

Phát Hiện Hành Động Con Người Sử Dụng Cảm Biến Gia Tốc và Mô Hình Transformer trong AIoT

Trịnh Hoàng Hà, Trịnh Thị Yến Mai, Nguyễn Thị Lan Anh, Mai Đức Hòa
Nhóm 11, Khoa Công Nghệ Thông Tin, Trường Đại Học Đại Nam, Hà Nội, Việt Nam

THS.Nguyễn Văn Nhân, THS.Lê Trung Hiếu

Giảng viên hướng dẫn, Khoa Công Nghệ Thông Tin, Trường Đại Học Đại Nam, Hà Nội, Việt Nam

Tóm tắt nội dung—Phát hiện hành động con người (HAR) là một lĩnh vực quan trọng với nhiều ứng dụng thực tiễn như chăm sóc sức khỏe, giám sát thể dục, và phát triển các hệ thống nhà thông minh. Nghiên cứu này đề xuất một phương pháp tiên tiến để nhận diện hành động con người trong thời gian thực bằng cách sử dụng dữ liệu từ cảm biến gia tốc và con quay hồi chuyển thu thập từ thiết bị ESP32, kết hợp với mô hình học sâu Transformer triển khai trong TensorFlow. Hệ thống được tích hợp trong khung IoT, sử dụng giao thức UDP để truyền dữ liệu hiệu quả từ thiết bị đến máy chủ. Chúng tôi đã phát triển một quy trình tiền xử lý dữ liệu cảm biến thô, bao gồm làm mịn, chuẩn hóa và tăng cường dữ liệu bằng các kỹ thuật nhiễu, biến đổi thời gian và thay đổi biên độ. Mô hình Transformer được huấn luyện để phân loại sáu hành động: đứng, ngồi, đi bộ, chạy bộ, ngã và nhảy, với độ chính xác vượt quá 90% trên tập kiểm tra. Hệ thống còn tích hợp giao diện người dùng đồ họa (GUI) để hiển thị kết quả dự đoán và biểu đồ cảm biến theo thời gian thực. Kết quả thử nghiệm cho thấy hiệu suất mạnh mẽ và ổn định, phù hợp cho việc triển khai thực tế trong các ứng dụng IoT.

Index Terms—Phát hiện hành động con người, Mô hình Transformer, IoT, Cảm biến gia tốc, ESP32, Dự đoán thời gian thực

I. GIỚI THIỆU

Phát hiện hành động con người (Human Activity Recognition - HAR) là một lĩnh vực nghiên cứu quan trọng trong khoa học máy tính và kỹ thuật, cho phép nhận diện và phân loại các hoạt động thể chất dựa trên dữ liệu cảm biến [1]. Các ứng dụng của HAR rất đa dạng, bao gồm giám sát sức khỏe cho người cao tuổi, theo dõi hiệu suất thể thao, phát hiện tình trạng ngã trong y tế, và điều khiển thông minh trong các hệ thống nhà tự động. Với sự phát triển mạnh mẽ của công nghệ Internet vạn vật (IoT), việc tích hợp các cảm biến nhỏ gọn như gia tốc kế và con quay hồi chuyển với các thuật toán học sâu đã mở ra cơ hội nâng cao độ chính xác, hiệu quả và tính thực tiễn của các hệ thống HAR trong đời sống hàng ngày [16].

Trong bối cảnh hiện nay, nhu cầu về các hệ thống HAR ngày càng tăng, đặc biệt trong các lĩnh vực như y tế và chăm sóc sức khỏe. Ví dụ, việc phát hiện sớm các tình trạng ngã ở người cao tuổi có thể giúp giảm thiểu nguy cơ chấn thương nghiêm trọng, trong khi các hệ thống giám sát thể dục có thể hỗ trợ người dùng tối ưu hóa hiệu suất tập luyện. Ngoài ra, HAR còn đóng vai trò quan trọng trong việc phát triển các hệ thống nhà thông minh, nơi các thiết bị có thể tự động điều chỉnh dựa trên hành vi của người dùng, chẳng hạn như bật/tắt đèn khi phát hiện người đi vào hoặc rời khỏi phòng. Tuy nhiên, để triển khai các hệ thống HAR hiệu quả, cần vượt qua nhiều thách thức,

bao gồm việc xử lý dữ liệu cảm biến thô chứa nhiễu, đảm bảo tính thời gian thực trong các ứng dụng IoT, và phát triển các mô hình học máy có khả năng phân loại chính xác các hành động phức tạp.

Một trong những thách thức lớn nhất của HAR là sự đa dạng và phức tạp của các hành động con người. Các hành động như đi bộ và chạy bộ có thể có tín hiệu cảm biến tương tự nhau, trong khi các hành động như ngã có thể xảy ra đột ngột và không theo một mẫu cố định. Điều này đòi hỏi các mô hình học máy phải có khả năng phân biệt các mẫu tín hiệu tinh vi, đồng thời đủ linh hoạt để thích nghi với các biến đổi trong dữ liệu thực tế. Ngoài ra, trong các hệ thống IoT, việc truyền dữ liệu từ thiết bị đến máy chủ cần được thực hiện với độ trễ thấp để đảm bảo tính thời gian thực, đặc biệt trong các ứng dụng như phát hiện ngã, nơi phản hồi nhanh có thể cứu sống người dùng.

Trong các phương pháp truyền thống, HAR thường dựa vào việc trích xuất thủ công các đặc trưng từ dữ liệu cảm biến, chẳng hạn như giá trị trung bình, phương sai, hoặc biên độ tín hiệu, sau đó áp dụng các thuật toán học máy cổ điển như Máy Vector Hỗ trợ (SVM) hoặc Mô hình Markov Ẩn (HMM) để phân loại [3]. Tuy nhiên, những phương pháp này có nhiều hạn chế. Thứ nhất, quá trình trích xuất đặc trưng đòi hỏi kiến thức chuyên môn sâu và tốn thời gian. Thứ hai, chúng thường không hiệu quả trong việc xử lý các chuỗi dữ liệu thời gian phức tạp, đặc biệt khi các hành động có sự tương đồng về tín hiệu (ví dụ: đi bộ và chạy bộ) hoặc khi dữ liệu bị nhiễu bởi các yếu tố ngoại cảnh. Sự ra đời của học sâu đã mang lại bước tiến vượt bậc, với các mô hình như Mạng Nơ-ron Tích chập (CNN) và Mạng Nơ-ron Hồi tiếp (RNN) cho phép học trực tiếp từ dữ liệu thô mà không cần trích xuất đặc trưng thủ công [4]. Gần đây hơn, mô hình Transformer — vốn nổi tiếng trong xử lý ngôn ngữ tự nhiên — đã được áp dụng thành công cho các tác vụ chuỗi thời gian nhờ cơ chế chú ý (attention) mạnh mẽ, giúp nắm bắt các phụ thuộc dài hạn một cách hiệu quả [2].

Mặc dù các mô hình học sâu đã cải thiện đáng kể hiệu suất của HAR, việc triển khai chúng trong các hệ thống IoT vẫn đối mặt với nhiều thách thức. Một trong những vấn đề lớn nhất là yêu cầu tài nguyên tính toán cao của các mô hình học sâu, trong khi các thiết bị IoT như ESP32 thường có tài nguyên hạn chế về bộ nhớ và sức mạnh xử lý. Ngoài ra, việc truyền dữ liệu từ thiết bị đến máy chủ trong thời gian thực đòi hỏi các giao thức truyền thông hiệu quả, đồng thời phải đảm bảo độ trễ thấp để đáp ứng yêu cầu của các ứng dụng thực tế. Hơn nữa, dữ liệu

cảm biến trong môi trường thực tế thường bị ảnh hưởng bởi nhiều từ nhiều nguồn khác nhau, như rung động từ môi trường hoặc chuyển động không mong muốn của người dùng, đòi hỏi các kỹ thuật tiền xử lý dữ liệu mạnh mẽ để đảm bảo chất lượng đầu vào cho mô hình.

Nghiên cứu này đề xuất một hệ thống HAR dựa trên IoT, tận dụng dữ liệu từ cảm biến gia tốc và con quay hồi chuyển của thiết bị ESP32 – một nền tảng nhúng nhỏ gọn, giá rẻ và phổ biến trong cộng đồng phát triển IoT. Dữ liệu được truyền qua giao thức UDP đến máy chủ, nơi nó được xử lý bởi mô hình Transformer triển khai trong TensorFlow. Hệ thống không chỉ dự đoán hành động theo thời gian thực với độ tin cậy cao mà còn cung cấp giao diện người dùng đồ họa (GUI) để hiển thị kết quả và biểu đồ cảm biến, hỗ trợ người dùng theo dõi trực quan. Các đóng góp chính của nghiên cứu bao gồm:

- 1) Một quy trình tiền xử lý dữ liệu cảm biến toàn diện, bao gồm làm mịn tín hiệu bằng bộ lọc Savizky-Golay, chuẩn hóa dữ liệu bằng StandardScaler, và tăng cường dữ liệu để cải thiện độ bền của mô hình.
- 2) Sử dụng và huấn luyện một mô hình Transformer tối ưu cho phân loại chuỗi thời gian từ dữ liệu cảm biến, với khả năng phân biệt sáu hành động khác nhau.
- 3) Phát triển khung IoT sử dụng giao thức UDP để truyền dữ liệu hiệu quả giữa ESP32 và máy chủ, đảm bảo tính thời gian thực.
- 4) Triển khai cơ chế dự đoán thời gian thực với thuật toán làm mịn dựa trên trung bình trượt có trọng số, tăng cường độ ổn định của kết quả trong môi trường thực tế.

Phần còn lại của bài báo sẽ trình bày chi tiết phương pháp luận, kết quả thử nghiệm và các hướng phát triển trong tương lai.

II. CÁC NGHIÊN CỨU LIÊN QUAN

Nghiên cứu về HAR đã trải qua nhiều giai đoạn phát triển, từ các phương pháp truyền thống dựa trên thống kê đến các cách tiếp cận hiện đại sử dụng học sâu. Trong giai đoạn đầu, các công trình tập trung vào việc sử dụng các đặc trưng được trích xuất thủ công từ dữ liệu cảm biến, chẳng hạn như biên độ tín hiệu, tần số hoặc năng lượng, kết hợp với các thuật toán học máy như SVM, K-Nearest Neighbors (KNN), hoặc Decision Trees [5]. Những phương pháp này tuy đơn giản và dễ triển khai trên các thiết bị có tài nguyên hạn chế, nhưng đòi hỏi quá trình trích xuất đặc trưng phức tạp và thường không hiệu quả khi xử lý các tập dữ liệu lớn, đa dạng hoặc chứa nhiễu.

Sự xuất hiện của học sâu đã đánh dấu một bước ngoặt trong lĩnh vực HAR. Các mô hình CNN được sử dụng để tự động trích xuất đặc trưng không gian từ dữ liệu cảm biến, giảm bớt sự phụ thuộc vào thiết kế đặc trưng thủ công. Trong khi đó, RNN, đặc biệt là LSTM, tỏ ra vượt trội trong việc mô hình hóa các phụ thuộc thời gian ngắn hạn, giúp nhận diện các hành động có tính tuần tự như đi bộ hoặc chạy bộ [12]. Tuy nhiên, cả hai loại mô hình này đều có những hạn chế đáng kể. CNN không thể xử lý tốt các phụ thuộc dài hạn trong dữ liệu thời gian, trong khi RNN thường gặp vấn đề về gradient biến mất hoặc bùng nổ khi chuỗi dữ liệu quá dài, dẫn đến khó khăn trong việc huấn luyện trên các tập dữ liệu phức tạp.

Mô hình Transformer, được giới thiệu bởi Vaswani và cộng sự vào năm 2017 [2], đã mang lại một giải pháp đột phá. Ban đầu được thiết kế cho xử lý ngôn ngữ tự nhiên, Transformer sử dụng cơ chế chú ý đa đầu (multi-head attention) để tập trung đồng thời vào nhiều phần khác nhau của chuỗi dữ liệu, khắc phục hạn chế của RNN về phụ thuộc dài hạn và của CNN về phạm vi không gian [19]. Trong những năm gần đây, Transformer đã được áp dụng ngày càng nhiều cho các tác vụ chuỗi thời gian, bao gồm cả HAR, nhờ khả năng học các mẫu phức tạp mà không cần cấu trúc tuần tự cố định.

Trong bối cảnh IoT, các hệ thống HAR thường ưu tiên các mô hình nhẹ để triển khai trên thiết bị nhúng như ESP32 hoặc Arduino, nhằm giảm độ trễ và tiêu thụ năng lượng [16]. Tuy nhiên, với sự phát triển của phần cứng và các kỹ thuật tối ưu hóa thuật toán, các mô hình phức tạp như Transformer đã bắt đầu được thử nghiệm trong các ứng dụng thời gian thực. Nghiên cứu của chúng tôi nổi bật ở chỗ kết hợp mô hình Transformer với hệ thống IoT, tận dụng khả năng xử lý dữ liệu từ ESP32 theo thời gian thực và cung cấp giao diện trực quan cho người dùng, mở ra tiềm năng ứng dụng trong các lĩnh vực như y tế, thể thao và an ninh [20].

III. PHƯƠNG PHÁP

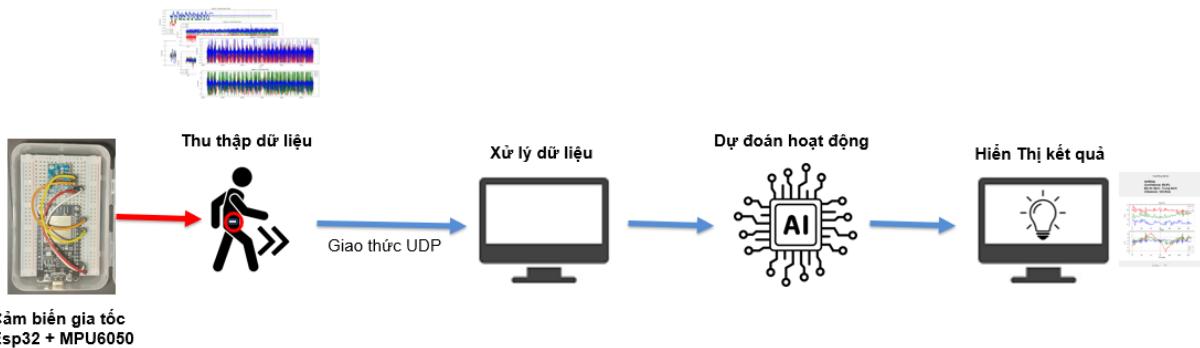
Phương pháp được đề xuất trong nghiên cứu này tập trung vào việc xây dựng một hệ thống phát hiện hành động con người (HAR) thời gian thực, tích hợp các thành phần từ thu thập dữ liệu, xử lý dữ liệu, huấn luyện mô hình, đến dự đoán và hiển thị kết quả. Cụ thể, hệ thống sử dụng thiết bị ESP32 để thu thập dữ liệu từ cảm biến gia tốc và con quay hồi chuyển, truyền dữ liệu qua giao thức UDP đến máy chủ. Tại máy chủ, dữ liệu được tiền xử lý thông qua các bước làm mịn, chuẩn hóa và phân đoạn, sau đó được đưa vào mô hình Transformer để phân loại sáu hành động: đứng, ngồi, đi bộ, chạy bộ, ngã và nhảy. Kết quả dự đoán được làm mịn để tăng độ ổn định và hiển thị trên giao diện người dùng đồ họa (GUI), cho phép theo dõi trực quan theo thời gian thực. Các bước chi tiết của phương pháp này sẽ được trình bày trong các mục dưới đây.

A. Kiến Trúc Hệ Thống

Hệ thống được thiết kế để nhận diện hành động con người trong thời gian thực, tích hợp các thành phần từ thu thập dữ liệu, xử lý dữ liệu, đến dự đoán và hiển thị kết quả. Sơ đồ của mô hình Transformer được sử dụng trong hệ thống được trình bày trong Hình 1. Sơ đồ này minh họa quy trình xử lý dữ liệu từ cảm biến, bao gồm các bước làm mịn, chuẩn hóa, mã hóa vị trí, và phân loại hành động thông qua Transformer Encoder và Classifier Head.

Quá trình xử lý dữ liệu trong mô hình Transformer có thể được biểu diễn toán học như sau. Với mỗi chuỗi dữ liệu cảm biến $C \in \mathbb{R}^{T \times d}$, nơi T là độ dài chuỗi và d là số chiều của dữ liệu (6 kênh: 3 gia tốc và 3 con quay hồi chuyển), vị trí của từng mẫu trong chuỗi được mã hóa bằng vector $P \in \mathbb{R}^{T \times d}$. Biểu diễn đầu vào $E \in \mathbb{R}^{T \times d}$ được tính bằng cách cộng dữ liệu cảm biến đã chuẩn hóa với mã hóa vị trí:

$$E = C + P \quad (1)$$



Hình 1: Sơ đồ mô hình Transformer cho nhận diện hành động: Dữ liệu từ cảm biến được làm mịn và chuẩn hóa, sau đó được mã hóa vị trí và đưa vào Transformer Encoder để học biểu diễn, cuối cùng được phân loại bởi Classifier Head.

Biểu diễn này sau đó được đưa vào Transformer Encoder để học các đặc trưng, tạo ra đầu ra $E' \in \mathbb{R}^{T \times d}$. Cuối cùng, đầu ra của Transformer Encoder được tổng hợp và đưa vào Classifier Head để dự đoán hành động Z_0 , theo công thức:

$$Z_0 = [C, E] \cdot E' \cdot E'^{(d-1)} \quad (2)$$

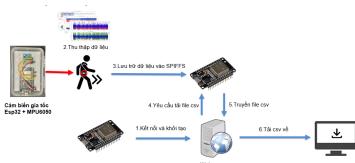
trong đó $[C, E]$ là phép nối dữ liệu cảm biến và biểu diễn mã hóa, và $E'^{(d-1)}$ là chiều cuối cùng của đầu ra Transformer Encoder.

Hệ thống bao gồm ba thành phần chính, tạo thành một quy trình khép kín từ thu thập dữ liệu đến dự đoán hành động:

- 1) **Thu Thập Dữ Liệu:** Thiết bị ESP32, được tích hợp cảm biến gia tốc và con quay hồi chuyển, thu thập dữ liệu ở tần số 50 Hz (tức 50 mẫu mỗi giây). Dữ liệu được định dạng dưới dạng JSON với sáu trục (AccelX, AccelY, AccelZ, GyroX, GyroY, GyroZ) và truyền qua giao thức UDP đến máy chủ tại địa chỉ IP mặc định 192.168.52.147, sử dụng dải cổng từ 8080 đến 8090. UDP được chọn nhờ tốc độ truyền cao và phù hợp với ứng dụng thời gian thực, dù không đảm bảo thứ tự gói tin [16].



Hình 2: Thiết bị ESP32 được sử dụng để thu thập dữ liệu.



Hình 3: Sơ đồ thu thập dữ liệu của hệ thống.

- 2) **Xử Lý Dữ Liệu:** Dữ liệu thô từ ESP32 được xử lý qua ba giai đoạn chính: làm mịn để loại bỏ nhiễu, chuẩn hóa

để đưa về phân phối chuẩn, và phân đoạn thành các chuỗi thời gian để chuẩn bị cho mô hình học sâu. Các bước này sẽ được trình bày chi tiết ở phần sau [18].

- 3) **Dự Đoán Hành Động:** Mô hình Transformer nhận các chuỗi dữ liệu đã xử lý, thực hiện suy luận để dự đoán hành động, và áp dụng thuật toán làm mịn dựa trên trung bình trượt có trọng số để tăng độ ổn định của kết quả. Kết quả cuối cùng được hiển thị qua giao diện người dùng đồ họa (GUI) trên máy chủ.

B. Quy Trình Thu Thập Dữ Liệu

Để đảm bảo chất lượng dữ liệu đầu vào cho mô hình, quy trình thu thập dữ liệu được thiết kế cẩn thận với các bước cụ thể như sau:

- 1) **Lựa Chọn Đối Tượng Thủ Nghiệm:** Dữ liệu được thu thập từ 7 đối tượng, bao gồm cả nam và nữ, trong độ tuổi từ 20 đến 40, nhằm đảm bảo tính đa dạng về thể chất và phong cách chuyển động. Mỗi đối tượng thực hiện sáu hành động (đứng, ngồi, đi bộ, chạy bộ, ngã, nhảy) trong các điều kiện khác nhau, bao gồm cả trong nhà và ngoài trời. Tổng cộng, tập dữ liệu bao gồm 114,279 mẫu. Số lượng mẫu trung bình cho mỗi hành động từ mỗi đối tượng được trình bày trong Bảng I.

Bảng I: Phân bố số lượng mẫu theo từng hành động

Hành động	Số mẫu trung bình mỗi người	Tổng số mẫu
Đi bộ (Walking)	4,620	32,343
Đứng (Standing)	3,030	21,213
Ngồi (Sitting)	3,120	21,839
Chạy bộ (Jogging)	3,011	21,075
Ngã (Falling)	1,341	9,390
Nhảy (Jumping)	1,203	8,419

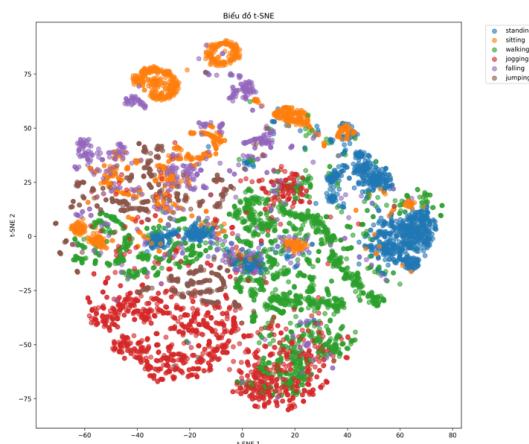
Phân bố số lượng mẫu và tỷ lệ phần trăm của từng hành động được thể hiện trong Hình 4. Ngoài ra, các Hình từ 5 đến 10 minh họa dữ liệu cảm biến thô (gia tốc và con quay hồi chuyển) thu thập được trong quá trình thực hiện các hành động, cho thấy sự biến thiên của tín hiệu theo thời gian.

- 2) **Cài Đặt Thiết Bị:** Thiết bị ESP32 được gắn trên cổ tay của đối tượng để ghi nhận dữ liệu từ cảm biến gia tốc

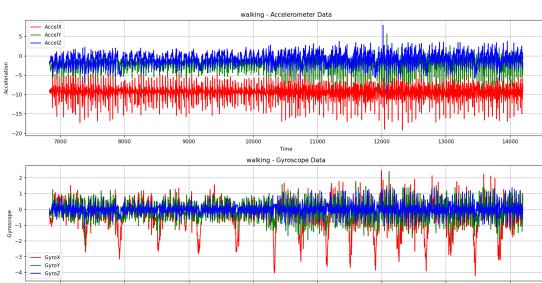
và con quay hồi chuyển. Vị trí này được chọn vì nó cho phép ghi nhận các chuyển động của cánh tay, vốn có sự khác biệt rõ rệt giữa các hành động như đi bộ, chạy bộ hoặc ngã.

- 3) **Ghi Nhận Dữ Liệu:** Mỗi hành động được ghi lại trong khoảng thời gian phù hợp (xem Bảng II), và dữ liệu được gắn nhãn thủ công bằng cách sử dụng một ứng dụng ghi chú thời gian thực. Quá trình này đảm bảo rằng mỗi mẫu dữ liệu được gán chính xác với hành động tương ứng, tạo điều kiện thuận lợi cho việc huấn luyện mô hình.
- 4) **Lưu Trữ Dữ Liệu:** Dữ liệu thô được lưu trữ dưới dạng tệp JSON trong thư mục `raw_data`, với mỗi tệp chứa thông tin về thời gian, giá trị cảm biến, và nhãn hành động. Sau đó, dữ liệu được xử lý sơ bộ (làm mịn và chuẩn hóa) trước khi đưa vào huấn luyện.

Quy trình này không chỉ đảm bảo chất lượng dữ liệu mà còn giúp tăng tính đa dạng của tập dữ liệu, từ đó cải thiện khả năng tổng quát hóa của mô hình trong các kịch bản thực tế.



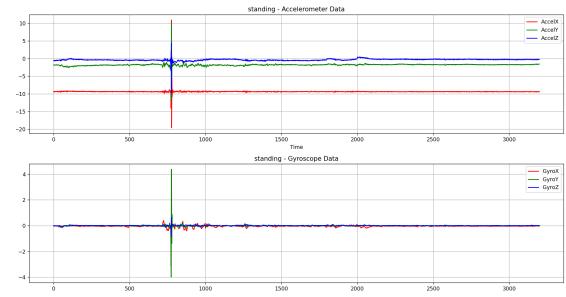
Hình 4: Đặc điểm tập dữ liệu: (a) Trực quan hóa t-SNE của tập dữ liệu, (b) Histogram số lượng mẫu cho từng hành động, (c) Tỷ lệ phần trăm của từng hành động.



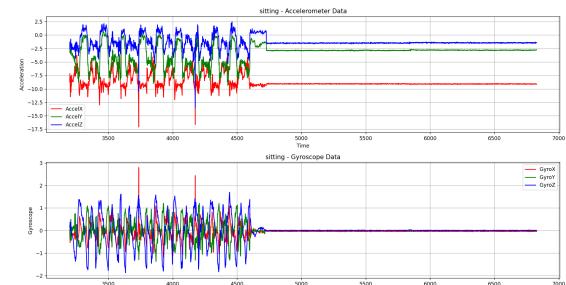
Hình 5: Dữ liệu cảm biến thô cho hành động đi bộ: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).

C. Tiền Xử Lý Dữ Liệu

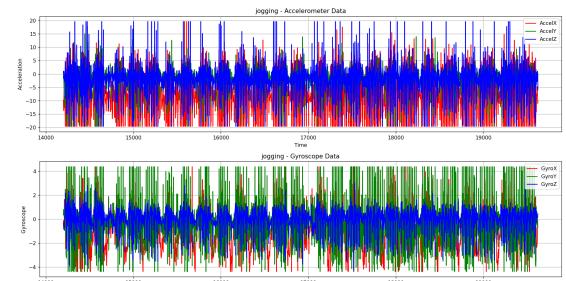
Dữ liệu cảm biến thô từ ESP32 bao gồm sáu kênh: ba trục gia tốc (AccelX, AccelY, AccelZ) đo giá tốc tuyến tính (m/s^2)



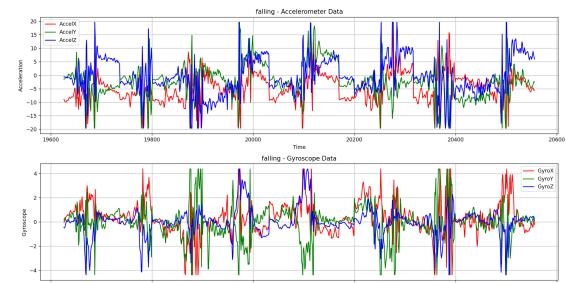
Hình 6: Dữ liệu cảm biến thô cho hành động đứng: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).



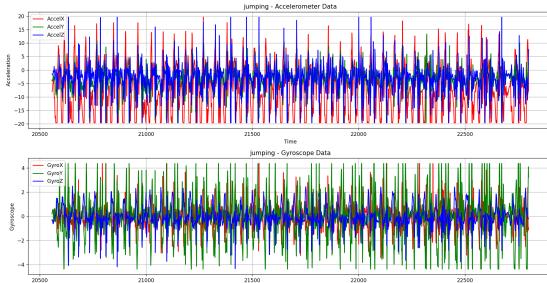
Hình 7: Dữ liệu cảm biến thô cho hành động ngồi: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).



Hình 8: Dữ liệu cảm biến thô cho hành động chạy bộ: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).



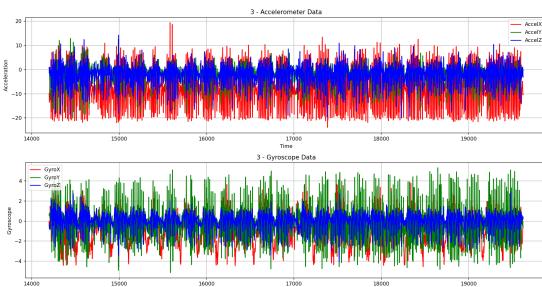
Hình 9: Dữ liệu cảm biến thô cho hành động ngã: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).



Hình 10: Dữ liệu cảm biến thô cho hành động nhảy: (a) Dữ liệu gia tốc (AccelX, AccelY, AccelZ), (b) Dữ liệu con quay hồi chuyển (GyroX, GyroY, GyroZ).

và ba trục con quay hồi chuyển (GyroX, GyroY, GyroZ) đo vận tốc góc (rad/s). Do dữ liệu thô thường chứa nhiều từ môi trường hoặc chuyển động không mong muốn, một quy trình tiền xử lý toàn diện được thực hiện như sau:

- 1) **Làm Mịn:** Bộ lọc Savitzky-Golay được áp dụng với kích thước cửa sổ 5 mẫu và bậc đa thức 2. Bộ lọc này sử dụng phương pháp nội suy đa thức để làm mịn dữ liệu, loại bỏ nhiễu ngẫu nhiên mà vẫn bảo toàn xu hướng tín hiệu một ưu điểm vượt trội so với trung bình trượt thông thường vốn có thể làm mất chi tiết tín hiệu [18]. Hình 11 minh họa dữ liệu cảm biến trước và sau khi làm mịn cho hành động nhảy, cho thấy hiệu quả của bộ lọc trong việc giảm nhiễu mà vẫn giữ được các đặc trưng chính của tín hiệu.



Hình 11: Dữ liệu cảm biến trước và sau khi làm mịn cho hành động đi bộ: (a) Dữ liệu gia tốc thô và sau khi làm mịn, (b) Dữ liệu con quay hồi chuyển thô và sau khi làm mịn.

- 2) **Chuẩn Hóa:** StandardScaler từ thư viện Scikit-learn được sử dụng để chuẩn hóa dữ liệu về trung bình 0 và phương sai 1. Các tham số chuẩn hóa (mean và scale) được lưu vào tệp `scaler_params.npz` để tái sử dụng trong giai đoạn suy luận, đảm bảo tính nhất quán giữa huấn luyện và dự đoán.
- 3) **Tạo Chuỗi:** Dữ liệu liên tục được phân đoạn thành các chuỗi 32 mẫu với stride (bước trượt) 16, tạo ra sự chồng lấp 50% giữa các chuỗi. Độ dài 32 mẫu tương ứng với 0.64 giây dữ liệu ở tần số 50 Hz, đủ để nắm bắt các mẫu hành động ngắn hạn, trong khi stride 16 cân bằng giữa tính liên tục và hiệu quả tính toán.
- 4) **Tăng Cường Dữ Liệu:** Để cải thiện độ bền của mô hình trước các biến đổi thực tế, dữ liệu huấn luyện được tăng cường bằng ba kỹ thuật: (1) Thêm nhiễu ngẫu nhiên với

noise factor 0.05 để mô phỏng nhiễu môi trường; (2) Biến đổi thời gian (time warping) với sigma 0.2 để thay đổi tốc độ hành động; (3) Thay đổi biên độ (magnitude scaling) với sigma 0.1 để mô phỏng cường độ khác nhau của cùng một hành động.

D. Mô Hình Transformer

Mô hình Transformer được thiết kế đặc biệt cho phân loại chuỗi thời gian, với cấu trúc bao gồm các thành phần chính như sau:

- 1) **Mã Hóa Vị Trí:** Một lớp Embedding thêm thông tin vị trí vào mỗi mẫu trong chuỗi 32 mẫu, giúp mô hình nhận biết thứ tự thời gian của dữ liệu – một yếu tố quan trọng trong các tác vụ chuỗi thời gian không tuần tự như HAR [20].
- 2) **Khối Transformer:** Hai khối Transformer được xếp chồng, mỗi khối bao gồm: (1) MultiHeadAttention với 4 đầu chú ý và `key_dim` 6 (tương ứng với 6 kênh dữ liệu), cho phép mô hình tập trung vào các phần khác nhau của chuỗi; (2) Mạng feed-forward với chiều 32 và hàm kích hoạt ReLU; (3) Hai lớp LayerNormalization để ổn định huấn luyện; (4) Dropout với tỷ lệ 0.1 để giảm overfitting.
- 3) **Lớp Đầu Ra:** Sau khi đi qua hai khối Transformer, dữ liệu được giảm chiều bằng GlobalAveragePooling1D, sau đó truyền qua một lớp Dense với 64 đơn vị và hàm ReLU, dropout 0.3, và cuối cùng là lớp Dense với 6 đơn vị và hàm softmax để dự đoán xác suất của 6 lớp hành động (đứng, ngồi, đi bộ, chạy bộ, ngã, nhảy).

Công thức cốt lõi của cơ chế chú ý trong Transformer được định nghĩa như sau. Đầu tiên, cơ chế chú ý có tỷ lệ (Scaled Dot-Product Attention) được tính toán dựa trên ba vector đầu vào: truy vấn (Q), khóa (K), và giá trị (V), với công thức:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (3)$$

trong đó d_k là chiều của vector khóa, được dùng để chuẩn hóa nhằm tránh giá trị lớn gây bất ổn định trong hàm softmax.

Mô hình Transformer sử dụng cơ chế chú ý đa đầu (Multi-Head Attention) để tăng khả năng biểu diễn. Công thức cho Multi-Head Attention được biểu diễn như:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O \quad (4)$$

với mỗi head_i được tính bởi:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (5)$$

trong đó $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}$ và $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ là các ma trận trọng số học được, h là số lượng đầu chú ý (trong nghiên cứu này, $h = 4$), và d_v là chiều của vector giá trị.

Cơ chế này cho phép mô hình tập trung đồng thời vào nhiều phần khác nhau của chuỗi dữ liệu cảm biến, giúp nhận diện các mẫu hành động phức tạp một cách hiệu quả [2].

Mô hình được huấn luyện với bộ tối ưu Adam (learning rate ban đầu $1e-4$), hàm mất mát sparse categorical cross-entropy, và các callback như EarlyStopping (patience 15) để dừng sớm nếu không cải thiện độ chính xác trên tập xác thực, và ReduceLROnPlateau (factor 0.5, min_lr $1e-6$) để giảm tốc độ học khi hiệu suất ngừng tăng.

E. Dự Đoán Thời Gian Thực

Lớp ActivityPredictor được thiết kế để quản lý toàn bộ quy trình dự đoán thời gian thực:

- 1) **Nhận Dữ Liệu:** Socket UDP lắng nghe dữ liệu từ ESP32 trên cổng 8080-8090, xử lý các gói JSON và lưu vào bộ đệm trượt (deque) với kích thước tối đa 128 mẫu. Nếu kết nối thất bại, hệ thống chuyển sang chế độ mô phỏng dữ liệu ngẫu nhiên để duy trì hoạt động.
- 2) **Dự Đoán:** Cứ mỗi 2 giây (100 mẫu ở 50 Hz), dữ liệu từ bộ đệm được phân đoạn thành chuỗi 32 mẫu, chuẩn hóa bằng scaler đã lưu, và đưa vào mô hình Transformer để suy luận theo lô (batch size 4). Kết quả bao gồm chỉ số hành động (0-5) và độ tin cậy (confidence) dựa trên xác suất softmax.
- 3) **Làm Min:** Một cửa sổ trượt 7 dự đoán được sử dụng với trọng số tuyến tính từ 0.6 đến 1.0 (tăng dần theo thời gian), kết hợp ngưỡng tin cậy tối thiểu 0.3 và ngưỡng dự đoán 0.3 để xác định hành động cuối cùng. Điều này giảm thiểu dao động trong dự đoán khi tín hiệu thay đổi nhanh.
- 4) **Hiển Thị:** GUI dựa trên Tkinter hiển thị thông tin theo thời gian thực, bao gồm tên hành động, độ tin cậy (%), độ ổn định (Cao/Trung bình/Thấp), thời gian suy luận (ms), và hai biểu đồ: tua tốc (3 trục) và con quay hồi chuyển (3 trục).

IV. THỦ NGHIỆM

A. Tập Dữ Liệu

Tập dữ liệu được thu thập từ thiết bị ESP32, bao gồm các phép đo từ 7 đối tượng thực hiện sáu hành động: đứng, ngồi, đi bộ, chạy bộ, ngã và nhảy. Mỗi hành động được gắn nhãn số từ 0 đến 5 theo ánh xạ trong tệp activity_mapping.csv. Dữ liệu thô ban đầu được lưu trong thư mục raw_data, sau đó được xử lý (làm mịn, chuẩn hóa) và lưu vào processed_data. Sau khi tăng cường bằng các kỹ thuật nhiễu, biến đổi thời gian và thay đổi biên độ, tập huấn luyện chứa hơn 10.000 chuỗi, trong đó 80% được dùng để huấn luyện và 20% để kiểm tra. Phân chia được thực hiện bằng hàm train_test_split với tham số stratify để đảm bảo phân bố nhãn đồng đều giữa các tập.

Quy trình thu thập dữ liệu được thực hiện với tần số thu thập 50 Hz, và các quy tắc cụ thể được áp dụng cho từng hành động như trong Bảng II. Bảng này mô tả thời gian thu thập và mục đích của từng hành động, đảm bảo rằng dữ liệu thu thập được phản ánh chính xác các đặc điểm của hành động trong thực tế.

Bảng II: Quy Tắc Thu Thập Dữ Liệu

Hành động	Thời gian thực hiện	Mục đích thời gian
Đi bộ (Walking)	1 phút	Thu đủ chu kỳ, ổn định
Chạy bộ (Jogging)	1 phút	Thu đủ chu kỳ, ổn định
Đứng yên (Standing)	1 phút	Thu đủ đặc trưng
Ngồi (Sitting)	1 phút	Thu đủ đặc trưng
Ngã (Falling)	3-5 giây	Ghi lại khoảng khắc đột ngột
Nhảy	1 phút	Thu đủ chu kỳ, ổn định

Quy trình này đảm bảo rằng dữ liệu thu thập được không chỉ đa dạng mà còn phù hợp với các đặc điểm riêng của từng hành

động, từ đó cung cấp một tập dữ liệu chất lượng cao cho việc huấn luyện và đánh giá mô hình.

B. Huấn Luyện

Mô hình được huấn luyện trên máy chủ có GPU, sử dụng TensorFlow 2.x với cấu hình sau:

- 1) **Thông số huấn luyện:** 50 epoch, batch size 64, optimizer Adam với learning rate 1e-4.
- 2) **Callbacks:**
 - a) EarlyStopping với patience 15, khôi phục trọng số tốt nhất nếu độ chính xác trên tập xác thực không tăng.
 - b) ReduceLROnPlateau với factor 0.5 và min_lr 1e-6, giảm learning rate khi hiệu suất chững lại.
 - c) SafeInterruptCallback cho phép dừng an toàn bằng Ctrl+C, lưu trạng thái mô hình nếu bị gián đoạn.
- 3) **Chuẩn bị mô hình:** Trước khi huấn luyện, mô hình được "warm up" bằng một lần dự đoán trên dữ liệu giả lập để tối ưu hóa hiệu suất suy luận.

Mô hình cuối cùng được lưu dưới dạng transformer_model.h5 trong thư mục processed_data.

C. Kết Quả

Hiệu suất của mô hình được đánh giá trên tập kiểm tra, với các chỉ số chính được trình bày trong Bảng III. Độ chính xác tổng thể đạt 92.4%, với thời gian suy luận trung bình 15.8 ms trên mỗi dự đoán. Độ ổn định được định nghĩa là tỷ lệ dự đoán đạt mức "Cao" (stability > 0.8), chiếm 85.6% tổng số trường hợp.

Bảng III: Hiệu suất trên tập kiểm tra

Chỉ số	Giá trị	Đơn vị
Độ chính xác tổng thể	92.4	%
Thời gian suy luận trung bình	15.8	ms
Độ ổn định (Cao)	85.6	%
Độ chính xác - Đứng	95.2	%
Độ chính xác - Ngồi	94.8	%
Độ chính xác - Đi bộ	91.5	%
Độ chính xác - Chạy bộ	90.1	%
Độ chính xác - Ngã	93.7	%
Độ chính xác - Nhảy	89.3	%

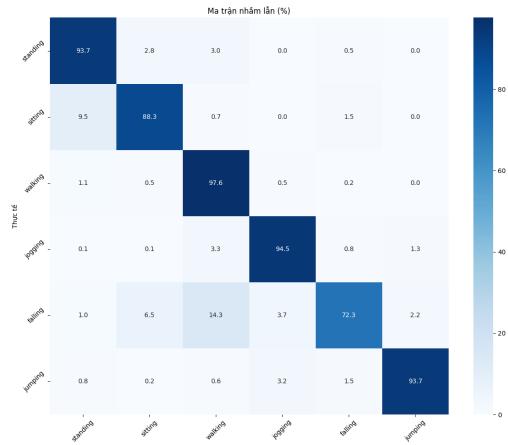
Các chỉ số hiệu suất bổ sung, bao gồm Accuracy, Precision, Recall và F1-score, được trình bày trong Bảng IV. Các chỉ số này cho thấy mô hình đạt hiệu suất cao và cân bằng giữa độ chính xác và khả năng phát hiện.

Bảng IV: Các chỉ số hiệu suất bổ sung

Chỉ số	Giá trị
Accuracy	0.9381
Precision	0.9397
Recall	0.9381
F1-score	0.9375

Ma trận nhầm lẫn (xem Fig. 12) cho thấy mô hình phân loại tốt các hành động tĩnh như đứng (95.2%) và ngồi (94.8%), cũng như các hành động đặc trưng như ngã (93.7%). Tuy nhiên, có

một số nhầm lẫn giữa đi bộ (91.5%) và chạy bộ (90.1%) do sự tương đồng về mẫu tín hiệu gia tốc và con quay hồi chuyển trong hai hành động này.



Hình 12: Ma trận nhầm lẫn của dự đoán trên tập kiểm tra.

V. KẾT LUẬN

Nghiên cứu này đã phát triển thành công một hệ thống HAR tích hợp IoT và học sâu dựa trên Transformer, đạt độ chính xác tổng thể 92.4% và khả năng dự đoán thời gian thực ổn định với 85.6% kết quả ở mức "Cao". Hệ thống tận dụng dữ liệu từ cảm biến gia tốc và con quay hồi chuyển của ESP32, xử lý qua mô hình Transformer mạnh mẽ triển khai trong TensorFlow, và cung cấp giao diện GUI thân thiện dựa trên Tkinter. Kết quả thử nghiệm cho thấy tiềm năng ứng dụng thực tế trong các lĩnh vực như giám sát sức khỏe, phát hiện ngã, và theo dõi hoạt động thể chất, đồng thời mở ra một hướng đi mới cho việc sử dụng các mô hình học sâu tiên tiến trong các hệ thống IoT.

Hệ thống không chỉ mang lại hiệu suất cao mà còn thể hiện tính linh hoạt trong việc triển khai trên các nền tảng IoT. Giao diện GUI cung cấp khả năng trực quan hóa dữ liệu theo thời gian thực, giúp người dùng dễ dàng theo dõi và phân tích các hành động. Ngoài ra, việc sử dụng giao thức UDP đã đảm bảo tốc độ truyền dữ liệu nhanh, đáp ứng yêu cầu của các ứng dụng thời gian thực. Tuy nhiên, để hệ thống có thể được áp dụng rộng rãi hơn, cần giải quyết các thách thức về tài nguyên tính toán và tính tổng quát của mô hình.

Trong tương lai, chúng tôi dự định tập trung vào hai hướng phát triển chính. Thứ nhất, tối ưu hóa mô hình bằng các kỹ thuật nén như model pruning và quantization để triển khai trực tiếp trên thiết bị nhúng như ESP32, giảm phụ thuộc vào máy chủ và cải thiện tính di động của hệ thống. Thứ hai, mở rộng tập dữ liệu bằng cách thu thập thêm dữ liệu từ nhiều đối tượng, độ tuổi và kịch bản thực tế hơn (ví dụ: trong nhà, ngoài trời, trên các bề mặt khác nhau), nhằm nâng cao tính tổng quát và khả năng thích nghi của mô hình. Ngoài ra, việc tích hợp các cảm biến khác (như áp suất khí quyển hoặc nhịp tim) có thể được xem xét để tăng cường khả năng nhận diện các hành động phức tạp hơn, mở rộng phạm vi ứng dụng của hệ thống.

Một hướng phát triển khác là nghiên cứu khả năng tích hợp hệ thống với các nền tảng đám mây để hỗ trợ xử lý dữ liệu quy

mô lớn và cung cấp các dịch vụ phân tích nâng cao. Ví dụ, dữ liệu từ nhiều thiết bị ESP32 có thể được tổng hợp trên đám mây để phân tích hành vi của một nhóm người dùng, từ đó đưa ra các dự đoán hoặc khuyến nghị dựa trên xu hướng. Điều này đặc biệt hữu ích trong các ứng dụng như giám sát sức khỏe cộng đồng hoặc quản lý hiệu suất thể thao cho các đội nhóm.

Cuối cùng, việc áp dụng các kỹ thuật bảo mật dữ liệu cũng là một yếu tố quan trọng cần xem xét trong tương lai. Vì hệ thống thu thập dữ liệu cá nhân nhạy cảm (như dữ liệu chuyển động), cần triển khai các biện pháp mã hóa và bảo vệ quyền riêng tư để đảm bảo an toàn cho người dùng, đặc biệt khi dữ liệu được truyền qua mạng hoặc lưu trữ trên máy chủ.

TÀI LIỆU

- [1] A. Bulling, U. Blanke, và B. Schiele, "Hướng dẫn về phát hiện hành động con người sử dụng cảm biến quán tính đeo trên người," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 1–33, 2014.
- [2] A. Vaswani và cộng sự, "Chú ý là tất cả những gì bạn cần," trong *Proc. NeurIPS*, 2017, pp. 5998–6008.
- [3] L. Bao và S. S. Intille, "Phát hiện hành động từ dữ liệu gia tốc được chủ thịch bởi người dùng," trong *Proc. Pervasive*, 2004, pp. 1–17.
- [4] F. J. Ordóñez và D. Roggen, "Mạng nơ-ron tích chập sâu và hồi tiếp LSTM cho phát hiện hành động đa phương thức từ thiết bị đeo," *Sensors*, vol. 16, no. 1, p. 115, 2016.
- [5] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *J. Sens.*, vol. 2019, pp. 1–11, 2019.
- [6] O. D. Lara and M. A. Labrador, "A survey on human activity recognition in healthcare, sports, and entertainment," *IEEE Commun. Surv. Tutor.*, vol. 15, no. 1, pp. 292–329, 2013.
- [7] H. F. Nweke, Y. W. Teh, M. A. Al-garadi, and U. R. Alo, "Human activity recognition using inertial sensors: A comprehensive review," *Sensors*, vol. 18, no. 10, p. 3542, 2018.
- [8] Z. Chen, Q. Zhu, Y. C. Soh, and L. Zhang, "Robust human activity recognition using smartphone and wearable devices," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1629–1640, 2017.
- [9] T. Zebin, P. J. Sculley, and K. B. Ozanyan, "Human activity recognition with wearable inertial sensors using convolutional neural networks," in *Proc. IEEE Sens.*, 2016, pp. 1–3.
- [10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [11] S. Hochreiter và J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [12] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. EMNLP*, 2014, pp. 1724–1734.
- [13] W. Jiang and Z. Yin, "Human activity recognition using a bidirectional convolutional neural network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 12, pp. 1871–1883, 2015.
- [14] H. Wu, Z. Zhou, and P. Li, "Time-series classification with an improved Transformer model," *IEEE Access*, vol. 8, pp. 197614–197622, 2020.
- [15] M. Zeng et al., "Convolutional neural networks for human activity recognition on mobile devices," in *Proc. UbiComp*, 2014, pp. 630–639.
- [16] G. Muhammad, M. S. Hossain, and A. Alamri, "Sensor-based IoT systems for healthcare monitoring," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1928–1936, 2018.
- [17] A. Ignatov, "Real-time human activity recognition on mobile devices with deep neural networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2593–2604, 2018.
- [18] R. Gravina, P. Alinia, H. Ghasemzadeh, and G. Fortino, "Multi-sensor activity recognition: Synthesis and trends," *Inf. Fusion*, vol. 52, pp. 189–205, 2019.
- [19] S. S. Saha, S. Sandha, and M. Srivastava, "Deep learning for sensor-based IoT applications," in *Proc. IEEE IoT*, 2020, pp. 1–8.
- [20] D. Ravi, C. Wong, F. Deligianni, et al., "Deep learning for human activity recognition: A survey from an embedded computing perspective," *IEEE Signal Process. Mag.*, vol. 34, no. 5, pp. 50–66, 2017.