BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH

ĐỒ ÁN TỐT NGHIỆP

ỨNG DỤNG HỌC SÂU TRONG HỆ THỐNG PHÁT HIỆN VÀ CẢNH BÁO TÉ NGÃ

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành: **CÔNG NGHỆ PHẦN MỀM**

Giảng viên hướng dẫn : ThS. Nguyễn Đình Ánh

Sinh viên thực hiện : Phạm Văn Tiến

MSSV: 2180604970 Lớp: 21DTHA5

TP. Hồ Chí Minh, 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH

ĐỒ ÁN TỐT NGHIỆP

ỨNG DỤNG HỌC SÂU TRONG HỆ THỐNG PHÁT HIỆN VÀ CẢNH BÁO TÉ NGÃ

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành: **CÔNG NGHỆ PHẦN MỀM**

Giảng viên hướng dẫn : ThS. Nguyễn Đình Ánh

Sinh viên thực hiện : Phạm Văn Tiến

MSSV: 2180604970 Lớp: 21DTHA5

Giảng viên hướng dẫn	Giảng viên phản biện	Chủ tịch hội đồng

TP. Hồ Chí Minh, 2025

LỜI CAM ĐOAN

Em xin cam đoan rằng, đề tài nghiên cứu "ỨNG DỤNG HỌC SÂU TRONG HỆ THỐNG PHÁT HIỆN VÀ CẢNH BÁO TÉ NGÃ" là công trình nghiên cứu của bản thân dưới sự hướng dẫn của ThS. Nguyễn Đình Ánh và không có sự sao chép hay tham khảo từ bất kỳ nguồn tài liệu nào khác, trừ khi được trích dẫn rõ ràng.

Em cũng cam đoan rằng tất cả các thông tin và dữ liệu được sử dụng trong đồ án này là chính xác và được thu thập từ các nguồn đáng tin cậy. Em đã kiểm tra kỹ lưỡng và xác nhận tính chính xác của các thông tin và dữ liệu này trước khi sử dụng trong đồ án của mình. Em đã hoàn thành tất cả các nhiệm vụ được yêu cầu trong đồ án và hoàn thành đúng tiến độ cũng nhưng tận dụng tối đa thời gian và nỗ lực của mình để hoàn thành đồ án này.

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn và lòng biết ơn sâu sắc đến thầy Nguyễn Đình Ánh, người đã giúp đỡ em trong quá trình chọn đề tài, định hình hướng nghiên cứu, tận tình hướng dẫn và chỉ bảo em trong quá trình thực hiện đồ án tốt nghiệp: "***Ứng dụng Học sâu trong Hệ thống phát hiện và cảnh báo té ngã***".

Em xin gửi lời cảm ơn các thầy, cô giáo trong trường TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP.HCM. Các thầy, cô giáo đã dạy bảo và truyền đạt cho em rất nhiều, giúp em có được một nền tảng kiến thức vững chắc sau những năm học tập tại trường.

Em cũng xin gửi lời cảm ơn chân thành tới các bạn cùng lớp 21THA5 đã ủng hộ, khuyến khích em trong suốt quá trình học tập tại trường.

Em xin chân thành cảm ơn!

Tp. Hồ Chí Minh, ngày ... tháng ... năm 2025

Sinh viên thực hiện

Phạm Văn Tiến

MỤC LỤC

LỜI CAM ĐOAN i
LỜI CẢM ƠN ii
MỤC LỤC iii
DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT vi
DANH MỤC CÁC BẢNG vii
DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ viii
Chương 1. TỔNG QUAN 1
1.1. Lý do chọn đề tài 1
1.2. Tổng quan nghiên cứu liên quan 2
1.3. Nhiệm vụ đồ án 4
1.3.1. Tính cấp thiết và lý do hình thành đề tài 4
1.3.2. Ý nghĩa khoa học và thực tiễn 5
1.3.3. Mục tiêu nghiên cứu 6
1.3.4. Đối tượng và phạm vi nghiên cứu 6
1.4. Cấu trúc báo cáo đồ án tốt nghiệp 7
Chương 2. CƠ SỞ LÝ THUYẾT 9
2.1. Tổng quan về Học sâu và Mạng nơ-ron 9
2.1.1. Giới thiệu Học sâu (Deep Learning) 9
2.1.2. Mạng nơ-ron nhân tạo 10
2.1.3. Ứng dụng của học sâu 12
2.2. Mạng nơ-ron tích chập 13
2.2.1. Giới thiệu mạng nơ-ron tích chập 13
2.2.2. Lớp tích chập (Convolutional layer) 14

2.2.3. Mạng nơ-ron tích chập 1D (Conv1D) 15
2.2.4. Ứng dụng của Mạng nơ-ron tích chập 16
2.3. Mạng nơ-ron hồi quy và các biến thể cải tiến 17
2.3.1. Mạng nơ-ron hồi quy – RNN 17
2.3.2. Mạng bộ nhớ ngắn hạn dài hạn – LSTM 19
2.3.3. Mạng bộ nhớ ngắn hạn dài hạn hai chiều – BiLSTM 21
2.3.4. Ứng dụng trong xử lý dữ liệu chuỗi thời gian 22
2.4. Mô hình học sâu CB-LSTM đề xuất 23
2.4.1. Kiến trúc tổng thể của mô hình CB-LSTM 23
2.4.2. Chi tiết các thành phần của mô hình 24
2.4.3. Ưu điểm của mô hình CB-LSTM đề xuất 26
2.5. Các bộ dữ liệu sử dụng 27
2.5.1. Bộ dữ liệu DU-MD 27
2.5.2. Bộ dữ liệu UMAFall 28
2.6. Các chỉ số đánh giá hiệu năng 29
2.7 Các Công nghệ Nền tảng của Hệ thống 31
2.7.1. Internet vạn vật (IoT) 31
2.7.2. Dịch vụ đám mây (Cloud Services) 31
2.7.3. Nền tảng phát triển ứng dụng di động Flutter 32
Chương 3. KẾT QUẢ THỰC NGHIỆM 33
3.1. Giới thiệu và phát biểu bài toán 33
3.1.1. Phát biểu bài toán 33
3.1.2. Môi trường phát triển và cài đặt 34
3.2. Quy trình thu thập và xử lý dữ liệu 34
3.2.1. Quy trình thu thập và xử lý bộ dữ liệu DU-MD 34

3.2.2. Quy trình thu thập và xử lý bộ dữ liệu bộ dữ liệu UMAFall 35
3.3. Huấn luyện và tối ưu hoá mô hình 35
3.3.1. Cấu hình mô hình 35
3.3.2. Kết quả huấn luyện 36
3.4. Đánh giá hiệu năng mô hình trên các bộ dữ liệu 39
3.4.1. Hiệu năng trên tập kiểm thử các bộ dữ liệu 39
3.4.2. So sánh hiệu năng với các mô hình học máy truyền thống 41
3.5. Phân tích kết quả và thảo luận 42
3.6. Triển khai hệ thống phát hiện và cảnh báo té ngã 44
3.7. Kết quả thử nghiệm thực tế 46
Chương 4. KẾT LUẬN VÀ KIẾN NGHỊ 51
4.1. Kết quả đạt được 51
4.2. Hạn chế 51
4.3. Hướng phát triển 52
TÀI LIỆU THAM KHẢO 53

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

Ký hiệu	Diễn giải tiếng Anh	Diễn giải tiếng Việt
ADL	Activities Daily Living	Hoạt động hằng ngày
BiLSTM	Bidirectional Long Short-Term Memory	Mạng nơ-ron bộ nhớ ngắn hạn dài hạn hai chiều
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
CNN 1D, Conv1D	Convolutional Neural Network 1D	Mạng nơ-ron tích chập 1 chiều
CB-LSTM	Convolutional Bidirectional Long Short-Term Memory	
DL	Deep Learning	Học sâu
DNN	Deep Neural Network	Mạng nơ-ron sâu
FC	Fully Connected Layers	Các lớp kết nối đầy đủ
FD	Fall Detection	Phát hiện té ngã
LSTM	Long Short-Term Memory	Mạng nơ-ron bộ nhớ ngắn hạn dài hạn
ML	Machine Learning	Học máy
NN	Neural Network	Mạng nơ-ron
RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy

DANH MỤC CÁC BẢNG

Bảng 2. 1: Mô tả bộ dữ liệu DU-MD 27

Bảng 2. 2: Mô tả bộ dữ liệu DU-MD 28

Bảng 3. 1: So sánh kết quả các mô hình học sâu trên bộ dữ liệu DU-MD..... 40

Bảng 3. 2: So sánh kết quả các mô hình học sâu trên bộ dữ liệu UMAFall 40

Bảng 3. 3: So sánh kết quả các mô hình học sâu và học máy trên bộ dữ liệu DU-MD 41

Bảng 3. 4: So sánh kết quả các mô hình học sâu và học máy trên bộ dữ liệu UMAFall .. 42

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

Hình 2. 1: Cấu tạo của mạng nơ-ron nhân tạo - Ảnh: Internet 11
Hình 2. 2: Hoạt động của lớp tích chập - Ảnh: indoml.com 15
Hình 2. 3: Cấu trúc cơ bản của Mạng nơ-ron hồi quy - Ảnh: Internet 18
Hình 2. 4: Cấu trúc bên trong một cell của LSTM - Ảnh: Internet 20
Hình 2. 5: Cấu trúc cơ bản của mạng BiLSTM - Ảnh: Internet 22
Hình 3. 1: Hình minh hoạ mô hình CB-LSTM 35
Hình 3. 2: Kết quả quá trình huấn luyện mô hình với bộ dữ liệu DU-MD 37
Hình 3. 3: Kết quả biểu đồ ma trận nhầm lẫn trên tập test 37
Hình 3. 4: Kết quả quá trình huấn luyện mô hình với bộ dữ liệu UMAFall 38
Hình 3. 5: Kết quả biểu đồ ma trận nhầm lẫn trên tập test 39
Hình 3. 6: Hình ảnh thực tế Orange Pi Zero 2W và cảm biến gia tốc ADXL345 44
Hình 3. 7: Kiến trúc tổng quan hệ thống phát hiện và cảnh báo té ngã 44
Hình 3. 8: Dữ liệu json giao tiếp giữa Orange Pi và AWS IoT Core 46
Hình 3. 9: Giao diện màn hình chính 47
Hình 3. 10: Màn hình danh sách sự kiện 48
Hình 3. 11: Màn hình cảnh báo khi té ngã 49
Hình 3. 12: Thông báo popup khi té ngã 49
Hình 3. 13: Email nhận được khi xảy ra sự kiện té ngã. 50

Chương 1. TỔNG QUAN

1.1. Lý do chọn đề tài

Trong vài thập kỷ trở lại đây, thế giới đang trải qua một sự thay đổi nhân khẩu học đáng kể với tỉ lệ già hoá dân số đang diễn ra ở hầu hết các quốc gia trên toàn cầu. Theo ước tính của Liên Hợp Quốc (2024), đến cuối những năm 2070, số người từ 65 tuổi trở lên sẽ nhiều hơn so với số trẻ dưới 18 tuổi [1]. Sự gia tăng này mang lại nhiều thách thức cho xã hội và y tế. Theo Tổ chức Y tế Thế giới (WHO), người cao tuổi thường mắc nhiều bệnh mãn tính cùng lúc như sa sút trí tuệ, tiểu đường, tim mạch và viêm khớp, làm gia tăng đáng kể chi phí chăm sóc y tế. Tuổi già cũng được đặc trưng bởi sự xuất hiện của một số tình trạng sức khỏe phức tạp thường được gọi là "hội chứng lão khoa". Chúng thường là hậu quả của nhiều yếu tố tiềm ẩn và bao gồm suy nhược, tiểu không tự chủ, té ngã, mê sảng và loét do tì đè [2]. Trong đó, té ngã ở người cao tuổi không chỉ là nguyên nhân hàng đầu gây chấn thương, tàn tật và thậm chí tử vong, mà còn dẫn đến gánh nặng đáng kể cho hệ thống chăm sóc sức khoẻ và giảm chất lượng cuộc sống của bản thân người bệnh [3]. Việc phát hiện té ngã kịp thời và chính xác là một yêu cầu cấp bách để đảm bảo an toàn, giảm thiểu hậu quả nghiêm trọng và nâng cao sự độc lập cho người cao tuổi từ đó giảm gánh nặng cho xã hội và y tế.

Trong bối cảnh trên, nhu cầu phát hiện sớm và kịp thời các tình huống té ngã ở người lớn tuổi để ngay lập tức có sự can thiệp kịp thời nhằm giảm thiểu những hậu quả ngày càng tăng, nhất là khi mà công nghệ đang đạt được những bước tiến rất nhanh như hiện nay. Các phương pháp giám sát truyền thống thường có những hạn chế cố hữu, chẳng hạn như yêu cầu sự giám sát thủ công liên tục từ cả người thân và người được giám sát hoặc phụ thuộc vào việc người dùng phải chủ động ấn nút khi gặp sự cố. Những phương pháp này hoàn toàn không hiệu quả khi tình huống xảy ra và người lớn tuổi không kịp phản ứng hoặc mất khả năng hành động. Với những bước phát triển đột phá công nghệ đầy hứa hẹn như hiện nay, việc nghiên cứu, ứng dụng các công nghệ tiên tiến, đặc biệt là học sâu (Deep Learning), kết hợp với các cảm biến nhỏ gọn có thể đeo, đa dụng như đồng hồ thông

minh đã nổi lên như một giải pháp đầy tiềm năng để giải quyết vấn đề này một cách hiệu quả [4].

Trong khoảng một thập kỷ trở lại đây, sự phát triển của các cảm biến có thể đeo mở ra một giải pháp đầy hứa hẹn cho phát hiện té ngã. Các cảm biến này được tích hợp ngày càng nhiều vào các thiết bị như đồng hồ thông minh, dây đeo cổ tay, thường là cảm biến gia tốc kế, con quay hồi chuyển, cảm biến áp suất hay cảm biến nhịp tim. Nhiều nghiên cứu đã khám phá các công nghệ cảm biến, thuật toán và thiết kế hệ thống khác nhau để cải thiện độ chính xác và độ tin cậy trong phát hiện té ngã. Hơn nữa, khi các cảm biến có thể đeo có mặt trên các thiết bị như đồng hồ thông minh, hay dây đeo cổ tay đã mở ra khả năng tích hợp với các công nghệ khác, chẳng hạn như Internet vạn vật (IoT) và trí tuệ nhân tạo (AI), tăng cường hơn nữa chức năng và tiềm năng của chúng đối với các giải pháp phát hiện té ngã cá nhân hóa [5]. Sự kết hợp giữa các cảm biến đeo tay thu nhỏ, mạnh mẽ được tích hợp vào các thiết bị như đồng hồ thông minh như một món trang sức nhiều công dụng mang lại cảm giác thoải mái cho người dùng với các mô hình tính toán tiên tiến (học sâu) đại diện cho một sự thay đổi mô hình từ phản ứng khẩn cấp sang giám sát sức khoẻ chủ động, liên tục. Điều này hướng đến một tương lai nơi việc theo dõi sức khoẻ cá nhân được tích hợp liền mạch vào cuộc sống hằng ngày.

Với tất cả các quan điểm trên, em quyết định lựa chọn đề tài "***Ứng dụng Học sâu trong Hệ thống phát hiện và cảnh báo té ngã***". Mục tiêu tổng thể của đề tài này là nghiên cứu, ứng dụng và kiểm chứng cho một hệ thống phát hiện và cảnh báo té ngã dựa trên học sâu, có khả năng hoạt động hiệu quả trên các thiết bị đeo tay hiện đại.

1.2. Tổng quan nghiên cứu liên quan

Dựa vào những nghiên cứu được công bố về phát hiện té ngã, các hệ thống phát hiện té ngã có thể được phân loại thành nhiều nhóm khác nhau, trong đó có ba nhóm lớn được đại diện cho các nhóm nhỏ hơn. Ba nhóm đó bao gồm: hệ thống dựa trên cảm biến (sensor-based); hệ thống dựa trên thị giác (vision); hệ thống dựa trên cảm biến môi trường (ambience) [6]. Trong số đó, các hệ thống sử dụng cảm biến (sensor-base) đeo được như cảm biến gia tốc kế, cảm biến gia tốc (có thể có thêm con quay hồi chuyển) trên điện thoại

di động, đồng hồ thông minh đã thu hút nhiều sự quan tâm do tính di động, nhỏ gọn, không làm mất sự riêng tư và tăng sự thoải mái cho người dùng và khả năng giám sát liên tục [5].

Các phương pháp phát hiện té ngã dựa trên cảm biến ban đầu sử dụng phương pháp ngưỡng (Threshold) để dự đoán. Hơn một thập kỷ trở lại đây, các phương pháp phát hiện té ngã bằng học máy truyền thống như Máy học vector hỗ trợ (SVM), Rừng ngẫu nhiên (Random Forest) và K - hàng xóm gần nhất (KNN) đã dần thay thế hoàn toàn. Bằng cách trích xuất đặc trưng của dữ liệu từ cảm biến cho các thuật toán học máy này có hiệu suất tốt khi phân biệt giữa các hoạt động bình thường và hành vi té ngã [4]. Tuy nhiên, những phương pháp này thường đòi hỏi quá trình trích xuất đặc trưng thủ công và chuyên sâu để mang lại kết quả tốt nhất. Trong nghiên cứu của tác giả Trần Công Án và các cộng sự (2017) [7] đã chỉ ra phương pháp học máy truyền thống phụ thuộc vào việc lựa chọn đặc trưng phù hợp mà việc lựa chọn đặc trưng lại phụ thuộc nhiều vào kinh nghiệm của người thực hiện để phân tích, chọn trích xuất các đặc trưng phù hợp. Qua đó nhóm tác giả cũng đề xuất sử dụng phương pháp học sâu, cho phép bỏ qua việc trích xuất đặc trưng. Trong bài đánh giá của tác giả Z. Jiang và các cộng sự (2024) [6] cũng đã nhận xét các thuật toán Fall Detection dựa trên học máy có hiệu suất tốt, nhưng việc trích xuất đặc trưng và đào tạo bộ phân loại mất nhiều thời gian. Do đó, không phù hợp với các ứng dụng phát hiện té ngã đòi hỏi thời gian phát hiện nhanh chóng, ngay khi vừa diễn ra. Sự phụ thuộc vào kỹ thuật trích xuất đặc trưng thủ công là một hạn chế đáng kể vì nó tạo ra một nút thắt cổ chai trong quy trình phát triển và triển khai.

Ngược lại, các phương pháp học sâu có thể giúp gỡ bỏ nút thắt cổ chai đó với khả trích xuất và học các đặc trưng trực tiếp từ dữ liệu cảm biến thô. Điều này giúp bỏ qua bước trích xuất đặc trưng phức tạp, giảm yêu cầu tiền xử lý dữ liệu qua đó giảm chi phí và thời gian phát triển các mô hình phát hiện té ngã. Ngoài ra, nhiều kiến trúc học sâu còn chuyên biệt cho việc cho các dữ liệu theo chuỗi thời gian như Recurrent Neural Network (RNN) hay phiên bản nâng cấp của nó là Long Short-Term Memory [7], mạng nơ-ron tích chập Convolutional Neural Network (CNN) với khả trăng trích xuất đặc trưng hiệu quả [9].

Nhiều bộ dữ liệu công khai đã được sử dụng rộng rãi trong các nghiên cứu phát hiện té ngã như UP-Fall, UMAFall, DU-MD, MobiFall, SisFall… Mỗi bộ dữ liệu lại có những đặc điểm riêng về số lượng đối tượng, loại hoạt động, loại té ngã được ghi lại.

Mặc dù học sâu đã cho thấy tiềm năng lớn nhưng vẫn còn một khoảng trống nghiên cứu cần được lấp đầy. Cụ thể, việc phát triển các kiến trúc học sâu được tối ưu hoá cho dữ liệu cảm biến gia tốc kế đeo ở cổ tay, kết hợp với việc đánh giá trên nhiều bộ dữ liệu đa dạng khác nhau và khả năng triển khai hệ thống phát hiện và cảnh báo té ngã trong thực tế.

1.3. Nhiệm vụ đồ án

1.3.1. Tính cấp thiết và lý do hình thành đề tài

Như đã đề cập, xu hướng già hoá dân số gia tăng trên toàn cầu dẫn đến một thách thức lớn cho hệ thống an sinh xã hội cũng như y tế tại hầu hết các quốc gia, mà trong đó té ngã là một trong những nguyên nhân hàng đầu. Điều này tạo nên một yêu cầu về một hệ thống phát hiện và cảnh báo té ngã hiệu quả nhằm giảm thiểu tối đa hậu quả do té ngã gây ra nhằm nâng cao sự an toàn và độc lập cho người cao tuổi. Các giải pháp giám sát hiện tại thường gặp phải những hạn chế như thiết bị cồng kềnh, phụ thuộc vào việc phải bấm nút thủ công khi xảy ra sự cố, ảnh hưởng đến quyền riêng tư của người sử dụng hoặc tỷ lệ báo động sai cao gây ra sự mệt mỏi cho người chăm sóc và nhân viên y tế.

Đề tài này tập trung vào việc sử dụng dữ liệu của các cảm biến có thể đeo ở tay như cảm biến gia tốc hiện đang được tích hợp vào hầu hết các thiết bị đeo tay phổ biến như đồng hồ thông minh, dây đeo tay. Việc chỉ sử dụng các loại cảm biến có thể đeo ở tay mang lại sự thoải mái và thân thiện đối với người dùng và cũng phù hợp với xu thế hiện nay. Đồng hồ thông minh ngày càng được phát triển với hiệu năng cao, cung cấp một nền tảng lý tưởng cho việc triển khai hệ thống giám sát tự động liên tục, không xâm lấn. Một hệ thống giám sát như thế còn có thể được ứng dụng để giám sát trẻ em, công nhân, người lao động trong các môi trường đặc biệt có nguy cơ cao về té ngã, thậm chí còn có thể giám sát các đối tượng mắc các bệnh đặc biệt ảnh hưởng đến khả năng vận động.

Việc lựa chọn học sâu được thúc đẩy bởi xu hướng phát triển của công nghệ và những lợi thế vốn có của nó: khả năng xử lý dữ mạnh mẽ, giảm đáng kể nhu cầu phân tích

và trích xuất đặc trưng thủ công tốn nhiều tài nguyên, mang lại hiệu suất tốt cùng khả năng tổng quát hoá cao. Sự kết hợp giữa sự thoải mái cho người dùng và tiềm năng của công nghệ là động lực quan trọng cho việc ứng dụng thực tế các hệ thống phát hiện té ngã. Điều này hàm ý rằng sự chấp nhận của người dùng cũng quan trọng như hiệu suất kỹ thuật đối với tác động trong thực tế.

Hơn nữa, việc tập trung vào cảm biến đeo ở cổ tay, mặc dù là một lựa chọn thiết kế cụ thể, đã gián tiếp giải quyết thách thức về tính không đồng nhất của dữ liệu trên các vị trí đặt cảm biến khác nhau. Bằng cách chứng minh hiệu quả đối với dữ liệu cảm biến ở cổ tay, nó gợi ý một con đường hướng tới các giải pháp phát hiện té ngã được tiêu chuẩn hoá và áp dụng phổ biến hơn, giảm sự phụ thuộc vào một loại thiết bị cụ thể. Mô hình học sâu không quá phụ thuộc vào thiết bị thu thập và triển khai, điều này là một ưu điểm đáng kể.

1.3.2. Ý nghĩa khoa học và thực tiễn

Đề tài này mang lại những đóng góp hữu ích cả về mặt khoa học và thực tiễn.

Về mặt khoa học:

- Đề xuất và kiểm chứng một kiến trúc học sâu mới, được thiết kế đặc biệt để xử lý dữ liệu gia tốc kế theo chuỗi thời gian được thu thập từ cảm biến đeo ở cổ tay.
- Chứng minh thực nghiệm hiệu quả của học sâu trong việc giảm thiểu nhu cầu kỹ thuật xử lý dữ liệu và trích xuất đặc trưng so với các kỹ thuật học máy truyền thống.
- Thực hiện đánh giá trên nhiều bộ dữ liệu đa dạng (DU-MD, UMAFall) để đánh giá khả năng tổng quát hoá và khả năng xử lý mạnh mẽ của mô hình.

Về mặt thực tiễn:

- Phát triển thử nghiệm một hệ thống phát hiện té ngã thực tế, hoạt động theo thời gian thực, có khả năng triển khai trên các thiết bị có tài nguyên hạn chế. Điều này phù hợp với xu hướng phát triển đồng hồ thông minh hiệu năng cao.
- Hệ thống được tích hợp công nghệ IoT với các dịch vụ đám mây và ứng dụng di động để cung cấp khả năng cảnh báo ngay lập tức cho người chăm sóc và nhân viên y tế.

- Cho thấy tiềm năng nâng cao sự độc lập và an toàn cho người cao tuổi, giảm gánh nặng cho người chăm sóc và nhân viên y tế từ đó giảm áp lực lên xã hội.

Sự tập trung kép vào tính mới trong khoa học và tính hữu ích trong thực tiễn định vị nghiên cứu này như một ví dụ về AI ứng dụng, thu hẹp khoảng cách giữa lý thuyết khoa học và lợi ích khi triển khai trong thực tiễn.

1.3.3. Mục tiêu nghiên cứu

- Nghiên cứu ứng dụng của học sâu trong phát hiện té ngã, phát triển và tối ưu hoá một mô hình học sâu có hiệu suất tốt để phát hiện té ngã sử dụng dữ liệu gia tốc kế từ cảm biến đeo ở cổ tay.
- Đánh giá hiệu suất mô hình so với các phương pháp học máy truyền thống trên các bộ dữ liệu công khai phù hợp được chọn và khả năng triển khai thực tế.
- Xây dựng thử nghiệm một hệ thống phát hiện và cảnh báo té ngã có triển khai mô hình học sâu đề xuất trên thiết bị có tài nguyên hạn chế mô phỏng thiết bị đeo trong hệ thống qua đó chứng minh tính khả thi trong thực tế.

1.3.4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: trọng tâm chính là ứng dụng học sâu vào hệ thống phát hiện và cảnh báo té ngã cho người cao tuổi nên đề tài này tập trung vào các đối tượng nghiên cứu chính sau:

- Các thuật toán học sâu và ứng dụng để xử lý dữ liệu gia tốc kế từ thiết bị cảm biến có thể đeo ở tay trong một hệ thống phát hiện và cảnh báo té ngã.
- Dữ liệu gia tốc kế được thu thập từ cảm biến đeo ở cổ tay từ các bộ dữ liệu té ngã được công khai.
- Mô hình truyền tải dữ liệu IoT trong hệ thống phát hiện và cảnh báo té ngã, bao gồm ba thành phần chính: thiết bị triển khai mô hình và cảm biến đeo ở cổ tay, dịch vụ cloud để xử lý và lưu trữ dữ liệu, ứng dụng di động nhận và thông báo kết quả.

Phạm vi giới hạn nghiên cứu:

- Loại dữ liệu cảm biến: Chủ yếu tập trung vào dữ liệu gia tốc kế ba trục (x, y, z) của bộ dữ liệu DU-MD, đây là bộ dữ liệu được thu thập từ duy nhất cảm biến đeo ở cổ tay, hoàn toàn phù hợp với hướng tiếp cận của đề tài [10]. Mặc dù bộ dữ liệu UMAFall có dữ liệu từ cảm biến đeo trên cổ tay, tuy nhiên đó là dữ liệu trong một bộ dữ liệu nhiều cảm biến kết hợp, không chuyện biệt và dẫn đến những khó khăn trong xử lý trích xuất dữ liệu từ cảm biến đeo ở cổ tay, nên trong đề tài này, bộ dữ liệu UMAFall được sử dụng để kiểm tra hiệu suất.
- Vị trí cảm biến: Tập trung vào dữ liệu cảm biến đeo ở cổ tay, phù hợp để ứng dụng lên các thiết bị như đồng hồ thông minh, dây đeo tay.
- Mô hình học sâu: Tập trung vào nghiên cứu, đề xuất mô hình sử dụng mạng nơ-ron hồi quy (Recurrent Neural Network), mạng nơ-ron bộ nhớ ngắn hạn-dài hạn (Long Short-Term Memory), mạng nơ-ron tích chập (Convolutional Neural Network).
- Nền tảng triển khai: Triển khai mô hình và cảm biến trên Orange Pi Zero 2W, dịch vụ đám mây Amazon Web Services (AWS), ứng dụng di động Android bằng Flutter.
- Giới hạn thử nghiệm: Giới hạn trong việc kiểm tra hiệu suất của mô hình học sâu phân biệt té ngã và hoạt động hằng ngày (phân loại nhị phân giữa té ngã và hoạt động hằng ngày) cho tính khả thi và do độ lớn của bộ dữ liệu. Các hoạt động hằng ngày và té ngã được chọn để kiểm thử phù hợp với thực tế có thể mô phỏng xuất hiện trong bộ dữ liệu đã chọn.

1.4. Cấu trúc báo cáo đồ án tốt nghiệp

Cấu trúc đồ án được chia thành các chương sau:

Chương 1. TỔNG QUAN

- Giới thiệu về đề tài.
- Tổng quan các nghiên cứu liên quan.
- Xác định rõ nhiệm vụ, mục tiêu, và phạm vi của đồ án.

Chương 2. CƠ SỞ LÝ THUYẾT

- Trình bày các khái niệm nền tảng về học sâu, mạng nơ-ron được sử dụng.
- Mô tả chi tiết kiến trúc mô hình học sâu đề xuất.

- Giới thiệu các bộ dữ liệu và chỉ số đánh giá hiệu năng được sử dụng.

Chương 3. KẾT QUẢ THỰC NGHIỆM

- Mô tả quy trình phát triển, huấn luyện, và tối ưu hóa mô hình đã đề xuất.
- Trình bày và phân tích các kết quả, đánh giá hiệu năng.
- Chi tiết quá trình triển khai hệ thống và kết quả thử nghiệm thực tế.

Chương 4. KẾT LUẬN VÀ KIẾN NGHỊ

- Tóm tắt những kết quả chính đạt được, khẳng định đóng góp của đề tài.
- Đề xuất các hướng phát triển trong tương lai.

PHỤ LỤC

TÀI LIỆU THAM KHẢO

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. Tổng quan về Học sâu và Mạng nơ-ron

2.1.1. Giới thiệu Học sâu (Deep Learning)

Học sâu (Deep Learning) là một lĩnh vực con quan trọng của Học máy (Machine Learning) và là một thành phần cốt lõi của Trí tuệ Nhân tạo (AI) hiện đại. Sự nổi bật của Học sâu gắn liền với khả năng xử lý lượng dữ khổng lồ, liên tục học hỏi từ nó để cung cấp kết quả chính xác cao, dẫn đến những cải tiến đáng kể về hiệu suất mô hình và mở rộng các lĩnh vực ứng dụng của Học sâu [11]. Về cơ bản, Deep Learning lấy cảm hứng từ kiến trúc và chức năng của bộ não con người. Nó sử dụng các mô hình tính toán gọi là Mạng nơ-ron (Neural Networks - NNs), bao gồm nhiều đơn vị xử lý cơ bản gọi là nút, nơ-ron hoặc perceptron hoạt động song song mà không có đơn vị điều khiển trung tâm [12]. Thuật ngữ "sâu" (deep) dùng để chỉ việc sử dụng nhiều lớp ẩn (hidden layer) trong mạng, cho phép chuyển đổi dữ liệu đầu vào thô thành các biểu diễn ngày càng trừu tượng và phức tạp hơn thông qua nhiều lớp biểu diễn đơn giản [13].

Thuật ngữ "học sâu" trở nên phổ biến sau công trình nghiên cứu của Geoffrey E. Hinton và các cộng sự vào năm 2006, họ đã giới thiệu mô hình Deep Belief Networks (DBNs), chứng minh khả năng huấn luyện các mạng nơ-ron sâu mà không gặp vấn đề với vanishing gradient nhờ vào chiến lược greedy layer-wise pretraining. Một đặc điểm nổi bật của Deep Learning là khả năng tự động học các mẫu phức tạp trực tiếp từ các tập dữ liệu lớn. Không giống các phương pháp học máy truyền thống có hiệu suất ổn định sau một kích thước dữ liệu nhất định, hiệu suất của học sâu có xu hướng tăng lên khi kích thước dữ liệu tăng. Các tính năng chính của DL bao gồm khả năng học đặc trưng tự động, loại bỏ nhu cầu kỹ thuật trích xuất đặc trưng thủ công, và khả năng mở rộng cao, cho phép nó nhận dạng, phân loại và phân tích hiệu quả nhiều loại dữ liệu khác nhau, bao gồm dữ liệu có cấu trúc, hình ảnh, văn bản và âm thanh. Điều này đã khiến Deep Learning được xem như là một cuộc cách mạng trong Trí tuệ Nhân tạo với khả năng ứng dụng rộng rãi trong nhiều lĩnh vực [13].

2.1.2. Mạng nơ-ron nhân tạo

Mạng nơ-ron nhân tạo (Artificial Neural Network - ANN) thường được gọi tắt là mạng nơ-ron (Neural Network - NN), là một hệ thống tính toán lấy cảm hứng từ cấu trúc và hoạt động của các nơ-ron trong hệ thần kinh sinh học của con người. Tuy lấy cảm hứng từ não bộ nhưng mục đích chính là mô phỏng cách hoạt động để giải quyết các bài toán, chứ không phải tái tạo chính xác hoạt động của não [16].

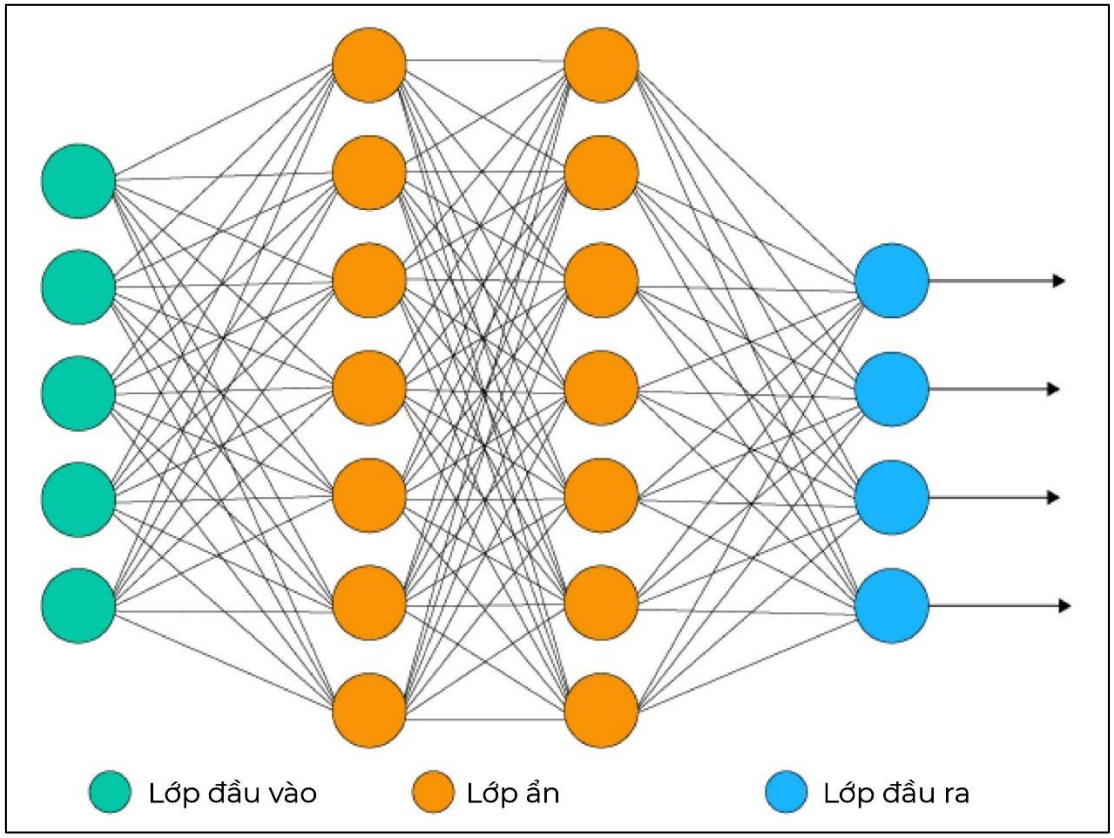
Một mạng nơ-ron nhân tạo cơ bản thường bao gồm các thành phần sau:

- Đơn vị cơ bản (nơ-ron): Đây là đơn vị xử lý thông tin cơ bản nhất của mạng, Mỗi nơ-ron nhận một hoặc nhiều đầu vào sau đó thực hiện tính toán và sau đó áp dụng một hàm kích hoạt (activation function) để tạo ra một đầu ra. Đầu ra này sau đó được lan truyền đến các nơ-ron khác trong mạng.
- Trọng số (Weights): Mỗi kết nối giữa hai nơ-ron có một trọng số. Nó biểu thị độ quan trọng của tin hiệu đi qua kết nối đó. Trong quá trình học, các trọng số này sẽ được điều chỉnh để mạng có thể học và đưa ra các dự đoán chính xác hơn.
- Độ lệch (Bias): Đây là một giá trị bổ sung được thêm vào tổng có trọng số của các đầu vào trước khi áp dụng hàm kích hoạt. Độ lệch giúp dịch chuyển đầu ra của nơ-ron và cho phép mạng linh hoạt hơn trong việc học các mẫu dữ liệu khác nhau.
- Hàm kích hoạt (Activation Function): Là các hàm phi tuyến được áp dụng sau khi tính tổng có trọng số các đầu vào của một nơ-ron. Các hàm kích hoạt phi tuyến là yếu tố quan trọng giúp mạng nơ-ron có khả năng biểu diễn các mối quan hệ phi tuyến phức tạp trong dữ liệu, vượt xa khả năng của các mô hình tuyến tính đơn giản như hồi quy tuyến tính. Các hàm kích hoạt phổ biến bao gồm Sigmoid, ReLU (Rectified Linear Unit), Tanh [13].
- Các lớp (Layers): Một mạng nơ-ron thường được tổ chức thành các lớp:
    - Lớp đầu vào (Input Layer): Nhận dữ liệu thô từ bên ngoài (ví dụ: pixel của hình ảnh, giá trị từ cảm biến). Mỗi nơ-ron trong lớp này tương ứng với một đặc trưng đầu vào [13].
    - Các lớp ẩn (Hidden Layers): Nằm giữa lớp đầu vào và lớp đầu ra. Các lớp này thực hiện phần lớn quá trình tính toán phức tạp, trích xuất các đặc trưng

và biến đổi dữ liệu đầu vào thành các biểu diễn trừu tượng hơn. Một mạng có nhiều hơn một lớp ẩn được gọi là mạng nơ-ron sâu (Deep Neural Network) [13].

- Lớp đầu ra (Output Layer): Tạo ra kết quả cuối cùng của mạng. Số lượng nơ-ron trong lớp này phụ thuộc vào loại bài toán (ví dụ: một nơ-ron cho bài toán phân loại nhị phân, nhiều nơ-ron cho phân loại đa lớp hoặc dự đoán giá trị liên tục) [13].

Cấu trúc minh hoạ của một mạng nơ-ron cơ bản được biểu diễn trong hình 2.1.



Hình 2. 1: Cấu tạo của mạng nơ-ron nhân tạo - Ảnh: Internet

Quá trình hoạt động của một mạng nơ-ron nhân tạo thường diễn ra theo hai giai đoạn chính:

- Forward propagation: Dữ liệu truyền từ lớp đầu vào qua các lớp ẩn đến lớp đầu ra, tại mỗi nút tính toán tổng có trọng số và áp dụng hàm kích hoạt.

- Backpropagation: Phương pháp lan truyền ngược sai số từ đầu ra về các lớp trước, mục đích cập nhật trọng số và bias, giảm thiểu sai số bằng thuật toán tối ưu như Gradient Descent.

Quá trình huấn luyện là quá trình thực hiện forward propagation và backpropagation lặp đi lặp lại nhiều lần (iterations) trên toàn bộ tập dữ liệu. Qua mỗi lần lặp, hàm mất mát sẽ giảm dần và dự đoán của mạng nơ-ron ngày càng chính xác hơn [17]. Thông qua quá trình huấn luyện phức tạp mà mạng nơ-ron có khả năng học được những đặc trưng phức tạp hơn của dữ liệu một cách tự động từ đó nâng cao độ chính xác, NN đã trở thành nền tảng cho nhiều công nghệ quan trọng, thúc đẩy sự phát triển nhanh chóng của Trí tuệ Nhân tạo.

2.1.3. Ứng dụng của học sâu

Hiện nay, không khó để bắt gặp được những ứng dụng thực tiễn của DL trong đời sống hiện đại. Có thể kể đến như:

- Xử lý ngôn ngữ tự nhiên (NLP – Natural Language Progressing):
    - Dịch tự động: Google Translate
    - Trợ lý ảo: Siri, Google Assistant, v.v.
    - Phân tích cảm xúc: Đánh giá cảm xúc, bình luận trên mạng xã hội; đánh giá sản phẩm.
- Thị giác máy tính:
    - Nhận diện khuôn mặt: Mở khóa điện thoại, camera giám sát
    - Chẩn đoán trong y học: Ảnh X-quang, MRI
- Tài chính – ngân hàng:
    - Phát hiện gian lận giao dịch (fraud detection)
    - Chấm điểm tín dụng (credit scoring)
    - Dự báo thị trường chứng khoán (kết hợp NN và mô hình chuỗi thời gian ARIMA).

2.2. Mạng nơ-ron tích chập

2.2.1. Giới thiệu mạng nơ-ron tích chập

Mạng nơ-ron tích chập (Convolutional Neural Network - CNN), hay còn gọi là mạng tích chập, là một loại mạng nơ-ron được thiết kế đặc biệt để xử lý các loại dữ liệu có cấu trúc dạng lưới [13]. Nó có nhiều lớp kết hợp lại với nhau, mỗi lớp sẽ làm nhiệm vụ trích xuất ra một thông tin có giá trị thông qua các bộ lọc (filter).

Mục đích của mạng nơ-ron tích chập là nó sẽ được tối ưu hóa để xử lý các dữ liệu dạng lưới, mà phổ biến nhất là hình ảnh (được biểu diễn dưới dạng ma trận hai chiều của các điểm ảnh) và dữ liệu chuỗi thời gian. CNN nổi bật so với các mạng nơ-ron truyền thẳng (MLPs) bởi khả năng tự động học các đặc trưng cấp độ khác nhau từ dữ liệu thô, thay vì yêu cầu kỹ thuật trích chọn đặc trưng thủ công [13]. Kiến trúc của CNN được xây dựng dựa trên ba ý tưởng kiến trúc chính: trường tiếp nhận cục bộ (local receptive fields), chia sẻ trọng số (shared weights) và gộp (pooling) [19].

- Trường tiếp nhận cục bộ (Local Receptive Fields): Mỗi nơ-ron trong một lớp tích chập chỉ kết nối với một vùng nhỏ (cục bộ) của đầu vào từ lớp trước đó, tương tự như cách các nơ-ron trong vỏ não thị giác phản ứng với các kích thích trong một vùng cụ thể của trường thị giác. Điều này giúp mạng tập trung vào các đặc trưng cục bộ như cạnh, đường nét hoặc các chi tiết nhỏ.
- Chia sẻ trọng số (Shared Weights): Thay vì mỗi nơ-ron có một tập hợp trọng số riêng, các nơ-ron trong cùng một bản đồ đặc trưng (feature map) của lớp tích chập sử dụng chung một bộ trọng số (gọi là bộ lọc - filter hoặc kernel) và độ lệch (bias). Điều này giúp giảm đáng kể số lượng tham số cần học, làm cho mô hình hiệu quả hơn về mặt tính toán và cải thiện khả năng tổng quát hóa, vì cùng một đặc trưng có thể được phát hiện ở các vị trí khác nhau trong đầu vào [18], [19].
- Gộp (Pooling): Thường theo sau các lớp tích chập, các lớp gộp (pooling layers) thực hiện lấy mẫu xuống (down-sampling) các bản đồ đặc trưng. Điều này giúp giảm kích thước không gian của dữ liệu, giảm số lượng tham số và tính toán, đồng thời làm cho mô hình bền vững hơn đối với các biến đổi nhỏ như dịch chuyển hoặc biến

dạng trong dữ liệu đầu vào [18], [19]. Các loại gộp phổ biến nhất là Max Pooling (lấy giá trị lớn nhất trong một vùng) và Average Pooling (lấy giá trị trung bình trong một vùng).

Kiến trúc điển hình của một mạng CNN bao gồm xen kẽ các lớp tích chập, lớp kích hoạt phi tuyến tính (thường là ReLU), và lớp gộp, tiếp theo là một hoặc nhiều lớp kết nối đầy đủ (Fully Connected) ở cuối để thực hiện phân loại hoặc hồi quy [18].

2.2.2. Lớp tích chập (Convolutional layer)

Là phép toán cốt lõi của CNN, được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc dạng lưới, phổ biến nhất là hình ảnh, nhưng cũng có thể là chuỗi thời gian hoặc các loại dữ liệu đa kênh, đa chiều khác [13]. Nói về tên gọi "Tích chập", nó bắt nguồn từ việc mô hình mạng này sử dụng phép toán tích chập (convolution) thay vì phép nhân ma trận thông thường ở ít nhất một lớp của nó [13]. Đây cũng là điểm khác biệt giữa mạng tích chập và mạng truyền thẳng (fully connected network).

Lớp này thường áp dụng một bộ lọc (kernel hoặc filter) có kích thước nhỏ (ví dụ: 3×3, 5×5) để quét toàn bộ ảnh đầu vào, từ đó tạo ra bản đồ đặc trưng (feature map) thể hiện sự hiện diện của các đặc trưng cục bộ như cạnh, góc, hoặc kết cấu [13]. Ví dụ đối với dữ liệu hình ảnh, các bộ lọc này sẽ học và trích xuất các đặc trưng như đường viền, màu sắc, hoặc hình dạng. Chính nhờ đặc trưng này của lớp tích chập mà đã chứng minh được sự hiệu quả vượt trội trong việc xử lý hình ảnh so với các mạng nơ-ron truyền thống, đặc biệt là về số lượng tham số và khả năng học [16].

Nguyên lý hoạt động:

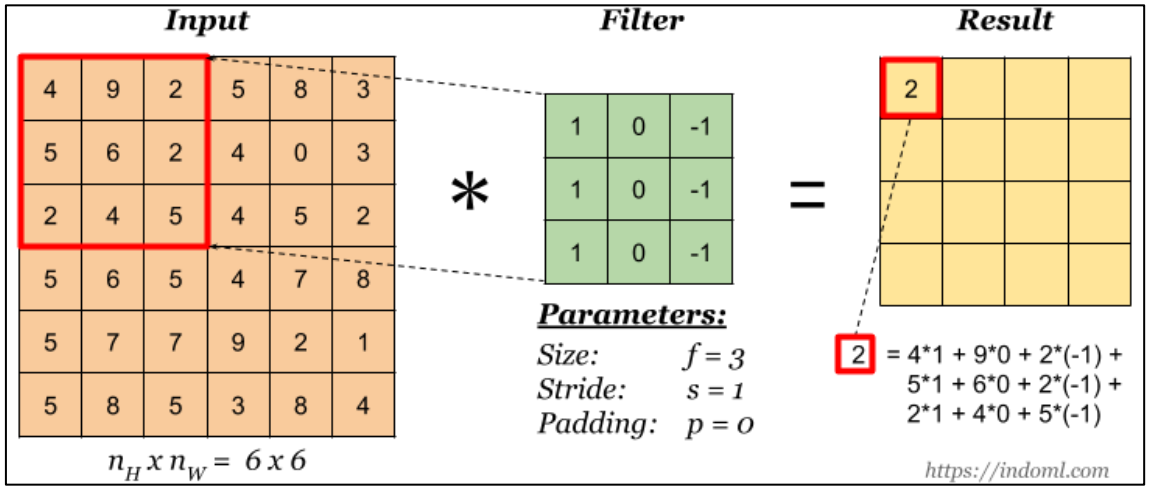
- Phép tích chập: Bộ lọc trượt qua đầu vào theo từng bước (stride). Tại mỗi vị trí, bộ lọc được nhân từng phần tử với vùng tương ứng của đầu vào, và tổng của các tích này được tính toán để tạo ra một giá trị duy nhất trong bản đồ đặc trưng đầu ra. Công thức tích chập hai chiều cho một pixel (i, j) trong bản đồ đặc trưng đầu ra S khi áp dụng bộ lọc K trên đầu vào I là:

$$S(i,j) = (I * K)(i,j) = \sum_{m} \sum_{n} I(i-m, j-n)K(m,n)$$

Trong đó:

- I: Ma trận đầu vào
- K: Ma trận bộ lọc (kernel)
- (m, n): Chỉ số của các phần tử trong bộ lọc

- Hàm kích hoạt: Sau khi thực hiện phép tích chập, đầu ra thường được đưa qua một hàm kích hoạt phi tuyến tính (ví dụ: ReLU) để thêm tính phi tuyến vào mô hình, giúp mạng học được các mối quan hệ phức tạp hơn.
- Bước trượt (Stride): Tham số stride xác định khoảng cách dịch chuyển của filter sau mỗi lần tính toán. Stride lớn hơn 1 giúp giảm kích thước feature map, đồng thời giảm độ phức tạp tính toán [22].
- Padding: Kỹ thuật thêm các giá trị 0 xung quanh ảnh đầu vào để duy trì kích thước feature map sau tích chập, thường được ký hiệu là "SAME" trong các framework như TensorFlow hoặc PyTorch [13].



Hình 2. 2: Hoạt động của lớp tích chập - Ảnh: indoml.com

Mỗi lớp tích chập có thể sử dụng nhiều bộ lọc khác nhau, mỗi bộ lọc phát hiện một loại đặc trưng khác nhau, từ đó tạo ra nhiều bản đồ đặc trưng riêng biệt.

2.2.3. Mạng nơ-ron tích chập 1D (Conv1D)

Là loại tích chập được áp dụng trên dữ liệu một chiều, thường là các chuỗi dữ liệu, được viết tắt là Conv1D. Lý do vì sao gọi là "một chiều" thì phải nói đến trong phép toán này, bộ lọc (kernel) chỉ di chuyển theo một chiều duy nhất, tức là dọc theo chiều dài của

vector dữ liệu đầu vào. Khi đi qua mỗi bước, bộ lọc sẽ tính toán tích chập giữa chính nó và một đoạn nhỏ của dữ liệu, từ đó học được các đặc trưng cục bộ theo chuỗi. Cũng chính vì đặc trưng này mà Conv1D phù hợp với dữ liệu tuần tự (sequential data) và dữ liệu thời gian (temporal data).

Đặc điểm và hoạt động:

- Dữ liệu đầu vào: 1D CNN phù hợp với các loại dữ liệu như tín hiệu âm thanh thô, dữ liệu cảm biến chuỗi thời gian (ví dụ: từ gia tốc kế, con quay hồi chuyển, ECG), chuỗi văn bản (sau khi được nhúng - embedding), hoặc bất kỳ dữ liệu nào có mối quan hệ phụ thuộc theo một trục duy nhất [20].
- Bộ lọc (Kernel): Bộ lọc trong 1D CNN là một vector một chiều. Nó trượt dọc theo chiều dài của chuỗi đầu vào, thực hiện phép tích chập để tạo ra một bản đồ đặc trưng một chiều.
- Ứng dụng: 1D CNN đặc biệt hiệu quả trong việc phát hiện các mẫu cục bộ, các đặc trưng tần số hoặc các mối quan hệ theo thời gian trong dữ liệu chuỗi. Chúng có thể được sử dụng cho các bài toán như phân loại chuỗi thời gian, dự đoán chuỗi, và nhận dạng hoạt động.

2.2.4. Ứng dụng của Mạng nơ-ron tích chập

Về ứng dụng của mạng nơ-ron tích chập CNN có mặt trong rất nhiều lĩnh vực trong đời sống. Dễ thấy và được ứng dụng nhiều nhất trong lĩnh vực về thị giác máy tính và xử lý hình ảnh. Chúng được thiết kế một cách độc đáo để tự động học và thích nghi với các hệ thống phân cấp không gian của các đặc trưng từ dữ liệu đầu vào [21]. Với cơ chế áp dụng các bộ lọc học và trích xuất được các đặc trưng, CNN có khả năng xử lý các dữ liệu có cấu trúc lưới (thường thấy trong hình ảnh, âm thanh và văn bản). Việc này đã giải thích được lý do tại sao CNN lại trở thành nền tảng cốt lõi cho hầu hết các ứng dụng thị giác máy tính hiện đại.

Bên cạnh đó, CNN cũng được mở rộng áp dụng sang nhiều lĩnh vực đa dạng hơn. Có thể kể đến việc có thể áp dụng CNN trong xử lý ngôn ngữ tự nhiên (NLP - Natural

Language Processing), xử lý âm thanh cho nhận dạng giọng nói, nhận diện hành động con người (HAR - Human Activity Recognition), v.v

2.3. Mạng nơ-ron hồi quy và các biến thể cải tiến

2.3.1. Mạng nơ-ron hồi quy – RNN

Mạng nơ-ron hồi quy (Recurrent Neural Network - RNN) là một loại mạng nơ-ron được thiết kế đặc biệt để xử lý các loại dữ liệu có cấu trúc tuần tự hoặc chuỗi, nơi thông tin từ các bước trước đó ảnh hưởng đến các bước hiện tại và tương lai. Khác với mạng nơ-ron truyền thẳng (feedforward network) – vốn xử lý từng đầu vào một cách độc lập, RNN có khả năng ghi nhớ thông tin từ quá khứ và sử dụng nó để đưa ra dự đoán hoặc xử lý đầu vào hiện tại [13].

Điểm khác biệt chính của RNN so với các mạng nơ-ron khác là sự hiện diện của các vòng lặp phản hồi (feedback loops) trong kiến trúc của chúng. Điều này cho phép thông tin từ bước thời gian hiện tại được truyền trở lại nơ-ron và ảnh hưởng đến đầu ra của nơ-ron đó ở bước thời gian tiếp theo [22].

Cấu trúc cơ bản của một nơ-ron hồi quy có thể được hình dung như sau:

- Trạng thái ẩn (Hidden State – h_t): Đây là "bộ nhớ" của mạng tại thời điểm t. Trạng thái ẩn được cập nhật dựa trên đầu vào hiện tại (x_t) và trạng thái ẩn của bước thời gian trước đó ($h_t - 1$).
- Vòng lặp: Vòng lặp thể hiện việc một phần của đầu ra của nơ-ron ở thời điểm $t - 1$ được đưa trở lại làm đầu vào cho nơ-ron ở thời điểm t. Điều này cho phép mạng ghi nhớ thông tin trong một khoảng thời gian nhất định.

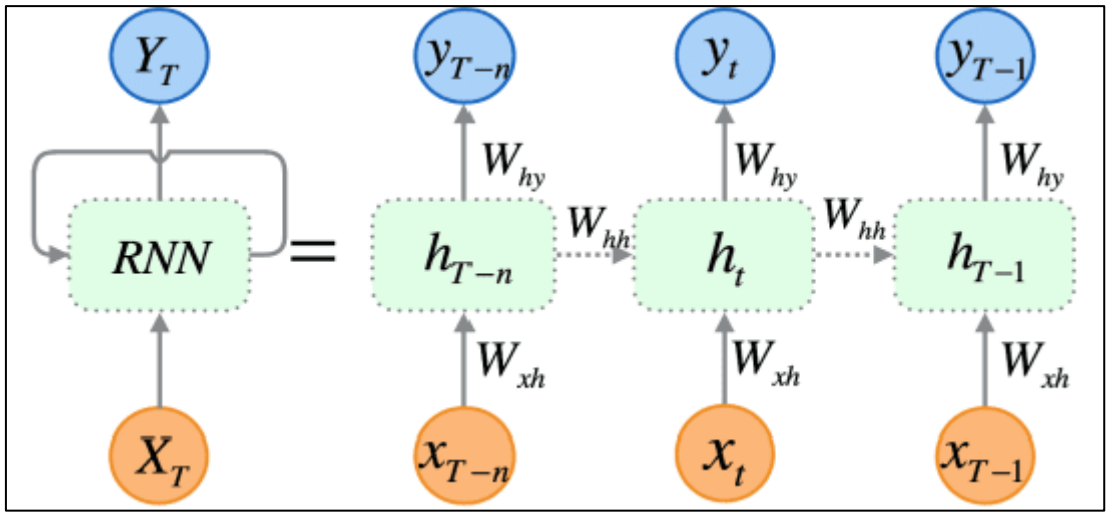
Công thức cập nhật trạng thái ẩn và đầu ra của một RNN cơ bản tại thời điểm t có thể được biểu diễn như sau:

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

$$y_t = W_{hy}h_t + b_y$$

Trong đó:

- x_t : Đầu vào tại thời điểm t
- h_t : Trạng thái ẩn tại thời điểm t
- y_t : Đầu ra tại thời điểm t
- W_{hh} : Ma trận trọng số cho kết nối từ trạng thái ẩn trước đó đến trạng thái ẩn hiện tại.
- W_{xh}: Ma trận trọng số cho kết nối từ đầu vào hiện tại đến trạng thái ẩn.
- W_{hy}: Ma trận trọng số cho kết nối từ trạng thái ẩn đến đầu ra.
- b_h, b_y : Các vector độ lệch (bias).
- f: Hàm kích hoạt phi tuyến tính (ví dụ: tanh hoặc ReLU).



Hình 2. 3: Cấu trúc cơ bản của Mạng nơ-ron hồi quy - Ảnh: Internet

RNN được huấn luyện bằng cách sử dụng thuật toán lan truyền ngược qua thời gian (Backpropagation Through Time - BPTT), một phần mở rộng của thuật toán lan truyền ngược tiêu chuẩn [23]. BPTT tính toán gradient của hàm mất mát đối với các trọng số của mạng bằng cách tổng hợp các gradient từ tất cả các bước thời gian.

2.3.2. Mạng bộ nhớ ngắn hạn dài hạn – LSTM

Mạng bộ nhớ ngắn hạn dài hạn (Long Short-Term Memory - LSTM) là một dạng đặc biệt của mạng nơ-ron hồi quy (RNN), được giới thiệu bởi Hochreiter & Schmidhuber vào năm 1997 [24]. LSTM được thiết kế để khắc phục những nhược điểm chính của RNN truyền thống, đặc biệt là vấn đề gradient biến mất (vanishing gradient problem), từ đó giúp mạng có khả năng học và ghi nhớ các phụ thuộc dài hạn (long-term dependencies) trong chuỗi dữ liệu một cách hiệu quả [25], [26].

Thay vì một mô-đun nơ-ron duy nhất như RNN truyền thống, mỗi đơn vị LSTM (LSTM cell) có một cấu trúc phức tạp hơn, bao gồm một "ô nhớ" (cell state) chạy dọc theo chuỗi và ba "cổng" (gates) điều khiển dòng thông tin vào, ra, và được lưu giữ trong ô nhớ [27], [28]:

- Ô nhớ (Cell State - C_t): Là "đường cao tốc" chính của thông tin trong LSTM. Nó chạy thẳng qua toàn bộ chuỗi, cho phép thông tin đi qua mà không bị thay đổi đáng kể. Các cổng sẽ thêm hoặc bớt thông tin vào ô nhớ này.
- Cổng quên (Forget Gate - f_t): Quyết định thông tin nào từ ô nhớ cũ (C_{t-1}) sẽ bị loại bỏ. Cổng này nhận đầu vào là trạng thái ẩn trước đó (h_{t-1}) và đầu vào hiện tại (x_t), sau đó áp dụng một hàm sigmoid (σ) để tạo ra một véc-tơ có giá trị từ 0 đến 1. Giá trị 0 nghĩa là "quên hoàn toàn", và 1 nghĩa là "giữ lại hoàn toàn".

$$ft = \sigma(Wf \cdot [ht - 1, xt] + bf)$$

- Cổng đầu vào (Input Gate - i_t) và Cập nhật ô nhớ ứng cử viên ($\widetilde{C_t}$): Quyết định thông tin mới nào sẽ được thêm vào ô nhớ.
    - Cổng đầu vào (i_t) sử dụng hàm sigmoid để xác định phần nào của thông tin mới cần được cập nhật (các giá trị gần 1 sẽ được giữ lại, gần 0 sẽ bị loại bỏ).
    - Ô nhớ ứng cử viên ($\widetilde{C_t}$) sử dụng hàm tanh để tạo ra một véc-tơ các giá trị tiềm năng sẽ được thêm vào ô nhớ. Các giá trị này nằm trong khoảng [-1, 1].

$$i_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_i)$$

$$\widetilde{C_t} = tanh(W_C \cdot [h_{t-1}, x_t] + b_C$$

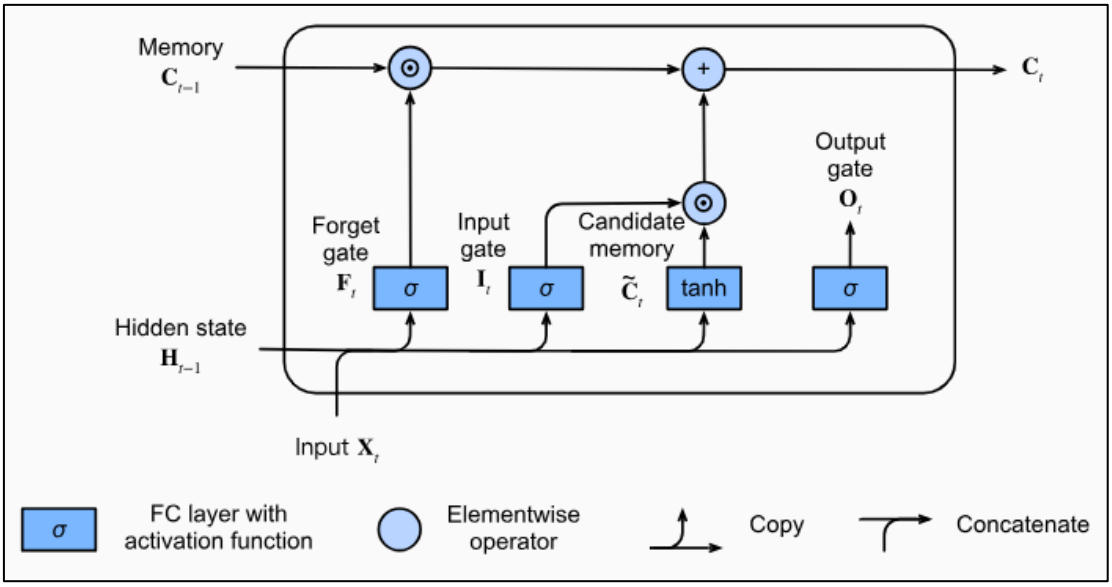
Sau đó, ô nhớ mới ($\widetilde{C_t}$) được cập nhật bằng cách nhân phần thông tin cũ được giữ lại ($f_t \odot C_{t-1}$) với phần thông tin mới được thêm vào ($i_t \odot \widetilde{C_t}$). (Ký hiệu $\odot$ là phép nhân phần tử theo phần tử - Hadamard product).

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t}$$

- Cổng đầu ra (Output Gate - o_t) Quyết định phần nào của ô nhớ hiện tại (C_t) sẽ được xuất ra làm trạng thái ẩn (ht) cho bước thời gian tiếp theo và đầu ra của ô LSTM tại thời điểm hiện tại. Cổng này cũng sử dụng hàm sigmoid.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$



Hình 2. 4: Cấu trúc bên trong một cell của LSTM - Ảnh: Internet

Cơ chế cổng của LSTM là chìa khóa để giải quyết vấn đề gradient biến mất trong RNN truyền thống. Bằng cách sử dụng ô nhớ (C_t) và các cổng điều khiển, LSTM có thể giữ lại thông tin quan trọng qua nhiều bước thời gian mà không bị suy giảm gradient [25].

Đường dẫn thông tin qua ô nhớ, nơi các phép nhân chủ yếu là với các giá trị 0 hoặc 1 (từ các cổng sigmoid), giúp duy trì gradient ổn định hơn, cho phép mô hình học được các phụ thuộc kéo dài hàng nghìn bước thời gian. Điều này giúp LSTM vượt trội hơn RNN truyền thống trong các bài toán đòi hỏi "bộ nhớ" dài hạn.

2.3.3. Mạng bộ nhớ ngắn hạn dài hạn hai chiều – BiLSTM

Mạng bộ nhớ ngắn hạn dài hạn hai chiều (Bidirectional Long Short-Term Memory - BiLSTM) là một cải tiến của mạng LSTM truyền thống, được thiết kế để xử lý dữ liệu tuần tự bằng cách sử dụng cả thông tin từ quá khứ (past) và tương lai (future) của chuỗi dữ liệu [30]. Trong khi LSTM đơn hướng (unidirectional LSTM) chỉ xử lý thông tin theo một chiều (từ trái sang phải hoặc từ quá khứ đến hiện tại) thì BiLSTM xử lý dữ liệu theo hai chiều độc lập, sau đó kết hợp các biểu diễn học được [29]. Việc học theo hai chiều này cho phép BiLSTM học các phụ thuộc phức tạp và tinh tế hơn trong dữ liệu tuần tự, dẫn đến hiệu suất được cải thiện so với các mô hình đơn hướng.

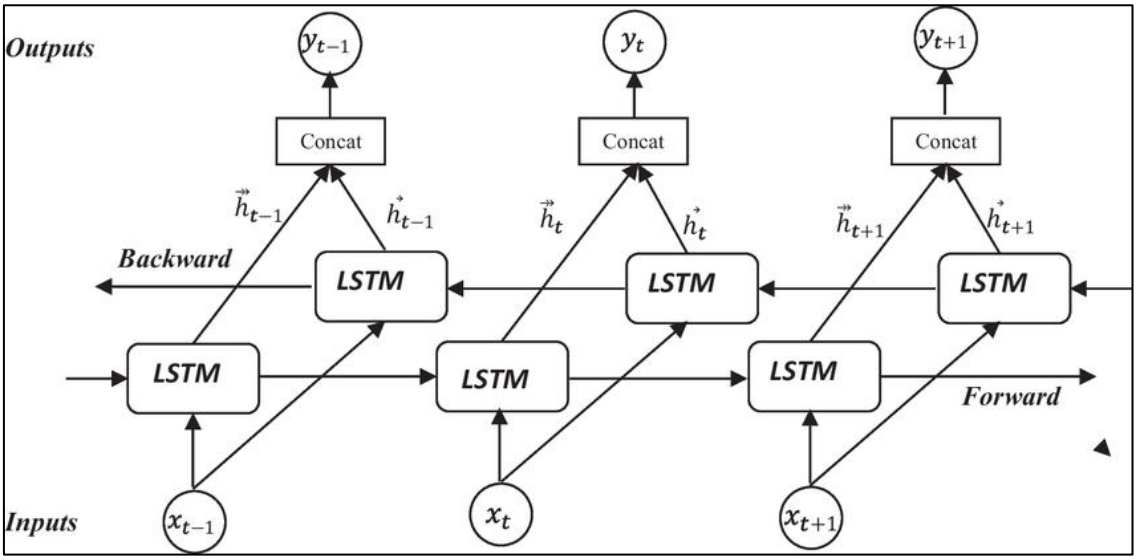
Cấu trúc của một mạng BiLSTM bao gồm hai lớp LSTM độc lập, mỗi lớp xử lý chuỗi đầu vào theo một hướng khác nhau [29]:

- Lớp LSTM tiến (Forward LSTM Layer): Xử lý chuỗi đầu vào theo thứ tự thời gian bình thường (từ thời điểm $t = 1$ đến $t = T$). Lớp này tính toán một chuỗi các trạng thái ẩn tiến ($h_{\overrightarrow{t}}$) dựa trên các đầu vào từ quá khứ.
- Lớp LSTM lùi (Backward LSTM Layer): Xử lý chuỗi đầu vào theo thứ tự thời gian ngược lại (từ thời điểm $t = T$ đến $t = 1$). Lớp này tính toán một chuỗi các trạng thái ẩn lùi ($h_{\overleftarrow{t}}$) dựa trên các đầu vào từ tương lai (tức là các phần tử sau nó trong chuỗi gốc).

Tại mỗi bước thời gian t, đầu ra cuối cùng của BiLSTM (y_t) được tạo ra bằng cách kết hợp (thường là nối - concatenation) trạng thái ẩn từ lớp tiến ($h_{\overrightarrow{t}}$) và trạng thái ẩn từ lớp lùi ($h_{\overleftarrow{t}}$).

$$y_t = [h_{\overrightarrow{t}}; h_{\overleftarrow{t}}]$$

Sự kết hợp này cho phép mạng truy cập toàn bộ ngữ cảnh của một phần tử trong chuỗi, bao gồm cả thông tin trước và sau nó, từ đó tạo ra một biểu diễn phong phú và đầy đủ hơn [29].



Hình 2. 5: Cấu trúc cơ bản của mạng BiLSTM - Ảnh: Internet

2.3.4. Ứng dụng trong xử lý dữ liệu chuỗi thời gian

Mạng nơ-ron hồi quy (RNN), đặc biệt là các biến thể như LSTM và BiLSTM, đã chứng tỏ hiệu quả vượt trội trong việc xử lý và phân tích dữ liệu chuỗi thời gian do khả năng nắm bắt các phụ thuộc động học và ngữ cảnh theo thời gian. Các ứng dụng này rất đa dạng và quan trọng trong nhiều lĩnh vực:

- Dự báo chuỗi thời gian (Time Series Forecasting): Đây là một trong những ứng dụng phổ biến nhất của các mạng nơ-ron hồi quy. Các mô hình này có khả năng học các mẫu và xu hướng tiềm ẩn trong dữ liệu chuỗi thời gian lịch sử để dự đoán các giá trị tương lai.
- Nhận dạng hoạt động (Activity Recognition): Dữ liệu chuỗi thời gian được thu thập từ các cảm biến đeo trên người (ví dụ: gia tốc kế, con quay hồi chuyển) hoặc cảm biến môi trường, RNN/LSTM/BiLSTM có thể được sử dụng để nhận dạng các hoạt động của con người (ví dụ: đi bộ, chạy, đứng, ngồi, ngã).

- Xử lý tín hiệu (Signal Processing): Các mạng hồi quy có thể phân tích và xử lý các loại tín hiệu một chiều khác như tín hiệu âm thanh (nhận dạng giọng nói, tổng hợp giọng nói), tín hiệu y sinh (ví dụ: điện tâm đồ ECG, điện não đồ EEG để phát hiện các bất thường)
- Kiểm soát chất lượng và bảo trì dự đoán: Phân tích dữ liệu cảm biến từ máy móc hoặc quy trình sản xuất để dự đoán các sự cố hoặc hỏng hóc sắp xảy ra, giúp lên kế hoạch bảo trì hiệu quả và giảm thiểu thời gian ngừng hoạt động.
- Phân tích hành vi người dùng: Phân tích chuỗi hành động của người dùng trên các nền tảng kỹ thuật số (ví dụ: lịch sử duyệt web, tương tác ứng dụng) để hiểu sở thích, dự đoán hành vi tương lai, và cá nhân hóa trải nghiệm.

2.4. Mô hình học sâu CB-LSTM đề xuất

Để tận dụng tối đa khả năng trích xuất đặc trưng không gian mạnh mẽ của Mạng nơ-ron tích chập (CNN) và khả năng học các phụ thuộc tuần tự theo thời gian của Mạng bộ nhớ ngắn hạn dài hạn hai chiều (BiLSTM), em xin đề xuất một mô hình học sâu có tên là CB-LSTM (Convolutional Bidirectional Long Short-Term Memory) cho bài toán phát hiện và cảnh báo té ngã. Mô hình này được lấy cảm hứng từ phương pháp tiếp cận trong nghiên cứu của Li et al. (2024) [9], vốn ứng dụng mô hình tương tự cho dữ liệu radar để phát hiện té ngã. Trong luận văn này, em sẽ điều chỉnh và áp dụng kiến trúc này để xử lý dữ liệu tín hiệu gia tốc từ cảm biến đeo trên người hoặc môi trường, nhằm cải thiện độ chính xác và độ tin cậy trong phát hiện té ngã.

Mô hình CB-LSTM đề xuất được thiết kế để đồng thời nắm bắt cả đặc trưng không gian (spatial features) và đặc trưng tuần tự theo thời gian (temporal sequential features) từ dữ liệu đầu vào. Cấu trúc tổng thể của mô hình bao gồm hai thành phần chính: một phần CNN để trích xuất đặc trưng không gian và một phần BiLSTM để xử lý các đặc trưng tuần tự.

2.4.1. Kiến trúc tổng thể của mô hình CB-LSTM

Kiến trúc của mô hình CB-LSTM đề xuất bao gồm các giai đoạn xử lý chính sau:

- Giai đoạn trích xuất đặc trưng (Feature Extraction): Sử dụng các lớp tích chập (Convolutional Layers) và lớp gộp (Pooling Layers) để trích xuất các đặc trưng phức tạp từ dữ liệu đầu vào.
- Giai đoạn trích xuất và học các đặc trưng tuần tự theo thời gian (Temporal Feature Extraction): Các đặc trưng của dữ liệu được trích xuất sau đó được làm phẳng (flatten) và đưa vào mạng BiLSTM để học các phụ thuộc tuần tự theo thời gian.
- Giai đoạn phân loại (Classification): Đầu ra từ mạng BiLSTM được đưa vào một lớp đầu ra để thực hiện phân loại cuối cùng.

2.4.2. Chi tiết các thành phần của mô hình

Phần CNN trong mô hình CB-LSTM chịu trách nhiệm trích xuất các đặc trưng không gian từ dữ liệu tín hiệu gia tốc. Do dữ liệu đầu vào trong đề tài này là dữ liệu gia tốc 3 trục từ cảm biến đeo tay (bộ dữ liệu DU-MD), vốn có tính chất là dữ liệu chuỗi thời gian một chiều, nên em sẽ sử dụng mạng CNN 1D (Conv1D). Điều này cho phép mô hình học các mẫu cục bộ và các đặc trưng quan trọng theo thời gian trực tiếp từ tín hiệu thô, thay vì chuyển đổi thành dạng ảnh tần phổ.

Kiến trúc cụ thể của phần CNN bao gồm các lớp sau:

- Lớp Conv1D thứ 1:
    - Conv1D(filters=128, kernel_size=5, stride=1, activation='relu', padding='same'): Lớp tích chập 1D với 128 bộ lọc, kích thước kernel là 5, bước nhảy 1, sử dụng hàm kích hoạt ReLU và padding 'same' để giữ nguyên kích thước đầu ra.
    - BatchNormalization(): Chuẩn hóa dữ liệu đầu ra từ lớp tích chập, giúp ổn định quá trình huấn luyện và tăng tốc độ hội tụ của mạng.
    - MaxPooling1D(pool_size=2): Giảm kích thước chuỗi của bản đồ đặc trưng bằng cách lấy giá trị lớn nhất trong cửa sổ có kích thước 2. Điều này giúp giảm số lượng tham số và tăng khả năng bất biến với các biến đổi nhỏ trong dữ liệu.
- Lớp Conv1D thứ 2:

    - Conv1D(filters=256, kernel_size=3, stride=2, activation='relu', padding='same'): Lớp tích chập 1D với 256 bộ lọc, kích thước kernel là 3, bước nhảy 2, sử dụng hàm kích hoạt ReLU và padding 'same'. Bước nhảy 2 giúp giảm kích thước đầu ra hiệu quả hơn.
    - BatchNormalization(): Tiếp tục chuẩn hóa đầu ra.
- Lớp Conv1D thứ 3:
    - Conv1D(filters=512, kernel_size=3, stride=1, activation='relu', padding='same'): Lớp tích chập 1D với 512 bộ lọc, kích thước kernel là 3, bước nhảy 1, sử dụng hàm kích hoạt ReLU và padding 'same'.
    - BatchNormalization(): Chuẩn hóa đầu ra.

Sau khi qua các lớp tích chập và gộp, các đặc trưng thu được chứa thông tin không gian và thời gian quan trọng về các mẫu chuyển động. Đầu ra của các đặc trưng này là một tensor 3 chiều, khớp hoàn hảo với dạng đầu vào mà BiLSTM mong đợi.

- Lớp BiLSTM: Bao gồm hai lớp LSTM hoạt động độc lập:
    - BiLSTM(128): Lớp BiLSTM với 128 đơn vị ẩn. Lớp này xử lý chuỗi đặc trưng từ cả hai hướng (tiến và lùi), cho phép nắm bắt ngữ cảnh toàn diện từ dữ liệu.
    - BatchNormalization(): Chuẩn hóa đầu ra của lớp BiLSTM để ổn định và tăng tốc quá trình huấn luyện.
    - Dropout(0.5): Áp dụng Dropout với tỷ lệ 0.5 (ngẫu nhiên tắt 50% nơ-ron) để ngăn chặn hiện tượng quá khớp (overfitting) và cải thiện khả năng tổng quát hóa của mô hình trên dữ liệu mới.

Đầu ra từ lớp BiLSTM sẽ được đưa trực tiếp vào lớp phân loại cuối cùng. Khác với một số kiến trúc CNN-RNN truyền thống thường sử dụng thêm các lớp kết nối đầy đủ (Fully Connected Layers - FC) sau BiLSTM, trong mô hình đề xuất này, em nhận thấy rằng việc sử dụng thêm các lớp FC có thể gây rối loạn, làm giảm hiệu suất của mô hình cũng như làm tăng đáng kể kích thước của mô hình. Trong thực tế huấn luyện, việc này đã xảy ra, cho thấy khả năng học đặc trưng tuần tự mạnh mẽ của BiLSTM đã là đủ để thực hiện tốt nhiệm vụ phân loại.

- Lớp Output (Đầu ra):
    - Dense(1, activation='sigmoid'): Một lớp tuyến tính với 1 nơ-ron đầu ra và hàm kích hoạt Sigmoid. Hàm Sigmoid được sử dụng cho bài toán phân loại nhị phân ("Té ngã" hoặc "Không té ngã"), đưa ra xác suất dự đoán nằm trong khoảng [0, 1].

2.4.3. Ưu điểm của mô hình CB-LSTM đề xuất

Mô hình CB-LSTM kết hợp những ưu điểm của cả CNN 1D và BiLSTM, mang lại nhiều lợi ích cho bài toán phát hiện té ngã:

- Trích xuất đặc trưng toàn diện từ dữ liệu 1D: Phần CNN 1D hiệu quả trong việc tự động trích xuất các đặc trưng cục bộ theo thời gian từ dữ liệu gia tốc 3 trục thô. Kết hợp với BiLSTM, mô hình có khả năng nắm bắt các phụ thuộc tuần tự theo thời gian từ cả quá khứ và tương lai của sự kiện, giúp mô hình hiểu sâu sắc hơn về các mẫu chuyển động phức tạp liên quan đến té ngã.
- Khả năng xử lý phụ thuộc dài hạn và ngữ cảnh đầy đủ: Nhờ kiến trúc BiLSTM, mô hình có thể giải quyết vấn đề gradient biến mất và học được các phụ thuộc dài hạn trong chuỗi dữ liệu. Việc xử lý hai chiều cho phép mạng nhận diện các giai đoạn trước té ngã và sau té ngã với ngữ cảnh đầy đủ, vốn là thách thức đối với RNN truyền thống.
- Độ chính xác và độ tin cậy cao: Bằng cách tận dụng các đặc trưng không gian và thời gian đồng thời, mô hình CB-LSTM dự kiến sẽ đạt được độ chính xác và độ tin cậy cao hơn trong việc phân biệt các sự kiện té ngã với các hoạt động sinh hoạt hàng ngày, giảm thiểu tỷ lệ báo động sai.
- Kiến trúc tinh gọn, hiệu quả: Việc loại bỏ các lớp kết nối đầy đủ bổ sung sau BiLSTM dựa trên kết quả thực nghiệm cho thấy mô hình trở nên hiệu quả hơn, tránh được tình trạng gây nhiễu hoặc giảm hiệu suất, đồng thời giảm độ phức tạp của mô hình, giúp mô hình dễ dàng triển khai hơn trên các thiết bị hạn chế phần cứng.

Tóm lại, mô hình CB-LSTM đề xuất cung cấp một khung làm việc mạnh mẽ để xử lý dữ liệu chuỗi thời gian từ cảm biến gia tốc trong hệ thống phát hiện té ngã, tận dụng khả

năng học sâu để tự động trích xuất và phân tích các đặc trưng phức tạp, từ đó đưa ra dự đoán chính xác và kịp thời.

2.5. Các bộ dữ liệu sử dụng

2.5.1. Bộ dữ liệu DU-MD

Bảng 2. 1: Mô tả bộ dữ liệu DU-MD

Tiêu chí	Mô tả chi tiết
Mô tả	Bộ dữ liệu DU-MD được phát triển bởi Đại học Duke (Duke University) với mục đích nghiên cứu phát hiện té ngã. Đây là bộ dữ liệu chính được sử dụng để huấn luyện mô hình CB-LSTM.
Phương pháp thu thập	Dữ liệu được thu thập bằng cách sử dụng cảm biến đeo ở cổ tay. Điều này là một điểm quan trọng vì nó trực tiếp phù hợp với xu hướng phát triển đồng hồ thông minh hiện nay.
Nội dung	Bộ dữ liệu bao gồm dữ liệu từ 10 đối tượng thực hiện 10 hoạt động sinh hoạt hàng ngày (ADL) khác nhau và 6 loại té ngã được kiểm soát. Các ADL bao gồm đi bộ, đứng dậy, ngồi xuống, v.v., trong khi các loại té ngã bao gồm té ngã về phía trước, phía sau, sang bên, v.v.
Định dạng dữ liệu	Dữ liệu gia tốc kế thô được cung cấp dưới dạng ba trục (x, y, z), với tốc độ lấy mẫu cụ thể 30Hz.
Lý do lựa chọn	DU-MD được chọn làm bộ dữ liệu huấn luyện chính vì nó tập trung vào việc thu thập dữ liệu gia tốc kế chỉ sử dụng cảm biến đeo ở cổ tay, điều này hoàn toàn phù hợp với xu hướng phát triển đồng hồ thông minh hiện nay và mục tiêu của đề tài. Điều này đảm bảo rằng mô hình được huấn luyện trên dữ liệu có liên quan trực tiếp đến ứng dụng thực tế.

2.5.2. Bộ dữ liệu UMAFall

Bảng 2. 2: Mô tả bộ dữ liệu DU-MD

Tiêu chí	Mô tả chi tiết
Mô tả	Bộ dữ liệu UMAFall mô tả các dấu vết chuyển động được thu thập thông qua việc mô phỏng có hệ thống một tập hợp các hoạt động sinh hoạt hàng ngày (ADL) và té ngã. Mục tiêu chính là cung cấp một tài nguyên chung để so sánh và đánh giá các thuật toán phát hiện té ngã.
Nội dung	Ban đầu với 17 tình nguyện viên, sau đó mở rộng lên 19 đối tượng thử nghiệm (bao gồm cả nam và nữ, với độ tuổi, chiều cao, cân nặng đa dạng). Bao gồm 12 loại hoạt động sinh hoạt hàng ngày (ADL) khác nhau và 3 loại té ngã được mô phỏng.
Các loại cảm biến	UMAFall bao gồm dữ liệu từ gia tốc kế, con quay hồi chuyển và từ kế, có thể từ năm vị trí khác nhau trên cơ thể người tham gia thông qua dải đàn hồi: mắt cá chân, cổ tay (phải), ngực, eo và điện thoại thông minh được đặt trong túi quần bên phải.
Vai trò trong luận văn	UMAFall đóng vai trò là một bộ dữ liệu huấn luyện và đánh giá độc lập thứ hai, củng cố thêm việc đánh giá tính tương thích và khả năng tổng quát hóa của mô hình trong các môi trường thu thập dữ liệu khác nhau.

Việc sử dụng các bộ dữ liệu công khai không chỉ đảm bảo khả năng tái tạo mà còn cho phép so sánh trực tiếp, công bằng với các nghiên cứu hiện có (mô hình học máy truyền thống và các mô hình học sâu khác). Điều này đóng góp vào tính chặt chẽ khoa học và cho phép luận văn định vị rõ ràng những đóng góp của mình trong bối cảnh nghiên cứu rộng lớn hơn.

2.6. Các chỉ số đánh giá hiệu năng

Để đánh giá toàn diện hiệu suất của một mô hình phân loại nhị phân, đặc biệt trong một lĩnh vực nhạy cảm như phát hiện té ngã, nơi các kết quả dương tính giả và âm tính giả có những hậu quả đáng kể, việc sử dụng một bộ chỉ số đánh giá toàn diện là rất quan trọng.

Ma trận nhầm lẫn (Confusion Matrix): Ma trận nhầm lẫn là nền tảng cho tất cả các chỉ số đánh giá tiếp theo. Nó tóm tắt hiệu suất của một thuật toán phân loại trên một tập dữ liệu thử nghiệm, hiển thị số lượng các trường hợp được phân loại đúng và sai:

- True Positives (TP): Số lượng té ngã thực tế được mô hình dự đoán là té ngã.
- True Negatives (TN): Số lượng không té ngã thực tế được mô hình dự đoán là không té ngã.
- False Positives (FP): Số lượng không té ngã thực tế bị mô hình dự đoán sai là té ngã (báo động giả).
- False Negatives (FN): Số lượng té ngã thực tế bị mô hình dự đoán sai là không té ngã (bỏ sót té ngã).

Precision (Độ chính xác - Positive Predictive Value):

- Công thức: $\text{Precision} = \frac{TP}{TP+FP}$

- Giải thích: Tỷ lệ các sự kiện té ngã được xác định đúng trong số tất cả các trường hợp được dự đoán là té ngã. Precision cao có nghĩa là ít báo động giả hơn, điều này rất quan trọng để ngăn ngừa sự mệt mỏi do cảnh báo đối với người chăm sóc/nhân viên y tế.

Recall (Độ nhạy - Sensitivity/True Positive Rate):

- Công thức: $\text{Recall} = \frac{TP}{TP+FN}$

- Giải thích: Tỷ lệ các sự kiện té ngã thực tế được hệ thống xác định đúng. Recall cao có nghĩa là ít té ngã bị bỏ sót hơn, điều này tối quan trọng đối với sự an toàn của bệnh nhân

F1-score:

- Công thức: $$\mathrm{F1-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- Giải thích: Là giá trị trung bình điều hòa của Precision và Recall. Nó cung cấp một thước đo cân bằng, đặc biệt hữu ích khi có sự phân bố lớp không đồng đều (ví dụ: té ngã là các sự kiện hiếm so với các hoạt động sinh hoạt hàng ngày).

Accuracy (Độ chính xác tổng thể):

- Công thức: $$\mathrm{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Giải thích: Tỷ lệ tổng thể các trường hợp được phân loại đúng. Mặc dù trực quan, cần lưu ý rằng Accuracy có thể gây hiểu lầm trong các bộ dữ liệu mất cân bằng (nơi độ chính xác cao có thể đạt được bằng cách đơn giản phân loại mọi thứ là lớp đa số).

Việc nhấn mạnh vào Precision và Recall, thay vì chỉ Accuracy, cho thấy sự hiểu biết sâu sắc về các tác động trong thế giới thực của việc phát hiện té ngã. Trong lĩnh vực này, chi phí của một kết quả âm tính giả (bỏ sót té ngã) là cực kỳ cao, trong khi một kết quả dương tính giả (báo động sai) dẫn đến sự mệt mỏi do cảnh báo. Điều này hàm ý rằng việc tối ưu hóa cân bằng các chỉ số này là rất quan trọng đối với tính hữu ích thực tế và sự tin tưởng của người dùng. Sự đánh đổi giữa Precision và Recall là một thách thức cơ bản trong phân loại mất cân bằng. Việc thảo luận về sự đánh đổi này trong bối cảnh phát hiện té ngã (ví dụ: ưu tiên Recall vì an toàn, nhưng quản lý Precision để tránh cảnh báo phiền toái) sẽ thể hiện sự hiểu biết tinh vi về vấn đề. Điều này hàm ý rằng việc tối ưu hóa mô hình có thể liên quan đến việc tìm ra một điểm hoạt động tối ưu trên đường cong Precision-Recall hơn là chỉ đơn giản là tối đa hóa một chỉ số duy nhất.

2.7 Các Công nghệ nền tảng của Hệ thống

2.7.1. Internet vạn vật (IoT)

Internet vạn vật (IoT) là một mạng lưới các thiết bị vật lý, phương tiện, thiết bị gia dụng và các vật dụng khác được nhúng cảm biến, phần mềm và các công nghệ khác cho phép chúng kết nối và trao đổi dữ liệu với các thiết bị và hệ thống khác qua internet [6]. Trong ngữ cảnh của hệ thống phát hiện té ngã, các thiết bị đeo như đồng hồ thông minh hoặc các bo mạch nhúng như Orange Pi, với cảm biến gia tốc kế tích hợp, chính là các "vật" trong IoT.

Các thành phần chính của IoT trong hệ thống:

- Thiết bị (Things): Là các cảm biến gia tốc kế đeo ở cổ tay (hoặc thiết bị Orange Pi mô phỏng) thu thập dữ liệu về chuyển động theo thời gian thực [6].
- Kết nối (Connectivity): Các giao thức truyền thông như MQTT (Message Queuing Telemetry Transport) được sử dụng để thiết bị Orange Pi gửi dữ liệu lên đám mây. MQTT là một giao thức nhẹ, hiệu quả, lý tưởng cho các thiết bị có tài nguyên hạn chế và môi trường mạng không ổn định [14].
- Nền tảng IoT (IoT Platform): Các nền tảng như AWS IoT Core cung cấp cơ sở hạ tầng để quản lý kết nối thiết bị, thu thập, định tuyến và xử lý dữ liệu từ hàng triệu thiết bị IoT [6].

2.7.2. Dịch vụ đám mây (Cloud Services)

Dịch vụ đám mây cung cấp các tài nguyên điện toán theo yêu cầu từ các ứng dụng đến máy chủ, lưu trữ, mạng và cơ sở dữ liệu qua internet với mô hình trả tiền theo mức sử dụng [14]. Trong hệ thống phát hiện té ngã, dịch vụ đám mây đóng vai trò là trung tâm xử lý, lưu trữ và điều phối thông báo.

Vai trò của Dịch vụ đám mây trong hệ thống:

- Lưu trữ dữ liệu: Dữ liệu cảm biến và các sự kiện té ngã được gửi từ thiết bị sẽ được lưu trữ trên cơ sở dữ liệu đám mây (ví dụ: Amazon DynamoDB, AWS S3) [14].

- Xử lý và phân tích: Các dịch vụ tính toán phi máy chủ (Serverless Compute) như AWS Lambda có thể được kích hoạt để xử lý dữ liệu ngay khi nhận được từ thiết bị IoT, thực hiện các tác vụ như ghi dữ liệu và kích hoạt cảnh báo [14].
- Điều phối thông báo: Các dịch vụ như Amazon Simple Notification Service (SNS) được sử dụng để gửi thông báo đa kênh (email, thông báo đẩy đến ứng dụng di động) đến người thân hoặc nhân viên y tế được ủy quyền [14].
- Cung cấp API cho ứng dụng: API Gateway cho phép các ứng dụng di động giao tiếp an toàn với các dịch vụ đám mây để truy xuất dữ liệu sự kiện hoặc cấu hình hệ thống [14].

2.7.3. Nền tảng phát triển ứng dụng di động Flutter

Flutter là một bộ công cụ phát triển giao diện người dùng (UI toolkit) mã nguồn mở được phát triển bởi Google để xây dựng các ứng dụng di động, web và desktop từ một codebase duy nhất [34]. Trong ngữ cảnh của hệ thống phát hiện và cảnh báo té ngã, Flutter được lựa chọn để phát triển ứng dụng di động vì những ưu điểm sau:

- Đa nền tảng: Viết mã một lần, triển khai trên cả Android và iOS, tiết kiệm thời gian và chi phí phát triển [34].
- Hiệu suất cao: Biên dịch thành mã máy gốc (native code), đảm bảo ứng dụng chạy mượt mà và phản hồi nhanh chóng, quan trọng cho cảnh báo thời gian thực [34].
- Giao diện linh hoạt: Cung cấp bộ widget phong phú và khả năng tùy chỉnh cao, giúp xây dựng giao diện trực quan và thân thiện, cải thiện trải nghiệm người dùng [34].
- Tăng tốc phát triển: Tính năng hot-reload/hot-restart giúp xem ngay lập tức các thay đổi, tăng tốc quá trình phát triển [34].

Sự kết hợp giữa IoT, dịch vụ đám mây và Flutter tạo nên một hệ thống mạnh mẽ, có khả năng mở rộng và đáng tin cậy. IoT thu thập dữ liệu, trong khi đám mây xử lý, lưu trữ và thông báo tức thì, và Flutter cung cấp giao diện người dùng hiệu quả, giúp đảm bảo an toàn tối đa cho người cao tuổi.

Chương 3. KẾT QUẢ THỰC NGHIỆM

3.1. Giới thiệu và phát biểu bài toán

3.1.1. Phát biểu bài toán

Như ở các chương trước, em đã trình bày mục đích và hướng nghiên cứu cho bài toán của mình. Trong chương này, từ những nền tảng lý thuyết mà em đã xây dựng từ trước, tiến hành nghiên cứu, thử nghiệm và phân tích những hiệu quả mà phương pháp học sâu mang lại trong việc xử lý tín hiệu gia tốc từ cảm biến đeo ở cổ tay được thu thập từ bộ dữ liệu DU-MD cho hệ thống phát hiện và cảnh báo té ngã. Vì giới hạn độ lớn của bộ dữ liệu (chỉ có 34 đối tượng tại thời điểm tải bộ dữ liệu so với mong muốn 50 đối tượng của nhóm tác giả [10]) và để đơn giản hoá bài toán tập trung vào phát hiện hành vi té ngã (Fall) hay hoạt động hằng ngày (Activities of Daily Living - ADL), nghĩa là bài toán phân loại nhị phân. Điều này phần nào sẽ giúp cho mô hình học sâu có hiệu năng tốt hơn, giảm tình trạng báo động giả, phù hợp với nhu cầu thực tế. Chính vì vậy nên đối với tất cả các bộ dữ liệu, nhãn của các hoạt động cụ thể sẽ được gộp lại thành hai nhãn là 0 và 1 tương ứng với ADL và Fall.

Vì các bộ dữ liệu được nghiên cứu và tạo ra với các mục đích khác nhau nên không có tính đồng nhất ở số lượng, vị trí cảm biến được sử dụng, đặc biệt là tần số ghi mẫu dữ liệu. Nó ảnh hưởng trực tiếp đến quá trình tiền xử lý cho mô hình nên em đã tiến hành đồng nhất việc chọn cửa sổ thời gian (window_size) cho mô hình là 90, đảm bảo phù hợp với cả ba bộ dữ liệu được chọn, hạn chế việc mất mát thông tin và tăng kích thước dữ liệu một cách tự nhiên nhất có thể cho quá trình huấn luyện. Khi này ở bộ dữ liệu DU-MD kích thước thời gian cho một cửa sổ sẽ tương ứng với khoảng 3 giây, đối với bộ UMAFall, đảm bảo dữ liệu theo thời gian sẽ liên tục từ trước khi ngã, trong lúc ngã và sau khi ngã, giúp mô hình tổng quát hoá dễ hơn. Và để đánh giá toàn diện hiệu năng của mô hình, em đã quyết định đánh giá trên hai trường hợp gồm: huấn luyện mô hình với từng bộ dữ liệu rồi đánh giá mô hình trên tập test của từng bộ dữ liệu; so sánh, đánh giá kết quả với các mô hình học sâu riêng lẻ và học máy của từng bộ dữ liệu.

Hệ thống phát hiện và cảnh báo té ngã sẽ được triển khai với ba thành phần được kết nối với nhau qua mạng lưới IoT bao gồm: bo mạch cảm biến đeo hoạt động như một thiết bị đeo tay thông minh chịu trách nhiệm thu thập dữ liệu và phân tích hoạt động của người dùng với kết quả đầu ra là té ngã hoặc hoạt động hằng ngày theo thời gian thực; dịch vụ đám mây (AWS) chịu trách nhiệm là trung tâm xử lý của hệ thống, nhận kết quả và dữ liệu từ thiết bị đeo, lưu trữ và gửi báo đến ứng dụng theo dõi của người thân và nhân viên y tế; ứng dụng mobile app có chức năng theo dõi và nhận cảnh báo về tình trạng của người được theo dõi.

3.1.2. Môi trường phát triển và cài đặt

Quá trình phát triển và kiểm thử mô hình CB-LSTM được thực hiện trên cùng một môi trường được trang bị đầy đủ để xử lý các tác vụ học sâu.

- Phần cứng: Hệ thống được sử dụng có CPU Intel Core i7–8750H, GPU NVIDIA GeForce GTX 1050Ti (4GB VRAM), 16GB RAM và ổ cứng HDD 1TB.
- Phần mềm: Hệ điều hành Windows 11, version 23H2. Ngôn ngữ lập trình Python. Môi trường Python 3.10.10. Các thư viện bao gồm: Tensorflow 2.10.0/Keras; Numpy 1.21.6; Pandas 1.3.5; Scikit-learn 1.6.1; Matplotlib 3.8.4; Seaborn 0.13.2.

Quá trình triển khai, thử nghiệm mô hình sử dụng bo mạch phát triển Orange Pi Zero 2W với cấu hình vi xử lý Allwinner H618, 4GB RAM LPDDR4.

Hệ thống được triển khai đầy đủ sử dụng dịch vụ đám mây của Amazon Web Services (AWS), ứng dụng mobile được viết trên nền tảng Flutter (Dart).

3.2. Quy trình thu thập và xử lý dữ liệu

3.2.1. Quy trình thu thập và xử lý bộ dữ liệu DU-MD

Bộ dữ liệu được tổ chức theo cấu trúc gồm có 34 thư mục, đại diện cho 34 đối tượng tham gia thử nghiệm. Mỗi thư mục có 10 thư mục con, đại diện cho 10 hành động bao gồm: 7 hoạt động hằng ngày (walking, sitting, sleeping, jogging, staircase up, staircare down, standing, falling unconsciousness, falling heart attack, falling slipping). Mỗi hành động có 10 file txt chứa 101 dòng dữ liệu gia tốc kế 3 trục (x, y, z). Quá trình xử lý dữ liệu bao gồm

việc load lần lượt dữ liệu, làm sạch dữ liệu (loại bỏ các giá trị thiếu, giá trị ngoại biên), gán nhãn, tạo cửa sổ thời gian với window_size = 90, stride = 10.

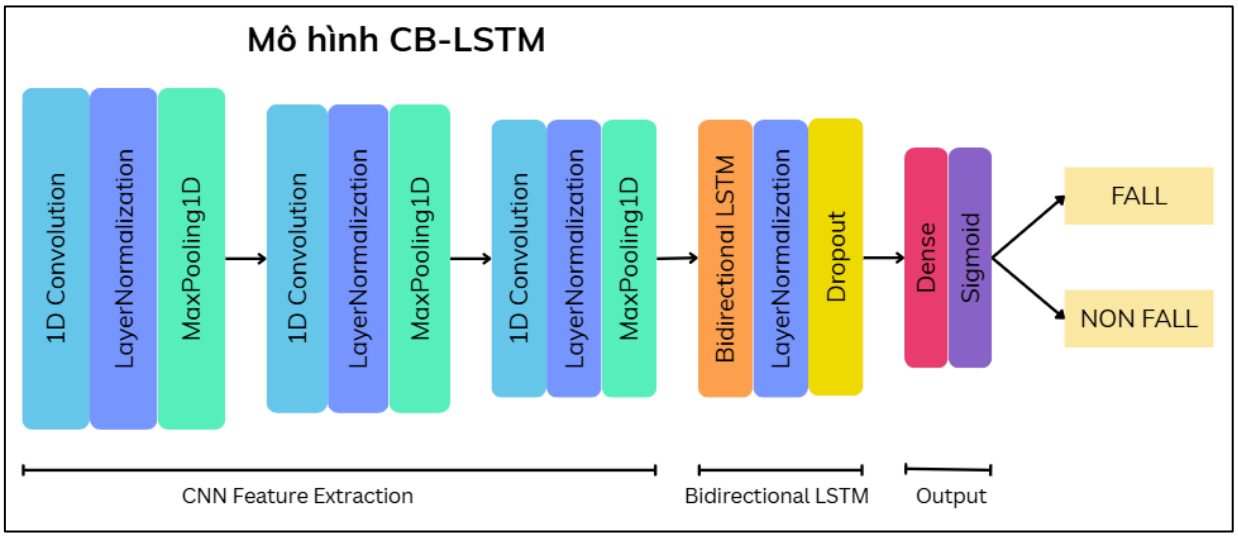
3.2.2. Quy trình thu thập và xử lý bộ dữ liệu bộ dữ liệu UMAFall

Bộ dữ liệu được tổ chức thành 2 thư mục lớn là ADL và Fall. Trong thư mục ADL chứa 538 file, trong thư mục Fall chứa 208 file, các file csv được đặt tên theo cấu trúc: "UMAFall_Subject_01_ADL_ClappingHands_1_2017-04-14_23-38-23.csv". Trong đó, chúng ta dễ dàng nhận thấy tên hành động trong tên của file để xử lý gán nhãn cho dữ liệu. Trong mỗi file csv, 40 dòng đầu tiên là các mô tả về file dữ liệu, dòng 41 chứa tên các cột (TimeStamp, Sample No, X-Axis, Y-Axis, Z-Axis, Sensor Type, Sensor ID). Tiến hành load dữ liệu, làm sạch dữ liệu (loại bỏ các giá trị thiếu, giá trị ngoại biên), trích xuất dữ liệu cảm biến gia tốc đeo ở cổ tay, gán nhãn, tạo cửa sổ thời gian với window_size = 90, stride = 10.

3.3. Huấn luyện và tối ưu hoá mô hình

3.3.1. Cấu hình mô hình

Mô hình CB-LSTM được sử dụng giống nhau cho cả ba bộ dữ liệu, chi tiết mô hình đã được trình bày ở chương 2, mục 2.4. Sau đây em xin trình bày lại chi tiết kiến trúc mô hình CB-LSTM bằng hình 3.1:



Hình 3. 1: Hình minh hoạ mô hình CB-LSTM

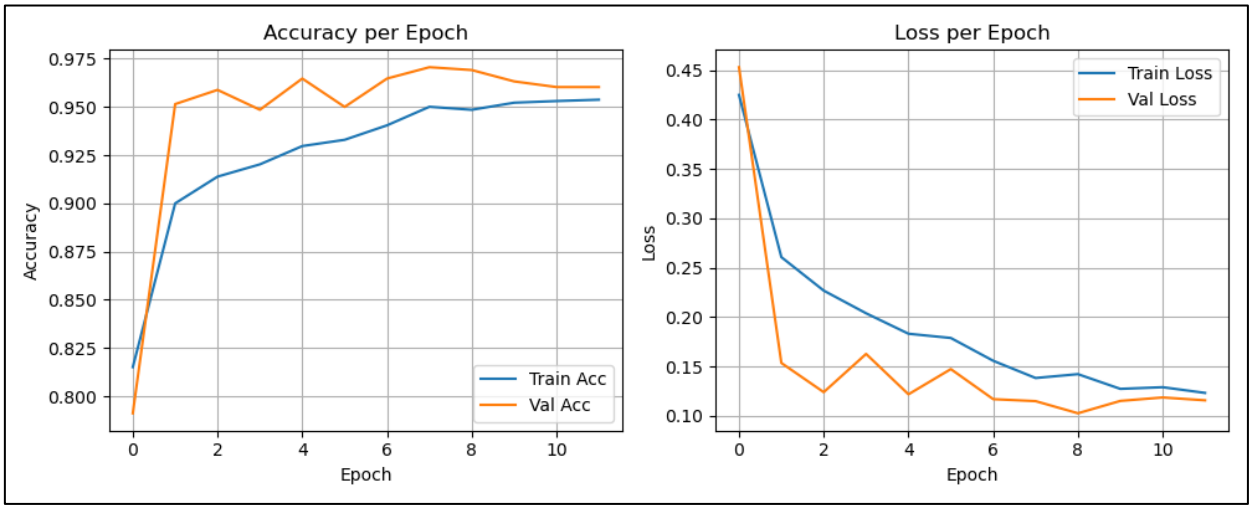
Quá trình huấn luyện:

- Class weight: Sử dụng thêm phương pháp class weight cho mô hình để xác định trọng số của lớp giúp mô hình học hỏi từ dữ liệu bị mất cân bằng.
- Hàm mất mát: Hàm mất mát Entropy chéo nhị phân (Binary Cross-Entropy) được sử dụng
- Bộ tối ưu hoá: Bộ tối ưu hoá Adam được chọn với tốc độ học ban đầu (learning_rate) được chọn là 0.001
- Kích thước lô và số epoch: Mô hình được huấn luyện với kích thước lô (batch_size) là 32 và số epoch là 50
- Dừng sớm: Cơ chế dừng sớm (Early Stopping) được triển khai để theo dõi hiệu suất trên tập kiểm định. Quá trình huấn luyện sẽ dừng lại nếu hiệu suất trên tập kiểm định không cải thiện sau 3 epoch (patience = 3), giúp ngăn chặn hiện tượng quá khớp (overfitting).

3.3.2. Kết quả huấn luyện

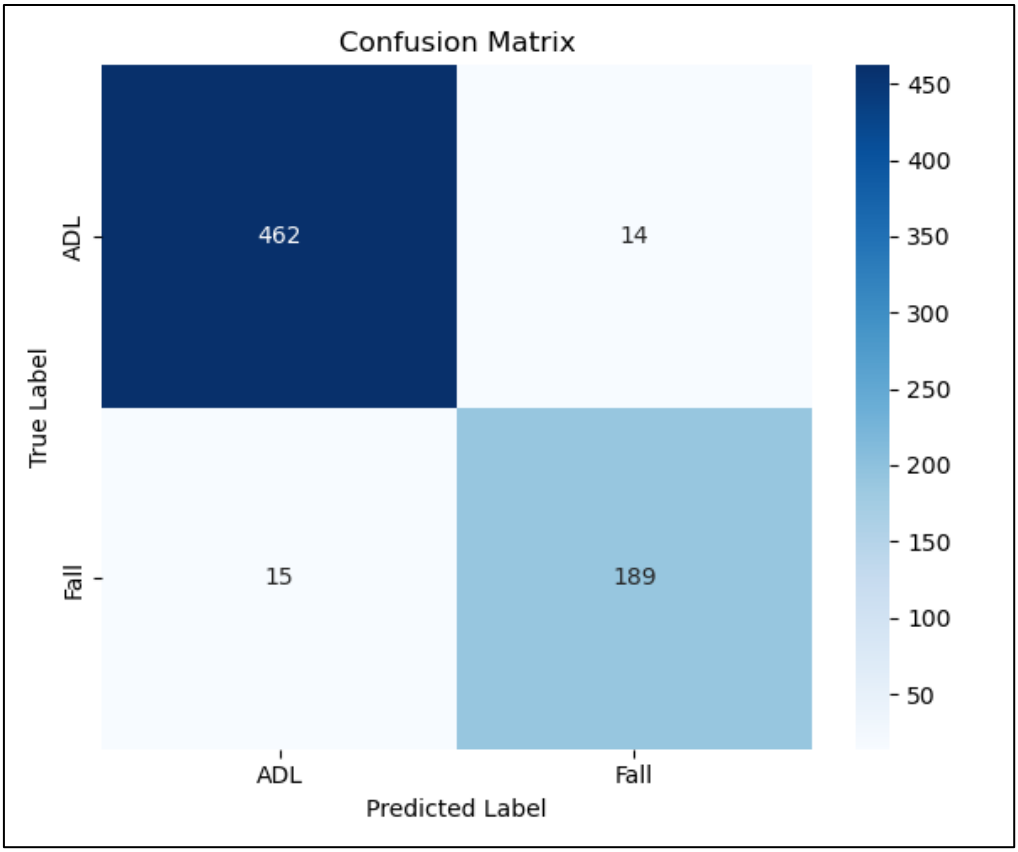
3.3.2.1. Kết quả huấn luyện với bộ DU-MD

Biểu đồ độ chính xác (accuracy per epoch) và hàm mất mát (loss per epoch) qua từng epoch trong quá trình huấn luyện mô hình CB-LSTM trên bộ dữ liệu DU-MD được trình bày trong hình 3.1, kết quả biểu đồ ma trận nhầm lẫn (confusion matrix) trên tập dữ liệu test được trình bày trong hình 3.2.



Hình 3. 2: Kết quả quá trình huấn luyện mô hình với bộ dữ liệu DU-MD

Biểu đồ độ chính xác cho thấy độ chính xác trên tập huấn luyện (train set) tăng dần đều và đạt khoảng 95% sau khoảng 11 epoch, độ chính xác trên tập validation cũng duy trì ở mức ổn định (≥ 95%) từ epoch số 2 trở đi cho thấy mô hình tổng quát hoá tốt. Biểu đồ mất mát cho thấy cả train và validation đều giảm rõ rệt, mô hình học đặc trưng dữ liệu hiệu quả ở năm epoch đầu và duy trì khá ổn định sau epoch số 5 cho thấy mô hình đã hội tụ. Việc dùng class weight và augment dữ liệu Fall đã giúp mô hình không chỉ học tốt ADL mà còn cải thiện nhận diện Fall.

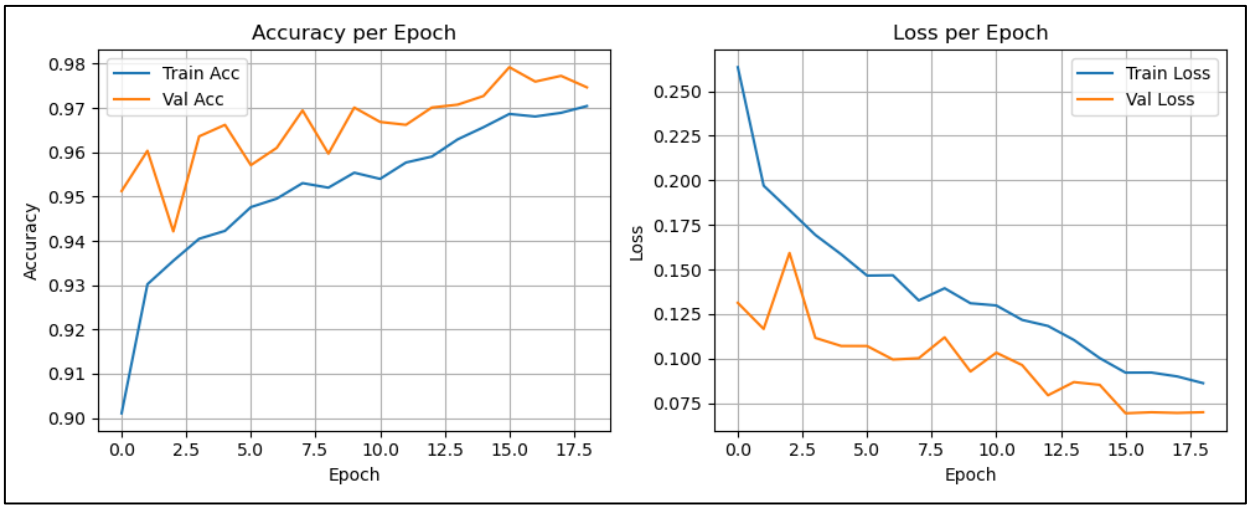


Hình 3. 3: Kết quả biểu đồ ma trận nhầm lẫn trên tập test

Kết quả ma trận nhầm lẫn phản ánh độ chính xác cao ở cả hai lớp với độ chính xác tổng thể đạt khoảng 95.74%, mô hình có khả năng phát hiện té ngã (Fall) tốt. Precision Fall đạt ≈ 93%, recall Fall đạt ≈ 93% cho thấy mô hình có tỉ lệ cảnh báo giả thấp. Chỉ có 14 mẫu ADL bị nhầm là Fall và 15 mẫu Fall bị nhầm ADL cho thấy tỉ lệ nhầm lẫn khá thấp.

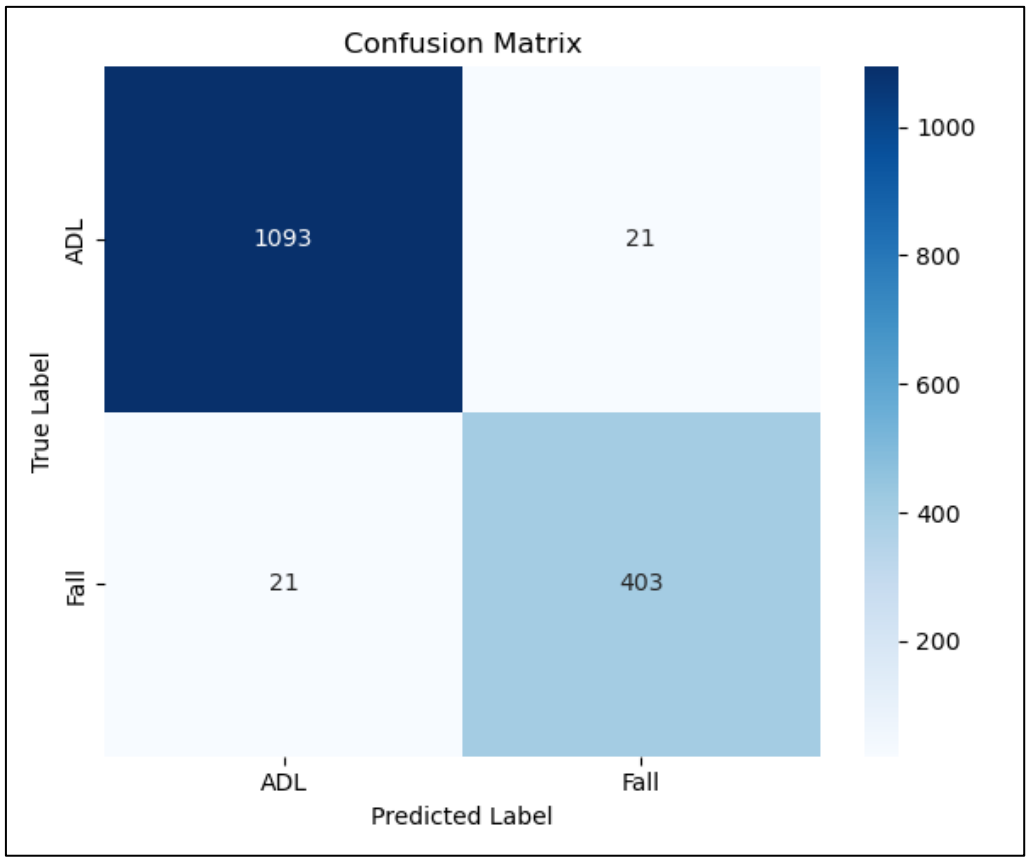
3.3.2.2. Kết quả huấn luyện với bộ UMAFall

Biểu đồ độ chính xác (accuracy per epoch) và hàm mất mát (loss per epoch) qua từng epoch trong quá trình huấn luyện mô hình CB-LSTM trên bộ dữ liệu UMAFall được trình bày trong hình 3.3, kết quả biểu đồ ma trận nhầm lẫn (confusion matrix) trên tập dữ liệu test được trình bày trong hình 3.4.



Hình 3. 4: Kết quả quá trình huấn luyện mô hình với bộ dữ liệu UMAFall

Biểu đồ độ chính xác cho thấy độ chính xác trên tập huấn luyện (train set) và tập validation đều tăng trong các epoch đầu và dần ổn định sau epoch số 5, accuracy trên tập validation luôn cao hơn hoặc tương đương với tập train, điều này cho thấy mô hình có khả năng tổng quát tốt. Biểu đồ mất mát cho thấy cả train và validation đều giảm đều trong các epoch, phản ánh quá trình tối ưu diễn ra hiệu quả. Loss validation nhỏ hơn loss train từ rất sớm và ổn định về sau cho thấy mô hình học được các đặc trưng quan trọng, không chỉ học thuộc dữ liệu.



Hình 3. 5: Kết quả biểu đồ ma trận nhầm lẫn trên tập test

Trên tập kiểm tra, confusion matrix cho thấy mô hình phân biệt ADL và Fall rất tốt, cả false positive và false negative đều ở mức rất thấp. Precision và recall cho class "Fall" đạt ≈ 95%, giúp giảm cảnh báo giả và tăng độ nhạy khi phát hiện ngã thật.

3.4. Đánh giá hiệu năng mô hình trên các bộ dữ liệu

3.4.1. Hiệu năng trên tập kiểm thử các bộ dữ liệu

Kết quả đánh giá trên tập dữ liệu test của mô hình CB-LSTM so với các mô hình học sâu CNN1D, LSTM, BiLSTM riêng lẻ được huấn luyện trên bộ dữ liệu DU-MD và UMAFall được trình bày lần lượt trong bảng 3.1 và 3.2.

Bảng 3. 1: So sánh kết quả các mô hình học sâu trên bộ dữ liệu DU-MD

Chỉ số (%) / Mô hình	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F1-score**
CB-LSTM	95.74	93	93	97	93
CNN 1D	92.65	89	86	95	88
LSTM	92.79	88	88	95	88
BiLSTM	93.82	90	90	96	90

Mô hình CB-LSTM cho kết quả tốt hơn các mô hình CNN1D, LSTM, và BiLSTM đơn lẻ trên toàn bộ các chỉ số đánh giá với độ chính xác (95.74%). Độ nhạy (96%) cho thấy khả năng phát hiện tốt té ngã thực sự (tránh bỏ sót), precision đạt (93%) giúp giảm cảnh báo giả là yếu tố quan trọng mà chúng ta hướng đến và F1-score (94%) cho thấy sự cân bằng tốt giữa khả năng phát hiện đúng té ngã và giảm thiểu báo động giả của mô hình CB-LSTM. Sự kết hợp giữa các lớp CNN 1D và BiLSTM giúp mô hình vừa học được đặc trưng cục bộ từ tín hiệu gia tốc, vừa nắm bắt được mối quan hệ chuỗi thời gian – điều này giải thích cho khả năng phân loại hiệu quả giữa hoạt động bình thường và té ngã.

Bảng 3. 2: So sánh kết quả các mô hình học sâu trên bộ dữ liệu UMAFall

Chỉ số (%) / Mô hình	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F1-score**
CB-LSTM	97.27	95	95	98	95
CNN 1D	96.68	92	96	97	94
LSTM	95.32	92	91	97	91
BiLSTM	96.81	93	95	97	94

Đối với bộ dữ liệu UMAFall, hầu hết các chỉ số của các mô hình học sâu đều ở mức cao trên 90%, có thể giải thích một chút cho điều này là vì lượng dữ liệu từ bộ dữ liệu UMAFall nhiều hơn một chút so với bộ dữ liệu DU-MD dẫn đến các mô hình học sâu có

điều kiện học tốt hơn. Sự chênh lệch không quá lớn giữa các mô hình học sâu đơn lẻ so với mô hình kết hợp phần nào chứng tỏ được sức mạnh trích xuất đặc trưng của CNN 1D và khả năng học các đặc trưng hiệu quả theo thời gian của LSTM hay BiLSTM. Việc kết hợp sức mạnh của hai mô hình đem lại hiệu quả khi precision đạt 95% giúp giảm đáng kể cảnh báo giả, độ nhạy (sensitivity), độ đặc hiệu (specificity) lần lượt đạt 95% và 98% cho thấy CB-LSTM tối ưu ở cả hai khía cạnh: ít bỏ sót sự kiện quan trọng (Fall) và ít cảnh báo sai. Điều này một lần nữa được thể hiện khi F1-Score đạt 95%, thể hiện sự cân bằng lý tưởng giữa precision và recall.

3.4.2. So sánh hiệu năng với các mô hình học máy truyền thống

Như đã làm rõ ở các mục trước, trong bài toán này em chỉ sử dụng phân loại nhị phân để đảm bảo tập trung vào việc phát hiện té ngã, giảm thiểu các tình huống cảnh báo giả gây phiền phức và có tính ứng dụng hơn. Kết quả đánh giá mô hình CB-LSTM so với các mô hình học máy SVM, Random Forest và KNN được huấn luyện trên bộ dữ liệu DU-MD và UMAFall được trình bày lần lượt trong bảng 3.3 và bảng 3.4.

Bảng 3. 3: So sánh kết quả các mô hình học sâu và học máy trên bộ dữ liệu DU-MD

Chỉ số (%) / Mô hình	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F1-score**
CB-LSTM	95.74	93	93	97	93
SVM	93.8	94	84	97	89
Random Forest	92.6	90	84	96	87
KNN	93.5	90	87	96	89

Bảng 3. 4: So sánh kết quả các mô hình học sâu và học máy trên bộ dữ liệu UMAFall

Chỉ số (%) / Mô hình	Accuracy	Precision	Sensitivity	Specificity	F1-score
CB-LSTM	97.27	95	95	98	95
SVM	94.7	93	86	97	89
Random Forest	92.5	90	80	97	85
KNN	95	92	88	97	90

Kết quả cũng cho thấy rõ sự phụ thuộc của các phương pháp sử dụng mô hình học máy vào kinh nghiệm của người xử lý và trích xuất các đặc trưng của dữ liệu. Mặc dù đã bám sát vào quy trình gần tương tự của tác giả S. Saha và các cộng sự [14] trong quá trình tiền xử lý và trích xuất đặc trưng của bộ dữ liệu DU-MD nhưng kết quả cho ra của các mô hình học máy vẫn phụ thuộc vào kinh nghiệm, trình độ của em nên có thể chưa đạt được hiệu suất tối đa. Tương tự với bộ UMAFall khi chỉ sử dụng dữ liệu từ cảm biến gia tốc đeo ở cổ tay và sự phụ thuộc vào quá trình trích xuất đặc trưng. Kết quả cũng cho thấy sự chênh lệch không đáng kể giữa tất cả các chỉ số đánh giá, điều này cho thấy việc sử dụng các mô hình học sâu sẽ loại bỏ bước nghiên cứu và trích xuất đặc trưng thủ công giúp tiết kiệm thời gian và tài nguyên mà vẫn đem lại kết quả từ ngang bằng đến vượt trội so với các mô hình học máy truyền thống.

3.5. Phân tích kết quả và thảo luận

Các kết quả thực nghiệm đã chứng minh rõ ràng hiệu suất vượt trội của mô hình CB-LSTM trong việc phát hiện té ngã. Với Precision và Recall cao trên cả hai bộ dữ liệu, mô hình cho thấy khả năng cân bằng giữa việc giảm thiểu báo động giả và giảm thiểu bỏ sót các sự kiện té ngã quan trọng. Trong ứng dụng thực tế, Recall thường được ưu tiên cao hơn Precision một chút vì chi phí của việc bỏ sót một cú ngã (có thể dẫn đến chấn thương nghiêm trọng, thậm chí tử vong) cao hơn chi phí của một báo động giả (gây phiền toái). Tuy nhiên, một Precision quá thấp cũng dẫn đến sự mệt mỏi do cảnh báo giả, làm giảm độ tin cậy của hệ thống. Các kết quả của Precision được trình bày ở bảng 3.1, 3.2 cho thấy các

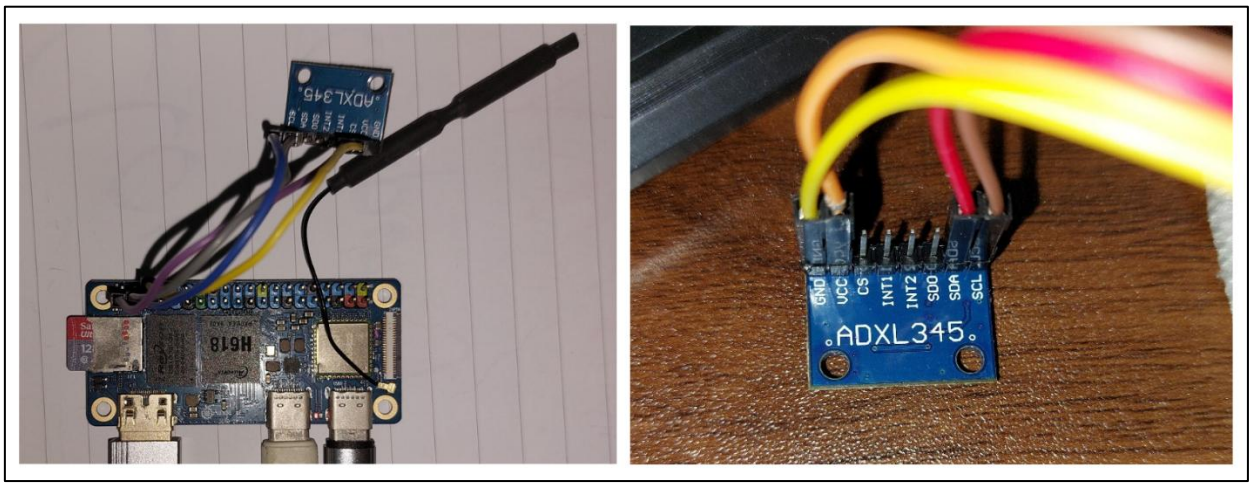
mô hình học sâu đều ở mức tốt (trên 85%), mô hình CB-LSTM thể hiện sự vượt trội khi Precision đạt mức 93%, 95% lần lượt ở hai bộ dữ liệu DU-MD, UMAFall. Chỉ số F1-score, là trung bình điều hoà của Precision và Recall, cung cấp một thước đo cân bằng và đáng tin cậy về hiệu suất tổng thể của mô hình.

Các kết quả so sánh với các mô hình học máy truyền thống trong bảng 3.3, 3.4 đã cho thấy lợi thế của các mô hình học sâu so với các phương pháp học máy truyền thống. Mô hình CB-LSTM, bằng cách tận dụng khả năng trích xuất đặc trưng tự động hiệu quả của mô hình CNN 1D từ dữ liệu thô và khả năng học các đặc trưng phức tạp theo thời gian của BiLSTM đã đạt được hiệu suất tốt hơn mà không cần đến quá trình trích xuất đặc trưng thủ công tốn kém, tiền xử lý dữ liệu thô phức tạp hoặc phụ thuộc vào người thực hiện hay thiết bị thu thập dữ liệu cụ thể. Hiệu suất cao của mô hình được duy trì trên bộ dữ liệu UMAFall, mặc dù bộ dữ liệu này ngay từ đầu không tập trung vào dữ liệu cảm biến ở cổ tay. Nó cho thấy mô hình có tính tổng quát, không quá phụ thuộc vào một bộ dữ liệu được phát triển riêng cho nó mà có thể tận dụng những bộ dữ liệu có tính tương đồng, có sử dụng cảm biến đeo ở cổ tay. Những điều này cũng cố lập luận rằng học sâu cung cấp một giải pháp hiệu quả, tiết kiệm tài nguyên, chi phí phát triển và có khả năng mở rộng.

Hiệu suất vượt trội và toàn diện của CB-LSTM: Mô hình CB-LSTM đề xuất của em đạt độ chính xác tổng thể (Accuracy) cao nhất là 95.74% và F1-score 93% cho nhiệm vụ phát hiện té ngã nhị phân. Mặc dù nghiên cứu "Feature Extraction, Performance Analysis and System Design Using the DU Mobility Dataset" của tác giả S. Saha và các cộng sự [14] có đề cập đến độ chính xác phát hiện té ngã tối đa 97%, mô hình của em vẫn thể hiện hiệu suất rất cạnh tranh và đáng tin cậy. Quan trọng hơn, em cung cấp một bộ chỉ số đánh giá toàn diện hơn (Precision, Sensitivity/Recall, Specificity, F1-score) cho phép đánh giá chi tiết về khả năng của mô hình. Trong các ứng dụng phát hiện té ngã, việc cân bằng giữa Precision (giảm báo động giả) và Recall (giảm bỏ sót té ngã) là cực kỳ quan trọng. F1-score của em đạt 93%, với Precision và Recall đều ở mức 93%, thể hiện sự cân bằng xuất sắc giữa hai yếu tố này, đảm bảo cả sự an toàn cho người dùng (ít bỏ sót té ngã) và sự tiện lợi cho người chăm sóc (ít báo động giả), điều mà chỉ một con số Accuracy đơn thuần khó có thể thể hiện đầy đủ.

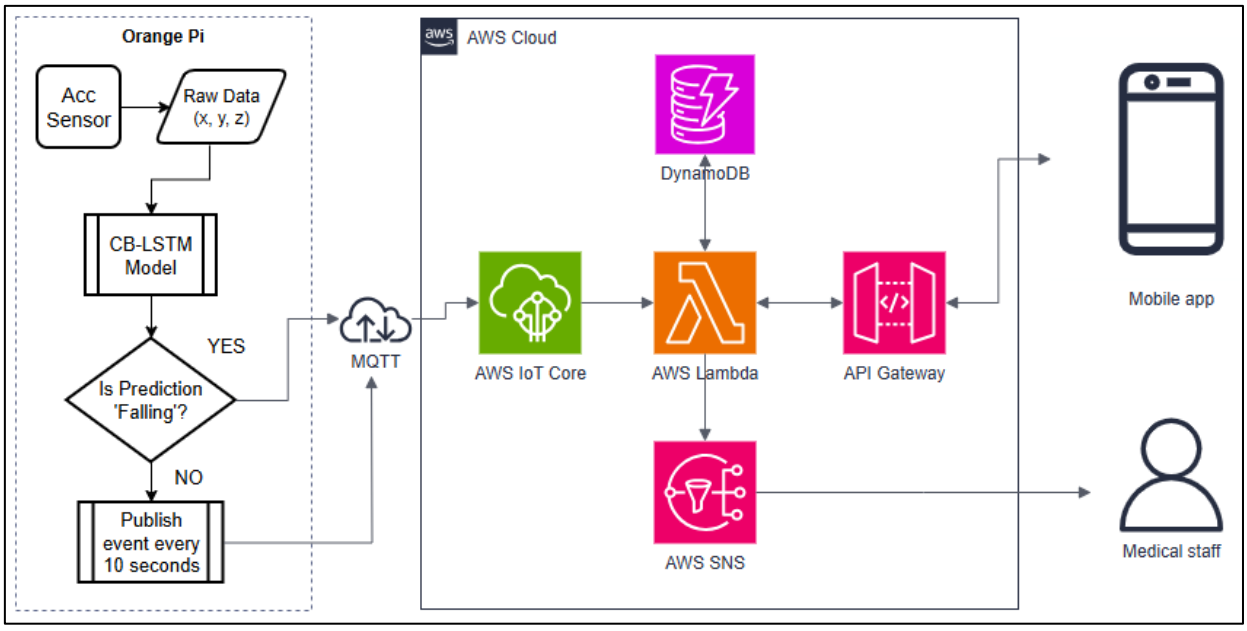
3.6. Triển khai hệ thống phát hiện và cảnh báo té ngã

Để chứng minh tính khả thi và hiệu quả của việc triển khai mô hình học sâu trong môi trường thực tế, mô hình CB-LSTM đã được triển khai trên thiết bị Orange Pi, mô phỏng hoạt động thực tế của một thiết bị đeo tương lai.



Hình 3. 6: Hình ảnh thực tế Orange Pi Zero 2W và cảm biến gia tốc ADXL345

Toàn bộ hệ thống phát hiện và cảnh báo té ngã được thiết kế theo một kiến trúc phân tán gồm các thành phần chính như sau:



Hình 3. 7: Kiến trúc tổng quan hệ thống phát hiện và cảnh báo té ngã

- Orange Pi mô phỏng thiết bị đeo thông minh có cảm biến gia tốc kế và được triển khai mô hình chịu trách nhiệm thu thập dữ liệu gia tốc kế, đưa vào mô hình để phân loại hoạt động hằng ngày hay té ngã theo thời gian thực.

- Dịch vụ cloud Amazon Web Services chịu trách nhiệm nhận dữ liệu và sự kiện theo thời gian thực để lưu trữ và phát đi cảnh báo ngay khi nhận được sự kiện ngã từ thiết bị Orange Pi. Đồng thời cũng đóng vai trò là trung tâm của hệ thống cho ứng dụng di động.

- Ứng dụng di động được phát triển bằng Flutter cho phép nhận cảnh báo ngay lập tức từ cloud.

Trên thiết bị Orange Pi, để mô hình có thể hoạt động hiệu quả trên tài nguyên hạn chế, trước tiên cần tiến hành tối ưu hoá mô hình bằng phương pháp Quantization, giúp tăng tốc thời gian xử lý của mô hình học sâu mà đảm bảo độ chính xác không giảm đi đáng kể bằng cách lưu trữ tensor ở kiểu dữ liệu có số bit thấp hơn kiểu dữ liệu float, từ đó giảm kích thước mô hình, giảm băng thông bộ nhớ, tăng tốc độ xử lý [31]. Để đảm bảo hệ thống hoạt động được trong thực tế, giảm tránh các cảnh báo giả do những hoạt động tay bất thường gây ra cảnh báo ngã (do dữ liệu chỉ được sử dụng để huấn luyện kiểm tra chỉ được chuẩn bị cho các trường hợp hoạt động cụ thể trong thí nghiệm thu thập). Em đã tiến hành nâng ngưỡng phát hiện ngã ở đầu ra của mô hình lên 0.95 để chắc chắn rằng đó là ngã thật sự. Thiết bị thu thập dữ liệu theo thời gian thực với tần số ghi mẫu là 30Hz đúng với cách bộ dữ liệu DU-MD được thu thập [10], khi này sau 3 giây đầu tiên, sẽ có đủ 90 mẫu dữ liệu và bắt đầu đưa vào mô hình để đánh giá, sử dụng bước trược là 30, cứ sau 1 giây, 90 mẫu dữ liệu sẽ được đưa vào mô hình để phân tích kết quả. Khi này để tiếp tục giảm cảnh báo giả trong thực tế và đảm bảo hệ thống hoạt động hiệu quả, em đã tiến hành đặt ngưỡng phát đi cảnh báo té ngã khi mô hình cho ra sự kiện là “Fall” trong ba lần liên tiếp. Vì một hành động té ngã có thể xảy ra trong từ 3 đến 5 giây, căn cứ vào độ nhạy của cảm biến trong thực tế và một lần đưa dữ liệu vào mô hình là 90 mẫu là 3 giây và chồng lắp 2 giây cho nên 3 lần mô hình đưa ra sự kiện ngã sẽ tương ứng với một sự kiện ngã trong thực tế. Hệ thống cũng được thiết lập để cứ sau 10 giây thì gửi dữ liệu đến AWS IoT Core một lần để theo dõi.

Đối với Amazon Cloud Services, cứ sau 5 giây dịch vụ AWS IoT Core sẽ nhận được một tín hiệu chứa dữ liệu từ thiết bị và kích hoạt dịch vụ AWS Lambda để lưu dữ liệu vào dịch vụ AWS DynamoDB. Khi nhận được một tín hiệu dữ liệu chứa sự kiện "Fall", lập tức dịch vụ Lambda được kích hoạt để điều hướng kích hoạt dịch vụ AWS SNS gửi thông báo qua email đến nhân viên y tế được uỷ quyền, đồng thời gửi một thông báo đẩy đến điện thoại di động của người thân được đăng ký.

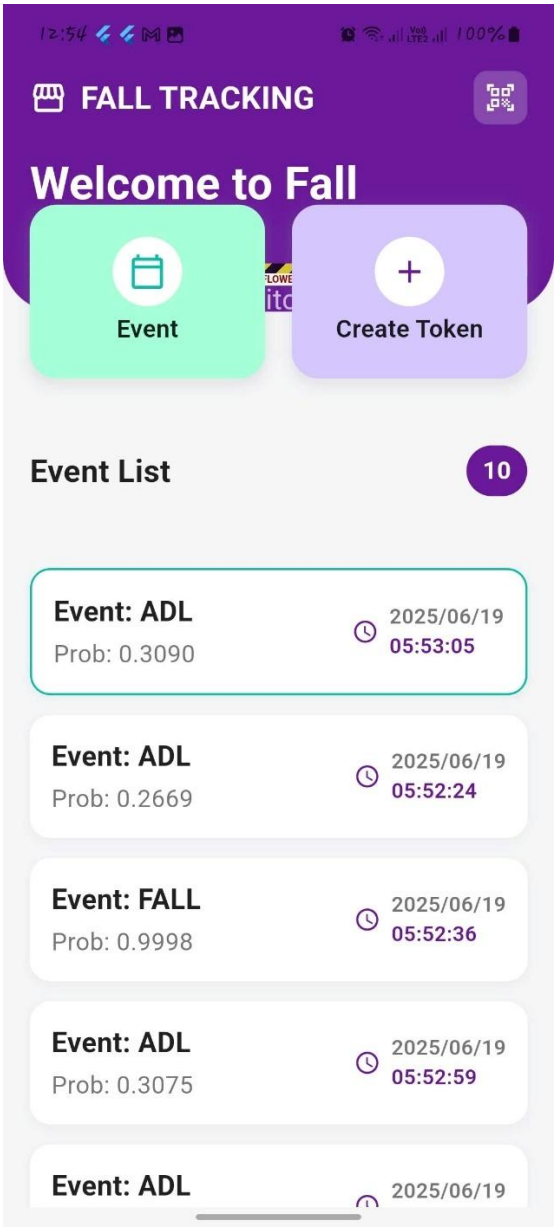
Ứng dụng mobile được viết bằng Flutter cho phép theo dõi hoạt động của người được giám sát và nhận thông báo khi sự kiện ngã xảy ra.

3.7. Kết quả thử nghiệm thực tế

Dữ liệu được truyền qua giao thức MQTT giữa Orange Pi và dịch vụ AWS IoT Core là một chuỗi Json chứa các dữ liệu chính gồm: timestamp, event, prob.

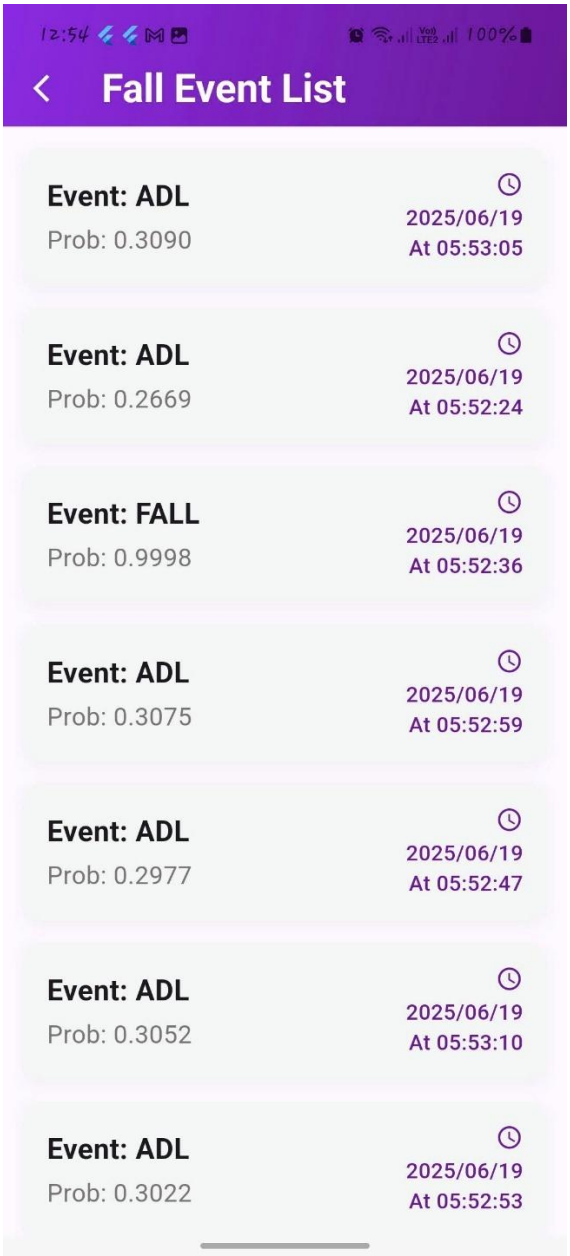
Hình 3. 8: Dữ liệu json giao tiếp giữa Orange Pi và AWS IoT Core

Giao diện người dùng chính trên ứng dụng mobile gồm giao diện màn hình chính, màn hình danh sách sự kiện theo thời gian, và màn hình cảnh báo khi có cảnh báo khẩn cấp và thông báo popup trên màn hình.



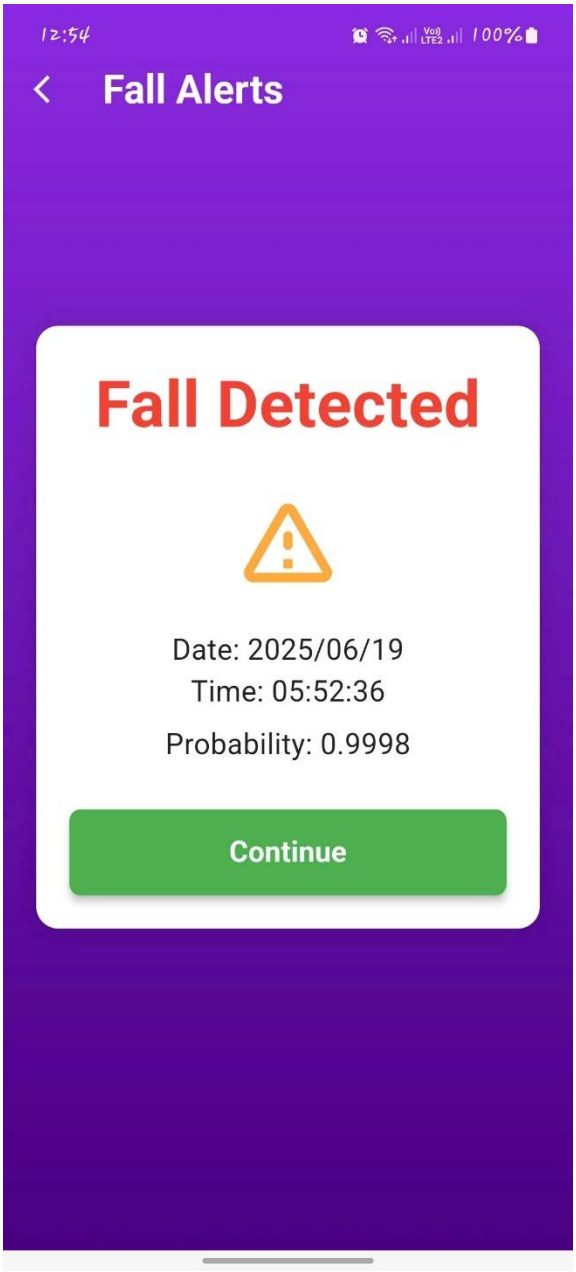
Hình 3. 9: Giao diện màn hình chính

Giao diện màn hình chính hiển thị ứng dụng "FALL TRACKING", dễ dàng cho thấy chức năng chính của ứng dụng là theo dõi các sự kiện té ngã. Màn hình bắt đầu với một lời chào "Welcome to Fall" và cung cấp hai nút lớn: một nút "Event" có thể dùng để xem các sự kiện đã ghi lại, và một nút "Create Token" dùng để tạo mã định danh cho thiết bị để nhận thông báo. Phía dưới là một "Event List" hiển thị các sự kiện theo trình tự thời gian, mỗi sự kiện bao gồm loại hoạt động (như ADL cho hoạt động sinh hoạt hàng ngày hoặc FALL cho té ngã), xác suất mà mô hình dự đoán (ví dụ: xác suất rất cao cho một sự kiện té ngã được phát hiện), và thời gian cụ thể của sự kiện. Các sự kiện sẽ được cập nhật tại màn hình này khi hệ thống nhận được thông báo sự kiện từ thiết bị. Màn hình chính của ứng dụng cung cấp một cái nhìn tổng quan nhanh chóng và chi tiết về các sự kiện té ngã và hoạt động thường ngày đã được ghi nhận.



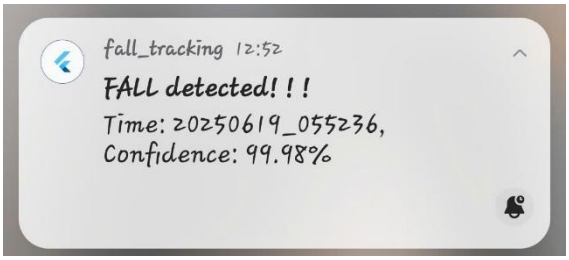
Hình 3. 10: Màn hình danh sách sự kiện

Giao diện di động này hiển thị một danh sách các sự kiện được ghi lại, với tiêu đề "Fall Event List", cho phép người dùng xem lại chi tiết các hoạt động đã xảy ra. Có một nút mũi tên quay lại ở góc trên bên trái, cho phép người dùng quay về màn hình trước. Mỗi mục trong danh sách cung cấp thông tin về loại sự kiện, như "Event: ADL" cho các hoạt động sinh hoạt hàng ngày hoặc "Event: FALL" cho một sự kiện té ngã được phát hiện. Kèm theo đó là "Prob" (xác suất), thể hiện mức độ tin cậy của mô hình đối với dự đoán đó, ví dụ, một giá trị cao như 0.9998 cho thấy mô hình rất tự tin rằng đó là một cú ngã. Cuối cùng, thông tin ngày và giờ chính xác khi sự kiện diễn ra cũng được hiển thị, giúp người dùng dễ dàng theo dõi dòng thời gian của các hoạt động. Khi người dùng sử dụng ứng dụng thì các sự kiện này sẽ được cập nhật theo thời gian thực cứ sau mỗi 10 giây. Sự kiện mới nhất sẽ được đẩy lên đầu của danh sách.



Hình 3. 11: Màn hình cảnh báo khi té ngã

Giao diện di động này hiển thị một màn hình cảnh báo về té ngã, với tiêu đề "Fall Alerts" ở trên cùng. Khi một sự kiện té ngã được phát hiện, một thông báo lớn "Fall Detected" màu đỏ sẽ hiển thị rõ ràng, kèm theo biểu tượng cảnh báo hình tam giác màu cam. Bên dưới thông báo này là các thông tin chi tiết về sự kiện, bao gồm ngày (2025/06/19), thời gian (05:52:36), và xác suất mô hình dự đoán (0.9998). Phía dưới cùng của màn hình cảnh báo là một nút "Continue" màu xanh lá cây, cho phép người dùng tương tác để trở về với màn hình chính của ứng dụng sau khi nhận được cảnh báo. Có một mũi tên quay lại ở góc trên bên trái để người dùng có thể trở về màn hình trước. Khi người dùng sử dụng ứng dụng mà nhận được sự kiện “FALL” thì ứng dụng lập tức chuyển sang màn hình này với cảnh báo rung điện thoại kèm âm thanh. Khi người dùng không dùng ứng dụng thì để tắt thông báo popup người dùng cũng sẽ được chuyển đến màn hình này.



Hình 3. 12: Thông báo popup khi té ngã

Giao diện điện thoại hiển thị một thanh thông báo bật lên (popup notification) xuất hiện từ ứng dụng "fall_tracking" khi một sự kiện té ngã được phát hiện. Thông báo này nổi bật với dòng chữ "FALL detected!!!" và cung cấp thông tin chi tiết về thời gian xảy

ra sự kiện (ví dụ: Time: 20250619_055236) cùng với mức độ tin cậy của mô hình (Confidence: 99.98%). Sự xuất hiện của thông báo này cho thấy chức năng cảnh báo hoạt động theo thời gian thực, ngay lập tức thông báo cho người dùng hoặc người chăm sóc về một cú ngã tiềm năng, đây là một tính năng quan trọng trong hệ thống cảnh báo té ngã. Thông báo xuất hiện rõ ràng trên màn hình, giúp đảm bảo rằng sự kiện quan trọng này không bị bỏ lỡ.

Hình 3. 13: Email nhận được khi xảy ra sự kiện té ngã.

Người đăng ký sẽ nhận được email cảnh báo khi có sự kiện té ngã xảy ra. Người đăng ký ở đây có thể là nhân viên y tế được uỷ quyền hoặc người thân trong gia đình.

Trong quá trình kiểm nghiệm hệ thống, các hoạt động đã được thử bao gồm: đi bộ ngắn, ngồi, nằm, đứng, ngã do ngất xỉu, ngã do đau tim. Tuy nhiên vì hạn chế của thiết bị mô phỏng là Orange Pi cần các kết nối có giây và nguồn, nên các hành động được thử nghiệm hầu như cũng chỉ mang tính mô phỏng hành động của người cao tuổi, các hành động không được thực hiện đầy đủ. Bộ dữ liệu chỉ ghi lại các hành động cụ thể, nhưng trong thực tế còn có cả các hoạt động khác không nằm trong bộ dữ liệu, điều này cũng là một hạn chế khiến hệ thống có thể hoạt động không đúng cách. Dù gặp những hạn chế trên nhưng đối với các mô phỏng cú ngã, hệ thống đều ghi nhận “FALL”. Những kết quả này cho thấy tiềm năng thực tế của hệ thống trong việc cảnh báo té ngã, mặc dù cần được kiểm nghiệm rộng rãi hơn trong các môi trường và kịch bản đa dạng để nâng cao độ tin cậy.

Chương 4. KẾT LUẬN VÀ KIẾN NGHỊ

4.1. Kết quả đạt được

Trong suốt khoảng thời gian làm đồ án, em đã cố gắng hết khả năng của mình vừa nghiên cứu, học hỏi những kiến thức mới vừa làm, vì thời gian và trình độ làm đồ án có hạn nên kết quả đạt được không nhiều, nhưng riêng bản thân em trong quá trình nghiên cứu cũng đã học hỏi được rất nhiều kiến thức mới trong lĩnh vực Trí tuệ Nhân tạo, cũng như kỹ năng tìm kiếm, đọc tài liệu và những kết quả đạt được như sau:

Về mặt khoa học:

- Đề xuất và kiểm chứng mô hình học sâu CB-LSTM, được thiết kế đặc biệt để kết hợp sức mạnh của CNN và BiLSTM để xử lý dữ liệu gia tốc kế theo chuỗi thời gian được thu thập từ cảm biến đeo ở cổ tay.
- Chứng minh thực nghiệm hiệu quả của học sâu trong việc giảm thiểu nhu cầu kỹ thuật xử lý dữ liệu và trích xuất đặc trưng thủ công so với các kỹ thuật học máy truyền thống.
- Thực hiện huấn luyện và đánh giá trên 2 bộ dữ liệu đa dạng DU-MD, UMAFall để đánh giá hiệu suất và khả năng xử lý mạnh mẽ của mô hình đối với những bộ dữ liệu khác nhau.

Về mặt thực tiễn:

- Phát triển thử nghiệm thành công một hệ thống phát hiện té ngã thực tế, hoạt động theo thời gian thực, có khả năng triển khai trên các thiết bị có tài nguyên hạn chế.
- Hệ thống được tích hợp công nghệ IoT với các dịch vụ đám mây và ứng dụng di động để cung cấp khả năng cảnh báo ngay lập tức cho người chăm sóc và nhân viên y tế.

4.2. Hạn chế

Trong quá trình làm đồ án, mặc dù đã rất cố gắng nhưng vẫn còn nhiều hạn chế và lỗi chưa được khắc phục, nên những lý luận chưa được chặt chẽ và hoàn thiện:

- Chưa tìm kiếm và nghiên cứu đầy đủ những nghiên cứu mới nhất về chủ đề này.
- Các bộ dữ liệu sử dụng còn hạn chế
- Kết quả về hiệu suất mô hình chưa cao
- Lý luận cho những nghiên cứu đề xuất chưa tốt
- Hệ thống vẫn còn nhiều hạn chế
- Giao diện và một số chức năng chưa được hoàn thiện

4.3. Hướng phát triển

Trong quá trình làm đồ án, bản thân cũng đã tự học hỏi, tìm hiểu trao dồi kiến thức từ việc tra cứu đọc tài liệu, cũng như các kiến thức học tập được từ người thầy hướng dẫn của mình. Mong muốn trong tương lai sắp tới, đồ án của em được phát triển hoàn thiện hơn theo các hướng như sau:

- Nghiên cứu sâu hơn về những nghiên cứu mới nhất về đề tài của các tác khác được công khai để tối ưu hơn nữa mô hình.
- Sử dụng và đánh giá nhiều bộ dữ liệu hơn nữa để có những đánh giá khách quan hơn.
- Kết hợp thêm những loại dữ liệu khác có thể thu thập được từ thiết bị đeo tay như con quay hồi chuyển, nhịp tim, …
- Nghiên cứu sử dụng các thiết bị đeo tay như smart watch, smart ring có cung cấp SDK để nghiên cứu và phát triển hệ thống sát với thực tế hơn.
- Tiếp tục hoàn thiện ứng dụng di động cho phép theo dõi tốt hơn và các dịch vụ cloud để hệ thống hoạt động tốt hơn.

TÀI LIỆU THAM KHẢO

[1] United Nations (2024). Ageing. https://www.un.org/en/global-issues/ageing

[2] World Health Organization (2024). Ageing and health. https://www.who.int/news-room/fact-sheets/detail/ageing-and-health

[3] Raju Vaishya, Abhishek Vaish (2020). Falls in Older Adults are Serious. Indian journal of orthopaedics, 54(1), 69–74.

[4] Li, R. (2024). A Review of Fall Detection Research Based on Deep Learning. Applied and Computational Engineering,115,87-96.

[5] Li, Y., Liu, P., Fang, Y., Wu, X., Xie, Y., Xu, Z., Ren, H., & Jing, F. (2025). A Decade of Progress in Wearable Sensors for Fall Detection (2015–2024): A Network-Based Visualization Review. Sensors, 25(7), 2205.

[6] Z. Jiang et al. (2024). "Fall Detection Systems for Internet of Medical Things Based on Wearable Sensors: A Review,". In IEEE Internet of Things Journal, vol. 11, no. 21, pp. 34797-34810.

[7] Trần Công Án, Lữ Minh Phúc, Đỗ Thanh Đức, Ngô Bá Hùng, Lê Đình Chiến, Phạm Thị Xuân Diễm, Sơn Búp Pha và Nguyễn Hữu Vân Long (2017). Phát hiện té ngã cho người cao tuổi bằng gia tốc kế và mô hình học sâu Long Short-Term Memory. Tạp chí Khoa học Trường Đại học Cần Thơ, Số chuyên đề: Công nghệ thông tin, 65-71.

[8] Ha, T.V., Nguyen, H., Huynh, S.T., Nguyen, T.T., Nguyen, B.T. (2022). Fall Detection Using Multimodal Data. In: Þór Jónsson, B., et al. MultiMedia Modeling. MMM 2022. Lecture Notes in Computer Science, vol 13141. Springer, Cham.

[9] Li, Z., Du, J., Zhu, B., Greenwald, S. E., Xu, L., Yao, Y., & Bao, N. (2024). Doppler Radar Sensor-Based Fall Detection Using a Convolutional Bidirectional Long Short-Term Memory Model. Sensors, 24(16), 5365.

[10] S. S. Saha, S. Rahman, M. J. Rasna, A. K. M. Mahfuzul Islam and M. A. Rahman Ahad (2018). "DU-MD: An Open-Source Human Action Dataset for Ubiquitous Wearable Sensors,". 2018 Joint 7th International Conference on Informatics, Electronics & Vision

(ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Kitakyushu, Japan, pp. 567-572.

[11] Mienye, I. D., & Swart, T. G. (2024). A Comprehensive Review of Deep Learning: Architectures, Recent Advances, and Applications. Information, 15(12), 755.

[12] Dawood, Afrah & Faris, Zena. (2023). Architecture of Deep Learning and Its Applications. Iraqi Journal of Computer, Communication, Control and System Engineering.

[13] Ian Goodfellow, Yoshua Bengio, Aaron Courville (2016). Deep Learning. MIT Press book. Nguyễn Đình Quý và Nguyễn Khánh Chi dịch.

[14] S. S. Saha, S. Rahman, M. J. Rasna, T. B. Zahid, A. K. M. M. Islam and M. A. R. Ahad (2018). "Feature Extraction, Performance Analysis and System Design Using the DU Mobility Dataset". in IEEE Access. vol. 6, pp. 44776-44786.

[15] Tensorflow: https://www.tensorflow.org/tutorials/

[16] Nguyễn Thanh Tuấn (2019). Deep Learning Cơ Bản (version 2, last update, August 2020). EBook

[17] https://vnptai.io/vi/blog/detail/neural-network-la-gi?

[18] https://aicandy.vn/convolutional-neural-networks-cnn-trong-deep-learning/

[19] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[20] A. O. Ige and M. Sibiya (2024). "State-of-the-Art in 1D Convolutional Neural Networks: A Survey". in IEEE Access, vol. 12, pp. 144082-144105

[21] https://www.geeksforgeeks.org/what-are-convolution-layers/

[22] https://aicandy.vn/recurrent-neural-network-rnn-ung-dung-va-cach-hoat-dong/

[23] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533-536.

[24] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735–1780.

[25] https://en.wikipedia.org/wiki/Long_short-term_memory

[26] https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/

[27] https://www.ultralytics.com/vi/glossary/long-short-term-memory-lstm

[28] https://viblo.asia/p/tim-hieu-lstm-bi-quyet-giu-thong-tin-lau-dai-hieu-qua-MG24BaezVz3

[29] https://www.geeksforgeeks.org/bidirectional-lstm-bi-lstm-in-deep-learning/

[30] Schuster, M., & Paliwal, K. K. (1997). Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

[31] https://viblo.asia/p/quantization-voi-pytorch-phan-1-3Q75wv6GlWb

[32] https://aws.amazon.com/what-is-cloud-computing/

[33] https://aws.amazon.com/vi/what-is/mqtt/

[34] https://aws.amazon.com/what-is/flutter