**BÁO CÁO**

**PHÁT TRIỂN HỆ THỐNG THÔNG MINH**

NHÓM: 05

**Đề tài 16: “Hệ thống hỗ trợ giao tiếp với người câm điếc”**

**Nhóm BTL: 11**

Thành viên :

* **Nguyễn Khánh Nam – B20DCCN454**
* Trần Thanh Tuấn – B20DCCN620
* Nguyễn Hoàng Việt – B20DCCN728
* Đàm Trọng Ngọc Hà – B20DCCN211

**Mục lục**

Chương I : Đánh giá lựa chọn thuật toán 2

Chương II : Thiết kế hệ thống 20

**Chương I : Đánh giá lựa chọn thuật toán**

# Phần A:Tổng quan và giới thiệu bài toán

**I. Giới thiệu bài toán**

Trong thế giới hiện đại, công nghệ đã tạo ra những cơ hội tuyệt vời để giúp đỡ và tạo điều kiện thuận lợi hơn cho những người gặp khó khăn trong việc giao tiếp do mất năng lực thính giác hoặc nói. Một trong những bài toán quan trọng trong lĩnh vực này là hệ thống hỗ trợ người câm điếc. Người câm điếc là những cá nhân với khả năng thính giác bị mất hoặc giới hạn, và vì vậy, họ thường gặp khó khăn trong việc sử dụng ngôn ngữ giọng nói để giao tiếp. Hệ thống hỗ trợ người câm điếc là một lĩnh vực đầy thách thức và đa dạng, nỗ lực để cung cấp cho họ các công cụ và phương pháp giao tiếp hiệu quả, giúp họ tham gia vào xã hội, giáo dục và nghề nghiệp một cách bình đẳng với mọi người khác. Do đó, việc phát triển các hệ thống và công nghệ để hỗ trợ giao tiếp cho họ là cực kỳ quan trọng.

Hệ thống hỗ trợ người câm điếc nhằm giúp họ có khả năng tham gia vào xã hội, truyền đạt thông điệp, tham gia vào giáo dục và nghề nghiệp, và tạo điều kiện thuận lợi cho cuộc sống hàng ngày. Điều này có thể được thực hiện thông qua việc phát triển các công cụ và ứng dụng điện tử cho việc chuyển đổi giọng nói thành văn bản hoặc ngược lại, sử dụng ngôn ngữ ký hiệu và cử chỉ, hay sử dụng công nghệ video gọi để giao tiếp qua hình ảnh.

Bài toán này không chỉ đòi hỏi sự phát triển công nghệ tiên tiến mà còn đặt ra các thách thức về việc hiểu và hỗ trợ nhu cầu cá nhân của từng người câm điếc. Từ việc xử lý giọng nói và ngôn ngữ ký hiệu đến việc phát triển ứng dụng di động và giao tiếp xã hội, hệ thống hỗ trợ người câm điếc đang là một lĩnh vực nghiên cứu và phát triển không ngừng, nhằm tạo ra một môi trường giao tiếp bình đẳng và hiệu quả cho tất cả mọi người.

**II.Các bài toán con liên quan.**

**1. Bài toán nhận dạng vị trí bàn tay.**

Bài toán nhận dạng vị trí bàn tay là một trong những nhiệm vụ quan trọng trong lĩnh vực thị giác máy tính. Mục tiêu của bài toán là xác định vị trí và đặc điểm của bàn tay trong các hình ảnh hoặc video. Điều này có thể có nhiều ứng dụng thú vị, từ điều khiển điện tử bằng cử chỉ đến nhận dạng ngôn ngữ ký hiệu trong việc trợ giúp người điếc.

Để giải quyết bài toán này, trước tiên chúng ta phải xử lý dữ liệu đầu vào. Dữ liệu này có thể là hình ảnh hoặc video chứa một hoặc nhiều bàn tay, và chúng ta cần làm sạch dữ liệu và loại bỏ nhiễu nếu có. Sau đó, chúng ta có thể sử dụng các phương pháp thị giác máy tính và học máy để phân tích hình ảnh và xác định vị trí của bàn tay.

Các bước cụ thể trong quá trình giải quyết bài toán này có thể bao gồm trích xuất đặc trưng từ hình ảnh, phát hiện đối tượng, theo dõi bàn tay qua các khung hình liên tiếp trong video, và sau đó dự đoán vị trí và cử chỉ của bàn tay.

**2. Bài toán chuyển đổi ngôn ngữ ký hiệu thành văn bản.**

Bài toán chuyển đổi ngôn ngữ ký hiệu thành văn bản tập trung vào việc chuyển đổi các cử chỉ và biểu đồ của ngôn ngữ ký hiệu, được thực hiện bởi người câm điếc trong quá trình giao tiếp, thành văn bản viết. Mục tiêu của bài toán này là giúp người câm điếc có khả năng truyền đạt ý kiến và thông điệp của họ bằng văn bản để giao tiếp với những người không hiểu ngôn ngữ ký hiệu.

Để một hệ thống có thể nhận dạng ngôn ngữ ký hiệu thì cần phải có một tập dữ liệu lớn chứa các ví dụ của ngôn ngữ ký hiệu được thực hiện bởi người sử dụng. Dữ liệu này thường chứa các video hoặc hình ảnh của người biểu đạt ngôn ngữ ký hiệu trong quá trình giao tiếp. Dữ liệu hình ảnh hoặc video từ bước trên cần được xử lý để trích xuất các đặc trưng quan trọng, như hình dạng và động tác của các cử chỉ và được gán nhãn tương ứng. Công nghệ xử lý hình ảnh và video được áp dụng để phân tích và nhận dạng các đặc điểm này. Sau khi dữ liệu đã được xử lý, bài toán nhận dạng tập trung vào việc nhận biết các cử chỉ và biểu đồ ngôn ngữ ký hiệu. Các mô hình máy học, deep learning, hoặc các phương pháp nhận dạng khác có thể được áp dụng để phân loại các cử chỉ này thành các loại ngôn ngữ ký hiệu cụ thể.

Sau khi các cử chỉ ngôn ngữ ký hiệu đã được nhận dạng và phân loại thì chúng sẽ được chuyển đổi thành văn bản tương ứng. Quá trình này liên quan đến việc dịch các cử chỉ và biểu đồ thành từ ngữ và câu trực tiếp. Kết quả quá trình chuyển đổi sẽ là một đoạn văn bản tương ứng với ngôn ngữ ký hiệu được mô tả. Qua đó có thể hỗ trợ người câm điếc giao tiếp được với mọi người xung quanh một cách dễ dàng hơn.

**3. Bài toán chuyển văn bản thành ảnh, video ngôn ngữ ký hiệu để hỗ trợ người bình thường giao tiếp với người câm điếc.(Không sử dụng thuật toán)**

Bài toán chuyển văn bản thành ngôn ngữ ký hiệu là một ứng dụng quan trọng trong lĩnh vực hỗ trợ giao tiếp cho người câm điếc. Nó tập trung vào việc chuyển đổi văn bản viết thành các cử chỉ và biểu đồ của ngôn ngữ ký hiệu, giúp người bình thường có khả năng giao tiếp với người câm điếc thông qua ngôn ngữ ký hiệu.

Quá trình bắt đầu bằng việc nhận dạng văn bản viết. Sau đó văn bản viết cần được xử lý để phân tích cú pháp và hiểu nghĩa của các từ, cụm từ và câu. Bao gồm việc phân loại từ loại, xác định mối quan hệ cú pháp và xây dựng một biểu đồ ngữ nghĩa. Sau khi văn bản đã được xử lý, bài toán chuyển đổi tập trung vào việc dự đoán các cử chỉ của ngôn ngữ ký hiệu phù hợp với nội dung văn bản. Các cử chỉ này phải tuân theo ngôn ngữ ký hiệu chuẩn và có ý nghĩa tương đương với văn bản. Kết quả của quá trình chuyển đổi cần được hiển thị trên màn hình hoặc thông qua một thiết bị để người câm điếc có thể xem và hiểu. Có thể bao gồm việc sử dụng mô hình 3D hoặc avatar để biểu diễn ngôn ngữ ký hiệu hoặc hiển thị video ngôn ngữ ký hiệu.

**III.Xây dựng dữ liệu**

**1.Quá trình xây dựng dữ liệu**

* Các thành viên trong nhóm thể hệ các ngôn ngữ ký hiệu bằng tay rồi chụp lại ảnh với mỗi ký hiệu tầm 30 ảnh với các góc máy khác nhau
* Các thành viên gán nhãn phần tay đang thực hiện ngôn ngữ ký hiệu trong ảnh sao cho phù hợp với từng ảnh sau khi chụp.
* Sau khi gán nhãn xong các mẫu sẽ có 1 file .xml được tạo ra là 1 đoạn mã chứa:
* Các thông số của bức ảnh(chiều rộng, chiều cao, độ sâu)
* Mô tả thông tin của đối tượng trong ảnh(tên,...)
* Thông tin về hình chữ nhật giới hạn (bounding box) của đối tượng, bao gồm tọa độ xmin, ymin, xmax và ymax của bounding box.

=> Kết quả thu được: mỗi hành động là 1 file ảnh và 1 file .xml cùng tên nằm cạnh nhau.

**2.Thống kê dữ liệu**

* Tổng số mẫu: 954
* Bộ dữ liệu gồm 32 nhãn tương ứng:
* Các ký tự từ a - y trong bảng chữ cái tiếng anh (trừ các ký tự f, j, z, w)
* Các số từ 0 - 9.

* Thống kê số lượng từng nhãn:

|  |  |  |  |
| --- | --- | --- | --- |
| Tên nhãn | Số lượng | Tên nhãn | Số lượng |
| 0 | 30 | H | 30 |
| 1 | 30 | I | 30 |
| 2 | 30 | K | 30 |
| 3 | 30 | L | 30 |
| 4 | 30 | M | 30 |
| 5 | 30 | N | 30 |
| 6 | 30 | O | 30 |
| 7 | 30 | P | 30 |
| 8 | 30 | Q | 30 |
| 9 | 30 | R | 30 |
| A | 28 | S | 30 |
| B | 30 | T | 30 |
| C | 28 | U | 30 |
| D | 30 | V | 30 |
| E | 30 | X | 30 |
| G | 28 | Y | 30 |

**Phần B:  Lựa chọn và đánh giá thuật toán**

**I.Các thuật toán**

Mỗi bài toán con nhóm sẽ chọn ra 4 thuật toán:

* Bài toán nhận dạng vị trí bàn tay:
* TT1: YOLO
* TT2: Faster R-CNN
* TT3: SSD
* TT4: CNNs
* Bài toán chuyển đổi ngôn ngữ ký hiệu thành văn bản:
* TT1: YOLO
* TT2: SSD
* TT3: CNNs
* TT4: Faster R-CNN

**1.Thuật toán YOLO**

**1.1 Giới thiệu thuật toán**

Thuật toán YOLO (You Only Look Once) là một trong những thuật toán nổi tiếng và mạnh mẽ trong lĩnh vực nhận diện đối tượng và phát hiện vật thể trong hình ảnh và video. YOLO đã được phát triển bởi Joseph Redmon và được công bố lần đầu vào năm 2016. Thuật toán YOLO giải quyết vấn đề phát hiện đối tượng một cách nhanh chóng và hiệu quả.

YOLO đề xuất sử dụng mạng thần kinh đầu cuối để đưa ra dự đoán về các hộp giới hạn (bounding box) và xác suất của đối tượng cùng một lúc. Nó khác với cách tiếp cận của các thuật toán phát hiện đối tượng trước đó, vốn sử dụng lại các trình phân loại để thực hiện phát hiện.

Theo một cách tiếp cận cơ bản khác để phát hiện đối tượng, YOLO đã đạt được kết quả tiên tiến, đánh bại các thuật toán phát hiện đối tượng thời gian thực khác với khoảng cách lớn.

**1.2 Nguyên lý hoạt động**

Thuật toán YOLO lấy hình ảnh làm đầu vào, sau đó sử dụng mạng nơ-ron tích chập sâu đơn giản để phát hiện các đối tượng trong ảnh. Kiến trúc của mô hình CNN tạo thành xương sống của YOLO được hiển thị bên dưới.

A diagram of a computer network

Description automatically generated

20 lớp tích chập đầu tiên của mô hình được đào tạo trước với ImageNet bằng cách cắm vào một lớp tổng hợp trung bình tạm thời (temporary average pooling) và lớp được kết nối đầy đủ (fully connected layer). Sau đó, mô hình đào tạo trước này được chuyển đổi để thực hiện phát hiện. Lớp được kết nối đầy đủ cuối cùng của YOLO dự đoán cả xác suất của lớp và tọa độ hộp giới hạn.

YOLO chia hình ảnh đầu vào thành lưới S × S. Nếu tâm của một đối tượng rơi vào một ô lưới thì ô lưới đó có nhiệm vụ phát hiện đối tượng đó. Mỗi ô lưới dự đoán các hộp giới hạn B và điểm tin cậy cho các hộp đó. Các điểm tin cậy này phản ánh mức độ tin cậy của mô hình rằng hộp chứa một đối tượng và mức độ chính xác mà mô hình cho rằng hộp được dự đoán.

YOLO dự đoán nhiều hộp giới hạn trên mỗi ô lưới. Tại thời điểm đào tạo, ta chỉ muốn một bộ dự đoán hộp giới hạn thể hiện cho từng đối tượng. YOLO chỉ định bộ dự đoán dựa trên chỉ số IOU hiện tại cao nhất với thực tế. Điều này dẫn đến sự chuyên môn hóa giữa các bộ dự đoán hộp giới hạn. Mỗi công cụ dự đoán trở nên tốt hơn trong việc dự báo các kích thước, tỷ lệ khung hình hoặc loại đối tượng nhất định, cải thiện tổng thể recall score.

Một kỹ thuật quan trọng được sử dụng trong các mô hình YOLO là NMS (non-maximum suppression). NMS là một bước hậu xử lý được sử dụng để cải thiện độ chính xác và hiệu quả của việc phát hiện đối tượng. Trong phát hiện đối tượng, thông thường có nhiều hộp giới hạn được tạo cho một đối tượng trong một hình ảnh. Các hộp giới hạn này có thể chồng lên nhau hoặc nằm ở các vị trí khác nhau, nhưng tất cả chúng đều đại diện cho cùng một đối tượng. NMS được sử dụng để xác định và loại bỏ các hộp giới hạn dư thừa hoặc không chính xác và đề xuất một hộp giới hạn duy nhất cho từng đối tượng trong ảnh.

**1.3 Ưu điểm và nhược điểm**

Ưu điểm:

* Thực hiện nhận diện đối tượng trong một lần truyền qua mạng neural, nên nó nhanh hơn so với một số phương pháp truyền thống chia thành nhiều giai đoạn.
* Khả năng tích hợp với các công nghệ trí tuệ nhân tạo như học sâu, giúp nâng cao khả năng nhận diện và phân loại đối tượng.
* Được triển khai trong các ứng dụng thời gian thực như giám sát an ninh và xe tự lái.
* Khả năng nhận diện đa dạng các loại đối tượng trong hình ảnh và video.
* Sử dụng một lượng lớn dữ liệu để đào tạo, nhưng quá trình này có thể thực hiện trên nhiều tài liệu học trực tuyến và nền tảng mã nguồn mở, giúp đơn giản hóa việc triển khai và tùy chỉnh.

Nhược điểm:

* Độ chính xác thấp ở các đối tượng nhỏ hoặc có độ phức tạp cao.
* Có thể tạo ra nhiều sai sót trong việc nhận diện và phân loại đối tượng, đặc biệt là khi có sự che khuất, ánh sáng yếu hoặc nhiễu trong hình ảnh.
* Cần có phần cứng mạnh, đặc biệt là trong các ứng dụng thời gian thực.
* Khó khăn khi xử lý các đối tượng chồng chéo hoặc gần nhau.

**2. Thuật toán SSD (Single Shot MultiBox Detector).**

**2.1 Giới thiệu thuật toán.**

SSD, hay Single Shot MultiBox Detector, là một thuật toán tiên tiến trong lĩnh vực thị giác máy tính, ra đời vào năm 2016 tại Google Research. Được phát triển bởi các nhà nghiên cứu hàng đầu như Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy và Andrew Rabinovich, SSD đã đem lại sự cách mạng trong việc phát hiện đối tượng trong hình ảnh.

Thuật toán này chứa trong mình những đặc điểm ấn tượng như khả năng phát hiện đa độ phân giải, đồng thời xử lý nhanh chóng và xác định vị trí chính xác của các đối tượng. SSD sử dụng một mạng neural cơ bản để trích xuất đặc trưng từ hình ảnh, sau đó dự đoán vị trí và điểm ghi nhãn của đối tượng thông qua các hộp giới hạn đa tỷ lệ.

Với khả năng áp dụng rộng rãi, SSD đã thành công trong nhiều ứng dụng thị giác máy tính quan trọng như phát hiện biển báo giao thông, giám sát video, và nhiều lĩnh vực khác. Tính linh hoạt và hiệu suất ấn tượng đã biến SSD trở thành một trong những công cụ quan trọng trong thế giới ngày nay, giúp nâng cao khả năng xử lý hình ảnh và video tự động.

**2.2 Kiến trúc mạng.**

**A screenshot of a computer

Description automatically generated**

Kiến trúc của SSD được xây dựng trên VGG-16 được loại bỏ tầng fully-connected. Lý do mà VGG-16 được sử dụng như tầng cơ sở là vì sự hiệu quả của nó trong bài toán phân loại ảnh với các ảnh có độ phân giải cao. Thay vì sử dụng tầng fully-connected của VGG, một tập các tầng convolution phụ trợ (cụ thể là 6 trong lưu đồ) được thêm vào, vì vậy ta có thể trích xuất được các features với nhiều tỉ lệ khác nhau, và giảm gần kích thước của đầu vào trong từng tầng mạng.

**2.3 Nguyên lý hoạt động**

Nguyên lý hoạt động của thuật toán SSD (Single Shot MultiBox Detector) là kết hợp giữa việc phát hiện đối tượng và xác định vị trí của chúng trong hình ảnh vào một lần dự đoán (single shot). Thuật toán này được thiết kế để giải quyết các vấn đề cơ bản trong lĩnh vực nhận diện đối tượng, và nó hoàn toàn khác biệt so với các phương pháp truyền thống dựa trên nhiều giai đoạn.

SSD sử dụng một mạng neural cơ bản (thường là một biến thể của CNN) để trích xuất đặc trưng từ hình ảnh đầu vào. Điều đặc biệt là SSD sử dụng nhiều tầng đặc trưng với độ phân giải khác nhau. Mỗi tầng đặc trưng tương ứng với một tỷ lệ của các hộp giới hạn (bounding boxes), cho phép phát hiện các đối tượng ở các kích thước khác nhau, dự đoán vị trí và xác định vị trí của các đối tượng trong hình ảnh. Thay vì sử dụng các giai đoạn riêng biệt cho việc phát hiện và xác định vị trí, SSD thực hiện cả hai công việc này trong một lần chạy qua mạng neural. Điều này giúp tăng tốc quá trình và giảm độ phức tạp của mô hình.

**2.4 Ưu và nhược điểm của thuật toán.**

Ưu điểm:

* Thực hiện phát hiện đối tượng trong một lần chạy qua mạng neural, giúp tăng tốc quá trình nhận diện.
* Sử dụng nhiều tầng đặc trưng với độ phân giải khác nhau, cho phép nó phát hiện các đối tượng ở các kích thước và tỷ lệ khác nhau trong cùng một lần dự đoán.
* Hiệu suất tốt trong việc phát hiện đối tượng, đặc biệt là với các đối tượng có kích thước khác nhau trong hình ảnh.
* Có thể được huấn luyện end-to-end.

Nhược điểm:

* Yêu cầu một tập dữ liệu huấn luyện lớn và đa dạng.
* Cần phải được cấu hình và tinh chỉnh kỹ lưỡng để phù hợp với nhiều ứng dụng cụ thể.
* Không xác định vị trí của đối tượng nhỏ một cách chính xác bằng các phương pháp khác.
* Gặp khó khăn trong việc xử lý các đối tượng chồng lấp lên nhau.

**3. Thuật toán Faster R-CNN**

**3.1.Giới thiệu thuật toán**

Faster R-CNN là sự kết hợp của hai phần chính: mạng CNN (Convolutional Neural Network) để trích xuất đặc trưng từ hình ảnh và mô hình R-CNN để nhận diện đối tượng.

Nó đã thay đổi cách chúng ta thực hiện nhận diện đối tượng bằng cách đề xuất một cách tự động các khu vực quan trọng trên hình ảnh (Region Proposal) thay vì cách tiếp cận tạo ra các cửa sổ trượt (sliding windows) như các phương pháp trước đây.

**3.2.Kiến trúc Faster R-CNN**

* Convolutional Backbone Network: Faster R-CNN thường sử dụng một mạng CNN như VGG16 hoặc ResNet để trích xuất đặc trưng từ hình ảnh đầu vào.
* RPN (Region Proposal Network): RPN là một mạng CNN dự đoán các khu vực đề xuất trên hình ảnh. Nó sử dụng các anchor boxes (hộp neo) để đề xuất các vị trí tiềm năng chứa đối tượng. RPN tạo ra các ứng viên đề xuất và điều chỉnh chúng dựa trên thông tin đặc trưng từ mạng CNN cơ bản.
* ROI Pooling (Region of Interest Pooling): Sau khi có các đề xuất, ROI Pooling được sử dụng để đưa các vùng này về cùng kích thước để đưa vào mạng phân lớp và hồi quy.
* Classifier và Regressor: Mạng phân lớp định danh loại của đối tượng và mạng hồi quy điều chỉnh vị trí của các đối tượng dựa trên các đề xuất.

**3.3.Nguyên lý hoạt động**

* Đầu tiên, mạng CNN trích xuất đặc trưng từ hình ảnh.
* RPN dự đoán các đề xuất dựa trên thông tin đặc trưng này.
* ROI Pooling được sử dụng để trích xuất các đặc trưng từ các đề xuất và đưa chúng qua các mạng phân lớp và hồi quy.
* Kết quả cuối cùng là danh sách các đối tượng đã được nhận diện với vị trí và nhãn tương ứng.

**3.4.Ưu điểm và nhược điểm**

Ưu điểm:

* Tự động đề xuất vùng quan trọng, giúp giảm thiểu số lượng đề xuất và tăng tốc độ tính toán.
* Đạt được hiệu suất cao trên nhiều bài toán nhận diện đối tượng.

Nhược điểm:

* Tính phức tạp của kiến trúc có thể khiến việc triển khai và đào tạo trở nên phức tạp hơn so với các phương pháp đơn giản hơn.
* Cần nhiều dữ liệu đào tạo để đạt được hiệu suất tốt.

**4.Thuật toán CNNs**

**4.1.Giới thiệu thuật toán**

CNNs là một loại mạng nơ-ron sử dụng để xử lý dữ liệu dạng lưới như hình ảnh. Đặc trưng nổi bật của chúng là khả năng tự học các đặc trưng cấp thấp đến cấp cao từ dữ liệu hình ảnh mà không cần sự can thiệp của con người.

**4.2.Kiến trúc CNNs cơ bản**

* Lớp tích chập (Convolutional Layer): Là lớp quan trọng của CNNs, nó thực hiện các phép tích chập trên dữ liệu đầu vào bằng cách trượt qua các bộ lọc (kernels) để trích xuất các đặc trưng ẩn. Các đặc trưng này có thể là cạnh, góc, hoặc các đặc điểm phức tạp khác của hình ảnh.
* Lớp tổng hợp (Pooling Layer): Lớp này được sử dụng để giảm kích thước của dữ liệu đầu ra từ các lớp tích chập bằng cách thực hiện các phép tổng hợp như tổng hợp cực đại (max pooling) hoặc tổng hợp trung bình (average pooling).
* Lớp kết nối đầy đủ (Fully Connected Layer): Sau khi các lớp tích chập và lớp tổng hợp, CNN thường có một hoặc nhiều lớp kết nối đầy đủ để thực hiện phân loại dựa trên các đặc trưng đã trích xuất.

**4.3.Nguyên lý hoạt động:**

Mạng nơ-ron tích chập (CNNs) hoạt động bằng cách sử dụng các lớp tích chập để quét qua hình ảnh với các bộ lọc để trích xuất đặc trưng như góc, cạnh, hoặc hình dáng.

Các lớp tổng hợp sau đó giúp giảm kích thước của dữ liệu, giữ lại thông tin quan trọng và giảm độ phức tạp. Cuối cùng, các lớp kết nối đầy đủ sử dụng các đặc trưng đã trích xuất để thực hiện phân loại hoặc dự đoán.

Quá trình này cho phép CNNs tự động học và hiểu các đặc trưng quan trọng trong hình ảnh, làm cho chúng hiệu quả trong việc nhận diện và phân loại hình ảnh.

**4.4.Ưu điểm và nhược điểm:**

Ưu điểm:

* Khả năng trích xuất đặc trưng tự động: CNNs có khả năng tự động học và trích xuất các đặc trưng quan trọng từ dữ liệu hình ảnh, giúp làm giảm công việc tiền xử lý dữ liệu.
* Hiệu suất cao trong các bài toán thị giác máy tính: CNNs đã đạt được hiệu suất ấn tượng trong nhiều bài toán như phân loại hình ảnh, nhận diện đối tượng, nhận diện khuôn mặt và nhiều ứng dụng khác.
* Tích hợp chặt chẽ với dữ liệu hình ảnh: CNNs được thiết kế đặc biệt để xử lý dữ liệu hình ảnh và có khả năng hiểu cấu trúc không gian của nó, làm cho chúng rất hiệu quả trong việc xử lý hình ảnh và video.
* Phân loại nhiều lớp: CNNs có thể phân loại hình ảnh thành nhiều lớp, cho phép xác định đối tượng hoặc đặc trưng phức tạp.

* Tiến bộ và biến thể: Với sự phát triển liên tục, đã xuất hiện nhiều biến thể của CNNs như ResNet, Inception và EfficientNet, cải thiện hiệu suất và hiệu năng của mạng.

Nhược điểm:

* Yêu cầu lượng dữ liệu lớn: Để đạt được hiệu suất cao, CNNs thường đòi hỏi lượng dữ liệu huấn luyện lớn. Điều này có thể là một thách thức trong các tình huống có ít dữ liệu.
* Tính toán phức tạp: Các mạng CNN phức tạp có thể đòi hỏi nhiều tài nguyên tính toán, đặc biệt là khi triển khai trên thiết bị có tài nguyên hạn chế như điện thoại di động.
* Tùy chỉnh khó khăn: Để tùy chỉnh một mạng CNN cho một ứng dụng cụ thể, bạn cần kiến thức sâu về việc điều chỉnh kiến trúc và siêu tham số (hyperparameters), điều này có thể đòi hỏi nhiều thời gian và kỹ năng.
* Kích thước mô hình lớn: Mạng CNNs có thể có kích thước lớn, đặc biệt là trong các biến thể mạng sâu, làm cho việc triển khai và sử dụng chúng trên các thiết bị có tài nguyên hạn chế trở nên khó khăn.

**II. Đánh giá thuật toán.**

**1.Bài toán nhận dạng vị trí bàn tay.**

**1.1 Chuẩn bị dữ liệu.**

* Đọc dữ liệu các nhãn từ các file dữ liệu dạng xml đã thu được sau khi gán nhãn vào 32 mảng, tương ứng với 32 nhãn của bài toán.
* Dùng hàm shuffle để trộn ngẫu nhiên các mảng sau khi đã chia.
* Lấy 7/10 phần tử của từng mảng trong 32 mảng, thêm tất cả vào mảng train.
* Lấy 3/10 phần tử của từng mảng trong 32 mảng, thêm tất cả vào mảng test.

Lặp lại 5 lần các bước trên để lưu lại từng file train và test khác nhau

=> Thu được tổng 5 file train và 5 file test đi đôi với nhau được đánh số lần lượt: train\_1 - test\_1; train\_2 - test\_2; train\_3 - test\_3; train\_4 - test\_4; train\_5 - test\_5

**1.2 Huấn luyện mô hình.**

Bài toán thuộc dạng có nhiều hơn 2 nhãn, mỗi mẫu có thể có nhiều hơn 1 nhãn, nhãn của nó là 1 điểm ảnh(pixel). Trong quá trình thu nhập dữ liệu, một vùng ảnh có bồn thông số bao gồm 2 thông số đầu là tâm (x,y) của vùng ảnh, các  thông số tiếp theo là chiều rộng và chiều cao của ảnh. Diện tích của vùng này được coi là nhãn thực tế. Sau khi chạy mô hình 1 vùng ảnh cũng cho ra 4 thông số tương tự và diện tích của nó được coi là nhãn dự đoán.

**1.3 Quá trình đánh giá.**

* Bài toán thuộc dạng 1 mẫu có thể có nhiều nhãn. Khi thu thập dữ liệu và gán nhãn thu được các thông số của vùng ảnh chứa bàn tay. Gọi Ai là diện tích của nhãn thực tế chứa bàn tay, Bi là diện tích của nhãn dự đoán và Ci là vùng diện tích giao nhau giữa nhãn dự đoán(Ai) và nhãn thực tế(Bi). Ta có:
* Acci =  1 nếu Ci = Ai = Bi

  0 còn lại

* Prei =( |Ci| / |Bi| ) \* 100%
* Reci = (|Ci| / |Ai|) \* 100%
* F1-sci = 2\*Pre\*Rec **/** (Pre + Rec)
* Tính trung bình của các giá trị trên theo từng thuật toán với mỗi thuật toán 5 lần theo các lần train với 5 bộ dữ liệu đã chuẩn bị ở phía trên:

* Acc = 1nAccuracy(i)n \* 100%
* Pre = 1nPrecision(i)n \* 100%
* Rec = 1nRecall(i)n \* 100%
* F1 score = 2 \* Pre \* RecPre + Rec \* 100%

**1.4 Kết quả.**

**1.4.1 Thuật toán Yolo**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 86.98% | 87.65% | 88.37 | 87.01% | 88.82% | 87.77% |
| Pre (%) | 80.67% | 81.38% | 82.20% | 80.73% | 82.61% | 81.52% |
| Rec (%) | 83.23% | 83.80% | 84.71% | 83.26% | 85.02% | 84.00% |
| F1 score (%) | 81.93% | 82.57% | 83.44% | 81.98% | 83.80% | 82.74% |

**1.4.2 Thuật toán Faster R-CNN**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 88.01% | 89.12% | 87.52% | 90.23% | 85.78% | 88.13% |
| Pre (%) | 85.97% | 87.27% | 85.41% | 87.89% | 83.97% | 86.10% |
| Rec (%) | 89.35% | 90.19% | 88.19% | 91.43% | 87.26% | 89.28% |
| F1 score (%) | 87.63% | 88.71% | 86.78% | 89.63% | 85.58% | 87.66% |

**1.4.3 Thuật toán SSD**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 83.91% | 84.35% | 85.27% | 82.98% | 86.12% | 84.53% |
| Pre (%) | 82.83% | 83.26% | 84.12% | 81.75% | 85.10% | 83.41% |
| Rec (%) | 84.78% | 85.19% | 86.41% | 84.13% | 87.14% | 85.53% |
| F1 score (%) | 83.79% | 84.21% | 85.25% | 82.92% | 86.11% | 84.47% |

**1.4.4 Thuật toán CNNs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 84.41% | 85.13% | 84.19% | 85.76% | 86.24% | 85.15% |
| Pre (%) | 83.02% | 83.81% | 82.72% | 84.53% | 85.19% | 83.85% |
| Rec (%) | 85.12% | 86.31% | 85.34% | 86.74% | 87.35% | 86.17% |
| F1 score (%) | 84.06% | 85.04% | 84.01% | 85.62% | 86.26% | 84.99% |

**=> Kết quả chung:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Thuật toán** | **Yolo** | **Faster R-CNN** | **SSD** | **CNNs** |
| **Acc (%)** | 87.77% | 88.13% | 84.53% | 85.15% |
| **Pre (%)** | 81.52% | 86.10% | 83.41% | 83.85% |
| **Rec (%)** | 84.00% | 89.28% | 85.53% | 86.17% |
| **F1-score (%)** | 82.74% | 87.66% | 84.47% | 84.99% |

Từ kết quả trên, ta chọn thuật toán Faster R-CNN do có F1-score là lớn nhất (87.66%):

* Faster R-CNN (87.66%): Có F1-score cao nhất trong 4 thuật toán. Có thể là do R-CNN thường có kiến trúc mạng neural phức tạp và chia sẻ trọng số và khả năng phát hiện nhiều vùng đối tượng khác nhau trong hình ảnh, bao gồm cả vùng bàn tay. Điều này giúp nó xác định chính xác vị trí và đường biên của bàn tay.
* Yolo (82.74%): Có F1-score thấp nhất trong 4 thuật toán. Có thể là do kiến trúc mạng Yolo được thiết kế để cân nhắc cả việc phát hiện vị trí và phân loại các đối tượng trong một lần. Tuy nhiên, kiến trúc này có thể không thích hợp cho các nhiệm vụ phức tạp như nhận diện bàn tay, nơi việc định vị và phân loại các đối tượng chính xác có thể đòi hỏi một mức độ phức tạp hơn.
* SSD (84.47%): Có F1-score thấp hơn Faster R-CNN, CNNs và cao hơn Yolo. Do SSD cũng có khả năng phát hiện nhiều đối tượng và đa vùng, nhưng có thể không chính xác bằng R-CNN trong một số trường hợp.
* CNNs (84.99%): Có F1-score thấp hơn Faster R-CNN và cao hơn 2 thuật toán còn lại. CNNs là một loạt các kiến trúc mạng neural sử dụng rộng rãi trong thị giác máy tính và xử lý hình ảnh và thường được sử dụng trong quy trình tiền xử lý dữ liệu trước khi sử dụng thuật toán phát hiện đối tượng khác như R-CNN.

**2. Bài toán chuyển đổi ngôn ngữ ký hiệu thành văn bản**

**2.1 Chuẩn bị dữ liệu**

* Đọc dữ liệu các nhãn từ các file dữ liệu dạng xml đã thu được sau khi gán nhãn vào 32 mảng, tương ứng với 32 nhãn của bài toán.
* Dùng hàm shuffle để trộn ngẫu nhiên các mảng sau khi đã chia.
* Lấy 7/10 phần tử của từng mảng trong 32 mảng, thêm tất cả vào mảng train.
* Lấy 3/10 phần tử của từng mảng trong 32 mảng, thêm tất cả vào mảng test.

Lặp lại 5 lần các bước trên để lưu lại từng file train và test khác nhau

=> Thu được tổng 5 file train và 5 file test đi đôi với nhau được đánh số lần lượt: train\_1 - test\_1; train\_2 - test\_2; train\_3 - test\_3; train\_4 - train\_4; train\_5 - train\_5

**2.2 Huấn luyện mô hình**

**2.2.1 Thuật toán Yolo**

Sử dụng mạng neural tích chập để thực hiện các nhiệm vụ này và được triển khai thông qua thư viện như YOLO v5. YOLO v5 được phát triển và triển khai thông qua các thư viện như PyTorch và được sử dụng rộng rãi trong lĩnh vực thị giác máy tính

**2.2.2 Thuật toán Faster R-CNN**

Sử dụng mạng neural tích chập (CNN) và mạng neural địa chỉ (Region Proposal Network - RPN) để thực hiện việc phát hiện và phân loại đối tượng trong hình ảnh và thường được triển khai thông qua các thư viện như TensorFlow hoặc PyTorch.

**2.2.3 Thuật toán SSD**

Sử dụng thư viện Tensorflow tiền xử lý dữ liệu, chuẩn bị dữ liệu trước khi đưa vào mô hình. Sử dụng hàm tf.GradientTape để tính gradient của hàm mất mát và sau đó áp dụng các thuật toán tối ưu hóa để cập nhật trọng số cho mô hình.

**2.2.4 Thuật toán CNNs**

Sử dụng thư viện Keras, tiền xử lý dữ liệu hình ảnh thông qua việc sử dụng các lớp Convolutional và Pooling. Mục tiêu là tạo và huấn luyện mô hình CNNs bằng hàm model.fit() để thực hiện các tác vụ liên quan đến hình ảnh như phân loại, phát hiện, hoặc nhận dạng.

**2.3 Quá trình đánh giá**

* Bài toán thuộc dạng 1 mẫu có thể có nhiều nhãn. Ví dụ: mẫu 1 là tập dữ liệu gồm các file miêu tả các chữ cái H,E,L,L,O là nhãn thực tế và sau khi chạy mô hình kết quả trả về các nhãn dự đoán H,S,L,L,O. Khi đó:

-Gọi Ai là tập các nhãn gốc của mẫu thứ i

- Gọi Bi là tập nhãn dự đoán cho mẫu i

- Ci = Ai giao Bi

* Acci =  1 nếu Ci = Ai = Bi

  0 còn lại

* Prei =( |Ci| / |Bi| ) \* 100%
* Reci = (|Ci| / |Ai|) \* 100%
* F1-sci = 2\*Pre\*Rec **/** (Pre + Rec)

**2.4 Kết quả**

**2.4.1 Thuật toán Yolo**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 85.13% | 89.19% | 90.87% | 88.48% | 86.69% | 88.07% |
| Pre (%) | 88.81% | 87.72% | 85.23% | 81.97% | 85.92% | 85.93% |
| Rec (%) | 86.31% | 85.34% | 87.49% | 84.27% | 88.11% | 86.30% |
| F1 score (%) | 87.54% | 86.51% | 86.35% | 83.10% | 87.00% | 86.11% |

**2.4.2 Thuật toán Faster R-CNN**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 87.11% | 86.42% | 83.10% | 85.21% | 89.43% | 86.25% |
| Pre (%) | 85.84% | 84.23% | 82.21% | 82.83% | 87.02% | 84.43% |
| Rec (%) | 83.20% | 87.24% | 84.38% | 86.21% | 90.12% | 86.23% |
| F1 score (%) | 84.50% | 85.71% | 83.28% | 84.49% | 88.54% | 85.32% |

**2.4.3 Thuật toán SSD**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 82.34% | 83.55% | 84.21% | 87.23% | 86.32% | 84.73% |
| Pre (%) | 80.32% | 86.12% | 84.34% | 89.32% | 82.51% | 84.52% |
| Rec (%) | 79.30% | 84.42% | 81.90% | 90.12% | 88.24% | 84.80% |
| F1 score (%) | 79.81% | 85.26% | 83.10% | 89.72% | 85.28% | 84.66% |

**2.4.4 Thuật toán CNNs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lần train/test | Lần 1 | Lần 2 | Lần 3 | Lần 4 | Lần 5 | Trung bình |
| Acc (%) | 78.61% | 80.35% | 80.27% | 77.98% | 81.12% | 79.67% |
| Pre (%) | 77.23% | 78.26% | 78.12% | 76.75% | 80.10% | 78.09% |
| Rec (%) | 78.78% | 80.19% | 80.41% | 79.13% | 81.14% | 79.93% |
| F1 score (%) | 80.00% | 79.21% | 79.25% | 77.92% | 80.62% | 79.00% |

**=> Kết quả chung:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Thuật toán** | **Yolo** | **Faster R-CNN** | **SSD** | **CNNs** |
| **Acc (%)** | 88.07% | 86.25% | 84.73% | 79.67% |
| **Pre (%)** | 85.93% | 84.43% | 84.52% | 78.09% |
| **Rec (%)** | 86.30% | 86.23% | 84.80% | 79.93% |
| **F1-score (%)** | 86.11% | 85.32% | 84.66% | 79.00% |

Từ kết quả trên, ta chọn thuật toán Yolo do có F1-score là lớn nhất (86.11%)

* Yolo (86.11%): Có F1-score lớn nhất so với 3 thuật toán Faster R-CNN, SSD, CNNs. Có thể là do YOLO sử dụng một mạng neural duy nhất để dự đoán đối tượng và vị trí của chúng trong một lần điều chỉnh, giúp tối ưu hóa thời gian tính toán.
* Faster R-CNN (85.32%): Có F1-score tốt hơn so với SSD và CNNs. Faster R-CNN phát hiện vùng đối tượng trước, sau đó cải thiện định vị và phân loại đối tượng bằng mạng neural chia sẻ. Thuật toán này thích hợp cho các ứng dụng đòi hỏi độ chính xác cao nhưng không quan trọng về thời gian đáp ứng nhanh.
* SSD (84.66%): Có F1-score thấp hơn so với 2 thuật toán Yolo và Faster R-CNN một chút. Dù SSD có khả năng kết hợp phát hiện đối tượng và phân loại trên các lớp khác nhau cùng một lúc, nhưng điều này có thể làm giảm độ chính xác so với cách tiếp cận dựa trên vùng của R-CNN.
* CNNs (79.00%): Là thuật toán có F1-score thấp nhất trong 4 thuật toán. Có thể là do CNNs thường được sử dụng để trích xuất đặc trưng hoặc phân loại hình ảnh chứ không phải để phát hiện đối tượng trực tiếp. Điều này có nghĩa rằng CNNs không được thiết kế đặc biệt để xác định vị trí và phân loại các đối tượng trong hình ảnh, mà thường được sử dụng trong các bước tiền xử lý hoặc trích xuất đặc trưng trước khi áp dụng các thuật toán phát hiện đối tượng.

**Chương II : Thiết kế hệ thống**

**Phần A : Thiết kế hậu cần server**

1. Thiết kế CSDL cho quản lý mẫu

A diagram of a data processing process

Description automatically generated with medium confidence

1. Chức năng thêm mẫu

A screenshot of a computer screen

Description automatically generated

Giao diện quản lý mẫu

A login form with text and a box

Description automatically generated

Giao diện đăng nhập

A screenshot of a computer screen

Description automatically generated

Giao diện thêm mẫu

A screenshot of a computer program

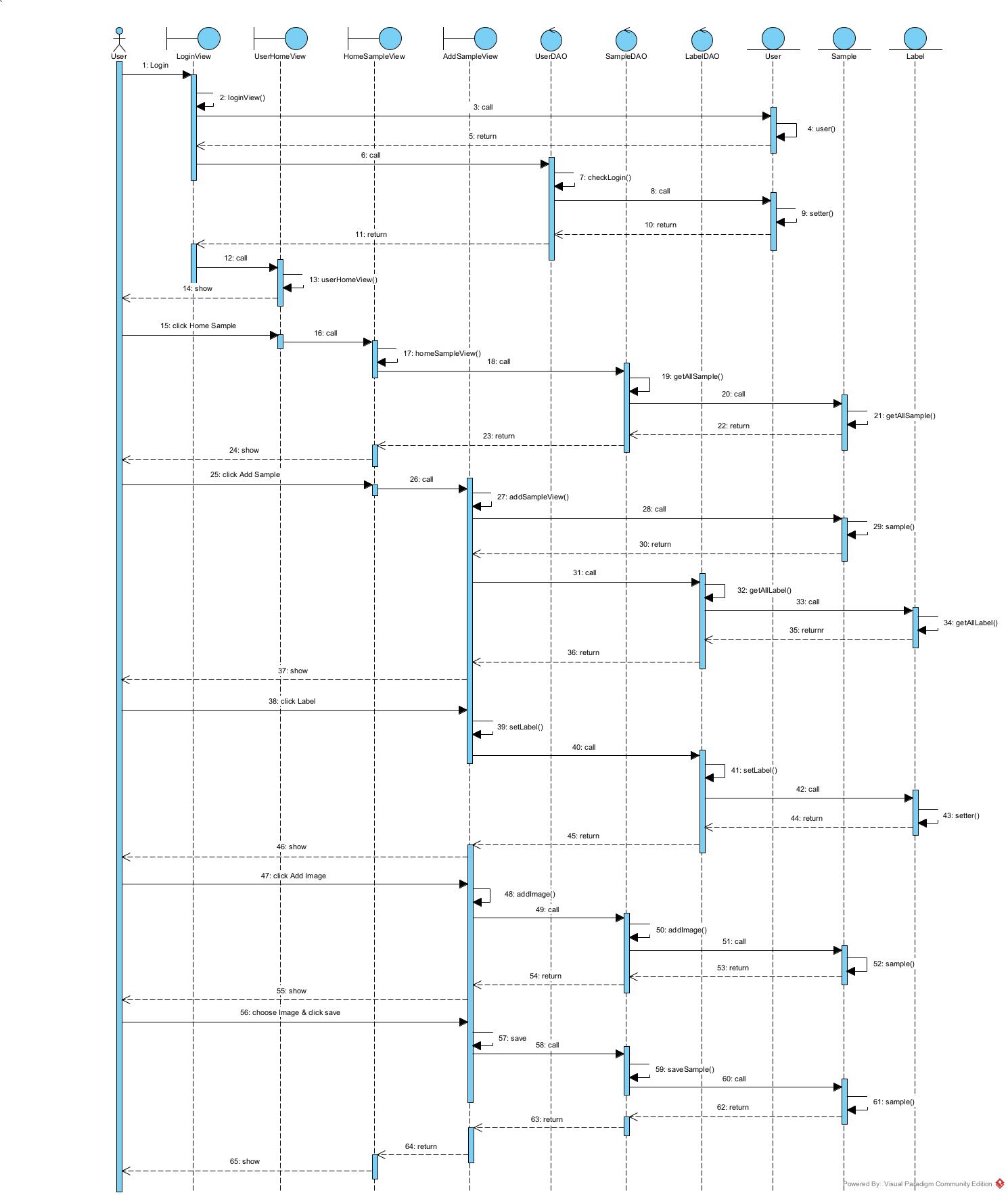
Description automatically generated

Giao diện trang chủ người dùng

A computer screen shot of a computer code

Description automatically generated with medium confidence

Biểu đồ lớp chi tiết chức năng thêm mẫu



Biểu đồ tuần tự chức năng thêm mẫu

1. Chức năng sửa mẫu

A screenshot of a computer

Description automatically generated

Giao diện sửa mẫu

A screenshot of a computer program

Description automatically generated

Giao diện trang chủ của người dùngA login screen with text and a box

Description automatically generated

Giao diện đăng nhập của người dùng

A diagram of a computer

Description automatically generated

Biểu đồ lớp chi tiết chức năng sửa mẫu

A diagram of a diagram

Description automatically generated

Biểu đồ tuần tự chức năng sửa mẫu

1. Chức năng xóa mẫu

A black and white diagram

Description automatically generated

Giao diện xóa mẫu

A screenshot of a computer program

Description automatically generated

Giao diện trang chủ của người dùng

A login screen with text and a box

Description automatically generated

Giao diện đăng nhập

A diagram of a computer

Description automatically generated with medium confidence

Biểu đồ lớp chi tiết chức năng xóa mẫu

A diagram of a project

Description automatically generated

Biểu đồ tuần tự chức năng xóa mẫu

**Phần B : Thiết kế server xử lý thông minh**

Quản lý các thao tác train, retrain cho bài toán chuyển đổi hình ảnh có ý nghĩa thành văn bản

Thiết kế cơ sở dữ liệu :

A diagram of a data flow

Description automatically generated

Sample THIEU LINK, BO STATUS

Biều đồ lớp chi tiết cho quản lý model :

A diagram of a computer program

Description automatically generated with medium confidence

Biểu đồ tuần tự quản lý model :

A diagram of a project

Description automatically generated