



TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO

PBL5 - ĐỒ ÁN KỸ THUẬT MÁY TÍNH

TÊN ĐỀ TÀI

HỆ THỐNG PHÁT HIỆN TẾ NGÃ TRONG PHÒNG BẰNG CAMERA

Cán bộ doanh nghiệp hướng dẫn: Trần Văn Huy

Giảng viên đồng hướng dẫn: TS. Ninh Khánh Duy

STT NHÓM: 15 HỌ VÀ TÊN SINH VIÊN	LỚP HỌC PHẦN ĐỒ ÁN
Phan Tiến Đạt	20.10B
Nguyễn Trần Mỹ Duyên	20.10B
Trần Thị Ngà	20.10B
Nguyễn Phước Đại Toàn	20.10B

ĐÀ NẴNG, 06/2023

TÓM TẮT ĐỒ ÁN

Té ngã là một vấn đề rất phổ biến ở người cao tuổi, và nó có thể gây ra những hậu quả nghiêm trọng cho sức khỏe của họ. Khi người cao tuổi bị ngã, họ có thể gặp nguy hiểm từ những thương tích nghiêm trọng, hoặc thậm chí là tử vong nếu không được cứu giúp kịp thời. Vì vậy, việc phát hiện sớm người bị ngã là rất quan trọng để giảm thiểu những rủi ro không đáng có.

Để hỗ trợ việc giám sát và phát hiện người bị ngã, đề tài “Hệ thống phát hiện té ngã trong phòng bằng camera” đã được thực hiện. Hệ thống này sử dụng luồng video trực tiếp từ camera và được xử lý trên Jetson Nano Develop để phát hiện người bị ngã trong phòng. Khi phát hiện ra một trường hợp người bị ngã, hệ thống sẽ thông báo cho người dùng qua các kênh như ứng dụng mobile, SMS hoặc điện thoại để giúp họ có thể cứu giúp nạn nhân kịp thời và tránh những hậu quả đáng tiếc.

Để đạt được kết quả tốt nhất, đề tài đã thử nghiệm mô hình học máy như SVM và mô hình học sâu dựa trên chuỗi thời gian LSTM. Các kết quả thử nghiệm cho thấy mô hình tốt nhất đã có khả năng phát hiện được các trường hợp té ngã với độ chính xác cao, đồng thời đáp ứng được việc té ngã được phát hiện trong thời gian thực. Với sự hỗ trợ của hệ thống này, việc phát hiện sớm các trường hợp người bị ngã sẽ giúp giảm thiểu những rủi ro và bảo vệ sức khỏe của người cao tuổi.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Đánh giá
Phan Tiến Đạt	<ul style="list-style-type: none">- Thu thập và xử lý dữ liệu.- Tìm hiểu các thuật toán khử nền và train model bằng SVM.	Đã hoàn thành
Nguyễn Trần Mỹ Duyên	<ul style="list-style-type: none">- Config Jetson Nano.- Tìm hiểu các bước tiền xử lý, trích xuất đặc trưng và train với phương pháp Bottom-up .	Đã hoàn thành
Trần Thị Ngà	<ul style="list-style-type: none">- Thu thập và xử lý dữ liệu.- Xây dựng ứng dụng Mobile.	Đã hoàn thành
Nguyễn Phước Đại Toàn	<ul style="list-style-type: none">- Thu thập và xử lý dữ liệu.- Custom train yolov8 segment.- Tìm hiểu các bước tiền xử lý, trích xuất đặc trưng và train với phương pháp Top-Down .	Đã hoàn thành

MỤC LỤC

1. Giới thiệu	8
1.1. Thực trạng.....	8
1.2. Các vấn đề cần giải quyết.....	9
1.3. Đề xuất giải pháp tổng quan.....	9
2. Giải pháp.....	10
2.1. Giải pháp về phần cứng và truyền thông.....	10
2.1.1. Danh sách linh kiện	10
2.1.2. Cơ sở lý thuyết phần cứng.....	11
2.1.2.1. Jetson Nano Developer Kit B01	11
2.1.2.2. Rapoo Camera C260.....	12
2.1.2.3. Bộ thu sóng TP-LINK TL-WN725N	12
2.2. Giải pháp về phần mềm.....	13
2.2.1. Phát hiện té ngã dựa trên hướng tiếp cận Top-Down.....	13
2.2.2. Phát hiện té ngã dựa trên hướng tiếp cận Bottom-up	21
2.2.3. Firebase.....	22
2.2.4. ClickSend.....	25
2.2.5. Giải pháp về ứng dụng di động	26
3. Kết quả.....	27
3.1. Dataset	27
3.2. Kết quả của phương pháp Top-down	28
3.3. Kết quả của phương pháp Bottom-up.....	32
3.4. So sánh kết quả giữa hai phương pháp	34
3.5. Kết quả deploy trên Jetson Nano Developer Kit B01	35
3.6. Kết quả xây dựng App Mobile	36
4. Kết luận.....	36
4.1. Kết luận.....	36
4.2 Hướng phát triển.....	36
5. Danh mục tài liệu tham khảo.....	37

DANH MỤC BẢNG BIỂU

Bảng 1. Đề xuất giải pháp tổng quan	9
Bảng 2. Danh sách linh kiện cần thiết	10
Bảng 3. Mô tả dữ liệu dataset	27
Bảng 4. Bảng kết quả phương pháp Top-down.....	30
Bảng 5. Bảng kết quả phương pháp Bottom-up	32
Bảng 6. Bảng kết quả so sánh 2 phương pháp.....	34

DANH MỤC HÌNH VẼ

Hình 1. Sơ đồ tổng thể của hệ thống	10
Hình 2. Sơ đồ thuật toán phương pháp Top-down	13
Hình 3. Hình ảnh so sánh các phiên bản của YOLO.....	14
Hình 4. Kiến trúc Yolov8	15
Hình 5. Trích xuất đặc trưng ROI	16
Hình 6. Minh hoạ về hyperplane	17
Hình 7. Ví dụ về chọn hyperplane.....	17
Hình 8. Minh hoạ về margin.....	18
Hình 9. Mô hình LSTM.....	19
Hình 10. Output của lớp LSTM mặc định.....	20
Hình 11. Output của lớp LSTM có return_sequence	20
Hình 12. Sơ đồ thuật toán phương pháp Bottom-up	21
Hình 13. Các vị trí trên khung xương.....	22
Hình 14. Hình ảnh lưu dữ liệu trên realtime database.....	23
Hình 15. Hình ảnh lưu video trên firebase storage.....	24
Hình 16. Hình ảnh demo gửi thông báo sử dụng FCM	24
Hình 17. Hình ảnh các hàm trên Firebase Functions	25
Hình 18. Tổ chức thư mục dataset.....	27
Hình 19. Các chỉ số đánh giá khi train yolov8-seg	28
Hình 20. Ma trận nhầm lẫn khi train yolov8-seg	29
Hình 21. Kết quả segment với yolov8-seg	29
Hình 22. Ma trận nhầm lẫn phương pháp Top-down với LSTM	30
Hình 23. Đồ thị accuracy phương pháp Top-down	31
Hình 24. Đồ thị loss phương pháp Top-down	31
Hình 25. Kết quả nhận diện phương pháp Top-down với LSTM	31
Hình 26. Độ chính xác phương pháp Top-down với SVM	31
Hình 27. Ma trận nhầm lẫn phương pháp Top-down với SVM	32
Hình 28. Kết quả nhận diện phương pháp Top-down với SVM	32
Hình 29. Ma trận nhầm lẫn phương pháp Bottom-up	33

Hình 30. Đồ thị accuracy phương pháp Bottom-up	33
Hình 31. Đồ thị loss phương pháp Bottom-up	33
Hình 32. Kết quả nhận diện phương pháp Bottom-up	33
Hình 33. Kết quả nhận diện hành động quỳ gối	35
Hình 34. Kết quả nhận diện hành động đi	35
Hình 35. Kết quả nhận diện hành động ngã	35
Hình 36. Giao diện app mobile.....	36

1. Giới thiệu

1.1. Thực trạng

Ngày nay, tỉ lệ người bị đột quỵ và tỉ lệ già hóa đang ngày càng tăng cao. Theo Hội đột quỵ thế giới 2022, mỗi năm thế giới có hơn 12,2 triệu ca đột quỵ não mới, trong đó hơn 16% xảy ra ở người trẻ 15-49 tuổi. Về con số tử vong, mỗi năm có tới 6,5 triệu ca với hơn 6% trong số đó là người người trẻ [16]. Trong quá trình phát hiện và điều trị bệnh đột quỵ thì thời gian phát hiện chiếm vai trò cốt lõi quyết định tỷ lệ sống của bệnh nhân. Đa phần các ca tử vong do đột quỵ và diễn biến nặng của người già khi ngã là do khi bị ngã ở nơi vắng người không được phát hiện kịp thời. Do đó, việc phát hiện té ngã sớm là cần thiết để giúp tăng cường khả năng sống sót của bệnh nhân.

Hiện nay có rất nhiều hệ thống cảnh báo ngã dựa vào cảm biến hoặc camera có thể kể đến như:

- Thiết bị đeo tay Fall Detector của ELDERCARE giám sát và phát hiện ngã dành cho người cao tuổi hoặc người khuyết tật. Nó được thiết kế dưới dạng một chiếc đồng hồ đeo tay, được trang bị các cảm biến đo chuyển động và gia tốc.
- Hệ thống Bitcare sử dụng các cảm biến áp suất được lắp đặt trên giường bệnh và xe lăn để giám sát và phân tích hoạt động của bệnh nhân.
- Hệ thống phát hiện ngã sử dụng các cảm biến gia tốc và mô hình học sâu của công ty Waylay: Hệ thống này sử dụng các cảm biến gia tốc được lắp đặt trên cơ thể để giám sát chuyển động của người dùng và đưa ra cảnh báo nếu phát hiện ngã.
- Hệ thống phát hiện ngã sử dụng đặc trưng chuyển động và hình dạng cơ thể dựa trên camera đơn của Trường đại học Công nghệ, Đại học Quốc gia Hà Nội.

Trong lĩnh vực phát hiện ngã, hệ thống sử dụng các cảm biến cố định chỉ dành riêng cho một người và vị trí duy nhất, mặc dù đem lại hiệu quả cao nhưng chi phí lại rất đắt đỏ. Trong khi đó, sử dụng hệ thống phát hiện ngã bằng hình ảnh và video sử dụng camera, cho phép phát hiện được nhiều người khác nhau mà không cần sử dụng các cảm biến.

Với sự phát triển của công nghệ, camera đã trở thành một yếu tố quan trọng cho việc phát triển các hệ thống phát hiện ngã một cách nhanh chóng và tiện lợi. Nhóm nghiên cứu cũng tận dụng ưu điểm này và phát triển đề tài trên một khu vực giới hạn là trong phạm vi nhà ở để tiện cho quá trình nghiên cứu.

1.2. Các vấn đề cần giải quyết

- Cần có các thiết bị phần cứng để thu nhận dữ liệu.
- Phát hiện và cảnh báo về ứng dụng, điện thoại khi té ngã
- Hệ thống chạy theo thời gian thực.

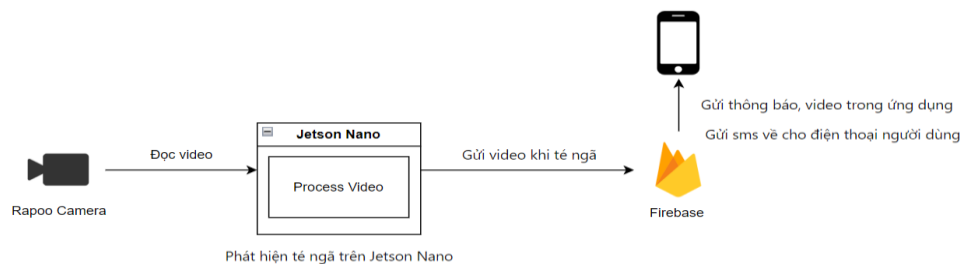
1.3. Đề xuất giải pháp tổng quan

Bảng 1. Đề xuất giải pháp tổng quan

Vấn đề	Giải pháp đề xuất
Phần cứng	Jetson Nano Developer Kit B01 Rapoo Camera Điện thoại thông minh
Phát hiện té ngã	Xây dựng và huấn luyện mô hình phát hiện té ngã Thử nghiệm theo hai hướng tiếp cận: Top-down và Bottom-up Framework với các model: SVM, LSTM,... Huấn luyện trên google colab
Ứng dụng	Xây dựng ứng dụng điện thoại Người dùng có thể đăng nhập, đăng xuất Xem được video té ngã và nhận được thông báo

2. Giải pháp

2.1. Giải pháp về phần cứng và truyền thông



Hình 1. Sơ đồ tổng thể của hệ thống

Phần xử lý chính của hệ thống nằm ở Jetson Nano Developer Kit B01 để đảm bảo xử lý được thực hiện một cách realtime. Đầu tiên, sử dụng Rapoo Camera để quay video, Jetson Nano sẽ đọc luồng video và xử lý phát hiện té ngã trên đó. Nếu phát hiện té ngã thì Jetson Nano sẽ gửi video té ngã lên Firebase Realtime Database. Sau đó, sẽ cập nhật về ứng dụng người dùng video té ngã cùng với các thông báo sms về điện thoại.

2.1.1. Danh sách linh kiện

Bảng 2. Danh sách linh kiện cần thiết

Tên linh kiện	Chức năng	Đơn giá (VNĐ)
Jetson Nano Developer Kit B01	Xử lý phát hiện té ngã	5,000,000
Rapoo camera C260	Đọc video	750,000
Dây dẫn	Truyền tải điện năng	10,000
Nguồn Adapter 5A	Cấp nguồn cho Jetson Nano	195,000
Bộ thu sóng TP-LINK TL-WN725N	Giúp Jetson Nano kết nối với mạng wifi	145,000

2.1.2. Cơ sở lý thuyết phần cứng

2.1.2.1. Jetson Nano Developer Kit B01

Jetson Nano Developer Kit B01 là một bo mạch tích hợp được thiết kế để cung cấp khả năng xử lý AI và computer vision mạnh mẽ cho các ứng dụng IoT, robot và các ứng dụng AI khác.

❖ Các tham số kỹ thuật đầu vào:

- Nguồn điện: 5V DC (2A)
- Cổng nguồn: Micro-USB hoặc DC jack
- RAM: 4GB LPDDR4
- CPU: ARM Cortex-A57 quad-core 64-bit @ 1.43 GHz
- GPU: 128-core NVIDIA Maxwell GPU
- Tích hợp camera: CSI hoặc USB

❖ Các tham số kỹ thuật đầu ra:

- HDMI (1.4) (4K-Capable HDMI and DisplayPort Output): Tính năng giải mã video với độ phân giải tối đa 4Kp30 và hỗ trợ định dạng video H.264 và H.265.
- USB 3.0 Ports (4x): có 4 cổng USB 3.0 để kết nối với các thiết bị USB khác như chuột, bàn phím, ổ cứng ngoài, thiết bị ghi hình,...
- Gigabit Ethernet Port: Đây là cổng kết nối Ethernet với tốc độ Gigabit, cho phép Jetson Nano Developer Kit B01 kết nối với mạng Internet hoặc mạng LAN.
- MIPI-CSI: Jetson Nano Developer Kit B01 cung cấp một cổng MIPI-CSI để kết nối với các camera hỗ trợ MIPI-CSI. Nó hỗ trợ lên đến 12 kênh và hỗ trợ tối đa độ phân giải 6K.
- 40 Pin Expansion Header: GPIO, I2C, I2S, SPI, UART: Đây là các cổng kết nối GPIO, I2C, I2S, SPI, UART cho phép Jetson Nano Developer Kit B01 kết nối với các thiết bị khác như cảm biến, thiết bị ngoại vi, v.v.

- ❖ **Nguyên tắc hoạt động:** Jetson Nano là một developer kit sử dụng GPU tích hợp của NVIDIA để xử lý dữ liệu AI và computer vision. Kit này sử dụng các công nghệ CUDA, cuDNN và TensorRT để tăng tốc độ xử lý dữ liệu và tính toán. Nó cũng tích hợp sẵn các thư viện và công cụ phát triển như OpenCV, TensorFlow và PyTorch để giúp các nhà phát triển dễ dàng xây dựng các ứng dụng AI và computer vision trên nền tảng này.

2.1.2.2. Rapoo Camera C260

- ❖ **Tham số kỹ thuật đầu vào:** Camera kết nối với máy tính hoặc thiết bị khác thông qua cổng USB 2.0 hoặc 3.0.
- ❖ **Tham số kỹ thuật đầu ra:**
 - Độ phân giải: Camera hỗ trợ độ phân giải đến 720p (1280x720 pixels) và 30 khung hình/giây cho chất lượng video tương đối tốt.
 - Góc quan sát: Rapoo Camera C260 có khả năng xoay 360 độ và điều chỉnh được độ nghiêng để phù hợp với nhu cầu sử dụng của người dùng.
 - Âm thanh: Camera tích hợp microphone với khả năng thu âm tốt, hỗ trợ giảm tiếng ồn và lọc tiếng vang.
 - Đèn LED: Có đèn LED trên camera giúp cải thiện chất lượng hình ảnh trong môi trường ánh sáng yếu.
- ❖ **Nguyên tắc hoạt động:**
 - Khi được kết nối với máy tính hoặc thiết bị khác thông qua cổng USB, driver của camera sẽ được cài đặt tự động để đảm bảo tương thích với hệ điều hành của thiết bị đó. Hoặc người dùng tự cài đặt driver tương thích.
 - Người dùng có thể sử dụng phần mềm của bên thứ ba để truy cập video và âm thanh từ camera.

2.1.2.3. Bộ thu sóng TP-LINK TL-WN725N

- ❖ **Các thông số kỹ thuật:**
 - Đầu vào: Cổng USB 2.0
 - Tần số: 2.4 GHz, chuẩn Wi-Fi: IEEE 802.11b/g/n

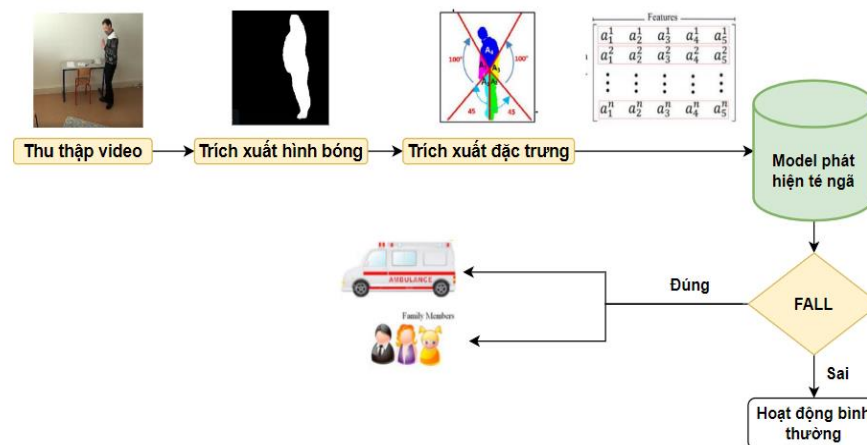
- Tốc độ truyền tải dữ liệu tối đa: 150 Mbps
- Băng thông: 20 MHz hoặc 40 MHz
- Công suất phát: 17 dBm (EIRP), độ nhạy thu tối đa: -96 dBm
- Hỗ trợ mã hóa bảo mật WEP, WPA/WPA2, WPA-PSK/WPA2-PSK (AES/TKIP)

❖ **Nguyên tắc hoạt động:** Thu nhận sóng Wifi từ thiết bị phát và chuyển đổi nó thành dữ liệu số để truyền qua cổng USB đến máy tính hoặc laptop.

2.2. Giải pháp về phần mềm

2.2.1. Phát hiện té ngã dựa trên hướng tiếp cận Top-Down

❖ **Mô hình thuật toán:**



Hình 2. Sơ đồ thuật toán phương pháp Top-down

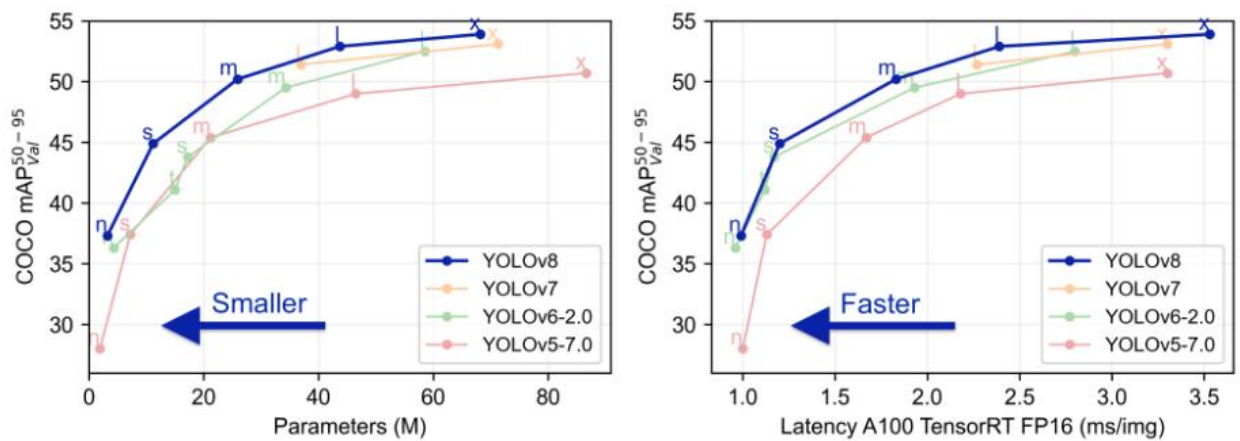
Ý tưởng chính của phương pháp Top-down là tìm vùng chứa cơ thể người và sau đó tiến hành trích xuất đặc trưng (ước tính tư thế) của con người trong vùng chứa đó. Từ các đặc trưng tìm được, ta tiến hành huấn luyện các model để phát hiện té ngã. Trong phạm vi đồ án, nhóm chọn mô hình SVM và LSTM để tiến hành huấn luyện, đánh giá và so sánh kết quả của mô hình học máy SVM và mô hình học sâu chuỗi thời gian LSTM.

Ưu điểm của phương pháp này là trích xuất hình bóng cơ thể mang lại những đặc trưng quan trọng liên quan đến tư thế con người. Tuy nhiên, phương pháp này tốn kém chi phí về mặt tính toán cũng như việc trích xuất hình bóng cơ thể chính xác trong thời gian thực là một việc tương đối không dễ dàng.

❖ Trích xuất hình bóng cơ thể:

Mô-đun phân đoạn trích xuất hình bóng cơ thể từ nền là trung tâm của tính năng phát hiện ngã dựa trên video. Từ đó ta có thể trích xuất đặc trưng dựa vào hình bóng cơ thể này.

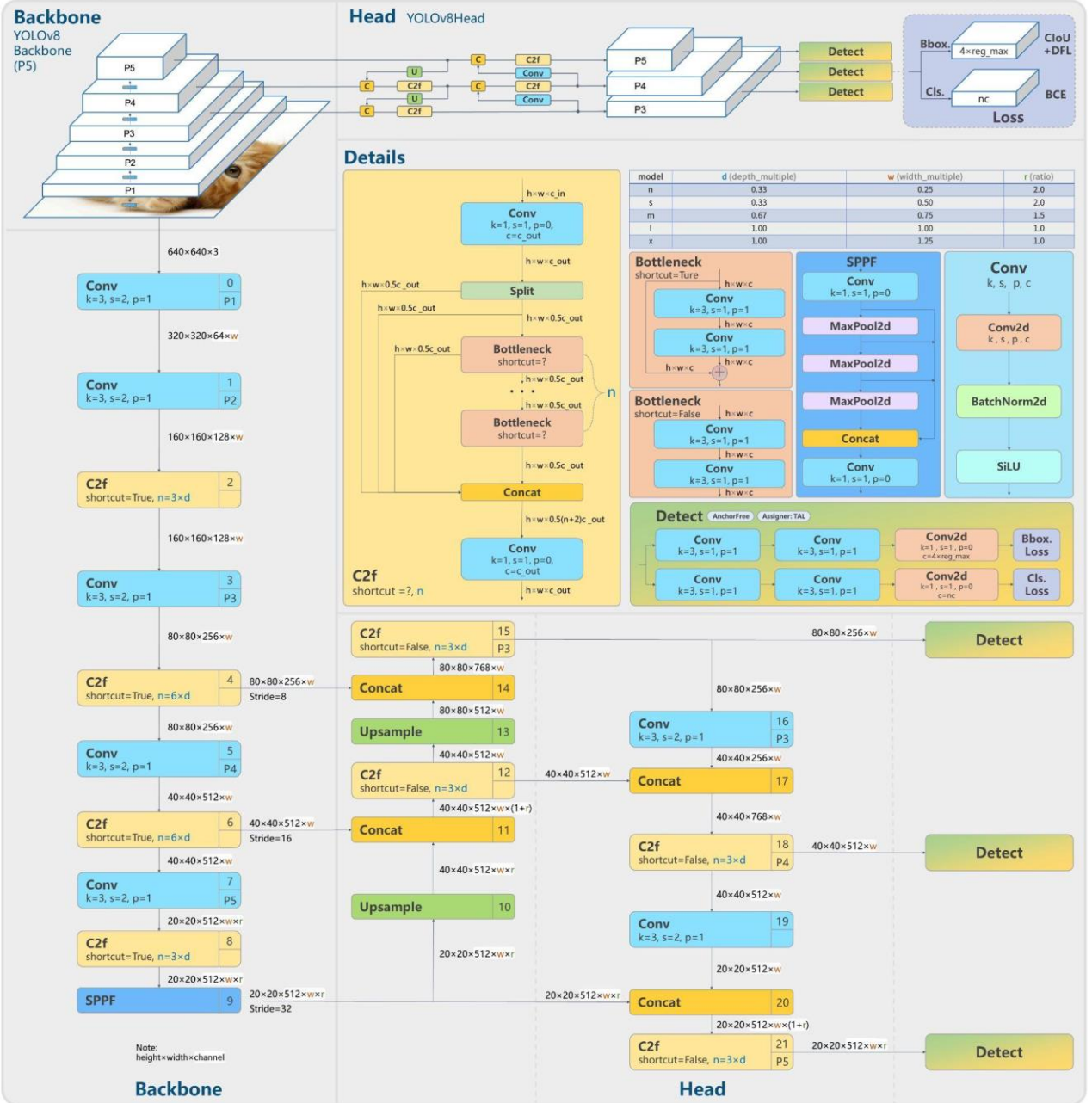
Hiện nay có rất nhiều thuật toán khử nền trích xuất hình bóng cơ thể như MOG, MOG2, KNN của thư viện OpenCV,... Ngoài ra còn có các mô hình nổi tiếng như Mask R-CNN, YOLO,... So với họ mô hình R-CNN, YOLO có tốc độ được cải thiện rất nhiều do chỉ sử dụng mô hình one-stage thay vì mô hình two-stage như các mô hình R-CNN. Sau quá trình tìm hiểu thì nhóm quyết định chọn mô hình YOLO v8 để thực hiện khử nền và trích xuất hình bóng cơ thể nhờ ưu điểm có độ chính xác cao và tốc độ thực thi nhanh phù hợp với việc phát hiện té ngã trong thời gian thực.



Hình 3. Hình ảnh so sánh các phiên bản của YOLO

Kiến trúc của YOLO v8:

YOLOv8



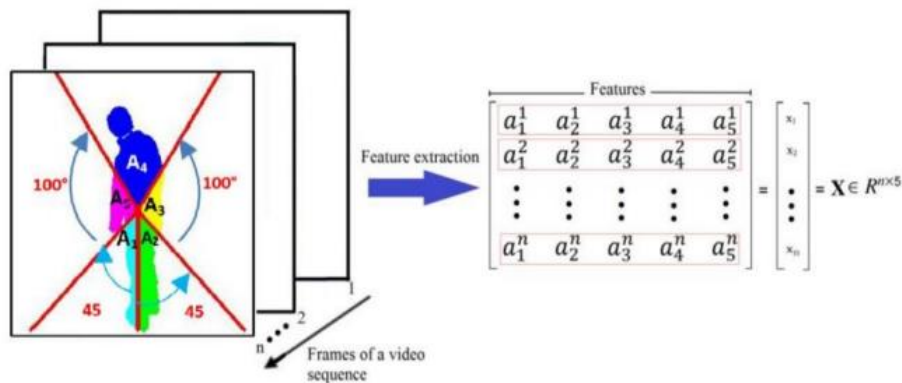
Hình 4. Kiến trúc Yolov8

Một số lý do chính mà ta nên cân nhắc sử dụng YOLOv8 trong dự án thị giác máy tính là:

- YOLO v8 có độ chính xác cao hơn so với các phiên bản YOLO trước đây.
- Nó hỗ trợ phát hiện đối tượng, phân đoạn cá thể và phân loại hình ảnh.
- Quá trình đào tạo YOLO v8 có thể sẽ nhanh hơn các mô hình phát hiện đối tượng hai giai đoạn khác.

❖ Trích xuất đặc trưng:

Sau khi khử nền và xác định bounding box chứa con người, chúng ta tiến hành trích xuất đặc trưng của frame. Trong việc trích xuất đặc trưng hình bóng cơ thể thì đã có nhiều đề xuất được sử dụng trước đây, tuy nhiên chúng phụ thuộc vào kích thước và vị trí của hình bóng so với camera. Do đó, chúng ta tiếp cận việc sử dụng năm khu vực cơ thể con người được xác định bằng cách xác định năm đường kẻ từ trọng tâm của hình bóng. Sau đó, từng khu vực được bình thường hóa bằng cách chia giá trị của nó cho toàn bộ diện tích khung hình.



Hình 5. Trích xuất đặc trưng ROI

Việc chia khu vực chứa người thành các phần nhỏ và trích xuất đặc trưng từng phần mang lại một số lợi ích sau:

- Các phần nhỏ có thể cung cấp thông tin chi tiết hơn về tư thế và vị trí của người, giúp cải thiện độ chính xác của quá trình nhận diện.
- Việc trích xuất đặc trưng từng phần giúp giảm thiểu sự phụ thuộc vào hình dáng và kích thước của người, bởi vì các đặc trưng như góc nghiêng, vị trí tâm, độ dài của chân, v.v. có thể được trích xuất độc lập từ mỗi phần.

Tuy nhiên, việc chia khu vực chứa người thành các phần nhỏ cũng có thể gây ra một số vấn đề, chẳng hạn như:

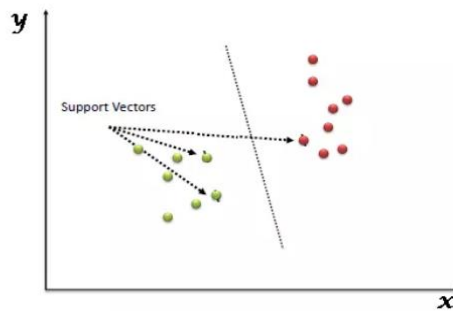
- Nếu việc chia phân vùng không chính xác hoặc vùng phân chứa thông tin quan trọng bị bỏ sót, quá trình nhận diện có thể trở nên không chính xác.
- Nếu các đặc trưng được trích xuất không đủ đa dạng và phù hợp, quá trình nhận diện có thể bị ảnh hưởng.

❖ Mô hình học máy SVM:

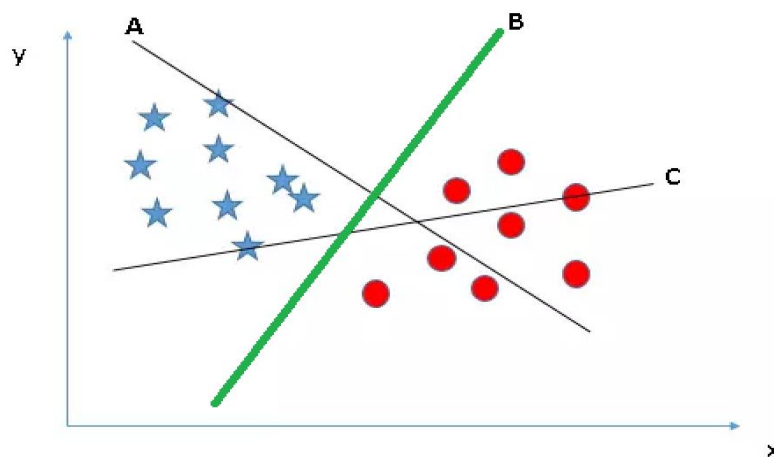
SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại (classifier)

SVM classifier hoạt động bằng cách tìm kiếm ranh giới quyết định tốt nhất gọi là siêu phẳng (hay hyper-plane) để phân tách dữ liệu thành các lớp khác nhau. Thuật toán SVM cố gắng tối đa hóa ranh giới giữa siêu phẳng và các điểm gần nhất của mỗi lớp, được gọi là các vector hỗ trợ. Khoảng cách từ siêu phẳng đến các vector hỗ trợ được gọi là độ rộng ranh giới (hay margin).

Siêu phẳng (hyperplane), nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt

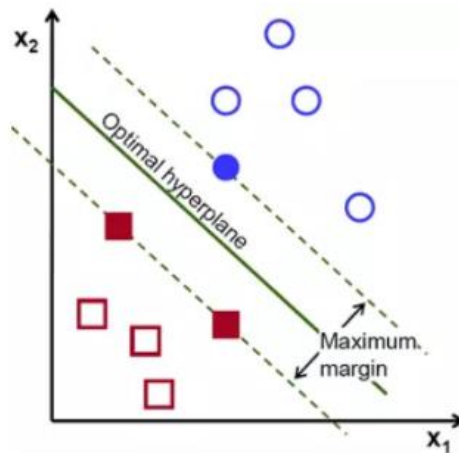


Hình 6. Minh họa về hyperplane



Hình 7. Ví dụ về chọn hyperplane

Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với các phân lớp



Hình 8. Minh họa về margin

SVM classifier nhận vào dữ liệu được biểu diễn bằng các vector đặc trưng, cùng với nhãn (label) cho mỗi điểm dữ liệu để biểu diễn hai lớp dữ liệu khác nhau

Sau đó tìm ra hyperplane để phân tách các điểm dữ liệu thuộc hai lớp khác nhau. Hyperplane này sẽ cách xa nhất với các vector hỗ trợ (support vectors) của hai lớp dữ liệu. Để tìm siêu mặt phẳng này, SVM sử dụng một hàm mất mát (loss function) và phương pháp tối ưu hóa để tìm giá trị tối thiểu của hàm mất mát

Khi có một điểm dữ liệu mới, SVM sử dụng siêu mặt phẳng đã tìm được để xác định xem điểm đó thuộc lớp nào. SVM tính toán giá trị của hàm phân loại $f(x) = wx + b$, với x là vector đặc trưng của điểm dữ liệu, w là vector trọng số và b là độ lệch (bias). Nếu giá trị $f(x) > 0$, điểm dữ liệu được dự đoán thuộc lớp 1, ngược lại nếu $f(x) < 0$ thì điểm dữ liệu được dự đoán thuộc lớp -1

❖ Mô hình LSTM (Long short term memory)

Mạng LSTM được cải tiến từ mạng thần kinh hồi quy (RNN–Recurrent Neural Network) nhằm khắc phục những nhược điểm về phụ thuộc xa (Long-term Dependency) của mạng RNN truyền thống

Về mặt lý thuyết, RNN có khả năng xử lý các phụ thuộc theo thời gian (temporal dependencies) bằng việc sử dụng bộ nhớ ngắn hạn và dựa trên việc xác định (luyện) các tham số một cách hiệu quả. Tuy nhiên, đáng tiếc trong thực tế RNN không thể giải quyết

các phụ thuộc theo thời gian khi chuỗi số liệu có các phụ thuộc xa (long-term dependencies).

Kiến trúc LSTM giúp giải quyết vấn đề xác định quan hệ phụ thuộc dài hạn giữa các thông tin trong dữ liệu chuỗi thời gian. Một vài ứng dụng của phổ biến của LSTM:

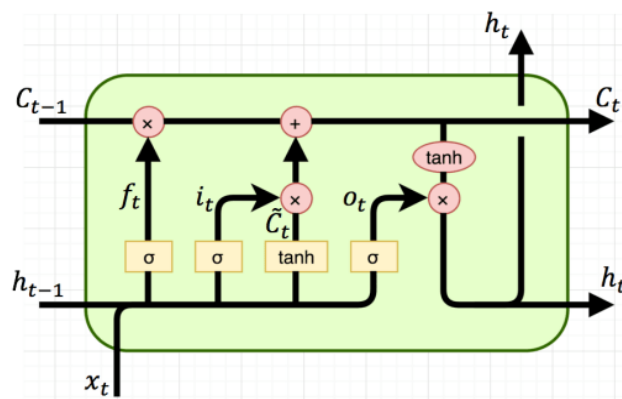
- Dự đoán chuỗi thời gian: dự đoán giá cổ phiếu, giá dầu vàng, hay lượng truy cập trang web trong tương lai,...
- Xử lý ngôn ngữ tự nhiên (NLP): LSTM được sử dụng trong các ứng dụng NLP như dịch máy, phân loại văn bản, sinh văn bản và nhận dạng giọng nói.
- Hệ thống gợi ý: LSTM được sử dụng để xây dựng các hệ thống gợi ý sản phẩm, bài viết và nội dung dựa trên hành vi và sở thích của người dùng trong quá khứ.

Kiến trúc của LSTM:

Ở state thứ t của mô hình LSTM:

Input: c_{t-1} , h_{t-1} , x_t . Trong đó x_t là input ở state thứ t của model. c_{t-1} , h_{t-1} là output của layer trước. C là điểm mới của LSTM.

Output: c_t , h_t , ta gọi c là cell state, h là hidden state.



Hình 9. Mô hình LSTM

f_t , i_t , o_t tương ứng với forget gate, input gate và output gate.

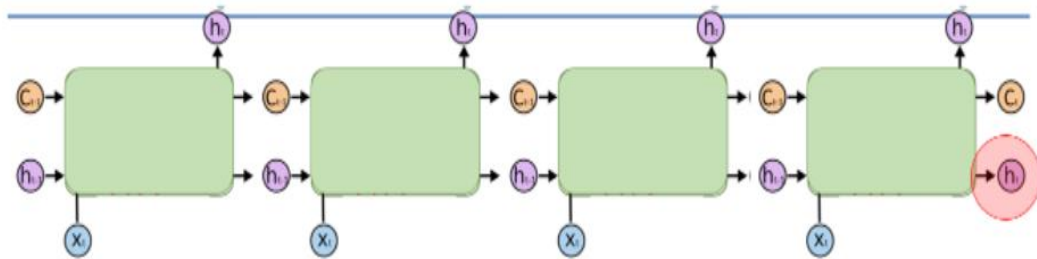
- Forget gate: $f_t = \sigma(U_f * x_t + W_f * h_{t-1} + b_f)$
- Input gate: $i_t = \sigma(U_i * x_t + W_i * h_{t-1} + b_i)$
- Output gate: $o_t = \sigma(U_o * x_t + W_o * h_{t-1} + b_o)$

$$c_t \sim \tanh(U_c * x_t + W_c * h_{t-1} + b_c)$$

$c_t = f_t * c_{t-1} + i_t * c_t$, forget gate quyết định xem cần lấy bao nhiêu từ cell state trước và input gate sẽ quyết định lấy bao nhiêu từ input của state và hidden layer của layer trước.

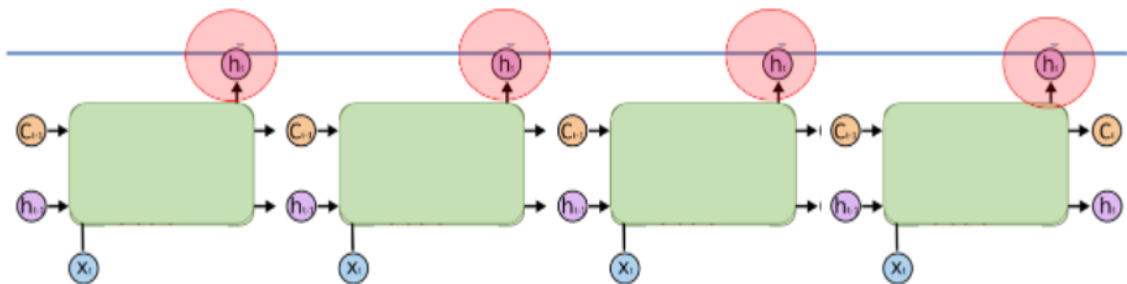
$h_t = o_t * \tanh(c_t)$, output gate quyết định xem cần lấy bao nhiêu từ cell state để trở thành output của hidden state. Ngoài ra h_t cũng được dùng để tính ra output y_t cho state t .

Đối với lớp LSTM mặc định (`return_sequence = False`): Kích thước đầu ra là mảng số thực 2 chiều. Chiều đầu tiên là số lượng mẫu được cung cấp cho lớp LSTM. Chiều thứ 2 là chiều không gian đầu ra được xác định bởi tham số số units



Hình 10. Output của lớp LSTM mặc định

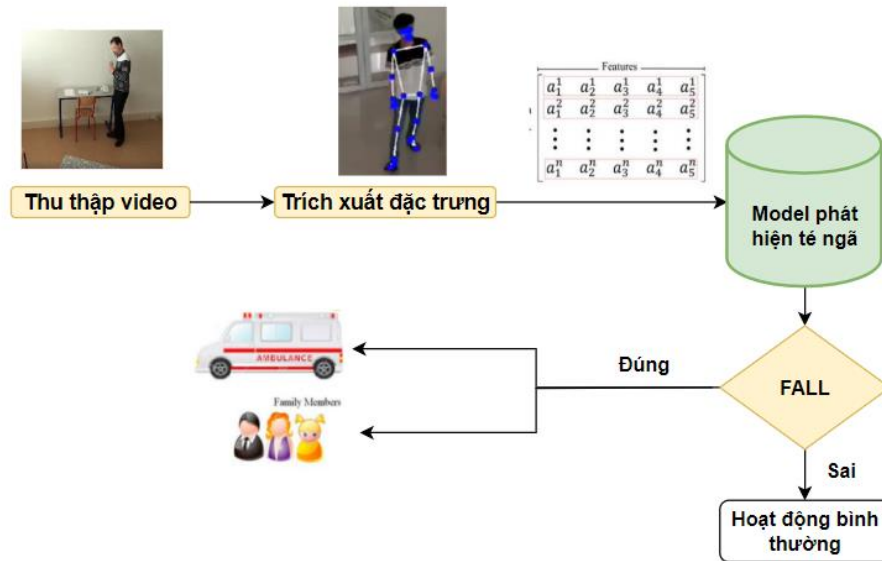
Đối với lớp LSTM có `return_sequence = True`: Kích thước đầu ra là mảng số thực 3 chiều. Chiều đầu tiên là số lượng mẫu được cung cấp cho lớp LSTM. Chiều thứ 2 là số lượng frame trong một mẫu và chiều thứ ba là chiều không gian đầu ra được xác định bởi tham số số units



Hình 11. Output của lớp LSTM có `return_sequence`

2.2.2. Phát hiện té ngã dựa trên hướng tiếp cận Bottom-up

❖ Mô hình thuật toán:



Hình 12. Sơ đồ thuật toán phương pháp Bottom-up

Khác với phương pháp Top-down thì phương pháp này không cần đi tìm vùng chứa cơ thể người mà ta đi xác định các điểm chính không có danh tính như các vị trí khung xương,... Sau đó dùng đặc trưng này để đi tiến hành huấn luyện các model phát hiện té ngã.

Ưu điểm của phương pháp này là ít phức tạp hơn phương pháp Top-down, dẫn đến chi phí tính toán về mặt thời gian nhanh hơn. Tuy nhiên, với việc phát hiện các điểm không có danh tính không chính xác có thể làm giảm độ chính xác của mô hình.

❖ Trích xuất đặc trưng khung xương

MediaPipe Pose là một thư viện phần mềm miễn phí được cung cấp bởi Google, cung cấp giải pháp nhận dạng vị trí các điểm chính trên cơ thể con người. Nó được sử dụng để xác định vị trí và hướng của các điểm chính trên cơ thể, bao gồm đầu, vai, cổ tay, khuỷu tay, bàn tay, hông, đầu gối và chân,...

Quá trình nhận diện của Pose Classification dựa trên solution về BlazePose, nó sẽ bóc tách các điểm trên cơ thể trong không gian 3 chiều (x, y, z) từ một RGB video

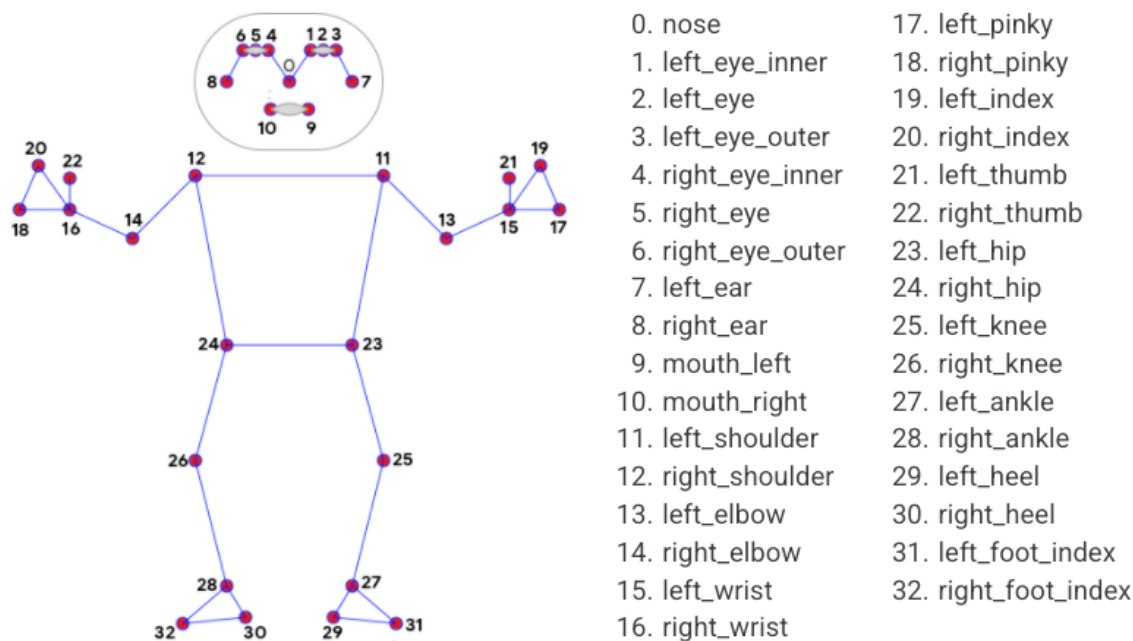
Kiến trúc của BlazePose gồm 2 thành phần chính:

- Pose Detector: Để detect ra vùng chứa person trên bức ảnh
- Pose Tracker: Để trích xuất ra các keypoints trên vùng chứa vị trí của person đã được crop ra từ bức ảnh đồng thời dự đoán vị trí của person trong frame tiếp theo

Các output của MediaPipe

- X, y, z: Tọa độ 3D trong thế giới thực tính bằng mét với điểm mốc nằm ở hông. Nằm trong khoảng giá trị từ 0 đến 1
- Visibility: khả năng các landmark có được nhìn thấy (không bị che khuất) trong hình ảnh

Để ứng dụng vào đề tài này, nhóm sử dụng MediaPipe Pose để nhận diện được khung xương của cơ thể. Các vị trí của khung dựa vào vị trí của cơ thể như sau:



Hình 13. Các vị trí trên khung xương

2.2.3. Firebase

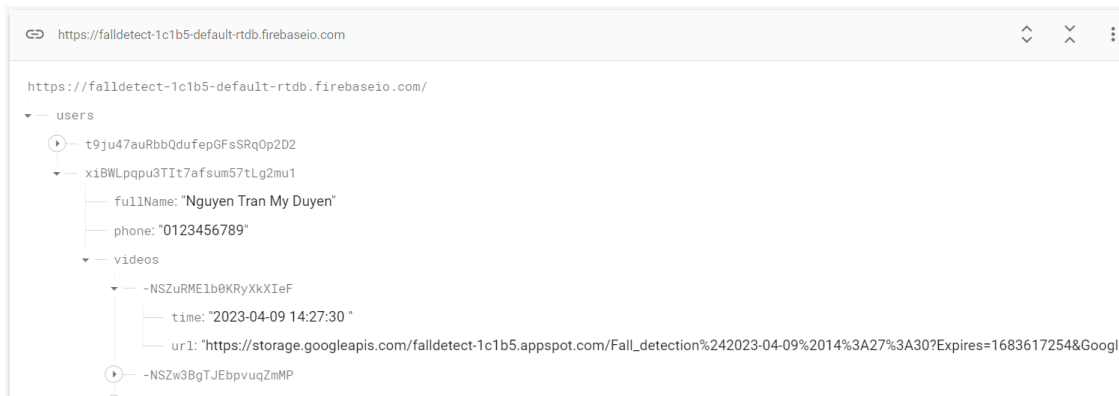
❖ Realtime Database

Firebase Realtime Database là một cơ sở dữ liệu NoSQL thời gian thực được lưu trữ trên đám mây, cho phép lưu trữ và đồng bộ dữ liệu. Cơ sở dữ liệu sử dụng định dạng cây JSON và đồng bộ hóa dữ liệu thời gian thực cho mọi kết nối.

Firebase Realtime Database có ba chức năng chính:

- Thời gian thực: Firebase Realtime Database đồng bộ dữ liệu ngay lập tức khi có sự thay đổi, cho phép các thiết bị kết nối nhận được thông tin mới chỉ trong vài mili giây.

- Khả năng hoạt động ngoại tuyến: Khi người dùng ngoại tuyến, dữ liệu sẽ được lưu trữ trên bộ nhớ cache của thiết bị và được tự động đồng bộ hóa khi thiết bị kết nối trực tuyến.
- Truy cập từ thiết bị khách: Firebase Realtime Database có thể truy cập từ một thiết bị di động hoặc trình duyệt web và không cần phải có một máy chủ ứng dụng riêng. Bảo mật và xác thực dữ liệu được thực hiện thông qua các quy tắc bảo mật Rule của Firebase Realtime Database, được thực thi khi dữ liệu được đọc hoặc ghi.



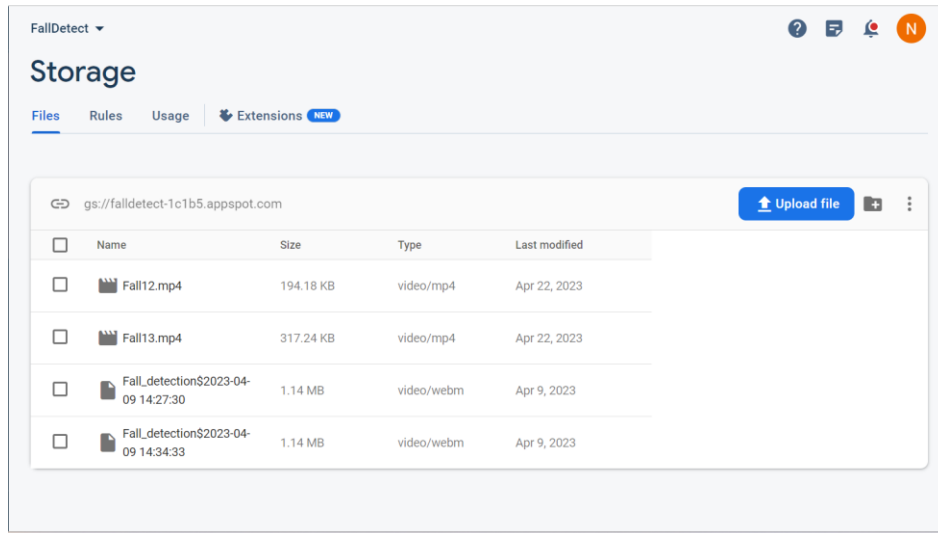
Hình 14. Hình ảnh lưu dữ liệu trên realtime database

❖ Storage

Firebase Storage là dịch vụ được xây dựng cho mục đích lưu trữ và quản lý các nội dung mà người dùng ứng dụng tạo ra như ảnh, videos, hay dữ liệu dạng file.

Firebase Storage cung cấp các API cho việc upload và download các file từ app của bạn một cách bảo mật và bạn không cần quan tâm đến chất lượng đường truyền mạng. Firebase Storage được xây dựng trên nền tảng Google Cloud Platform nên có nhiều điểm mạnh như:

- Robust: Firebase Storage thực hiện việc upload và download không phụ thuộc vào chất lượng đường truyền mạng hơn nữa các quá trình đó có thể bắt đầu lại khi bị tạm dừng giúp tiết kiệm thời gian và băng thông.
- Secure: Được tích hợp Firebase Authentication cho việc bảo mật nên dễ dàng quản lý quyền truy cập vào các files.
- Scalable: Firebase Storage được xây dựng trên nền tảng Google Cloud Platform nên khả năng mở rộng có thể lên đến hàng Petabyte dữ liệu.

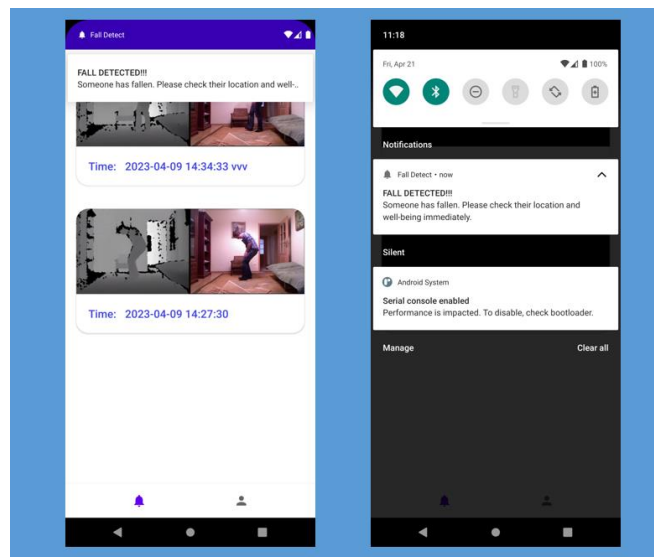


Hình 15. Hình ảnh lưu video trên firebase storage

❖ Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) là một dịch vụ gửi thông báo, tin nhắn đa nền tảng được cung cấp bởi Google, cho phép bạn gửi tin nhắn, thông báo một cách đáng tin cậy và hoàn toàn miễn phí tới các thiết bị đã được đăng ký với FCM.

Nguyên tắc hoạt động: Các thiết bị client sẽ đăng ký device token lên cho FCM. Các thông báo, tin nhắn được soạn và gửi từ một website, từ Notifications composer của firebase cung cấp, FCM sẽ nhận những thông báo này và xử lý gửi về các thiết bị đã đăng ký với FCM từ trước. Khi các thiết bị có kết nối mạng thì thông báo sẽ được gửi về ứng dụng thành công.



Hình 16. Hình ảnh demo gửi thông báo sử dụng FCM

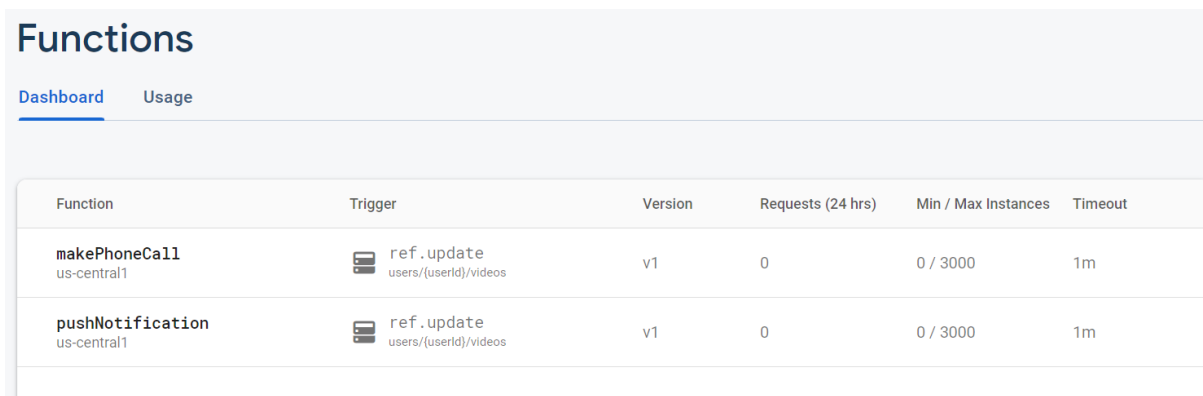
❖ Functions

Firebase Functions là một dịch vụ tích hợp trong Firebase, cho phép viết và triển khai các hàm máy chủ (serverless functions) một cách dễ dàng và nhanh chóng. Với Firebase Functions, có thể viết các hàm JavaScript hoặc TypeScript và triển khai chúng trên nền tảng Firebase mà không cần quản lý một máy chủ riêng.



Firebase Functions cung cấp các tính năng như:

- Được tích hợp sẵn với các dịch vụ khác của Firebase, cho phép các hàm máy chủ có thể kết nối đến cơ sở dữ liệu Firebase, xử lý thông báo, hoặc gửi email thông qua Firebase Cloud Messaging, Firebase Cloud Firestore và Firebase Authentication.
- Triển khai nhanh chóng với tính năng chỉnh sửa và triển khai trực tiếp trên giao diện người dùng của Firebase Console.
- Hỗ trợ các ngôn ngữ khác nhau như JavaScript, TypeScript, Python, Go, Java và .NET.

Các ứng dụng của Firebase Functions là rất đa dạng, bao gồm xử lý các sự kiện của người dùng, tạo và gửi email tự động, xử lý các yêu cầu API, cập nhật dữ liệu, và xử lý các sự kiện của hệ thống. Firebase Functions giúp cho việc phát triển ứng dụng trở nên dễ dàng hơn bao giờ hết.



The screenshot shows the 'Functions' dashboard in the Firebase console. It has two tabs: 'Dashboard' (selected) and 'Usage'. Below the tabs is a table listing functions. The table has columns: Function, Trigger, Version, Requests (24 hrs), Min / Max Instances, and Timeout. Two functions are listed: 'makePhoneCall' and 'pushNotification', both triggered by 'ref.update' on the 'users/(userId)/videos' path, version 'v1', with 0 requests in the last 24 hours, 0 / 3000 instances, and a 1m timeout.

Function	Trigger	Version	Requests (24 hrs)	Min / Max Instances	Timeout
makePhoneCall us-central1	 ref.update users/(userId)/videos	v1	0	0 / 3000	1m
pushNotification us-central1	 ref.update users/(userId)/videos	v1	0	0 / 3000	1m

Hình 17. Hình ảnh các hàm trên Firebase Functions

2.2.4. ClickSend

ClickSend là một nền tảng dịch vụ cung cấp cho khách hàng của mình các giải pháp gửi tin nhắn SMS, email, fax, thư định kỳ và thư tín, cũng như các tính năng liên quan đến việc quản lý danh sách khách hàng và chuyển đổi tỷ lệ chốt. ClickSend được sử dụng bởi các doanh nghiệp và tổ chức để gửi thông báo, cập nhật, quảng cáo và các tin nhắn khác đến khách hàng hoặc nhân viên của họ.

Một vài ưu điểm của ClickSend là:

- Gửi tin nhắn, email và fax hàng loạt: ClickSend cho phép người dùng gửi hàng ngàn tin nhắn, email hoặc fax cùng một lúc, tiết kiệm thời gian và công sức.
- Tính năng quản lý thư tín và thư định kỳ: ClickSend cung cấp tính năng quản lý thư tín và thư định kỳ, cho phép người dùng cài đặt các thông báo tự động và nhắc nhở định kỳ cho khách hàng.
- Giao diện trực tuyến và API: ClickSend cung cấp một giao diện trực tuyến và API cho phép người dùng gửi tin nhắn, email hoặc fax từ bất kỳ đâu và bất kỳ ứng dụng nào.
- Tuân thủ quy định về quyền riêng tư và an ninh thông tin: ClickSend cam kết tuân thủ các quy định về quyền riêng tư và an ninh thông tin, đảm bảo an toàn và bảo mật thông tin khách hàng.

2.2.5. Giải pháp về ứng dụng di động

❖ Các chức năng của hệ thống

- Đăng nhập, đăng xuất
- Hiện thị video té ngã: cho phép người dùng xem các video té ngã. Các video này được lưu trữ trên Firebase Storage và được hiển thị cho người dùng trong app.
- Nhận thông báo khi phát hiện té ngã: sử dụng Firebase Cloud Messaging để gửi thông báo đến người dùng khi phát hiện sự cố té ngã. Firebase Functions được sử dụng để push thông báo khi có thay đổi trong dữ liệu. Như vậy, người dùng có thể nhanh chóng được cảnh báo và phản ứng khi có sự cố xảy ra.
- Nhận cuộc gọi khi phát hiện té ngã: sử dụng ClickSend để gọi đến người dùng khi phát hiện sự cố té ngã để người dùng có thể phản ứng kịp thời.

❖ Các công nghệ sử dụng trong ứng dụng

- Ngôn ngữ lập trình: Java
- IDE: Android Studio
- Database: Lưu trữ trên Firebase
- Server: Cloud Functions for Firebase
- Điện thoại: ClickSend.

3. Kết quả

3.1. Dataset

Dataset thu thập từ các nguồn sau:

- Dataset CAUCAFall [11]
- Dataset NTU RGB+D [12]
- Fall Dataset ImVia [14]
- Nhóm tiến hành tự quay

Mô tả dữ liệu trong tập dataset:

Bảng 3. Mô tả dữ liệu dataset

Classes	Số lượng video	Tổng thời gian	Format	Resize	Fps
Fall	195	400s	mp4	640 x 480	30
No Fall	235	815s	mp4	640 x 480	30

Dữ liệu được thu thập đã được nhóm tiến hành xử lý. Những video ngã được nhóm tiến hành cắt để trong video đó chỉ chứa hành động ngã để tránh đánh nhãn fall cho những phân đoạn có các hành động khác như đi, ngồi,... Ngoài ra, dữ liệu đều được chuyển đổi có fps bằng 30, với kích thước khung hình là 640 x 480.

Dữ liệu được nhóm tiến hành tự quay được thực hiện với vị trí camera đặt ở góc trên tại căn phòng khu Smart building Đại học Bách Khoa – Đại học Đà Nẵng vào ngày 09/04/2023.

fall	02/06/2023 17:32	File folder
fall_backwards	02/06/2023 17:32	File folder
fall_forward	02/06/2023 17:32	File folder
fall_left	02/06/2023 17:32	File folder
fall_right	02/06/2023 17:32	File folder
fall_sitting	02/06/2023 17:32	File folder
hop	02/06/2023 17:32	File folder
jump	02/06/2023 17:32	File folder
kneel	02/06/2023 17:32	File folder
pick_up_object	02/06/2023 17:32	File folder
sit_down	02/06/2023 17:32	File folder
stand_up	02/06/2023 17:32	File folder
walk	02/06/2023 17:32	File folder

Hình 18. Tổ chức thư mục dataset

Phân chia dữ liệu: Train: 70%, Val: 10%, Test: 20%.

3.2. Kết quả của phương pháp Top-down

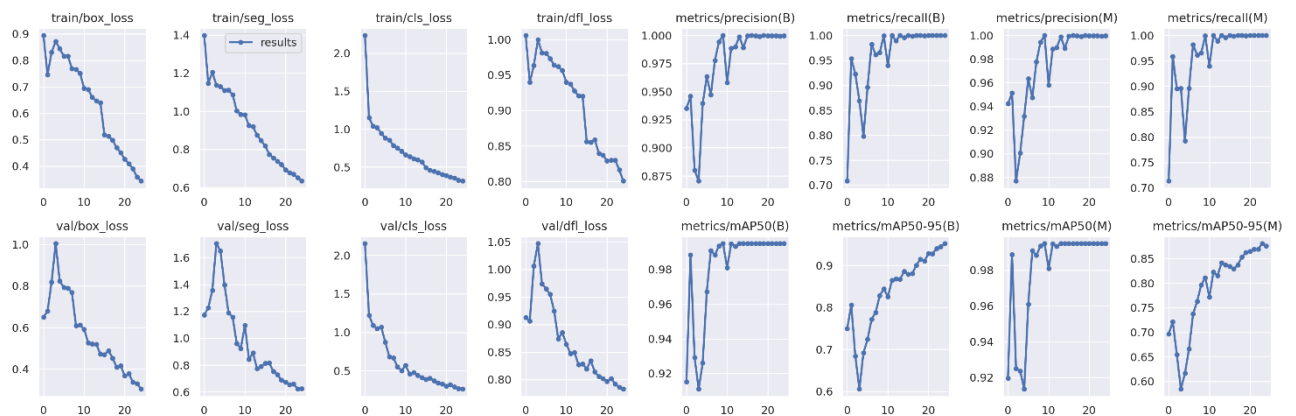
❖ Kết quả custom train model YOLO v8 segment:

Với model yolov8n-seg mặc định để khử nền thì nó hoạt động tốt ở các hành động đi, ngồi,... Tuy nhiên, khi nhóm thử nghiệm model yolov8n-seg mặc định với video thì có nhiều frame không phát hiện được vùng chứa người, điều này dẫn đến sự mất mát thông tin, cũng như có thể làm quá trình phát hiện té ngã tăng khả năng nhầm lẫn.

Để tăng cường khả năng phát hiện và khử nền của mô hình Yolov8n-seg mặc định, nhóm đã quyết định tiến hành huấn luyện lại mô hình trên một dataset tùy chỉnh. Trước tiên, nhóm đã xây dựng dataset bằng cách thu thập các ảnh chứa các hành động khác nhau. Quá trình tạo dataset được thực hiện thông qua trang web Roboflow, đảm bảo tính đa dạng và đại diện của dữ liệu huấn luyện.

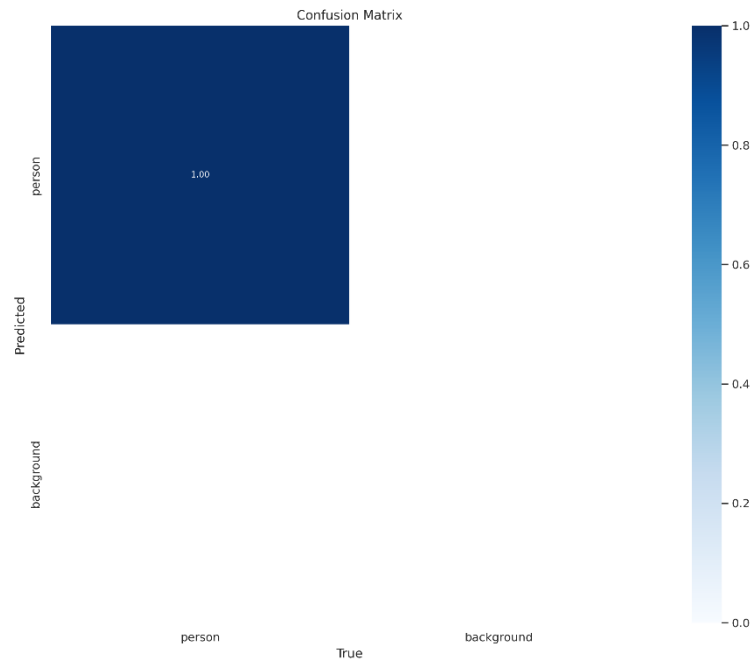
Dataset dùng để khử nền bao gồm tổng cộng 1273 ảnh. Nhóm đã chia dataset thành ba phần: tập train (874 ảnh), tập val (183 ảnh) và test (116 ảnh).

Các kết quả thu được khi tiến hành train model yolo8n-seg:



Hình 19. Các chỉ số đánh giá khi train yolov8-seg

Ma trận nhầm lẫn:



Hình 20. Ma trận nhầm lẫn khi train yolov8-seg

Kết quả test ở tập valid:



Hình 21. Kết quả segment với yolov8-seg

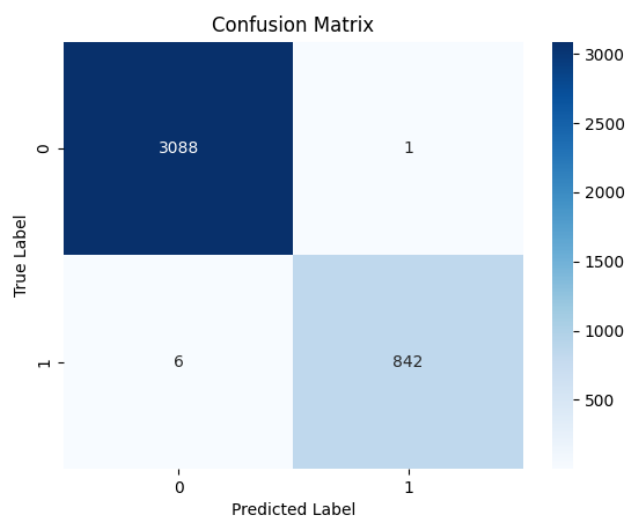
❖ Kết quả train model bằng LSTM

Kết quả được thử nghiệm với số frame trong 1 lần quan sát lần lượt là: 30, 35, 40. Với số frame 1 lần quan sát, nhóm tiến hành chia tập dữ liệu và huấn luyện 3 lần và lựa chọn kết quả tốt nhất. Với mỗi lần train, dữ liệu được chia thành train, test, val và lưu thành file npy. Sau đây là bảng kết quả mà nhóm thu được:

Bảng 4. Bảng kết quả phương pháp Top-down

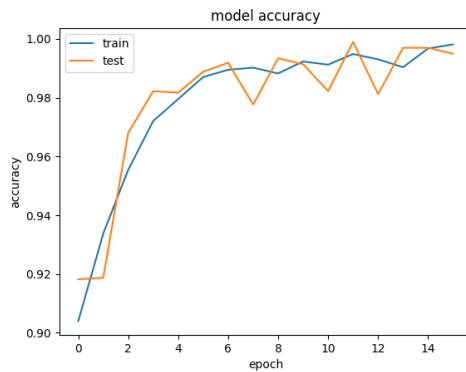
Number frames of observation	Version	Lost_train	Accuracy_train	Lost_val	Accuracy_val	Lost_test	Accuracy_test
30 frames	1	0.025	99.27%	0.023	99.54%	0.023	99.21%
	2	0.026	99.18%	0.026	99.00%	0.024	99.12%
	3	0.033	98.94%	0.011	99.83%	0.015	99.58%
35 frames	1	0.019	99.37%	0.009	99.69%	0.009	99.79%
	2	0.013	99.59%	0.013	99.72%	0.008	99.70%
	3	0.015	99.55%	0.033	99.26%	0.021	99.42%
40 frames	1	0.006	99.81%	0.018	99.49%	0.009	99.82%
	2	0.011	99.66%	0.012	99.70%	0.010	99.59%
	3	0.021	99.40%	0.009	99.74%	0.014	99.44%

Độ chính xác cao nhất thu được là với số frame trong 1 lần quan sát bằng 40 ở phiên bản huấn luyện 1 với độ chính xác tập test là 99.82%. Ma trận nhầm lẫn có được:

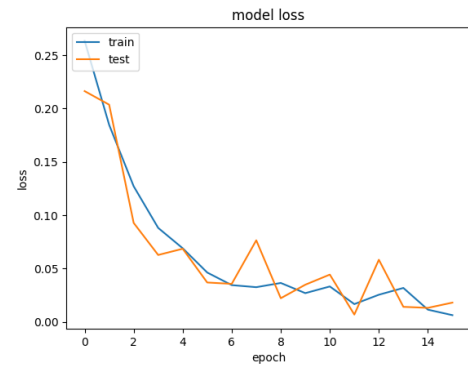


Hình 22. Ma trận nhầm lẫn phương pháp Top-down với LSTM

Kết quả đồ thị:



Hình 23. Đồ thị accuracy phương pháp Top-down



Hình 24. Đồ thị loss phương pháp Top-down

Kết quả test từ video quay sẵn:



Hình 25. Kết quả nhận diện phương pháp Top-down với LSTM

❖ Kết quả train model bằng SVM

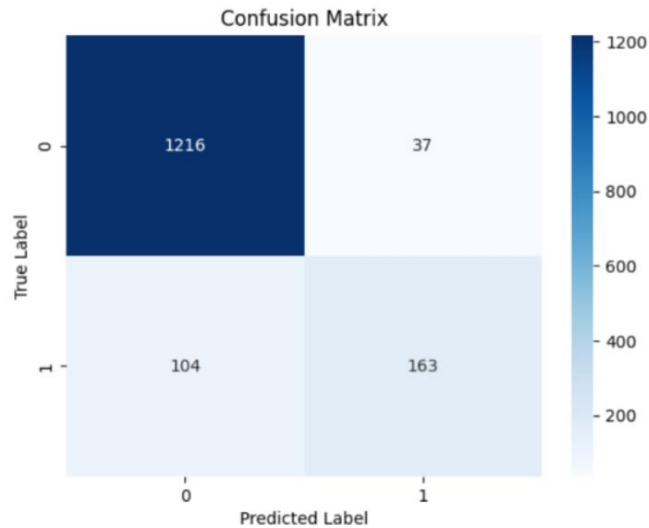
Độ chính xác

```
accuracy = 0.9072368421052631
```

	precision	recall	f1-score	support
0	0.92	0.97	0.95	1253
1	0.81	0.61	0.70	267
accuracy			0.91	1520
macro avg	0.87	0.79	0.82	1520
weighted avg	0.90	0.91	0.90	1520

Hình 26. Độ chính xác phương pháp Top-down với SVM

Ma trận nhầm lẫn:



Hình 27. Ma trận nhầm lẫn phương pháp Top-down với SVM

Kết quả test từ video quay sẵn:



Hình 28. Kết quả nhận diện phương pháp Top-down với SVM

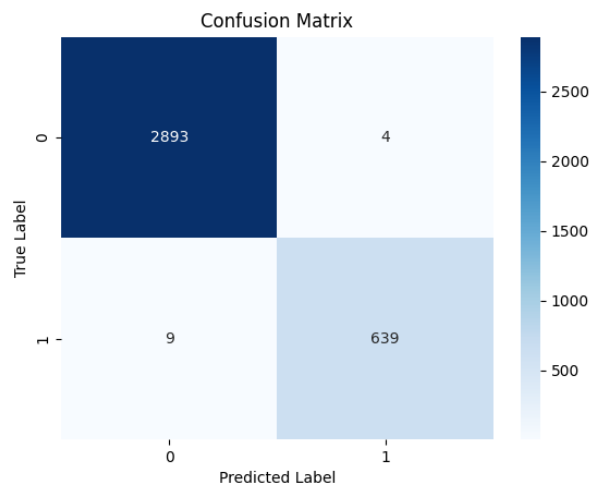
3.3. Kết quả của phương pháp Bottom-up

Kết quả các model LSTM:

Bảng 5. Bảng kết quả phương pháp Bottom-up

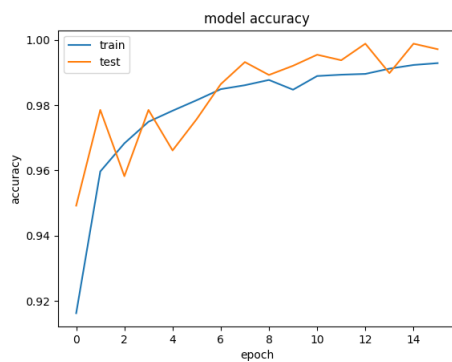
Number frames of observation	Version	Lost_train	Accuracy_train	Lost_val	Accuracy_val	Lost_test	Accuracy_test
30 frames	1	0.029	99.07%	0.021	99.34%	0.023	99.24%
	2	0.025	99.17%	0.028	99.19%	0.023	99.26%
	3	0.024	99.18%	0.032	99.19%	0.026	99.16%
35 frames	1	0.024	99.29%	0.008	99.72%	0.009	99.63%
	2	0.025	99.21%	0.028	98.93%	0.042	98.76%
	3	0.028	99.17%	0.027	99.15%	0.025	99.04%
40 frames	1	0.024	99.24%	0.011	99.62%	0.008	99.59%
	2	0.017	99.43%	0.033	98.62%	0.032	98.87%
	3	0.017	99.40%	0.021	99.37%	0.031	99.34%

Độ chính xác cao nhất thu được là với số frame trong 1 lần quan sát bằng 35 ở phiên bản huấn luyện 1 với độ chính xác tập test là 99.63%. Ma trận nhầm lẫn có được:

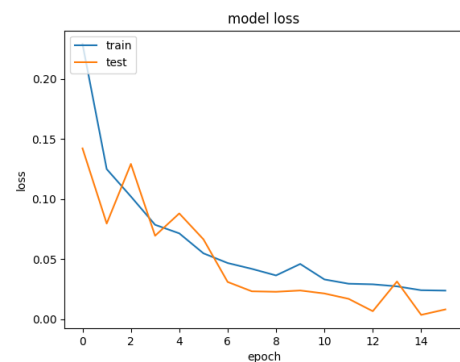


Hình 29. Ma trận nhầm lẫn phương pháp Bottom-up

Kết quả đồ thị:



Hình 30. Đồ thị accuracy phương pháp Bottom-up



Hình 31. Đồ thị loss phương pháp Bottom-up

Kết quả test từ video quay sẵn:



Hình 32. Kết quả nhận diện phương pháp Bottom-up

3.4. So sánh kết quả giữa hai phương pháp

Số frame trong 1 quan sát là 30 frames và được train với model LSTM

Kết quả được thử nghiệm ở bộ vi xử lý: Intel(R) Core(TM) i7-10750H CPU
@ 2.60GHz 2.59 GHz

Bảng 6. Bảng kết quả so sánh 2 phương pháp

Tiêu chí	Top-down Framework	Bottom-up Framework
Thời gian trung bình trích xuất đặc trưng của 1 frame	0.11959s	0.02451s
Thời gian trung bình dự đoán 1 quan sát	4.20541s	1.10993s
Số frame không detect được	0	2
Độ chính xác trên tập test	99.58%	99.26%

Từ bảng kết quả trên, ta nhận thấy rằng với hướng tiếp cận Bottom-up thì ta thu được kết quả tốt hơn về mặt hiệu suất, tuy độ chính xác có thấp hơn một ít nhưng nhìn chung với các phiên bản huấn luyện mà nhóm thu được thì độ chính xác giữa phương pháp Top-down và Bottom-up tương đối gần bằng nhau.

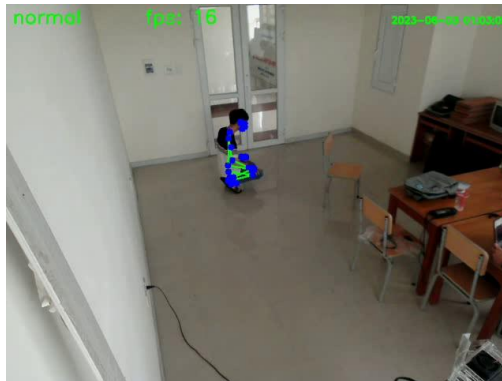
Với hướng tiếp cận Top-down thì việc khử nền là yếu tố quan trọng nhất cho việc trích xuất đặc trưng ROI. Tuy nhiên việc khử nền thường khó khăn bởi các môi trường nhiễu, có vật cản,... Việc sử dụng mô hình khử nền yolov8 segment cho kết quả tương đối tốt nhưng lại tốn thời gian thực thi lớn.

Việc trích xuất đặc trưng của 1 frame mang ý nghĩa đặc trưng quan trọng trong việc phát hiện té ngã, thư viện Mediapipe pose đã tìm được các đặc trưng khung xương một cách chính xác, cũng như nhanh chóng về mặt hiệu suất. Điều này giúp việc phát hiện té ngã được phát hiện trong thời gian thực.

Vì kết quả độ chính xác không chênh lệch nhiều cùng với phần cứng có tốc độ xử lý còn hạn chế, do đó nhóm quyết định chọn phương pháp Bottom-up để đảm bảo yếu tố té ngã được phát hiện trong thời gian thực.

3.5. Kết quả deploy trên Jetson Nano Developer Kit B01

Kết quả nhận diện với hành động quỳ gối:



Hình 33. Kết quả nhận diện hành động quỳ gối

Kết quả nhận diện với hành động đi:



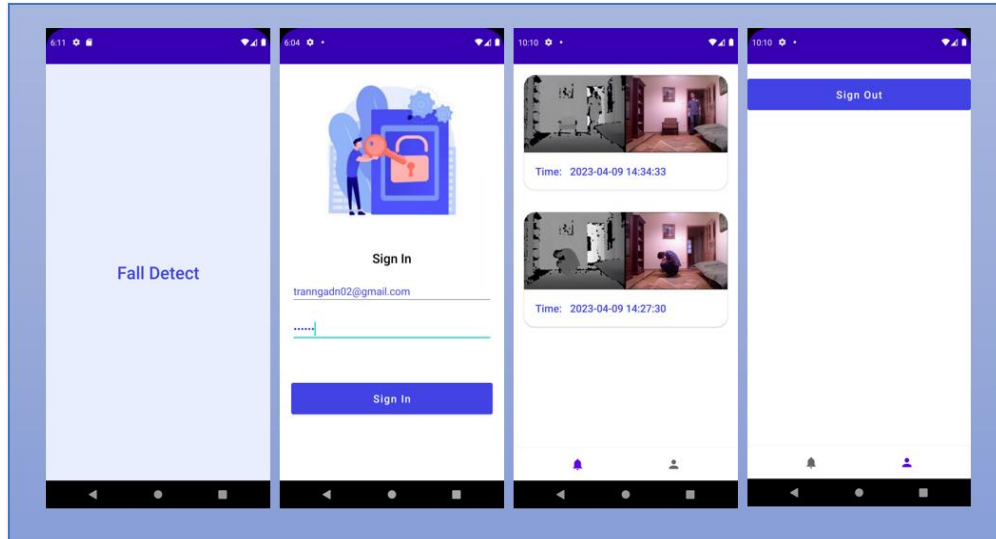
Hình 34. Kết quả nhận diện hành động đi

Kết quả nhận diện với hành động ngã:



Hình 35. Kết quả nhận diện hành động ngã

3.6. Kết quả xây dựng App Mobile



Hình 36. Giao diện app mobile

4. Kết luận

4.1. Kết luận

- Đồ án đã đạt được kết quả đặt ra là phát hiện té ngã dựa trên luồng video.
- Đã xây dựng được model và nhận diện đã hành động ngã.
- Đã thiết lập được phần cứng có thể thu hình ảnh thông qua camera và xử lý phát hiện té ngã trực tiếp trên phần cứng đó.
- Xây dựng được thiết bị mobile phát ra thông báo khi phát hiện hành động ngã.
- Dataset với các hành động còn hạn chế so với các hoạt động diễn ra hằng ngày, dẫn đến nhiều tư thế, hành động chưa được phát hiện một cách chính xác.

4.2 Hướng phát triển

- Xây dựng tập dataset tổng quát cho nhiều hành động với các góc quay và điều kiện môi trường khác nhau.
- Cải thiện tốc độ xử lý, độ chính xác của mô hình.

5. Danh mục tài liệu tham khảo

- [1] Charles Andrew Q. Bugarin; Juan Miguel M. Lopez; Scud Gabriel M. Pineda; Ma. Franzeska C. Sambrano, Pocholo James M. Loresco, “Machine Vision-Based Fall Detection System using MediaPipe Pose with IoT Monitoring and Alarm”, 2022.
- [2] Jesús Gutiérrez, Víctor Rodríguez and Sergio Martin, “Comprehensive Review of Vision-Based Fall Detection Systems”, 2021.
- [3] Nguyễn Việt Anh, “Phát hiện ngã sử dụng đặc trưng chuyển động và hình dạng cơ thể dựa trên camera đơn”, 2016.
- [4] Qi Dang, Jianqin Yin, Bin Wang, and Wenqing Zheng, “2D Human Pose Estimation: A Survey”, 2019.
- [5] Abderrazak Iazzi, Mohammed Rziza and Rachid Oulad Haj Thami, “Fall Detection System-Based Posture-Recognition for Indoor Environments”, 2021.
- [6] Harrou, Fouzi; Zerrouki, Nabil; Sun, Ying; Houacine, Amrane, “Vision-based fall detection system for improving safety of elderly people”, 2017.
- [7] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, Gang Wang, "NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [8] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, Alex C. Kot, "NTU RGB+D 120: A Large-Scale Benchmark for 3D Human Activity Understanding", IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2019.
- [9] [Learn About Long Short-Term Memory \(LSTM\) Algorithms](#), truy cập 02/02/2023.
- [10] [Recurrent neural network](#), truy cập 02/02/2023.
- [11] <https://data.mendeley.com/datasets/7w7fccy7ky/4>, truy cập 20/02/2023.
- [12] <https://imvia.u-bourgogne.fr/en/database/fall-detection-dataset-2.html>, truy cập 15/04/2023.
- [13] https://github.com/YifeiYang210/Fall_Detection_dataset, truy cập 05/05/2023
- [14] <https://rose1.ntu.edu.sg>, truy cập 09/05/2023.
- [15] <https://www.v7labs.com/blog/mean-average-precision>, truy cập 28/5/2023.

[16] <https://tytphuongtruongtho.medinet.gov.vn/giao-duc-suc-khoe/gia-tang-so-ca-dot-quy-chuyen-gia-canh-bao-neu-thay-co-mot-trong-nhung-dau-hieu-cmobile8196-74055.aspx>, truy cập ngày 29/05/2023.