|  |
| --- |
| **-**  ****BỘ GIÁO DỤC VÀ ĐÀO TẠO****  ****TRƯỜNG ĐẠI HỌC HOA SEN****  ****KHOA KHOA HỌC KỸ THUẬT**** |

KHÓA LUẬN TỐT NGHIỆP

Tên đề tài: Human Activity Recognition

Giảng viên hướng dẫn : Nguyễn Trùng Lập

Lớp : TIN499DV01

Thời gian thực hiện : Từ 16/7/2018 đến 26/10/2018

Nhóm sinh viên thực hiện : Lưu Tấn Nguyên - 2152192

Số nhóm : 1

**THÁNG 10 / NĂM 2018**

# Trích yếu

Khóa luận tốt nghiệp là một trong những thử thách cuối cùng của tôi tại trường đại học do đó tôi muốn thử thách mình với một hướng khác với thuần viết ứng dụng mà tôi đã làm rất nhiều trong những năm đại học. Đề tài của tôi là Human Activity Recognition – Nhận diện hoạt động của con người thông qua cảm biến của thiết bị thông minh mà trong đề tài tôi sử dụng là hai cảm biến Accelerometer và Gyroscope. Thông qua bộ dữ liệu đã được thu thập và chuẩn hóa từ những bài nghiên cứu trước đó, tôi tiếp tục sử dụng các kỹ thuật của Machine Learning nhằm để máy “học” được, từ đó sử dụng hai cảm biến của Android để xác định và đo đạc thực tế. Kết quả đạt được tương đối khả quan, tuy nhiên vẫn không thật sự chính xác, vẫn còn nhầm lẫn qua lại giữa nhiều hoạt động. Từ những nhận diện đó, tôi tiếp tục mở rộng bằng cách tính toán lượng năng lượng tiêu hao trong mỗi hoạt động để đưa ra những cảnh báo cơ bản về sức khỏe.

# Mục lục

[Trích yếu II](#_Toc528310125)

[Mục lục III](#_Toc528310126)

[Danh mục hình ảnh V](#_Toc528310127)

[Lời cảm ơn VI](#_Toc528310128)

[Nhận xét của giảng viên hướng dẫn VII](#_Toc528310129)

[Nhập đề 1](#_Toc528310130)

[Nội dung 2](#_Toc528310131)

[1. Tổng quan 2](#_Toc528310132)

[1.1. Đặt vấn đề 2](#_Toc528310133)

[1.2. Tiến trình xử lý 3](#_Toc528310134)

[1.3. Giới thiệu mẫu 4](#_Toc528310135)

[1.3.1. Phương thức lấy mẫu 4](#_Toc528310136)

[1.3.2. Xử lý tín hiệu với các cơ sở lý thuyết 5](#_Toc528310137)

[1.3.2.1. Median Filter 6](#_Toc528310138)

[1.3.2.2. 3rd – order Low-pass Butterworth Filter 8](#_Toc528310139)

[1.4. Hướng tiếp cận về huấn luyện 11](#_Toc528310140)

[1.4.1. Tiếp cận từ bộ dữ liệu đã xử lý 11](#_Toc528310141)

[1.4.1.1. Thuật toán Simple Neural Network với Keras 11](#_Toc528310142)

[1.4.1.2. Kết quả huấn luyện với Simple Neural Network 14](#_Toc528310143)

[1.4.2. Tiếp cận từ bộ dữ liệu thô 16](#_Toc528310144)

[1.4.2.1. Thuật toán Recurrent Neural Network với Keras 16](#_Toc528310145)

[1.4.2.2. Kết quả huấn dùng RNN 16](#_Toc528310146)

[1.4.2.3. Thuật toán Long-Short Term Mermory 18](#_Toc528310147)

[1.4.2.4. Kết quả huấn luyện dùng LSTM 19](#_Toc528310148)

[1.5. Chuyển đổi, nhúng mô hình vào Android và kết quả 20](#_Toc528310149)

[1.6. Những ứng dụng cơ bản về sức khỏe 22](#_Toc528310150)

[1.7. Hướng phát triển trong tương lai 24](#_Toc528310151)

[2. Các kết luận và đề nghị 25](#_Toc528310152)

[3. Tài liệu tham khảo 26](#_Toc528310153)

# Danh mục hình ảnh

Hình 1 Cảm biến Gyroscope và Acceleromter 2

Hình 2 Tiến trình xử lý 3

Hình 3 Cách thu thập dữ liệu thô 4

Hình 4 Trình tự thu thập dữ liệu của nhóm nghiên cứu 5

Hình 5 Nguyên lý bộ lọc Median 6

Hình 6 Áp dụng bộ lọc Median vào xử lý ảnh 7

Hình 7 Áp dụng bộ lọc Median trong xử lý tín hiệu 7

Hình 8 Bộ lọc Low-pass Butterworth 8

Hình 9 Độ thoải tương ứng số bậc 8

Hình 10 Sự tương quan giữa bậc 2 với bán kính cắt bỏ D 9

Hình 11 So sánh Butterworth với các bộ lọc khác 10

Hình 12 So sánh High-pass và Low-pass Butterworth 10

Hình 13 Ví dụ về Overfitting - mô tả sự phân loại dữ liệu dựa vào các đường ranh 13

Hình 14 Quá trình học với Simple Neural Network 14

Hình 15 Kết quả sau khi huấn luyện Simple Neural Network 1 14

Hình 16 Kết quả sau khi huấn luyện Simple Neural Network 2 15

Hình 17 Ý tưởng thuật toán RNN - Feedforward 16

Hình 18 Tiến trình huấn luyện bằng RNN 17

Hình 19 Kết quả huấn luyện bằng RNN 17

Hình 20 Kết quả huấn luyện bằng LSTM 19

Hình 21 Đồ thị Gyroscope và Accelerometer theo thời gian thực 20

Hình 22 Kết quả nhận diện 6 hoạt động cơ bản 21

Hình 23 Công thức tính lượng Calorie bị đốt dựa vào MET 22

Hình 24 Công thức tính BMR theo Harris-Benedict (Wikipedia, 2018) 23

# Lời cảm ơn

Tôi cảm ơn Trường Đại học Hoa Sen đã tạo điều kiện cho tôi được thực hiện khóa luận tốt nghiệp kéo dài ba tháng. Để hoàn thành được bài nghiên cứu này, tôi đặc biệt cảm ơn thầy Nguyễn Trùng Lập đã có thư giới thiệu và dẫn tôi được sang Trường Đại học Quốc Gia Jeju – Hàn Quốc để tham gia trại hè nghiên cứu khoa học về Machine Learning dành cho những sinh viên, nghiên cứu sinh từ Trung Quốc, Ấn Độ, Philipines, Thái Lan, Việt Nam… Cũng tại đây, tôi đặc biệt cảm ơn giáo sư Yung Cheol-Byun công tác tại Đại học Quốc Gia Jeju đã tạo điều kiện mời chuyên gia về Machine Learning của Google để tập huấn sử dụng Google Cloud Platform và những kiến thức liên quan, đồng thời cũng có những chuyến tham quan, giao lưu với bạn bè quốc tế trên đất nước Hàn Quốc xuyên suốt trại hè.

Tuy nhiên, vì lượng kiến thức vẫn còn rất hạn chế nên sự thiếu xót, sai xót hoàn toàn không thể tránh khỏi, do đó tôi rất mong muốn nhận được sự đóng góp để tôi có thể nâng cấp, sử dụng thành bài báo quốc tế, nếu như tôi vẫn tiếp tục có cơ hội mở rộng hướng nghiên cứu này.

# Nhận xét của giảng viên hướng dẫn

Giảng viên hướng dẫn

# Nhập đề

Hiện nay có rất nhiều thiết bị được sử dụng để nhận dạng và thu thập hoạt động của con người, nổi trội hơn cả là đồng hồ thông minh (smartwatch). Tuy nhiên, do bộ dữ liệu thô (raw dataset) không nhiều, cũng như không có thiết bị phù hợp mà nhà sản xuất cho phép mở API để nghiên cứu. Vì thế sau khi tham khảo những bộ dữ liệu đã được thu thập từ điện thoại thông minh và được chuẩn hóa có độ chính xác cao, tôi quyết định sử dụng điện thoại thông minh (smartphone) để phát triển bài nghiên cứu của mình, cụ thể tôi có những mục tiêu như sau:

* Mục tiêu 1: Tham khảo các bài báo nghiên cứu với những bộ dữ liệu thô đã được chuẩn hóa. Nghiên cứu và tái dựng cách chuẩn hóa nhằm tái sử dụng cho Android.
* Mục tiêu 2: Thử nghiệm huấn luyện với nhiều thuật toán khác nhau của Machine Learning và đưa ra kết quả. Bao gồm những kỹ thuật tiền xử lý, xử lý, huấn luyện, thử nghiệm mô hình dựa trên bộ dữ liệu thử nghiệm và bộ dữ liệu thực tế từ hai cảm biến Android.
* Mục tiêu 3: Áp dụng kết quả nhận diện cho những ứng dụng cơ bản về sức khỏe trên Android.

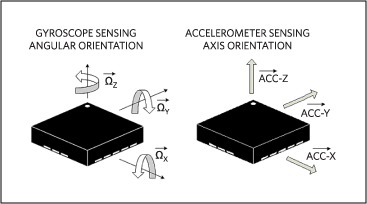
# Nội dung

## Tổng quan

### Đặt vấn đề

Hiện nay có rất nhiều thiết bị hỗ trợ sức khỏe được trang bị trên eo, cổ tay, ngực và đùi để phát hiện, báo cáo tình trạng sức khỏe được tốt nhất. Điển hình là sử dụng đồng hồ thông minh - Smartwatch để đo nhịp tim, nhận diện hoạt động thường ngày của con người, thậm chí tự động nhận diện những hoạt động thể thao như bơi lội, đánh cầu hoặc kể cả chất lượng giấc ngủ (BELLE, 2014) [[1]](#footnote-1).

Tuy nhiên, liệu điện thoại thông minh - Smartphone có những khả năng đó hay không nếu chỉ dựa vào những cảm biến được tích hợp sẵn? Để trả lời câu hỏi đó, một nhóm nghiên cứu đến từ Đại học Genova (Ý) và Bách khoa Catalonia (Tây Ban Nha) vào năm 2012 đã nghiên cứu và thu thập dữ liệu về con người từ hai cảm biến Con quay hồi chuyển (Gyroscope) và Gia tốc kế (Accelerometer) để phát hiện 6 hoạt động cơ bản: Đứng, ngồi, nằm, đi, lên và xuống cầu thang.



Hình 1 Cảm biến Gyroscope và Acceleromter

Kết quả thu thập được của nhóm rất khả quan, độ chính xác 90-96% trên mẫu thử nghiệm, nhưng chưa có được dùng trên mẫu thời gian thực – nghĩa là chưa có áp dụng ngược lại vào Smartphone. Do đó tôi quyết định tái dựng (reproduce) kết quả nghiên cứu này để áp dụng vào Android với những ứng dụng cơ bản đi kèm. Ngoài ra cũng vì đồ án chuyên ngành A tại Đại học Hoa Sen của tôi là Health Care Tracking (Nguyen, 2017) [[2]](#footnote-2), nên tôi muốn nghiên cứu hướng này để nâng cấp mở rộng tính năng cho ứng dụng. Cụ thể, sau khi nhận dạng những hoạt động cơ bản, tôi sẽ dựa vào đó tính toán lượng Calorie bị đốt, để từ đó đưa ra những khuyến nghị sức khỏe.

### Tiến trình xử lý

Hình 2 Tiến trình xử lý

Tiến trình xử lý cơ bản là nghiên cứu bài báo và lấy mẫu dataset, sau đó tái dựng, huấn luyện và phân tích kết quả. Tùy thuật toán khác nhau mà kết quả sẽ khác nhau, vì thế nếu kết quả không được như kỳ vọng thì tôi sẽ trở lại bước thứ 2 thậm chí bước đầu tiên. Chi tiết mẫu tôi sẽ đi sâu hơn vào mục kế tiếp.

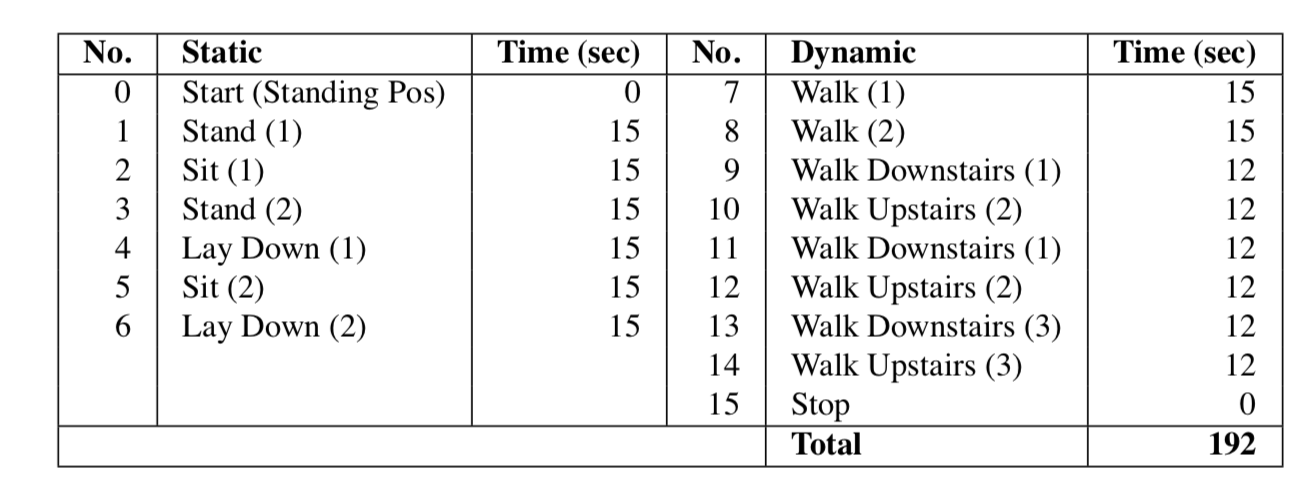
### Giới thiệu mẫu

### Phương thức lấy mẫu

Mẫu dataset đến từ nhóm nghiên cứu (Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, & Jorge L. Reyes-Ortiz, 2012) [[3]](#footnote-3) được thu thập từ 30 tình nguyên viên từ độ tuổi 19 đến 48 thực hiện hoạt động thường ngày – Activities of Daily Living (ADL). Các tình nguyện viên sẽ đeo điện thoại Galaxy S2 ngay vùng thắt lưng [Hình 3] và thực hiện những động tác theo trình tự ở lần 1 và động tác ngẫu hứng ở lần 2, tất cả động tác đều thực hiện trong một khoảng thời gian quy ước như [Hình 4]



Hình 3 Cách thu thập dữ liệu thô



Hình 4 Trình tự thu thập dữ liệu của nhóm nghiên cứu

### Xử lý tín hiệu với các cơ sở lý thuyết

Trước khi trình bày quy trình xử lý tín hiệu, tôi mặc định giữ nguyên những thuật ngữ tiếng Anh sẽ được dùng thường xuyên hơn thay vì dịch ra tiếng Việt:

* Accelerometer: Gia tốc kế
* Gyroscope: Con quay hồi chuyển
* Gravitational: Hấp dẫn
* Body motion: Chuyển động của chính cảm biến đó
* Training: Huấn luyện
* Testing: Thử nghiệm
* Validation: Xác thực

Tín hiệu được thu thập ở tần số 50Hz, khử nhiễu bằng thuật toán bộ lọc Median [1.3.2.1] và bộ lọc 3rd –order low-pass Butterworth [1.3.2.2] được cắt bỏ 20Hz – Nhóm nghiên cứu đã đo đạc và rút ra kết quả hầu hết năng lượng 99% đều ở ngưỡng dưới 15Hz.

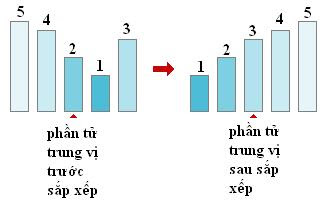
Tín hiệu Accelerometer bao gồm Gravitational và Body motion được tách bằng cách sử dụng tiếp bộ lọc Low-pass Butterworth độc lập với trên thành tín hiệu Gravity và Body Acceleration. Kết quả từ thí nghiệm của nhóm nhận xét và giả định hằng số 0.3 Hz là tần số góc tối ưu của tín hiệu Gravity. Một vài thông tin khác về mẫu được ghi lại như:

* Bước chân mặc định từ 90 đến 130 bước mỗi phút, trung bình 1.5 bước mỗi giây.
* Người già hoặc bị khuyết tật được mặc định tối thiểu phải đạt 50% trung bình.

Ngoài ra còn có những chỉ số tín hiệu khác với tổng cộng 561 chỉ số được tính toán sau đó sau khi lấy được dữ liệu thô, tuy nhiên chúng chỉ mang ý nghĩa để thuật toán Machine Learning tốt hơn chứ không có mang nhiều ý nghĩa trong việc trích xuất trực tiếp theo thời gian thực – do điện thoại phải tính toán rất nhiều trong 50Hz nên tôi quyết định chỉ sử dụng hai bộ lọc trên 3 trục Acceleromter và 3 trục Gyroscope.

### Median Filter

Bộ lọc Trung Vị - Median Filter được sử dụng để lọc những loại nhiễu bằng cách sắp xếp các phần tử theo hướng lớn đến nhỏ hoặc nhỏ đến lớn rồi lấy phần tử chính giữa nếu tổng phần tử là số lẻ hoặc lấy trung bình của hai phần tử giữa nếu tổng phần là số chẵn.



Hình 5 Nguyên lý bộ lọc Median

Ví dụ 1: Với tổng phần tử là số lẻ

Phần tử của tập X = {5, 9, 38, 2, 8, 6, 24}

Phần tử X sắp xếp tăng dần: X = {2, 5, 6, 8, 9, 24, 38}; Tổng phần tử X = 7

* Median của X = 8

Ví dụ 2: Với tổng phần tử là số chẵn

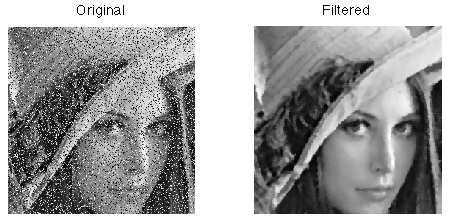
Phần tử của tập X = {5, 9, 38, 2, 8, 6, 24, 21}

Phần tử X sắp xếp tăng dần: X = {2, 5, 6, 8, 9, 21, 24, 38}; Tổng phần tử X = 8

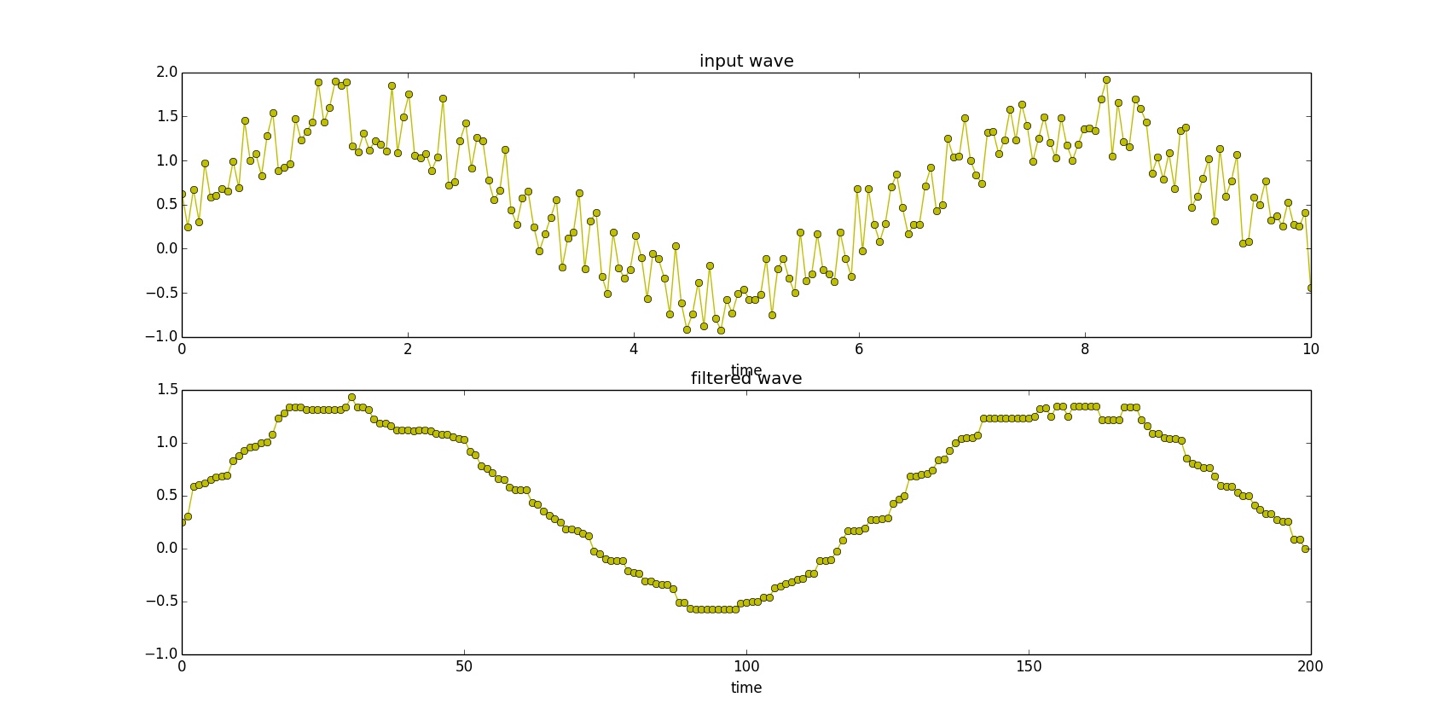
Chọn hai phần tử giữa: 8 và 9

* Median của X = (8+9)/2 = 8.5

Kỹ thuật lọc này sử dụng rất nhiều trong xử lý ảnh bị nhiễu salt hoặc pepper [Hình 6], tuy nhiên trong xử lý tín hiệu thì cũng mang hiệu quả không kém [Hình 7]



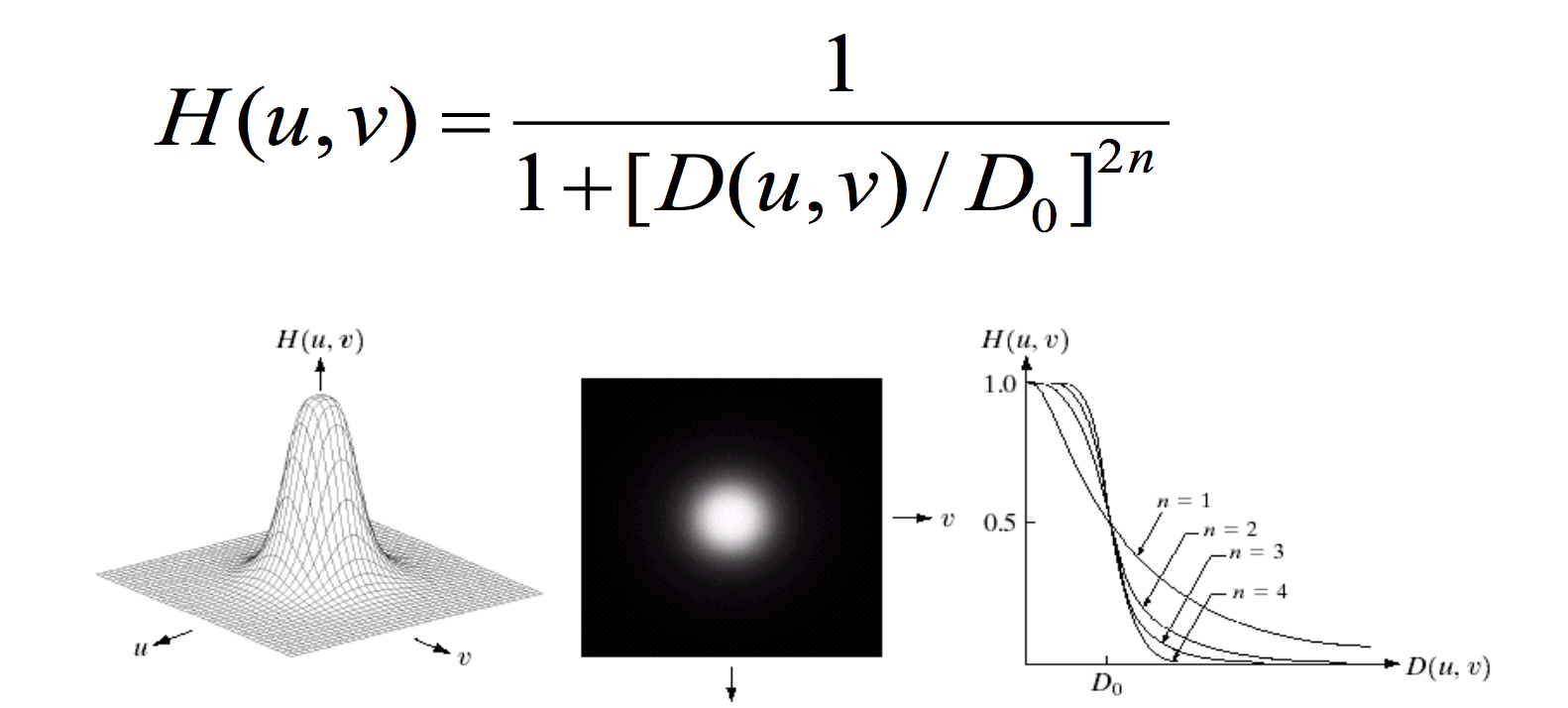
Hình 6 Áp dụng bộ lọc Median vào xử lý ảnh



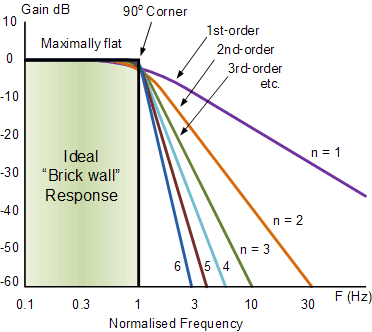
Hình 7 Áp dụng bộ lọc Median trong xử lý tín hiệu

### 3rd – order Low-pass Butterworth Filter

Low-pass Butterworth Filter còn gọi là bộ lọc thông thấp là một hàm chuyển đổi bậc n và cắt bỏ tần số (Cutoff frequency) tại độ rộng khoảng cách D (Nayak) [[4]](#footnote-4) để tạo nên độ thoải hoặc dốc sau của đồ thị.

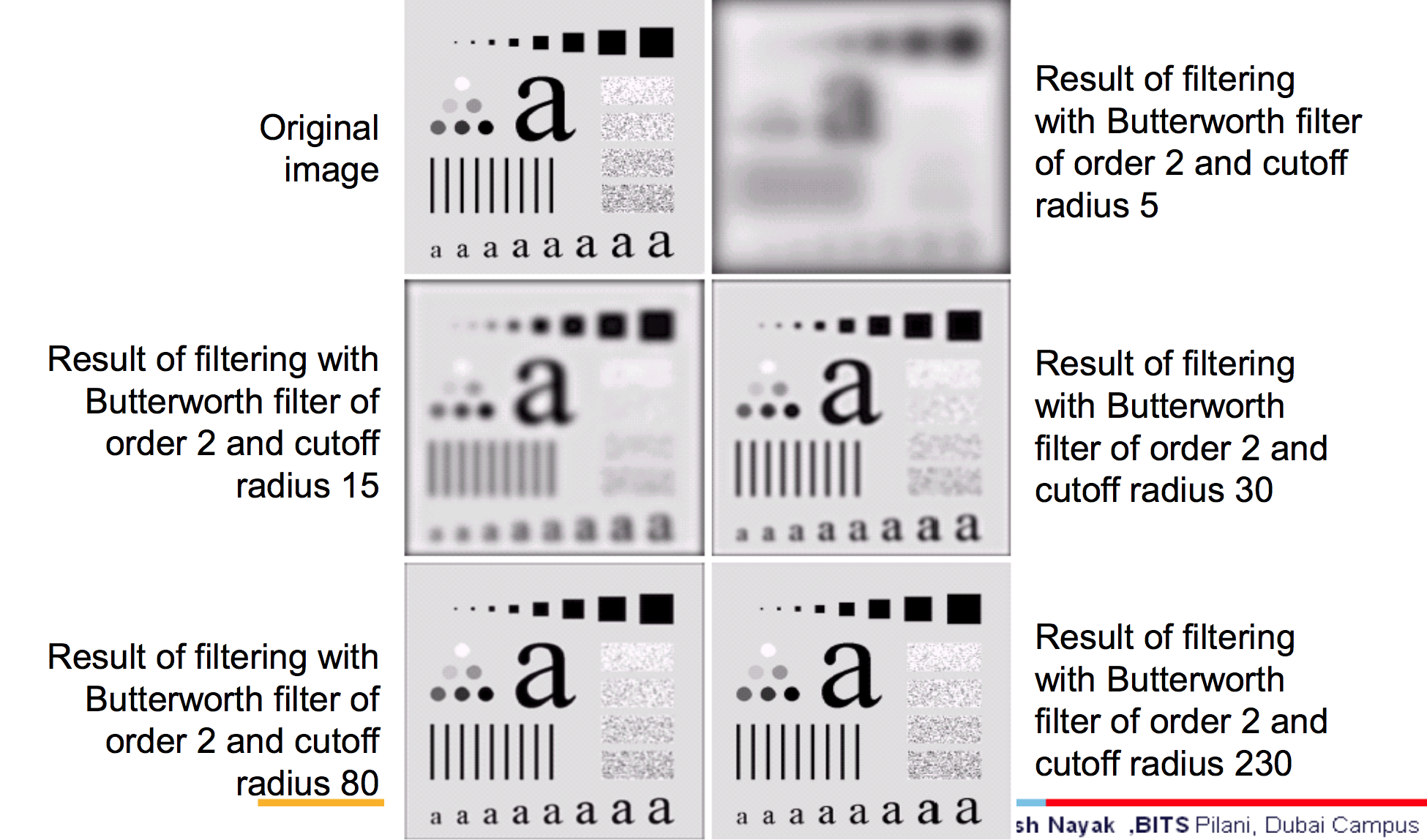


Hình 8 Bộ lọc Low-pass Butterworth



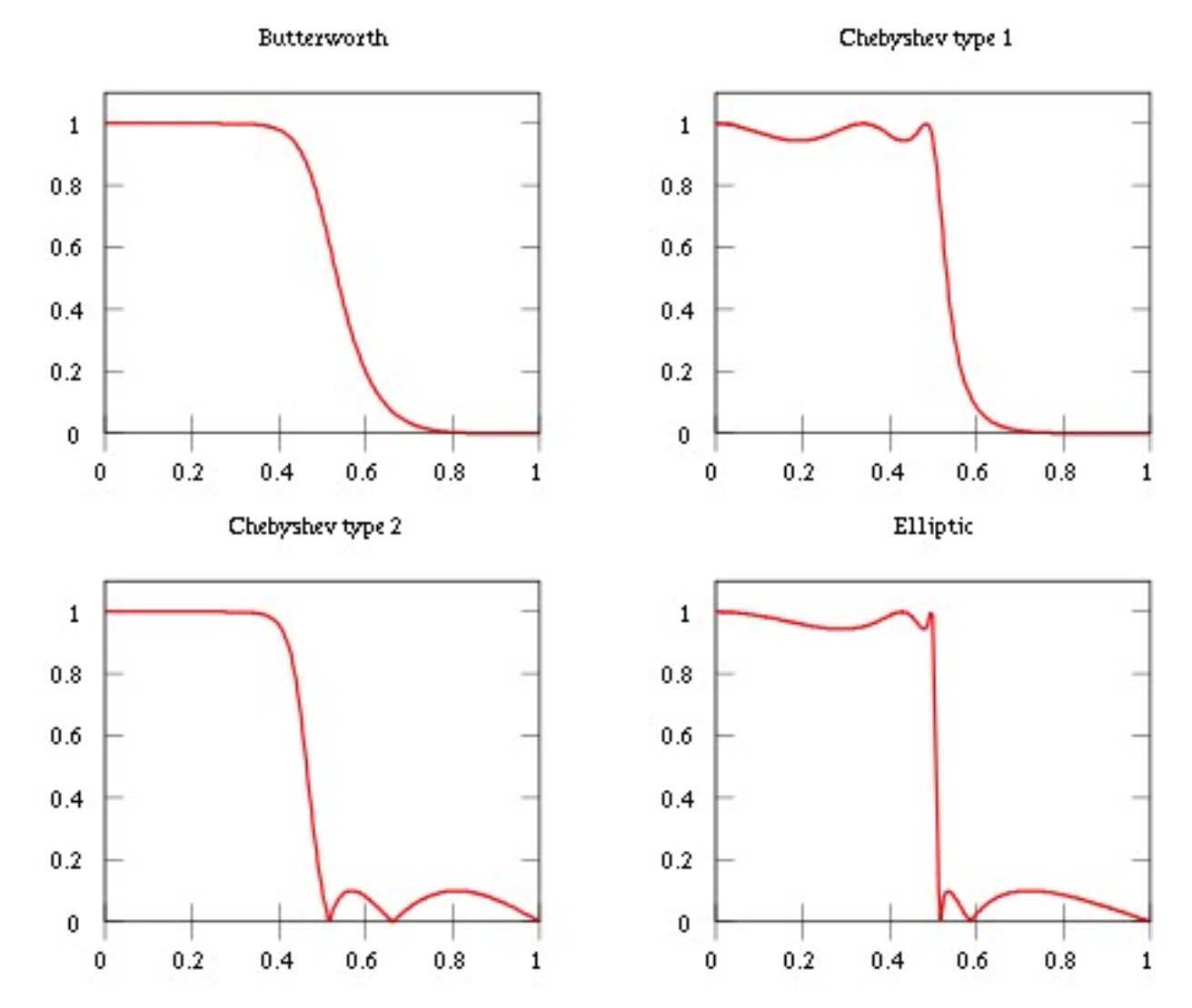
Hình 9 Độ thoải tương ứng số bậc

Bậc càng cao, đồ thị càng dốc và ngược lại, trong bài nghiên cứu nhóm đã sử dụng bậc 3 (3rd – order) để được tín hiệu tốt nhất. Ngoài ra bán kính – hoặc độ rộng khoảng cách D cũng quyết định được chất lượng của ảnh hoặc tín hiệu.



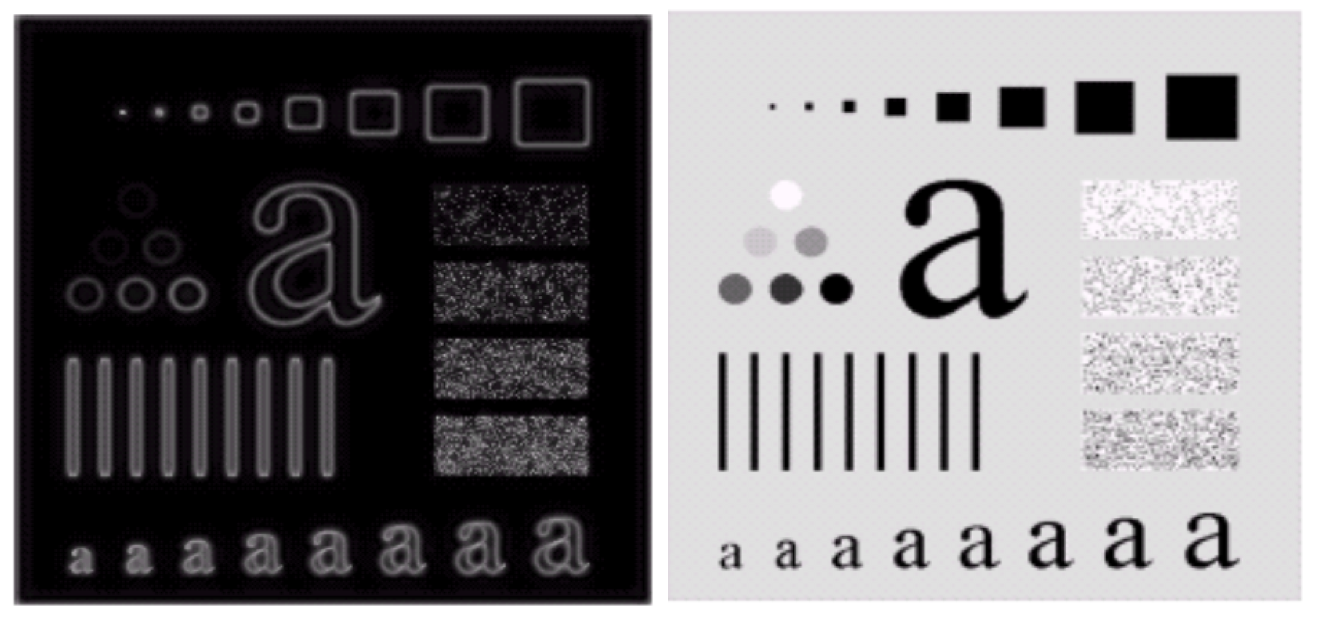
Hình 10 Sự tương quan giữa bậc 2 với bán kính cắt bỏ D

Điểm mạnh của bộ lọc này là kết quả tạo ra sau đó sẽ không nhấp nhô như những bộ lọc khác



Hình 11 So sánh Butterworth với các bộ lọc khác

Ngoài ra, việc lựa chọn Low-pass (thông thấp) nghĩa là cho qua những tần số thấp thay vì dùng High-pass (thông cao), do đó hàm Low-pass là một hàm bù so với hàm High-pass. Nguyên nhân của việc lựa chọn này là để xóa bỏ độ giật ngoài ý muốn trong quá trình thực hiện hoạt động (Đi, đứng, ngồi,… )



Hình 12 So sánh High-pass và Low-pass Butterworth

### Hướng tiếp cận về huấn luyện

Tôi tiếp cận bộ dữ liệu đã xử lý được đăng tải trên Kaggle (UCI Machine Learning, 2017) [[5]](#footnote-5), tuy nhiên sau khi huấn luyện dữ liệu tôi không thể ánh xạ từ điện thoại được do có đến 561 chỉ số hay còn gọi là vector/node/feature ở tầng đầu tiên. Lợi điểm của bộ dữ liệu này là training rất nhanh, cho ra những kết quả tốt nhưng hoàn toàn không khả thi về mặt ứng dụng thực tế trên smartphone do hiệu năng xử lý của Android không thực sự ổn, cũng như không thể ánh xạ hoàn toàn các feature từ bộ dữ liệu thô trích xuất từ Android. Sau khi chuyển qua bộ dữ liệu gốc với thời gian huấn luyện tuy lâu nhưng có khả quan hơn khi chỉ giữ 9 vector gốc bao gồm Accelerometer được phân tích thành 3 trục Body Acceleration và 3 trục Gravity, còn lại là 3 trục Gyroscope. Toàn bộ kỹ thuật trong Machine Learning tôi sử dụng 2 công cụ Keras và Tensorflow để hỗ trợ hiện thực hóa. Về phần Android tôi sử dụng thuần Java Android và thư viện Tensorflow nhúng trong Android, ngoài ra còn một số đồ thị của bên thứ ba để mô tả sinh động theo thời gian thực.

### Tiếp cận từ bộ dữ liệu đã xử lý

Tôi tạm chia bộ dữ liệu này làm 3 bộ, trong đó:

* Bộ testing: 30% ~ 2000 records
* Bộ training: 70% ~ 8000 records

Bộ validation: 10% phần tử cuối từ bộ training

### Thuật toán Simple Neural Network với Keras

Thuật toán tôi sử dụng trong trường hợp này là Simple Neural Network với Keras trên Tensorflow [[6]](#footnote-6), thực hiện gần 700 epochs (mỗi epoch là 1 vòng huấn luyện), đạt độ chính xác 93.69%. Simple Neural Network trong hướng tiếp cận này là đơn giản với 3 lớp chính lần lượt là

|  |
| --- |
| model = Sequential()  model.add(Dense(64, activation='relu', input\_dim=561))  model.add(Dropout(0.3))  model.add(Dense(64, activation='relu'))  model.add(Dropout(0.3))  model.add(Dense(6, activation='softmax')) |

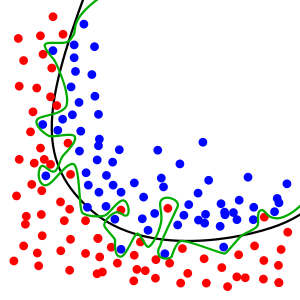
Lớp đầu tiên là chiều input với kích thước 561 vector, xuất ra mảng output có kích thước 64. Sau đó thực hiện Dropout với tỷ lệ 30% (Thuật ngữ dropout nghĩa là sẽ cho các unit thành 0 ngẫu nhiên trong 30% tổng số records sau mỗi vòng lặp epochs)

Lớp thứ hai sử dụng thuật toán ReLU (Rectified Linear Unit) là thuật toán có độ hội tụ nhanh khi sử dụng tối ưu Stochastic Gradient Descent (SGD) được tích hợp sẵn trong Keras giúp tốc độ học nhanh hơn. Sau khi hoàn thành thuật toán sẽ tiếp tục cho ra mảng có kích thước 64 và thực hiện Dropout 30%

Lớp thứ ba sử dụng thuật toán hồi quy Softmax (Tiệp, n.d.) [[7]](#footnote-7) nhằm phân loại các hoạt động với output là một mảng 6 chiều gồm các hoạt động đi, đứng, ngồi, nằm, lên và xuống cầu thang.

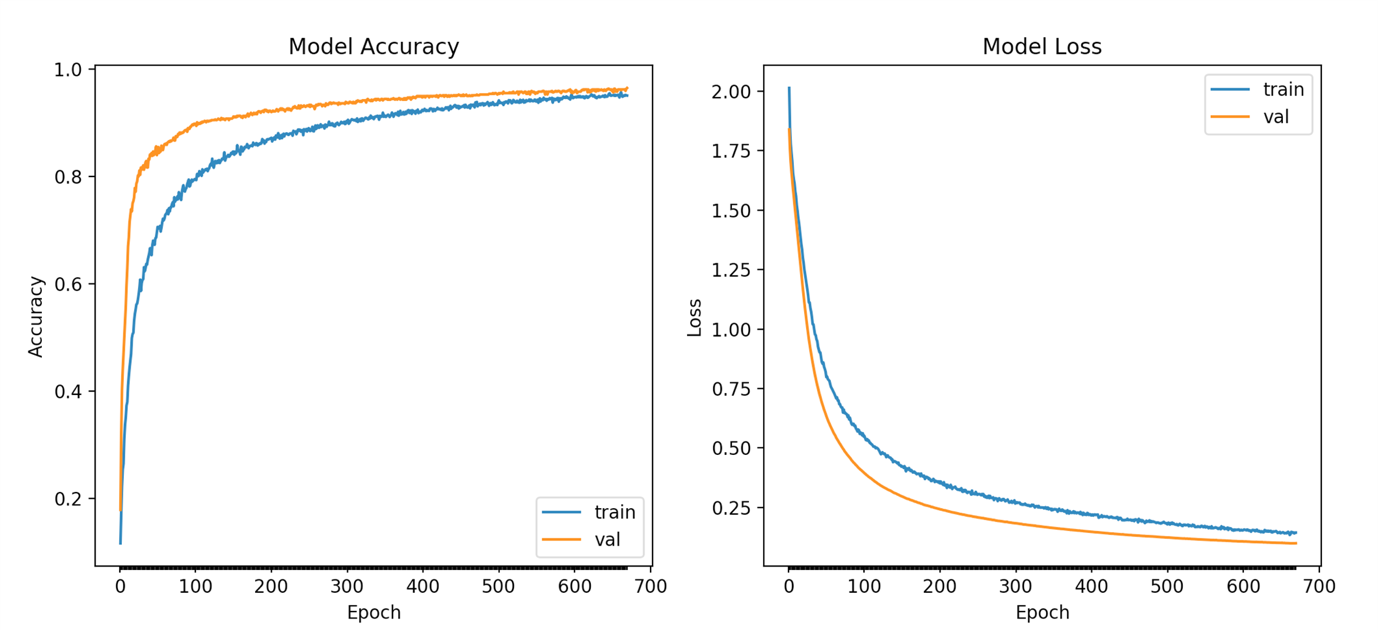
Xuyên suốt quá trình huấn luyện, tôi sử dụng tối ưu SGD với Learning Rate 0.0001 (Learning Rate càng thấp, học càng lâu và sâu), thực hiện mỗi 5 epochs thì lưu lại kết quả tốt nhất. Cũng tại đó, tôi cho tự động dừng huấn luyện sớm (Early Stopping) khi hàm mất mát chênh lệch ít hơn 0.0001. Phần chênh lệch này là độ xấp xỉ lệch cho phép để tránh mô hình bị Overfitting .

Overfitting là sự phân tích thống kê mà tại đó kết quả quá chính xác so với tập dữ liệu được dùng để phân tích, do đó kết quả này rất nhạy cảm với độ nhiễu dẫn đến những dữ liệu thêm vào sau này khó dự đoán được. Thuật toán Machine Learning càng chạy lâu, model càng phức tạp, càng nhiều bậc lũy thừa được sinh ra để “fit” với dữ liệu training. [Hình 13] cho thấy đường màu xanh có bậc lũy phức tạp mới có thể cho ra nhiều đường cong mà tại đó **hàm mất mát đạt giá trị cực tiểu cục bộ** - từ model này nếu cho kiểm tra với bộ dữ liệu traning thì sẽ cho **độ mất mát lớn**. Do đó ta kỳ vọng đường cong màu đen sẽ là một đường đơn giản hơn sao cho **độ mất mát vừa đủ** để kết quả cũng tương thích với độ nhiễu cũng với **độ mất mát vừa đủ.**



Hình 13 Ví dụ về Overfitting - mô tả sự phân loại dữ liệu dựa vào các đường ranh

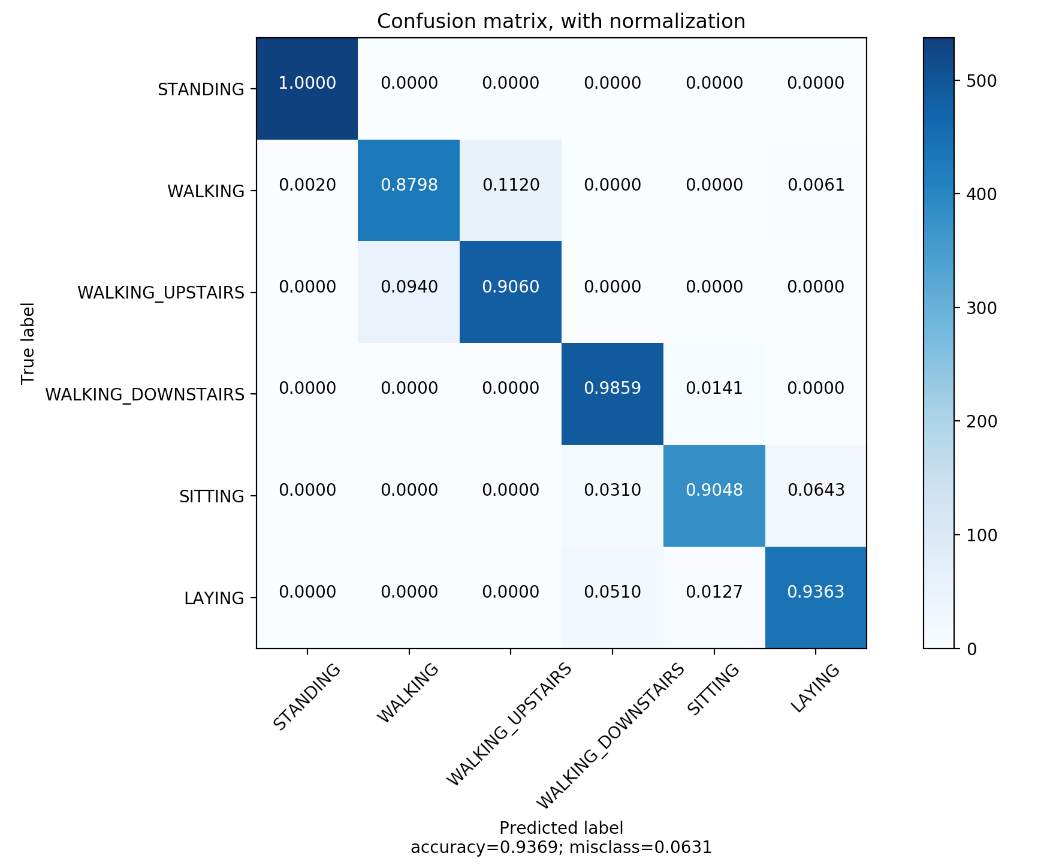
Nói ngắn gọn, Overfitting là khi model “học vẹt” và mất đi tính tổng quát, làm mất đi ý nghĩa Machine Learning khi không thể dự đoán được nhiều trường hợp khác.



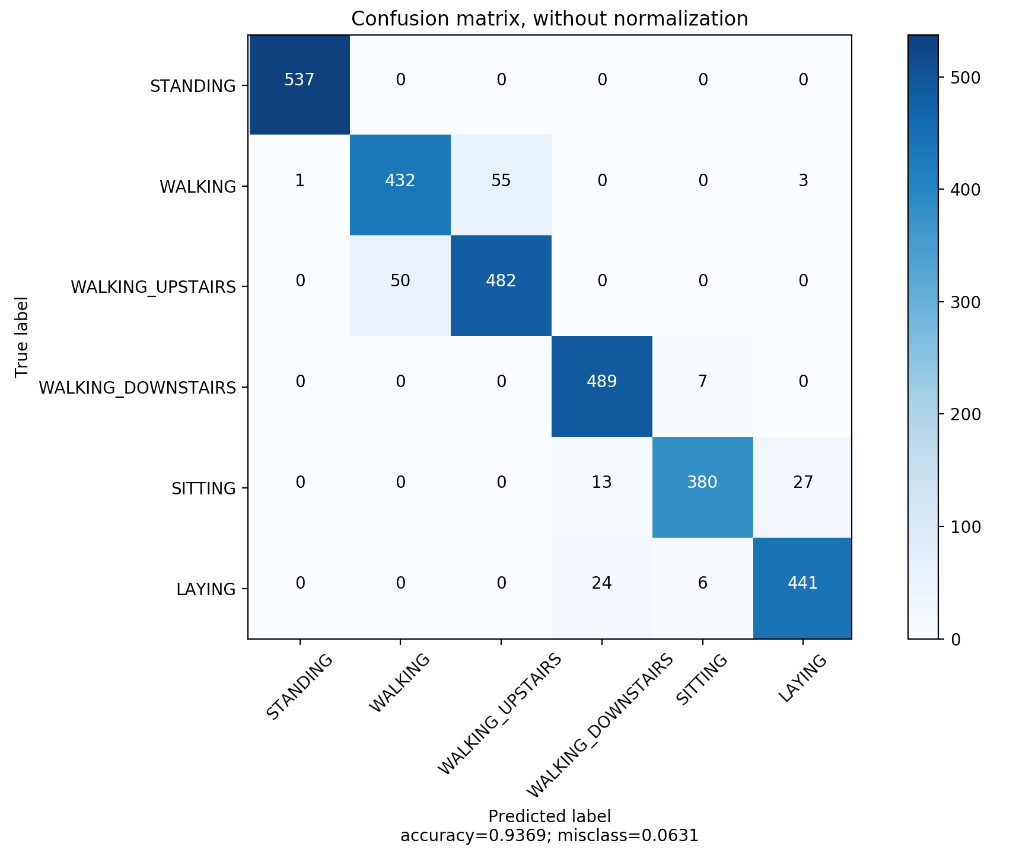
Hình 14 Quá trình học với Simple Neural Network

### Kết quả huấn luyện với Simple Neural Network

Sau khi mô hình được huấn luyện, tôi cho mô hình dự đoán với bộ testing, đạt được độ khả quan, tuy nhiên Walking và Walking upstair vẫn còn nhầm lẫn nhưng ở mức độ chấp nhận được.



Hình 15 Kết quả sau khi huấn luyện Simple Neural Network 1



Hình 16 Kết quả sau khi huấn luyện Simple Neural Network 2

Tuy nhiên, trong quá trình chuyển đổi nhúng mô hình sang Android, tôi gặp sơ sót khi không thể chuyển được toàn bộ 561 vector từ bộ dữ liệu thô được thu thập từ 3 trục Accelerometer và 3 trục Gyroscope. Vì thế tôi quyết định bảo lưu kết quả này và chuyển sang hướng tiếp cận kế tiếp.

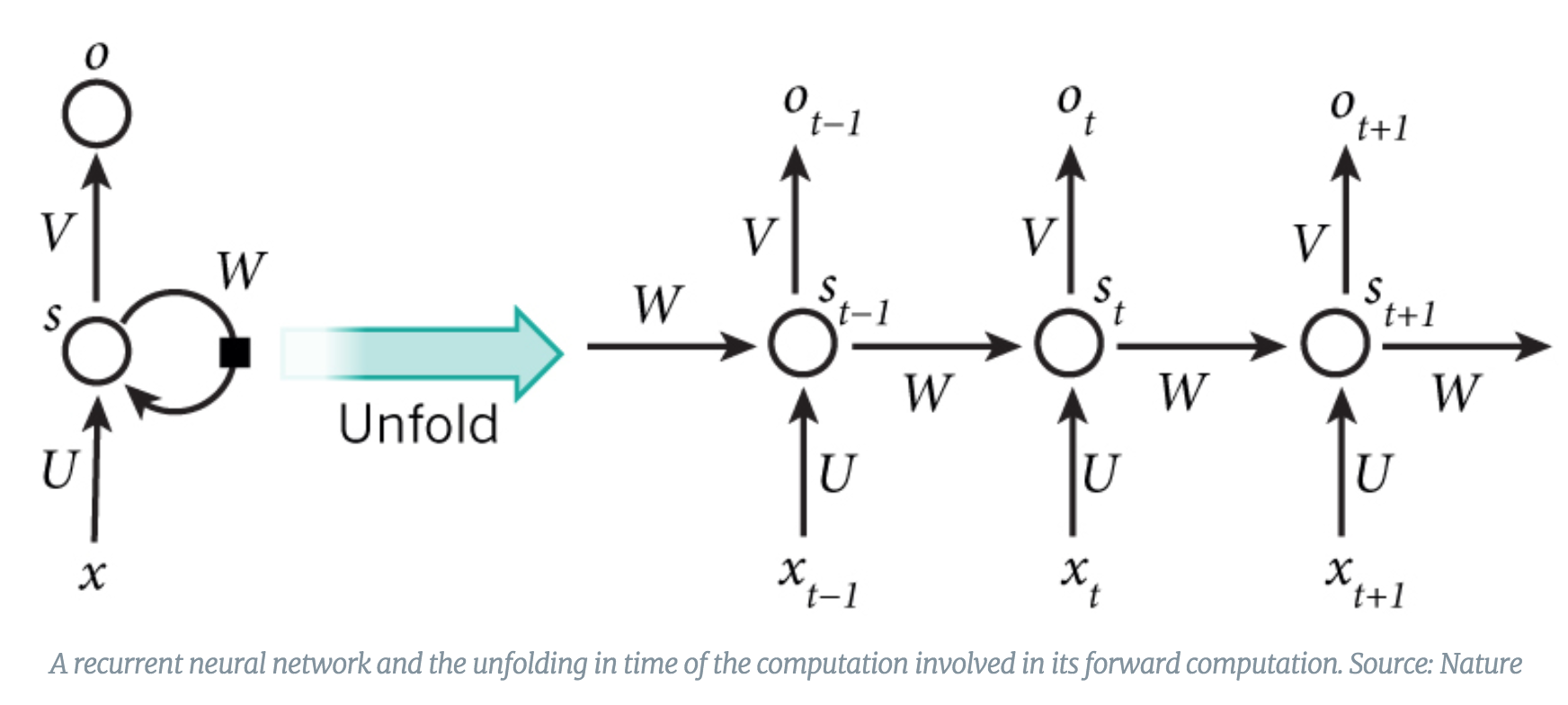
### Tiếp cận từ bộ dữ liệu thô

Trong bộ dữ liệu thô, tôi chia thành 2 bộ:

* Bộ testing và validation: 30%
* Bộ training: 70%

### Thuật toán Recurrent Neural Network với Keras

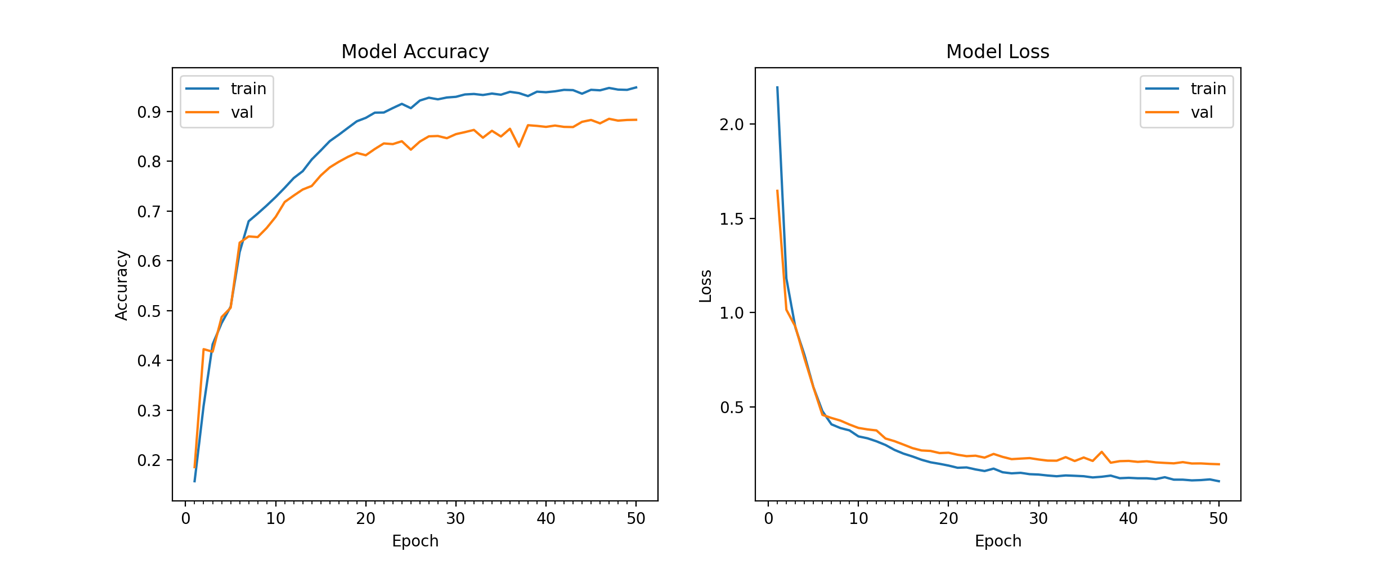
Thuật toán Recurrent Neural Network – RNN còn gọi mạng neural hồi quy được ứng dụng rất nhiều trong những gợi ý từ tiếp theo của một từ - hoặc một câu. Ví dụ khi tôi search “Hôm nay trời” thì từ ngữ tiếp theo được gợi ý là “đẹp”, “mưa”, “nắng”. Ý tưởng của thuật toán này là nhằm “nhớ lại” và tự triển khai ra nội dung phù hợp. Kỹ thuật sử dụng trong trường hợp này là “Many to One” để chuyển đổi một dãy chuỗi vector tạo thành một hoạt động.



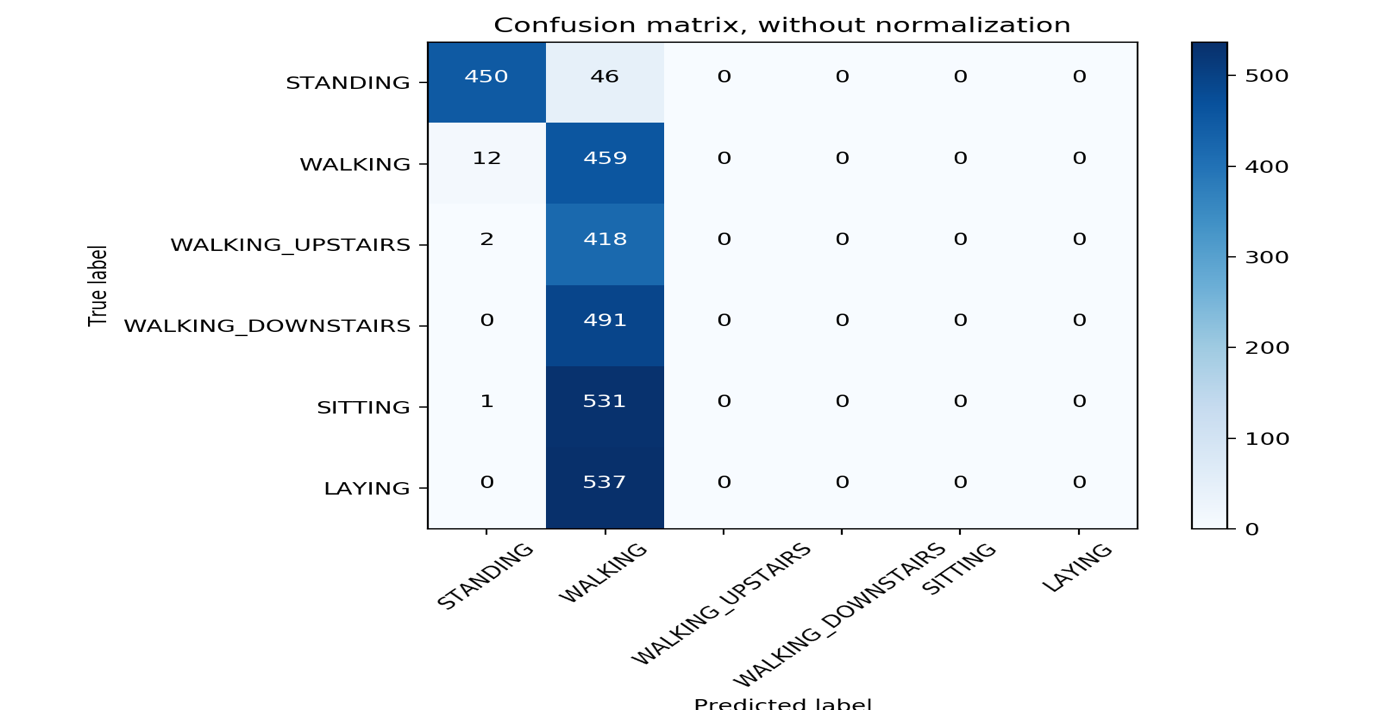
Hình 17 Ý tưởng thuật toán RNN - Feedforward

### Kết quả huấn dùng RNN

Nhược điểm của phương pháp này là training rất lâu, mỗi epoch có thể tốn đến 5 phút để thực thi. Tuy nhiên, phương pháp này ko hiệu quả do kết quả rất kém



Hình 18 Tiến trình huấn luyện bằng RNN



Hình 19 Kết quả huấn luyện bằng RNN

Từ confusion matrix tôi nhận thấy kết quả rất kém chỉ có thể nhận biết đứng hoặc ngồi, còn lại là bị chồng chéo nhau không được như kỳ vọng. Vì vậy tôi chuyển sang một dạng nâng cấp của RNN là LSTM.

### Thuật toán Long-Short Term Mermory

Mạng LSTM (Hải, 2017) [[8]](#footnote-8) - Long-short term memory là một hình thái nâng cấp đặc biệt hơn RNN khi RNN không thể nhớ được với một khoảng cách xa. Ví dụ ở trường hợp này: “Tôi có một chiếc xe hiệu \_\_\_ được sản xuất ở Nhật Bản, chiếm đa số xe phổ thông ở Việt Nam”, như vậy tại chỗ trống có thể là Honda, Toyota,… Thì RNN hoàn toàn không thể giải quyết. LSTM sẽ có khả năng thu thập ở “xa”, cụ thể là ở những câu sau “chiếm đa số” và “Việt Nam”, như vậy khả năng dự đoán ra Honda là rất chắc chắn. Do hoạt động của con người dựa trên nhiều bộ tần số khác nhau trong một dãy tần số liên tục vì thế LSTM được khuyến khích dùng để đưa ra dự đoán chính xác hơn. Để hiểu được bên trong LSTM hoạt động thì thực sự rất phức tạp, vì vậy nên tôi mặc định thừa kế thuật toán từ Keras nhằm tránh lan man. Mô hình cơ bản của hướng này gồm 2 lớp

|  |
| --- |
| model = Sequential()  model.add(LSTM(n\_hidden = 32, input\_shape=(timesteps = 128, input\_dim = 9)))  model.add(Dropout(0.5))  model.add(Dense(n\_classes = 6, activation='sigmoid')) |

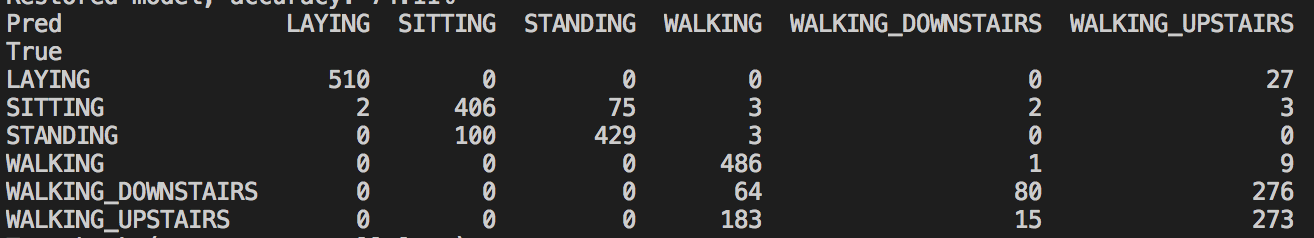
Lớp đầu tiên sử dùng LSTM với đầu vào n\_hidden, timesteps là những chỉ số mặc định, riêng input\_dim là 9, cụ thể:

* 3 trục Accelerometer mà tại đó có thể phân tích thành 2 phần nhỏ mà tôi đã đề cập ở [0]:
  + 3 trục Gravity
  + 3 trục Body Acceleration
* 3 trục Gyroscope

Lớp thứ 2 sử dụng sigmoid là một yếu tố đầu ra mặc định của LSTM, sau đó đưa đầu ra là 6 hoạt động cơ bản.

### Kết quả huấn luyện dùng LSTM

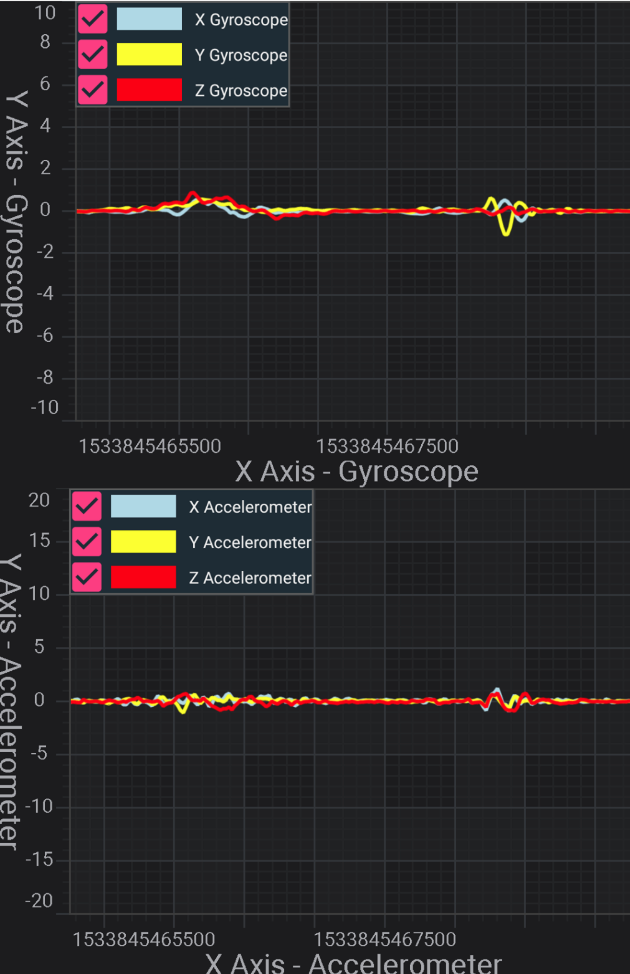
Trải qua 50 epochs, kết quả cho ra độ chính xác khoảng 74.11%, không thật sự tốt nhưng vẫn đủ chấp nhận để sử dụng vào Android. Tuy nhiên, sự nhầm lẫn giữa đi lên và xuống cầu thang vẫn còn mập mờ với đi bộ.



Hình 20 Kết quả huấn luyện bằng LSTM

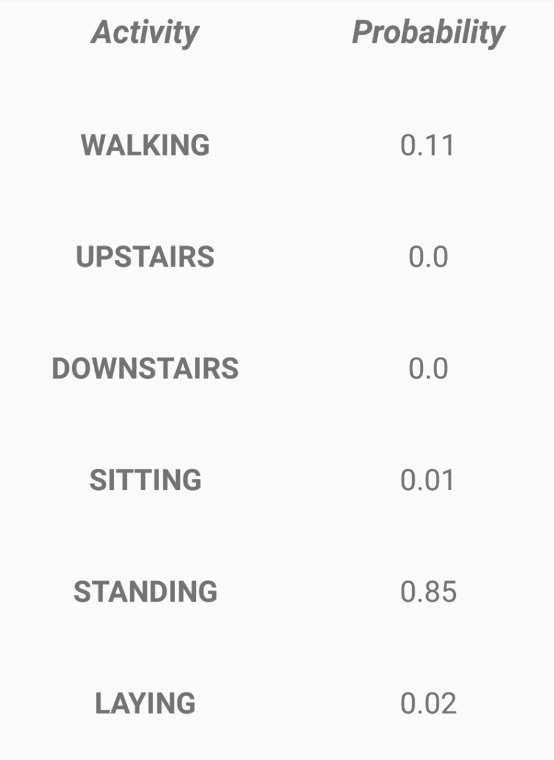
### Chuyển đổi, nhúng mô hình vào Android và kết quả

Mô hình từ Keras sau chuyển đổi không thực sự có thể dùng được trong Android mà cần phải qua một bước chuyển đổi sang tệp mà thư viện Tensorflow Android cho phép dùng. Sau đó tôi tiến hành thu thập dữ liệu thô trên Android và tạo một đồ theo thời gian thực như hình dưới.



Hình 21 Đồ thị Gyroscope và Accelerometer theo thời gian thực

Tiếp theo tôi sử dụng thư viện bộ lọc Butterworth (Porr, 2016) [[9]](#footnote-9) và bộ lọc Median (KalebKE, 2016) [[10]](#footnote-10) trên Android nhằm xử lý nhiễu ở tín hiệu thô. Bước cuối cùng là ghép tín hiệu thu được vào mô hình để xử lý.



Hình 22 Kết quả nhận diện 6 hoạt động cơ bản

Kết quả mô hình này từ LSTM không thực sự tốt khi độ chính xác chỉ ~76%, tuy nhiên đã phần nào đó nhận diện được hoạt động hằng ngày cơ bản.

### Những ứng dụng cơ bản về sức khỏe

Có được những nhận diện cơ bản, tôi tập trung vào những ứng dụng cơ bản về sức khỏe, cụ thể đối tượng áp dụng là đốt Calorie trong mỗi hoạt động.

Theo tính toán của nhóm nghiên cứu mặc định mỗi giây trung bình đi được 1.5 bước, theo (Howmany.wiki, n.d.) [[11]](#footnote-11), mỗi 1 bước chân xấp xỉ 76.2 cm, như vậy vận tốc xấp xỉ khoảng 114.3 cm/s 🡺 1.143m/s = 4.1148 km/h. Tôi sử dụng đơn vị MET (Metabolic equivalent, 2018) trong mỗi hoạt động, dựa vào công thức sau để chuyển đổi sang lượng Calorie bị đốt:



Hình 23 Công thức tính lượng Calorie bị đốt dựa vào MET

Theo (Captain Calculator, 2018) [[12]](#footnote-12), hoạt động sau lần lượt tiêu tốn:

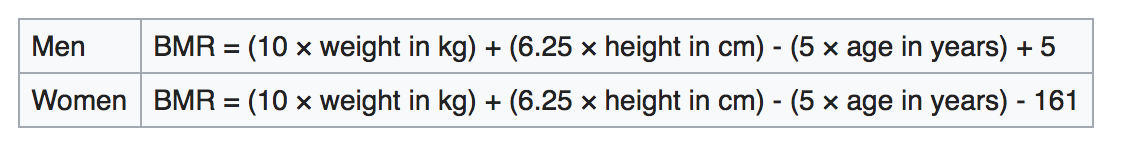
* Đi bộ (4km/h) 2.9 MET
* Nằm, ngủ 0.9 MET
* Ngồi xem tivi 1.0 MET
* Ngồi làm việc 1.5 MET
* Lên cầu thang 4 MET
* Xuống cầu thang 3.5 MET

Do đa phần người ngồi thường sẽ làm việc nhiều hơn là xem tivi, nên tôi mặc định lấy 1.5 MET là thước đo của hoạt động ngồi.

Như vậy, giả sử A là nam có thể trạng 80kg, cao 1m8, 20 tuổi, mỗi ngày thực hiện ngồi làm việc và ngồi rãnh tổng cộng 14 tiếng, ngủ 7 tiếng, đi bộ bình thường 3 tiếng, lên cầu thang 3 phút, xuống cầu thang 3 phút thì sẽ có:

* Ngồi 14 tiếng: 1.5 \* 80 \* 14 = 1680 cal
* Ngủ 7 tiếng: 0.9 \* 80 \* 7 = 504 cal
* Đi bộ 3 tiếng: 2.9 \* 80 \* 3 = 696 cal
* Lên cầu thang 3 phút: 4 \* 80 \* (3/60) = 16 cal
* Xuống cầu thang 3 phút: 3.5 \* 80 \* (3/60) = 14 cal

🡺 Tổng 2910 cal bị đốt trong những hoạt động này



Hình 24 Công thức tính BMR theo Harris-Benedict (Wikipedia, 2018) [[13]](#footnote-13)

Một cách tính khác là sử dụng BMR - là phương thức để từ đó tính toán mức độ Calorie cần thiết để giữ cân nặng. Tôi mặc định hằng số 1.53 PAL (Physical Activity Level) ở đây là làm việc nhẹ, văn phòng. Vì vậy ta có:

* BMR = (10\*80) + (6.25\*180) – (5\*20) + 5 = 1830
* Năng lượng cần cung cấp mỗi ngày là = 1.53 \* 1830 = 2799.9 Calorie cần nạp mỗi ngày để giữ cân. Nếu nhiều hơn thì sẽ tăng cân và ngược lại.

So sánh 2 cách tính thông qua BMR và MET, ta thấy gần tương đương nhau 2910 ~ 2799.9, do đó khi sử dụng Android để nhận diện hoạt động thường ngày và đưa ra được tính toán trên MET thì hoàn toàn đáng tin cậy. Giả sử theo lý thuyết BMR A đang giữ cân với chế độ ăn uống ~2799.9, nếu hoạt động thường ngày theo MET có sự thay đổi nhiều hơn, ví dụ như lên cầu thang nhiều hơn, thì A sẽ đốt Calories nhiều hơn (>2799.9) => Rút ra kết luận cơ bản A có xu hướng giảm cân nếu theo nhịp độ làm việc này.

Tính năng tính lượng Calorie trên được tích hợp vào ứng dụng Health Care Tracking mà tôi đã phát triển vào năm ngoái khi thực hiện đồ án chuyên ngành A. Ứng dụng cho phép thu thập thông tin sức khỏe chẳng hạn như nhập Huyết Áp, lượng thức ăn hằng ngày, Chỉ số Creatinine,… Sau đó thông quá một quá trình tính toán (chủ yếu là sử dụng hồi quy tuyến tính) để đưa ra xu hướng hiện tại của cơ thể, đưa ra báo cáo những lần huyết áp tăng đã tiêu thụ những thức ăn gì và ngược lại. Cơ sở dữ liệu được sử dụng trên Cloud Computing – Firebase do đó đảm bảo đầy đủ về mặt bảo mật cũng như đồng nhất dữ liệu thông qua những lần đồng bộ với thiết bị Android.

### Hướng phát triển trong tương lai

Trong tương lai, dự án này có thể cải thiện thêm về mặt thuật toán nhằm đạt độ chính xác cao nhất. Cho đến hiện nay do lượng kiến thức vẫn chưa nhiều nên tôi vẫn chưa dự định sẽ sử dụng thuật toán gì mới để cải tiến, tuy nhiên tôi đang xem xét một số thuật toán tối ưu để phụ trợ độ chính xác và cần nghiên cứu thêm nhiều về Toán Tối Ưu để hiểu nó. Tiếp theo, tôi muốn tự mình xây dựng thêm những hoạt động khác trong 6 hoạt động nghiên cứu, cụ thể đo đạc được người dung đi với vận tốc bao nhiêu, quãng đường bao xa? Sự phối hợp này cần có sự tham gia của cảm biến GPS nhằm định vị được thời điểm của người ở những vị trí khác nhau, từ đó tính toán được những chỉ số phù hợp.

Ngoài ra xem xét sử dụng đồng hồ thông minh Smartwatch hoặc vòng đeo tay thông minh Band để tự thu thập và xây dựng một bộ dữ liệu thô, nghiên cứu thêm nhiều hướng xử lý tín hiệu sao cho đạt độ chuẩn cao nhất. Về mặt ứng dụng, Smartwatch và Band có ứng dụng rộng rãi hơn do có thể đo được nhịp tim, kết hợp Gyroscope và Accelerometer có thể nhận diện cụ thể hoạt động thể thao như đánh cầu, bơi lội, đá banh. Tuy nhiên, thực tế để xây dựng bộ dữ liệu này hoàn toàn không dễ, cần rất nhiều tình nguyện viên tham gia với chi phí lớn, cần sự hợp tác nghiên cứu của nhiều người trên nhiều lĩnh vực rộng hơn vì kiến thức về xử lý tín hiệu của tôi còn có hạn.

Càng nhiều hoạt động được nhận diện, mức độ tính toán lượng tiêu thụ Calorie sẽ càng chi tiết hơn, thậm chí nếu lượng Calorie về thức ăn hằng ngày được đầy đủ, việc đưa ra thời gian giảm cân, tăng cân sẽ được chính xác từng ngày.

Song song với việc nhận diện hoạt động của con người, đề tài này còn có thể khai thác sự tương quan giữa các chỉ số sức khỏe khác như lượng đường trong máu, huyết áp để từ đó chứng minh con người càng siêng tập thể thao thì sẽ càng có được một sức khỏe tốt và ngược lại.

## Các kết luận và đề nghị

Về cơ bản, tôi đã đạt được 3 mục tiêu chính:

* Mục tiêu 1: Đã nghiên cứu và tái dựng (reproduce) được mẫu dữ liệu từ những bài báo nghiên cứu
* Mục tiêu 2: Đã nghiên cứu nhiều thuật toán để training trên mẫu dữ liệu có sẵn. Sau đó đánh giá độ chính xác và đưa ra mô hình tối ưu. Mô hình tối ưu được chuyển đổi sang dạng nhúng vào Android, các vector dữ liệu đã qua lọc nhiễu được ánh xạ trực tiếp vào mô hình Android để đưa ra dự báo với độ chính xác khoảng 74.11%
* Mục tiêu 3: Đã áp dụng để đo mức tiêu thụ Calorie hằng ngày nhằm đưa ra mối liên hệ giữa các hoạt động với chỉ số sức khỏe đang có.

Qua đề tài này, tôi đã có thể tự nghiên cứu từ những bài báo quốc tế, học được cách thừa kế những thành quả kiến thức nhằm vận dụng vào ý tưởng của mình một cách toàn vẹn hơn so với lập trình ứng dụng thuần túy. Tôi cũng học được cách sử dụng những kỹ thuật hiện đại, mới nhất của Machine Learning, so về lịch sử, lĩnh vực này không hẳn là mới mà đã có từ thế kỷ trước, tuy nhiên nó được gợi lại nhiều hơn ở hiện tại do có rất nhiều tiến bộ vượt bậc, rất nhiều tập đoàn đã nỗ lực đầu tư nhằm đưa con người sang một kỷ nguyên mới – Kỷ nguyên trí tuệ nhân tạo.

Để đề tài này đạt được cấp độ cao hơn, ngoài những mục tôi đã trình bày trong hướng phát triển tương lai, nếu được phép tiếp tục đề tài, tôi cần xây dựng một bộ dữ liệu chuyên nghiệp hơn nữa thay vì dùng bộ dữ liệu của nhóm nghiên cứu.

## Tài liệu tham khảo

BELLE. (2014, 2 16). *Fitness tracker and sleep*. Retrieved 10 8, 2018, from Exist.io blog: https://exist.io/blog/fitness-tracker-sleep/

Captain Calculator. (2018, 8 8). *Calories burned on stairs calculator*. Retrieved 10 9, 2018, from Captain Calculator: https://captaincalculator.com/health/calorie/calories-burned-stairs-calculator/

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, & Jorge L. Reyes-Ortiz. (2012). *A Public Domain Dataset for Human Activity Recognition Using Smartphones.* University of Genova - DITEN; Universitat Politècnica de Catalunya - CETpD.

Hải, Đ. M. (2017, 10 20). *[RNN] LSTM là gì?* Retrieved 9 24, 2018, from Blog Đỗ Minh Hải: https://dominhhai.github.io/vi/2017/10/what-is-lstm/

Howmany.wiki. (n.d.). Retrieved 9 7, 2018, from Howmany.wiki: https://www.howmany.wiki/u/How-many--cm--are-in--1--step

KalebKE. (2016). *Acceleration Explorer*. Retrieved 8 13, 2018, from Github: https://github.com/KalebKE/AccelerationExplorer

*Metabolic equivalent*. (2018, 10 8). Retrieved 10 9, 2018, from Wikipedia: https://en.wikipedia.org/wiki/Metabolic\_equivalent

Nayak, D. J. (n.d.). *Digital Image Processing.* Retrieved 8 10, 2018, from BITS Pilani Dubai Campus University: http://universe.bits-pilani.ac.in/uploads/JNKDUBAI/ImageProcessing7-FrequencyFiltering.pdf

Nguyen, L. T. (2017). *Ứng dụng thu thập thông tin sức khỏe bằng Smartphone.* Đại học Hoa Sen, Khoa học và Kỹ thuật.

Porr, B. (2016). *IIR Filter Library Written in Java*. Retrieved 8 12, 2018, from Github: https://github.com/berndporr/iirj

Tiệp, V. H. (n.d.). *Softmax Regression*. Retrieved 5 20, 2018, from Machine Learning Cơ Bản: https://machinelearningcoban.com/2017/02/17/softmax/

UCI Machine Learning. (2017). *Human Activity Recognition with Smartphones*. Retrieved 8 1, 2018, from Kaggle: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones/home

Wikipedia. (2018, 3 20). *Harris–Benedict equation*. Retrieved 10 9, 2018, from Wikipedia: https://en.wikipedia.org/wiki/Harris%E2%80%93Benedict\_equation

1. BELLE. (2014, 2 16). *Fitness tracker and sleep*. Retrieved 10 8, 2018, from Exist.io blog: https://exist.io/blog/fitness-tracker-sleep/ [↑](#footnote-ref-1)
2. Nguyen, L. T. (2017). *Ứng dụng thu thập thông tin sức khỏe bằng Smartphone.* Đại học Hoa Sen, Khoa học và Kỹ thuật. [↑](#footnote-ref-2)
3. Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, & Jorge L. Reyes-Ortiz. (2012). *A Public Domain Dataset for Human Activity Recognition Using Smartphones.* University of Genova - DITEN; Universitat Politècnica de Catalunya - CETpD. [↑](#footnote-ref-3)
4. Nayak, D. J. (n.d.). Digital Image Processing. Retrieved 8 10, 2018, from BITS Pilani Dubai Campus University: <http://universe.bits-pilani.ac.in/uploads/JNKDUBAI/ImageProcessing7-FrequencyFiltering.pdf> [↑](#footnote-ref-4)
5. UCI Machine Learning. (2017). *Human Activity Recognition with Smartphones*. Retrieved 8 1, 2018, from Kaggle: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones/home [↑](#footnote-ref-5)
6. Keras là một thư viện được phát triển bằng Python cho phép xây dựng mô hình nhanh và có khả năng tương thích cao với những thư viện Machine Learning như Tensorflow của Google [↑](#footnote-ref-6)
7. Tiệp, V. H. (n.d.). *Softmax Regression*. Retrieved 5 20, 2018, from Machine Learning Cơ Bản: https://machinelearningcoban.com/2017/02/17/softmax/ [↑](#footnote-ref-7)
8. Hải, Đ. M. (2017, 10 20). *[RNN] LSTM là gì?* Retrieved 9 24, 2018, from Blog Đỗ Minh Hải: https://dominhhai.github.io/vi/2017/10/what-is-lstm/ [↑](#footnote-ref-8)
9. Porr, B. (2016). *IIR Filter Library Written in Java*. Retrieved 8 12, 2018, from Github: https://github.com/berndporr/iirj [↑](#footnote-ref-9)
10. KalebKE. (2016). *Acceleration Explorer*. Retrieved 8 13, 2018, from Github: https://github.com/KalebKE/AccelerationExplorer [↑](#footnote-ref-10)
11. Howmany.wiki. (n.d.). Retrieved 9 7, 2018, from Howmany.wiki: https://www.howmany.wiki/u/How-many--cm--are-in--1--step [↑](#footnote-ref-11)
12. Captain Calculator. (2018, 8 8). Calories burned on stairs calculator. Retrieved 10 9, 2018, from Captain Calculator: https://captaincalculator.com/health/calorie/calories-burned-stairs-calculator/ [↑](#footnote-ref-12)
13. Wikipedia. (2018, 3 20). *Harris–Benedict equation*. Retrieved 10 9, 2018, from Wikipedia: https://en.wikipedia.org/wiki/Harris%E2%80%93Benedict\_equation [↑](#footnote-ref-13)