ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Phạm Văn Hệ

ĐÁNH GIÁ HIỆU NĂNG CỦA CÁC THUẬT TOÁN HỌC MÁY TRONG BÀI TOÁN ĐỊNH VỊ SỬ DỤNG CƯỜNG ĐỘ TÍN HIỆU

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
Ngành: Kỹ thuật máy tính

HÀ NỘI - 2022

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Phạm Văn Hệ

ĐÁNH GIÁ HIỆU NĂNG CỦA CÁC THUẬT TOÁN HỌC MÁY TRONG BÀI TOÁN ĐỊNH VỊ SỬ DỤNG CƯỜNG ĐỘ TÍN HIỆU

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
Ngành: Kỹ thuật máy tính

Cán bộ hướng dẫn: TS. Đinh Thị Thái Mai

HÀ NỘI - 2022

LỜI CAM ĐOAN

Tôi tên là Phạm Văn Hệ, sinh viên lớp QH-2018-I/CQ-K2, ngành Kỹ thuật máy tính. Tôi xin cam đoan đề tài Đồ án tốt nghiệp "*Đánh giá hiệu năng của các thuật toán học máy trong bài toán định vị sử dụng cường độ tín hiệu"* là công trình nghiên cứu của tôi dưới sự hướng dẫn của TS. Đinh Thai Mai và Th.S. Dương Ngọc Sơn. Các phần sử dụng tài liệu tham khảo trong đồ án này đã được tôi nhắc đến trong mục Tài liệu tham khảo. Các số liệu, kết quả trình bày trong đồ án này là hoàn toàn trung thực.

Nếu có bất kỳ gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đề tài của mình.

Hà Nội, ngày 20 tháng 11 năm 2022

Người cam đoan

Phạm Văn Hệ

LỜI CẢM ƠN

Trải qua bốn năm học tập, nghiên cứu tại Khoa Điện tử viễn thông, Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội, tôi xin được gửi lời cảm ơn chân thành nhất đến Ban Giám hiệu nhà trường, Ban Chủ nhiệm khoa cùng toàn thể các thầy, cô đã tận tình hướng dẫn, truyền đạt những tri thức khoa học, kinh nghiệm thực tiễn cho mình. Đặc biệt nhất, tôi muốn gửi lời tri ân sâu sắc đến TS. Đinh Thái Mai và Th.S. Dương Ngọc Sơn đã hướng dẫn, góp ý để tôi có thể hoàn thiện đề tài đồ án này.

Tôi cũng xin được gửi lời cảm ơn đến gia đình, các anh chị, các bạn, các em trong Khoa Điện tử viễn thông nói riêng và Trường Đại học Công nghệ nói chung đã luôn chia sẻ, động viên tôi để tôi có thể hoàn thiện bản thân cũng như trau dồi kinh nghiệm sống.

Cuối cùng, tôi xin gửi lời cảm ơn đến toàn thể các bạn sinh viên lớp QH-2018-I/CQ-K2 đã luôn hỗ trợ tôi cả trong học tập lẫn rèn luyện. Nhờ đó, tôi có thể vượt qua khó khăn, hoàn thành cuộc hành trình bốn năm rưỡi đại học khó quên.

Hà Nội, ngày 20 tháng 11 năm 2022

Sinh viên

Phạm Văn Hệ

TÓM TẮT

Ngày nay, định vị trong nhà đã thu hút được sự chú ý lớn của các nhà nghiên cứu trong những năm gần đây, chúng tỷ lệ thuận với sự phát triển các ngành công nghệ mới nổi như IoT, rô-bốt, tự động hóa, … Hệ thống định vị toàn cầu(GPS) là công nghệ được sử dụng rộng rãi cho môi trường ngoài trời với độ chính xác khá cao. Thế nhưng khi sử dụng trong nhà, tín hiệu từ vệ tin bị vật liệu cản trở, GPS trở nên không đáng tin cậy, do đó đối với trong nhà cần sử dụng tín hiệu khác để định vị. Chúng cần đáp ứng các yêu cầu như đơn giản, phổ biến, dễ dàng tiến hành, Bluetooth và Bluetooth năng lượng thấp thỏa mãn các yêu cầu trên. Vì vậy, khóaluận này trình bày các phương pháp định vị trong nhà dựa trên cường độ tín hiệu (RSS) sử dụng kỹ thuật lấy dấu vân tay đồng thời đánh giá so sánh chúng. Kết quả thu nhận được cao nhất với lần lượt các mạng học sâu như mạng KNN đạt độ chính xác 86.6%, mạng nơ-ron 90.6%, mạng LSTM 93.6% và cao nhất là mạng tích chập CNN độ chính xác lên tới 95,5% với sai số nhỏ hơn 0.28 m.

Từ khóa: *indoor positioning system, BLE beacon, RSS Fingerprinting, Neural Network, convolution neural network, recurrent neural network, long-short term memory.*

MỤC LỤC

MỤC LỤC HÌNH ẢNH

MỤC LỤC BẢNG

Danh sách từ viết tắt

Từ viết tắt	Từ đầy đủ	Ý nghĩa
RSSI	Received Signal Strength Indicator	Cường độ tín hiệu
IPS	Indoor positioning system	Hệ thống định vị trong nhà
UWB	Ultra-wideband	Công nghệ bang thông siêu rộng
TOA	Time of arrival	Thời gian đến
TDOA	time difference of arrival	chênh lệch thời gian đến
TOF	Time of fly	Thời gian bay
ANN	Artificial neural network	Mạng nơ-ron nhân tạo
CNN	Convolution neural network	Mạng nơ-ron tích chập
RNN	Recurrent neural network	Mạng nơ-ron hồi quy
LSTM	Long-short term memory	Mạng bộ nhớ dài ngắn

CHƯƠNG 1. Giới thiệu

1.1. Đặt vấn đề

Hệ thống định vị trong nhà (IPS-Indoor positioning system) là một mạng lưới các thiết bị được sử dụng để xác định vị trí người hoặc vật thể mà GPS và các công nghệ vệ tinh khác thiếu độ chính xác hoặc bị lỗi hoàn toàn trong các môi trường trong tòa nhà nhiều tầng, sân bay, ngõ, nhà để xe hay vị trí dưới lòng đất. Gần đây, định vị trong nhà đã trở thành một trong những chủ đề nghiên cứu được quan tâm nhiều trong những năm trở lại đây và đã có những hệ thống được thương mại hoá. Một vài ứng dụng tiêu biểu của hệ thống định vị trong nhà:

- Tìm địa điểm trong các tòa nhà văn phòng lớn, các nhà trường đại học, khu trung tâm, viện bảo tàng, bệnh viện…
- Tình huống khẩn cấp: điều hướng cứu hộ và khoanh vùng tình huống khẩn cấp.
- Theo dõi người và tài sản- bệnh nhân, trẻ em, khách tham quan, du khách, ví dụ theo dõi theo hành lý tại các sân bay; giao nhận vận chuyển hàng hóa và theo dõi container trong kho, bến cảng, sân bay…. xác định vị trí nhà máy văn phòng và bệnh viện.
- Các ứng dụng xã hội: tìm người hay tìm chỗ mua sắm, hỗ trợ đỗ xe trong nhà.
- Ứng dụng trong lĩnh vực quảng cáo.

Hiện nay đã có rất nhiều những nghiên cứu về hệ thống định vị trong nhà với các phương thức truyền tín hiệu khác nhau như Wifi, Ultrawideband (UWB), Bluetooth, BLE, … cho các thiết bị nhỏ gọn như điện thoại thông minh, … Nhưng vẫn còn hạn chế do thiết kế cũng như ràng buộc giữa hiệu năng và giá thành. Chúng tôi mong chờ sự cải tiến về hiệu năng độ chính xác của hệ thống định vị trong nhà với một chi phí rẻ.

1.2. Yêu cầu và mục tiêu

Yêu cầu của hệ thống định vị trong nhà: là phải xác định được vị trí của các mục tiêu di động theo thời gian thực với độ chính xác cao. Hệ thống định vị trong nhà cần bao gồm các đặc trưng sau:

- Phần lớn các mục tiêu đều chuyển động trong không gian thực.
- Tín hiệu RSS nhận được bị ảnh hưởng bởi mất dữ liệu và giả dữ liệu do đặc tính không hoàn hảo của thiết bị cũng như bị ảnh hưởng bởi môi trường thực tế.

Mục tiêu của đồ án là xây dựng được hệ thống định vị trong nhà sử dụng tín hiệu Bluetooth tiêu thụ ít năng lượng (BLE), dễ dàng cài đặt cũng như độ bao phủ tín hiệu rộng. Bằng cách đo RSS từ các thiết bị, chúng ta sẽ được sử dụng để ước lượng vị trí của mục tiêu bằng việc sử dụng các thuật toán trong học sâu và phương pháp dấu vân tay (Fingerprinting).

1.3. Tổng quan kiến trúc đồ án

Chương 2: Cơ sở kiến thức của hệ thống định vị trong nhà. Thảo luận và so sánh các phương pháp truyền tín hiệu không dây khác nhau như wi-fi, sóng siêu âm (Ultrasonic), Bluetooth…Đồng thời đề cập tới các kĩ thuật định vị trong nhà, các thuật toán cũng như các kiến thức cơ bản về học sâu và các phương pháp tiền xử lý.

Chương 3: Kiến trúc của hệ thống định vị trong nhà bao gồm lưu đồ thuật toán, phần cứng và các thuật toán được cài đặt đánh giá các thuật toán khác nhau.

Chương 4: Kết luận đề cập tới ưu điểm và những vẫn đề còn tồn tại, đề xuất những ý tưởng mới phát triển trong tương lai.

CHƯƠNG 2. Kiến thức cơ sở

2.1. Tín hiệu truyền trong kỹ thuật định vị trong nhà

2.1.1. Wi-fi

Wi-fi là một họ các giao thức truyền thông mạng không dây được xây dựng dựa trên tiêu chuẩn của IEE 802.11 [1] được sử dụng rộng rãi trong cuộc sống với phạm vi sóng có thể bao phủ từ 50-100m tùy theo không gian cũng như dải tần 2.4 GHz hoặc 5.0 GHz và băng thông 20MHz-40MHz [2]. Việc sử dụng WiFi như một trong những tín hiệu dùng để định vị trong nhà xuất phát từ những lợi thế của nó như phạm vi bao phủ rộng, độ phổ biến cao được lắp đặt ở hầu hết các hộ gia đình cũng như tòa nhà, khả năng mở rộng mô hình dễ dàng. Chi phí cho mỗi thiết bị mạng (Access Point) không quá cao có giá từ 20-90. Thế nhưng cũng có một vài bất lợi của WiFi cần nhắc tới như tín hiệu sóng có thể bị hấp thụ bởi các vật liệu như thạch cao, bê tông cốt thép dẫn tới việc suy hao tín hiệu làm cho giảm độ chính xác và ổn định đặc biệt với các phương pháp định vị dựa trên RSS [3].

2.1.2. Công nghệ băng thông siêu rộng (Ultra-wideband-UWB)

Công nghệ băng thông siêu rộng là công nghệ truyền vô tuyến nhanh an toàn và tiêu thụ năng lượng thấp được xây dựng trên tiêu chuẩn IEE802.15 [1]. UWB hoạt động bằng cách bên phát gửi hàng loạt các xung UWB trên tần số phổ rộng, bên thu sẽ nhận được các xung tương ứng và chuyển thành dữ liệu dựa trên việc lắng nghe một chuỗi xung do bên phát gửi. Với đặc điểm là được truyền tin trên một băng thông siêu rộng (>500 MHz), các xung được truyền đi trong khoảng một hai nano giây giúp UWB đạt được độ chính xác theo thời gian thực. Với ứng dụng định vị trong nhà, UWB có lợi thế hơn các thiết bị WiFi hay Bluetooth là khả năng đâm xuyên qua các chướng ngại vật như con người tường đồ đạc và sử dụng phương pháp tính thời gian truyền (TOF) làm cho nó đạt được độ chính xác và đáng tin cậy cao. Tuy nhiên nhược điểm lớn nhất của các thiết bị này là giá thành vẫn còn cao, các thiết bị di động có khả năng thu nhận sóng chỉ có ở các mẫu thiết bị đời mới như iPhone 11 trở lên…. Mặc dù vậy đây chính là công nghệ mà sẽ được ứng dụng rất nhiều trong việc truyền tin gần cũng như định vị trong nhà trong tương lai.

2.1.3. Bluetooth

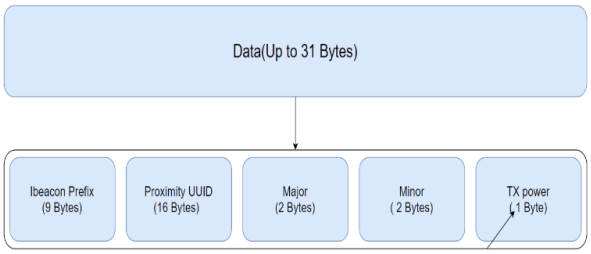
Bluetooth là kỹ thuật truyền thông không dây được thiết kế truyền thông cho phạm vi ngắn 10-15 m được xây dựng trên tiêu chuẩn IEE802.15.1 [1]. Được ứng dụng rộng rãi cho các thiết bị kết nối không dây như điện thoại, máy tính, hay các thiết bị ngoại vi như bàn phím chuột, máy in…. Ưu điểm của bluetooth đó là tiêu thụ ít năng lượng trung bình khoảng 100mW với băng thông dưới 800Kbps và tần số 2.4GHz [2]. Thêm vào đó Bluetooth là một tiêu chuẩn được áp dụng rộng rãi cho hầu hết các thiết bị. Nhược điểm do sử dụng công nghệ dựa trên RSS nên tín hiệu bluetooth cũng dễ bị ảnh hưởng trong môi trường như có người đi, vật cản.

2.1.4. Bluetooth tiêu thụ ít năng lượng-Bluetooth Low Energy (BLE)

Bluetooth tiêu thụ ít năng lượng (BLE) được phát triển từ công nghệ Bluetooth nhưng tiêu thụ ít năng lượng hơn. BLE được thiết kế để cài đặt trên các thiết bị nhỏ sử dụng những nguồn điện từ pin với ứng dụng nhỏ gọn, truyền dữ liệu không liên tục [2]. BLE sử dụng 40 kênh 2MHz để truyền dữ liệu bằng cách sử dụng điều chế dịch tần Gaussian (GFSK) làm cho việc nhảy tần số tạo ra ít vấn đề hơn so với truyền thông Bluetooth tiêu chuẩn. Đối với định vị trong nhà, ta khai thác RSS cùng với kỹ thuật dấu vân tay để cải thiện ước lượng tính toán định vị trong nhà. Mặc dù WiFi dễ dàng có sẵn ở các khu vực công cộng nhưng nó khá khó để xác định vị trí các điểm APs dẫn tới hạn chế các kỹ thuật định vịtrong nhà. Hiện nay trên thị trường, với mong muốn cài đặt tối thiểu các APs mà vẫn cho độ chính xác tin cậy cao, BLE đang trở nên phổ biến do tính linh động cũng như dễ dàng lắp cố định tại các vị trí mong muốn. Do đó trong khóa luận tốt nghiệp này tôi sử dụng BLE iBeacon (một trong những thiết bị BLE phổ biến trên thị trường ngày nay) làm dữ liệu để định vị trong nhà.

2.1.5. iBeacon

Hình 2-1. Ảnh ibeacon



Hình 2-2. Thông tin của một gói tin quảng bá

iBeacon là giao thức được Apple phát triển từ năm 2013 dựa trên công nghệ BLE. iBeacon cho phép các ứng dụng di động có thể lắng nghe được các tín hiệu phát ra từ beacon và phản hồi lại nó. Với ưu điểm sử dụng BLE, một thiết bị iBeacon có thể hoạt động trong vòng 3 năm chỉ với một viên pin, đồng thời BLE sử dụng các gói tin quảng bá thông qua sóng vô tuyến với tần số khoảng 10Hz [6]. Đồng thời iBeacon đã định nghĩa chi tiết hơn về BLE quảng bá (advertising). Một gói tin quảng bá bao gồm 4 thông tin:

1. UUID: Thông số này là một chuỗi string 16byte dùng để định danh các hãng sản xuất. Ví dụ beacon do 1 bên thứ 3 sản xuất nên có cùng thông tin UUID. Lúc này các ứng dụng di động có thể phân biệt beacon của hãng đó với các hãng khác.
2. Major: Thông số này là một chuỗi 2byte dùng để chia nhỏ các nhóm beacon trong cùng 1 hãng.
3. Minor: Thông số này cũng là một chuỗi 2byte dùng để phân biệt các beacon với nhau. Các beacon trong cùng UUID và Major nên có thông số Minor là duy nhất.
4. TxPower: Thông số này dùng để tính toán khoảng cách của người dùng đến các beacon. TxPower định nghĩa như độ mạnh của tính hiệu mà thiết bị nhận được khi cách beacon 1 mét. TxPower phụ thuộc vào từng hãng sản xuất và sẽ được cài đặt sẵn trong các beacon [6].

Trong đồ án tốt nghiệp này tôi sử dụng Major và TxPower là 2 dữ liệu đầu vào cho hệ thống định vị trong nhà [8].

2.2. Các phương pháp đo

Để định vị được vị trí của đối tượng, ta cần xác định được khoảng cách của đối tượng tới các điểm cố định, có 4 cách cơ bản để tính được khoảng cách bao gồm: thời

gian đến (TOA- Time of Arrival), chênh lệch thời gian đến (Time Difference of Arrival-TDOA), RSSI.

Phương pháp TOA và TDOA sử dụng các mối quan hệ hình học dựa trên khoảng cách hoặc chênh lệch khoảng cách giữa thiết bị thu di động và một số thiết bị phát cố định để xác định tọa độ vị trí của mục tiêu di động. Dữ liệu để ước tính khoảng cách được lấy từ thời gian đến của tín hiệu vô tuyến tại một hoặc nhiều máy thu. Phương pháp TOA sử dụng thời gian truyền trực tiếp giữa các máy phát và máy thu khoảng cách để tìm ra khoảng cách, trong khi đó phương pháp TDOA tính toán vị trí từ sự khác biệt của thời gian đến được đo trên các cặp đường truyền giữa mục tiêu và thiết bị thu phát cố định [7]. Cả TOA và TDOA đều dựa trên nguyên tắc đo khoảng cách theo thời gian truyền tín hiệu (TOF), trong đó tham số cảm biến, khoảng thời gian, được chuyển đổi thành khoảng cách bằng cách nhân với tốc độ lan truyền. Trong TOA, ước tính vị trí được tìm thấy bằng cách xác định các điểm giao nhau của các vòng tròn hoặc hình cầu mà tâm của chúng nằm tại các trạm cố định và bán kính là khoảng cách ước tính đến mục tiêu. TDOA định vị mục tiêu tại các giao điểm của các hypebol hoặc hypebol được tạo ra với các điểm tại mỗi trạm cố định của một cặp.

Phương pháp RSSI dựa trên sự mất mát năng lượng trong quá trình truyền tín hiệu, giá trị RSSI được liên kết với kích thước của sự suy giảm tín hiệu. Giá trị RSSI càng nhỏ thì càng ít bị suy giảm. Các phương pháp phổ biến để tính toán được giá trị của RSSI dựa trên mô hình lý thuyết như mô hình suy hao trên đường truyền không gian tự do, mô hình chuẩn logarit. Mô hình này được xây dựng dựa trên một vài giá trị của RSSI của các vị trí được biết dựa trên thông tin vị trí và khoảng cách truyền tín hiệu của nó.

Mô hình suy hao trên đường truyền không gian tự do là 1 môi trường truyền dẫn lý tưởng biết rằng xung quanh anten có chân không vô hạn, năng lượng truyền tín hiệu chỉ liên quan đến khoảng cách truyền, có mối quan hệ tuyến tính giữa năng lượng truyền tín hiệu và khoảng cách truyền, mô hình này không ảnh hưởng đến các chướng ngại vật và phản xạ phân tán. Công thức tính toán suy hao năng lượng trên đường đi:

$$\text{Loss } = 32.44 + 10n\log(d) + 10n\log(f) \quad (1)$$

Trong đó:

- Loss: năng lượng suy hao trên đường đi.

- d: khoảng cách truyền tín hiệu (m).
- f: tần số tín hiệu không dây (MHz).

n: hệ số suy giảm đường đi trong môi trường truyền thực tế.

Tuy nhiên trong môi trường thực tế, tín hiệu bị ảnh hưởng bởi sự hấp thụ bởi các chướng ngại vật và sự giao thoa, phản xạ phân tán…. Đặc tính suy giảm của các kênh trong khoảng cách dài tuân theo phân bố chuẩn và thường được sử dụng bởi mô hình khối logarit. Công thức tính toán suy hao:

$$P_L(d) = P_L(d_0) + 10n\log\left(\frac{d}{d_0}\right) + x_\sigma \quad (2)$$

Trong công thức trên, $P_L$là suy hao năng lượng của tín hiệu khi khoảng cách đo là d (m) nó thường là giá trị tuyệt đối được tính bằng đơn vị dBm. $P_L(d_0)$ chỉ suy hao năng lượng khoảng cách là d_0, n là hệ số mất mát ở các mô trường. x_σ là độ lệch chuẩn. Giá trị này càng lớn thì độ chính xác mô hình càng thấp [18]. Cường độ tín hiệu được tín theo công thức:

$$\text{RSSI} = P_t - P_L(d) \quad (3)$$

$$\text{RSSI} = P_t - P_L(d) \quad (4)$$

Trong công thức trên P_t là công suất truyền tín hiệu. $P_L(\text{d})$ chỉ mất mát khi khoảng cách là d được tính bằng dBm.Trong công thức trên ta chỉ cần khoảng cách giữa bên thu và bên nhận có thể tín ra giá trị RSSI theo một công thức nhất định rất đơn giản và dễ dàng có được nhưng lại bị ảnh hưởng rất nhiều bởi môi trường.

Phương pháp	Ưu điểm	Nhược điểm
TOA	Độ chính xác cao	Yêu cầu đồng bộ hóa thời gian trên tất cả máy thu và nhận. Giả định tầm nhìn thẳng giữa thiết bị thu và nhận.
TDOA	Độ chính xác cao. Không yêu cầu đồng bộ thời gian bên thu và nhận.	Giả định tầm nhìn thẳng giữa các thiết bị thu và nhận.
RSS	Đơn giản và rẻ. Không yêu cầu đồng bộ thời gian giữa bên thu và nhận.	Độ chính xác và tin cậy thấp. Dễ bị ảnh hưởng bởi nhiễu và vật chắn

Bảng 2-1. So sánh các phương pháp đo khác nhau [4].

2.3. Phép đo cường độ tín hiệu (RSSI)

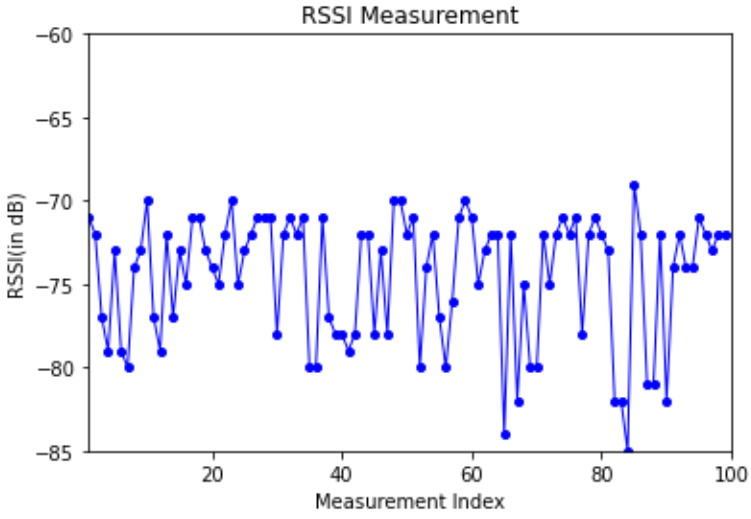
Trong phần này, các vấn đề về cường độ tín hiệu (RSSI) sẽ được phân tích và miêu tả, qua đó đề xuất các phương pháp hay bộ lọc để gia tăng độ chính xác đối với định vị trong nhà.

2.3.1. Vấn đề của RSSI

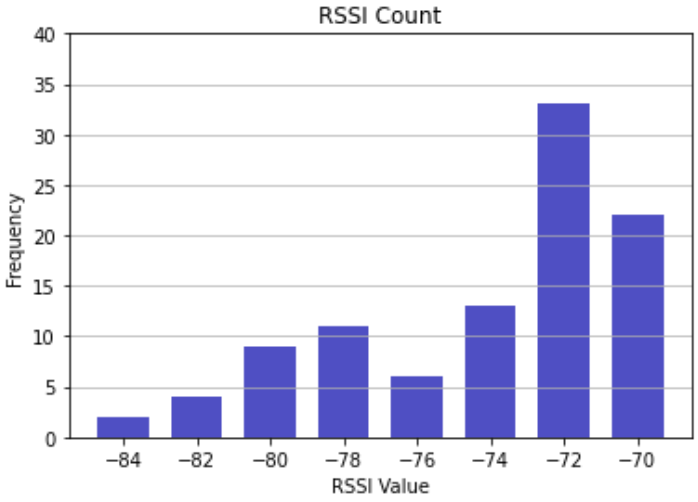
Để đạt được độ chính xác cao cho việc định vị, cần phải có các phương pháp đáng tin cậy để đo lường RSS và ước tính khoảng cách RSS. Các tín hiệu được truyền bởi các thiết bị BLE trong không gian là sóng điện từ và giống như các sóng vô tuyến băng thông thấp khác, chúng bị ảnh hưởng bởi môi trường, tạo ra các kiểu giao tiếp như đa đường và tạo bóng. Các kết quả định vị được phân tích trong đó chứng minh ảnh hưởng của con người (môi trường) của môi trường đối với phép đo RSS ảnh hưởng của nhiều anten thu đối với phép đo RSSI.

2.3.2. Phân tích RSSI

Để phân tích được thuộc tính của RSSI, rất nhiều các phép đo cho một iBeacon được thực hiện trên một vị trí cố định không thay đổi thực hiện liên tục. Kết quả thu được như Hình 2.3 và hình 2.4. Nhìn vào đây ta có thể thấy các giá trị RSSI sẽ dao động liên tục từ -70dB-85dB với độ lệch chuẩn bằng 3.77dB. Trong đó các giá trị sẽ tập trung nhiều trong khoảng từ 70-72dB. Kết quả cho thấy tín hiệu RSSI không quá chính xác, dễ bị ảnh hưởng bởi các yếu tố môi trường, do đó cần có những bộ lọc hoặc các kỹ thuật xử lí các vấn đề tín hiệu không ổn định của RSSI.



Hình 2-2. RSSI qua 100 lần đo

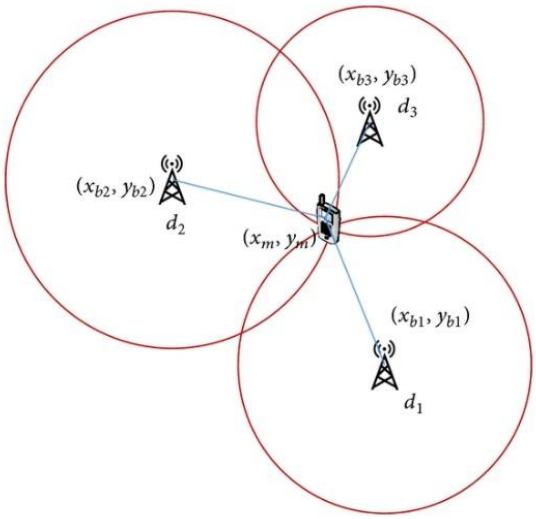


Hình 2-3. Đếm số lần xuất hiện của các giá trị RSSI

2.4. Kỹ thuật định vị trong nhà

2.4.1. Kỹ thuật Triangulation

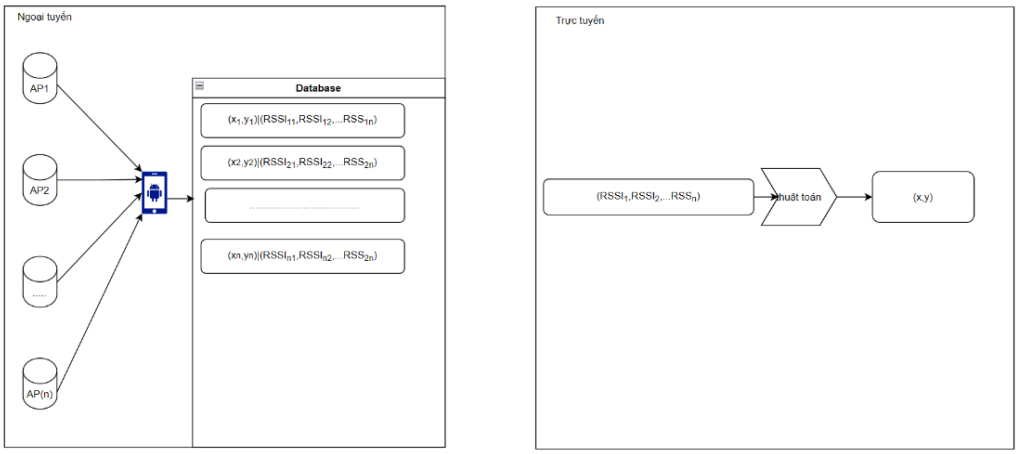
Kỹ thuật lượng giác là một trong những phương pháp định vị cơ bản nhất dựa trên khoảng cách giữa đối tượng và các điểm cố định cho trước để tính ra được vị trí của đối tượng. Phương pháp này tạo ra các vòng tròn có bán kính được xác định bằng sự suy giảm tín hiệu từ thiết bị đầu cuối di động hoặc được đo bằng thời gian lan truyền tín hiệu giữa điểm truy cập và thiết bị đầu cuối di động. Giao điểm của 3 hay nhiều đường tròn đó là vị trí ước tính của đối tượng [9]. Tuy nhiên trong thực tế hầu như ta không có được một điểm giao hoàn hảo duy nhất do sai số trong phép đo hay cường độ tín hiệu bị sai lệch do bị chắn bởi các chướng ngại vật hay người di chuyển.



Hình 2-4. Kỹ thuật lượng giác

2.4.2. Kỹ thuật dấu vân tay-Fingerprinting

Kỹ thuật dấu vân tay là một kỹ thuật mà được sử dụng khá phổ biến trong IPS nhờ những ưu điểm nổi bật như không bị ảnh hưởng bởi bộ truyền cũng như giảm ảnh hưởng của môi trường thực tế lên tín hiệu RSSI so với kỹ thuật lượng giác. Kỹ thuật kỹ thuật dấu vân tay này được xây dựng trên 2 trang thái: trạng thái ngoại tuyến và trạng thái trực tuyến.



Hình 2-5. Kỹ thuật dấu vân tay

Giai đoạn ngoại tuyến: Các giá trịRSSI được thu thập ở tất cả các vị trí khác nhau nhiều lần trong môi trường thực tế ta sẽ nhận được giá trị thực được nhận được trong môi trường thật. Tất cả chúng được lưu trữ trên cơ sở dữ liệu phục vụ cho quá trình xây

dựng bản đồ tín hiệu ở từng các vị trí khác nhau. Thông qua các thuật toán như k-Nearest Neighbor (kNN) hay các mô hình học máy như mạng nơ-ron (Artificial Neural Network-ANN) hay mạng hồi quy (Recurrent Neural Network - RNN). Các giá trị RSSI được đưa vào mô hình cho máy học qua đó ánh xạ được các giá trị RSSI sang vị trí bản đồ thật trên thực tế.

Giai đoạn trực tuyến: Ở gia đoạn này, các thiết bị thu nhận sóng RSSI ở vị trí bất kì trong phạm vi đã được thu thập dữ liệu, thông qua các mô hình học máy chúng có thể giúp ta định lượng được vị trí thực tế mà thiết bị thu nhận sóng đang đứng với độ chính xác cao.

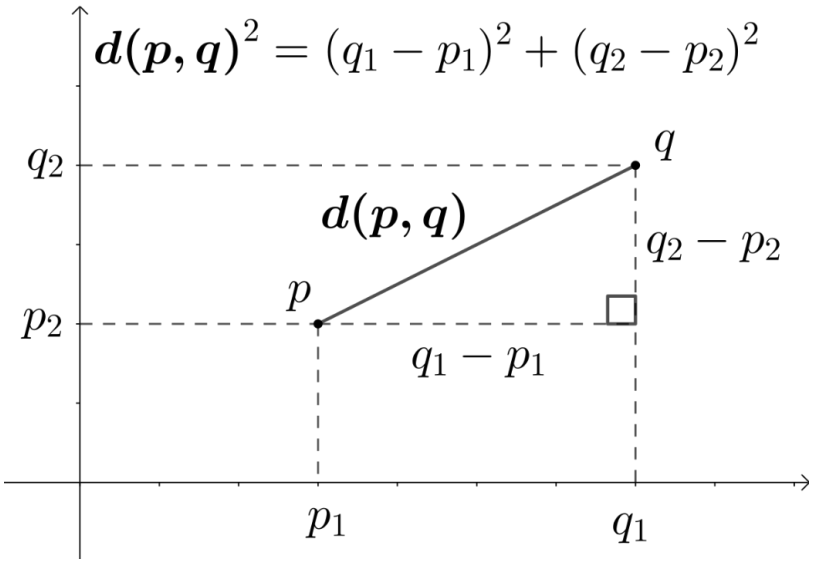
2.5. K-Nearest Neighbor

kNN (k-Nearest Neighbor) là một trong những thuật toán học có giám sát đơn giản nhất, chủ yếu được sử dụng để phân loại dữ liệu trên cơ sở cách phân loại *hàng xóm* của nó. kNN lưu trữ tất cả các trường hợp có sẵn và phân loại các trường hợp mới dựa trên một biện pháp tương tự. k trong kNN là một tham số đề cập đến số lượng các nước láng giềng gần nhất để đưa vào quá trình bỏ phiếu theo đa số (major voting).

Bài toán đặt ra: Tìm cách ước lượng $\mathbb{P}(Y = c \mid X = x)$ trong lân cận của *x với* $x \in \mathbb{R}^d, y \in \{1,2, \dots, C\}$.

Các bước thực hiện:

Bước 1: Lấy k mẫu dữ liệu trong gần *x* nhất trong D (D là tập dữ liệu). Khái niệm "gần" liên quan tới khoảng cách trong không gian $\mathbb{R}^d$.



Hình 2-6. Tính khoảng cách Euclidean

Khoảng cách được sử dụng ở đây là Euclidean:

$$d(x,y) = \| x - y \| 2 = \sqrt{\sum i = 1^d (x_i - y_i)^2} \quad (5)$$

Bước 2: Xấp xỉ $\mathbb{P}(Y = c \mid X = x)$ bằng tỉ lệ nhãn c trong k mẫu đó:

$$\widehat{\mathbb{P}}(Y = c \mid X = x) = \frac{\sum_{i=1}^{n} \mathbb{I}(x_i \in V_k(x)) \mathbb{I}(y_i = c)}{k} \quad (6)$$

Trong đó $V_k(x)$ và lân cận của x chứa k mẫu dữ liệu trong D. Tử số chính là số lượng mẫu dữ liệu trong $V_k(x)$ mà có nhãn là c. Suy ra $h^{KNN}(x)$ chọn nhãn c xuất hiện nhiều nhất trong k mẫu dữ liệu gần x nhất trong D.

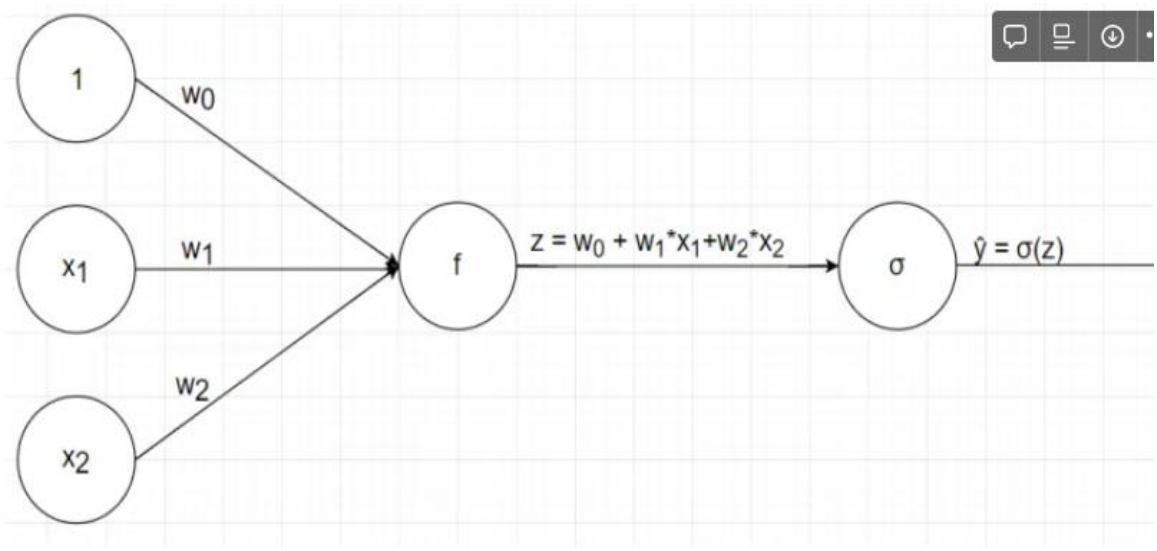
2.6. Mạng nơ-ron

Mạng Nơ-ron (Artificial Neural Network - ANN) là một mô hình toán học được xây dựng dựa trên các mạng nơ-ron sinh học. Nó gồm có một nhóm các nơ-ron nhân tạo (nút) nối với nhau, và xử lý thông tin bằng cách truyền các kết nối và tính giá trị mới tại các nút (cách tiếp cận connectionism đối với tính toán). Trong nhiều trường hợp, mạng nơ-ron nhân tạo là một hệ thích ứng (*adaptive system*) tự thay đổi cấu trúc của mình dựa trên các thông tin bên ngoài hay bên trong qua mạng trong quá trình học.

Để đơn giản hóa sự phức tạp của mạng nơ-ron ,chúng ta cần tìm hiểu từng phần của nó với một lớp đầu vào ($x \in \mathbb{R}^{d+1}$), một lớp trung gianvà lớp đầu ra phân loại hai nhãn.

2.6.1. Thuật toán lan truyền tới

Với $\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} (x \in \mathbb{R}^{d+1})$ là ma trận chứa tập huấn luyện mà mỗi cột x_i là một điểm dữ hiệu trong không gian $d+1$ chiều. Tương ứng với vector trọng số $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \cdots \\ x_n \end{bmatrix} (w \in \mathbb{R}^{d+1})$[12].



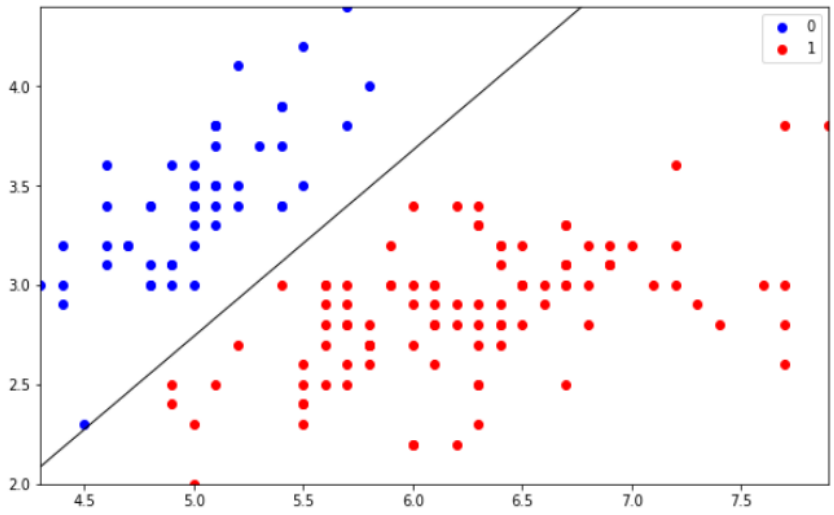
Hình 2-7. Mô hình mạng nơ-ron lan truyền tới

Tổng quát:

$$f = w_1^T x + w_0 = \mathbf{w}_1^T \mathbf{x} = \sum_{i=0}^{n} (x_1 w_1, x_2 w_2, \dots, x_n w_n, w_0) \quad (7)$$

Việc cần làm đó chính là ta cần tối ưu hóa véc-tơ trọng số w sao cho ta tìm được hệ số tạo nên mặt phẳng có thể phân biệt được 2 nhãn.

Giải thích: Tại một thời điểm ranh giới là một siêu phẳng có phương trình: $f(x) = 0$ với w là véc-tơ đặc trưng bằng 1. Ví dụ về một siêu phẳng:



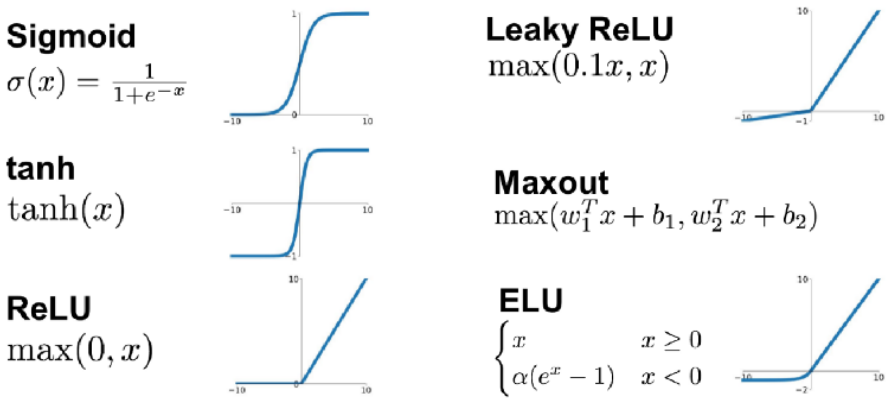
Hình 2-8. Ví dụ về siêu phẳng

Và khoảng cách từ một điểm x_A bất kì đến siêu phẳng sẽ được tính theo công thức:

$$d(x_A) = \frac{|f(x_A)|}{|w|} = \frac{|w^T x + w_0|}{|w|} \left(\text{với } \| w \| = \sqrt{w_1^2 + w_2^2 + \cdots . + w_n^2} \right) > 0 \quad (8)$$

Công thức tổng quát ta không lấy giá trị tuyệt đối như vậy các giá trị nằm cùng phía so với siêu phẳng làm cho hàm số $f(x)$ mang cùng dấu. Qua đó ta cho đi qua hàm kích hoạt như Relu, Sigmoid, Tanh,… để tính được xác xuất và phân loại chúng.

$$y = \sigma(f(x))$$



Hình 2-9. Các hàm kích hoạt

2.6.2. Xây dựng hàm mất mát

Một trong những việc quan trọng của mạng nơ-ron ta cần xây dựng được hàm mất mát theo tham số w bất kỳ. Tham số w là nghiệm tối ưu sao cho nó không gây ra điểm bị lỗi nào (hoặc số điểm lỗi là nhỏ nhất). Như vậy, hàm đếm số lượng điểm bị phân lỗi có thể coi là hàm mất mát của mô hình. Trong bài toán chúng ta cần xử lí là phải xây dựng được hàm mất mát để phân biệt 2 nhãn.

Hàm khả năng (likelihood function) của bộ tham số đối với dữ liệu $D = \{(x_1, y_1), (x_2, y_2)\}$.

$$L(w, w_0) = P(D) = \prod_{i=1}^{n} P(y_i \mid x_i) = \prod_{i=1}^{n} \mu_i^{y_i}(1 - \mu_i)^{1-y_i} \tag{9}$$

Trong đó: μ_{ic} là giá trị của hàm sigmoid được biểu diễn bằng công thức:

$$\mu_{ic} = \frac{1}{1 + e^{-x}} \tag{10}$$

và mã hoá **one-hot** của nhãn $y_{ic} = \mathbb{I}(c = y_i)$ (tức là thay $y_i \in \{y1, y2\}$ bằng $\{0,1\}$.

Ta có hàm khả năng dạng log âm:

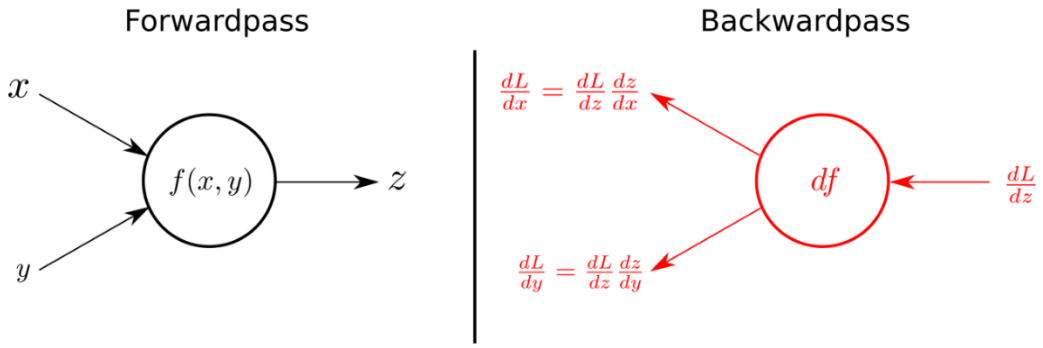
$$\ell(w, w_0) = -\log L(w, w_0) = \sum_{i=1}^{n} - y_i \log \mu_i - (1 - y_i)\log(1 - \mu_i) \tag{11}$$

Để sự sai sót trong phân loại là nhỏ nhất ta cần tối ưu hóa hàm lỗi (NLL hay cross entropy loss) sao cho chúng đạt giá trị nhỏ nhất [13] bằng cách thay đổi giá trị W. Hàm $\ell(\mathbf{W})$ có gradient khá ngắn gọn ta có thể sử dụng cơ chế xuống đồi (go down from the hill) bằng đạo hàm (gradient descent) để tối ưu hóa.

2.6.3. Thuật toán lan truyền ngược

Thuật toán lan truyền ngược là một cách để tính đạo hàm của một biểu thức mà sử dụng quy tắc chuỗi (chain rule). Vấn đề cốt lõi của bài toán mạng nơ-ron đó là khi ta xây dựng được một hàm $f(x)$ nhiều lớp ta muốn tính đạo hàm của $f(x)$ theo x, w qua đó tối ưu hóa hàm lỗi $\ell(\mathbf{W})$.

Sử dụng quy tắc chuỗi, ta có:



Hình 2-10. Quy tắc chuỗi

$$\frac{dL}{dx} = \frac{dL}{dz} * \frac{dz}{dx} \tag{12}$$

Hàm kích hoạt:

$$\mu_i = \sigma(f(x)) \tag{13}$$

Ở lớp phân lớp cuối cùng, chúng ta cần đầu ra là giá trị xác xuất của các giá trị cần dự đoán. Nhưng các tín hiệu của mạng nơ ron có thể là giá trị âm. Vậy nên với bài toán phân loại 2 lớp người ta thường sử dụng hàm sigmoid.

Ưu điểm: Do nó là hàm mũ nên luôn cho giá trị đầu ra là số dương suy ra $\mu_i = \sigma(f(x))$ hàm số luôn đồng biến. Nên ta dựa vào đó để tìm ra xác xuất lớn nhất của các lớp.

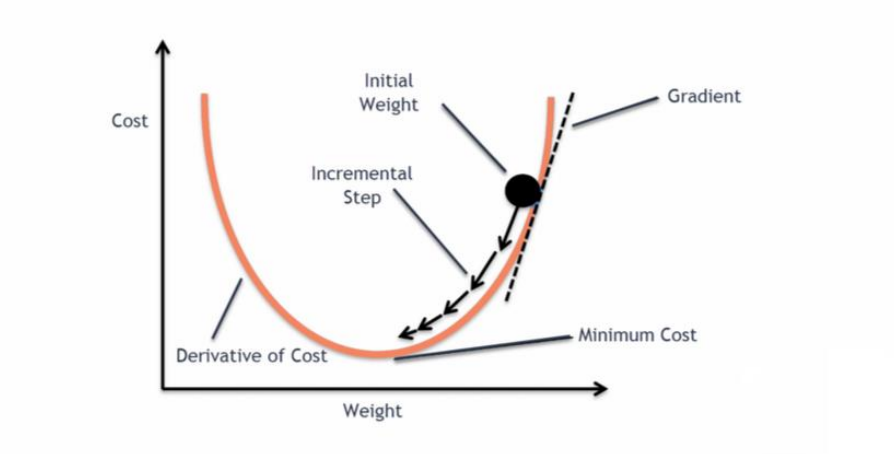
Đạo hàm của hàm sigmoid:

$$\sigma'(z) = \sigma(z)(1 - \sigma(z)) \tag{14}$$

$$\begin{aligned}\nabla_w \ell(w, w_0) &= \sum_{i=1}^{n} \frac{\partial \ell}{\partial \mu_i} \frac{\partial \mu_i}{\partial w} \\ &= \sum_{i=1}^{n} \left(-\frac{y_i}{\mu_i} + \frac{1-y_i}{1-\mu_i}\right) \mu_i(1-\mu_i) x_i \\ &= \sum_{i=1}^{n} (-y_i(1-\mu_i) + (1-y_i)\mu_i) x_i \\ &= \sum_{i=1}^{n} (\mu_i - y_i) x_i\end{aligned} \tag{15}$$

$$\nabla_{w_0} \ell(w, w_0) = \sum_{i=1}^{n} (\mu_i - y_i)$$

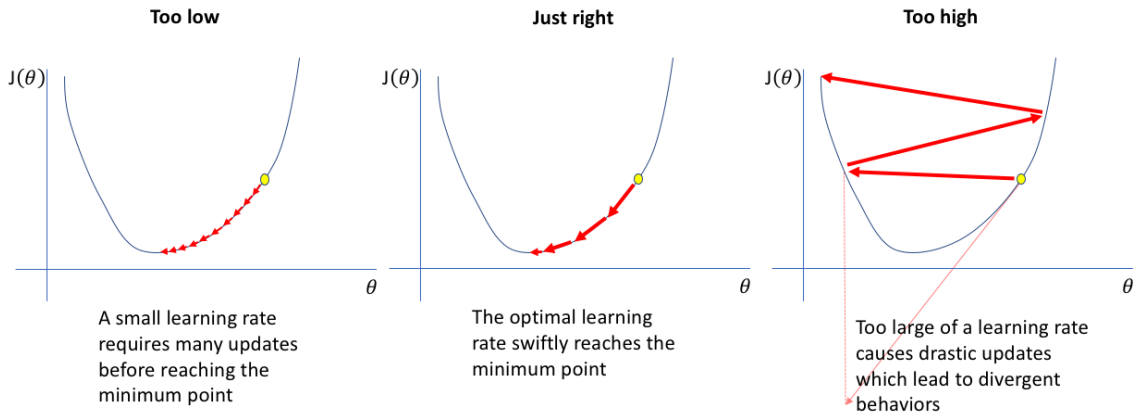
Vì giá trị của hàm cross entropy là hàm lồi nên ta sử dụng phương pháp xuống đồi bằng đạo hàm để tối ưu hóa các hệ số weight hay bias.



Hình 2-11. Phương pháp xuống đồi bằng đạo hàm

$$\begin{aligned} w &\leftarrow w - \lambda \nabla_w \ell(w, w_0) \\ w_0 &\leftarrow w_0 - \lambda \nabla_{w_0} \ell(w, w_0) \end{aligned} \tag{16}$$

Với λ là tốc độ học (learning rate) trong quá trình huấn luyện. Quá trình học nhanh hay chậm phụ thuộc khá nhiều vào λ . Chọn giá trị tốc độ học cũng cần phù hợp, giá trị của nó thường là số dương nằm trong khoảng từ 0 đến 1. Độ lớn của λ sẽ ảnh hưởng trực tiếp tới tốc độ hội tụ của hàm loss tới điểm cực trị toàn cục.



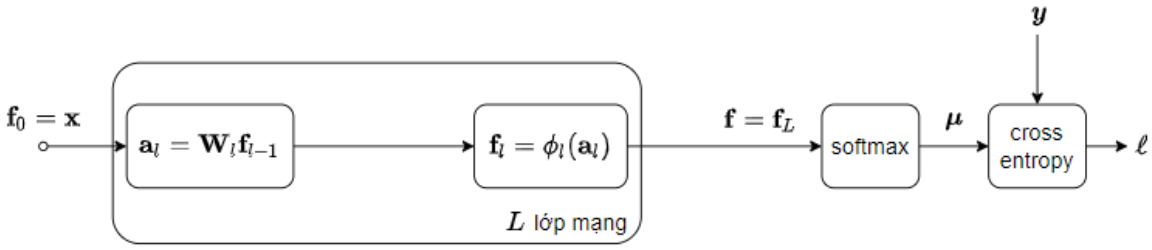
Hình 2-12. Các vấn đề liên quan tới tốc độ học (learning rate)

Với trường hợp λ quá nhỏ dẫn đến các cập nhật đối với trọng số là nhỏ, điều này sẽ giúp tìm tới điểm cực tiểu được chuẩn xác hơn vì bước nhảy là rất nhỏ. Tuy nhiên điều này sẽ khiến mất rất nhiều thời gian để hội tụ, hoặc nếu các bạn biết tới khái niệm cực tiểu cục bộ, thì với learning rate quá nhỏ này có thể dẫn tới việc bị kẹt trong cực tiểu cục bộ không mong muốn.

Với trường hợp λ quá lớn, thuật toán sẽ học nhanh, nhưng có thể thấy thuật toán bị dao động xung quanh hoặc thậm chí nhảy qua điểm cực tiểu.

Sau cùng, việc chọn tốc độ học phù hợp sẽ hài hòa được giữa tốc độ hội tụ của thuật toán cũng như việc tìm ra được điểm cực tiểu.

Trong nghiên cứu này, chúng tôi đã sử dụng mạng nơ-ron nhiều lớp để giải quyết vấn đề.



Hình 2-13. Mô hình mạng nơ-ron nhiều lớp

Phương pháp này phát triển từ bài toán phân lớp nhị phân và ta muốn không chỉ là phân lớp cho dữ liệu phức tạp hơn. Đối với hướng tiếp cận này, chúng tôi sử dụng mạng nơ-ron với 2 lớp ẩn (hidden layers). Quá trình huấn luyện được thực hiện qua 3 bước.

Bước đầu tiên là bước lan truyền xuôi (feed forward). Trong bước này ta thực hiện cho dữ liệu đi thẳng từ đầu tới đầu ra theo các mũi tên tới đầu ra theo công thức:

$$\begin{aligned} \mathbf{f}_0 &= x(\text{ input }) \\ a_l &= W_l f_{l-1}, l = 1,2,3, \dots L \\ f_l &= \sigma(a_l), l = 1,2,3, \dots L \\ \boldsymbol{\mu} &= \mathcal{S}(\mathbf{f}) \end{aligned} \tag{17}$$

Với bài toán phân lớp nhiều lớp người ta thường sử dụng hàm Softmax ở lớp cuối cùng để giá trị xác xuất đầu ra của các lớp cộng lại có tổng bằng 1.

$$\mathcal{S}(f_1, f_2, \dots f_C) = \left[\frac{e^{f_c}}{\sum_{c'=1}^{C} e^{z_{c'}}}\right]_{c=1,2,\dots,C} \tag{18}$$

Hàm khả năng của bộ tham số đối với dữ liệu bằng:

$$\begin{aligned} D &= \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\} \\ L(\mathbf{W}) = \mathbb{P}(D) &= \prod_{i=1}^{n} \mathbb{P}(Y = y_i \mid x_i) \\ &= \prod_{i=1}^{n} \prod_{c=1}^{C} \mu_{ic}^{y_{ic}} \end{aligned} \tag{19}$$

Tương ứng ta sẽ tính được hàm lỗi cross entropy:

$$\begin{aligned} \ell = -\log L(\mathbf{W}) &= -\sum_{i=1}^{n} \sum_{c=1}^{C} y_{ic} \log \mu_{ic} = -y^T \log(\boldsymbol{\mu}^T) \in \mathbb{R} \\ \ell &= -\mathbf{y}^T \log\left(\mathcal{S}\left(\phi_L\left(\mathbf{W}_L \phi_{L-1}\left(\mathbf{W}_{L-1} \dots \phi_1(\mathbf{W}_1 \mathbf{x})\right)\right)\right)\right) \end{aligned} \tag{20}$$

Thuật toán lan truyền lan truyền ngược:

Quá trình lan truyền ngược vẫn khá giống với lan truyền xuôi cho 2 lớp đi tính đạo hàm hàm lỗi. Ta sẽ thực hiện lan truyền ngược từ hàm lỗi qua hàm Softmax, qua từng lớp mạng để tối ưu hóa các hệ số *weight* và *bias*. Để dễ dàng cho việc tính toán đạo hàm với hàm nhiều biến nhiều giá trị ta sử dụng ma trận đạo hàm Jacobian kết hợp cùng với quy tắc chuỗi.

$$\mathbf{J_f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_d} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_d} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_d} \end{bmatrix} \in \mathbb{R}^{n \times d} \tag{21}$$

Hàm hợp nhiều biến:

$$\begin{gathered} (\mathbf{f} \circ \mathbf{g})(\mathbf{x}) = \mathbf{f}(\mathbf{g}(\mathbf{x})) \text{ với } \mathbf{x} \in \mathbb{R}^d, \mathbf{y} = \mathbf{g}(\mathbf{x}) \in \mathbb{R}^n, \mathbf{f}(\mathbf{y}) \in \mathbb{R}^m \\ \mathbf{J}_{\mathbf{f} \circ \mathbf{g}}(\mathbf{x}) = \mathbf{J}_{\mathbf{f}}(\mathbf{g}(\mathbf{x}))\mathbf{J}_{\mathbf{g}}(\mathbf{x}) \end{gathered} \tag{22}$$

Đạo hàm của hàm loss:

$$\mathbf{J}_\ell(\mu) = \frac{-y^T}{\mu^T} \tag{23}$$

Đạo hàm của của hàm softmax:

$$\begin{aligned} \mathbf{J}_\mu(f) = \frac{\partial \mu_i}{\partial f_j} &= \frac{u'v - v'u}{v^2} = \frac{\mathbb{I}(i=j)e^{f_j}\sum_{c=1}^{C} e^{f_c} - e^{f_j}e^{f_i}}{(\sum_{c=1}^{C} e^{f_c})^2} \\ &= \mathbb{I}(i=j)\mu_j - \mu_i\mu_j = \begin{cases} (1-\mu_i)\mu_i & i=j \\ -\mu_i\mu_j & i \neq j \end{cases} \\ &= \begin{bmatrix} (1-\mu_1)\mu_1 & -\mu_2\mu_1 & \cdots & -\mu_K\mu_1 \\ -\mu_1\mu_2 & (1-\mu_2)\mu_2 & \cdots & -\mu_K\mu_2 \\ \vdots & \vdots & & \vdots \\ -\mu_1\mu_K & -\mu_2\mu_K & \cdots & (1-\mu_K)\mu_K \end{bmatrix} \in \mathbb{R}^{C \times C} \end{aligned} \tag{24}$$

Đạo hàm của hàm mất mát theo f:

$$\mathbf{J}_\ell(f) = \mathbf{J}_\ell(\mu) * \mathbf{J}_\mu(f) \tag{25}$$

Sử dụng mã hóa one-hot các lớp.

$$\mathbf{J}_\ell(\mu) = \frac{-y^T}{\mu^T} \tag{26}$$

$$\left[0,0,0,\dots,\frac{1}{\mu_i},\dots,0\right] \in \mathbb{R}^{1\times C}$$

Đạo hàm của hàm lỗi theo f:

$$\begin{gathered}\mathbf{J}_\ell(f) = \mathbf{J}_\ell(\mu) * \mathbf{J}_\mu(f) \\ = \left[0 \quad 0 \quad 0 \quad 0 \cdots \quad \frac{-1}{\mu_i} \cdots 0\right] * \\ \begin{bmatrix} (1-\mu_1)\mu_1 & -\mu_2\mu_1 & \cdots & -\mu_K\mu_1 \\ -\mu_1\mu_2 & (1-\mu_2)\mu_2 & \cdots & -\mu_K\mu_2 \\ \vdots & \vdots & & \vdots \\ -\mu_1\mu_K & -\mu_2\mu_K & \cdots & (1-\mu_K)\mu_K \end{bmatrix} \\ = [\mu_1 \quad \mu_2 \quad \cdots \mu_i - 1 \quad \cdots \mu_K]\end{gathered} \tag{27}$$

Sau khi tính toán được tới đạo hàm của hàm lỗi theo f. Lúc này việc cần làm đó là tính toán f theo các hệ số weight và bias qua đó để tối ưu các hệ số của từng lớp.

Ta có lớp lan truyền tới của các lớp:

$$\begin{aligned}\mathbf{a}_l &= \mathbf{W}_l\mathbf{f}_{l-1} \in \mathbb{R}^{p_l} \\ \mathbf{f}_l &= \phi_l(\mathbf{a}_l) \in \mathbb{R}^{p_l}\end{aligned} \tag{27}$$

Trong mạng nơ ron này ta sử dụng hàm kích hoạt: $\phi_l(\mathbf{a}_l)$ là hàm ReLu. Đạo hàm của hàm ReLu sử dụng ma trận Jacobian ta có: $\mathbf{J}_{\mathbf{f}_l}(\mathbf{a}_l) = \mathbf{I} \in \mathbb{R}^{p_l\times p_l}$.

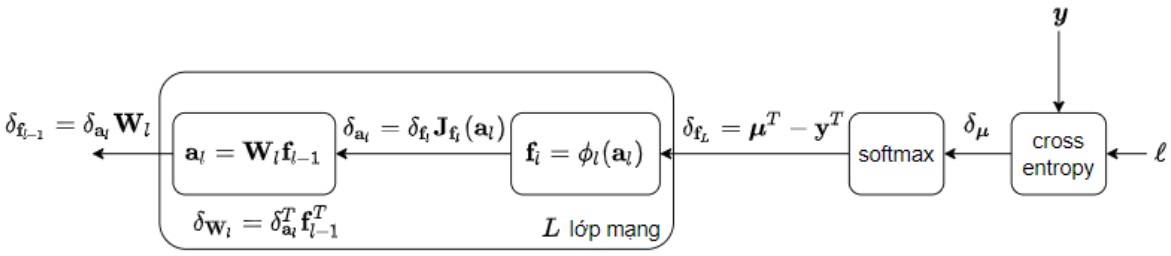
Sau đó đạo hàm của $\mathbf{a}_l$ đối với $\mathbf{W}_l$: $\mathbf{J}_{\mathbf{a}_l}(\mathbf{f}_{l-1}) = \mathbf{W}_l$. Tương ứng ta sẽ có ma trận Jacobian:

$$\frac{\partial a_{li}}{\partial \mathbf{W}_l} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ - & \mathbf{f}_{l-1}^T & - & - \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{p_l\times p_{l-1}} \tag{28}$$

Đến đây, ta sử dụng phương pháp xuống đồi đạo hàm ta sẽ tối ưu được từng hệ số weight của từng lớp:

$$w \leftarrow w - \lambda\frac{\partial a_{li}}{\partial \mathbf{W}_l} \tag{29}$$

Đạo hàm các tầng mạng - lan truyền ngược. Lặp ngược L lần với $l = L, L-1, \dots, 1$.



Hình 2-14. Mô hình mạng nơ-ron đầy đủ

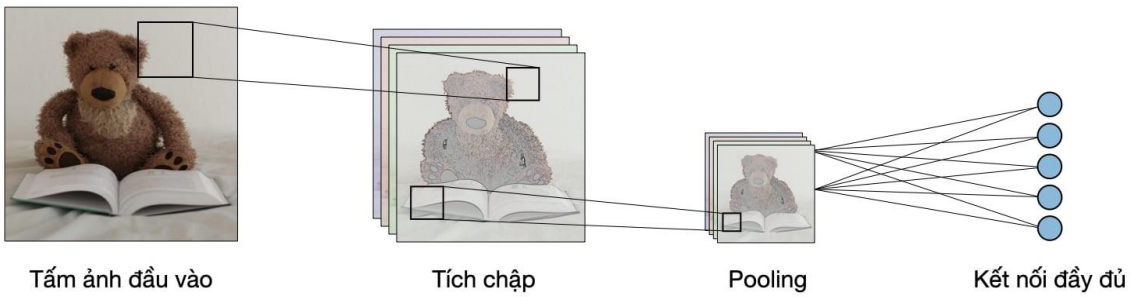
Quá trình lan truyền ngược đạo hàm:

$$\begin{aligned}\delta_{\mathbf{a}_l} &= \delta_{\mathbf{f}_l}\mathbf{J}_{\mathbf{f}_l}(\mathbf{a}_l) \in \mathbb{R}^{1\times p_l}\\ \delta_{\mathbf{W}_l} &= \delta_{\mathbf{a}_l}^T\mathbf{f}_{l-1}^T \in \mathbb{R}^{p_l\times p_{l-1}}\\ \delta_{\mathbf{f}_{l_{l-1}}} &= \delta_{\mathbf{a}_l}\mathbf{W}_l \in \mathbb{R}^{1\times p_{l-1}}\end{aligned} \tag{30}$$

Kết quả: $\delta_{\mathbf{a}_l}, \delta_{\mathbf{W}_l}, \delta_{\mathbf{f}_l}$ với $l = 1,2,\dots L$.Đặc biệt $\delta_{\mathbf{x}} = \delta_{\mathbf{f}_0}$.

2.7. Mạng tích chập (Convolutional neural network)

Mạng tích chập (Convolution Neural Network là mạng nơ-ron mà chúng có thể chia sẻ các tham số. Mạng tích chập sử dụng các bộ lọc (kernel) nhân chập hay trượt trên ma trận đầu vào qua đó tìm ra được các đặc trưng để phù hợp cho nhiều bài toán khác nhau. Một trong số đó là dùng để phân loại. Trong luận văn này, chúng tôi sử dụng mạng tích chập để phân loại các lớp với đầu vào tương ứng. Đầu tiên chúng ta sẽ tìm hiểu cơ bản về kiến trúc truyền thống của một mạng CNN.

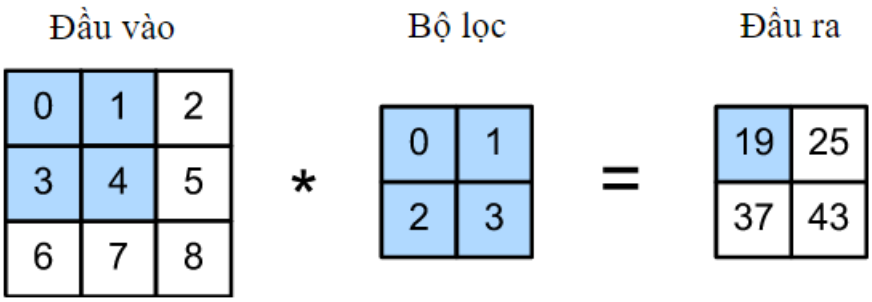


Hình 2-15. Mô hình mạng nơ-ron đầy đủ

Trong toán học phép tích chập giữa hai hàm số f, g.

$$(f * g)(x,y) = \int_{u,v} f(u,v)g(x-u, y-v)dudv = (g * f)(x,y)$$
$$(f * g)[x,y] = \sum_{i,j=-\infty}^{\infty} f[i,j]g[x-i, y-j] \quad (31)$$

Đối với tích chập trong bức ảnh, phép toán mà chúng biểu diễn là phép tương quan chéo (cross corelation).



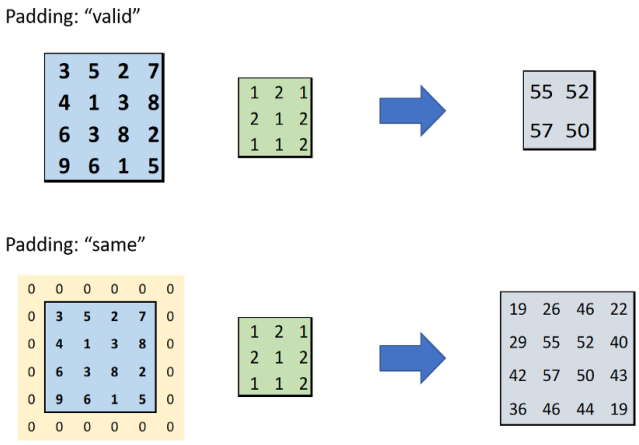
Hình 2-16. Ví dụ về nhân chập

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 & y_2 \\ y_3 & y_4 \end{bmatrix}$$
$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_4 + w_4x_5 \\ w_1x_2 + w_2x_3 + w_3x_5 + w_4x_6 \\ w_1x_4 + w_2x_5 + w_3x_7 + w_4x_8 \\ w_1x_5 + w_2x_6 + w_3x_8 + w_4x_9 \end{bmatrix} \quad (32)$$

Ở đây ta có thể nhận thấy một điều ta có thể đưa chúng thành một ma trận trong số thưa có chia sẻ để dễ dàng hơn trong việc tính toán.

$$= \begin{bmatrix} w_1 & w_2 & 0 & w_3 & w_4 & 0 & 0 & 0 & 0 \\ 0 & w_1 & w_2 & 0 & w_3 & w_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_1 & w_2 & 0 & w_3 & w_4 & 0 \\ 0 & 0 & 0 & 0 & w_1 & w_2 & 0 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} = \mathbf{Cx} \tag{33}$$

Tích chập bao gồm các lớp nhân chập, lớp gộp (pooling), các hàm kích hoạt và để phân loại chúng thì ta cần thêm lớp mạng nơ-ron kết nối đầy đủ (fully connected network). Lớp tích chập ngoài phép toán giúp chúng ta hiểu về nguyên lý hoạt động của chúng. Ngoài ra còn một vài kiến thức liên quan tới nhân chập cần nhắc tới. Đầu tiên là bộ vá (padding). Nó giúp chúng ta thêm các giá trị vào biên ảnh để chúng ta được các loại ảnh đầu ra với các kích thước tương ứng: same, valid, full.



Hình 2-17. Ví dụ về bao viền(padding)

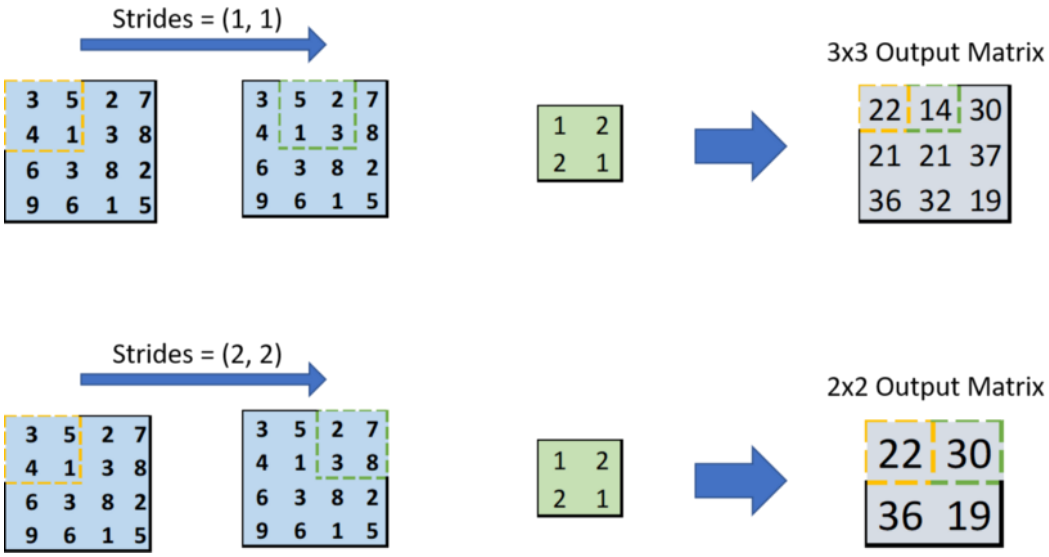
Với đầu vào có kích thước là $n * n$, *bộ lọc là* $f * f$.

Với kiểu padding valid ta không thêm các giá trị 0 xung quanh viền do đó dẫn tới kết quả sau khi nhân chập có đầu ra tương ứng là $\left(n - \frac{f}{2} + 1\right) * \left(n - \frac{f}{2} + 1\right)$.

Với kiểu padding same ta thêm các giá trị 0 ở xung quanh viền có tổng bằng $f - 1$ mỗi chiều do đó kết quả đầu ra sẽ có kích thước giống như đầu vào $n * n$

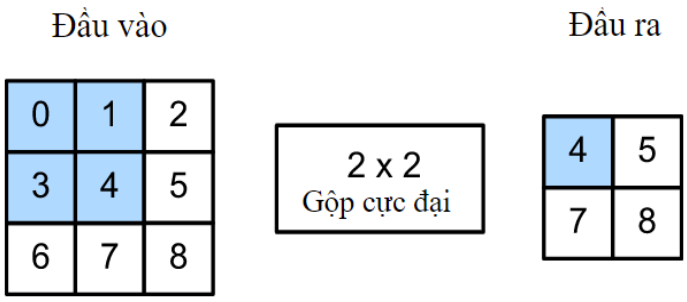
Với kiểu padding full, ta thêm các giá trị 0 ở xung quanh viền có tổng bằng $2(f-1)$ mỗi chiều do đó kết quả đầu ra tương ứng có kích thước lớn hơn đầu vào và bằng $\left(n+\frac{f}{2}+1\right)*\left(n+\frac{f}{2}+1\right)$.

Trong tích chập ta còn tiếp cận tới khái niệm bước nhảy (stride) cho chúng ta biết được tốc độ di chuyển của bộ lọc trên ma trận đầu vào. Nếu bước nhảy là (1,1) thì bộ lọc sẽ di chuyển từng bước bước một trên ma trận đầu vào. Nếu bước nhảy là (2,2) thì nó sẽ nhảy 2 bước trên mỗi hàng và cột. Điều đó đồng nghĩa với việc nếu bước nhảy càng lớn thì kích thước ma trận đầu ra càng nhỏ. Việc đặt các bước nhảy giúp chúng ta giảm các giá trị lặp lại.



Hình 2-18. Ví dụ về bước nhảy(stride)

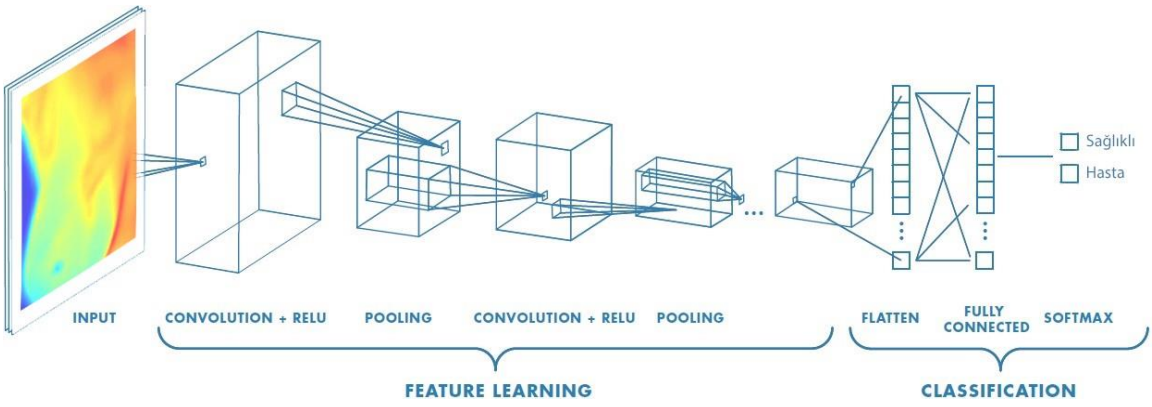
Lớp gộp thường được sử dụng sau tầng tích chập giúp giảm kích thước bản đồ đặc trưng, giảm số lượng tham số cũng như tính toán ở các lớp phía sau, qua đó tăng tính bất biến tính ổn định đối với các biến đổi nhỏ trong đầu vào. Cách gộp nó cũng khá giống cách tính các lớp tích chập, các toán tử bao gồm một cửa sổ có kích thước cố định được trượt trên tất cả các vùng đầu vào với giá trị bước nhảy nhất định dùng để tính toán giá trị đầu ra duy nhất tại mỗi vị trí mà cửa sổ đó trượt qua. Có hai phương pháp thường được sử dụng đó là phương pháp gộp cực đại (max pooling) hoặc phương pháp gộp trung bình (average pooling).



Hình 2-19. Ví dụ về lớp gộp max-pooling

Với hàm kích hoạt và lớp mạng nơ-ron đầy đủ thì đã được nhắc đến và chỉ ra ở mạng nơ-ron chi tiết.

Để hiểu rõ hơn cách hoạt động của mạng nơ ron tích chập, chúng ta cùng tìm hiểu các bước thực hiện cũng như các công thức tính toán của chúng được áp dụng đối với nghiên cứu áp dụng được sử dụng vào trong bài toán RSSI này.



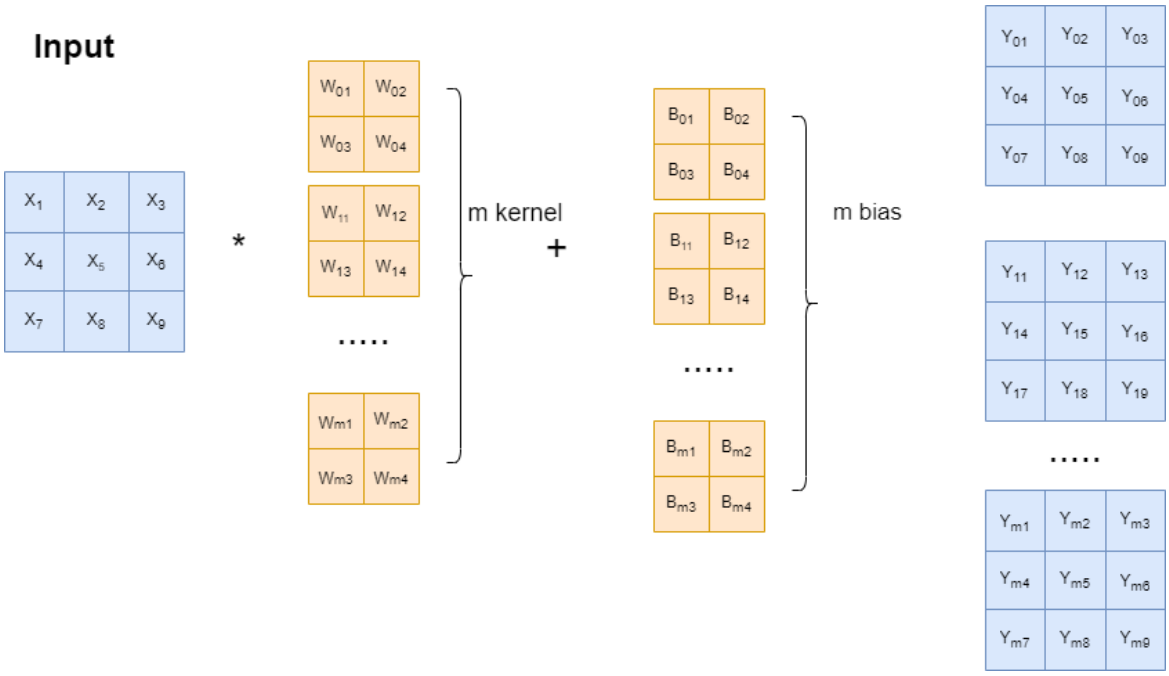
Hình 2-20. Mô hình tổng quát về mạng tích chập dùng để phân loại

Đầu tiên chúng ta cần hiểu làm thế nào để có được ma trận ảnh RSSI hai chiều như vậy. Ta có RSSI là tín hiệu một chiều, Để tạo thành ma trận 2 chiều có khá là nhiều cách khác nhau. Nếu chúng ta có nhiều iBeacon. Chúng ta có thể chuyển trực tiếp ma trận một chiều thành 2 chiều ví dụ như 256 tín hiệu RSSI tương đương với ma trận ảnh 16*16. Đối với bài toán này, ta sử dụng một cách khác để tạo ma trận ảnh lại vừa khai thác được sự không ổn định của RSSI bằng cách lấy các giá trị RSSI ở các thời điểm liên tiếp để tạo ra ma trận ảnh. Tại thời điểm t_0, ta thu được các giá trị tín hiệu: $[RSSI_{01}, RSSI_{02}, RSSI_{03}, \cdots, RSSI_{0n}]$. Nếu lấy đầu đầu vào là m các thời điểm t ta sẽ thu được ma trận RSSI [14]:

$$\mathbf{x} = \begin{bmatrix} \mathrm{RSSI}_{01} & \mathrm{RSSI}_{02} & \mathrm{RSSI}_{03} & \cdots \mathrm{RSSI}_{0n} \\ \mathrm{RSSI}_{11} & \mathrm{RSSI}_{12} & \mathrm{RSSI}_{13} & \cdots \mathrm{RSSI}_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{RSSI}_{m1} & \mathrm{RSSI}_{m2} & \mathrm{RSSI}_{m3} & \cdots \mathrm{RSSI}_{mn} \end{bmatrix} \quad (34)$$

Tuy nhiên để đảm bảo tính thời gian thực ta cần cân nhắc số lần thời điểm t.

Chúng cũng khá giống với mạng nơ-ron ta cũng đi tìm hiểu: thuật toán lan truyền tới, cách tính hàm lỗi và thuật toán lan truyền ngược để có thể tối ưu mạng tích chập. Với thuật toán lan truyền tới ta sử dụng phép nhân chập như trình bày ở trên. Thêm vào đó ta có thêm số lượng các bộ lọc, giúp tăng khả năng tìm kiếm được đặc trưng của ảnh nhưng đồng thời cũng làm tăng thêm số lượng tham số, tăng thêm độ phức tạp khi tính toán.



Hình 2-21. Nhân chập trong mạng tích chập

$$Y_i = B_i + \sum_{j=1}^{n} X_j * W_{ij}, i = 1,2, \dots, m \quad (35)$$

$$y = \begin{bmatrix} w_{i1} & w_{i2} & 0 & w_{i3} & w_{i4} & 0 & 0 & 0 & 0 \\ 0 & w_{i1} & w_{i2} & 0 & w_{i3} & w_{i4} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{i1} & w_{i2} & 0 & w_{i3} & w_{i4} & 0 \\ 0 & 0 & 0 & 0 & w_{i1} & w_{i2} & 0 & w_{i3} & w_{i4} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} + w_0 = \mathbf{C}\mathbf{x} + w_0 \quad (36)$$

Đến đây ta thấy thuật toán lan truyền tới khá giống với mạng nơ-ron. Cách tính hàm lỗi cũng sử dụng hàm cross-entropy lan truyền ngược. Với các lớp mạng nơ-ron đầy đủ, các hàm kích hoạt chúng ta đã trình bày ở trên phần mạng nơ ron. Sự khác biệt ở đây chúng ta cần biết cách tính đạo hàm của lớp tích chập, cũng như lớp maxpooling qua đó sử dụng quy tắc chuỗi để tính được đạo hàm của hàm lỗi đối với từng trọng số ở từng lớp giúp tối ưu được các trọng số.

Đạo hàm của lớp tích chập:

$$\begin{aligned} \mathbf{J_y}(\mathbf{x}) &= \mathbf{C} \\ \delta_{\mathbf{x}} &= \delta_{\mathbf{y}}\mathbf{C} \\ \mathbf{J_y}(\mathbf{w}) &= \begin{bmatrix} x_1 & x_2 & x_4 & x_5 \\ x_2 & x_3 & x_5 & x_6 \\ x_4 & x_5 & x_7 & x_8 \\ x_5 & x_6 & x_8 & x_9 \end{bmatrix} \\ \delta_{\mathbf{w}} &= \delta_{\mathbf{y}}\mathbf{J_y}(\mathbf{w}) \end{aligned} \quad (37)$$

Ta có lớp maxpooling được tính:

$$\begin{gathered} y = \max(x_1, x_2, \dots, x_n) \\ y = \sum_{i=0}^{n} w_i * x_i \text{ với } w_i = \begin{cases} 1 & \text{nếu } x_i = \max(x_1, x_2, \dots, x_n) \\ 0 & \text{trường hợp còn lại} \end{cases} \end{gathered} \quad (38)$$

Đạo hàm lớp maxpooling:

$$\frac{\partial y}{\partial \mathbf{x}_i} = w_i = \begin{cases} 1 & \text{nếu } x_i = \max(x_1, x_2, \dots, x_n) \\ 0 & \text{trường hợp còn lại} \end{cases} \quad (39)$$

Trong nghiên cứu này chúng tôi sử dụng hai lớp tích chập với kích thước bộ lọc bằng 5 nhân5. Đồng thời kèm với đó là 2 lớp maxpooling, sau đó cho qua một lớp mạng

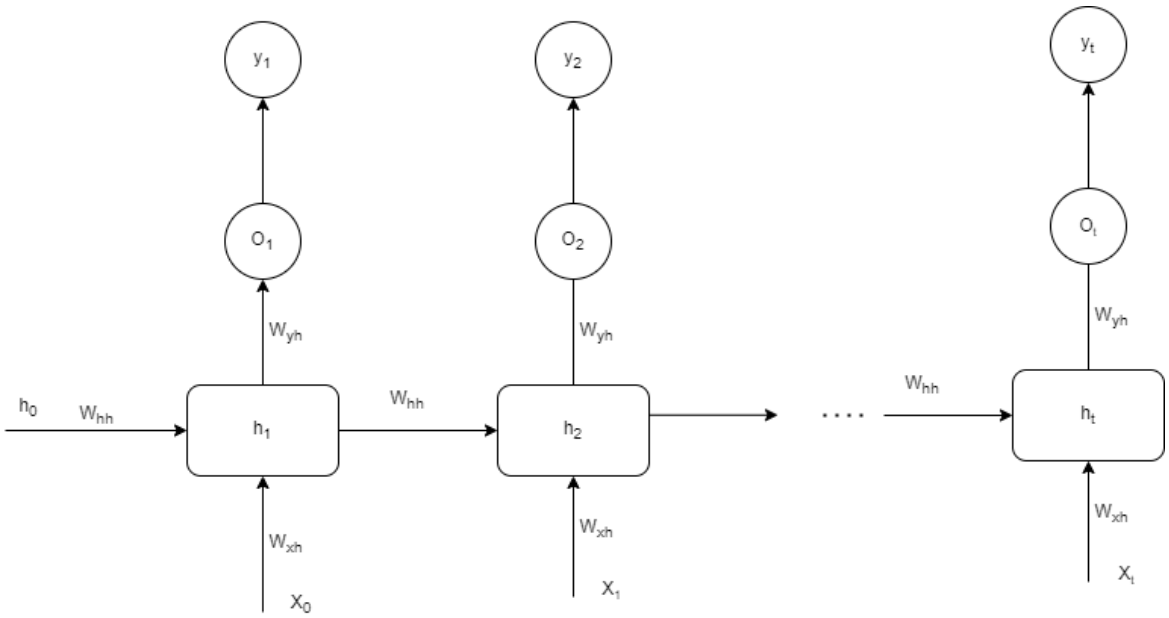
nơ-ron đầy đủ cùng với hàm kích hoạt softmax để định vị được vị trí dựa trên dữ liệu RSSI đầu vào.

2.8. Mạng nơ-ron có nhớ

2.8.1. Mạng nơ-ron hồi quy (Recurrent Neural Network-RNN)

Mạng nơ-ron hồi quy(RNN) là một dạng của mạng nơ-ron trong đó đầu ra từ bước trước được cung cấp làm đầu vào cho bước hiện tại.Mạng này phù hợp xử lý dữ liệu theo chuỗi thời gian và dữ liệu tuần tự khác. Cấu trúc RNN khá giống với ANN nhưng có thêm vòng phản hồi qua đó có thể lưu trữ ghi nhớ được trạng thái trước đó. Ý tưởng của việc sử dụng mạng RNN để xử lý bài toán định vị dựa trên cường độ tín hiệu của RSSI nhằm mục đích khai thác dựa trên mối tương quan tuần tự của RSSI đo. Vì tốc độ di chuyển của người dùng trong một môi trường trong nhà bị giới hạn, thông tin tạm thời có thể để phân biệt các vị trí có các tín hiệu RSSI tương tự nhau. Qua đó giúp cải thiện được các vị trí mơ hồ cũng như giảm thiểu sự ảnh hưởng bởi sự không ổn định của RSSI trong thời gian ngắn [16].

Mô hình RNN:

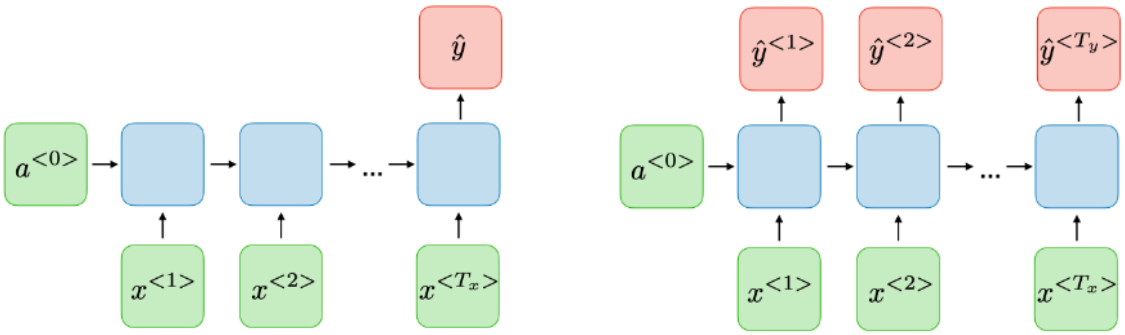


Hình 2-22. Ảnh mạng nơ-ron hồi quy

Công thức liên hệ giữa các tầng mạng RNN:

$$h_t = f_h(X_t, h_{t-1})$$
$$\hat{y}_t = f_o(h_t) \quad (40)$$

Mạng RNN được xây dựng có nhiều kiểu khác nhau. Nhưng bài toán RSSI này chúng ta tiếp cận chúng với 2 kiểu chính: nhiều về một(many-to-one), nhiều tới nhiều (many to many).



Hình 2-23. Ảnh các kiểu mạng nơ-ron hồi quy

Ta xét trường hợp nhiều tới nhiều, tổng quát:

$$h_t = f_h(X_t, h_{t-1}) = \phi_h\left(W_{xh}^T \cdot X_t + W_{hh}^T \cdot h_{t-1} + b_h\right)$$
$$\hat{y}t = f_o(h_t) = \phi_o\left(Wyh^T \cdot h_t + b_y\right) \quad (41)$$

Với W_{xh} , W_{hh}, W_{yh} là trọng số của ma trận đầu vào

ϕ_h, ϕ_o là hàm phi tuyến, các hàm phi tuyến thường được sử dụng như hàm sigmoid hay Softmax, ReLu,…Ma trận trọng số được tái sử dụng qua đó giảm số lượng tham số.

Thuật toán lan truyền tới:

Đầu ra tại thời điểm t:

$$\begin{aligned} h_t &= \tanh(X_t \cdot W_{xh} + h_{t-1} \cdot W_{hh} + b_h) \\ &= \phi_h([X_t h_{t-1}] \cdot W + b_h) \\ o_t &= h_t \cdot W_{yh} + b_y \\ \hat{y}_t &= \text{softmax}(o_t) \end{aligned} \quad (42)$$

X_t : ma trận input đầu vào $m * n_{input}$.

h_{t-1}: trạng thái lớp ẩn $m * n_{neuron}$.

W_{hh}: ma trận trọng số kết nối trọng số liên kết của 2 lớp ẩn $n_{neuron} * n_{neuron}$.

W_{xh}: ma trận trong số giữa đầu vào và lớp ẩn $n_{input} * n_{neuron}$.

W_{yh}: ma trận trọng số liên kết giữa lớp ẩn và đầu ra $n_{neuron} * n_{neuron}$.

b_h, b_y : hệ số bias của mỗi lớp ẩn và đầu ra.

y_t : là ma trận đầu ra tại thời điểm t $m * n_{neuron}$.

Thuật toán lan truyền ngược:

Giống như mạng nơ-ron hay tích chập, để tối ưu ma trận trọng số của RNN ta cần tính được hàm lỗi. Vẫn sử dụng hàm cross-entropy:

$$\begin{aligned} L(\hat{y}, y) &= \sum_{t=1}^{T} L_t(\hat{y}_t, y_t) \\ &= -\sum_{t}^{T} y_t \log \hat{y}t \\ &= -\sum t = 1^T y_t \log(\hat{y}_t) \end{aligned} \tag{43}$$

Bước tiếp theo ta cần tính đạo hàm của hàm lỗi với từng trọng số.

Với hàm kích hoạt ở lớp cuối là hàm softmax ta có:

Hàm lan truyền tới:

$$\begin{aligned} o_t &= h_t \cdot W_{yh} + b_y \\ \frac{\partial L}{\partial W_{yh}} &= \sum_{t}^{T} \frac{\partial L_t}{\partial W_{yh}} \\ &= \sum_{t}^{T} \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}t}{\partial o_t} \frac{\partial o_t}{\partial Wh} \\ &= \sum_{t}^{T} (\hat{y}_t - y_t) \otimes h_t \\ \frac{\partial L}{\partial b_y} &= \sum_{t}^{T} \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial o_t} \frac{\partial o_t}{\partial b_y} \\ &= \sum_{t}^{T} (\hat{y}_t - y_t) \end{aligned} \tag{44}$$

Đạo hàm W_{hh} của hàm lỗi qua từng lớp:

Đạo hàm của hàm lỗi 1 lớp $t+1 \to t$ để tính được W_{hh}

$$h_t = \tanh(X_t \cdot W_{xh} + h_{t-1} \cdot W_{hh} + b_h)$$
$$\frac{\partial L_{t+1}}{\partial W_{hh}} = \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}} \quad (45)$$

Đạo hàm của hàm lỗi $t+1 \to 1$ tính W_{hh}

$$\frac{\partial L_{t+1}}{\partial W_{hh}} = \sum_{k=1}^{t+1} \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial h_k} \frac{\partial h_k}{\partial W_{hh}} \quad (46)$$

Sử dụng quy tắc chuỗi với đạo hàm $\frac{\partial h_{t+1}}{\partial h_k}$ ta có:

$$\frac{\partial L_{t+1}}{\partial W_{hh}} = \sum_{k=1}^{t+1} \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \left(\prod_{j=k}^{t} \frac{\partial h_{j+1}}{\partial h_j} \right) \frac{\partial h_k}{\partial W_{hh}} \quad (47)$$

Đạo hàm của hàm lỗi với W_{xh}:

Hàm lan truyền tới:

$$h_t = \tanh(X_t \cdot W_{xh} + h_{t-1} \cdot W_{hh} + b_h) \quad (48)$$

Đạo hàm hàm lỗi 1 lớp $t+1 \to t$ để tính W_{xh}

$$\frac{\partial L_{t+1}}{\partial W_{xh}} = \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial W_{xh}}$$
$$\frac{\partial L_{t+1}}{\partial W_{xh}} = \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial W_{xh}} + \frac{\partial L_{t+1}}{\partial \hat{y}_{t+1}} \frac{\partial \hat{y}_{t+1}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial h_t} \frac{\partial h_t}{\partial W_{xh}} \quad (49)$$

Đạo hàm của hàm lỗi $t+1 \to 1$:

$$\frac{\partial L}{\partial W_{xh}} = \sum_{t}^{T} \sum_{k=1}^{t+1} \frac{\partial L_{t+1}}{\partial \hat{y}t+1} \frac{\partial \hat{y}t+1}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial h_k} \frac{\partial h_k}{\partial W_{xh}} \tag{50}$$

Bằng việc đạo hàm hàm lỗi với từng trọng số ta có thể sử dụng phương pháp xuống đồi bằng đạo hàm để tối ưu các trọng số. Thế nhưng có một vấn đề đối với RNN đó chính là biến mất đạo hàm.

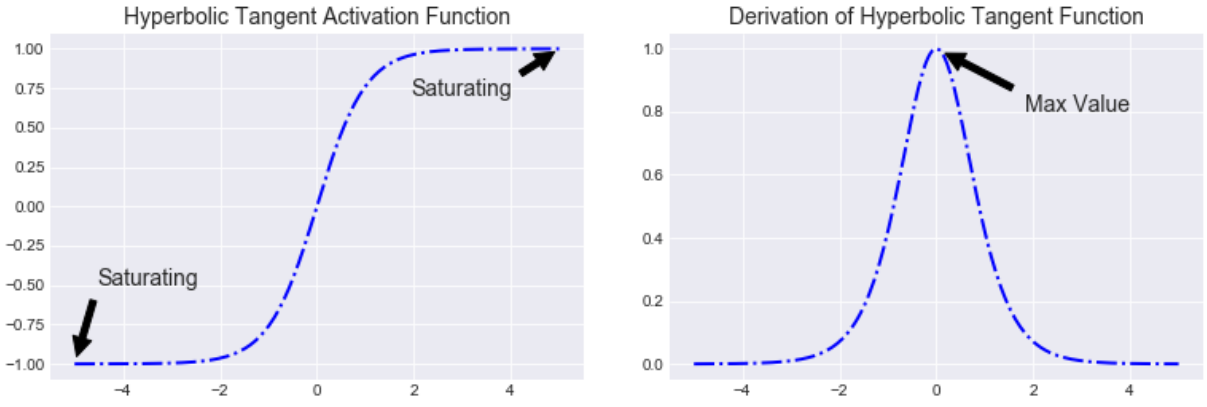
$$\prod_{j=k}^{t} \frac{\partial h_{j+1}}{\partial h_j} = \frac{\partial h_{t+1}}{\partial h_k} = \frac{\partial h_{t+1}}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}} \dots \frac{\partial h_{k+1}}{\partial h_k} \tag{51}$$

Ở đây ta sẽ chú ý tới đạo hàm của $\frac{\partial h_{t+1}}{\partial h_t}$

$$\begin{gathered} h_t = \phi_h([X_t h_{t-1}] \cdot W + b_h) \\ \frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi_h'\left(W_{xh}^T \cdot X_{j+1} + W_{hh}^T \cdot h_j + b_h\right) W_{hh}\right. \end{gathered} \tag{52}$$

Lan truyền ngược từ $t+1 \rightarrow k$ ta sẽ có:

$$\prod_{j=k}^{t} \frac{\partial h_{j+1}}{\partial h_j} = \prod_{j=k}^{t} \operatorname{diag}\left(\phi_h'\left(W_{xh}^T \cdot X_{j+1} + W_{hh}^T \cdot h_j + b_h\right) W_{hh}\right. \tag{53}$$



Hình 2-24. Các hàm kích hoạt softmax, tanh và đạo hàm

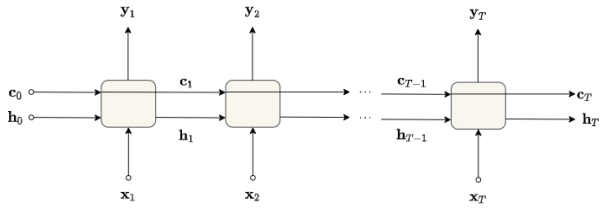
Quan sát hình trên ta thấy, vai trò của các hàm kích hoạt là sẽ chuyển đổi các giá trị đầu vào thành các giá trị trong khoảng từ (0,1). Nếu như W_{hh} lớn khiến đầu vào khá lớn (âm hoặc dương) hàm sẽ bị bão hòa tại (0,1). Dẫn tới đạo hàm sẽ rất gần với 0 dẫn

tới việc lan truyền ngược trở nên rất khó các trọng số W ở lớp sau không ảnh hưởng gì tới đầu ra. Đây là hiện tượng biến mất đạo hàm. Việc lặp đi lặp lại quá trình đạo hàm qua các hàm phi tuyến dẫn tới việc W_{hh} không được cập nhật do đạo hàm quá nhỏ. Hiện tượng này không chỉ xảy ra riêng trong RNN mà còn ở tất cả các mạng học máy khác, nhưng do RNN là một mạng sâu, nên hiện tượng này xảy ra phổ biến hơn

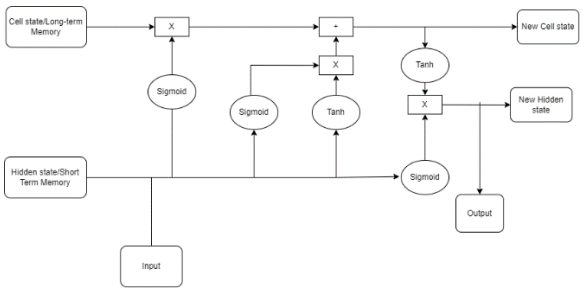
2.8.2. Mạng bộ nhớ dài-ngắn (Long-Short Term Memory - LSTM)

Mạng LSTM được thiết kế để giải quyết vấn đề biến mất đạo hàm của RNN khi ở các lớp phụ thuộc xa. Để làm được điều này LSTM có độ phức tạp cao hơn so với RNN [17], tại mỗi thời điểm t, LSTM được nhận ba thông tin khác nhau bao gồm dữ liệu đầu vào hiện tại, bộ nhớ ngắn hạn từ trạng thái ẩn trước đó, cuối cùng là bộ nhớ dài hạn.

Mô hình LSTM:

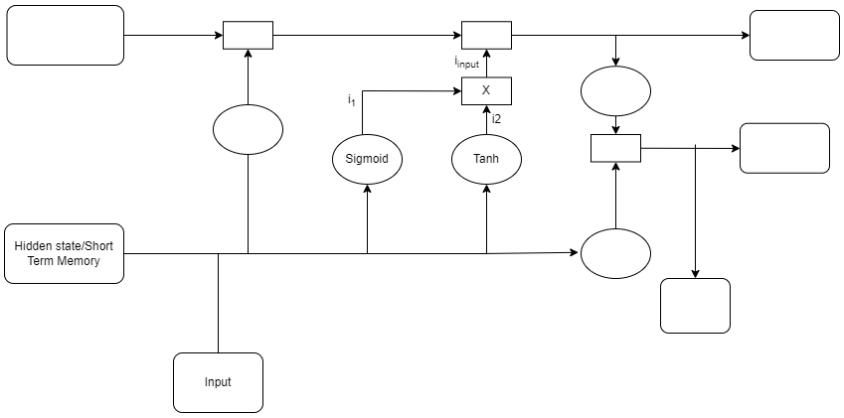


Hình 2-25. Mô hình LSTM đầy đủ



Hình 2-26. Mô hình một lớp LSTM

Cổng đầu vào:



Hình 2-27. Cổng đầu vào

Cổng đầu vào quyết định xem liệu thông tin mới có được lưu vào trong bộ nhớ dài hạn hay không. Nó có đầu vào là đầu vào hiện tại và trạng thái trước đó.

Cổng đầu vào được sử dụng bằng hai lớp. Lớp đầu tiên với việc sử dụng hàm sigmoid sẽ quyết định thông tin nào sẽ được cập nhật.

$$i_1 = \sigma\left(W_{i_1} \cdot (H_{t-1}, x_t) + \text{bias}_{i_1}\right) \quad (54)$$

Hàm Sigmoid biến các giá trị nằm trong khoảng 0-1 với 0 chỉ các phần thông tin không quan trọng sẽ bị loại bỏ và 1 chỉ thông tin quan trọng. Lớp này giúp quyết định thông tin nào sẽ được giữ lại.

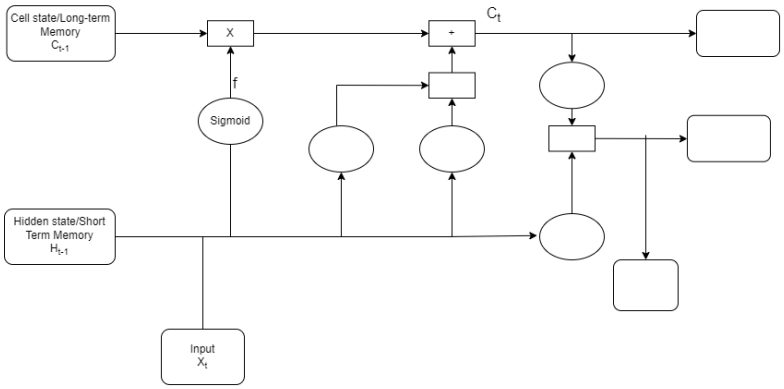
Tiếp theo là một lớp đưa thêm véc-tơ mới vào trong trạng thái.

$$i_2 = \tanh\left(W_{i_2} \cdot (H_{t-1}, x_t) + \text{bias}_{i_2}\right) \quad (55)$$

Đầu ra của 2 lớp sẽ được nhân với nhau kết quả sẽ được những thông tin cần được lưu trữ dài hạn và sử dụng:

$$i_{input} = i_1 * i_2 \quad (56)$$

Cổng quên:

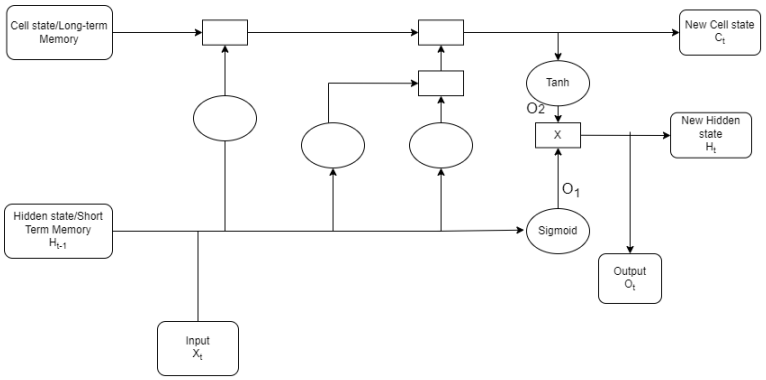


Hình 2-28. Cổng quên

Cổng quên sẽ quyết định thông tin nào của bộ nhớ dài hạn sẽ được giữ lại hoặc bỏ. Để quyết định thông tin nào sẽ được loại bỏ, giống như cổng đầu vào, ta cần có vector cổng quên, ta cho đầu vào và lớp ẩn qua hàm Sigmoid. Giá trị của hàm Sigmoid sẽ quyết định bao nhiêu phần trăm của bộ nhớ dài hạn sẽ được giữ lại.

$$\begin{aligned} f &= \sigma\left(W_{\text{forget}} \cdot (H_{t-1}, x_t) + \text{ bias forget }\right) \\ C_t &= C_{t-1} * f + i_{\text{input}} \end{aligned} \tag{57}$$

Cổng đầu ra:



Hình 2-29. Cổng đầu ra

Cổng đầu ra sẽ lấy giá trị đầu vào x hiện tại, và trạng thái nhớ lớp ẩn trước đó và bộ nhớ dài hạn mới tính toán để tạo ra lớp ẩn mới sẽ được chuyển tiếp sang lớp tiếp theo. Đầu ra của bước hiện tại có thể được rút ra từ trạng thái ẩn này. Chúng sẽ được tính theo công thức:

$$\begin{aligned} O_1 &= \sigma\left(W_{\text{output 1}} \cdot (\text{Ht} - 1, x_t) + \text{ bias }_{\text{output 1}}\right) \\ O_2 &= \tanh\left(W_{\text{output 2}} \cdot C_t + \text{ bias }_{\text{output 2}}\right) \\ H_t, O_t &= O_1 * O_2 \end{aligned} \tag{58}$$

Thuật toán lan truyền ngược cho LSTM tương tự như RNN. Thành phần chính gây ra hiện tượng biến mất đạo hàm của RNN chính là $\frac{\partial h_{t+1}}{\partial h_t}$ Tương đương với đạo hàm $\frac{\partial C_{t+1}}{\partial C_t}$ trong mạng LSTM.

$$C_t = C_{t-1} * f + i_{input}$$
$$C_t = C_{t-1} * f + i_{t-1} * i_t \quad (59)$$

$$\begin{aligned}\frac{\partial c_t}{\partial c_{t-1}} &= \frac{\partial}{\partial c_{t-1}}[c_{t-1} \cdot f_t + i_{t-1} \cdot i_t] \\ &= \frac{\partial}{\partial c_{t-1}}[c_{t-1} \cdot f_t] + \frac{\partial}{\partial c_{t-1}}[i_{t-1} \cdot i_t] \\ &= \frac{\partial f_t}{\partial c_{t-1}} \cdot c_{t-1} + \frac{\partial c_{t-1}}{\partial c_{t-1}} \cdot f_t + \frac{\partial i_t}{\partial c_{t-1}} \cdot i_{t-1} + \frac{\partial i_{t-1}}{\partial c_{t-1}} \cdot i_t \\ &= \frac{\partial f_t}{\partial c_{t-1}} \cdot c_{t-1} + f_t + \frac{\partial i_t}{\partial c_{t-1}} \cdot i_{t-1} + \frac{\partial i_{t-1}}{\partial c_{t-1}} \cdot i_t\end{aligned} \quad (60)$$

$$\begin{aligned}\frac{\partial f_t}{\partial c_{t-1}} \cdot c_{t-1} &= \frac{\partial}{\partial c_{t-1}}\left[\sigma\left(W_f \cdot [h_{t-1}, x_t]\right)\right] \cdot c_{t-1} \\ &= \sigma'\left(W_f \cdot [h_{t-1}, x_t]\right) \cdot W_f \cdot \frac{\partial h_t}{\partial c_{t-1}} \cdot c_{t-1} \\ &= \sigma'\left(W_f \cdot [h_{t-1}, x_t]\right) \cdot W_f \cdot o_{t-1} \cdot \tanh'(c_{t-1}) \cdot c_{t-1}\end{aligned} \quad (61)$$

$$\begin{aligned}\frac{\partial i_t}{\partial c_{t-1}} \cdot i_{t-1} &= \frac{\partial}{\partial ct-1}[\sigma(W_i \cdot [h_{t-1}, x_t])] \cdot i_{t-1} \\ &= \sigma'(W_c \cdot [ht-1, x_t]) \cdot W_c \cdot \frac{\partial H_t}{\partial c_{t-1}} \cdot i_{t-1} \\ &= \sigma'(W_c \cdot [ht-1, x_t]) \cdot W_c \cdot o_{t-1} \cdot \tanh'(c_{t-1}) \cdot i_{t-1} \\ \frac{\partial i_{t-1}}{\partial ct-1} \cdot i_t &= \frac{\partial}{\partial c_{t-1}}[\sigma(W_c \cdot [h_{t-1}, x_t])] \cdot i_t = \\ &= \sigma'(W_c \cdot [h_{t-1}, x_t]) \cdot W_c \cdot \frac{\partial H_t}{\partial c_{t-1}} \cdot i_t = \\ &= \sigma'(W_c \cdot [h_{t-1}, x_t]) \cdot W_c \cdot o_{t-1} \cdot \tanh'(c_{t-1}) \cdot i_t\end{aligned} \quad (62)$$

Với mạng RNN ở các lớp sâu sẽ luôn bị ảnh hưởng cuối cùng sẽ nhận giá trị trên 1 hoặc gần với 0 dẫn tới biến mất gradient, ở đây chúng ta có thể điều khiển nó bằng cổng quên f_t để đưa các giá trị đạo hàm gần với 1 hơn qua đó tránh được việc biến mất đạo hàm một cách nhanh chóng. Một điều quan trọng nữa đó là các giá trị f_t, i_t, fO_tđều

được học từ các trạng thái hiện tại. Điều này giúp đảm bảo việc không bị biến mất đạo hàm.

CHƯƠNG 3. Mô hình đề xuất, thực nghiệm và đánh giá

3.1. Phần cứng được sử dụng

Trong thực nghiệm, thiết bị được sử dụng để phát tín hiệu là ibeacon do những ưu điểm mà nó mang lại như dễ triển khai, tiêu thụ ít năng lượng và sẵn có. Để xác định được tín hiệu RSSI tại các vị trí khác nhau, tôi đã sử dụng thiết bị Raspberry Pi 3 là thiết bị đo đạc cường độ tín hiệu tại mỗi điểm được gán nhãn. Để có thể thu thập tín hiệu, tôi đã sử dụng một thư viện được viết trên ngôn ngữ Python, tôi đã cài đặt nó trên thiết bị Raspberry Pi 3. Để tăng tính linh hoạt cũng như dễ dàng theo dõi hoạt động, tôi sử dụng thêm một chiếc máy tính Dell để điều khiển từ xa thông qua SSH. Mọi dữ liệu đo đạc được đều được lưu trữ vào file csv trên Raspberry Pi 3. Sau đó được trích xuất sang thiết bị mạnh hơn phục vụ cho quá trình máy học và xây dựng các mạng.

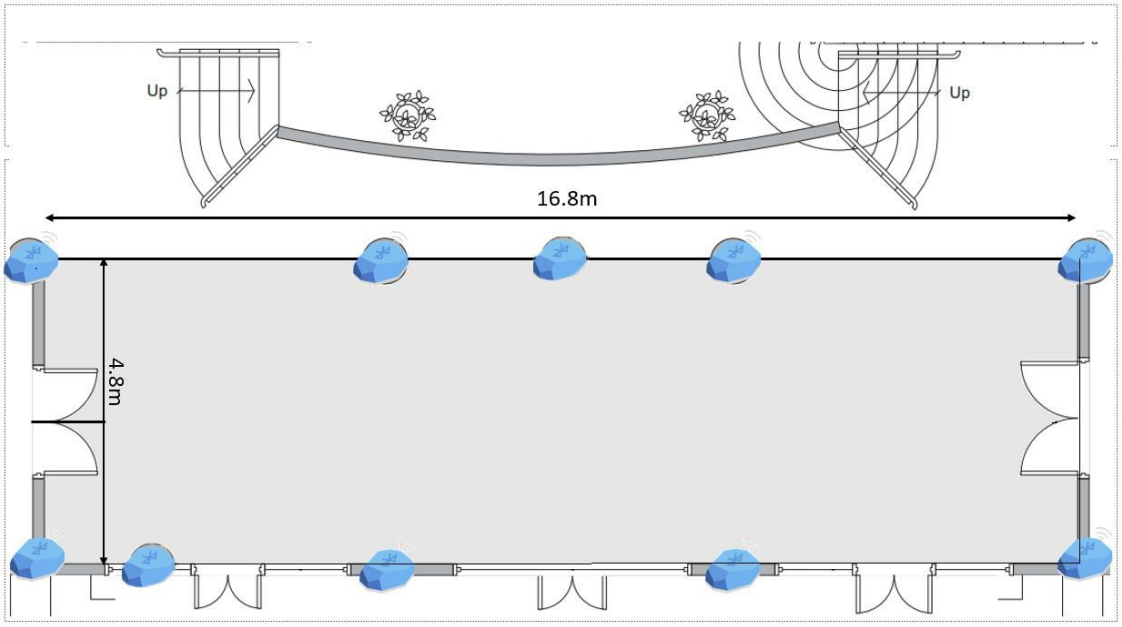
Thiết bị	Cấu hình
Raspberry Pi 3B v1.2	Xung nhịp 1.2GHz 64bit chip arm v8 Bluetooth 4.1 Python 3.7.3
Các beacons ước lượng tín hiệu (bộ thiết bị dành cho nhà phát triển)	64MHz ARM cortex-M4F BLE 5.0/2.4GHz Transmitting Power: 0dBm Transmitting cycle: 100ms
Laptop Dell	Core i7-6700HQ 2.6GHz 8.00 GB RAM Window 10 64-bit

Bảng 3-1. Mô tả thông số và thiết bị sử dụng

Hình 3-1. Ảnh iBeacon và lắp đặt

3.2. Thực nghiệm

Để kiểm tra và đánh giá mô hình được đề xuất,một phần của tầng 1 tòa nhà G2, trường đại học Công Nghệ, Đại học Quốc Gia Hà Nội được sử dụng làm khu vực đo với diện tích khoảng (3*16.8) m^2.Ở trong môi trường này tín hiệu bị thay đổi và khá nhiễu do có sự hoạt động của các bạn sinh viên, nhưng đồng thời đó cũng là cách đánh giá thực tế tính chính xác của mô hình đề xuất trong môi trường thực.



Hình 3-2. Hình ảnh sơ đồ khu vực thực nghiệm

Quá trình cài đặt thực nghiệm

Để thu thập dữ liệu, 10 iBeacon được sử dụng, chúng được đặt ở các vị trí trong khu vực ở độ cao khoảng 1.7 m. Khu vực đánh giá được chia thành 80 ô vị trí khác nhau. Quá trình đo lấy mẫu cũng như quá trình lấy mẫu để thử các vị trí của beacon không được thay đổi. Cả hai quá trình này đều được theo dõi và điều khiển thông qua mạng

LAN bằng máy tính cá nhân. Raspberry Pi 3 thu nhận các ID và tín hiệu RSSI từ các iBeacon. Đối với mỗi vị trí quá trình lấy mẫu diễn ra liên tục các tín hiệu sẽ được lần lượt ghi vào mảng các tín hiệu sẽ lấy liên tiếp, trong một lần đo nếu tín hiệu đó bị lặp lại thì lấy tín hiệu sau nhằm đảm bảo tính thời gian thực. Số lần lấy mẫu trên mỗi một vị trí là 600 lần.

Position La	Beacon1	RSSI1	Beacon2	RSSI2	Beacon3	RSSI3	Beacon4	RSSI4	Beacon5	RSSI5	Beacon6	RSSI6	Beacon7	RSSI7	Beacon8	RSSI8	Beacon9	RSSI9	Beacon10	RSSI10
0	20	-63	30	-69	60	-74	70	-74	80	-77	90	-71	130	-72	140	-64	150	-62	160	-70
0	20	-60	30	-64	60	-75	70	-76	80	-66	90	-71	130	-75	140	-64	150	-70	160	-70
0	20	-60	30	-63	60	-73	70	-70	80	-75	90	-70	130	-76	140	-67	150	-68	160	-69
0	20	-64	30	-68	60	-71	70	-72	80	-77	90	-74	130	-76	140	-61	150	-69	160	-73
0	20	-60	30	-64	60	-74	70	-75	80	-71	90	-77	130	-70	140	-63	150	-69	160	-75
0	20	-60	30	-70	60	-71	70	-77	80	-67	90	-70	130	-69	140	-66	150	-61	160	-71
0	20	-61	30	-66	60	-73	70	-80	80	-71	90	-79	130	-73	140	-67	150	-68	160	-75
0	20	-62	30	-63	60	-73	70	-76	80	-80	90	-78	130	-69	140	-63	150	-70	160	-72
0	20	-64	30	-64	60	-71	70	-75	80	-79	90	-70	130	-70	140	-62	150	-69	160	-73
0	20	-64	30	-68	60	-72	70	-81	80	-78	90	-76	130	-69	140	-60	150	-69	160	-71
0	20	-64	30	-68	60	-74	70	-70	80	-72	90	-75	130	-69	140	-61	150	-67	160	-72
0	20	-63	30	-67	60	-72	70	-77	80	-72	90	-75	130	-69	140	-60	150	-62	160	-76
0	20	-61	30	-69	60	-74	70	-75	80	-73	90	-76	130	-69	140	-60	150	-62	160	-71
0	20	-62	30	-66	60	-73	70	-72	80	-71	90	-80	130	-70	140	-67	150	-61	160	-76
0	20	-60	30	-64	60	-73	70	-70	80	-79	90	-78	130	-73	140	-60	150	-61	160	-76
0	20	-62	30	-69	60	-75	70	-74	80	-66	90	-79	130	-69	140	-63	150	-60	160	-74
0	20	-62	30	-65	60	-73	70	-69	80	-78	90	-74	130	-71	140	-60	150	-69	160	-71
0	20	-61	30	-69	60	-75	70	-86	80	-70	90	-71	130	-73	140	-68	150	-67	160	-71
0	20	-62	30	-69	60	-71	70	-70	80	-68	90	-73	130	-68	140	-68	150	-66	160	-71

Hình 3-3. Ví dụ tập dữ liệu thu thập

Mỗi dòng trên có thông tin của nhãn được đánh vị trí và các giá trị RSSI cũng như ID của các ibeacon. Các giá trị ID của beacon sẽ là thông tin để sắp xếp thứ tự các giá trị RSSI được đưa vào các mạng học sâu phục vụ cho quá trình học, giá trị cột position chính là đầu ra mong muốn của các mô hình. Thêm vào đó là tọa độ của các vị trí trên thực thế đối với mỗi điểm đặt.

Position Le	Beacon1	RSSI1	Beacon2	RSSI2	Beacon3	RSSI3	Beacon4	RSSI4	Beacon5	RSSI5	Beacon6	RSSI6	Beacon7	RSSI7	Beacon8	RSSI8	Beacon9	RSSI9	Beacon10	RSSI10	PositionX	PositionY
0	20	-58	30	-71	60	-71	70	-78	80	-68	90	-77	130	-69	140	-67	150	-61	160	-66	0	0
0	20	-62	30	-71	60	-71	70	-73	80	-78	90	-85	130	-77	140	-61	150	-60	160	-69	0	0
0	20	-64	30	-67	60	-78	70	-79	80	-68	90	-79	130	-69	140	-66	150	-77	160	-69	0	0
0	20	-59	30	-70	60	-79	70	-72	80	-69	90	-71	130	-71	140	-64	150	-65	160	-68	0	0
0	20	-58	30	-62	60	-69	70	-72	80	-70	90	-79	130	-80	140	-62	150	-77	160	-68	0	0
0	20	-61	30	-67	60	-73	70	-73	80	-80	90	-77	130	-69	140	-63	150	-78	160	-69	0	0
0	20	-62	30	-72	60	-78	70	-73	80	-81	90	-72	130	-71	140	-65	150	-78	160	-68	0	0
0	20	-59	30	-66	60	-74	70	-70	80	-68	90	-83	130	-83	140	-65	150	-64	160	-69	0	0
0	20	-65	30	-66	60	-69	70	-71	80	-70	90	-85	130	-67	140	-62	150	-76	160	-71	0	0
0	20	-62	30	-69	60	-70	70	-77	80	-68	90	-72	130	-76	140	-65	150	-60	160	-67	0	0
0	20	-64	30	-67	60	-71	70	-72	80	-68	90	-79	130	-83	140	-67	150	-60	160	-68	0	0
0	20	-58	30	-66	60	-70	70	-72	80	-71	90	-87	130	-70	140	-60	150	-77	160	-67	0	0
0	20	-61	30	-67	60	-70	70	-72	80	-74	90	-78	130	-72	140	-65	150	-77	160	-70	0	0
0	20	-62	30	-62	60	-69	70	-72	80	-67	90	-80	130	-70	140	-62	150	-64	160	-65	0	0
0	20	-65	30	-70	60	-74	70	-80	80	-79	90	-73	130	-72	140	-59	150	-60	160	-67	0	0
0	20	-62	30	-62	60	-74	70	-71	80	-68	90	-97	130	-70	140	-65	150	-64	160	-69	0	0
0	20	-65	30	-62	60	-70	70	-70	80	-67	90	-73	130	-70	140	-64	150	-64	160	-67	0	0
0	20	-58	30	-70	60	-74	70	-77	80	-78	90	-72	130	-70	140	-65	150	-59	160	-65	0	0
0	20	-57	30	-71	60	-69	70	-71	80	-70	90	-72	130	-70	140	-65	150	-60	160	-65	0	0
0	20	-58	30	-66	60	-71	70	-72	80	-69	90	-72	130	-69	140	-64	150	-64	160	-68	0	0
0	20	-64	30	-66	60	-73	70	-77	80	-69	90	-80	130	-68	140	-65	150	-63	160	-69	0	0
0	20	-58	30	-70	60	-79	70	-70	80	-68	90	-80	130	-71	140	-66	150	-64	160	-65	0	0
0	20	-65	30	-62	60	-73	70	-71	80	-78	90	-79	130	-80	140	-61	150	-65	160	-68	0	0
0	20	-58	30	-70	60	-80	70	-70	80	-68	90	-79	130	-83	140	-66	150	-65	160	-69	0	0
0	20	-66	30	-71	60	-81	70	-73	80	-68	90	-79	130	-83	140	-66	150	-61	160	-67	0	0
0	20	-59	30	-67	60	-80	70	-73	80	-77	90	-79	130	-68	140	-65	150	-77	160	-69	0	0
0	20	-65	30	-68	60	-74	70	-79	80	-78	90	-80	130	-69	140	-62	150	-60	160	-68	0	0
0	20	-59	30	-70	60	-71	70	-72	80	-69	90	-72	130	-82	140	-64	150	-64	160	-66	0	0
0	20	-61	30	-63	60	-82	70	-70	80	-70	90	-71	130	-70	140	-67	150	-61	160	-70	0	0
0	20	-61	30	-68	60	-74	70	-72	80	-69	90	-97	130	-86	140	-61	150	-65	160	-68	0	0
0	20	-59	30	-71	60	-73	70	-79	80	-69	90	-78	130	-68	140	-60	150	-64	160	-67	0	0.2
0	20	-66	30	-63	60	-74	70	-71	80	-78	90	-72	130	-83	140	-61	150	-79	160	-67	0	0.2
0	20	-65	30	-61	60	-71	70	-71	80	-77	90	-78	130	-81	140	-65	150	-61	160	-68	0	0.2
0	20	-58	30	-69	60	-75	70	-72	80	-68	90	-92	130	-81	140	-63	150	-78	160	-69	0	0.2
0	20	-61	30	-66	60	-80	70	-72	80	-77	90	-72	130	-79	140	-60	150	-66	160	-66	0	0.2
0	20	-61	30	-63	60	-73	70	-78	80	-67	90	-83	130	-83	140	-62	150	-65	160	-66	0	0.2
0	20	-61	30	-69	60	-70	70	-72	80	-68	90	-82	130	-83	140	-65	150	-65	160	-67	0	0.2
0	20	-58	30	-62	60	-73	70	-71	80	-68	90	-73	130	-71	140	-64	150	-78	160	-65	0	0.2
0	20	-62	30	-69	60	-73	70	-70	80	-68	90	-77	130	-70	140	-64	150	-77	160	-70	0	0.2
0	20	-64	30	-67	60	-73	70	-70	80	-73	90	-71	130	-75	140	-66	150	-60	160	-69	0	0.2
0	20	-61	30	-67	60	-86	70	-70	80	-74	90	-75	130	-69	140	-64	150	-60	160	-67	0	0.2
0	20	-58	30	-72	60	-74	70	-77	80	-69	90	-86	130	-72	140	-64	150	-76	160	-66	0	0.2
0	20	-65	30	-64	60	-87	70	-74	80	-74	90	-73	130	-71	140	-64	150	-64	160	-67	0	0.2
0	20	-64	30	-71	60	-73	70	-71	80	-67	90	-83	130	-76	140	-64	150	-79	160	-65	0	0.2
0	20	-65	30	-70	60	-73	70	-77	80	-73	90	-79	130	-69	140	-64	150	-74	160	-70	0	0.2
0	20	-57	30	-62	60	-73	70	-72	80	-73	90	-80	130	-82	140	-61	150	-74	160	-67	0	0.2

Hình 3-4. Ví dụ dữ liệu thu thập dùng để kiểm tra mô hình

Đối với quá trình kiểm thử, việc lấy mẫu kiểm thử tính toán được thực hiện lấy xung quanh ngẫu nhiên trong phạm vi các ô vị trí. Các tín hiệu RSSI này sau khi được đo đánh nhãn được đưa vào mô hình nhằm đánh giá mức độ chính xác của mô hình

Đối với quá trình kiểm thử, việc lấy mẫu kiểm thử tính toán được thực hiện lấy xung quanh ngẫu nhiên trong phạm vi các ô vị trí. Các giá trịRSSI này sau khi được đo đánh nhãn được đưa vào mô hình nhằm đánh giá mức độ chính xác của mô hình.

3.3. Mô hình sử dụng

Trong nghiên cứu này, tôi sử dụng các mạng kNN, mạng nơ-ron, mạng tích chập và mạng LSTM với mục đích nhằm so sánh độ chính xác của các mô hình học máy cũng như đánh giá ưu nhược điểm của các mô hình trong việc định vị trong nhà dựa trên RSSI. Các mô hình sử dụng RSSI từ thiết bị thu thập tín hiệu BLE của iBeacon, ở đây nó chính là Raspberry Pi. Với mục đích đánh giá các mô hình học máy ta sử dụng chính dữ liệu thô để đánh giá tính chính xác của bài toán đối với mỗi mô hình học máy.

Để đánh giá được các mô hình học máy đầu tiên chúng ta cùng nhìn qua kiến trúc của các mô hình học máy được sử dụng ở đây.

- Mạng kNN: sử dụng vòng lặp tính toán ra khoảng cách Ecludian của các điểm đối với điểm đang xét, sau đó tìm ra *k* điểm gần nhất. Xác định vị trí của nó là vị trí mà có số lần xuất hiện nhiều nhất trong k điểm đã xét [10]
- Với mạng nơ-ron: chúng tôi sử dụng 2 lớp với đầu vào là 10 nút tương ứng với 10 giá trị RSSI các lớp ẩn lần lượt có 1024 và 512 nơ-ron lớp phân loại gồm 80 lớp [11].
- Mạng LSTM: sử dụng 3 lớp LSTM với đầu vào là các giá trị đo lần lượt tại thời điểm *t-2, t-1, t.* Sau đó cho đi qua một lớp mạng nơ-ron đầy đủ qua đó có thể xác định được vị trí hiện tại là vị trí có xác xuất lớn nhất. Thời gian để đo 3 lần liên tiếp mất khoảng 0.4 giây.
- Mạng tích chập sử dụng dữ liệu đo 10 lần liên tiếp qua hai lớp tích chập có bộ lọc bằng (5,5). Sau đó cho qua lớp mạng nơ-ron đầy đủ để phân loại các lớp. Thời gian để đo 10 lần liên tiếp mất khoảng 1.5 giây.

Sau quá trình cho mạng ANN, CNN, LSTM học, ta sẽ có được mạng lưới các trọng số được tối ưu, sau đó các giá trị thu thập được chỉ cần cho qua mạng ta sẽ có được vị trí hiện tại của đối tượng sẽ nằm trên các vị trí mà chúng ta đánh nhãn từ trước.

3.4. Đánh giá

```
accuracy = accuracy_score(Y_test, predictions) * 100
print("Thời gian dự đoán 1 dữ liệu:"+str(delta))
print('Accuracy: ' + repr(accuracy) + '%')
```

```
Thời gian dự đoán 1 dữ liệu:0.41936612129211426
Accuracy: 85.24691358024693%
```

Hình 3-5. Độ chính xác của KNN và thời gian để thực hiện

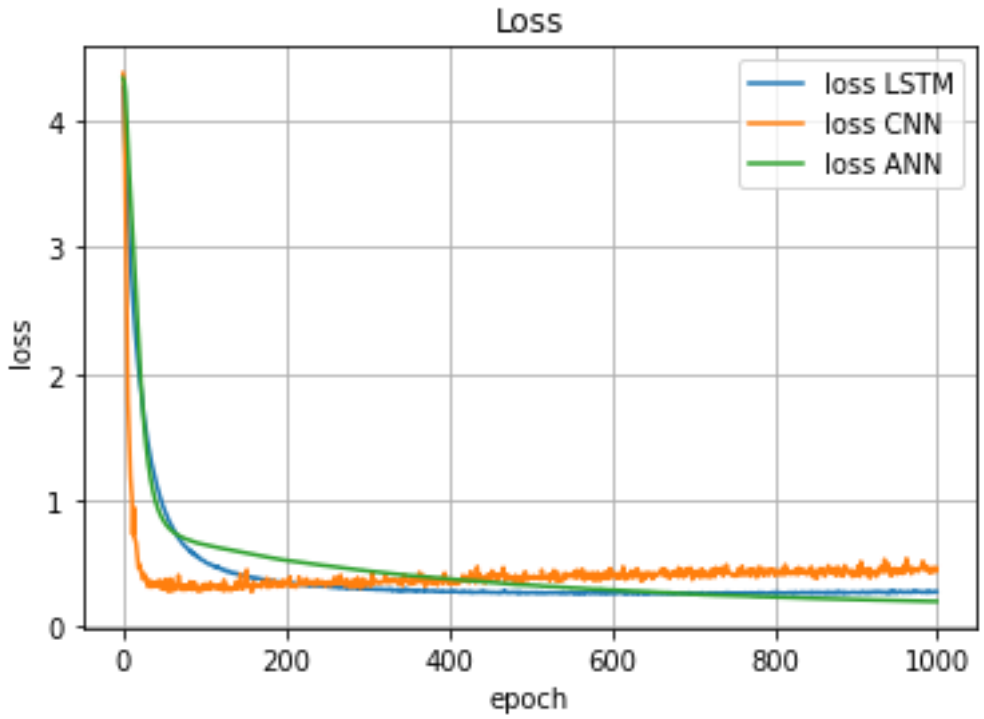
Hình trên mô tả độ chính xác cũng như thời gian để nó thực hiện một lần tính toán và độ chính xác của mô hình kNN. Đây được đánh giá là một thuật toán lười, khi đó quá trình học không diễn ra ở trên tập dữ liệu, việc tính toán chỉ thực hiện khi có các điểm dữ liệu cần xác định vị trí. Do đó mặc dù độ chính xác khá cao 85.24% nhưng tiêu tốn nhiều thời gian để tính toán.
Ưu điểm của KNN:

- Độ phức tạp tính toán của quá trình training là bằng 0.
- Việc dự đoán kết quả của dữ liệu mới rất đơn giản.

- Không cần giả sử gì về phân phối của các class.

Nhược điểm của KNN

- *k*NN rất nhạy cảm với nhiễu khi *k* nhỏ.
- *k*NN là một thuật toán mà mọi tính toán đều nằm ở khâu kiểm tra. Trong đó việc tính khoảng cách tới *từng* điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với *k* càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của *k*NN.

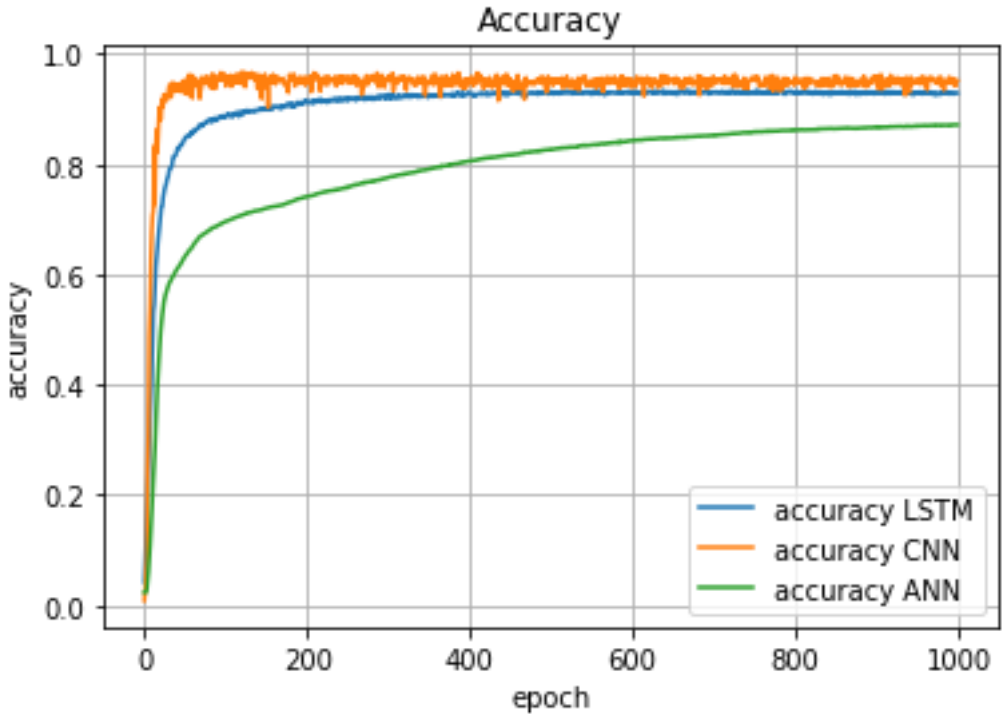


Hình 3-6. So sánh hàm lỗi của các mạng học sâu

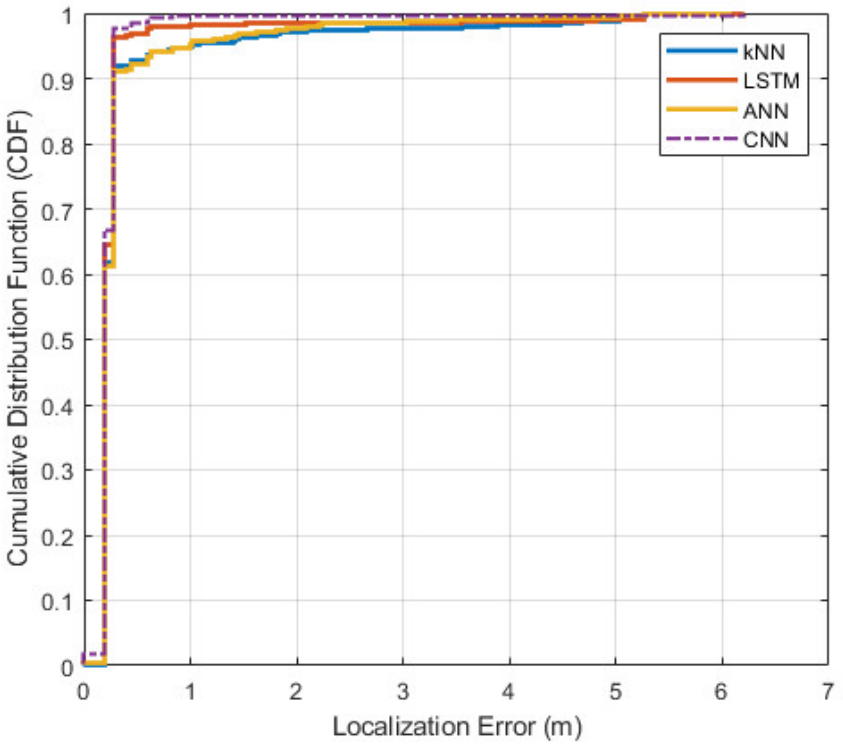
Hình 3.5 đánh giá quá trình học của các mô hình học máy với bộ dữ liệu có 48600 mẫu. Ban đầu các giá trị khởi tạo thường được tạo theo hàm phân phối chuẩn nên độ chính xác thường thấp. Thông qua việc lặp đi lặp lại quá trình học, độ chính xác tăng dần, hàm lỗi giảm dần, đến một mức nào đó sẽ đạt tới mức bão hòa, khi đó dù có tăng số lần học thì độ chính xác cũng không có sự thay đổi. Ở đây ta thấy một điều mạng tích chập có giá trị lỗi cao nhất sau đó đến LSTM và ANN. Điều này cũng phản ánh đúng việc mạng tích chập là mạng chia sẻ trọng số nên hàm lỗi sẽ cao hơn hai mạng còn lại. Với LSTM mặc dù việc sử dụng thêm cổng quên đã giúp cho làm giảm hiện tượng biến

mất đạo hàm nhưng khi đến một mức nhất định hàm lỗi đủ nhỏ việc biến mất này vẫn xảy ra.

Hình 3.6 đánh giá độ chính xác của các mô hình học máy cao nhất là mạng nơ-ron tích chập 95.5%, sau đó đến LSTM 93.5%và cuối cùng là mạng nơ-ron 90.08%. Điều này có được là do mạng LSTM đã khai thác việc không ổn định của tín hiệu RSSI. Chúng sử dụng thêm giá trị đầu vào là các tín hiệu RSSI đo ở các thời gian trước đó gần nhất để tìm ra vị trí chính xác nhất.

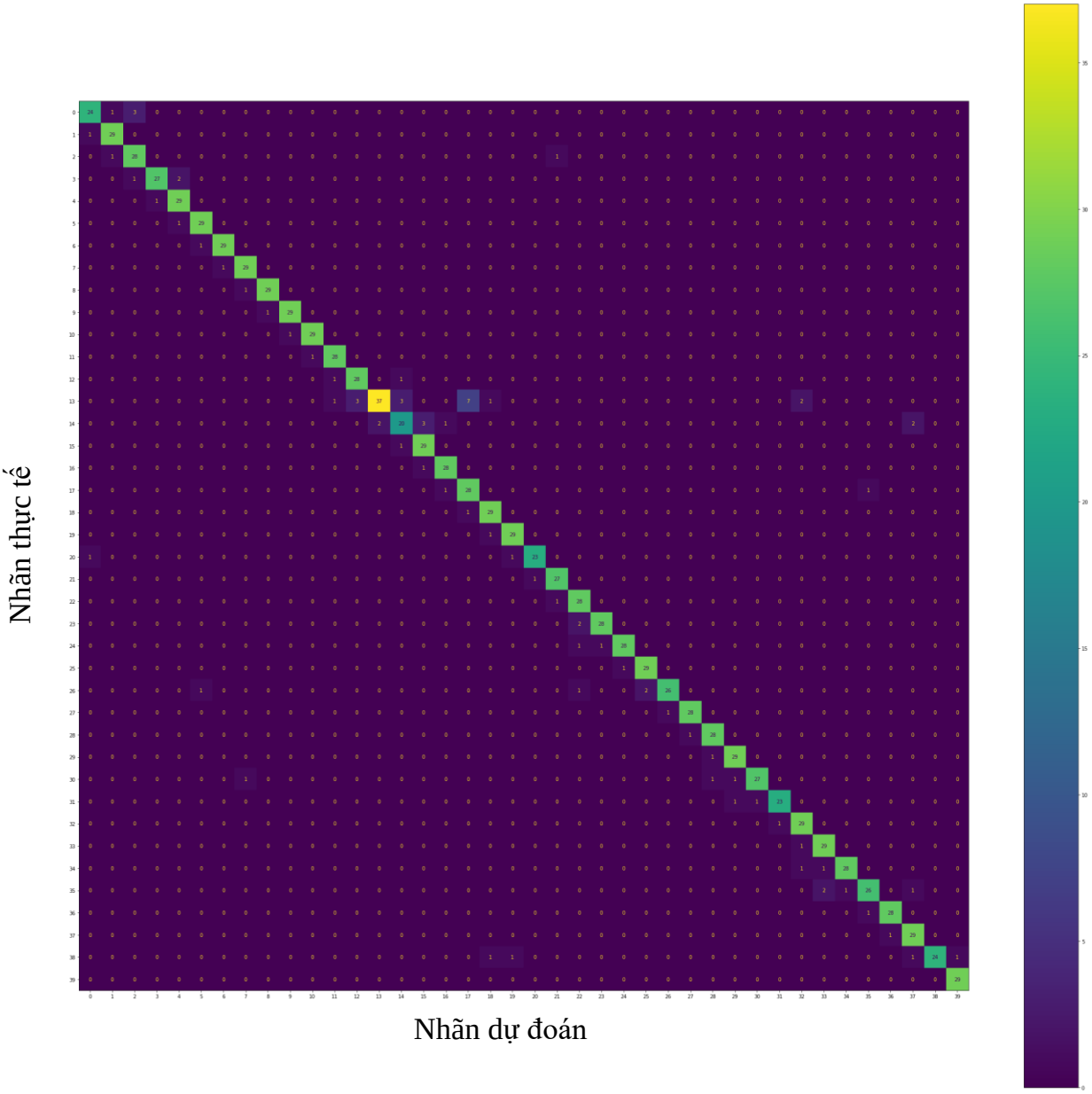


Hình 3-7. So sánh độ chính xác của các mạng học sâu

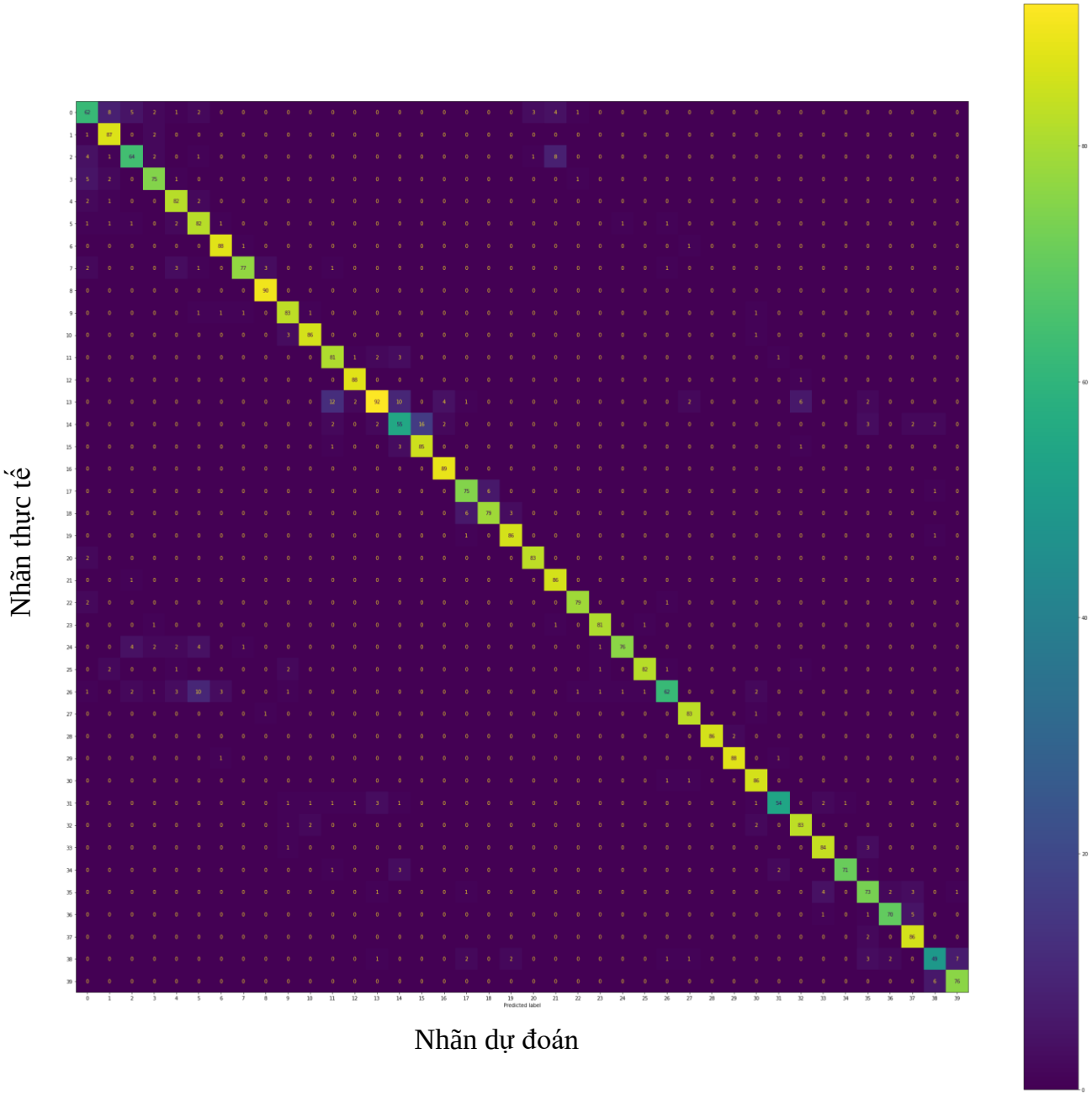


Hình 3-8. Hàm phân phối tích lũy CDF

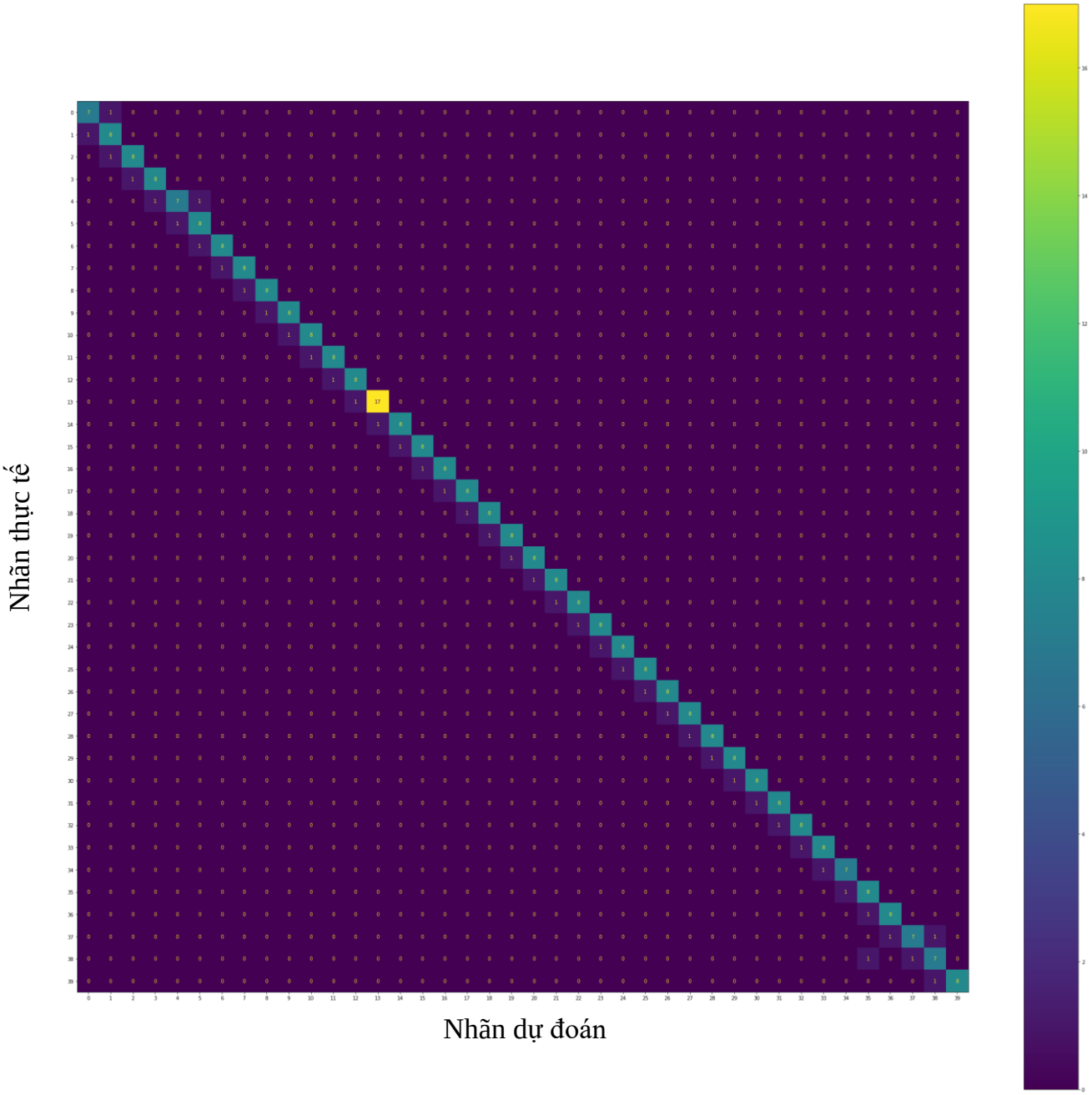
Hình 3.7 mô tả kết quả đạt được: Quá trình kiểm tra toàn hệ thống lỗi trung bình xấp xỉ 0.28m. Quan sát hàm phân phối tích lũy ta thấy 90% lỗi có khoảng cách nhỏ hơn 0.28m đây là khoảng cách xa nhất từ điểm đặt vị trí kiểm tra đến vị trí được cho vào mô hình học. Có thể thấy ở đây với khoảng cách nhỏ hơn 0.8m thì CNN cho độ chính xác xấp xỉ bằng 1. Khoảng cách này là khoảng cách giữa hai vị trí ô đặt. Với lỗi khoảng cách nhỏ hơn 3m thì tất cả các mô hình đều cho thấy độ chính xác, tin cậy lên tới 98%. Có được kết quả này là do các mô hình đã đạt độ chính xác khá cao trên 85% do đó dẫn tới sai số khoảng cách phần lớn nằm trong phạm vi của ô 0.4*0.4 thêm vào đó là điểm đặt trong quá trình học chúng tôi để ở chính giữa các ô dẫn tới tối ưu được khoảng cách lỗi.



Hình 3-9. Ma trận nhầm lẫn của LSTM



Hình 3-10. Ma trận nhầm lẫn của ANN



Hình 3-11. Ma trận nhầm lẫn của CNN

Hình 3.8,3.9,3.10 là ma trận nhầm lẫn của các mô hình học sâu, chúng ta có thể thấy ở đây phần lớn các giá trị nhầm lẫn đều nằm ở các ô có gần nó điều này cũng góp phần khẳng định nguyên nhân tại sao các mô hình có sai số khoảng cách nhỏ như vậy.

CHƯƠNG 4. Kết luận và định hướng phát triển

4.1. Các kết quả đạt được

Trong đồ án này tôi đã xây dựng các hệ thống định vị trong nhà với tín hiệu BLE với việc sử dụng 10 iBeacon kết hợp cùng với Raspberpy Pi 3. Các kết quả đạt được như sau:

- Nghiên cứu, khảo sát, phân tích được các vấn đề liên quan tới RSSI, tìm ra giải pháp khắc phục bằng các thuật toán học sâu như CNN, LSTM.
- Xây dựng các mạng học sâu và đánh giá nhận xét độ chính xác của từng mạng KNN 85.24%, ANN - 90.8%, LSTM 93.5%, CNN 95.5% sai số khoảng cách 0.28 m
- Thực nghiệm đánh giá độ tin cậy của các mô hình
- Giải thích được nguyên nhân tại sao có sự khác biệt độ chính xác của từng mạng.

4.2. Định hướng phát triển

Bên cạnh những kết quả đạt được, nghiên cứu trong đồ án còn có thể phát triển tiếp theo các định hướng sau:

- Sử dụng thêm thông tin dữ liệu cảm biến như cảm biến chuyển động gia tốc trên thiết bị điện thoại để tối ưu hóa xác định vị trí
- Nâng cao hiệu suất tính toán, tối ưu mô hình.
- Kết hợp cùng công thức toán tính toán vị trí tọa độ để giảm tối đa sai số khoảng cách.

Tài liệu tham khảo

[1]. Various authors, "802-2014 - IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture".

[2]. Gomez, Carles, et al. "*Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology*". Sensors (Basel, Switzerland) vol. 12,9 11734–11753. 29 Aug. 2012.

[3]. R. Zekavat and R. M. Buehrer, "*Impact of anchor placement and anchor selection on localization accuracy*" in Handbook Position Location.2019, pp. 435–465.

[4]. R. Zekavat and R. M. Buehrer, "*Source localization: algorithms and analysis*" in Handbook Position Location. 2019, pp. 59-106.

[5]. F. Della Rosa, M. Pelosi, and J. Nurmi, "*Human-induced effects on RSS ranging measurements for cooperative positioning*", Int. J. of Navigation and Observation, Vol.2012, 2012.

[6]. Apple, *"Getting Started with iBeacon – Version 1.0",* June 2014.

[7]. Various Authors, *"Handbook of Position Location, Theory, Practice, and Advances",* IEEE Press, John Wiley & Sons, Inc, 2012.

[8]. D. Torstensson, *"Indoor Positioning System Using Bluetooth Beacon Technology",* Academy of Innovation, Design and Engineering, Malardalen University, 2016.

[9]. Maciej Ciezkowski*,"Triangulation positioning system based on a static IR beacon-receiver system",* in International Conference on Methods and Models in Automation and Robotics, 2017.

[10]. Jooyoung Kim, Myungin Ji, Ju-il Jeon, Sangjoon Park, Youngsu Cho, *"K-NN based Positioning Performance Estimation for Fingerprinting Localization"*, Eighth International Conference on Ubiquitous and Future Networks (ICUFN),11 August 2016.

[11]. Apiruk Puckdeevongs,*"Indoor Localization using RSSI and Artificial Neural Network"* 9th International Electrical Engineering Congress (iEECON), Pattaya, Thailand, March 2021.

[12]. Andy Thomas, *"An Introduction to Neural Network for beginners"*, 2018.

[13]. H. Kinsley, D. Kukiela, *"Neural Network from scratch in Python",* 2020.

[14]. Mai Ibrahim, Marwan Torki and Mustafa ElNainay, *"CNN based Indoor Localization using RSS Time-Series",* IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil,28 June 2018.

[15]. Jianxin Wu, *"Introduction to Convolutional Neural Networks",* May 2017.

[16]. Minh Tu Hoang, Brosnan Yuen, Xiaodai Dong, Tao Lu, Robert Westendorp, and Kishore Reddy, *"Recurrent Neural Networks for Accurate RSSI Indoor Localization",* IEEE Internet of Things Journal, December 2019.

[17]. Gang Chen, *"A Gentle Tutorial of Recurrent Neural Network with Error",* Jan 14 2018.

[18] Fengjun Shang, Wen Su and Qiang Fu*,"A Location Estimation Algorithm Based on RSSI Vector Similarity Degree"*, International Journal of Distributed Sensor Networks,August 2014.

[19] Carter N Brown, *"Gradient for RNN",* Jun 6 2017.