn.model\_selection import train\_test\_split

from tensorflow.keras.utils import to\_categorical

label\_map = {label:num for num, label in enumerate(actions)}

sequences, labels = [], []

for action in actions:

for sequence in range(no\_sequences):

window = []

for frame\_num in range(sequence\_length):

res = np.load(os.path.join(DATA\_PATH, action, str(sequence), "{}.npy".format(frame\_num)))

window.append(res)

sequences.append(window)

labels.append(label\_map[action])

np.array(sequences).shape

np.array(labels).shape

X = np.array(sequences)

y = to\_categorical(labels).astype(int)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.05)

+ Tạo cấu trúc các hành động. Lặp lại các hành động và với mỗi sequence của chúng ra sẽ có 30 video và sau đó sẽ tạo một mảng trống

+ Lặp lại từng khung trong phạm vi độ dài của sequence, sau đó chúng ta sử dụng numpy để load các khung đó .

+ Sau mỗi khung hình chúng ta sẽ bổ sung vào mảng sequences và labels.

### Build and Train LSTM Neural Network

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

from tensorflow.keras.callbacks import TensorBoard

log\_dir = os.path.join('Logs')

tb\_callback = TensorBoard(log\_dir=log\_dir)

model = Sequential()

model.add(LSTM(64, return\_sequences=True, activation='relu', input\_shape=(sequence\_length,126)))

model.add(LSTM(128, return\_sequences=True, activation='relu'))

model.add(LSTM(64, return\_sequences=False, activation='relu'))

model.add(Dense(64, activation='relu'))

model.add(Dense(32, activation='relu'))

model.add(Dense(actions.shape[0], activation='softmax'))

res = [.7, 0.2, 0.1]

actions[np.argmax(res)]

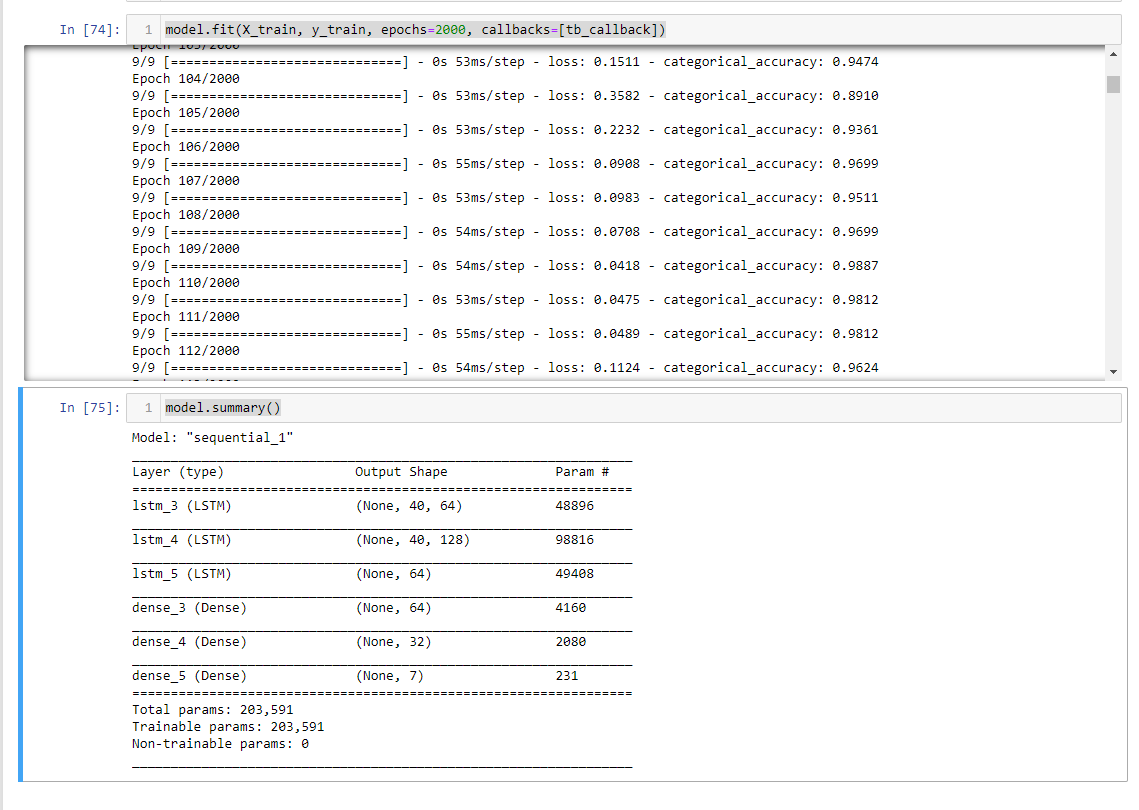
model.compile(optimizer='Adam', loss='categorical\_crossentropy', metrics=['categorical\_accuracy'])

model.fit(X\_train, y\_train, epochs=2000, callbacks=[tb\_callback])

model.summary()

+ Tạo dữ liệu file Logs sau đó cho phép chúng ta theo dõi thần kinh của chúng ta đào tạo ,nếu ta muốn theo dõi quá trình đào tạo chỉnh cần vào TensorBoard

+ Tạo mô hình Sequential(), sau đó add thêm 3 bộ lớp LSTM và Dense



### Đưa ra dự đoán, lưu và load model

res = model.predict(X\_test)

print(X\_test.shape)

actions[np.argmax(res[0])]

actions[np.argmax(y\_test[0])]

model.save('action.h5')

model.load\_weights('action.h5')

### Đánh giá ma trận nhầm lẫn và độ chính xác

from sklearn.metrics import multilabel\_confusion\_matrix, accuracy\_score

yhat = model.predict(X\_test)

ytrue = np.argmax(y\_test, axis=1).tolist()

yhat = np.argmax(yhat, axis=1).tolist()

print(len(yhat))

print(len(ytrue))

accuracy\_score(ytrue, yhat)

multilabel\_confusion\_matrix(ytrue, yhat)

### Chạy chương trình

sequence = []

sentence = []

threshold = 0.8

cap = cv2.VideoCapture(0)

# Set mediapipe model

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic:

while cap.isOpened():

…………………………………………………………………

………………………………………..

# Viz probabilities

#image = prob\_viz(res, actions, image, colors)

cv2.rectangle(image, (0,0), (640, 40), (245, 117, 16), -1)

cv2.putText(image,' '.join(sentence), (7,30),cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 4, cv2.LINE\_AA)

cv2.imshow('OpenCV Feed', image)

# Break gracefully

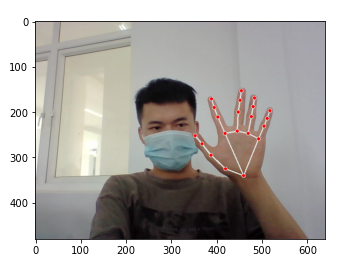
if cv2.waitKey(10) & 0xFF == ord('q'):

break

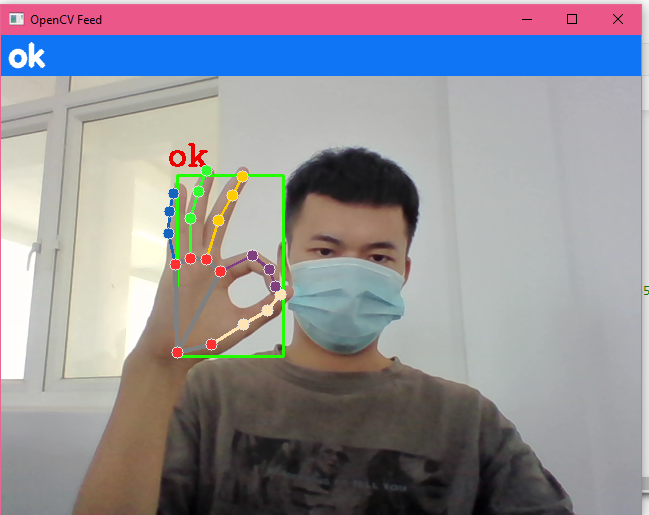
cap.release()

cv2.destroyAllWindows()

### Kết quả chương trình



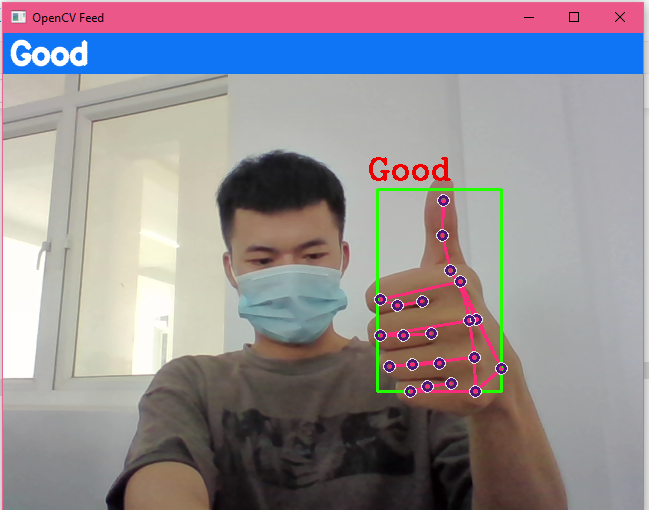
Hình 3: Mẫu thử 1



Hình 4: Mẫu thử 2

# 

Hình 5: Mẫu thử 3



Hình 6: Mẫu thử 4

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Đánh giá kết quả đạt được

* Sau một thời gian tìm hiểu và thực hiện, đề tài đã đạt được mục tiêu đã đề ra, thu được những kết quả ý nghĩa
* Đã tóm tắt được lý thuyết liên quan đến thư viện Opencv, ngôn ngữ lập trình Python
* Tìm hiểu quá trình nhận dạng và xử lý ảnh và ứng dụng vào bài toán nhận diện phát hiện hành động ngôn ngữ ký hiệu.
* Nhận diện được hành động từ dữ liệu đầu vào là hình ảnh từ camera.

2. Hạn chế

* Kết quả nhận diện chưa tuyệt đối chính xác.
* Nâng cao hiệu quả chương trình, hạn chế các sai sót nhầm lẫn trong quá trình nhận dạng và xử lý
* Phát triển để có thể nhận dạng hành động có độ tương phản giữa màu sắc, ánh sáng.

TÀI LIỆU THAM KHẢO

1. Giới thiệu về python, <<https://openplanning.net/11385/gioi-thieu-ve-python>/>, xem 12/3/2022
2. Scitkit-learn, <<https://scikit-learn.org/stable/>>, xem 12/3/2022
3. Opencv, <<https://opencv.org/> >, xem 12/3/2022
4. Mediapipe, < <https://mediapipe.dev/>>, xem 12/3/2022
5. TensorFlow, <https://www.tensorflow.org/tutorials/keras/save\_and\_load/, xem 12/3/2022