**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

NGUYỄN NGỌC MINH

**ĐỒ ÁN 1**

**ỨNG DỤNG MÔ HÌNH MÁY HỌC VÀO BÀI TOÁN PHÂN LOẠI HOẠT ĐỘNG CỦA NGƯỜI DÙNG, TRÊN CÁC THIẾT BỊ ĐEO TAY THEO DÕI SỨC KHỎE.**

**TP. HỒ CHÍ MINH, 2021**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**NGUYỄN NGỌC MINH – 19520165**

**ĐỒ ÁN 1**

**ỨNG DỤNG MÔ HÌNH MÁY HỌC VÀO BÀI TOÁN PHÂN LOẠI HOẠT ĐỘNG CỦA NGƯỜI DÙNG, TRÊN CÁC THIẾT BỊ ĐEO TAY THEO DÕI SỨC KHỎE.**

**GIẢNG VIÊN HƯỚNG DẪN**

**PHAN ĐÌNH DUY**

**TP. HỒ CHÍ MINH, 2021**

MỤC LỤC

[Chương 1. TỔNG QUAN ĐỀ TÀI 3](#_Toc90888139)

[1.1. Các công nghệ hiện nay 3](#_Toc90888140)

[1.2. Giải pháp đề xuất 5](#_Toc90888141)

[1.3. Mục tiêu và giới hạn của đề tài: 6](#_Toc90888142)

[1.3.1. Mục tiêu của đề tài: 6](#_Toc90888143)

[1.3.2. Hạn chế: 6](#_Toc90888144)

[1.4. Ý nghĩa thực tiễn của đề tài: 6](#_Toc90888145)

[1.5. Cấu trúc của bài báo cáo đồ án: 7](#_Toc90888146)

[Chương 2. CƠ SỞ LÝ THUYẾT 8](#_Toc90888147)

[2.1. Chuẩn truyền thông nối tiếp 2 dây I2C 8](#_Toc90888148)

[2.1.1. Giới thiệu về giao tiếp I2C 8](#_Toc90888149)

[2.1.2. Đặc điểm 8](#_Toc90888150)

[2.1.3. Phần cứng 8](#_Toc90888151)

[2.1.4. Đặc điểm khung truyền dữ liệu 9](#_Toc90888152)

[2.2. Giao tiếp phần cứng: 9](#_Toc90888153)

[2.2.1. Module IMU 6 trục GY-521: 9](#_Toc90888154)

[2.2.1.1. Chức năng chính: 9](#_Toc90888155)

[2.2.1.2. Giao tiếp với MCU: 10](#_Toc90888156)

[2.2.2. OLED SSD1306 11](#_Toc90888157)

[2.3. Machine learning 11](#_Toc90888158)

[2.4. RNN – Recurrent Neural Network. LSTM – Long Short Term Memory. 14](#_Toc90888159)

[2.4.1. Định nghĩa 14](#_Toc90888160)

[2.4.2. Phân loại 15](#_Toc90888161)

[2.4.3. Mô hình bài toán 16](#_Toc90888162)

[2.4.4. Loss Function 17](#_Toc90888163)

[2.5. Thuật toán phát hiện bước chân 19](#_Toc90888164)

[2.5.1. Pha thu thập dữ liệu: 19](#_Toc90888165)

[2.5.2. Pha lọc nhiễu: 19](#_Toc90888166)

[2.5.3. Pha phát hiện đỉnh: 20](#_Toc90888167)

[2.6. Thuật toán tính khoảng cách bước chân 21](#_Toc90888168)

[2.6.1. Phương pháp 1[2] 21](#_Toc90888169)

[2.6.2. Phương pháp 2[2] 24](#_Toc90888170)

[2.7. Hệ điều hành thời gian thực FreeRTOS 24](#_Toc90888171)

[2.7.1. Tìm hiểu Hệ điều hành RTOS: 24](#_Toc90888172)

[2.7.2. Nguyên lý RTOS, Tìm hiểu API và cách sử dụng: 25](#_Toc90888173)

[Chương 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 28](#_Toc90888174)

[3.1. Sơ đồ hệ thống 28](#_Toc90888175)

[3.1.1. Các bước khởi tạo chính: 28](#_Toc90888176)

[3.1.2. Các tác vụ chính 29](#_Toc90888177)

[3.1.2.1. Tác vụ 1: Tác vụ đọc – xử lý dữ liệu thô 29](#_Toc90888178)

[3.1.2.2. Tác vụ 2 – Thực thi mạng neural: 30](#_Toc90888179)

[3.1.2.3. Tác vụ 3 – Thực hiện giải thuật đếm bước và đo độ dài bước 30](#_Toc90888180)

[3.1.2.4. Mô liên hệ giữa các tác vụ 32](#_Toc90888181)

[3.2. Sơ đồ phần cứng 33](#_Toc90888182)

[3.3. Phần mềm 34](#_Toc90888183)

[3.3.1. Training process 34](#_Toc90888184)

[3.3.2. Deploy process 36](#_Toc90888185)

[Chương 4. THỰC NGHIỆM VÀ ĐÁNH GIÁ 37](#_Toc90888186)

[4.1. Trực quan hóa dữ liệu 37](#_Toc90888187)

[4.2. Quá trình train model: 41](#_Toc90888188)

[4.3. Quá trình triển khai lên MCU 45](#_Toc90888189)

[4.3.1. Đánh giá mô hình bằng công cụ 45](#_Toc90888190)

[4.3.2. Đánh giá mô hình thực tế 47](#_Toc90888191)

[4.4. Thử nghiệm & đánh giá toàn bộ hệ thống 48](#_Toc90888192)

[4.4.1. Một số hình ảnh hệ thống 48](#_Toc90888193)

[4.4.2. Video demo 51](#_Toc90888194)

[4.4.3. Đánh giá 51](#_Toc90888195)

[Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 53](#_Toc90888196)

[5.1. Kết quả thu được thông qua quá trình thực hiện đề tài 53](#_Toc90888197)

[5.2. Hướng phát triển của đề tài 53](#_Toc90888198)

DANH MỤC HÌNH VẼ

[Hình 1.1 Machine learning embedded với sự hỗ trợ từ tensorflowlite 3](#_Toc90842997)

[Hình 1.2 Các bước giải quyết bài toán bằng học máy 4](#_Toc90842998)

[Hình 1.3 Một số cấu trúc mạng neural phổ biến 5](#_Toc90842999)

[Hình 2.1 Sơ đồ kết nối I2C 9](#_Toc90843000)

[Hình 2.2 Cấu trúc của 1 gói truyền thông 9](file:///C:\Users\LENOVO-PC\Documents\a\project1\Human-activity-recognize\docs\BAOCAO.docx#_Toc90843001)

[Hình 2.3 Sơ đồ kết nối MPU 6050 10](#_Toc90843002)

[Hình 2.4 Các nhánh của machine learning 13](#_Toc90843003)

[Hình 2.5 Mô hình RNN 15](#_Toc90843004)

[Hình 2.6 Phân loại RNN 15](#_Toc90843005)

[Hình 2.7 Mô hình mạng RNN của bài toán 16](#_Toc90843006)

[Hình 2.8 Cấu trúc cell của LSTM 17](#_Toc90843007)

[Hình 2.9 Softmax Activation 18](#_Toc90843008)

[Hình 2.10 Mối tương qua giữa Loss function và Độ chính xác của mô hình 18](#_Toc90843009)

[Hình 2.11 Độ biến thiên gia tốc ở 3 trục khi bước đi 19](#_Toc90843010)

[Hình 2.12 Giá trị độ lớn gia tốc được tính bằng phép norm 3 20](#_Toc90843011)

[Hình 2.13 Pha phát hiện bước 20](#_Toc90843012)

[Hình 2.14 Các điểm quan trọng trong quá trình phát hiện bước 21](#_Toc90843013)

[Hình 2.15 Mô hình hóa bước đi 22](#_Toc90843014)

[Hình 2.16 Mô phỏng vật thể trong không gian 22](#_Toc90843015)

[Hình 2.17 Một số công thức tính khoảng cách bước 24](#_Toc90843016)

[Hình 2.18 Task và lập lịch 25](#_Toc90843017)

[Hình 2.19 Đồng bộ sử dụng semaphore 26](#_Toc90843018)

[Hình 2.20 Đồng bộ sử dụng Mutex 26](#_Toc90843019)

[Hình 2.21 Chia sẻ dữ liệu dùng Message queue 26](#_Toc90843020)

[Hình 2.22 Chia sẻ dữ liệu sử dụng Mail queue 27](#_Toc90843021)

[Hình 3.1 Sơ đồ khởi tạo 28](file:///C:\Users\LENOVO-PC\Documents\a\project1\Human-activity-recognize\docs\BAOCAO.docx#_Toc90843022)

[Hình 3.2 Mô tả tác vụ 1 29](#_Toc90843023)

[Hình 3.3 Mô tả tác vụ 2 30](#_Toc90843024)

[Hình 3.4 Mô tả tác vụ 3 31](#_Toc90843025)

[Hình 3.5 Mô tả hệ thống 32](#_Toc90843026)

[Hình 3.6 Sơ đồ kết nối 33](#_Toc90843027)

[Hình 3.7 Kết nối thực tế 34](#_Toc90843028)

[Hình 4.1 Phân bổ lớp dữ liệu 37](#_Toc90843029)

[Hình 4.2 Mô hình thử nghiệm 1 41](file:///C:\Users\LENOVO-PC\Documents\a\project1\Human-activity-recognize\docs\BAOCAO.docx#_Toc90843030)

[Hình 4.3 Mô hình 2 42](file:///C:\Users\LENOVO-PC\Documents\a\project1\Human-activity-recognize\docs\BAOCAO.docx#_Toc90843031)

[Hình 4.4 Quá trình training 43](#_Toc90843032)

[Hình 4.5 Kết quả hàm loss 43](#_Toc90843033)

[Hình 4.6 Độ chính xác qua các epoch 44](#_Toc90843034)

[Hình 4.7 Đánh giá độ chính xác 44](#_Toc90843035)

[Hình 4.8 Confusion matrix của model 45](#_Toc90843036)

DANH MỤC BẢNG

[Bảng 2.1 Khung truyền single-read data 10](#_Toc90843037)

[Bảng 2.2 Khung truyền multiple-read data 10](#_Toc90843038)

[Bảng 2.3 Khung truyền single-write data 10](#_Toc90843039)

[Bảng 2.4 Khung truyền multiple-write data 10](#_Toc90843040)

[Bảng 2.5 Chú thích 11](#_Toc90843041)

[Bảng 2.6 Tính chất của mọi hàm số 12](#_Toc90843042)

[Bảng 4.1 Phân bổ lớp dữ liệu 37](#_Toc90843043)

DANH MỤC TỪ VIẾT TẮT

TÓM TẮT ĐỒ ÁN

Các thiết bị đeo tay theo dõi sức khỏe thông minh như smartwatch, smartband đã ra đời khá lâu nhưng chỉ thật sự gây được tiếng vang vào những năm gần đây bởi vì tiện ích mà nó đem lại, cũng như người tiêu dùng đang dần quan tâm hơn sức khỏe cá nhân.

Do đó, mục tiêu của đề tài hướng đến giải quyết các bài toán đặc thù của lĩnh vực vòng đeo tay thông minh.

1. Cung cấp giải pháp phân loại hoạt động của người dùng dựa trên hành vi, sự chuyển động trong thời gian thực (real-time).
2. Bên cạnh đó, đề tài cũng giải quyết bài toán đếm bước chân và đo khoảng cách giữa các bước để đưa ra thông tin phù hợp.

MỞ ĐẦU

Đồng hồ thông minh ra đời để nhằm hỗ trợ tối đa cho thao tác hằng ngày và thay thế một số chức năng của smartphone trong một số trường hợp mà bạn không thể tiện tay lấy điện thoại ra, ví dụ như trong khi lái xe hoặc tập thể dục.

Một trong những tính năng hàng đầu hiện nay của đồng hồ thông minh đó là giám sát các chỉ số sức khỏe của cơ thể, theo dõi vận động hàng ngày của người dùng.

Dự đoán được loại hoạt của người dùng đóng vai trò quan trọng trong việc đưa ra các cảnh báo phù hợp hoặc các lời khuyên hũu ích cho người dùng. Ví dụ:

* Cảnh báo ngồi quá lâu
* Đánh giá chất lượng giấc ngủ
* Tính toán lượng calo tiêu thụ

Các loại hoạt động của người dùng thường ngày được dùng để phân loại:

* Nằm
* Ngồi
* Đứng yên
* Đang đi
* Đang đi lên cầu thang
* Đang đi xuống cầu thang

Phương pháp dùng để phân loại là ứng dụng mô hình máy học với đầu vào là dữ liệu từ cảm biến gia tốc và con quay hồi chuyển được lấy mẫu thời gian thực.

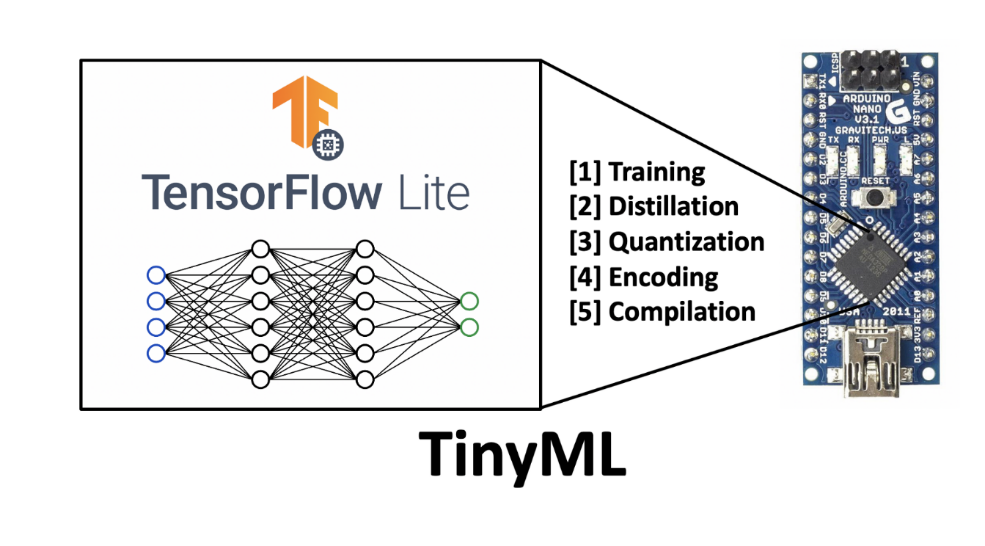
Kế tiếp là khả năng nhận dạng bước chân và ước lường khoảng cách giữa các bước. Thông tin này để đưa tới người dùng, đặt mục tiêu, khuyến khích họ vận động. Đánh giá lượng calo tiêu thụ.

# TỔNG QUAN ĐỀ TÀI

## Các công nghệ hiện nay

Ngày nay việc tích hợp các mô hình máy học, AI trực tiếp lên các thiết bị di động, hệ thống nhúng nhỏ gọn đã không còn quá xa lạ. Nó được quan tâm và được coi là một lĩnh vực quan trọng trong khoa học, được ứng dụng nhiều lĩnh vực như: y tế, giáo dục, sức khỏe, giao thông, … nâng cao đời sống, dịch vụ, tiện ích nói chung. Nhưng để máy tính có thể làm được việc tương tự như con người thì cần phải giải quyết rất nhiều bài toán cấp cao và thuật toán xử lý phù hợp. Vì vậy các khái niệm machine learning, deep learning và neural network được sinh ra để đáp ứng nhu cầu dự đoán kết quả dựa trên tập dữ liệu phức tạp. Các neural network luôn được quan tâm và phát triển để đạt kết quả tốt nhất về tốc độ xử lý và độ chính xác. Ngoài ra, các yếu tố của môi trường luôn là vấn đề cần được lưu tâm vì gây ảnh hưởng trực tiếp đến kết quả thu được từ các mạng. [[1]](https://developers.google.com/location-context/activity-recognition)

Nói một cách đơn giản, học máy là các chương trình máy tính có khả năng tự học hỏi và hoàn thành các nhiệm vụ được đề ra. Từ đó đưa ra các quy luật, mô hình toán học để giải quyết một bài toán khó mà ở đó con người gần như bất khả thi tìm ra các thuật giải.

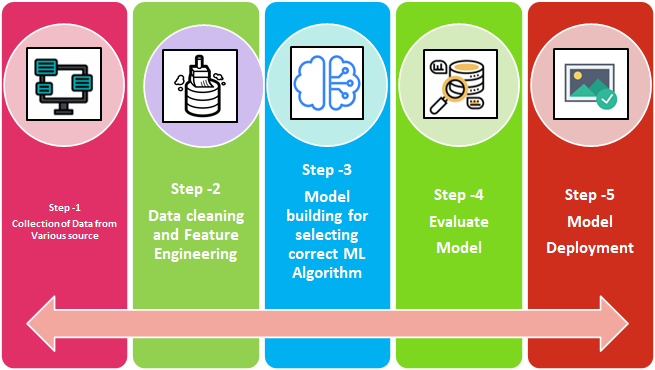


Hình . Machine learning embedded với sự hỗ trợ từ tensorflowlite

Một bài toán được ứng dụng học máy để giải quyết thông thưởng phải trải qua các bước như sau:

* Thu gom dữ liệu có input và output (label) phù hợp với yêu cầu bài toán.
* Tiền xử lý dữ liệu và trích xuất ra các đặc trưng cần thiết
* Xây dựng model với các loại mô hình toán học phù hợp
* Đánh giá model
* Triển khai model lên hệ thống đích.

Hệ thống đích của chúng ta trong trường hợp này đó là các hệ thống nhúng có MCU với bộ nhớ hạn chế, có thể có hoặc không có hệ điều hành và thiết bị di động.

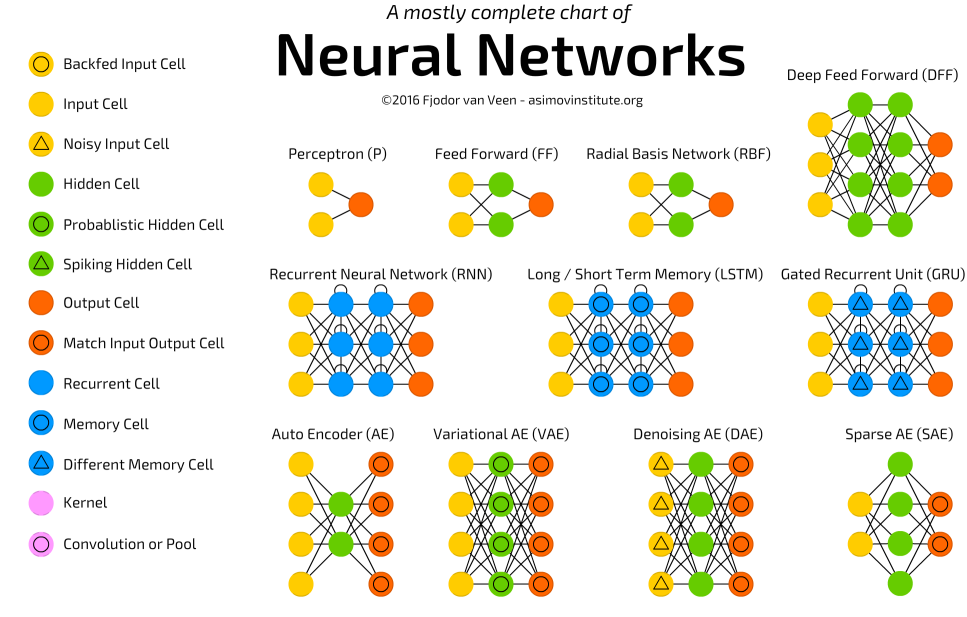


Hình . Các bước giải quyết bài toán bằng học máy

Việc triển khai một model lên các hệ thống nhúng được đơn giản hóa cho developer, với các API hỗ trợ, nhờ sự hỗ trợ của Tensorflow Keras và STM32 Platform.

Các mô hình mạng nơ ron dùng trong học máy hiện nay:

* Mạng thần kinh truyền thẳng - Feed Forward NN
* Mạng cơ sở xuyên tâm – Redial bias NN
* Mạng thần kinh hồi quy tuyến tính – Recurrent NN
* Mạng thần kinh tích chập – Convolution NN
* Bộ tự Mã hóa – Auto encoder



Hình . Một số cấu trúc mạng neural phổ biến

## Giải pháp đề xuất

Trước hết, cần làm rõ yêu cầu bài toán của đề tài mà ở đó ứng dụng học máy để

xử lý. Đầu vào dùng để xử lý là dữ liệu đọc thô từ cảm biến IMU 6 trục – 3 trục gia tốc, 3 trục con qua hồi chuyển. Từ đó cho qua mạng học sâu để dự đoán hành vi hiện tại của người dùng, với các lớp phân loại chính:

* Nằm (LAYING)
* Ngồi (SITTING)
* Đứng yên (STANDING)
* Đang đi (WALKING)
* Đang đi lên cầu thang (WALKING\_UPSTAIRS)
* Đang đi xuống cầu thang (WALKING\_DOWNSTAIRS)

Đối với loại dữ liệu dạng chuỗi được làm mới liên tục theo thời gian, sẽ rất phù hợp với các loại mạng neural có tính hồi tiếp như: Recurent Neural Network, Long/Short Term Memory, Gate Recurrent Unit ...

Đề xuất của Nhóm là sẽ sử dụng kết hợp mạng hồi quy tuyến tính, mạng tích chập, fully connected, các lớp Pooling, … thử nghiệm nhiều mô hình khác nhau, sau đó đánh giá và lựa chọn mô hình tốt nhất trên cùng tập dữ liệu. Mà tại đó mô hình phải thỏa mãn các yêu cầu của đề tài (1)

## Mục tiêu và giới hạn của đề tài:

### Mục tiêu của đề tài:

1. Cung cấp giải pháp phân loại hoạt động của người dùng dựa trên hành vi, sự chuyển động trong thời gian thực (real-time). (1)

* Độ chính xác tối thiểu 90% (test trên thiết bị nguồn)
* Tốc độ đáp ứng tối thiểu 2,5Hz trên thiết bị đích (Hệ thống nhúng độc lập sử dụng STM32F4)
* Sử dụng hiệu quả cả RAM và Flash (dưới 50%)

1. Bên cạnh đó, đề tài cũng giải quyết bài toán đếm bước chân và đo khoảng cách giữa các bước để đưa ra thông tin phù hợp.
2. Vấn đề về ổn định giá trị cảm biến thông qua các bộ lọc vật lý hoặc các thuật toán phần mềm. Bởi vì hệ thống hoạt động thời gian thực nên vấn đề lọc nhiễu cần được quan tâm tương xứng để đưa ra kết quả mong đợi.

### Hạn chế:

Đề tài chỉ tập trung giải quyết bài toán tồn đọng trong lĩnh vực thiết bị đeo thông minh, do đó sẽ không có bản dựng phần cứng hoàn chỉnh, các mô hình được đánh giá và nghiệm thu ngay trên breadboard hoặc PCB thử nghiệm.

## Ý nghĩa thực tiễn của đề tài:

Đây sẽ là giải pháp quan trọng, có thể được tích hợp trong các thiết bị thông minh theo dõi sức khỏe như điện thoại, smartwatch, smartband, … trong tương lai với sự phát triển mạnh mẽ về các phương pháp học máy, các thiết bị này hoàn toàn có thể đóng vai trò như bác sĩ cá nhân, chuyên gia sức khỏe, giám sát thể chất của người dùng.

## Cấu trúc của bài báo cáo đồ án:

Phần còn lại của bài báo cáo được tổ chức như sau, trong Chương 2 nhóm sẽ trình bày những cơ sơ lý thuyết liên quan đến đề tài . Chương 3 nhóm trình bày chi tiết các kiến trúc và phương pháp liên quan đến quy trình xây dựng mô hình dự đoán trên máy tính cá nhân và triển khai mô hình đó xuống hệ thống đích. Chương 4 nhóm sẽ trình bày các thực nghiệm, kết quả thu được từ việc train, test model và đánh giá. Cuối cùng Chương 5 nhóm đưa ra kết luận và hướng phát triển.

# CƠ SỞ LÝ THUYẾT

## Chuẩn truyền thông nối tiếp 2 dây I2C

### Giới thiệu về giao tiếp I2C

Nó là một giao thức giao tiếp được phát triển bởi Philips Semiconductors để truyền dữ liệu giữa một bộ xử lý trung tâm với nhiều IC trên cùng một board mạch chỉ sử dụng hai đường truyền tín hiệu.

Đây là một loại giao thức giao tiếp nối tiếp đồng bộ. Nó có nghĩa là các bit dữ liệu được truyền từng bit một theo các khoảng thời gian đều đặn được thiết lập bởi một tín hiệu đồng hồ tham chiếu.

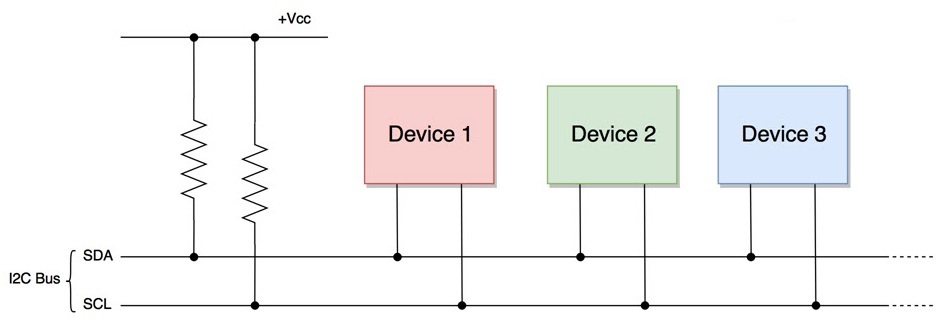
### Đặc điểm

Sau đây là một số đặc điểm quan trọng của giao thức giao tiếp I2C:

* Chỉ cần có hai đường bus (dây) chung để điều khiển bất kỳ thiết bị / IC nào trên mạng I2C
* Không cần thỏa thuận trước về tốc độ truyền dữ liệu như trong giao tiếp UART. Vì vậy, tốc độ truyền dữ liệu có thể được điều chỉnh bất cứ khi nào cần thiết
* Cơ chế đơn giản để xác thực dữ liệu được truyền
* Sử dụng hệ thống địa chỉ 7 bit để xác định một thiết bị / IC cụ thể trên bus I2C
* Các mạng I2C dễ dàng mở rộng. Các thiết bị mới có thể được kết nối đơn giản với hai đường bus chung I2C

### Phần cứng

Bus I2C (dây giao tiếp) chỉ gồm hai dây và được đặt tên là Serial Clock Line (SCL) và Serial Data Line (SDA). Dữ liệu được truyền đi được gửi qua dây SDA và được đồng bộ với tín hiệu đồng hồ (clock) từ SCL. Tất cả các thiết bị / IC trên mạng I2C được kết nối với cùng đường SCL và SDA như sau:



Hình . Sơ đồ kết nối I2C

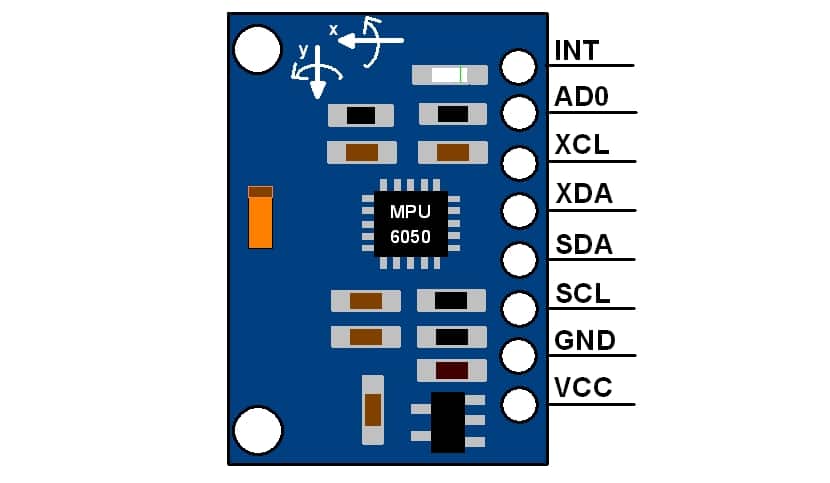
### Đặc điểm khung truyền dữ liệu

Hình . Cấu trúc của 1 gói truyền thông

## Giao tiếp phần cứng:

### Module IMU 6 trục GY-521:

* + - 1. Chức năng chính:
* Cảm biến chuyển động là một trong những loại cảm biển quan trọng, được tích hợp trong vô số các thiết bị từ smartphone, máy bay, tàu vũ trụ, máy ảnh, … đem đến trải nghiệm tốt hơn cho các game, chống rung ảnh số, cho đến định hướng đường bay, theo dõi sức khỏe.
* Module GY-521 chứa cảm biến chính MPU-6050 là thiết bị tich hợp 6 trục theo dõi chuyển động, 3 trục theo dõi gia tốc(accelerometer), 3 trục theo dõi tốc độ góc quay(gyroscope).
* Với độ phân giải 16 bit ở cả 6 trục.



Hình . Sơ đồ kết nối MPU 6050

* + - 1. Giao tiếp với MCU:
* Sử dụng phương thức giao tiếp I2C với tốc độ tối đa đạt 400KHz
* Địa chỉ giao tiếp: 0x68
* Quy ước khung truyền đọc đơn dữ liệu:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Master | S | AD+W |  | RA |  | S | AD+R |  |  | NACK | P |
| Slave |  |  | ACK |  | ACK |  |  | ACK | DATA |  |  |

Bảng 2.1 Khung truyền single-read data

* Quy ước khung truyền đọc đa dữ liệu:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Master | S | AD+W |  | RA |  | S | AD+R |  |  | ACK |  | NACK | P |
| Slave |  |  | ACK |  | ACK |  |  | ACK | DATA |  | DATA |  |  |

Bảng 2.2 Khung truyền multiple-read data

* Quy ước khung truyền ghi đơn dữ liệu:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Master | S | AD+W |  | RA |  | DATA |  | P |
| Slave |  |  | ACK |  | ACK |  | ACK |  |

Bảng 2.3 Khung truyền single-write data

* Quy ước khung truyền ghi đa dữ liệu:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Master | S | AD+W |  | RA |  | DATA |  | DATA |  | P |
| Slave |  |  | ACK |  | ACK |  | ACK |  | ACK |  |

Bảng 2.4 Khung truyền multiple-write data

|  |  |
| --- | --- |
| Signal | Mô Tả |
| S | Start condition |
| AD | Địa chỉ |
| W | Write bit (0) |
| R | Read bit(1) |
| ACK  NACK | Acknowledge  Not-Acknowledge |
| RA  DATA | Địa chỉ thanh ghi muốn tương tác  Dữ liệu đọc hay ghi |
| P | Stop condition |

Bảng 2.5 Chú thích

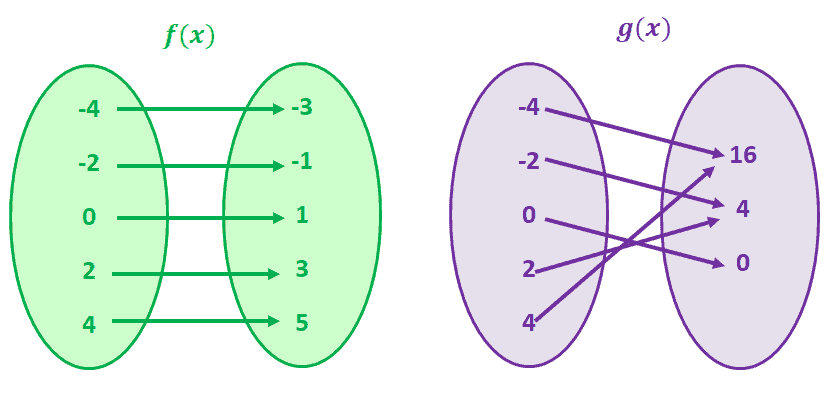
### OLED SSD1306

* Được dùng để debug, hiển thị kết quả dự đoán từ mạng học sâu. Nghiệm thu kết quả.
* Màn hình xanh đơn sắc
* Số điểm ảnh: 128x64
* Sử dụng chuẩn giao tiếp I2C
* Địa chỉ giao tiếp: 0x3C
* Có 2 chế độ sử dụng: đọc/ghi command (dùng để cấu hình), đọc/ghi data (dùng để hiển thị)
* Độ sáng cấu hình trong khoảng: Từ 0 đến 255.

## Machine learning

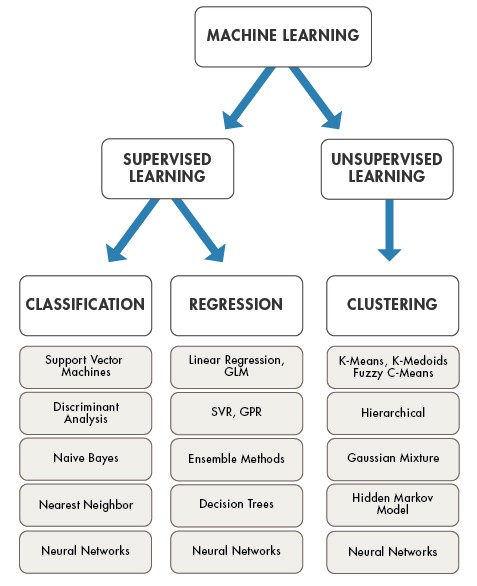
Machine Learning là một thuật ngữ rộng để chỉ hành động bạn dạy máy tính cải thiện một nhiệm vụ mà nó đang thực hiện. Cụ thể hơn, machine learning đề cập tới bất kỳ hệ thống mà hiệu suất của máy tính khi thực hiện một nhiệm vụ sẽ trở nên tốt hơn sau khi hoàn thành nhiệm vụ đó nhiều lần. Hay nói cách khác, khả năng cơ bản nhất của machine learning là sử dụng thuật toán để phân tích những thông tin có sẵn, học hỏi từ nó rồi đưa ra quyết định hoặc dự đoán về một thứ gì đó có liên quan. Thay vì tạo ra một phần mềm với những hành động, hướng dẫn chi tiết để thực hiện một nhiệm vụ cụ thể, máy tính được “huấn luyện” bằng cách sử dụng lượng dữ liệu và các thuật toán để học cách thực hiện nhiệm vụ.

Dưới góc nhìn của Machine Learning, mọi bài toán đều có input và output, ở giữa của quy trình đó là một hàm số rất phức tạp. Hàm số đó là kết quả của quá trình học tập của máy tính. Hàm là một tập hợp các quy tắc được dùng để các phần tử ở tập nguồn đến các phần tử ở tập đích sao cho mỗi một phần tử ở tập nguồn chỉ có một và chỉ một phần tử ở tập đích.



Bảng 2.6 Tính chất của mọi hàm số

Machine learning được chia làm 2 nhánh chính là Supervised Learning (học có giám sát) và Un-Supervised Learning (học không giám sát). Trong cả 2 nhánh đều có các mạng thần kinh nơ ron thích hợp.



Hình . Các nhánh của machine learning

Supervised learning là thuật toán dự đoán đầu ra (outcome) của một dữ liệu mới (new input) dựa trên các cặp (input, outcome) đã biết từ trước. Cặp dữ liệu này còn được gọi là (data, label), tức (dữ liệu, nhãn). Supervised learning là nhóm phổ biến nhất trong các thuật toán Machine Learning.

Regression(Bài toán hồi quy). Hồi quy là một thuật ngữ rất khó hiểu. Hiểu đơn giản thì những cái gì có tính liên tục và tiếp nối với nhau thì người ta gọi là là hồi quy. Vậy khi tập đích trong tập train là một tập dữ liệu có dạng liên tục không thể phân thành nhóm mà là một dữ liệu cụ thể thì bài toán được xếp vào dạng hồi quy. Bài toán tiêu biểu cho dạng này thường được ví dụ là tính toán giá cả của sản phẩm dựa trên thông số trước đó, hoặc dự đoán biến động tài chính...

Classification (Bài toán phân loại). Một bài toán được gọi là classification nếu các label của input data được chia thành một số hữu hạn nhóm. Ví dụ: Gmail xác định xem một email có phải là spam hay không; các hãng tín dụng xác định xem một khách hàng có khả năng thanh toán nợ hay không.

Unsupervised Learning.Trong thuật toán này, chúng ta không biết được outcome hay nhãn mà chỉ có dữ liệu đầu vào. Thuật toán unsupervised learning sẽ dựa vào cấu trúc của dữ liệu để thực hiện một công việc nào đó. Một cách toán học, Unsupervised learning là khi chúng ta chỉ có dữ liệu vào X mà không biết nhãn Y tương ứng.

Clustering (Bài toán gom nhóm). Một bài toán phân nhóm toàn bộ dữ liệu X thành các nhóm nhỏ dựa trên sự liên quan giữa các dữ liệu trong mỗi nhóm. Ví dụ: phân nhóm khách hàng dựa trên hành vi mua hàng.

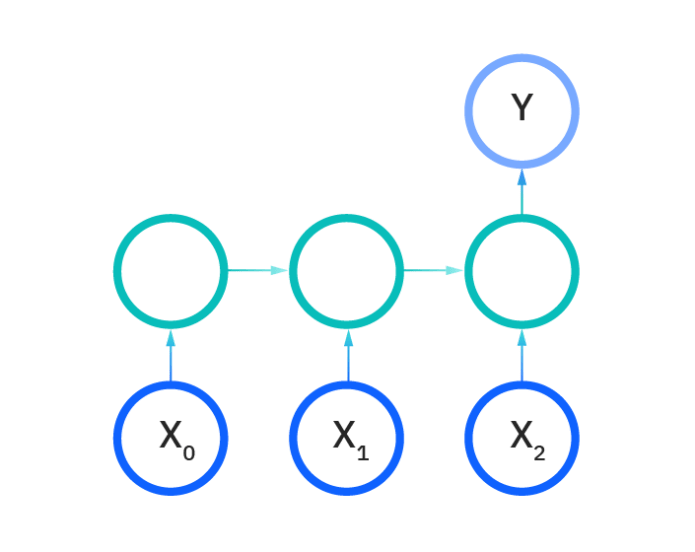
## RNN – Recurrent Neural Network. LSTM – Long Short Term Memory.

### Định nghĩa

Deep learning có 2 mô hình lớn là Convolutional Neural Network (CNN) cho bài toán có input là ảnh và Recurrent neural network (RNN) cho bài toán dữ liệu dạng chuỗi (sequence data).

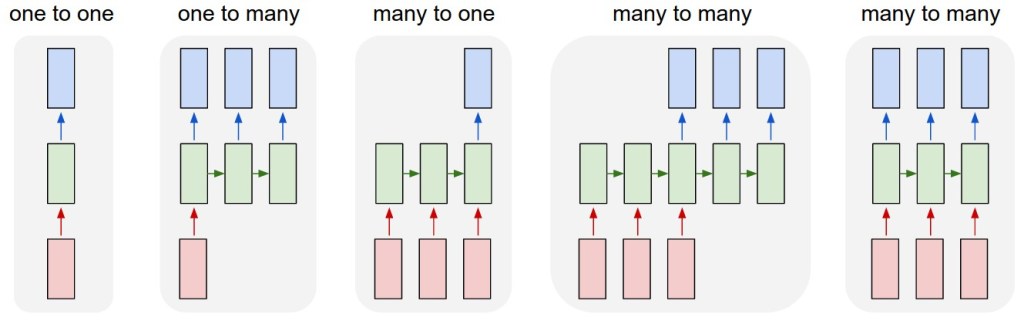
Hãy hình dung, để phân loại hoạt động của người dùng vào các nhóm cho trước (Đứng, ngồi, đi bộ, đi lên cầu thang, đi xuống cầu thang, nằm) thì phải dựa trên chuỗi giá trị chuyển động của người dùng. Không thể chỉ sử dụng 1 giá trị chuyển động duy nhất tại 1 thời điểm để đưa ra dự đoán. Đó là điều không thể chính xác.

Và như vậy RNN ra đời với ý tưởng chính là sử dụng một bộ nhớ để lưu lại thông tin từ từ những bước tính toán xử lý trước để dựa vào nó có thể đưa ra dự đoán chính xác nhất cho bước dự đoán hiện tại.



Hình . Mô hình RNN

### Phân loại



Hình . Phân loại RNN

One to one: 1 input và 1 output.

One to many: bài toán có 1 input nhưng nhiều output.

Many to one: bài toán có nhiều input nhưng chỉ có 1 output.

Many to many: bài toán có nhiều input và nhiều output.

Mô hình sử dụng cho đồ án: many to one (nhiều input – một output)

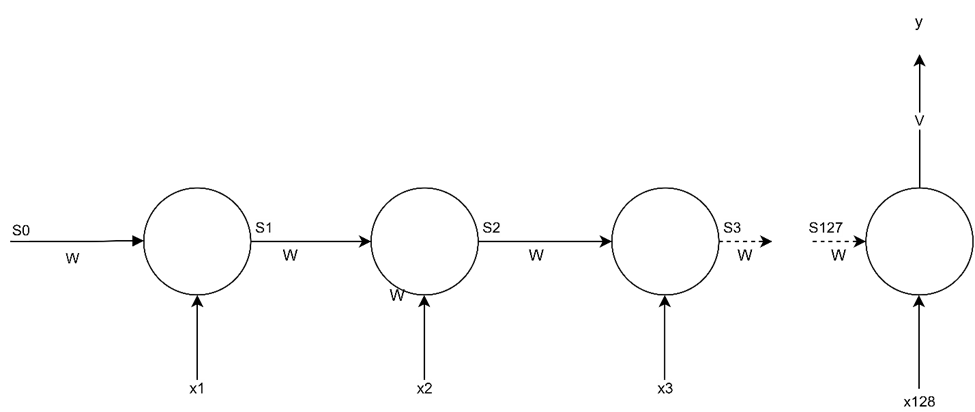
Đối với bài toán “phân loại hoạt động của người dùng” thì input của ta là 9 features (3 trục gia tốc tổng thể, 3 trục tốc độ quay, 3 trục body motion)

### Mô hình bài toán

Bài toán: Phân loại hoạt động của người dùng dựa trên 128 giá trị liên tục mới nhất lấy từ cảm biến.

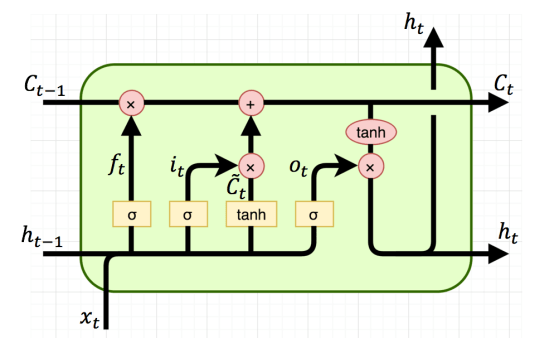
Do đó, đặc điểm của mô hình sẽ có:

* Mô hình có 128 input và 1 output
* Mỗi S phía sau sẽ đem toàn bộ thông tin từ S phía trước
* Trong bài toán phân loại nên ở bước cuối đầu ra V sẽ cho qua activation là softmax y = g(S128\*V)



Hình . Mô hình mạng RNN của bài toán

Trên thực tế người ta không còn sử dụng mô hình RNN mà sử dụng phiên bản nâng cấp của nó là LSTM để tránh hiện tượng vanishing gradient. Nghĩa là, các state ở sau sẽ không học được dữ liệu ở xa nó.

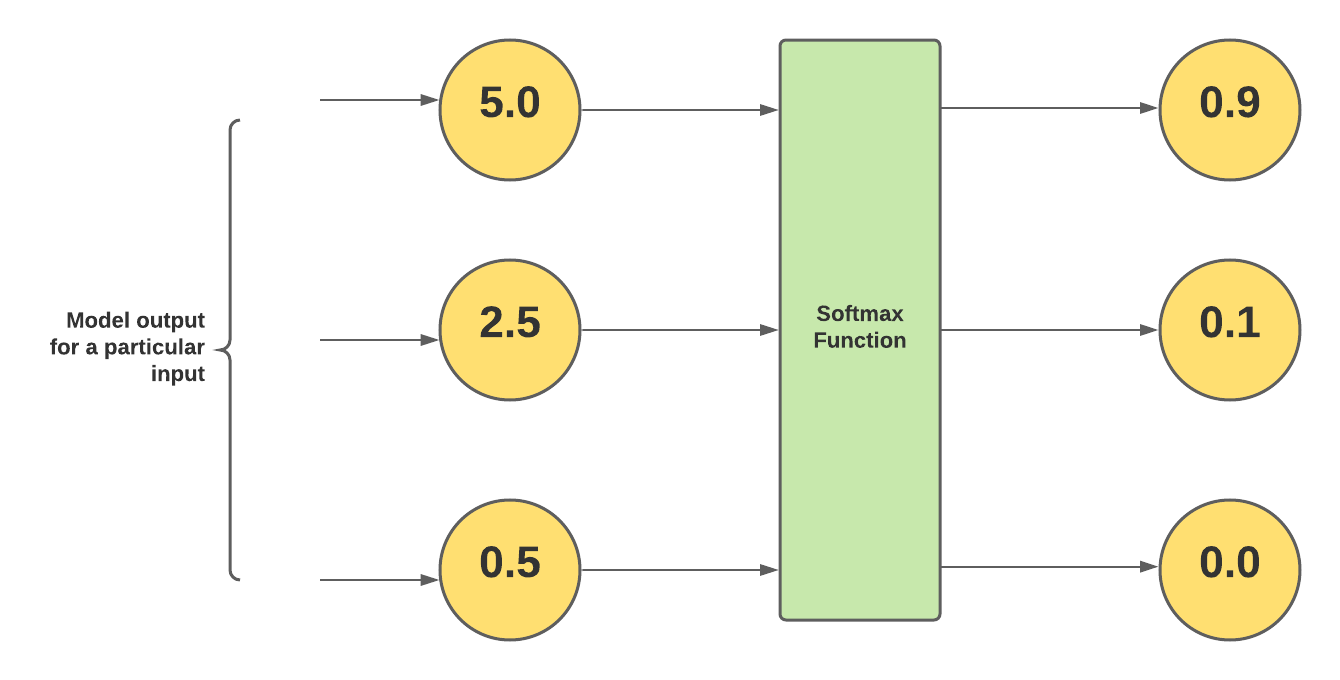
Về cơ bản, LSTM có kiến trúc khá giống RNN nhưng ở các cell được nâng cấp giúp lưu trữ thông tin tốt hơn. 

Hình . Cấu trúc cell của LSTM

### Loss Function

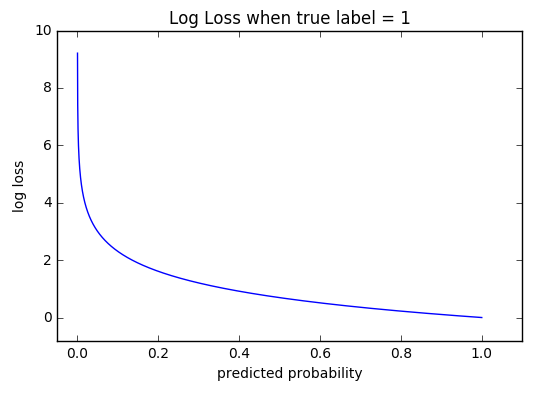
Loss function của cả mô hình bằng tổng loss của mỗi output, tuy nhiên ở mô hình trên chỉ có 1 output và là bài toán phân loại nên categorical cross entropy loss sẽ được sử dụng.

Cross entropy loss thường được dùng với Softmax và Sigmoid activation function điều này giúp cho ngõ ra có giá trị tổng cộng luôn bằng 1, phù hợp với bài toán đưa ra xác suất giữa các lớp.



Hình . Softmax Activation

Khả năng dự đoán càng chính xác khi hàm loss càng tiến về 0.



Hình . Mối tương qua giữa Loss function và Độ chính xác của mô hình

## Thuật toán phát hiện bước chân

Bao gồm 3 pha làm việc:

### Pha thu thập dữ liệu:

* Trong suốt pha này, sensor sẽ thu được giá trị gia tốc trên 3 trục x, y, z.

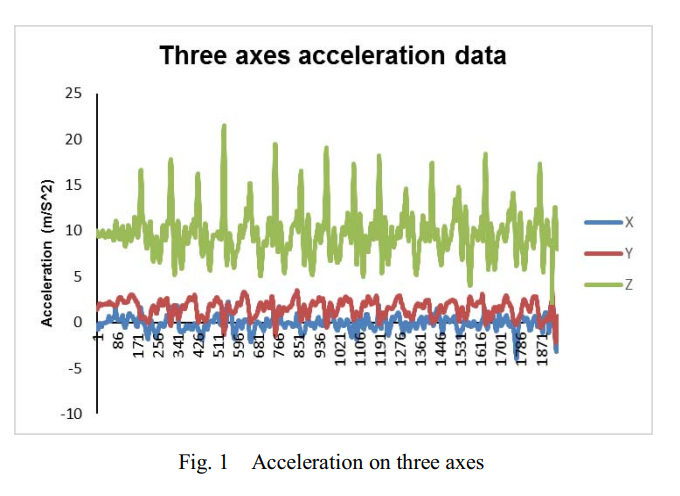
Sau đó tính ra giá trị độ lớn theo quy tắc:

* Từ đó loại bỏ hiệu ứng từ trường bằng cách:

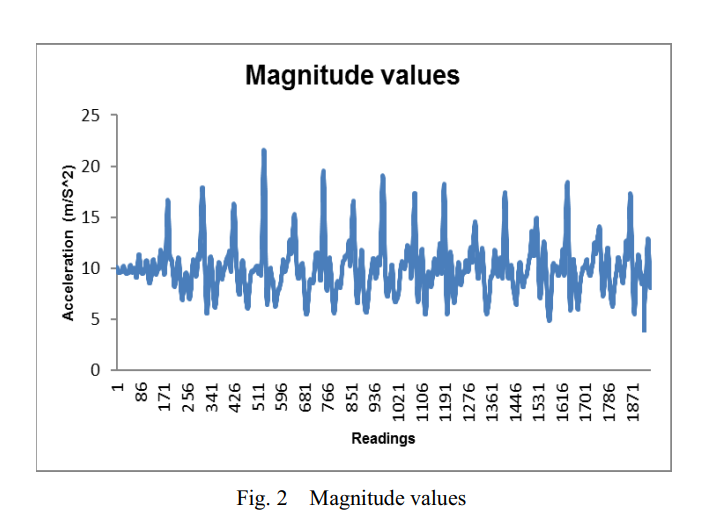
thu được gia tốc chuyển động của vật thể.

### Pha lọc nhiễu:

* Thông thường các gia tốc kế cho ra giá trị nhiễu loạn liên tục, bởi vì cảm biến khá nhạy. Do đó việc lọc nhiễu là cần thiết để cho ra giá trị mượt hơn.
* Việc lọc nhiễu đóng góp vào độ chính xác của thuật toán. Công đoạn chọn đỉnh cũng xảy ra suôn sẻ hơn.
* Bộ lọc kalman sẽ được sử dụng đưa ra kết quả dự đoán.



Hình . Độ biến thiên gia tốc ở 3 trục khi bước đi



Hình . Giá trị độ lớn gia tốc được tính bằng phép norm 3

### Pha phát hiện đỉnh:

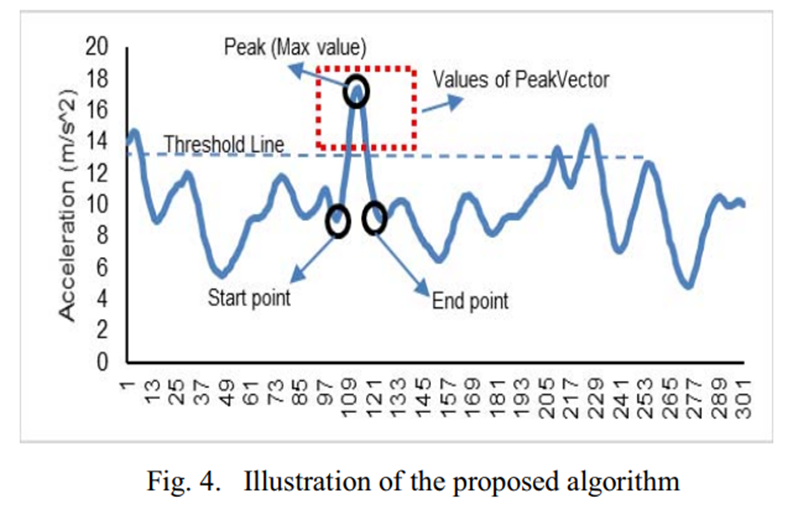
* Suốt quá trình chuyển động, giá trị độ lớn gia tốc đo đạc được thay đổi liên tục.
* Hầu hết các giải thuật hiện nay dùng phương pháp đặt ngưỡng. Nếu độ lớn gia tốc vượt giá trị ngưỡng thì bước được phát hiện.
* Tuy nhiên trong 1 số trường hợp giải thuật trên có thể bị sai lệch so với thực tế.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình . Pha phát hiện bước

* Vì thế, có đề xuất phát hiện bước cải tiến như sau:
  + Dữ liệu từ cảm biến được đọc bởi một cửa sổ trượt window có độ dài là window size
  + Thuật toán bắt đầu theo dõi các giá trị chứa trong window.
  + Nếu 3 giá trị liên tiếp nhau tăng dần thì điều đó báo hiệu có thể có 1 Peak phía trước. Và điểm đầu tiên trong 3 điểm gọi là Start point.
  + Tiếp tục quét các điểm ở phía sau. Nếu có giá trị lớn hơn ngưỡng (đã đặt trước đó) thì gom chúng vào một vector gọi là Peak Vector.
  + Lấy giá lớn nhất trong Peak Vector, giá trị đó là giá trị biểu diễn cho 1 bước thật sự.

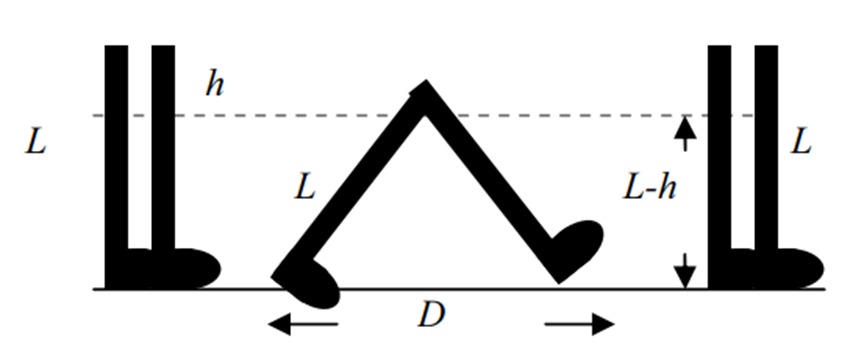


Hình . Các điểm quan trọng trong quá trình phát hiện bước

## Thuật toán tính khoảng cách bước chân

### Phương pháp 1[2]

* Khi bước cơ thể chúng ta di chuyển theo chiều dọc từ dưới lên hoăc từ trên xuống, gần giống với dao động điều hòa của chất điểm.



Hình . Mô hình hóa bước đi

* Nếu chúng ta giả sử chiều dài chân là L, thì ta có độ dài từ thắt lưng xuống đất nằm trong khoảng từ L đến (L-h) với h phần dịch chuyển của thắt lưng khi di chuyển.
* Với L đã biết, có thể dùng công thức toán học để tích chiều dài bước D nếu đo đạc được h.
* Ý tưởng tính h, tích phân 2 lớp gia tốc theo chiều dọc thu được tổng quãng đường chuyển động cơ thể theo theo chiều dọc.
* Giả sử ta đặt cảm biến có trục x (cảm biến có 3 trục) hợp với phương thẳng đứng một góc Ө ( 0 < Ө < 180).

Ảnh có chứa mũi tên

Mô tả được tạo tự động

Hình . Mô phỏng vật thể trong không gian

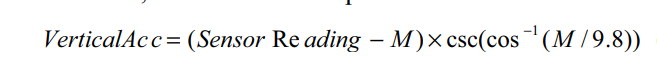
* Dữ liệu từ cảm biến bao gồm gia tốc từ trường và gia tốc của **External Force** (gia tốc gây chuyển động cho vật).
* Do đó để tính được gia tốc của **External**

Ảnh có chứa văn bản

Mô tả được tạo tự độngẢnh có chứa văn bản

Mô tả được tạo tự động

* Trong đó:
  + M: là giá trị cảm biến trục x, lúc ổn định vật.
  + gravity = 9,8 m/s2
* Cuối cùng ta có công thức tổng quát tính gia tốc gây ra chuyển động cho vật:

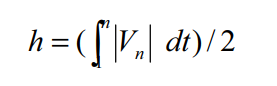


* Tính tích phân gia tốc thu được vận tốc:

Ảnh có chứa văn bản, đồng hồ

Mô tả được tạo tự động

* Tính tích phân vận tốc thu được quãng đường:



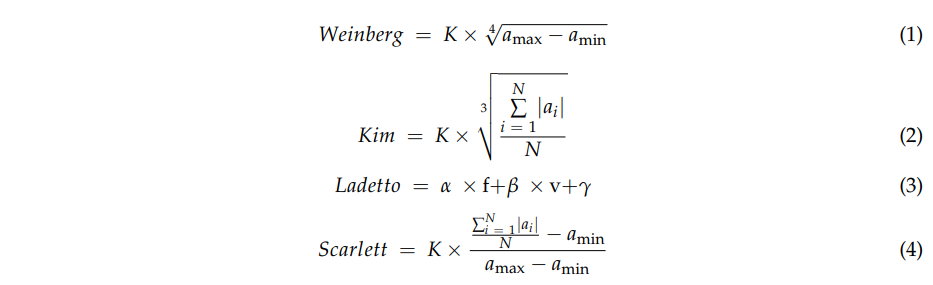
* Sau đó tính độ dài bước D:

Ảnh có chứa văn bản, đồng hồ

Mô tả được tạo tự động

### Phương pháp 2[2]

* Nhìn chung phương pháp 2 khá đơn giản, công thức được đúc kết từ thực nghiệm của một số tác giả:



Hình . Một số công thức tính khoảng cách bước

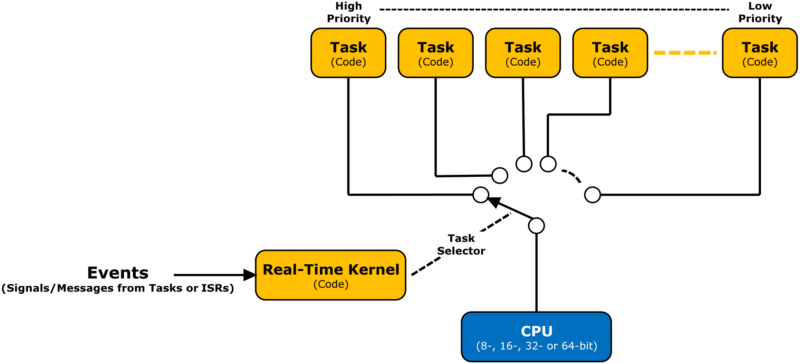
## Hệ điều hành thời gian thực FreeRTOS

### Tìm hiểu Hệ điều hành RTOS:

* Một hệ thống nhúng lớn phải thực thi nhiều tác vụ thì cần thiết có một hệ điều hành để lập lịch cũng như phân vùng, sử dụng bộ nhớ hiệu quả.
* Đối với STM32F4 (arm cortex M4) platform có tới 5 loại hệ điều hành khác nhau: μClinux, μC / OS-II, eCos, FreeRTOS, Hệ điều hành Dujiangyan.
* Trong đó, FreeRTOS nổi tiếng với mã nguồn mở, miễn phí, dễ sử dụng.
* Các ứng dụng cần RTOS
  + Ứng dụng nhiều trạng thái máy (States Machine)
  + Ứng dụng lớn
  + Ứng dụng liên quan tới các tác vụ xử lý nhanh, xử lý ảnh, âm thanh.
* Lợi ích khi sử dụng OS nói chung và RTOS nói riêng:
  + Chia sẻ tài nguyên một cách đơn giản: cung cấp cơ chế để phân chia các yêu cầu về bộ nhớ và ngoại vi của MCU
  + Dễ debug và phát triển: Mọi người trong nhóm có thể làm việc một cách độc lập, các lập trình viên thì có thể tránh được các tương tác với ngắt, timer, với phần cứng (cái này mình không khuyến khích lắm vì hiểu được phần cứng vẫn sẽ tốt hơn nhiều)
  + Tăng tính linh động và dễ dàng bảo trì: thông qua API của RTOS,…
* Vì thế, sử dụng RTOS trong đồ án giúp tối ưu cũng như dễ mở rộng và nâng cấp.

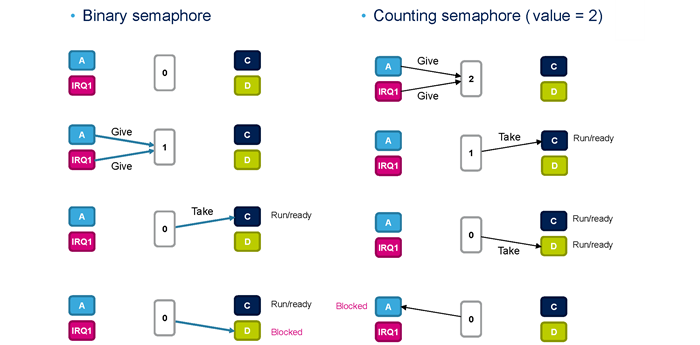
### Nguyên lý RTOS, Tìm hiểu API và cách sử dụng:

* RTOS là một phân đoạn hoặc một phần của chương trình, trong đó nó giải quyết việc điều phối các task, lập lịch và phân mức ưu tiên cho task, nắm bắt các thông điệp gửi đi từ task.
* **Task** là một đoạn chương trình thực thi một hoặc nhiều vấn đề gì đó, được Kernel quản lý.



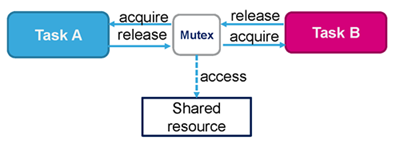
Hình . Task và lập lịch

* Cơ chế Đồng bộ:
  + Semaphores – Truy xuất tài nguyên liên tục từ các task khác nhau



Hình . Đồng bộ sử dụng semaphore

* + Mutex – Đồng bộ hóa truy cập tài nguyên sử dụng Mutual Exclusion



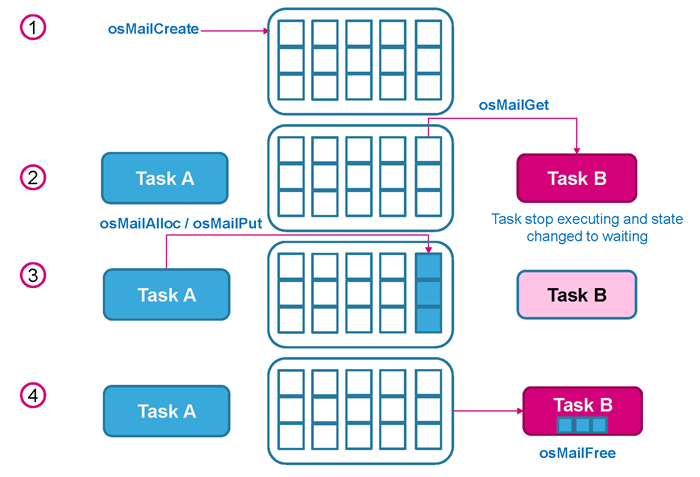
Hình . Đồng bộ sử dụng Mutex

* Cơ chế giao tiếp:
  + Message queue – Trao đổi tin nhắn giữa các task trong hoạt động giống như FIFO



Hình . Chia sẻ dữ liệu dùng Message queue

* + Mail queue – Trao đổi dữ liệu giữa các task sử dụng hằng đợi của khối bộ nhớ

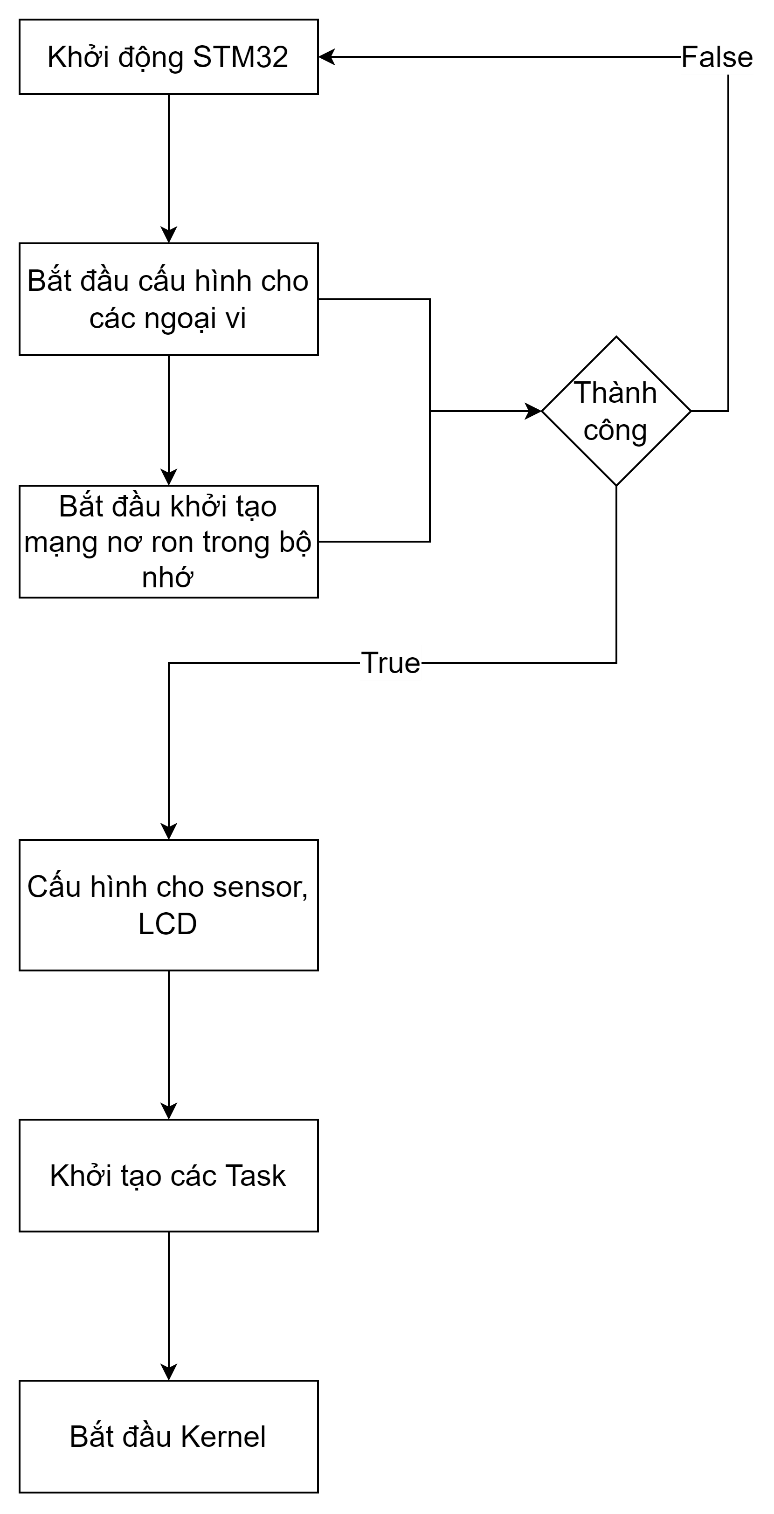


Hình . Chia sẻ dữ liệu sử dụng Mail queue

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Sơ đồ hệ thống

### Các bước khởi tạo chính:

* Khởi động STM32

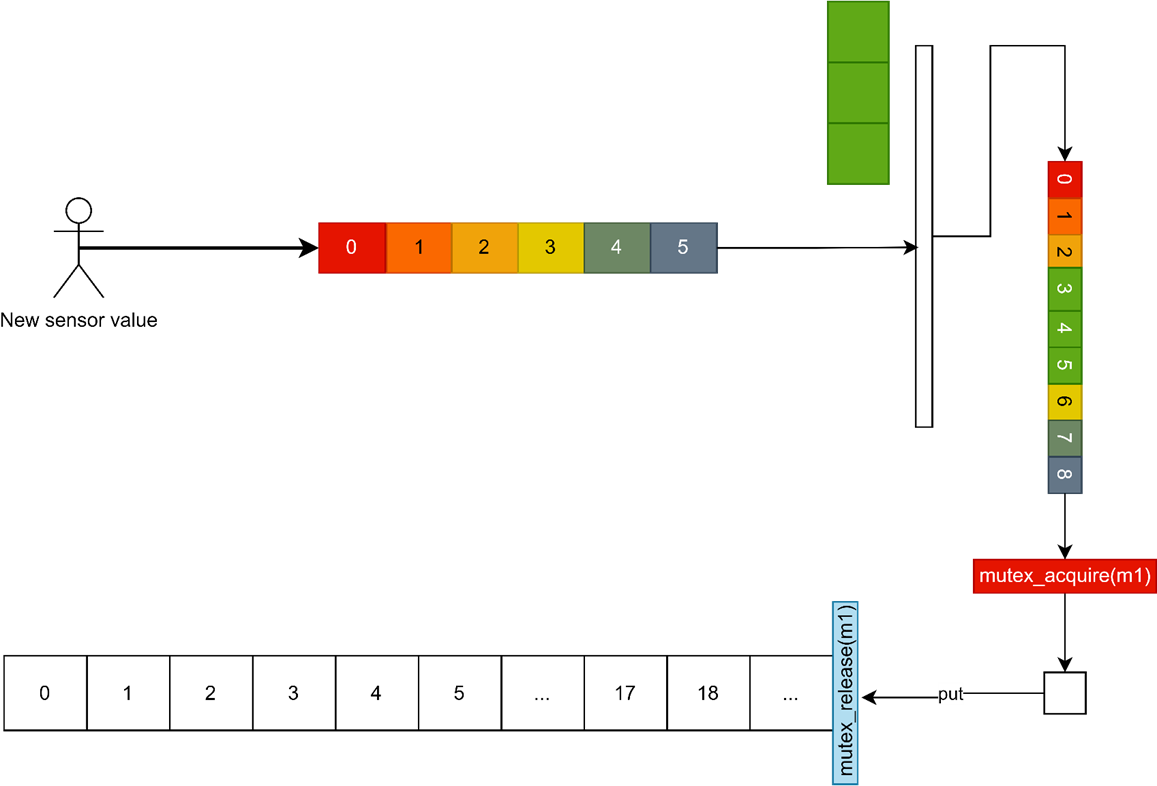
Hình . Sơ đồ khởi tạo

* Cấu hình cho các ngoại vi như I2C, Timer, GPIO… để sử dụng.
* Khởi động mạng nơ ron đã được train trước đó bằng file model.h
* Cấu hình cho sensor MPU6050 và OLED SSD1306
* Khởi tạo 3 task chính, các semophore, mutex, hàng đợi chia sẻ dữ liệu.
* Bắt đầu Kernel, khởi chạy các dịch vụ của hệ điều hành, các Task được lập lịch trước đó cũng bắt đầu chạy.

### Các tác vụ chính

* Bao gồm 3 tác vụ chính:
  + Task 1: Tác vụ lấy data từ cảm biến và xử lý, sau đó cho vào hàng đợi dữ liệu chung (message queue).
  + Task 2: Tác vụ lấy toàn bộ data từ hàng đợi chung message queue, đưa toàn bộ vào input của neural để đưa ra kết quả dự đoán.
  + Task 3: Tác vụ detect bước chân và đo khoảng cách. Chỉ chạy khi có 30 dữ liệu mới.

#### Tác vụ 1: Tác vụ đọc – xử lý dữ liệu thô

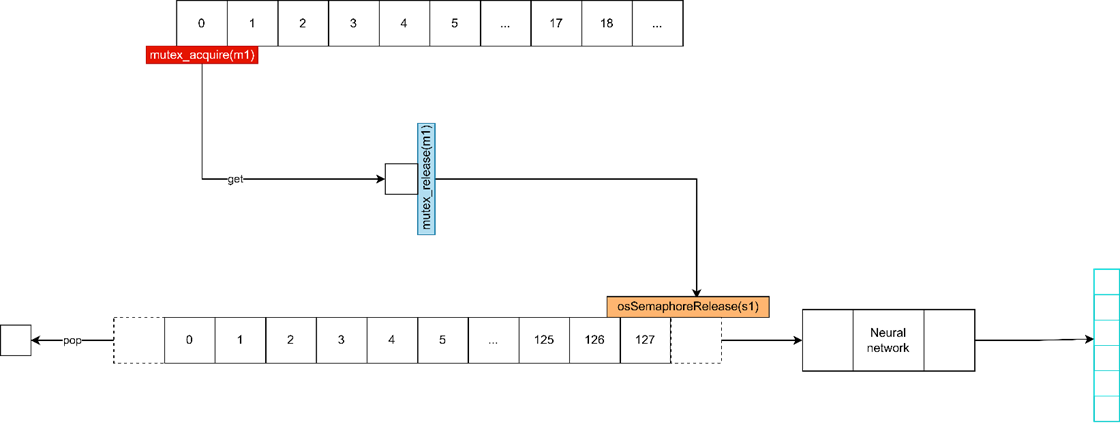


Hình . Mô tả tác vụ 1

* Tác vụ này có nhiệm vụ chính đó là đọc liên tục thông tin từ cảm biến cung cấp thông qua chuẩn giao tiếp I2C, bao gồm 6 giá trị cần sử dụng ( 3 giá trị gia tốc trên 3 trục x, y, z. 3 giá trị tốc độ góc quay trên 3 trục x, y, z).
* Sau đó, xử lý dữ liệu để thu được thêm 3 giá trị chuyển động của cơ thể gọi là body motion.
* Để đưa toàn bộ 9 giá trị đó vào hàng đợi chia sẻ dữ liệu giữa các tác vụ, chúng ta cần có cơ chế bảo vệ vùng tranh chấp. Ở đây sử dụng Mutex – Đồng bộ hóa truy cập tài nguyên Mutual Exclusion
* Đầu tiên, chúng ta yêu cầu đi vào vùng tranh chấp. Tiếp đó, đưa dữ liệu vào hàng đợi. Cuối cùng, yêu cầu thoát khỏi vùng tranh chấp.

#### Tác vụ 2 – Thực thi mạng neural:

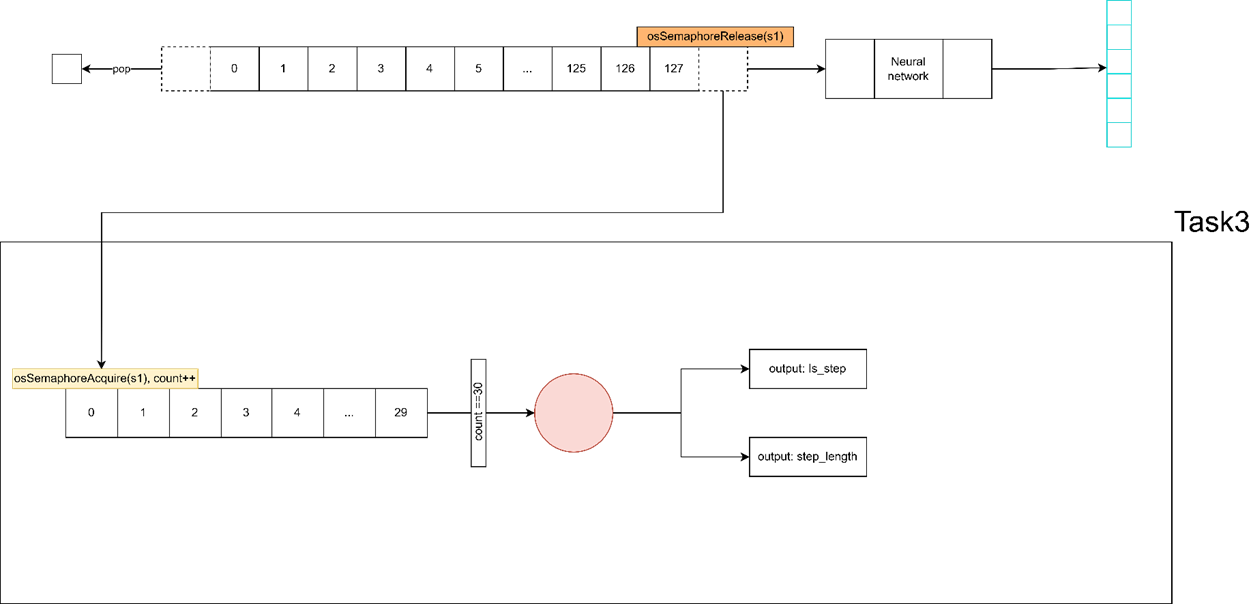
* Tác vụ này có nhiệm vụ chính đó là lấy toàn bộ dữ liệu mới nhất từ hàng đợi chia sẻ, từ đó chạy mạng nơ ron để lấy kết quả dự đoán loại hoạt động của người dùng.
* Trước đó, để lấy được dữ liệu từ hàng đợi chia sẻ, tác vụ cần yêu cầu đi vào vùng tranh chấp. Sau khi lấy ra toàn bộ dữ liệu thì thoát khỏi vùng tranh chấp.
* Với từng giá trị được lấy ra thì tương ứng Producer tạo ra một tài nguyên mới để tác vụ 3 sử dụng sau đó.
* Tác vụ 2 chiếm dụng CPU nhất, tốn xấp xỉ 100 ms (với cortex M4 100MHz).



Hình . Mô tả tác vụ 2

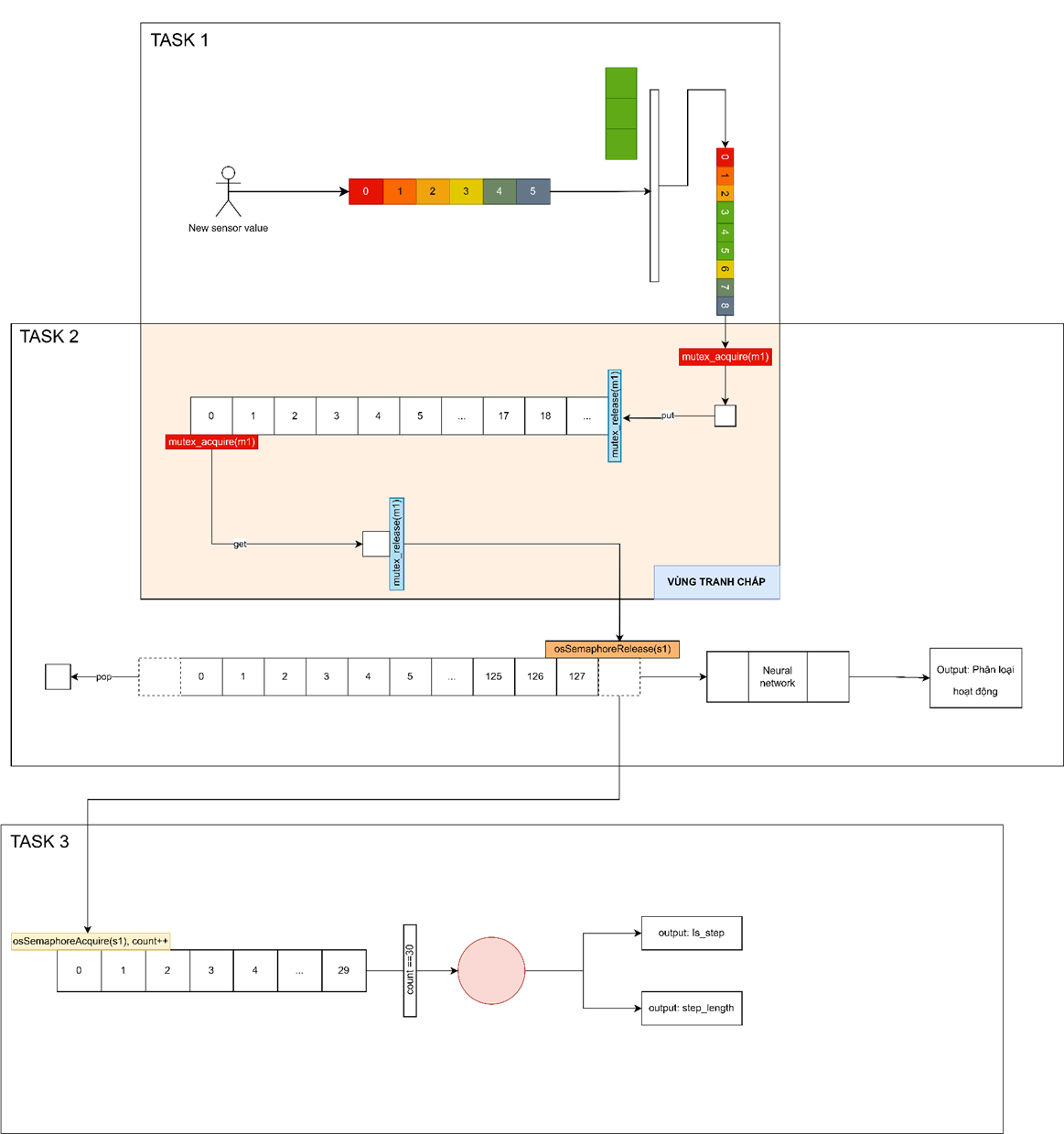
#### Tác vụ 3 – Thực hiện giải thuật đếm bước và đo độ dài bước

* Tác vụ này có nhiệm vụ chính là phát hiện bước chân của người dùng và đo lường độ dài của bước chân đó.
* Sử dụng giải thuật đã giới thiệu trước đó.
* Tác vụ này yêu cầu 30 tài nguyên mỗi lần chạy, tài nguyên này được tạo mới từ Tác vụ 2.



Hình . Mô tả tác vụ 3

#### Mô liên hệ giữa các tác vụ

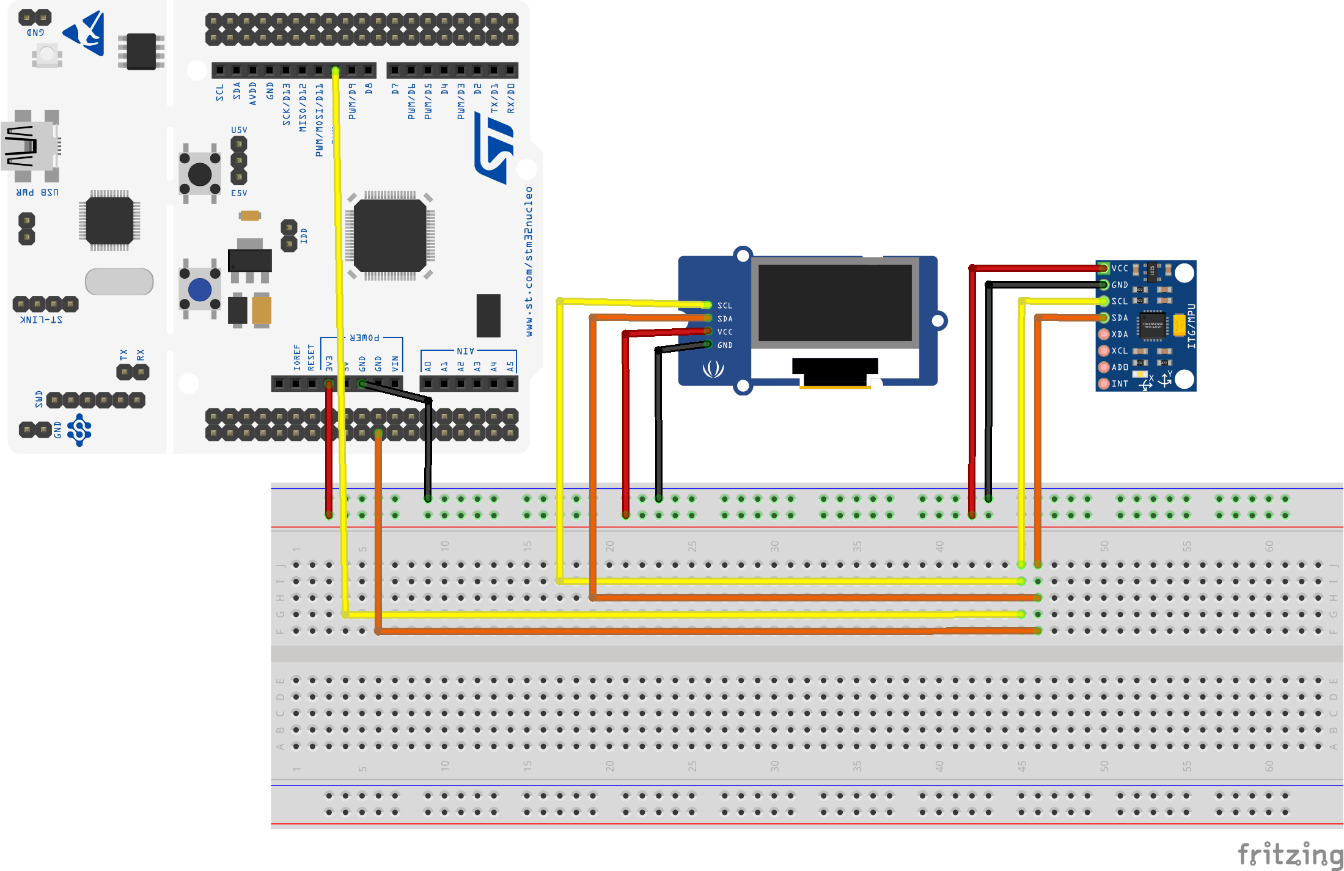


Hình . Mô tả hệ thống

* Cơ chế giao tiếp task: sử dụng message queue để trao đổi dữ liệu giữa task 1 & 2.
* Cơ chế đồng bộ:
  + Bảo vệ vùng tranh chấp (message queue): Mutex giữa task 1 & 2.
  + Yêu cầu 30 dữ liệu mới để chạy task 3: semaphore. Producer ở task 2 & Consumer ở task 3.

## Sơ đồ phần cứng

STM32F411: Bộ xử lý trung tâm

* Model: BOARD NUCLEO-F411RE
* CPU: Single-core ARM cortex-m4 @100Mhz
* Memory: SRAM 128 KB
* Storage: Flash 512KB
* USB: Micro-B
* GPIO, I2C, I2S, SPI, UART,
* STM32F4 là họ vi điều khiển với nhân ARM cortex m4 mạnh mẽ đi kèm với chi phí tối thiểu. Thích hợp cho các dự án yêu cầu về cả tốc độ lẫn giá thành tốt. 

Hình . Sơ đồ kết nối

Ảnh có chứa văn bản, thiết bị điện tử, cáp, đầu nối

Mô tả được tạo tự động

Hình . Kết nối thực tế

## Phần mềm

### Training process

Để dễ dàng tiếp cận và xử lý các chức năng về học máy, Python là một ngôn ngữ dễ sử dụng và được hỗ trợ các thư viện mạnh mẽ về tốc độ xử lý và khả năng thích ứng. Vì vậy ngôn ngữ lập trình chính cho đề tài mà nhóm sử dụng là Python.

Cài đặt môi trường training, các Package sử dụng:

* Tensorflow:
  + TensorFlow chính là thư viện mã nguồn mở cho machine learning nổi tiếng nhất thế giới, được phát triển bởi các nhà nghiên cứu từ Google. Việc hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning đã giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều.
* Keras:
  + Keras cũng là một thư viện giúp xây dựng các mạng học sâu, và được phát triển bởi Google.
  + Keras được coi là một thư viện ‘high-level’ với phần ‘low-level’ (còn được gọi là backend) có thể là TensorFlow, CNTK, hoặc Theano. Keras có cú pháp đơn giản hơn TensorFlow rất nhiều.
  + Keras ưu tiên trải nghiệm của người lập trình
* numpy:
  + Numpy là một thư viện lõi phục vụ cho khoa học máy tính của Python, hỗ trợ cho việc tính toán các mảng nhiều chiều, có kích thước lớn với các hàm đã được tối ưu áp dụng lên các mảng nhiều chiều đó. Numpy đặc biệt hữu ích khi thực hiện các hàm liên quan tới Đại Số Tuyến Tính.
* matplotlib.pyplot:
  + Để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu của bạn và Matplotlib là một trong những giải pháp như vậy cho người dùng Python.
  + Nó là một thư viện vẽ đồ thị rất mạnh mẽ hữu ích cho những người làm việc với Python và NumPy. Module được sử dụng nhiều nhất của Matplotib là Pyplot cung cấp giao diện như MATLAB nhưng thay vào đó, nó sử dụng Python và nó là nguồn mở.
* pandas:
  + Pandas là một thư viện Python cung cấp các cấu trúc dữ liệu nhanh, mạnh mẽ, linh hoạt và mang hàm ý.
  + Ở đây chúng ta sẽ sử dụng khả năng đọc dữ liệu liệu dạng bảng (file .csv) của thư viện.
* Các lớp sẽ sử dụng trong mạng thần kinh:
  + Lớp Hồi quy tuyến tính (RNN-LSTM): thích hợp với dữ liệu dạng chuỗi thời gian
  + Lớp Tích chập (CNN): dùng để chiết xuất đặc trưng từ dữ liệu.
  + Lớp MaxPooling, AvaragePooling: giảm chiều dài của dữ liệu mà không ảnh hưởng quá nhiều tới độ chính xác.
  + Lớp Fully connected: các lớp phi tuyến là không thể thiếu trong mọi mô hình mạng học sâu từ đơn giản tới phức tạp
  + Lớp Softmax Activation: Lớp cuối cùng của mọi mô hình phân loại dữ liệu nhiều lớp.
  + Không thể thiếu hàm loss categorical\_crossentropy: Thường được dùng trong các bài toán phân loại nhiều lớp
  + Thuật toán tối ưu hàm mất mát: Adam

### Deploy process

Đối với các họ vi điều khiển dòng STM32 thì STM32CubeIDE là một môi trường phát triển tích hợp được phát triển bởi chính hãng STMicroelectronics.

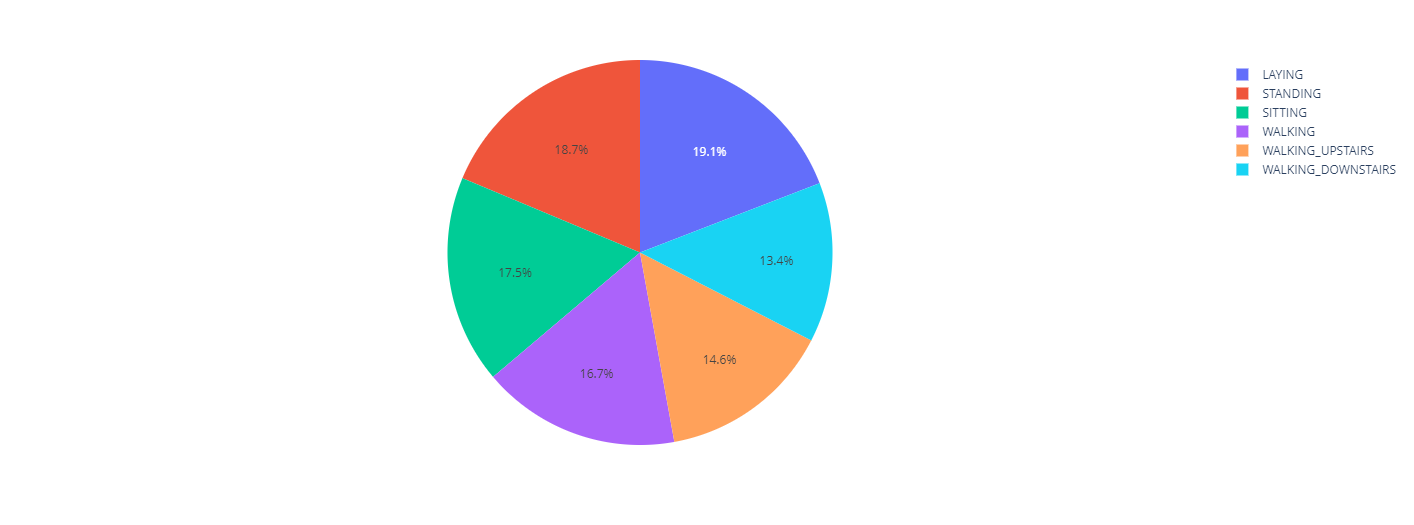
STM32CubeIDE là sự kết hợp giữa một môi trường phát triển và công cụ generate code tự động mạnh mẽ dựa trên cấu hình CubeMX. Khiến nó là thứ vũ khí mạnh mẽ nhất khi so sánh với các dòng vi điều khiển cao cấp trên thị trường kể cả Arduino.

Ngoài ra, còn được tích hợp nhiều công cụ hỗ trợ debug nâng cao giúp nhà phát triển dễ dàng hơn trong quá trình gỡ lỗi và các gói mã nguồn mở rộng cực kì đa dạng và hữu ích. Như gói Hệ điều hành thời gian thực FreeRTOS và gói STM32.AI hỗ trợ hiện thực mạng neural đươc training trước đó.

# THỰC NGHIỆM VÀ ĐÁNH GIÁ

## Trực quan hóa dữ liệu

* Phân bổ dữ liệu dataset

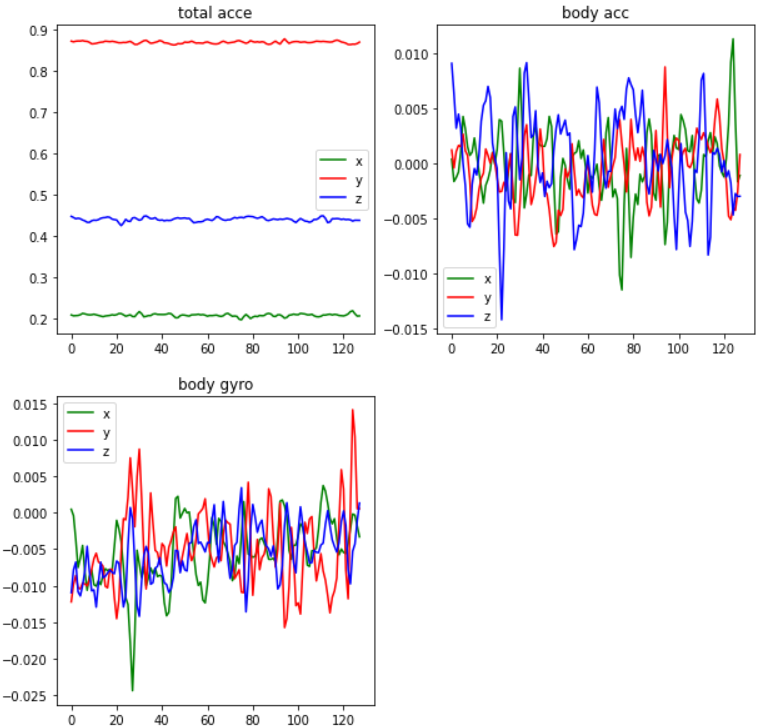


Hình . Phân bổ lớp dữ liệu

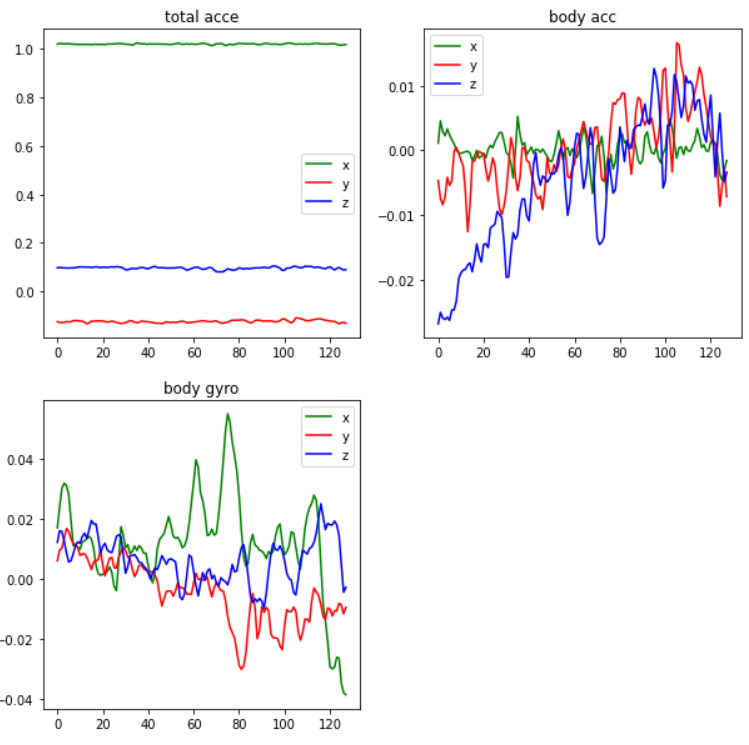
|  |  |  |
| --- | --- | --- |
| **LABEL** | **Số lượng** | **Tỷ trọng** |
| Laying | 1407 | 19.1% |
| Standing | 1374 | 18.7% |
| Sitting | 1286 | 17.5% |
| Walking | 1226 | 16.7% |
| Walking\_Upstairs | 1073 | 14.6% |
| Walking\_Downstairs | 986 | 13.4% |

Bảng 4.1 Phân bổ lớp dữ liệu

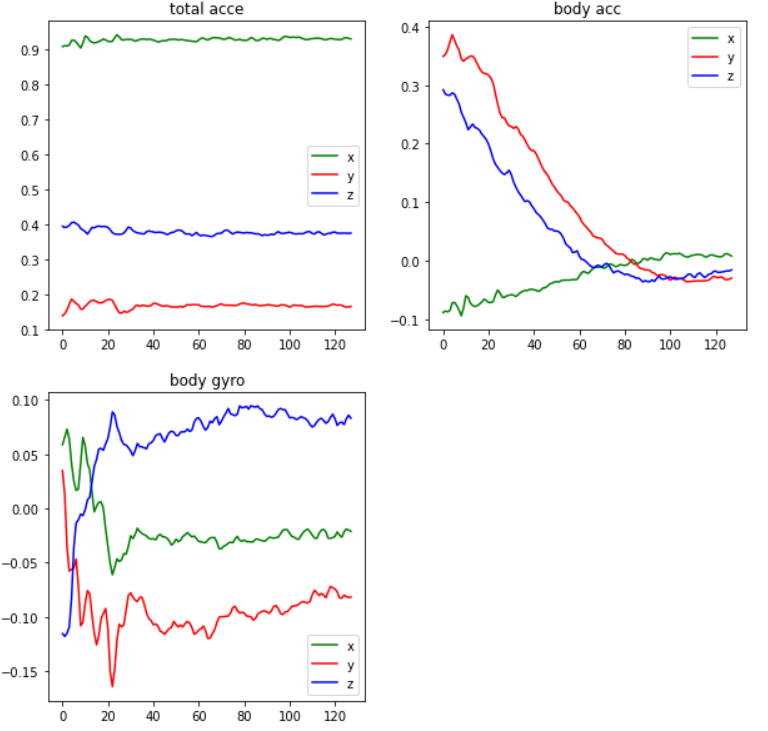
* Giá trị cảm biến khi hành động là “LAYING”:



* Giá trị cảm biến khi hành động “STANDING”:



* Giá trị cảm biến khi hành động là “SITTING”:

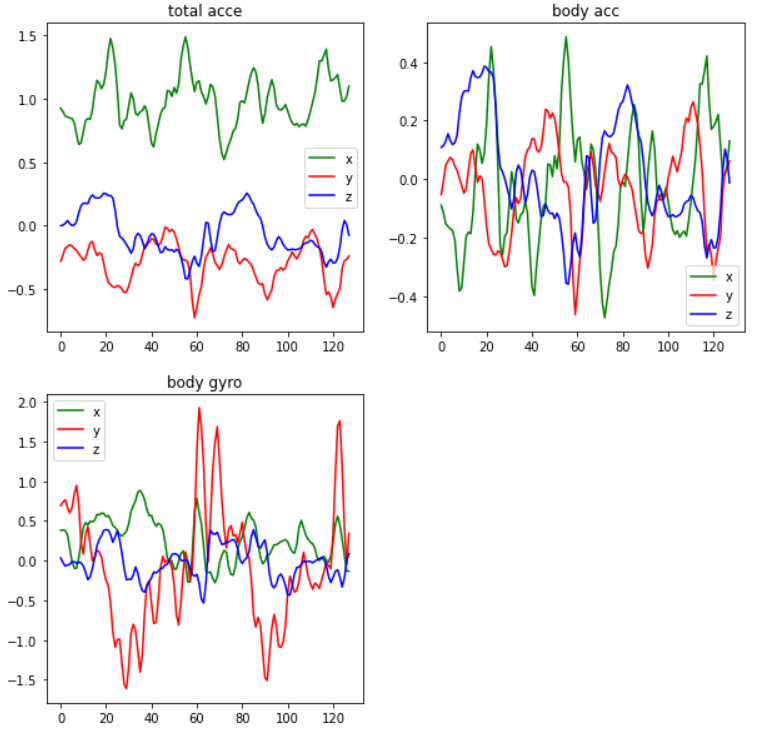


* Giá trị cảm biến khi hành động là “WALKING”:

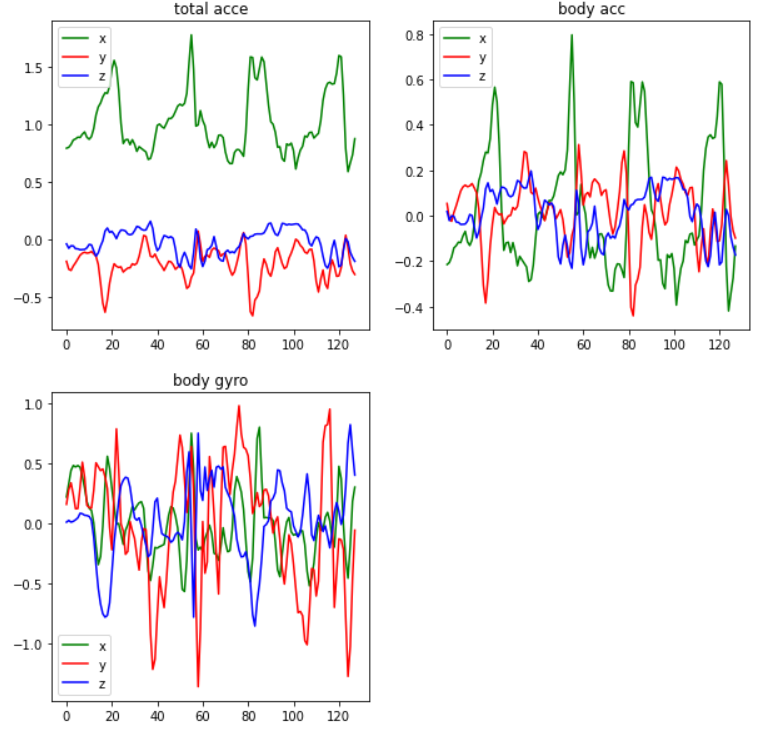
Ảnh có chứa văn bản, công cụ viết, văn phòng phẩm, bút chì

Mô tả được tạo tự động

* Giá trị cảm biến khi hành động là “WALKING\_UPSTAIRS”:

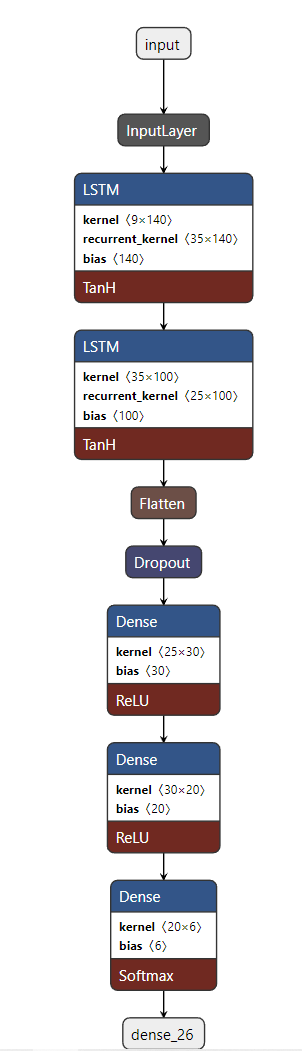


* Giá trị cảm biến khi hành động “WALKING\_DOWNSTAIRS”:



## Quá trình train model:

* Mô hình thử nghiệm 1: gồm 2 lớp LSTM xếp chồng để học dạng dữ liệu time-seris và các lớp phi tuyến.[3]



Layer (type) Output Shape Param #

=================================================================

lstm\_27 (LSTM) (None, 128, 35) 6300

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

lstm\_28 (LSTM) (None, 25) 6100

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten\_8 (Flatten) (None, 25) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout\_8 (Dropout) (None, 25) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_24 (Dense) (None, 30) 780

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_25 (Dense) (None, 20) 620

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_26 (Dense) (None, 6) 126

=================================================================

Total params: 13,926

Trainable params: 13,926

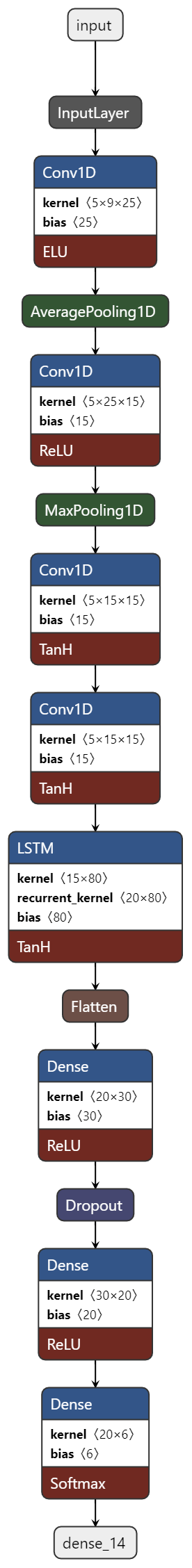
Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

93/93 [==============================] - 3s 24ms/step - loss: 0.3793 - accuracy: 0.9138

[0.37927448749542236, 0.913810670375824]

Hình . Mô hình thử nghiệm 1

* Độ chính xác: 91,3%
* Số param: 13926
* Loss: 0.38
* Mô hình 2: Bao gồm lớp LSTM, avarage\_pooling và CNN để trích xuất đặc trưng, các lớp phi tuyến.[4][5]

Hình . Mô hình 2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv1d\_8 (Conv1D) (None, 128, 25) 1150

average\_pooling1d\_4 (Averag (None, 64, 25) 0

ePooling1D)

conv1d\_9 (Conv1D) (None, 64, 15) 1890

max\_pooling1d\_4 (MaxPooling (None, 32, 15) 0

1D)

conv1d\_10 (Conv1D) (None, 32, 15) 1140

conv1d\_11 (Conv1D) (None, 32, 15) 1140

lstm\_5 (LSTM) (None, 20) 2880

flatten\_4 (Flatten) (None, 20) 0

dense\_12 (Dense) (None, 30) 630

dropout\_4 (Dropout) (None, 30) 0

dense\_13 (Dense) (None, 20) 620

dense\_14 (Dense) (None, 6) 126

=================================================================

Total params: 9,576

Trainable params: 9,576

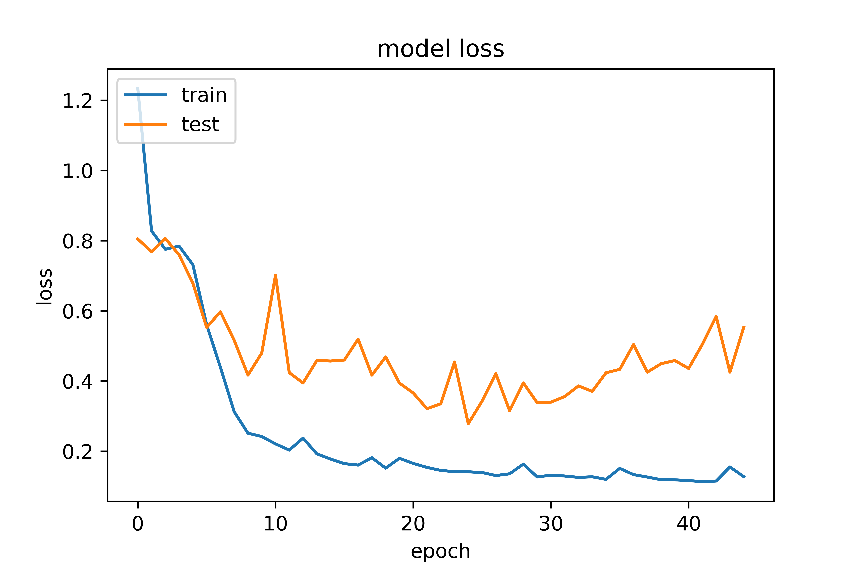
Non-trainable params: 0

* Độ chính xác: 92%
* Số param: 9,576
* Loss: 0.34
* Với bộ dữ liệu gồm 7352 chuỗi hành động, mỗi chuỗi có độ dài 128 data sensor.
* Nhóm tiến hành train 2 mô hình trên với hàm loss function categorical\_crossentropy.
* Tốc độ train khá nhanh bởi vì mô hình được tối ưu có bộ nhớ nhỏ nhất có thể để phù hợp với MCU.
* Mỗi epoch chỉ mất hơn 8s.
* Cuối cùng nhóm chọn sử dụng model 2. Bởi vì độ chính xác cao hơn cũng như đảm bảo tối ưu bộ nhớ sử dụng.
* Nhận xét: Lớp neural tích chập CNN cho thấy sự vượt trội của mình trong các mô hình học sâu.
* Kết quả training cho độ chính xác cao ngay nhưng epoch đầu

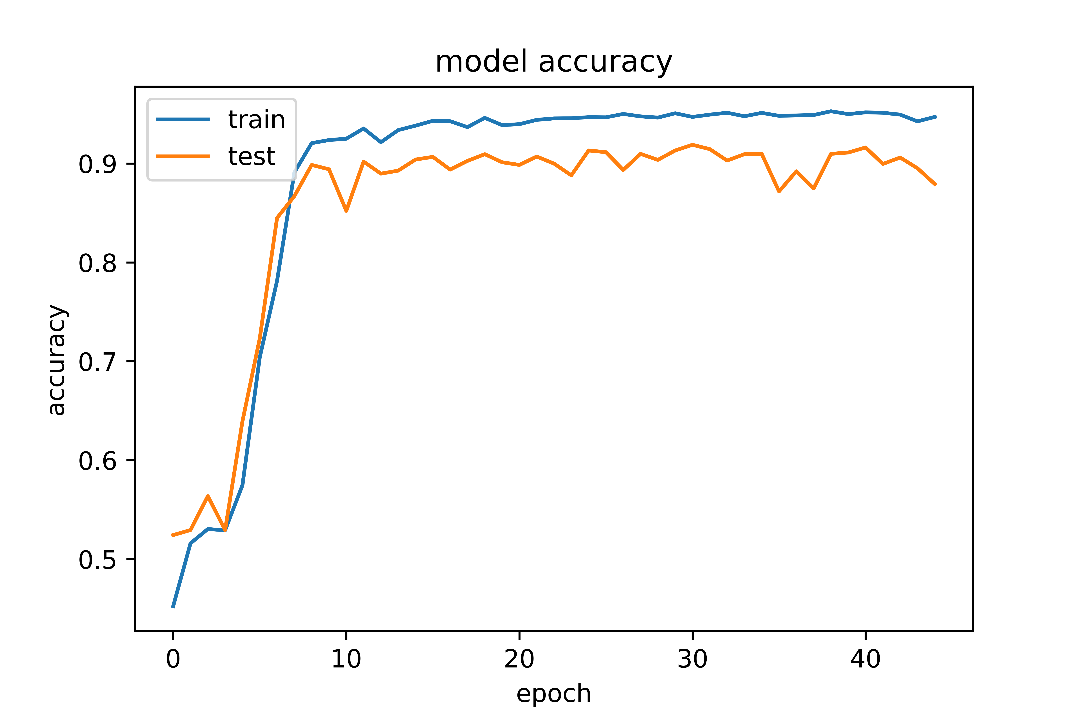
Ảnh có chứa bàn

Mô tả được tạo tự động

Hình . Quá trình training



Hình . Kết quả hàm loss



Hình . Độ chính xác qua các epoch

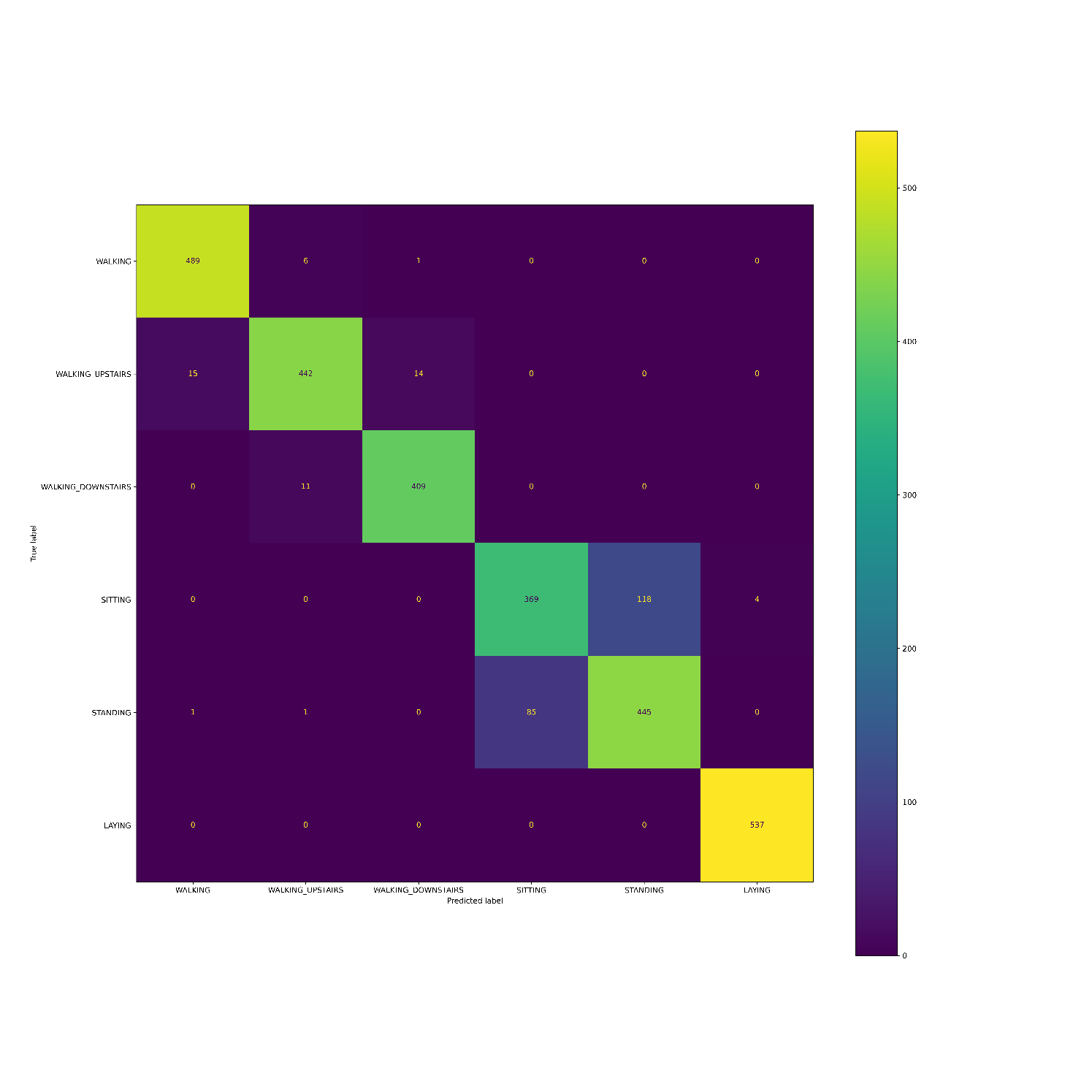
* Đánh giá model:

Ảnh có chứa bàn

Mô tả được tạo tự động

Hình . Đánh giá độ chính xác

* Confusion matrix:



Hình . Confusion matrix của model

* Nhận xét:
  + Kết quả dự đoán có tỷ lệ nhầm lẫn sai đặc biệt cao giữa 2 lớp “STANDING” và “SITTING”.
  + Các lớp còn lại, đặc biệt là lớp “LAYING” cho kết quả dự đoán khá tốt gần như tuyệt đối.

## Quá trình triển khai lên MCU

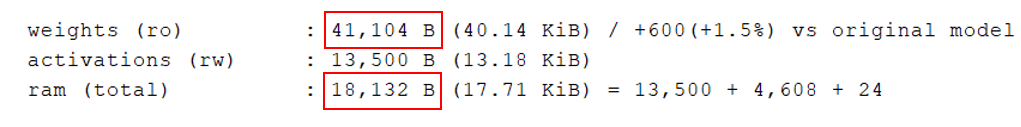
### Đánh giá mô hình bằng công cụ

* Sử dụng Package hỗ trợ mạng nơ ron STM32CubeAI
* Thông số CPU: Tần số sử dụng 100MHz, Ram 128KiB, Rom 512KiB.

Ảnh có chứa bàn

Mô tả được tạo tự động

* Độ phức tạp của mô hình: 584101 MACC
* Thời gian chiếm dụng CPU dự đoán bởi công cụ: 100ms (10Hz)



* Rom sử dụng 40.1 KiB (8%)
* Ram sử dụng 17.8 KiB (13%)
* Mô hình đạt yêu cầu về tốc độ cũng như tiết kiệm bộ nhớ sử dụng



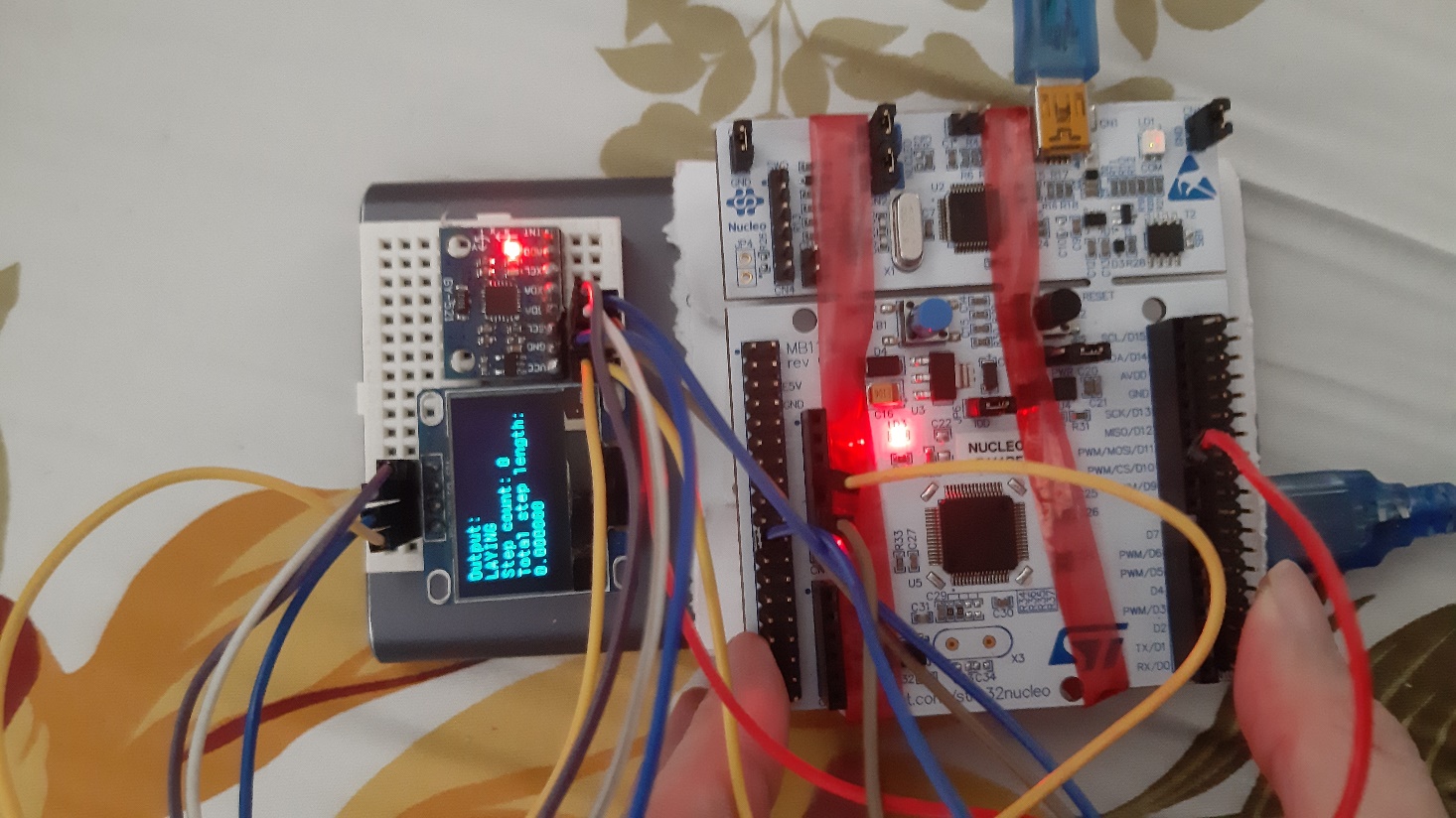
### Đánh giá mô hình thực tế

* Thời gian chạy đúng như sử dụng công cụ dự đoán trước đó, mất khoảng 100ms (10Hz) để tính toán.

|  |  |  |  |
| --- | --- | --- | --- |
| Đánh giá | Dùng công cụ | Chạy thực tế | Chênh lệch |
| Thời gian | 100.1 ms | 102 ms | 1 ms |

## Thử nghiệm & đánh giá toàn bộ hệ thống

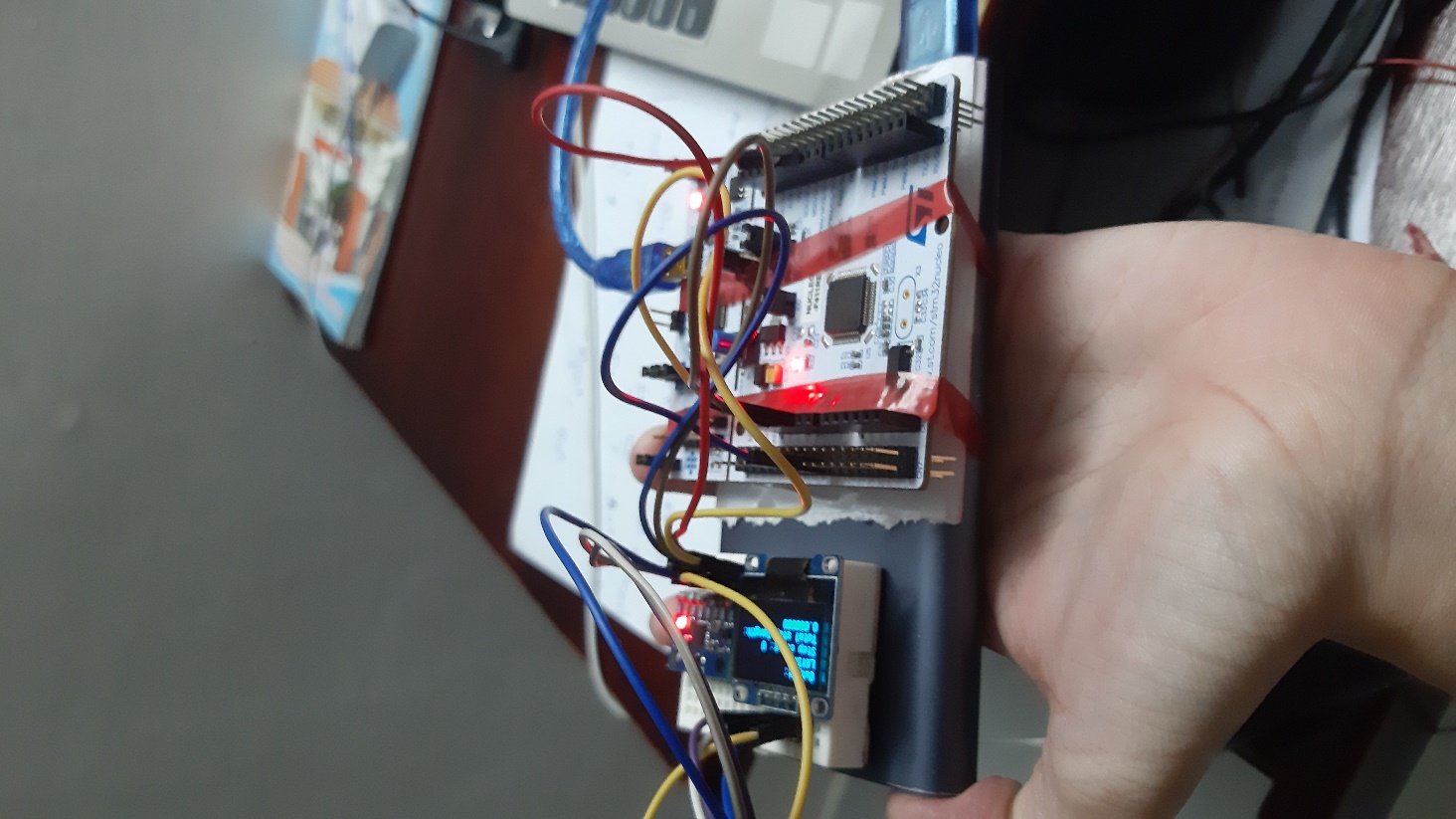
### Một số hình ảnh hệ thống





Ảnh có chứa văn bản, thiết bị điện tử, mạch

Mô tả được tạo tự động



### Video demo

Link: [onedrive](https://uithcm-my.sharepoint.com/:v:/g/personal/19520165_ms_uit_edu_vn/EaQ0LGUOUVVErIq5mKalpxoBIoGcpPErO8nw7xeXYfVLJw?e=dCCHPF) (Tải xuống xem để có chất lượng tốt nhất).

### Đánh giá

* Model đạt được yêu cầu đã đề ra trước đó (1)
* Dù model lúc training cho kết quả khá tốt với độ chính xác hơn 90% và loss dưới 0.5 nhưng kết quả thực nghiệm khi triển khai lên vi điều khiển với data thời gian thực từ sensor lại không tốt như mong đợi. Có thể do môi trường lấy dataset khác so với môi trường triển khai.
* Theo đó, Mô hình có thể dễ dàng phân loại giữa 3 nhóm (LAYING), (WALKING, WALKING\_UP, WALKING\_DOWN), (SITTING, STANDING). Nhưng các lớp cùng nhóm lại dễ bị phân loại nhầm lẫn với nhau.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết quả thu được thông qua quá trình thực hiện đề tài

* Hiểu được về Machine Learning và deep learning cơ bản.
* Nắm được các phương pháp, đặc tính của các mạng neural, áp dụng vào đề tài thực tế.
* Biết cách hiện thực các giải thuật từ các mô hình, các phép toán học.
* Biết cách lập trình nhúng trên HĐH RTOS.

**Ưu điểm:**

* Ngôn ngữ Python hỗ trợ đa dạng, linh hoạt
* Các thư viện lập trình cấp cao thân thiện với mọi lập trình viên như Keras.
* Độ chính xác khi training model trên 90%
* Model và các giải thuật được nhóm tự đề xuất. Model nhỏ gọn tiết kiệm bộ nhớ, phù hợp để triển khai lên hệ thống IoT.

**Nhược điểm:**

* Độ chính xác khi triển khai thực tế lên MCU không cao
* Hệ thống còn đơn giản, ít chức năng
* Các thuật toán đếm bước chân và tính độ dài dù khá hợp lý theo lý thuyết nhưng vẫn chưa chính xác khi triển khai thực tế. Do vậy, cần tham khảo thêm thuật toán tốt hơn cho hệ thống thực tế.

## Hướng phát triển của đề tài

Trong tương lai hệ thống cần được cải tiến, cải thiện để phù hợp với ứng dụng trong thực tế hơn, như là:

* Cải thiện độ chính xác của hệ thống trong sử dụng thực tế
* Nâng cấp hệ thống thành một thiết bị đeo thông minh thực thụ với nhiều chức năng hơn.
* Liên kết với phần mềm điện thoại tự phát triển để cung cấp trải nghiệm đồng bộ cho người dùng.
* Có các kết nối không dây phổ biến như BLE, Wifi,…
* Nhiều hơn nữa các thông tin giám sát sức khỏe như: nồng độ SpO2, nhịp tim, giám sát chất lượng ngủ, phản hồi xúc giác, …

TÀI LIỆU THAM KHẢO

Theo chuẩn IEEE

[1]Google APIs for Android: ActivityRecognitionAPI, <https://developers.google.com/location-context/activity-recognition>

(retrieved: Dec. 14, 2016)

[2]H. T. Duong and Y. S. Suh, "Walking distance estimation of a walker user using a wrist-mounted IMU," 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), 2017, pp. 1061-1064, doi: 10.23919/SICE.2017.8105462.

[3]J. Wang, C. Lin, Y. C. Yang and Y. Ho, "Walking Pattern Classification and Walking Distance Estimation Algorithms Using Gait Phase Information," in IEEE Transactions on Biomedical Engineering, vol. 59, no. 10, pp. 2884-2892, Oct. 2012, doi: 10.1109/TBME.2012.2212245.

[4]Song-Mi Lee, Sang Min Yoon and Heeryon Cho, "Human activity recognition from accelerometer data using Convolutional Neural Network," 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), 2017, pp. 131-134, doi: 10.1109/BIGCOMP.2017.7881728.

[5]W. Jiang and Z. Yin, “Human activity recognition using wearable sensors by deep convolutional neural networks,” in Proc. of the 23rd ACM Int’l Conf. on Multimedia (MM ’15). ACM, New York, NY, USA, pp. 1307-1310, 2015.