

TRƯỜNG ĐẠI HỌC BÁCH KHOA KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN

PBL5 – ĐỒ ÁN KỸ THUẬT MÁY TÍNH

TÊN ĐỀ TÀI: XÂY DỰNG HỆ THỐNG NHẬN DIỆN NGÔN NGỮ KÝ HIỆU

Giảng viên hướng dẫn: TS. Phạm Công Thắng

NHÓM SINH VIÊN THỰC HIỆN	LỚP HỌC PHẦN	MSSV
Nguyễn Huynh	22T_KHDL	102220024
Hoàng Đức Mạnh	22T_KHDL	102220029
Nguyễn Thị Thuý Hằng	22T_DT5	102220228
Nguyễn Thị Mỹ Lệ	22T_DT5	102220238

Đà Nẵng, ngày 06/2025

MỤC LỤC

DANH MỤC HÌNH ẢNH.....	5
DANH MỤC BẢNG BIỂU.....	6
TÓM TẮT ĐỒ ÁN.....	7
BẢNG PHÂN CÔNG NHIỆM VỤ.....	8
BẢNG TIMELINE DỰ ÁN	9
I. TỔNG QUAN VỀ ĐỀ TÀI.....	10
1. Mục tiêu và ý nghĩa thực tiễn	10
1.1. Lý do chọn đề tài.....	10
1.2. Mục tiêu dự án	11
2. Ứng dụng	11
2.1. Ý nghĩa đối với xã hội.....	12
2.2. Ý nghĩa đối với bản thân.....	12
II. CƠ SỞ LÝ THUYẾT	13
1. Tổng quan về Deep Learning.....	13
1.1. Deep Learning là gì?	13
1.2. Convolutional Neural Network (CNN).....	14
2. Xử lý ảnh (Image Processing) và Thị giác máy tính (Computer Vision).....	16
3. Xử lý tín hiệu và dữ liệu thời gian thực	16
4. Kiến trúc hệ thống Client-Server	17
III. TRIỂN KHAI DỰ ÁN.....	18
1. Các vấn đề cần giải quyết và đề xuất giải pháp	18
1.1. Các vấn đề	18
1.2. Đề xuất giải pháp tổng quan	19
2. Chuẩn bị dữ liệu.....	20
2.2. Tiền xử lý dữ liệu	20

PBL5: Dự án kỹ thuật máy tính

3. Chuẩn hoá ảnh	20
4. Trục quan hoá dữ liệu	21
5. Định hình lại dữ liệu	21
6. Mã hóa nhãn.....	21
7. Xây dựng mô hình	22
7.1. Các thành phần chính của mô hình	22
7.2. Tham số huấn luyện mô hình	25
7.3. Tăng cường dữ liệu (Data Augmentation)	26
8. Xây dựng Server xử lý và Web hiển thị	26
8.1. Mô hình Client-Server.....	26
8.2. Các thành phần chính của Server	27
9. Giải pháp.....	28
9.1. Giải pháp phần cứng và truyền thông	28
9.2. Giải pháp AI/KHDL.....	31
9.3. Giải pháp phần mềm	33
10. Kết quả.....	35
10.1. Tập dữ liệu	35
10.2. Huấn luyện mô hình	36
10.3. Kết quả huấn luyện mô hình	36
10.4. Sản phẩm phần cứng	37
10.5. Kết quả nhận diện.....	38
IV. KẾT LUẬN	39
1. Đánh giá.....	39
2. Hướng phát triển	39
2.1. Nâng cao độ chính xác và khả năng nhận diện trong điều kiện khó khăn	39
2.2. Mở rộng tính năng nhận diện	40

PBL5: Dự án kĩ thuật máy tính

2.3. Tích hợp và tối ưu trên các thiết bị	40
2.4. Ứng dụng và phát triển.....	40
V. TÀI LIỆU THAM KHẢO	41

DANH MỤC HÌNH ẢNH

Hình 1. Bảng chữ cái ngôn ngữ ký hiệu	10
Hình 2. Mối quan hệ AL, ML, DL	13
Hình 3. Cấu trúc mô hình CNN.....	14
Hình 4. Lựa chọn tham số cho mạng CNN	15
Hình 5 Ngôn ngữ ký hiệu dùng cho một từ hoàn chỉnh (Tôi yêu bạn)	18
Hình 6. Tổng quan cấu trúc	24
Hình 7. Sơ đồ tổng quan hệ thống	28
Hình 8. Cấu trúc mô hình	32
Hình 9. Sơ đồ tuần tự của hệ thống nhận diện ngôn ngữ ký hiệu	34
Hình 10. Ảnh minh chứng dữ liệu tự thu thập.....	35
Hình 11. Biểu đồ độ chính xác huấn luyện	36
Hình 12. Biểu đồ hàm mất mát huấn luyện	37
Hình 13 Phần cứng của hệ thống nhận diện	37
Hình 14 Kết quả nhận diện được hiển thị lên web	38

DANH MỤC BẢNG BIỂU

Bảng 1: Bảng phân công nhiệm vụ.....	8
Bảng 2. Bảng timeline dự án	9
Bảng 3. Bảng đề xuất giải pháp.....	19
Bảng 4. Bảng chứa thông số huấn luyện chính	26
Bảng 5: Danh sách các thiết bị phần cứng	31

TÓM TẮT ĐỒ ÁN

Trong thực tế, việc giao tiếp với người khiếm thính hoặc sử dụng cử chỉ trong điều khiển thiết bị vẫn còn gặp nhiều hạn chế do thiếu các hệ thống hỗ trợ thông minh, dẫn đến khó khăn trong sinh hoạt hằng ngày, đặc biệt là trong môi trường gia đình hoặc khu vực sinh sống hiện đại.

Nhóm đã thiết kế và xây dựng một hệ thống nhận diện ngôn ngữ ký hiệu thông minh, sử dụng Raspberry Pi làm bộ điều khiển trung tâm, kết hợp với mô hình trí tuệ nhân tạo (AI) để nhận dạng cử chỉ tay theo ngôn ngữ ký hiệu. Hệ thống cho phép người dùng sử dụng các ký hiệu tay để giao tiếp, điều khiển thiết bị gia dụng, mở/đóng cửa, hoặc gửi tín hiệu cảnh báo, đặc biệt phù hợp cho người khiếm thính hoặc môi trường cần giao tiếp không lời.

Ngoài ra, hệ thống được tích hợp với giao diện website, cho phép người dùng theo dõi trạng thái hệ thống, cấu hình cử chỉ, và điều khiển từ xa một cách dễ dàng và thuận tiện.

Hệ thống hoạt động ổn định, mô hình AI nhận diện ký hiệu tay cho độ chính xác cao, thời gian phản hồi nhanh, xử lý nhẹ phù hợp với phần cứng hạn chế như Raspberry Pi. Giao diện website thân thiện, dễ sử dụng, hỗ trợ người dùng trong việc điều khiển và giám sát thiết bị từ xa.

Mô hình có khả năng mở rộng và ứng dụng cao trong các lĩnh vực như nhà thông minh, hỗ trợ người khuyết tật, giáo dục ngôn ngữ ký hiệu, và các môi trường đòi hỏi giao tiếp không lời.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Đánh giá
Nguyễn Huỳnh	Trainning model	Hoàn thành
	Tạo slide	Hoàn thành
	Viết báo cáo	Hoàn thành
Hoàng Đức Mạnh	Code server	Hoàn thành
	Thiết lập phần cứng	Hoàn thành
	Code web hiển thị	Hoàn thành
	Viết báo cáo	Hoàn thành
Nguyễn Thị Thúy Hằng	Viết báo cáo	Hoàn thành
	Thiết lập phần cứng	Hoàn thành
	Code Raspberry Pi Client	Hoàn thành
Nguyễn Thị Mỹ Lệ	Viết báo cáo	Hoàn thành
	Thiết lập phần cứng	Hoàn thành
	Code Raspberry Pi Client	Hoàn thành

Bảng 1: Bảng phân công nhiệm vụ

BẢNG TIMELINE DỰ ÁN

STT	Công việc	Thời gian thực hiện
1	Phân tích yêu cầu	08/01 - 13/01
2	Xác định mục đích, mục tiêu	14/01 - 18/01
3	Tìm tài liệu tham khảo	19/01 - 25/01
4	Phân công nhóm	26/01 - 28/01
5	Mua & lắp ráp phần cứng (module camera, Raspberry Pi)	29/01 – 07/02
6	Cấu hình phần cứng và camera	08/02 - 14/02
7	Tiền xử lý dữ liệu	15/02 - 22/02
8	Xử lý dữ liệu đầu vào mô hình	23/02 - 01/03
9	Huấn luyện mô hình	02/03 - 26/03
10	Xây dựng Server xử lý dữ liệu, giao diện Web	27/03 - 10/04
11	Xây dựng kết nối truyền nhận dữ liệu (camera – Pi – Server – Pi - Speaker)	11/04 - 25/04
12	Tích hợp mô hình vào hệ thống	26/04 - 08/05
13	Kiểm thử hệ thống	09/05 - 20/05
14	Tối ưu và sửa lỗi toàn hệ thống	21/05 - 31/05
15	Viết & hoàn thiện báo cáo	01/06 - 12/06

Bảng 2. Bảng timeline dự án

I. TỔNG QUAN VỀ ĐỀ TÀI

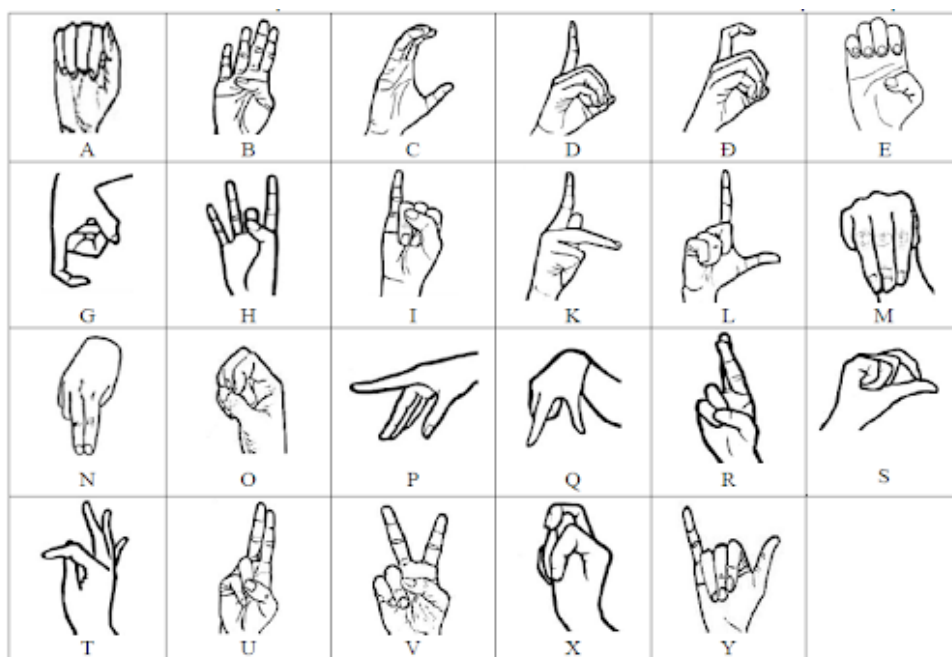
1. Mục tiêu và ý nghĩa thực tiễn

1.1. Lý do chọn đề tài

Ngôn ngữ đóng vai trò thiết yếu trong đời sống con người, không chỉ là công cụ giao tiếp mà còn là cầu nối tri thức, cảm xúc và văn hóa. Nó giúp chúng ta diễn đạt ý tưởng, chia sẻ trải nghiệm và xây dựng các mối quan hệ xã hội. Đặc biệt, ngôn ngữ còn là phương tiện để hình thành tư duy, bởi mọi suy nghĩ đều cần được biểu đạt thông qua hệ thống ký hiệu ngôn ngữ.

Tuy nhiên, đối với cộng đồng người khiếm thính, việc sử dụng ngôn ngữ nói gặp nhiều rào cản do hạn chế về khả năng nghe. Để khắc phục điều này, họ giao tiếp chủ yếu thông qua ngôn ngữ ký hiệu – một hệ thống sử dụng cử chỉ tay, biểu cảm khuôn mặt và chuyển động cơ thể để truyền tải thông tin thay cho âm thanh.

Trong thời đại công nghệ số phát triển mạnh mẽ như vậy, nhu cầu hỗ trợ giao tiếp cho người khiếm thính ngày càng trở nên cấp thiết. Việc ứng dụng trí tuệ nhân tạo (AI) và các thiết bị nhúng như Raspberry Pi mở ra nhiều cơ hội để xây dựng những hệ thống thông minh, tiết kiệm và dễ triển khai. Đề án này tập trung vào việc phát triển một hệ thống nhận diện ngôn ngữ ký hiệu nhằm hỗ trợ người khiếm thính trong giao tiếp hàng ngày, giúp tăng cường khả năng tương tác giữa họ với cộng đồng.



Hình 1. Bảng chữ cái ngôn ngữ ký hiệu

1.2. Mục tiêu dự án

Hệ thống hướng đến khả năng xử lý thời gian thực bằng cách kết hợp phần cứng module camera (đảm nhận thu hình liên tục 30 khung hình/giây) và máy chủ Raspberry Pi (thực hiện nhận diện và phản hồi). Dữ liệu video từ camera sẽ được gửi đến Raspberry Pi, sau đó gửi đến Server - nơi mô hình học sâu – cụ thể là mô hình CNN được huấn luyện trên tập dữ liệu cử chỉ tay – tiến hành phân tích, nhận diện và trả về kết quả tương ứng.

Mô hình được xây dựng dựa trên mạng nơ-ron tích chập (CNN) cải tiến, có khả năng học và phân biệt các đặc trưng phức tạp trong từng khung hình của ngôn ngữ ký hiệu. Thông qua quá trình huấn luyện với tập dữ liệu đã chuẩn hóa, mô hình có thể xử lý chuỗi khung hình liên tiếp và đưa ra dự đoán chính xác với độ trễ tối thiểu. Kết quả nhận diện sẽ được phát ra dưới dạng âm thanh thông qua loa tích hợp, giúp quá trình giao tiếp trở nên trực quan và gần gũi hơn.

Hệ thống hướng tới việc hoạt động ổn định, chi phí thấp, dễ triển khai và ứng dụng thực tế, đặc biệt trong môi trường giáo dục, hỗ trợ người khiếm thính hoặc các tình huống giao tiếp cần tính thời gian thực.

2. Ứng dụng

Ứng dụng nhận diện và dịch tự động ngôn ngữ ký hiệu đóng vai trò quan trọng trong việc hỗ trợ người khiếm thính và tăng cường khả năng giao tiếp giữa họ với cộng đồng xung quanh. Bằng cách nhận diện các cử chỉ tay và chuyển đổi chúng thành chữ cái hoặc văn bản, ứng dụng không chỉ tạo điều kiện cho việc trao đổi thông tin trong cuộc sống hằng ngày mà còn hỗ trợ quá trình học ngôn ngữ ký hiệu cơ bản.

Người dùng có thể sử dụng ứng dụng để giao tiếp những từ trong ngôn ngữ ký hiệu, nhờ đó nâng cao khả năng giao tiếp cá nhân một cách linh hoạt. Bên cạnh đó, ứng dụng còn có giá trị trong lĩnh vực giáo dục, nơi nó có thể được sử dụng như một công cụ học tập giúp giáo viên và học sinh luyện tập nhận diện các ký hiệu tay và phát âm tương ứng, góp phần tăng cường sự hiểu biết và tương tác giữa các đối tượng sử dụng.

Đặc biệt, với khả năng hoạt động thời gian thực thông qua camera và thiết bị nhúng như Raspberry Pi, ứng dụng càng trở nên hữu ích trong các tình huống đòi hỏi phản hồi nhanh và chính xác.

2.1. Ý nghĩa đối với xã hội

Ứng dụng giúp tạo cầu nối giao tiếp cho người khiếm thính, nâng cao khả năng hòa nhập của họ vào các hoạt động xã hội và tăng cường nhận thức cộng đồng về việc hỗ trợ nhóm người này.

2.2. Ý nghĩa đối với bản thân

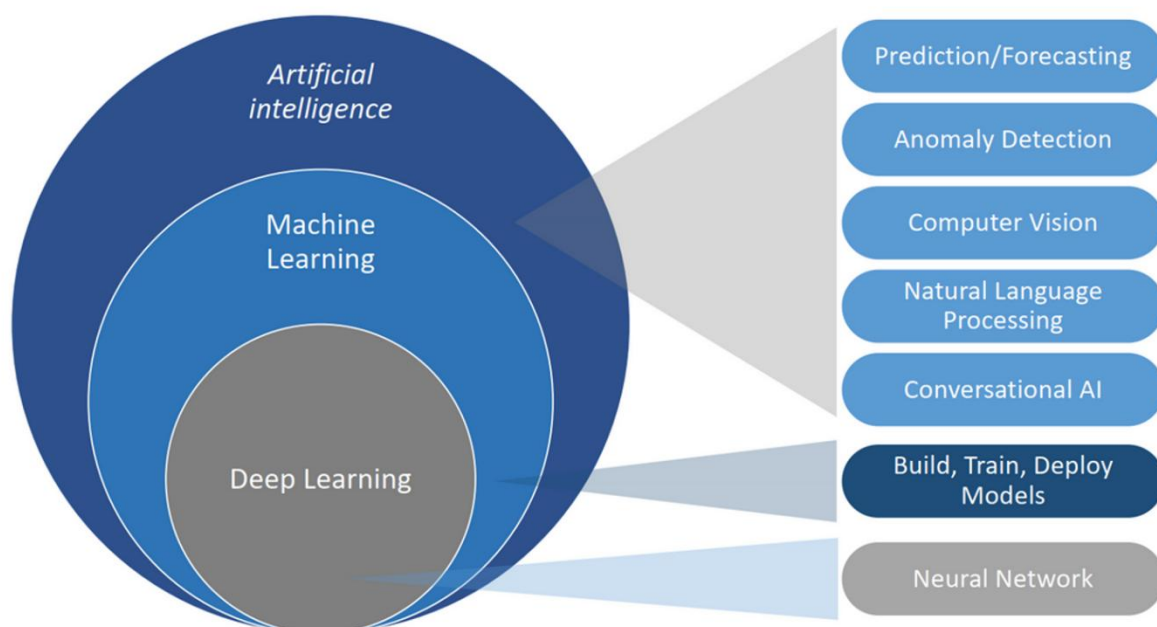
Đồ án cung cấp cơ hội áp dụng kiến thức về học máy và xử lý hình ảnh, giúp nâng cao kỹ năng giải quyết vấn đề và làm việc với các công nghệ tiên tiến, là nền tảng cho sự phát triển sự nghiệp trong lĩnh vực công nghệ hỗ trợ.

II. CƠ SỞ LÝ THUYẾT

1. Tổng quan về Deep Learning

1.1. Deep Learning là gì?

AI - Artificial Intelligence (Trí Tuệ Nhân Tạo) cụ thể là Machine Learning đã đạt được thành công đáng kể trong những năm gần đây. AI xuất hiện hầu như trong tất cả lĩnh vực của đời sống, dễ dàng nhận thấy trong các ứng dụng như lái xe tự hành, Deep Blue... Machine Learning sau huấn luyện sẽ có thể đưa ra dự đoán chính xác sau khi được cung cấp dữ liệu thay vì lập trình nó theo một sườn cứng để thực hiện nhiệm vụ đó từng bước một. Từ đó đem đến sự tiện lợi, nhanh chóng cho người dùng.



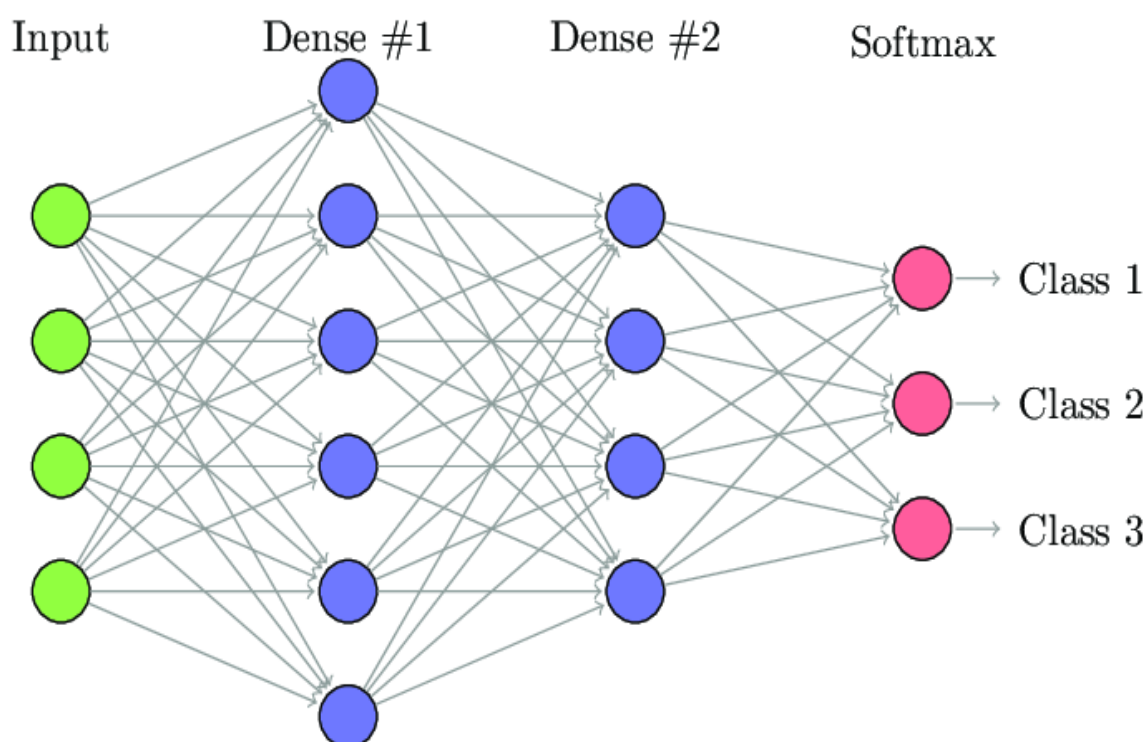
Hình 2. Mối quan hệ AL, ML, DL

AI ngày càng được các nhà đầu tư quan tâm và vươn lên phát triển như vũ bão. Nổi bật trong các đột phá của AI phần nhiều đến Deep learning. Nhờ có Deep learning mà máy tính ngày càng vượt qua hoặc tiến gần đến việc kết hợp hiệu suất của con người. Qua hình 2 ta có thể thấy rõ Deep Learning chỉ là một tập hợp con của Machine Learning. Tuy nhiên Deep Learning có khả năng khác biệt ở một số khía cạnh quan trọng so với Machine Learning nông truyền thống, cho phép máy tính giải quyết một

loạt các vấn đề phức tạp không thể giải quyết được. Trong đó có thể kể đến như khả năng khai thác các tập dữ liệu lớn (Big Data) với độ chính xác cao. [1]

1.2. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) là một thuật ngữ dùng để chỉ mạng nơ ron tích chập. Đây là mô hình có độ chính xác cao và được sử dụng phổ biến trong Deep learning. Ý tưởng đằng sau là chúng ta sẽ cho ảnh đi qua các bộ lọc trước khi huấn luyện mạng Neural. Sau khi đi qua bộ lọc các đặc trưng của ảnh được thể hiện rõ và chúng ta sẽ dùng chúng để nhận diện hình ảnh. Như hình ta có cấu trúc của mô hình CNN.



Hình 3. Cấu trúc mô hình CNN

Đầu vào của mô hình là hình ảnh. Sau đó, dữ liệu đi qua các Convolutional Layer (lớp tích chập), nơi hình ảnh đầu vào được xử lý bằng cách thực hiện tích chập với các Kernel (bộ lọc). Các Kernel trượt từ trái qua phải và từ trên xuống dưới, áp dụng các phép tính với các bước nhảy (stride), padding nếu cần thiết, và sau đó áp dụng hàm kích hoạt phi tuyến (nonlinear activation) lên ma trận hình ảnh.

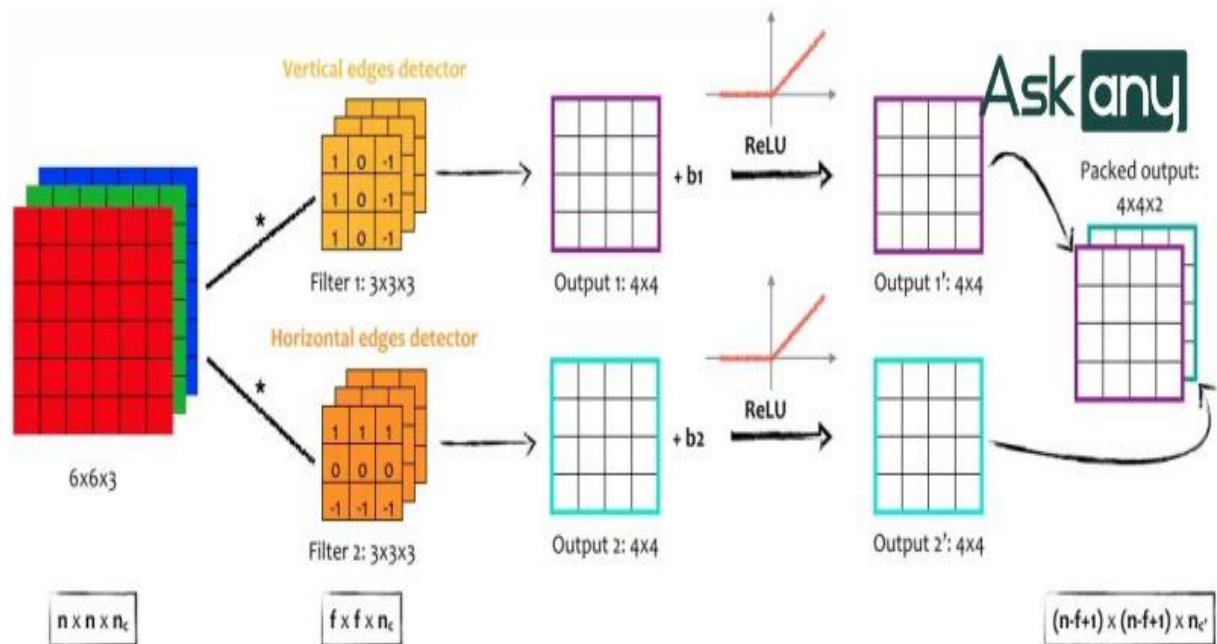
Các hàm kích hoạt phổ biến bao gồm tanh, sigmoid, nhưng ReLU (Rectified Linear Unit) được ưa chuộng hơn nhờ hiệu suất cao. Công thức của ReLU là

PBL5: Dự án kỹ thuật máy tính

$f(x)=\max(0,x)$, đảm bảo đầu ra không âm, phù hợp với việc học các giá trị tuyến tính không âm. Kết quả sau hàm kích hoạt được truyền đến các lớp tiếp theo.

Tiếp theo, quá trình Pooling được áp dụng để giảm kích thước của hình ảnh, giữ lại các thông tin quan trọng. Các loại Pooling phổ biến bao gồm Max Pooling, Average Pooling, và Sum Pooling. Số lượng lớp tích chập có thể được điều chỉnh linh hoạt tùy theo mục đích sử dụng. Cuối cùng, dữ liệu được làm phẳng và đưa vào các lớp kết nối đầy đủ (Fully Connected Layers), tương ứng với Dense #1 và Dense #2 trong hình, trước khi đến lớp đầu ra. Lớp đầu ra sử dụng Softmax để chuyển đổi thành xác suất cho ba lớp (Class 1, Class 2, Class 3), với tổng xác suất bằng 1, giúp mô hình đưa ra dự đoán phân loại.

Để lựa chọn tham số phù hợp cho mạng CNN, bạn cần lưu ý đến một số yếu tố quan trọng bao gồm số lượng convolution layer, kích thước filter, kích thước pooling và số lần train test.



Hình 4. Lựa chọn tham số cho mạng CNN

- Convolution layer: Số lượng lớp càng nhiều, mô hình sẽ càng được cải thiện. Hơn nữa, việc sử dụng nhiều lớp có thể giảm bớt sai lệch và đảm bảo mô hình hoạt động hiệu quả. Thông thường, chỉ cần từ 3 - 5 lớp là đã có thể đạt kết quả tốt.
- Filter size: Kích thước filter phổ biến là 3x3 hoặc 5x5.
- Pooling size: Với những hình ảnh thông thường, kích thước pooling nên sử dụng là 2x2. Nếu xử lý hình ảnh với kích thước lớn hơn, bạn hãy dùng kích thước 4x4.

- Train test: Số lần train test càng được thực hiện nhiều thì càng dễ thu được các tham số tối ưu, giúp mô hình trở nên thông minh và hiệu quả hơn.

2. Xử lý ảnh (Image Processing) và Thị giác máy tính (Computer Vision)

Xử lý ảnh là quá trình chuyển đổi một hình ảnh sang dạng kỹ thuật số và thực hiện các thao tác nhất định để nhận được một số thông tin hữu ích từ hình ảnh đó. Trong lĩnh vực xử lý ảnh và thị giác máy tính, hình ảnh số không chỉ đơn thuần là tập hợp các điểm ảnh mà còn chứa đựng thông tin giá trị về đặc trưng vật thể, hình dạng và chuyển động. Để khai thác triệt để các thông tin này, hệ thống cần thực hiện hàng loạt các thao tác nhằm biến đổi và trích xuất đặc trưng phù hợp với mục tiêu cụ thể của bài toán. [2]. Một số khái niệm cơ bản trong xử lý ảnh:

- Không gian màu: Hình ảnh số thường được biểu diễn trong các không gian màu như RGB (Red, Green, Blue) hoặc HSV (Hue, Saturation, Value). Chuyển đổi giữa các không gian này giúp làm nổi bật các đặc tính phù hợp với bài toán cụ thể (ví dụ: phát hiện màu da tay tốt hơn ở HSV).
- Lọc nhiễu: Ảnh thu nhận từ cảm biến thường chứa nhiễu do ánh sáng, bụi hoặc rung lắc. Các bộ lọc như Gaussian, Median hoặc Bilateral Filter được sử dụng để làm mịn ảnh, loại bỏ chi tiết thừa nhưng vẫn giữ được cạnh.
- Phát hiện chuyển động: Dựa trên sự khác biệt giữa các khung hình liên tiếp, có thể xác định được vùng chuyển động trong ảnh. Kỹ thuật phổ biến gồm Background Subtraction, Optical Flow.
- Xử lý ngưỡng: Áp dụng ngưỡng để phân biệt vùng quan tâm (Region of Interest - ROI) với nền, giúp tách rời đối tượng (như bàn tay) khỏi phần còn lại của ảnh. Có thể dùng ngưỡng cố định hoặc ngưỡng thích nghi (Adaptive Thresholding).

3. Xử lý tín hiệu và dữ liệu thời gian thực

Dữ liệu thời gian thực là thông tin được cập nhật liên tục và có sẵn ngay lập tức hoặc gần như ngay lập tức. Dữ liệu thời gian thực có thể truy cập khi được tạo hoặc thu thập hoặc sau khi phân tích thời gian thực mà không có độ trễ đáng kể. [3]

Dữ liệu hình ảnh thu nhận liên tục từ camera tạo ra chuỗi dữ liệu thời gian thực. Một số khái niệm chính:

PBL5: Dự án kỹ thuật máy tính

- Lọc tín hiệu: Dữ liệu landmark hoặc vector đặc trưng thường chứa nhiễu. Các bộ lọc như Kalman Filter, Moving Average giúp làm mượt, dự đoán chính xác hơn.
- Chuẩn hóa tín hiệu: Để đảm bảo tính đồng nhất, dữ liệu đầu vào cần được chuẩn hóa (scale) về cùng kích thước, tọa độ hoặc hệ quy chiếu.
- Xử lý luồng dữ liệu: Trong các hệ thống real-time, dữ liệu cần được xử lý liên tục với độ trễ thấp. Việc tách luồng (multi-threading) giúp xử lý song song việc thu thập và nhận dạng.

4. Kiến trúc hệ thống Client-Server

Mô hình Client-Server [4] là kiến trúc phổ biến cho các hệ thống IoT và xử lý dữ liệu từ xa:

- Client: Thiết bị đầu cuối (như Raspberry Pi) chịu trách nhiệm thu thập dữ liệu, tiền xử lý nhẹ và gửi lên server.
- Server: Thực hiện các phép tính nặng như nhận dạng, phân loại dựa trên mô hình học sâu.
- Kết nối: Các giao thức TCP/IP hoặc HTTP được sử dụng để đảm bảo truyền dữ liệu ổn định. Các kỹ thuật đồng bộ hóa giúp giữ thứ tự và toàn vẹn dữ liệu.

III. TRIỂN KHAI DỰ ÁN

1. Các vấn đề cần giải quyết và đề xuất giải pháp

1.1. Các vấn đề

- Làm sao để hệ thống có thể nhận diện chính xác các ký hiệu tay trong ngôn ngữ ký hiệu tiếng Việt.
- Làm thế nào để xử lý tín hiệu video theo thời gian thực trên thiết bị có cấu hình thấp như Raspberry Pi.
- Cách thức chuyển đổi ký hiệu tay sang văn bản hoặc giọng nói để hỗ trợ người nghe hiểu nội dung giao tiếp.
- Làm sao để giao diện hệ thống dễ sử dụng, thân thiện với người dùng khiếm thính và có thể mở rộng cho nhiều người dùng.



Hình 5 Ngôn ngữ ký hiệu dùng cho một từ hoàn chỉnh (Tôi yêu bạn)

1.2. Đề xuất giải pháp tổng quan

Nhóm đề xuất xây dựng một hệ thống nhận diện ngôn ngữ ký hiệu sử dụng Raspberry Pi làm bộ điều khiển trung tâm. Hệ thống tích hợp mô hình AI dựa trên học sâu để nhận diện các cử chỉ tay thông qua module Camera Raspberry hoặc webcam, sau đó chuyển đổi thành văn bản hoặc phát âm thanh tương ứng thông qua loa. Giao diện web đơn giản hoặc ứng dụng di động sẽ giúp người dùng theo dõi và tương tác với hệ thống. Giải pháp này vừa tiết kiệm chi phí, vừa dễ triển khai thực tế, góp phần hỗ trợ người khiếm thính hòa nhập tốt hơn với cộng đồng.

Vấn đề	Giải pháp đề xuất
Phần cứng	Raspberry Pi 3 Loa toàn dải 2 inch 4Ω 5W Module CAMERA Raspberry Module MP03 DFPLAYER
Nhận diện ký hiệu	Thử nghiệm với các mô hình <ul style="list-style-type: none"> - LSTM - LSTM + GRU - CNN + LSTM
Chuyển văn bản thành âm thanh	Thử nghiệm với thư viện (Text-to-Speech - TTS)
Ứng dụng	Xây dựng Website hỗ trợ nhận diện ngôn ngữ ký hiệu Kết nối với Arduino để gọi kết quả cũng như trạng thái xử lý
Server	Viết bằng Flask

Bảng 3. Bảng đề xuất giải pháp

2. Chuẩn bị dữ liệu

2.1. Nguồn dữ liệu

Tự tổng hợp từ video của các thành viên trong nhóm, ghi lại đa dạng cử chỉ tay, góc máy, điều kiện ánh sáng. Mỗi video trong tập dữ liệu đại diện cho các chữ giao tiếp đơn giản: hello, goodbye, happy, love, sad, hungry, beautiful, thankyou.

2.2. Tiền xử lý dữ liệu

Tiền xử lý dữ liệu được thực hiện trong file `data_preprocessing.py` để chuẩn bị dữ liệu video cho việc huấn luyện mô hình. Các bước chính bao gồm:

Trích xuất keypoints: Sử dụng Mediapipe Holistic để trích xuất các keypoints từ video, bao gồm:

- 33 keypoints từ pose (tư thế cơ thể).
- 21 keypoints từ bàn tay trái.
- 21 keypoints từ bàn tay phải.

Tổng cộng, mỗi khung hình tạo ra một mảng 75 keypoints, được lưu dưới dạng mảng NumPy có kích thước (75,).

Xử lý video: Các video được đọc bằng OpenCV, sau đó mỗi khung hình được thay đổi kích thước về (224, 224) và chuyển đổi không gian màu từ BGR sang RGB trước khi đưa vào Mediapipe. Sau khi trích xuất keypoints, dữ liệu được chuẩn hóa và lưu dưới dạng file `.npy`.

Đệm dữ liệu (padding): Hàm `padding_keypoints` đảm bảo rằng mỗi chuỗi video có độ dài cố định (mặc định là 30 khung hình). Nếu chuỗi ngắn hơn, các khung hình trống (zero-padding) được thêm vào đầu và cuối. Nếu dài hơn, các khung hình được lấy mẫu đều theo khoảng cách.

3. Chuẩn hoá ảnh

Quá trình chuẩn hóa ảnh được thực hiện trong hàm `mediapipe_detection` (cả trong `ASL_detection.py` và `data_preprocessing.py`):

- Chuyển đổi không gian màu: Ảnh đầu vào được chuyển đổi từ không gian màu BGR (OpenCV) sang RGB để phù hợp với Mediapipe.
- Thay đổi kích thước: Mỗi khung hình được thay đổi kích thước về (224, 224) để đảm bảo tính đồng nhất và giảm tải tính toán.

PBL5: Dự án kỹ thuật máy tính

- Z-score normalization: Trong ASL_detection.py và train_model.py, dữ liệu keypoints được chuẩn hóa bằng phương pháp Z-score:

$$\text{datanormalized} = \frac{\text{data} - \mu}{\sigma}$$

Trong đó: μ là giá trị trung bình và σ là độ lệch chuẩn, được tính trên tập huấn luyện và lưu vào các file mean.npy và std.npy.

4. Trục quan hoá dữ liệu

Trong file train_model.py, quá trình huấn luyện được trục quan hóa thông qua các biểu đồ loss và accuracy:

- Biểu đồ loss: Hàm plot_training_history vẽ biểu đồ thể hiện giá trị loss trên tập huấn luyện và tập validation qua các epoch. Biểu đồ được lưu tại D:/2025/ASL_model/plots/loss_plot.png.
- Biểu đồ accuracy: Tương tự, biểu đồ accuracy trên tập huấn luyện và validation được vẽ và lưu tại D:/2025/ASL_model/plots/accuracy_plot.png.
- Công cụ sử dụng: Sử dụng thư viện Matplotlib để tạo biểu đồ, với các thông số như kích thước (10, 5), nhãn trục, tiêu đề và lưới.

5. Định hình lại dữ liệu

Dữ liệu được định hình lại để phù hợp với đầu vào của mô hình CNN và LSTM:

- Trong ASL_detection.py, chuỗi keypoints được thu thập thành một danh sách 30 khung hình, sau đó được định hình thành mảng NumPy có kích thước (30, 225), trong đó $225 = 75 \text{ keypoints} \times 3$ (tọa độ x, y, z).
- Trong data_preprocessing.py, các chuỗi keypoints được xử lý để có độ dài cố định (30 hoặc 40 khung hình, tùy thuộc vào file). Hàm padding_keypoints thực hiện đệm hoặc lấy mẫu để đảm bảo kích thước đồng nhất.
- Trong dataset.py, dữ liệu keypoints được tải từ file .npy và chuyển đổi sang kiểu np.float32 để đảm bảo tính tương thích với PyTorch hoặc TensorFlow.

6. Mã hóa nhãn

Mã hóa nhãn được thực hiện trong dataset.py và train_model.py:

PBL5: Dự án kỹ thuật máy tính

- Gán nhãn: Nhãn được lấy từ tên thư mục chứa các file keypoints, tương ứng với các lớp ngôn ngữ ký hiệu (ASL). Danh sách nhãn duy nhất được tạo bằng `sorted(set(self.labels))`.
- Chuyển đổi nhãn thành chỉ số: Mỗi nhãn được ánh xạ thành một chỉ số nguyên (`label_idx`) bằng cách sử dụng danh sách `label_list`. Ví dụ, nhãn "A" có thể được mã hóa thành 0, "B" thành 1, v.v.
- Lưu nhãn: Nhãn được lưu dưới dạng tensor PyTorch (`torch.tensor`) hoặc mảng NumPy trong `train_model.py` để sử dụng trong huấn luyện.
- Xuất nhãn: Trong `ASL_detection.py`, file `labels.csv` chứa danh sách nhãn được đọc bằng Pandas để sử dụng trong dự đoán.

7. Xây dựng mô hình

7.1. Các thành phần chính của mô hình

Mô hình được xây dựng trong file `model.py` là một kiến trúc kết hợp giữa mạng nơ-ron tích chập (CNN) và mạng nơ-ron hồi tiếp (LSTM/GRU), nhằm khai thác đồng thời các đặc trưng không gian và các đặc trưng theo thời gian từ chuỗi keypoints trích xuất từ video ngôn ngữ ký hiệu (ASL). Dữ liệu đầu vào của mô hình có dạng chuỗi 30 khung hình, mỗi khung chứa 225 tọa độ keypoints, tức kích thước đầu vào là (30, 225).

Khối CNN (Convolutional Neural Network)

Mục đích của khối này là trích xuất các đặc trưng không gian cục bộ từ chuỗi dữ liệu keypoints.

- Conv1D - Lớp thứ nhất:
 - Số filters: 64
 - Kernel size: 3
 - Activation: ReLU
 - Đầu vào: (30, 225)
 - Mục đích: Phát hiện các mẫu cục bộ theo chiều thời gian trong chuỗi keypoints.
- BatchNormalization:

PBL5: Dự án kỹ thuật máy tính

- Chuẩn hóa đầu ra của Conv1D để ổn định gradient và tăng tốc quá trình học.
- MaxPooling1D:
 - Kích thước pool: 2
 - Giảm chiều dài chuỗi đặc trưng, giảm số lượng tham số và tăng tính khái quát.
- Dropout (0.3):
 - Loại bỏ ngẫu nhiên 30% đơn vị đầu ra nhằm giảm hiện tượng quá khớp.
- Conv1D - Lớp thứ hai:
 - Filters: 128
 - Kernel: 3
 - Tiếp tục trích xuất đặc trưng sâu hơn.
- BatchNormalization + MaxPooling1D + Dropout (0.3):
 - Lặp lại để duy trì tính ổn định và khả năng tổng quát hóa.

Khối RNN (Recurrent Neural Network)

Khối này đảm nhiệm xử lý đặc trưng theo trình tự thời gian, rất quan trọng trong nhận diện chuỗi ký hiệu.

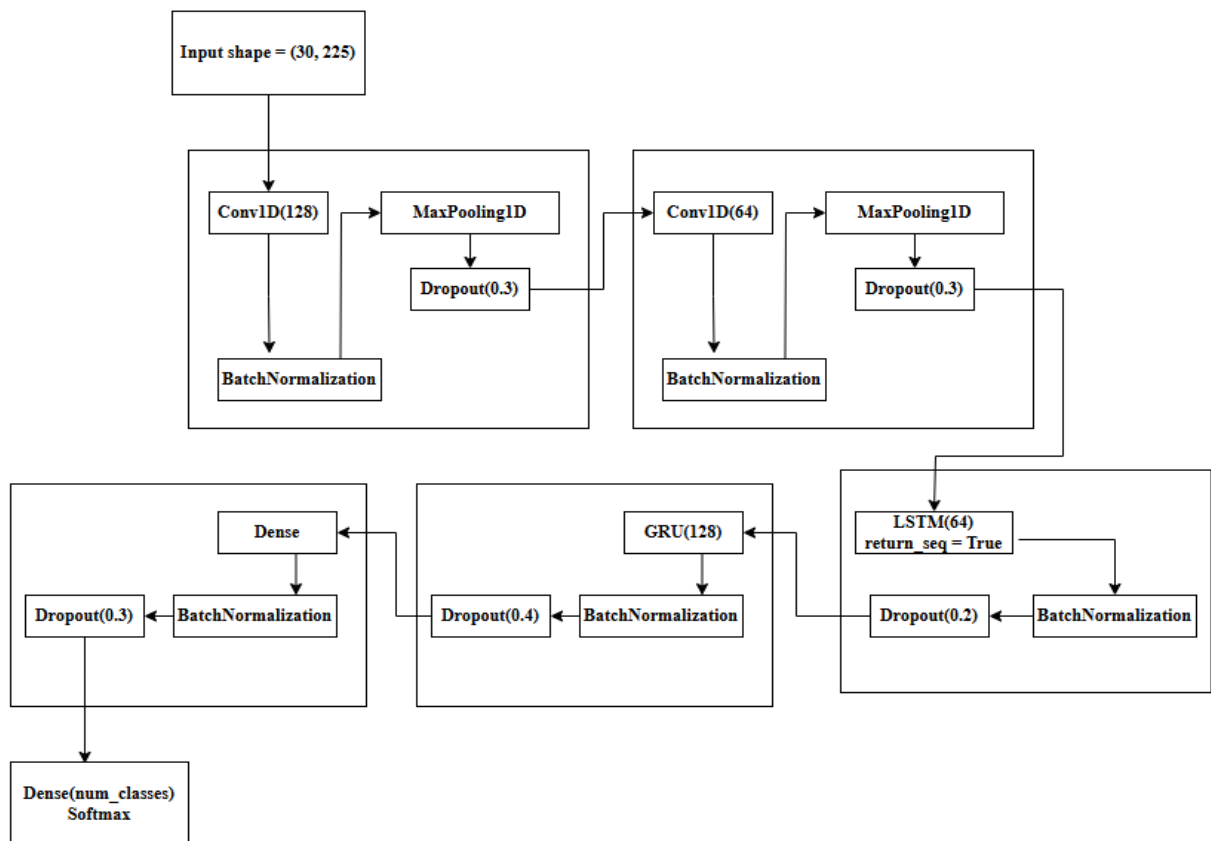
- LSTM Layer:
 - Units: 64
 - return_sequences=True để giữ lại toàn bộ chuỗi đầu ra, phục vụ cho lớp RNN tiếp theo.
 - Giúp ghi nhớ thông tin trong chuỗi dài và học các mối quan hệ theo thời gian.
- BatchNormalization + Dropout (0.2):
 - Duy trì ổn định và chống overfitting sau LSTM.
- GRU Layer:
 - Units: 128
 - Đơn giản và nhanh hơn LSTM, giúp mô hình học được các mẫu dài hạn hiệu quả.
- BatchNormalization + Dropout (0.4):

- Tăng cường khả năng tổng quát hóa và tránh quá khớp.

Khối Dense (Fully Connected)

Khối này thực hiện việc tổng hợp và phân loại đầu ra từ các đặc trưng đã học:

- Dense 1:
 - Units: 64
 - Activation: ReLU
 - Kết hợp các đặc trưng cấp cao từ các khối CNN + RNN.
- BatchNormalization + Dropout (0.3):
 - Chuẩn hóa và chống overfitting.
- Dense cuối (đầu ra):
 - Units: num_classes (số nhãn ASL cần phân loại).
 - Activation: softmax
 - Trả về xác suất phân loại cho từng lớp ký hiệu.



Hình 6. Tổng quan cấu trúc

7.2. Tham số huấn luyện mô hình

Quá trình huấn luyện mô hình được thực hiện trong file `train_model.py`, sử dụng các tham số tối ưu và kỹ thuật callback để đảm bảo mô hình hội tụ ổn định và tránh hiện tượng quá khớp.

Cấu hình huấn luyện:

- Optimizer:
 - Thuật toán: Stochastic Gradient Descent (SGD)
 - Learning rate ban đầu: 0.001
 - Momentum: 0.9
 - Nesterov: True
 - Giúp cập nhật trọng số theo hướng được "dự đoán trước", tăng tốc độ hội tụ.
- Hàm mất mát:
 - `sparse_categorical_crossentropy`
 - Thích hợp cho bài toán phân loại đa lớp khi nhãn ở dạng số nguyên (không one-hot).
- Chỉ số đánh giá (Metric):
 - `accuracy`
 - Theo dõi độ chính xác của mô hình trên tập huấn luyện, validation và kiểm thử.

Callbacks sử dụng:

- EarlyStopping:
 - Dừng huấn luyện nếu `val_loss` không cải thiện sau 8 epochs liên tiếp.
 - `restore_best_weights=True`: Tự động phục hồi trọng số tốt nhất trước khi dừng.
- ReduceLROnPlateau:
 - Giảm learning rate nếu `val_loss` không giảm trong 3 epochs liên tiếp.
 - Hệ số giảm: `factor = 0.5`

PBL5: Dự án kỹ thuật máy tính

- Learning rate tối thiểu: 1e-6

→ Giúp mô hình tiếp tục tối ưu ở tốc độ nhỏ hơn khi gần hội tụ.

Thông số huấn luyện chính:

Tham số	Giá trị
Epochs	120
Batch size	64
Lưu mô hình	D:/2025/ASL_model/Codev2/model.h5

Bảng 4. Bảng chứa thông số huấn luyện chính

7.3. Tăng cường dữ liệu (Data Augmentation)

Padding chuỗi keypoints:

- Được thực hiện trong các file data_preprocessing.py và dataset.py.
- Hàm padding_keypoints đảm bảo mọi chuỗi đều có độ dài cố định (30 hoặc 40 frame).
- Điều này giúp mô hình xử lý video có độ dài khác nhau một cách nhất quán, tương tự một dạng chuẩn hóa đầu vào.

Z-score Normalization:

- Áp dụng trong ASL_detection.py và train_model.py.
- Công thức: $z = \frac{x - \mu}{\sigma}$
- Giúp dữ liệu đầu vào ổn định, loại bỏ ảnh hưởng của các giá trị ngoại lai.

Dropout (Regularization):

- Sử dụng nhiều lớp Dropout (từ 0.2 đến 0.4) trong mô hình (model.py).
- Tăng tính đa dạng trong huấn luyện bằng cách ngẫu nhiên loại bỏ một phần nơ-ron ở mỗi lượt, giảm hiện tượng overfitting.

8. Xây dựng Server xử lý và Web hiển thị

8.1. Mô hình Client-Server

Client: Thiết bị Raspberry Pi gắn camera có nhiệm vụ thu thập liên tục dữ liệu video. Dữ liệu này sau đó được gửi về máy chủ thông qua kết nối mạng LAN để xử lý.

Server: Máy tính xử lý trung tâm đảm nhận việc phân tích hình ảnh, nhận dạng cử chỉ tay, và hiển thị kết quả cho người dùng thông qua trình duyệt web.

8.2. Các thành phần chính của Server

8.2.1. Video Stream Server

Đây là module chịu trách nhiệm chính trong việc tiếp nhận và xử lý luồng dữ liệu video từ Raspberry Pi.

Chức năng:

- Nhận stream video từ Client.
- Tách và xử lý từng khung hình (frame) để phục vụ cho việc nhận dạng.
- Gửi kết quả xử lý cho Web Server hoặc lưu trữ tạm thời.

Công nghệ sử dụng:

- OpenCV: Thư viện xử lý ảnh dùng để trích xuất frame, chuyển đổi không gian màu, lọc nhiễu, phát hiện chuyển động.
- Socket: Dùng để giao tiếp mạng giữa Raspberry Pi và Server.
- Threading: Giúp xử lý đa luồng, cho phép việc thu nhận và xử lý dữ liệu được thực hiện song song, tránh nghẽn luồng.

8.2.2. Web Server

Đây là module đảm nhiệm việc giao tiếp với người dùng thông qua trình duyệt web.

Chức năng:

- Hiện thị giao diện trực quan cho người dùng.
- Hiện thị kết quả nhận dạng cử chỉ theo thời gian thực.

Công nghệ sử dụng:

- HTML/CSS: Thiết kế giao diện người dùng thân thiện.
- JavaScript: Tạo các hiệu ứng động và hỗ trợ tương tác trực tiếp.
- WebSocket: Giao tiếp hai chiều thời gian thực giữa server và trình duyệt, đảm bảo cập nhật liên tục mà không cần refresh trang.

8.2.3. Module Xử lý nhận dạng

Chức năng:

- Nhận dạng hình dáng bàn tay và các cử chỉ.
- Phát hiện chính xác 21 điểm landmark đặc trưng trên bàn tay.
- Theo dõi chuyển động của bàn tay nhằm xác định cử chỉ.

Công nghệ sử dụng:

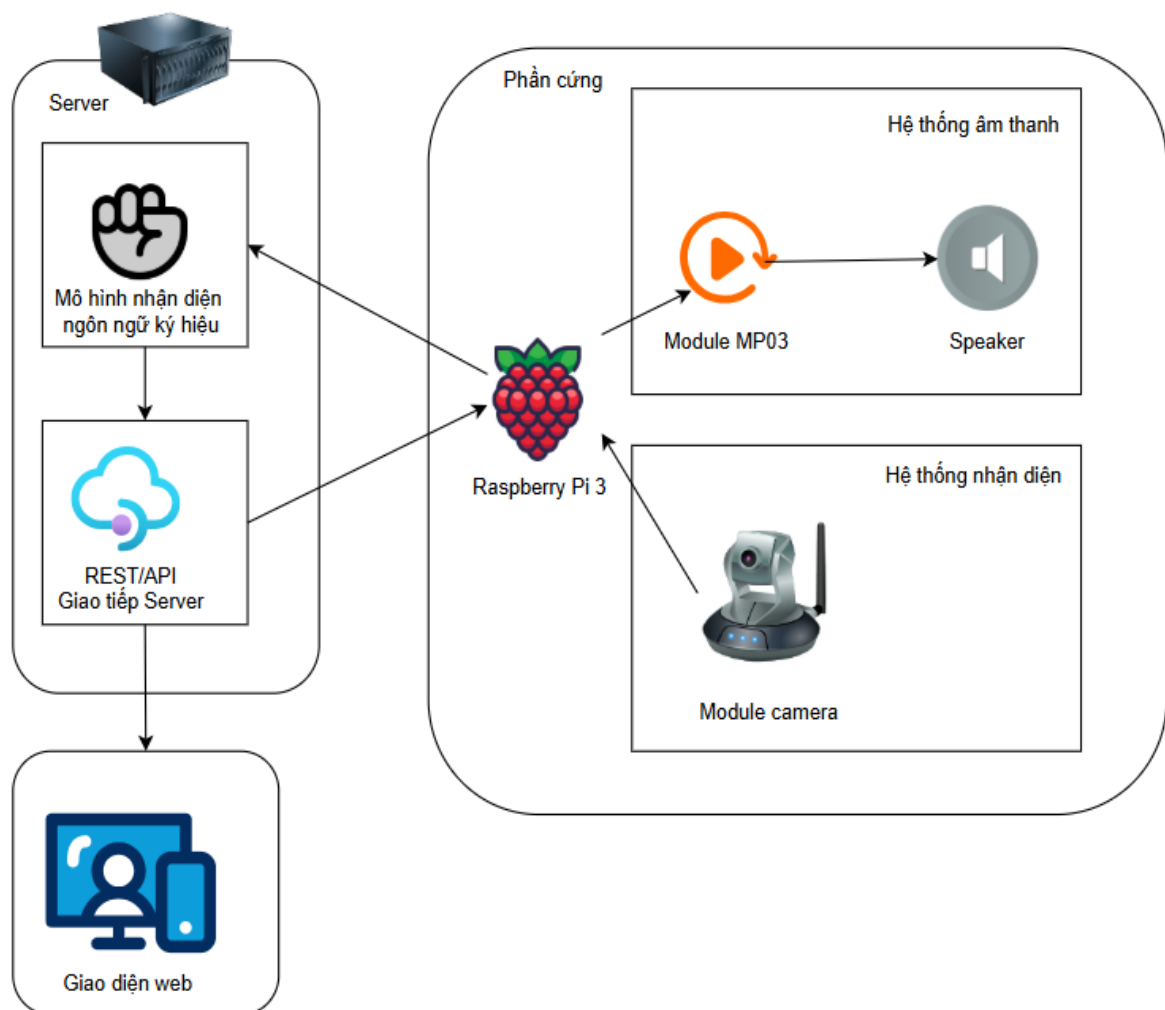
PBL5: Dự án kỹ thuật máy tính

- MediaPipe Hands: Framework mạnh mẽ của Google hỗ trợ phát hiện và theo dõi bàn tay thời gian thực.
- OpenCV: Hỗ trợ tiền xử lý hình ảnh trước khi đưa vào MediaPipe.
- Machine Learning: Có thể tích hợp thêm mô hình học sâu nhằm tăng độ chính xác nhận dạng.

9. Giải pháp

9.1. Giải pháp phần cứng và truyền thông

9.1.1. Sơ đồ tổng quan hệ thống



Hình 7. Sơ đồ tổng quan hệ thống



9.1.2. Luồng dữ liệu điều khiển


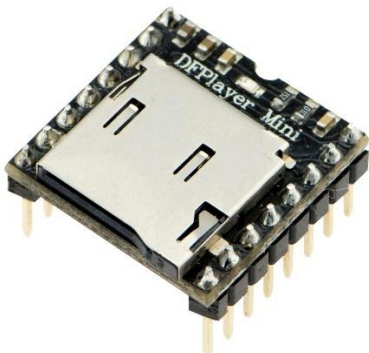
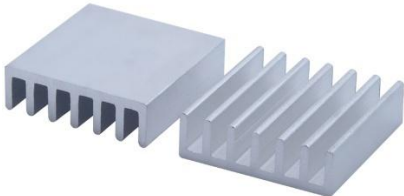
- Hệ thống được xây dựng với trung tâm là Raspberry Pi 3 đảm nhận chức năng điều khiển, tiếp nhận và truyền tải dữ liệu chính:
 - Bước 1: Tiếp nhận hình ảnh được thu thập từ Raspberry Pi Camera Module.

PBL5: Dự án kỹ thuật máy tính

- Bước 2: Truyền hình ảnh cho server để tiến hành xử lý và nhận diện ngôn ngữ ký hiệu.
 - Bước 3: Nhận kết quả từ server và tiến hành truyền kết quả cho Module MP03 Player Mini Arduino DFPLAYER.
 - Bước 4: Tại Module MP03 Player Mini Arduino DFPLAYER kết quả được chuyển đổi sang âm thanh (Text-to-Speech) để tiến hành phát âm thanh ra loa.
- Server đảm nhận vai trò chính trong việc tiếp nhận xử lý và nhận dạng ngôn ngữ ký hiệu.
- Bước 1: Tiếp nhận các frame ảnh được gửi từ Raspberry Pi 3 và tiến hành tiền xử lý, chuẩn hóa.
 - Bước 2: Khi tập hợp đủ 30 frame thì tiến hành nhận diện liên tục đưa ra kết quả ngôn ngữ ký hiệu nhận diện được.
 - Bước 3: Truyền kết quả cho Raspberry Pi 3 và giao diện web kết quả để tiến hành phát âm thanh và xuất kết quả ra giao diện.

9.1.3. Thiết bị phân cứng

STT	Tên thiết bị	Hình ảnh minh họa	Thông tin chi tiết
1	Raspberry Pi 3		<ul style="list-style-type: none">- Vi xử lý: Broadcom BCM2837B0, quad-core A53 (ARMv8) 64-bit SoC @1.4GHz- RAM: 1GB LPDDR2 SDRAM- Kết nối: 2.4GHz and 5GHz.- Nguồn điện sử dụng: 5V/2.5A
2	Raspberry Pi Camera		<ul style="list-style-type: none">- Camera Raspberry Pi V2 8MP dùng cho máy tính nhúng Raspberry Pi.- Cảm biến IMX219.- Số điểm ảnh: 8MP.- Lens: Fixed focus- Camera specifications:<ul style="list-style-type: none">+ CCD size: 1/4inch+ Aperture (F): 2.0+ Focal Length : 3.04mm+ Angle of View (diagonal) : 62.2 degree

			<ul style="list-style-type: none"> - Camera Resolution: 3280 x 2464 pixel stills - Video Resolution: HD 1080p30, 720p60 and 640x480p90 video - Dimensions: 25mm x 23mm x 9mm - Connector: FPC 15 Pin 1.0mm Ribbon connector
3	Loa toàn dải		<ul style="list-style-type: none"> - Kích thước: 2 inch. - Trở kháng: 4Ω. - Công suất: 5W. - Loại: Toàn dải, nghĩa là nó có thể tái tạo âm thanh từ dải thấp đến dải cao.
4	Module MP03 Player Mini Arduino DFPLAYER		<ul style="list-style-type: none"> - Tốc độ lấy mẫu (Khz); 8 / 11.025 / 12 / 16 / 22.05 / 24 / 32 / 44.1 / 48 với ngõ ra 24bit - Hỗ trợ đầy đủ FAT16, FAT32, thẻ TF hỗ trợ tối đa 32Gb. - Có thể điều khiển qua các chân IO hay chuẩn nối tiếp. - Các file âm thanh có thể sắp xếp theo thư mục (tối đa 100 mục), mỗi mục chứa tối đa 255 bài hát. - Âm thanh có thể chỉnh 6 mức
5	Nhôm tản nhiệt		

6	Thẻ nhớ		<ul style="list-style-type: none"> - Dung lượng lưu trữ: 4GB. - Tốc độ đọc dữ liệu: lên đến 100MB/s
---	---------	---	---

Bảng 5: Danh sách các thiết bị phần cứng

9.2. Giải pháp AI/KHDL

9.2.1. Bài toán đặt ra

Với bài toán nhận diện ngôn ngữ ký hiệu, một bài toán đặt ra với hệ thống AI là từ chuỗi các frame ảnh đầu vào làm sao để nhận diện và phân loại ngôn ngữ ký hiệu một cách chính xác và hiệu quả.

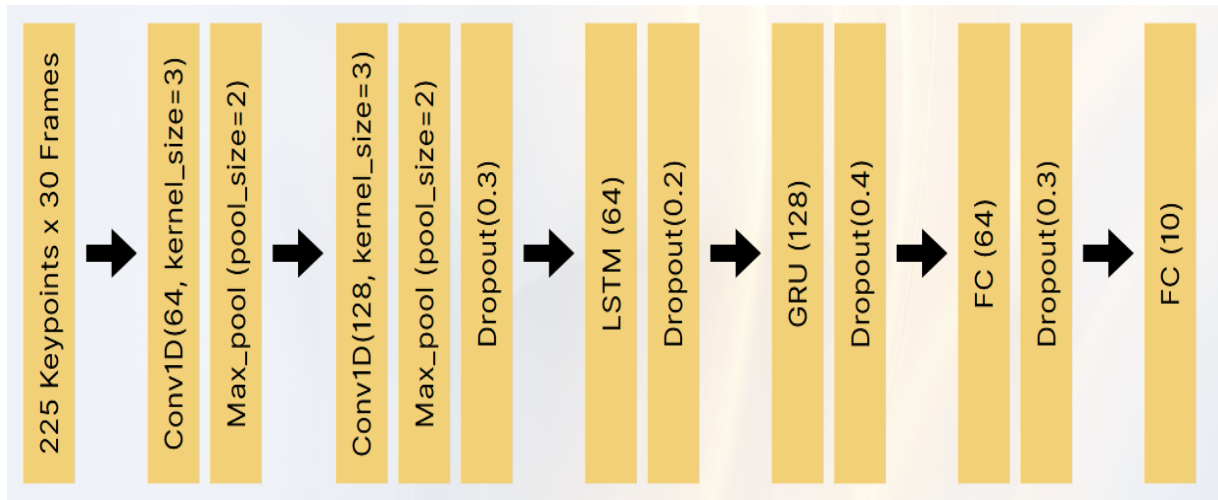
9.2.2. Giải pháp:

Tiền xử lý và trích xuất keypoints:

- Từ các video đầu vào ta trích xuất lấy 30frames/video
- Từ các frame ta sử dụng mediapipe trích xuất lấy 75 keypoints (Pose: 33 keypoints và Hand: 21 keypoints/tay)
- Chuẩn hóa danh sách keypoints theo z-score

Thiết kế mô hình: Mô hình gồm có 3 thành phần chính

- Khối CNN: trích xuất các đặc trưng cục bộ từ dữ liệu.
- Khối RNN: học các phụ thuộc dài hạn của dữ liệu trong chuỗi thời gian.
- Khối Dens (Fully connected): tổng hợp và phân loại đầu ra từ các đặc trưng đã học.



Hình 8. Cấu trúc mô hình

Huấn luyện và đánh giá mô hình: optimizer=SGD; lr = 0.001.

9.3. Giải pháp phần mềm

9.3.1. Phát triển bài toán

1. Bài toán đặt ra

Hệ thống cần nhận diện và chuyển đổi ngôn ngữ ký hiệu của người khiếm thính thành văn bản hoặc âm thanh để hỗ trợ giao tiếp với người nghe. Ba yêu cầu chính được đặt ra như sau:

- Nhận diện ngôn ngữ ký hiệu (tay) thông qua camera để xác định nội dung người khiếm thính muốn truyền đạt.
- Chuyển văn bản sang giọng nói (Text-to-Speech) để tạo đầu ra âm thanh giúp người bình thường hiểu nội dung.
- Xây dựng giao diện người dùng để hiển thị văn bản nhận diện và hỗ trợ tương tác, thiết lập tùy chỉnh hoặc ghi lại lịch sử giao tiếp.

Mỗi bài toán yêu cầu tích hợp mô hình trí tuệ nhân tạo (AI) xử lý hình ảnh và giọng nói với phần mềm backend, frontend, và các thành phần điều khiển thiết bị thông qua Raspberry Pi hoặc phần cứng tương đương.

2. Mục tiêu phát triển

Xây dựng phần mềm với các chức năng chính:

- Giao tiếp với người dùng qua giao diện web hoặc ứng dụng di động (frontend).
- Tiếp nhận tín hiệu từ camera, xử lý hình ảnh và văn bản, điều khiển giọng nói (backend).
- Tích hợp mô hình nhận diện ngôn ngữ ký hiệu (AI) sử dụng học sâu (deep learning).
- Kết nối và điều khiển thiết bị phần cứng như Raspberry Pi, ESP32-CAM, loa, hoặc các cảm biến liên quan.

9.3.2. Công nghệ sử dụng

- **Ngôn ngữ chính:** Python.
- **Framework backend:** Flask.
- **Frontend:** JavaScripts + HTML/CSS.
- **AI Model:** CNN + LSTM

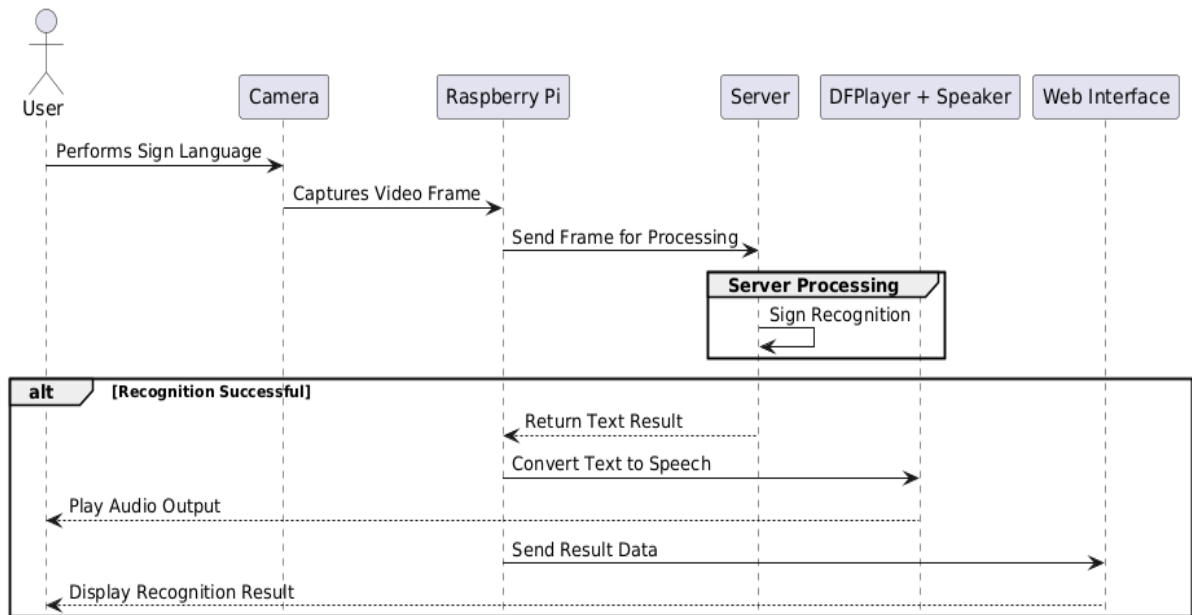
- Giao tiếp phần cứng:

+ Giao tiếp với Camera

+ Kết nối loa hoặc module âm thanh qua USB

+ HTTP request và serial với Arduino.

9.3.3. Biểu đồ tuần tự



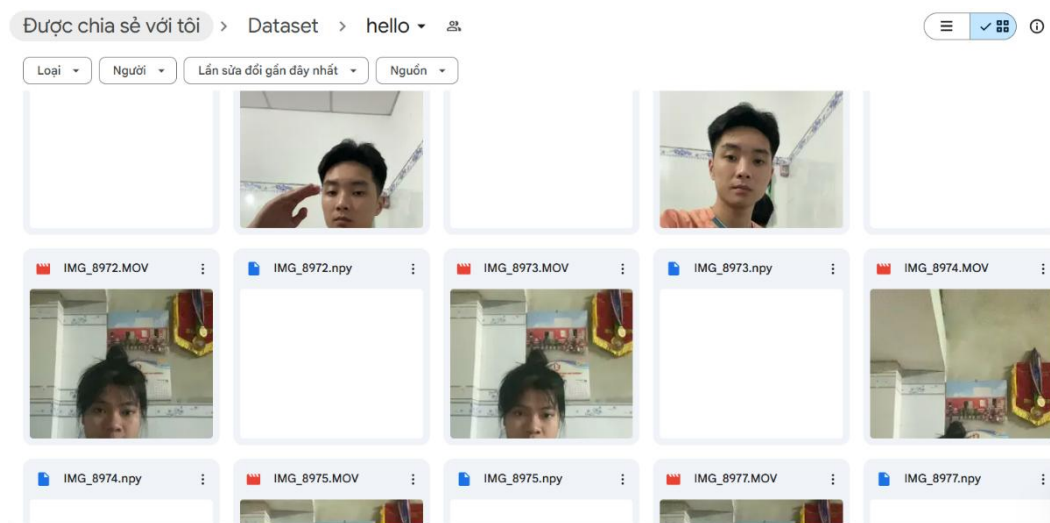
Hình 9. Sơ đồ tuần tự của hệ thống nhận diện ngôn ngữ ký hiệu

10. Kết quả

10.1. Tập dữ liệu

Dữ liệu gốc:

- Tự tổng hợp từ video của các thành viên trong nhóm, ghi lại đa dạng cử chỉ tay, góc máy, điều kiện ánh sáng.
- Bổ sung từ các bộ dữ liệu mở (nếu cần mở rộng).
- ASL (American Sign Language) datasets (ví dụ: ASL Alphabet, MS-ASL).
- Vietnamese Sign Language datasets (nếu có sẵn).



Hình 10. Ảnh minh chứng dữ liệu tự thu thập

Quy trình tiền xử lý:

- Cắt khung hình (frame extraction) từ video.
- Gán nhãn thủ công (labeling) cho từng cử chỉ tương ứng với ký hiệu.
- Cân bằng dữ liệu (data balancing) để tránh overfitting.
- Tăng cường dữ liệu (data augmentation): Xoay, thay đổi độ sáng, thêm nhiễu để mô hình học tốt hơn.

Cấu trúc thư mục dữ liệu người dùng, ví dụ:

```
dataset/  
├── hello/  
│   ├── hello1.MOV  
│   ├── hello2.MOV  
│   └── ...  
├── goodbye/  
│   ├── goodbye1.MOV  
│   ├── goodbye2.MOV  
│   └── ...
```

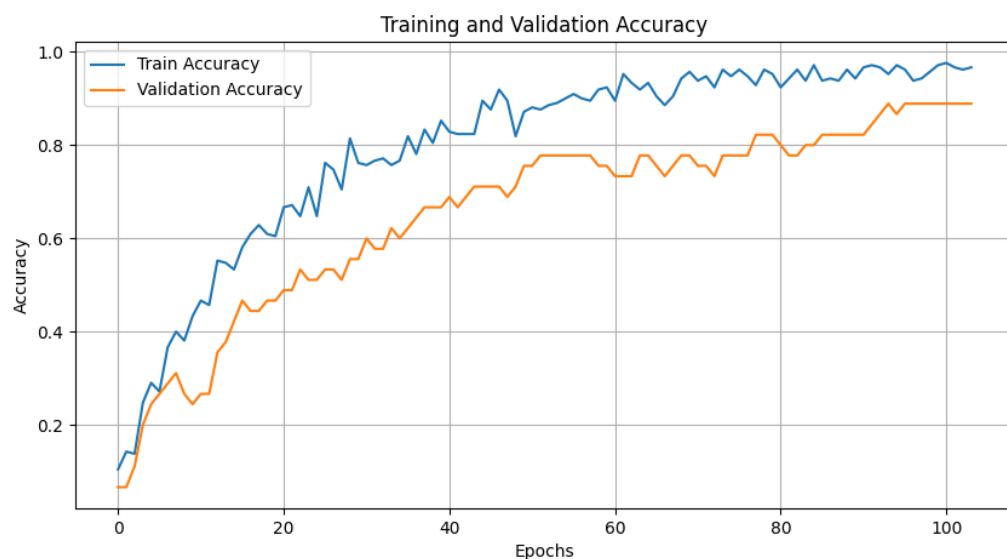
Trong đó:

- Mỗi thư mục con như hello, goodbye, ... tương ứng với từ cụ thể.
- Mỗi thư mục chứa khoảng 30 video của thành viên biểu thị ký hiệu đó, được quay trong điều kiện ánh sáng hợp lý, gương mặt rõ nét và nhìn chính diện.

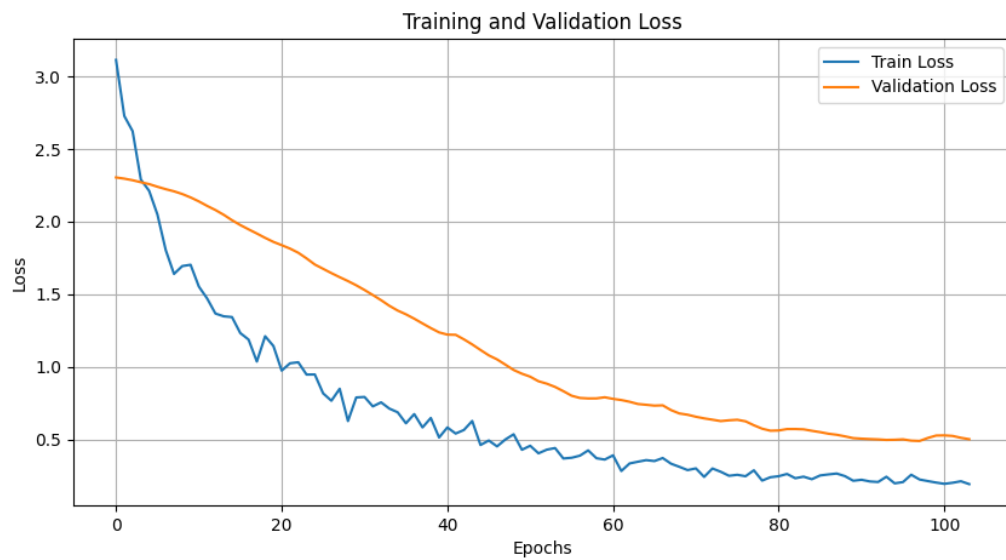
10.2. Huấn luyện mô hình

- Không sử dụng mô hình pre-trained cho nhận diện ký hiệu tùy chỉnh (vì mỗi người có cách thể hiện khác nhau).
- Huấn luyện từ đầu trên tập dữ liệu tự thu thập.
- Đánh giá bằng cross-validation để đảm bảo độ chính xác.

10.3. Kết quả huấn luyện mô hình



Hình 11. Biểu đồ độ chính xác huấn luyện



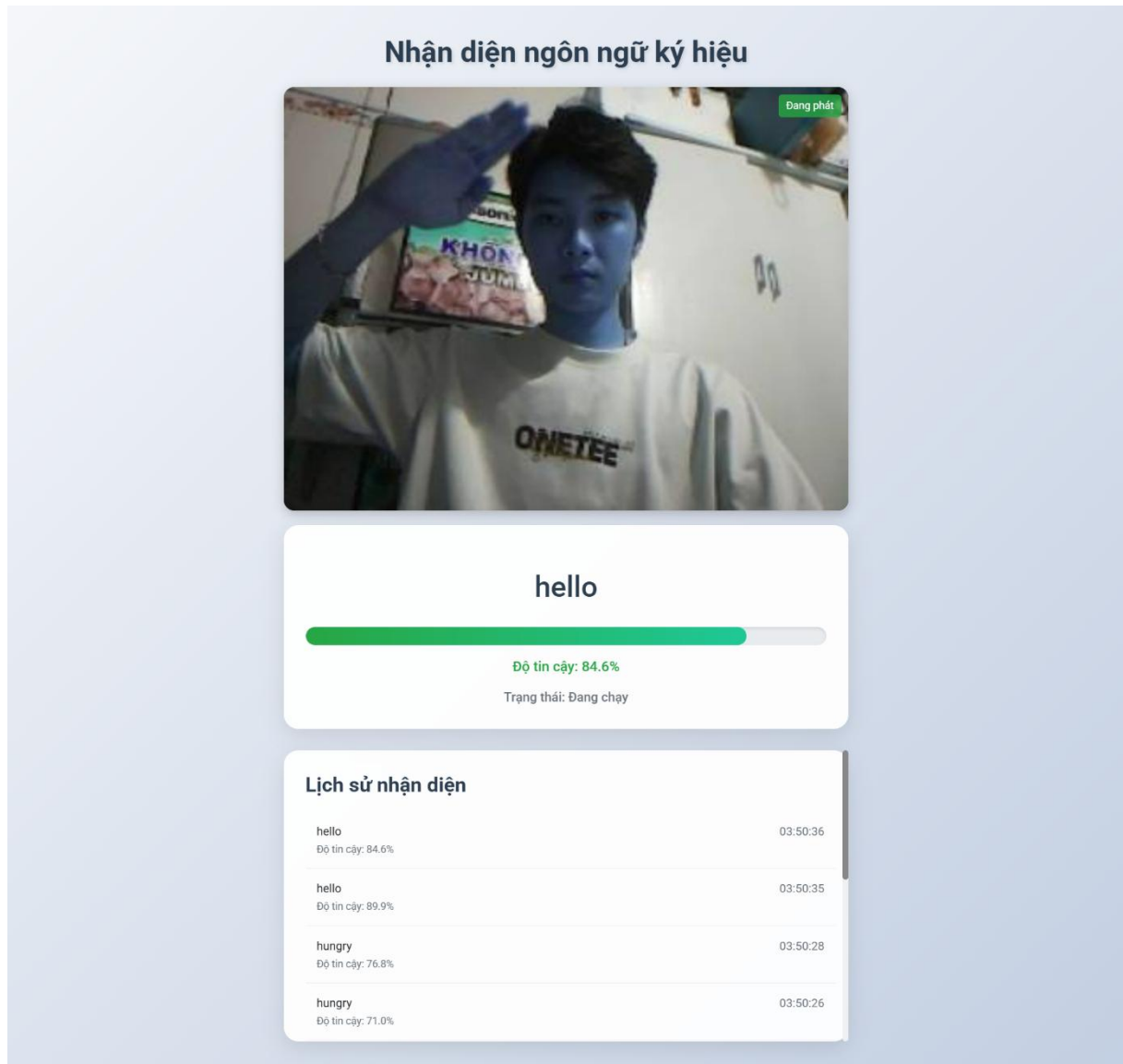
Hình 12. Biểu đồ hàm mất mát huấn luyện

10.4. Sản phẩm phần cứng



Hình 13 Phần cứng của hệ thống nhận diện

10.5. Kết quả nhận diện



Hình 14 Kết quả nhận diện được hiển thị lên web

IV. KẾT LUẬN

1. Đánh giá

Đồ án xây dựng hệ thống nhận diện ngôn ngữ ký hiệu hỗ trợ giao tiếp với Raspberry Pi làm trung tâm điều khiển, kết hợp các mô hình AI nhận diện cử chỉ tay và biểu cảm khuôn mặt đã đạt được các mục tiêu đề ra.

- Hệ thống nhận diện cử chỉ tay hoạt động hiệu quả với độ chính xác cao nhờ sử dụng mô hình MediaPipe Hands kết hợp với CNN được fine-tune để phát hiện chuyển động tay trong nhiều điều kiện ánh sáng khác nhau.
- Mô hình nhận diện biểu cảm khuôn mặt (Facial Expression Recognition) dựa trên CNN hoặc ResNet giúp bổ sung thông tin cảm xúc trong quá trình giao tiếp.
- Hệ thống hỗ trợ chuyển đổi ngôn ngữ ký hiệu thành văn bản hoặc giọng nói (Text-to-Speech) để người dùng có thể tương tác dễ dàng hơn.
- Giao diện web/app thân thiện, trực quan giúp người dùng thuận tiện trong việc sử dụng và tùy chỉnh hệ thống.

Tổng thể, sản phẩm vận hành ổn định, đáp ứng tốt các yêu cầu về tính năng và hiệu năng trên phần cứng Raspberry Pi, đồng thời mở ra hướng phát triển ứng dụng hỗ trợ cộng đồng người khiếm thính trong tương lai.

2. Hướng phát triển

2.1. Nâng cao độ chính xác và khả năng nhận diện trong điều kiện khó khăn

Hiện tại, mô hình MediaPipe Hands và YOLOv8n hoạt động tốt trong điều kiện ánh sáng ổn định và góc nhìn rõ ràng. Tuy nhiên, để áp dụng trong môi trường thực tế với ánh sáng thay đổi, nhiễu nền phức tạp hoặc cử động nhanh, cần cải thiện bằng cách:

- Tăng cường dữ liệu (data augmentation) như thay đổi độ sáng, thêm nhiễu, xoay góc để mô hình học được đa dạng trường hợp.
- Thử nghiệm các mô hình hiện đại hơn như 3D CNN, Transformer-based (ViT), hoặc kết hợp với OpenPose để tăng độ chính xác khi nhận diện cử chỉ động (dynamic gestures).
- Áp dụng tiền xử lý ảnh (histogram cân bằng, background subtraction) để cải thiện chất lượng đầu vào.

2.2. Mở rộng tính năng nhận diện

Bên cạnh nhận diện ký hiệu tay cơ bản, có thể mở rộng để:

- Nhận diện biểu cảm khuôn mặt (sử dụng Facial Expression Recognition) để bổ sung ngữ cảnh cảm xúc trong giao tiếp.
- Nhận diện cử chỉ cơ thể (body pose estimation) giúp hiểu đầy đủ hơn ngôn ngữ ký hiệu.
- Hỗ trợ đa ngôn ngữ ký hiệu (ASL, Vietnamese Sign Language...) bằng cách huấn luyện trên nhiều bộ dữ liệu đa dạng.

2.3. Tích hợp và tối ưu trên các thiết bị

- Nghiên cứu tối ưu mô hình (quantization, pruning, TensorRT) để giảm độ trễ và tăng tốc độ xử lý.
- Tích hợp Text-to-Speech (TTS) để chuyển ngữ ký hiệu thành giọng nói thời gian thực.
- Xây dựng giao diện tương tác đơn giản, hỗ trợ phản hồi nhanh cho người dùng khiếm thính.

2.4. Ứng dụng và phát triển

- Ứng dụng trong giáo dục: Hỗ trợ trẻ em khiếm thính học ngôn ngữ ký hiệu.
- Hệ thống dịch thuật thời gian thực: Kết hợp AI để phiên dịch giữa ngôn ngữ ký hiệu và tiếng nói/text.
- Mở rộng sang các thiết bị di động (Android/iOS) để tiện dụng hơn.

V. TÀI LIỆU THAM KHẢO

- [1] L. A. T. V. Eli Stevens, Deep Learning with PyTorch, 2020.
- [2] “VINBIGDATA,” [Trực tuyến]. Available: <https://vinbigdata.com/kham-pha/xuly-du-lieu-anh-mot-so-kien-thuc-can-ban.html>.
- [3] “ScyllaDB,” [Trực tuyến]. Available: <https://www.scylladb.com/glossary/real-time-data/>.
- [4] W. Stallings, Data and Computer Communications (10th Edition), 2017.
- [5] S. V. Navendu K., “Word Level Sign Language Recognition using MediaPipe and LSTM-GRU Network,” International Symposium on Embedded Systems, 2024.
- [6] R. S. A. Diksha Kumari, “Isolated Video-Based Sign Language Recognition Using a Hybrid CNN-LSTM Framework Based on Attention Mechanism,” Electronics 2024, 13, 1229, 2024.
- [7] C. V. Huang J, “Video-based sign language recognition via ResNet and LSTM network,” Journal of Imaging 2024, 10, 149, 2024.