

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

**BÁO CÁO**

**PBL4 - ĐỒ ÁN HỆ THỐNG THÔNG MINH**



**HỆ THỐNG NHẬN DIỆN CỬ CHỈ BÀN TAY   
GIAO TIẾP VỚI NGƯỜI KHIẾM THÍNH**

**Giảng viên hướng dẫn: TS. Phạm Công Thắng**

|  |  |
| --- | --- |
| STT NHÓM: 3  HỌ VÀ TÊN SINH VIÊN | LỚP HỌC PHẦN ĐỒ ÁN |
| Lý Văn Tánh | 20.99A |
| Phạm Tiến Hữu | 20.99A |
| Lê Hoàng Long | 20.99A |
| Trần Nhật Minh | 20.99A |

ĐÀ NẴNG, 12/2022

**TÓM TẮT ĐỒ ÁN**

Xã hội ngày nay không ngừng phát triển trên mọi phương diện, từ đời sống sinh hoạt thường ngày đến các lĩnh vực văn hóa, giải trí, họạt động vui chơi ngày càng phong phú và đa dạng hơn. Việc giao tiếp giữa người với người, tạo thêm nhiều mối quan hệ mới và gắn kết tập thể, cộng đồng với nhau đang trở thành một kỹ năng mềm quan trọng và rất cần được trau dồi học tập. Tuy nhiên xã hội còn nhiều người không được may mắn khi sinh ra bị các tật bẩm sinh khiếm thính, dẫn đến đánh mất khả năng giao tiếp. Do đó, chúng em đã đề xuất và cùng nhau xây dựng đồ án với đề tài “Xây Dựng Hệ Thống Nhận Diện Cử Chỉ Bàn Tay Giao Tiếp Với Người Khiếm Thính” với mong muốn giúp người khiếm thính giao tiếp dễ dàng hơn và cải thiện đời sống tinh thần.

Nhóm em đã sử dụng công nghệ AI như các thuật toán thị giác máy tính sử dụng mạng CNN, các thư viện OpenCV, cvzone để thu thập hình ảnh và hiển thị cấu trúc bàn tay, sử dụng model VGG16 phục vụ train model và dùng giao thức socket client để giao tiếp giữa server và Raspberry Pi 3

Sau khi nghiên cứu và thử nghiệm, hệ thống đã đáp ứng các yêu cầu cơ bản và hoạt động tốt. Mặc dù đã dành rất nhiều tâm huyết, công sức để hoàn thiện, song sản phẩm của đề tài không thể tránh khỏi những thiếu sót. Chúng em rất mong nhận được phản hồi tích cực và góp ý chân thành từ phía thầy cô

**BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** | **Các nhiệm vụ** | **Tiến độ** |
| **Lý Văn Tánh** | - Sử dụng Socket để gửi ảnh từ Client sang Server  - Code Hand Sign Detection  - Tìm hiểu Raspbery Pi 3, WinSCP. VNC Viewer | Đã hoàn thành |
| **Phạm Tiến Hữu** | - Sử dụng Socket để nhận ảnh và trả message về Client từ Server  - Train model VGG16  - Code Hand Sign Detection | Đã hoàn thành |
| **Lê Hoàng Long** | - Xây dựng dataset cho Hand Sign Detection  - Code Hand Sign Detection  - Tìm hiểu Raspberry Pi 3, WinSCP, VNC Viewer | Đã hoàn thành |
| **Trần Nhật Minh** | - Tìm hiểu Raspberry Pi 3, WinSCP, VNC Viewer  - Tìm hiểu cách sử dụng HandDetection  - Viết báo cáo | Đã hoàn thành |

**Mục lục**

[1. Giới thiệu 6](#_Toc123072240)

[2. Giải pháp 7](#_Toc123072241)

[2.1. Giải pháp về phần cứng và truyền thông 7](#_Toc123072242)

[2.1.1. Truyền thông 7](#_Toc123072243)

[2.1.2. Phần cứng 10](#_Toc123072244)

[2.1.3. Nguyên lý mắc mạch 11](#_Toc123072245)

[2.2. Giải pháp về phần mềm 13](#_Toc123072246)

[2.2.1. Luồng xử lý chương trình 13](#_Toc123072247)

[2.2.2. Mạng CNN và VGG16 13](#_Toc123072248)

[2.2.3. Các bước thực hiện 15](#_Toc123072249)

[3. Kết quả 16](#_Toc123072250)

[3.1. Tập dữ liệu ( Dataset) 16](#_Toc123072251)

[3.3. Kết quả nhận diện 18](#_Toc123072252)

[4. Kết luận và hướng phát triển 18](#_Toc123072253)

[4.1. Kết luận 18](#_Toc123072254)

[4.2. Hướng phát triển 19](#_Toc123072255)

[5. Danh mục tài liệu tham khảo 19](#_Toc123072256)

**Danh mục hình vẽ**

[Hình 1. Sơ đồ hoạt động dự án của nhóm sinh viên HCMUT 6](#_Toc123056001)

[Hình 2. Mô hình socket 7](#_Toc123056002)

[Hình 3. Nguyên lý hoạt động Client Server 8](#_Toc123056003)

[Hình 4. Module I2C LCD 11](#_Toc123056004)

[Hình 5. Sơ đồ mắc mạch LCD và Raspberry Pi 3 12](#_Toc123056005)

[Hình 6. Sơ đồ mắc mạch tổng thể 12](#_Toc123056006)

[Hình 7. Sơ đồ tổ chức làm việc của chương trình 13](#_Toc123056007)

[Hình 8. Mô hình hoạt động CNN 14](#_Toc123056008)

[Hình 9. Cấu trúc mạng VGG16 15](#_Toc123056009)

[Hình 10. Tập dataset chuẩn bị train model 16](#_Toc123056010)

[Hình 11. Kết quả train model 17](#_Toc123056011)

[Hình 12. Ma trận nhầm lẫn 17](#_Toc123056012)

[Hình 13. Nhận diện thủ ngữ 18](#_Toc123056013)

[Hình 14. Hiển thị kết quả lên màn hình LCD 16x2 18](#_Toc123056014)

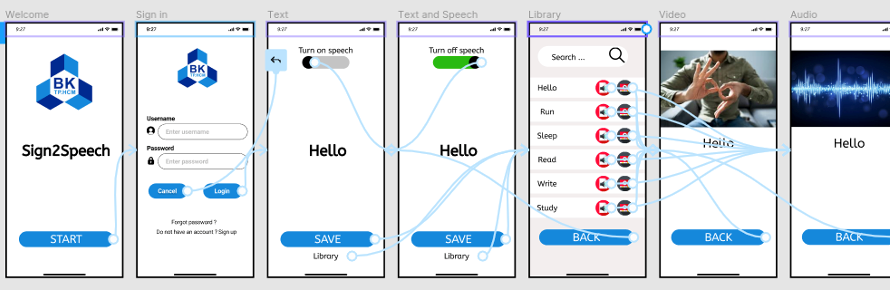
# 1. Giới thiệu

Nhận dạng là một ngành thuộc lĩnh vực học máy đã có lịch sử khá lâu, bắt đầu từ thập kỷ 60 của thế kỷ XX. Khi đó hầu hết vấn đề nhận dạng chỉ dừng lại ở việc nghiên cứu lí thuyết thống kê. Về sau với sự phát triển mạnh mẽ của máy tính đã mở ra nhiều cơ hội thực nghiệm hơn. Đối với vấn đề tự động hóa trong bối cảnh xã hội ngày nay thì việc thông tin được nhận và xử lý một cách tự động là rất cần thiết. Khuynh hướng này làm cho vấn đề nhận dạng mẫu trở nên rất quan trọng trong ứng dụng kỹ thuật và trong nghiên cứu ngày nay. Nhận dạng mẫu tích hợp hầu hết vào các hệ thống máy móc thông minh, có khả năng tự đưa ra quyết định để giải quyết vấn đề.

Các ứng dụng phổ biến là nhận dạng tiếng nói tự động, phân loại văn bản thành nhiều loại khác nhau (ví dụ: những thư điện tử nào là thư rác và không phải thư rác), nhận dạng tự động các mã bưu điện viết tay trên các bao thư, hay hệ thống nhận dạng danh tính dựa vào mặt người v.v...

Hiện nay, nước ta có khoảng 1 đến 2,5 triệu người khiếm thanh và người khiếm thính, tương đương dân số một tỉnh. Tuy nhiên có rất ít phiên dịch viên ngôn ngữ ký hiệu. Đặc biệt, một số tỉnh có rất ít các trung tâm hay thậm chí không mở trung tâm dạy thủ ngữ. Vì thế, những người bị khiếm thanh, khiếm thính hiện vẫn đang sử dụng thủ ngữ để giao tiếp trong cuộc sống. Cách này có khá nhiều điểm hạn chế như: người giao tiếp buộc phải biết thủ ngữ, thời gian tương tác lâu, dễ nhầm lẫn v.v…

Năm 2021, một nhóm gồm 6 sinh viên thuộc trường ĐH Bách Khoa HCM (HCMUT) đã nghiên cứu dự án “Thiết bị giao tiếp thông minh dành cho người khiếm thanh, khiếm thính – Speak Your Mind (SYM)”. Dự án được các nhóm tư vấn gồm nhiều TS đánh giá rất tích cực. Tính độc đáo, sáng tạo của hệ thống không chỉ dừng lại ở một thiết bị dịch thủ ngữ sang văn bản và lời nói bằng tiếng Việt mà còn hỗ trợ học tập thủ ngữ vô cùng hiệu quả.



Hình 1. Sơ đồ hoạt động dự án của nhóm sinh viên HCMUT

Trên tinh thần đó, nhóm em đã ứng dụng lý thuyết nhận dạng trên để áp dụng vào việc nhận dạng thủ ngữ giúp chúng ta có thể giao tiếp được với người khiếm thanh, khiếm thính, truyền đạt nội dung thông qua sản phẩm của đề tài này.

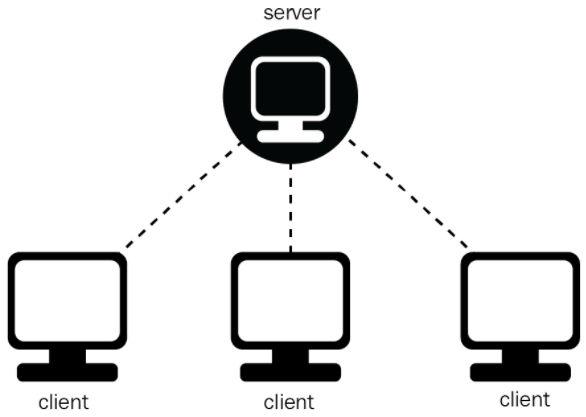
# 2. Giải pháp

## **2.1. Giải pháp về phần cứng và truyền thông**

### **2.1.1. Truyền thông**

**a. Giới thiệu**

Socket [1] là một giao diện lập trình ứng dụng mạng, được dùng để truyền và nhận dữ liệu trên internet. Giữa hai chương trình chạy trên mạng cần có một liên kết giao tiếp hai chiều (two-way communication) để kết nối 2 process trò chuyện với nhau. Điểm cuối (endpoint) của liên kết này được gọi là socket. Mô hình socket điển hình được biểu diễn như hình 2.

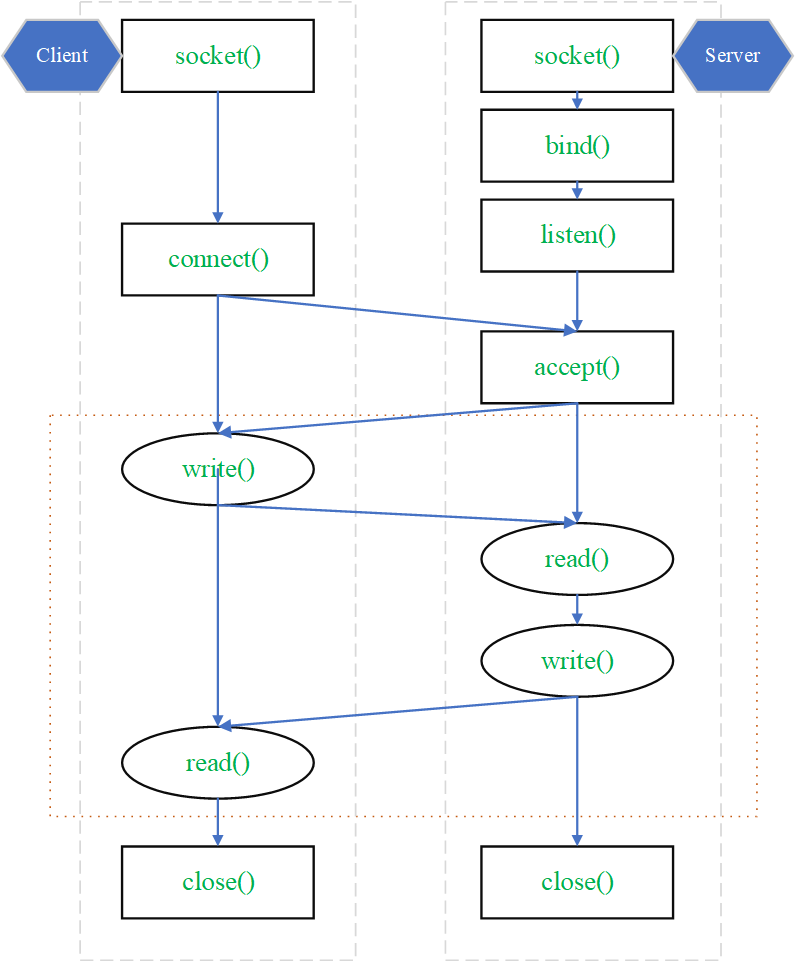


Hình 2. Mô hình socket

Socket gồm 4 loại:

* Stream Socket: Dựa trên giao thức TCP, việc truyền dữ liệu chỉ được thực hiện giữa 2 quá trình đã thiết lập kết nối. Việc thiết lập sẽ đảm bảo dữ liệu truyền đi một cách đáng tin cậy và đúng thứ tự. Để thiết lập kết nối, 1 trong 2 tiến trình này phải đợi tiến trình kia yêu cầu kết nối. Mô hình Client và Server được triển khai dựa trên TCP.
* Datagram Socket: Dựa trên giao thức UDP, việc truyền dữ liệu không yêu cầu có sự thiết lập kết nối giữa 2 quá trình. Vì thế nên tốc độ giao thức khá nhanh, thuận tiện cho việc truyền dữ liệu nhanh như chat… Tuy việc, UDP không đảm bảo dữ liệu truyền đến nơi nhận 1 cách tin cậy và toàn vẹn, có thể không đúng trình tự và lặp lại.
* Web Socket: là công cụ hỗ trợ kết nối qua lại giữa Client và Server trên Internet. Không chỉ dùng riêng cho ứng dụng web, loại này có thể áp dụng với bất kỳ mục đích trao đổi thông tin khác. Cần đảm bảo hoạt động phải diễn ra trên không gian mạng
* Unix Socket: Hỗ trợ trao đổi giữa các ứng dụng khác nhau trên cùng máy tính. Đặc điểm là tránh được các bước kiểm tra và routing, giúp quá trình truyền tin dễ dàng hơn nhiều. Nhược điểm là hệ thống sẽ không thể kết nối khi ứng dụng nằm trên những máy chủ khác nhau

Nguyên lý hoạt động của socket là thiết lập kết nối giữa 2 tiến trình và bắt đầu trao đổi dữ liệu cho nhau. Sơ đồ hoạt động giữa client và server trong một mô hình socket được biểu diễn như hình 3.



Hình 3. Nguyên lý hoạt động Client Server

* Đầu tiên cần tạo ra một máy chủ bằng cách mở một socket
* Sau khi mở socket, ta cần liên kết nó với một host hoặc một máy và một port - Bind()
* Tiếp theo Server sẽ lắng nghe trên port đó - Listen()
* Nhận được yêu cầu kết nối từ Client gửi tới Server – Connect()
* Server sẽ accept yêu cầu từ Client và thiết lập kết nối giữa 2 tiến trình – Accept()
* Trong quá trình kết nối, cả hai có thể gửi và nhận tin – Read() | Write()
* Sau khi hoàn thành đưa tin, ta đóng kết nối lại – Close()

**b. Module Socket trong Python**

Python cung cấp module socket giúp dễ dàng thực hiện kết nối client và server để giao tiếp với nhau.

Tạo một socket:

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

* Tham số đầu tiên là AF (Address Family): tạo kiểu thiết lập kết nối. Python hỗ trợ 3 tham số:
  + AF\_INET: Ipv4
  + AF\_INET6: Ipv6
  + AF\_UNIX
* Tham số thứ hai là Socket Type: thiết lập giao thức kết nối
* SOCK\_STREAM: giao thức TCP
* SOCK\_DGRAM: giao thức UDP

**Một số phương thức của đối tượng socket:**

|  |  |
| --- | --- |
| sock.bind(HOST, PORT) | Đăng ký tên cho socket, ràng buộc địa chỉ vào socket |
| sock.listen(2) | Cho socket lắng nghe tới tối đa 2 kết nối |
| client, addr = sock.accept() | Client gửi lời mời và server chấp nhận kết nối. Một socket mới client được tạo ra |
| data = client.recv(1024) | Nhận gói dữ liệu |
| str\_data = data.decode("utf8") | Phân tích gói dữ liệu vừa nhận |
| sock.sendall(bytes(msg, "utf8")) | Gửi dữ liệu thông qua giao thức TCP |
| sock.close() | Đóng một kết nối |

### **2.1.2. Phần cứng**

Các linh kiện phần cứng được sử dụng:

|  |  |  |
| --- | --- | --- |
| **Tên linh kiện** | **Hình ảnh** | **Thông số, hoạt động** |
| **Webcam Rappo C260** | Binai Shop: Rapoo C260 - Webcam Họp Trực Tuyến Phân Giải HD 1080p, Góc Siêu  Rộng 95°, Tự Động Lấy Nét, Giảm Tiếng Ồn Kép | + Độ phân giải: Full HD 1080p  + Góc camera: 80 độ  + Tầm nhìn xa hồng ngoại: không  + Kết nối: USB 3.0 & 2.0  + Kích thước:7.5cm x4.7cm x 4.5cm  + Khối lượng: 97.5g |
| **Raspberry  Pi 3** |  | + Bộ vi xử lý 64-bit quad-core ARM Cortex-A53  + Mạng không dây 802.11 b/g/n Bluetooth 4.1  + Bộ nhớ LPDDR2 1GB  + Nguồn 2.5A cổng MicroUSB  + 1 cổng Ethernet 10/100  + 1 cổng HDMI  + 1 Jack cắm + + RCA 4 cổng + USB 2.0  + Khe cắm thẻ nhớ MicroSD |
| **LCD 16x2** | LCD Display (16×2) – Perintang.com | + Điện áp MAX: 7V  + Điện áp MIN: -0.3V  + Hoạt động ổn định: 2.7-5.5V  + Điện áp ra mức cao: > 2.4V  + Điện áp ra mức thấp: < 0.4V  + Dòng điện cấp nguồn: 350uA-600uA |
| **Nút nhấn 4 chân** | Nút nhấn 4 chân 6x6x4.3mm DIP( bộ 5 con) | Shopee Việt Nam | + Kích thước: 6x6x4.3mm  + 4 chân vuông |

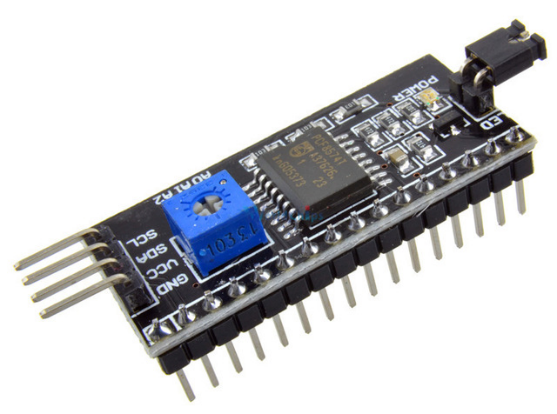
Chi phí phần cứng:

|  |  |
| --- | --- |
| **Tên linh kiện** | **Đơn giá** |
| 1 x Raspberry Pi 3 | 930.000 VNĐ |
| 1 x Webcam Rapoo C260 | 500.000 VNĐ |
| 1 x Thẻ nhớ 32GB | 140.000 VNĐ |
| 1 x LCD 16x2 | 48.000 VNĐ |
| 6 x Nút nhấn 4 chân | 30.000 VNĐ |
| 1 x Hộp Mica đóng gói sản phẩm | 55.000 VNĐ |

### **2.1.3. Nguyên lý mắc mạch**

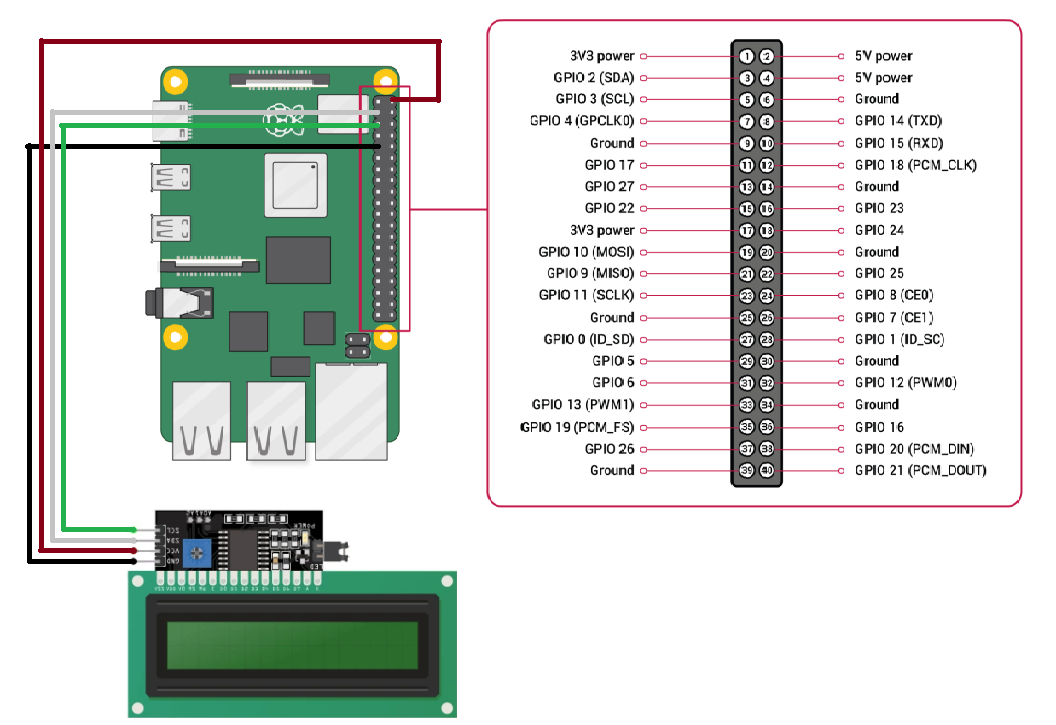
Một LCD 16x2 gồm 16 chân trong đó có 8 chân dữ liệu (D0-D7), 3 chân điều khiển (RS, RW, EN) và 5 chân còn lại dùng để cấp nguồn và đèn nền. Vì có nhiều chân nên có thể gây khó khăn trong quá trình nối dây cũng như chiếm dụng nhiều chân trên vi điều khiển.

Để thuận tiện, ta sẽ sử dụng module I2C LCD. Thay vì mất 6 chân vi điều khiển để kết nối LCD 16x2 (RS, EN và 4 chân dữ liệu) thì với module I2C, ta chỉ tốn 2 chân (SCL, SDA) để giao tiếp



Hình 4. Module I2C LCD

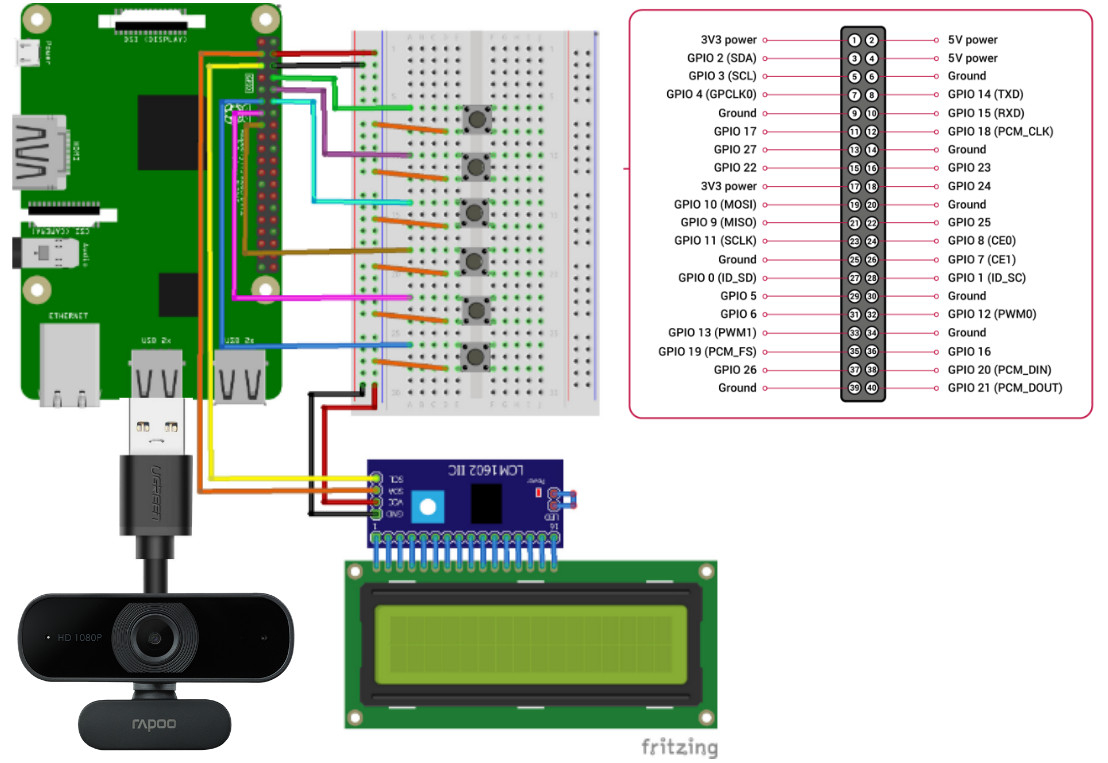
Một I2C LCD có 4 chân (GND, VCC, SDA, SCL). Vi điều khiển Raspberry Pi 3 [2] đã hỗ trợ sẵn 4 chân trên nên ta chỉ cần mắc nối từng chân trên I2C tương ứng với chân vi điều khiển [3]. Sơ đồ mắc mạch được thể hiện ở hình 6



Hình 5. Sơ đồ mắc mạch LCD và Raspberry Pi 3

Sau khi mắc thành công, LCD 16x2 sẽ sáng đèn (vàng hoặc xanh dương)

Sơ đồ mắc mạch tổng được mô tả như hình sau:



Hình 6. Sơ đồ mắc mạch tổng thể

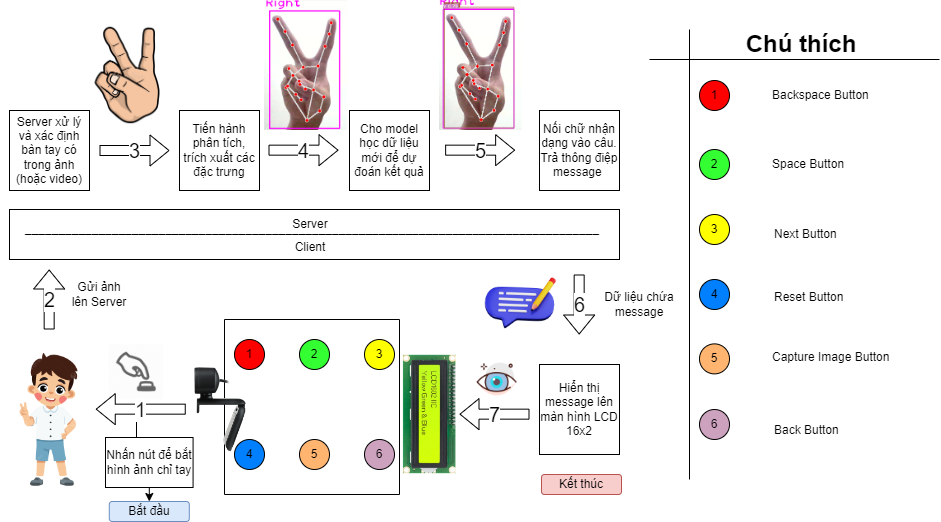
## **2.2. Giải pháp về phần mềm**

### **2.2.1. Luồng xử lý chương trình**

Chương trình chia làm ba giai đoạn chính:

* **Giai đoạn 1**: từ client, bắt hình ảnh cử chỉ tay và gửi lên server.
* **Giai đoạn 2**: tại server, tiến hành nhận dạng mẫu tay, xử lý phân tích và trả về kết quả dự đoán
* **Giai đoạn 3**: Gửi thông điệp message về client và hiển thị dữ liệu lên màn hình LCD 16x2

Sơ đồ làm việc của chương trình được tóm tắt ở hình 7 sau



Hình 7. Sơ đồ tổ chức làm việc của chương trình

### **2.2.2. Mạng CNN và VGG16**

Trong mạng neural, mô hình mạng neural tích chập CNN [4] (Convolutional Neural Network) là một trong những mô hình Deep Learning để nhận dạng và phân loại hình ảnh. Trong đó, xác định đối tượng và nhận dạng là một trong số những lĩnh vực được ứng dụng rộng rãi .

Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Các layer liên kết được với nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.

Diagram, engineering drawing

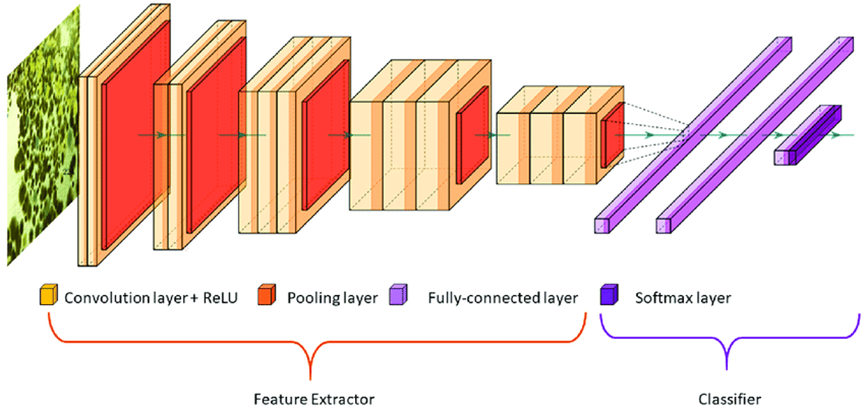
Description automatically generated

Hình 8. Mô hình hoạt động CNN

Trong mô hình CNN có 2 khía cạnh cần quan tâm là **tính bất biến** (Location Invariance) và **tính kết hợp** (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các góc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể

Pooling layer sẽ tạo tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter.

**Mạng VGG16** [5] được coi là mô hình CNN nâng cao AlexNet bằng cách thay thế các bộ lọc có kích thước hạt nhân khổng lồ (11 và 5 trong hai lớp phức hợp đầu tiên) có nhiều bộ lọc kích thước hạt nhân 3\*3 tuần tự. 16 trong VGG16 chỉ ra rằng nó có 16 lớp với các trọng lượng liên quan. Việc tổng hợp không gian được thực hiện bằng cách sử dụng 5 lớp tổng hợp tối đa theo sau một vài lớp tích chập. Ở phần cuối nó có 2 lớp được kết nối đầy đủ và ngay sau đó là softmax cho đầu ra.



Hình 9. Cấu trúc mạng VGG16

**2.2.4. Các thư viện hỗ trợ**

**Cvzone:** thư viện giúp hỗ trợ nhận dạng và xác định bàn tay. Ngoài ra thư viện này hoạt động rất tốt trong việc tái hiện cấu trúc skeleton của các đối tượng như bàn tay, hình dáng con người v.v…

**Socket:** thư viện giúp lập trình ứng dụng socket.

**Threading:** thư viện hỗ trợ tạo và xử lý các tác vụ đa luồng

**Numpy:** thư viện hỗ trợ xử lý mảng đa chiều, các ma trận. Nó được hỗ trợ các cấu trúc dữ liệu mạnh mẽ để tính toán hiệu quả các mảng và ma trận đa chiều

**OpenCV:** là một thư viện mã nguồn mở hàng đầu cho thị giác máy tính, xử lý ảnh và học máy.

### **2.2.3. Các bước thực hiện**

**\* Khởi tạo dataset**

* Khởi tạo data bằng cách chụp ảnh bàn tay đang thực hiện các ký tự.
* Sử dụng thư viện HandDetector của cvzone để trích xuất đặc trưng bàn tay và lưu vào thư mục Data. Các ảnh được gắn nhãn đầy đủ bằng tên folder tương ứng.

**\* Train Model**

* Thực hiện thao tác tiền xử lý ảnh như phóng to, xoay… để làm giàu dữ liệu, giúp việc train tốt hơn
* Chia dữ liệu thành 2 phần: huấn luyện (train) và kiểm thử (val) với tỉ lệ 8:2
* Khởi tạo model VGG16 kích thước (128, 128, 3) và thêm các lớp Fully Connect và Dropout sử dụng hàm relu, softmax
* Tiến hành train model qua 100 epochs, lưu model với điều kiện loss thấp nhất và accuracy cao nhất.

**\* Nhận diện thủ ngữ**

* Trên server, chạy file server\_VGG16.py để bật server. Server sẽ khởi tạo model và thiết lập socket để lắng nghe
* Trên client, chạy file client.py để bật camera. Nhấn nút 5 để chụp ảnh gửi cho server
* Server sẽ nhận ảnh từ client, đưa vào model để nhận diện và trả kết quả về cho client
* Client sẽ nhận kết quả từ server, hiển thị kết quả lên màn hình LCD

# 3. Kết quả

## **3.1. Tập dữ liệu ( Dataset)**

**- Nguồn gốc dữ liệu: ký hiệu tay của các sinh viên trong trường**

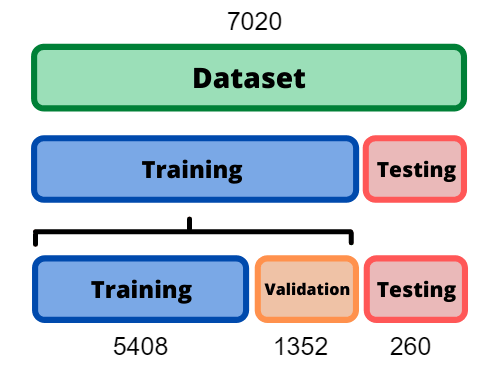
**- Cách thức thu thập gồm các bước sau:**

* Khởi tạo thư mục Data chứ các folder label gồm nhãn của 26 ký tự từ A đến Z
* Tiến hành lấy ảnh từng người, với mỗi ký hiệu là 20 ảnh đầu vào
* Gán nhãn tự động cho từng ảnh trong từng folder label

**- Tính chất của dữ liệu:**

* Độ phân giải: 300 x 300 pixels
* Dải giá trị: 26 giá trị là các chữ cái từ A đến Z

**- Tổ chức tập dữ liệu:**

* Tổng số ảnh đầu vào: 26 x 270 = 7020 ảnh được chia thành 3 tập dữ liệu nhỏ bao gồm:
* Tập huấn luyện
* Tập xác nhận
* Tập kiểm thử
* Trước khi dùng 1352 ảnh để đánh giá, trích 5408 ảnh đầu tiên để phục vụ cho huấn luyện. Tập dữ liệu kiểm thử được tạo bằng cách sử dụng 260 ảnh từ tập dữ liệu.
* Sơ đồ tập dữ liệu hình ảnh đầu vào được biểu thị theo Hình 10 dưới đây:

Hình 10. Tập dataset chuẩn bị train model

## **3.2. Các công cụ và thư viện đã dùng**

* Công cụ:
* Train Model: Google Colab
* Code Editor: Visual Studio Code
* Diagram: Fritzing, Diagrams
* Connect to Raspberry: VNC Viewer, WinSCP
* Thư viện :
* HandDetector: Nhận diện tay, mô phỏng cấu trúc xương tay
* Socket: Giao tiếp giữa Client và Server để gửi ảnh lên và trả thông điệp về.
* OpenCV: Chụp ảnh từ camera để train và nhận diện
* VGG16: Model chính dùng để nhận diện ký hiệu thủ ngữ
* Numpy, math, time, os, Dense, keras, …

**3.3. Điều kiện thực nghiệm**

**-** Độ chính xác khi nhận dạng ký hiệu thủ ngữ có thể bị thay đổi dưới nhiều môi trường ánh sáng và nhiễu khác nhau

## **3.4. Kết quả nhận diện**

Số lần train: 100 epochs

Kích thước batch: 64

Độ chính xác: 91%

Độ chính xác và hàm mất mát của Model qua 100 lần train được biểu diễn như Hình 11:

A picture containing shape

Description automatically generated

Hình 11. Kết quả train model

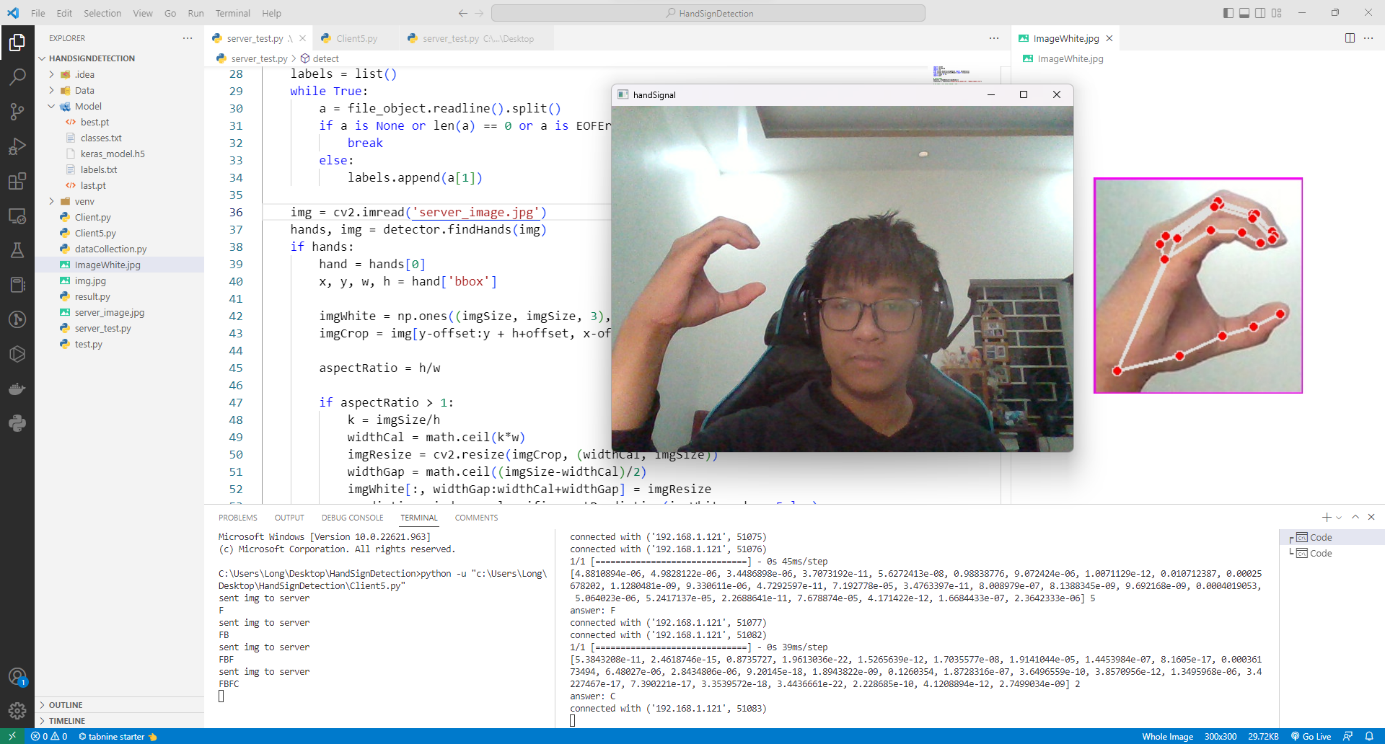
Tiến hành kiểm thử Model bằng cách cho tập kiểm thử qua Model để nhận diện. Kết quả nhận diện được thể hiện ở ma trận nhầm lẫn ở Hình 12:

Chart, scatter chart

Description automatically generated

Hình 12. Ma trận nhầm lẫn

* Kết quả đánh giá hệ thống:
  + Những ký hiệu nhận diện sai nhiều nhất: U
  + Những ký hiệu nhận diện chính xác nhất: A, O, P, Q, S, T, V, W, X, Y, Z
* Kết luận: Mô hình dự đoán khá chính xác, cho kết quả đúng với nhiều điều kiện môi trường, ánh sáng khác nhau. Tuy nhiên, vẫn còn sự nhầm lẫn giữa các ký tự như: U với P, R, V hoặc B với W, … do tập dữ liệu train vẫn chưa giàu, điều kiện ánh sáng cũng như các yếu tố nhiễu vẫn còn ít. Vì vậy khi thực nghiệm trên nhiều môi trường khác nhau thì vẫn xảy ra tình trạng nhận diện sai ký hiệu thủ ngữ.
* Kết quả nhận diện ký tự thủ ngữ chạy ở Server như Hình 13:



Hình 13. Nhận diện thủ ngữ

* Hiển thị kết quả thông điệp nhận được lên LCD như Hình 14:

**A picture containing indoor

Description automatically generated**

Hình 14. Hiển thị kết quả lên màn hình LCD 16x2

# 4. Kết luận và hướng phát triển

## **4.1. Kết luận**

* Độ chính xác khi nhận diện khá cao tuy nhiên một số ký tự có khẩu hình tay khá giống nhau nên việc nhầm lẫn vẫn còn xuất hiện
* Vấn đề ánh sáng cũng gây ảnh hưởng đến việc nhận diện
* Dùng socket nên kết quả xử lý được trả về và hiển thị khá nhanh

## **4.2. Hướng phát triển**

* Thu thập thêm nhiều dữ liệu dataset của nhiều người, nhiều khẩu hình tay khác nhau với nhiều môi trường khác nhau để tăng độ chính xác khi nhận diện
* Triển khai sản phẩm trên Mobile App tận dụng phần cứng như: camera, màn hình, loa,… giúp giảm chi phí và gọn nhẹ hơn trong việc nhận diện thủ ngữ
* Thực hiện train model với ngôn ngữ thủ ngữ tiếng Việt

# 5. Danh mục tài liệu tham khảo

[1] Lập trình socket bằng Python

URL: <https://viblo.asia/p/lap-trinh-socket-bang-python-jvEla084Zkw>

Ngày truy cập : 28/10/2022

[2] Tìm hiểu tổng quan Raspberry Pi 3

URL: <https://viblo.asia/p/raspberry-pi-3-chiec-may-tinh-nho-nhung-co-vo-3P0lPzggKox>

Ngày truy cập: 29/09/2022

[3] Tìm hiểu các chân GPIO của Raspberry Pi 3

URL: <https://dientuviet.com/tim-hieu-ve-cac-chan-gpio-cua-raspberry-pi/>

Ngày truy cập: 29/09/2022

[4] Tìm hiểu về mạng tích chập CNN

URL: <https://viblo.asia/p/deep-learning-tim-hieu-ve-mang-tich-chap-cnn-maGK73bOKj2>

Ngày truy cập : 07/10/2022

[5] Tìm hiểu train model bằng VGG16

URL : <https://aicurious.io/posts/2019-10-03-xay-dung-bo-phan-loai-vgg-16-voi-keras-ap-dung-phan-loai-cho-meo/>

Ngày truy cập: 7/10/2022