BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

PHÁT HIỆN THƯ RÁC BẰNG PHƯƠNG PHÁP HỌC SÂU TRONG PYTHON

(DETECTING SPAM EMAILS USING DEEP LEARING IN PYTHON)

Sinh viên: Hà Vũ Khang
Mã số: B2111846
Khóa: K47

Cần Thơ, 11/2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

PHÁT HIỆN THƯ RÁC BẰNG PHƯƠNG PHÁP HỌC SÂU TRONG PYTHON

(DETECTING SPAM EMAILS USING DEEP LEARNING IN PYTHON)

Người hướng dẫn
ThS/TS. Lâm Nhựt Khang

Sinh viên thực hiện
Hà Vũ Khang
Mã số: B2111846
Khóa: K47

Cần Thơ, 11/2024

MỤC LỤC

MỤC LỤC....3
CHƯƠNG 1. GIỚI THIỆU....7
1. Đặt vấn đề....7
2. Các nghiên cứu liên quan....8
3. Mục tiêu....9
4. Đối tượng và phạm vi nghiên cứu....10
4.1. Đối tượng nghiên cứu....10
4.2. Phạm vi nghiên cứu....10
5. Phương pháp nghiên cứu....11
5.1. Thu thập dữ liệu (Data Colection)....11
5.2. Tiền xử lý dữ liệu (Data Preprocessing)....11
A. Làm sạch dữ liệu (Data Cleaning)....12
B. Loại bỏ từ dừng (Stopword Removal)....12
C. Phân loại từ (Tokenization)....12
D. Bổ đề ngôn ngữ (Lemmatization)....12
5.3. Trích xuất đặc trưng (Features Extraction)....12
5.4. Huấn luyện mô hình (Model Training)....13
5.4.1. Convolutional Neural Network (CNN)....13
5.4.2. Recurrent Neural Network (RNN)....13
5.4.3. Long Short-term Memory (LSTM)....13
5.4.4. Feedforward Neural Network (FNN)....13
5.5. Đánh giá mô hình (Model Evaluation)....14
6. Bố cục của bài niên luận cơ sở....14
CHƯƠNG 2. Cơ sở lý thuyết....15
1. Thư rác....15
2. Máy học, học sâu và xử lý ngôn ngữ tự nhiên....15
3. Mô hình....16
4. Mô hình mạng nơ-ron nhân tạo....16
4.1. Nơ-ron sinh học....16
4.2. Nơ-ron nhân tạo....17

4.2.1. Các thành phần cơ bản của một mô hình nơ-ron nhân tạo 18
4.3. Mạng nơ-ron sâu (Deep Neural Network – DNN).....20
4.3.1. Cấu trúc.....20
4.3.2. Nguyên lý hoạt động.....21
4.3.3. Hàm kích hoạt (Activation Functions).....22
4.4. Mạng nơ-ron tích chập (Convolutional Neural Network – CNN) 22
4.4.1. Cấu trúc.....22
4.4.2. Nguyên lý hoạt động.....23
4.4.3. Hàm kích hoạt (Activation Functions).....24
4.5. Mạng nơ-ron hồi tiếp (Recurrent Neural Network – RNN).....24
4.5.1. Cấu trúc.....24
4.5.2. Nguyên lý hoạt động.....25
4.5.3. Vấn đề vanishing gradient trong RNN 26
4.5.4. Hàm kích hoạt (Activation Functions).....26
4.6. Bộ nhớ dài-ngắn hạn (Long Short-Term Memory – LSTM) 27
4.6.1. Cấu trúc.....27
4.6.2. Nguyên lý hoạt động.....28
4.7. Mạng Nơ-ron truyền thẳng (Feedforward Neural Network).....29
CHƯƠNG 3. PHÂN LOẠI Thư rác 31
1. Tập dữ liệu thư rác 31
2. Dữ liệu mẫu.....31
3. Kiến trúc.....33
3.1. CNN.....33
3.2. RNN.....34
3.3. LSTM.....34
3.4. FNN 34
CHƯƠNG 4. Kết quả nghiên cứu.....36
1. Thực nghiệm và kết quả.....36
1.1. Môi trường.....36
1.2. Thực nghiệm.....36
1.2.1. CNN 36
1.2.2. LSTM.....38

1.2.3. FNN.....40
1.2.4. RNN 44
1.3. Kết quả.....45
Tài liệu tham khảo.....47

PHỤ LỤC

Hình 1. Lưu đồ quá trình xử lý ngôn ngữ tự nhiên. 11
Hình 2. Sự khác biệt giữa máy học và học sâu 15
Hình 3. Cấu trúc nơ-ron sinh học 17
Hình 4. Cấu trúc mạng nơ-ron nhân tạo 18
Hình 5. Cấu trúc các mô hình học sâu 20
Hình 6. Cấu trúc mô hình CNN 22
Hình 7. Cấu trúc của một mô hình RNN 24
Hình 8. Cấu trúc của một mô hình LSTM 27
Hình 9. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử 36
Hình 10. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử 37
Hình 11. ROC Curve và điểm AUC 38
Hình 12. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử 39
Hình 13. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử 40
Hình 14. ROC Curve và điểm AUC 40
Hình 15. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử 41
Hình 16. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử 42
Hình 17. ROC Curve và điểm AUC 43
Hình 18. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử 44
Hình 19. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử 45

CHƯƠNG 1. GIỚI THIỆU

1. Đặt vấn đề

Trò chuyện trực tuyến đã trở thành một phần không thể thiếu và là một khía cạnh quan trọng trong cuộc sống của chúng ta, nhờ việc sử dụng Internet và mạng xã hội ngày càng gia tăng. Do khả năng dễ dàng tiếp cận, tốc độ và độ tin cậy của nó, Email là một trong những hình thức liên lạc và giao tiếp chính thức và chuyên nghiệp được sử dụng phổ biến nhất. Có tổng cộng 361.1 tỷ email được gửi đi và nhận về mỗi ngày [1] và khoảng 45.6% trong số email này là thư rác (spam email) [2].

Spam email hay còn gọi là **"thư rác"**, được định nghĩa là các tin nhắn không mong muốn được gửi hàng loạt, thường nhằm mục đích quảng cáo hoặc lừa đảo. Các nội dung phổ biến trong thư rác bao gồm: kế hoạch giảm nợ, hẹn hò trực tuyến, các sản phẩm liên quan đến sức khỏe, v.v. Ngoài việc gây phiền toái cho người dùng, thư rác còn đặt ra nhiều thách thức nghiêm trọng, cụ thể như:

1. **Làm tràn ngập hộp thư của người dùng**: Nếu không được phát hiện và lọc bỏ, số lượng thư rác có thể áp đảo, khiến các email quan trọng bị "trôi" và bị lẫn trong vô số thư vô dụng.
2. **Rủi ro bảo mật thông tin**: Một số thư rác chứa mã độc hoặc các liên kết nguy hiểm, có thể đánh cắp thông tin nhạy cảm nếu người dùng vô tình nhấp vào.
3. **Áp lực tài nguyên cho các nhà cung cấp dịch vụ email**: Lượng lớn thư rác tiêu tốn băng thông mạng và dung lượng lưu trữ, gây khó khăn cho các nhà cung cấp dịch vụ Internet (ISP). [3]

Hiện nay, nhằm việc tránh người dùng nhận được các thư rác thì các tổ chức đều cài đặt và giám sát các bộ lọc thư rác. Tuy nhiên, những kẻ gửi thư rác (Spammers) cũng đã phát triển các kỹ thuật mới để tránh được sự phát hiện của bộ lọc thư rác, bảo mật thông tin qua email và phòng tránh lừa đảo qua email là một công việc không dễ dàng nên chúng ta cần cải tiến các phương pháp trước đây.

Trong lĩnh vực phát hiện thư rác, các phương pháp dựa trên học máy (Machine Learning) đã được ứng dụng rộng rãi. Các mô hình kinh điển như Naive Bayes, Random Forest, và SVM (Support Vector Machines) đã chứng minh được hiệu quả trong việc phân loại thư rác. Các mô hình phân loại khác nhau có thể cho ra các kết quả khác nhau, tùy theo loại tập dữ liệu (dataset) và nhiệm vụ mà nó được giao, hầu hết thì

các mô hình này đều cho lại kết quả đáng kinh ngạc khi sử dụng trong việc phát hiện thư rác. Tuy nhiên, các mô hình này vẫn tồn tại một số nhược điểm:

Naive Bayes: Hiệu quả của mô hình này phụ thuộc mạnh mẽ vào chất lượng và độ đa dạng của dữ liệu huấn luyện. Các vấn đề như từ hiếm, từ đồng nghĩa, hay các email có cấu trúc phức tạp khiến mô hình khó hoạt động tốt.

SVM: Khi số đặc trưng (features) lớn, SVM đối mặt với độ phức tạp tính toán cao. Việc xử lý các tập dữ liệu lớn, phức tạp như email có thể tiêu tốn nhiều tài nguyên và thời gian.

Random Forest: Mặc dù có độ chính xác cao, nhưng việc giải thích tại sao một email bị phân loại là spam vẫn là thách thức lớn. để có thể cải thiện hệ thống ta cần biết lý do mà mô hình đưa ra dự đoán.

Để cải thiện hiệu suất, một số nghiên cứu đã kết hợp các mô hình trên bằng kỹ thuật Ensemble Learning, hoặc chuyển sang các phương pháp hiện đại hơn như Deep Learning. Học sâu (Deep Learning) đã cách mạng hóa lĩnh vực học máy, đặc biệt trong các bài toán xử lý ngôn ngữ tự nhiên (NLP). Khả năng học tự động từ dữ liệu thô và trích xuất các đặc trưng phức tạp đã giúp các mô hình học sâu vượt trội so với các thuật toán truyền thống. Các mô hình như Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), và đặc biệt là Transformer (ví dụ: BERT) đã chứng minh hiệu quả vượt trội trong việc phân loại văn bản và phát hiện thư rác.

Mục tiêu của nghiên cứu này là khai thác sức mạnh của các mô hình học sâu như RNN, CNN, LSTM và FNN trong việc phát hiện thư rác, và so sánh hiệu suất của chúng. Điều này không chỉ giúp cải thiện độ chính xác mà còn mở ra các hướng đi mới trong việc xây dựng hệ thống bảo mật qua email.

2. Các nghiên cứu liên quan

Bài toán phát hiện thư rác là chủ đề thu hút sự quan tâm của nhiều nhà nghiên cứu. Nghiên cứu gần đây cho thấy rằng việc áp dụng học sâu để phát hiện thư rác đạt được kết quả tốt hơn đáng kể so với các phương pháp truyền thống.

Nghiên cứu của Isra’a AbdulNabi và Qussai Yaseen %. đã chỉ ra rằng cả hai mô hình học sâu DNN (Deep Neural Network), bao gồm BERT (Bidirectional Encoder Representations from Transformers) và BiLSTM (Bidirectional Long Short-Term Memory), đều đạt độ chính xác cao hơn các phương pháp truyền thống như KNN, Naive Bayes. Đặc biệt, mô hình BERT đã đạt được độ chính xác lên tới 98.67% và điểm F1 là 98.66 [4].

Souad Larabi-Marie-Sainte, Sanaa Ghouzali và cộng sự [5] sau khi thử nghiệm các mô hình khác nhau đã điều chỉnh cài đặt của mô hình DRNN (Deep Recurrent Neural Network) với hàm kích hoạt Tanh, tỉ lệ Dropout bằng 0.1 và số epoch đạt 100, đạt được độ chính xác cao lên đến 99.7%. Phương pháp này vượt trội so với cách tiếp cận trước đó là Hybrid-GRU-RNN với độ chính xác đạt được là 98.7%, và so với độ chính xác 94.2% từ mô hình rừng ngẫu nhiên (Random Forest) vào năm 2018 [6].

Năm 2021, Ala Mughaid, Shadi AlZu'bi và cộng sự [7] đã đề xuất một phương pháp phát hiện sử dụng kỹ thuật máy học bằng cách chia dữ liệu để huấn luyện và xác thực bằng dữ liệu thử nghiệm. Phương pháp này đã cho thấy độ chính xác lên tới 0.88, 1.00 và 0.97 đối với cây quyết định nâng cao (Boosted Decision Tree) trên các tập dữ liệu được sử dụng.

Khan Farhan Rafat, Qin Xin và đồng nghiệp [8] chỉ ra rằng việc tiền xử lý văn bản giúp nâng cao độ chính xác của mô hình phát hiện. Họ đã thực hiện các thí nghiệm với hai trường hợp: 1) trước khi tiền xử lý văn bản, sử dụng mô hình NB và LSTM, đạt độ chính xác lần lượt là 90.02% và 96.01%; 2) sau khi tiền xử lý, mô hình LSTM đạt độ chính xác 97.18%. Điều này cho thấy LSTM (mô hình học sâu) không chỉ đạt kết quả tốt hơn so với NB (mô hình máy học truyền thống), mà còn mang lại hiệu suất cao hơn về tổng thể.

Mohammad Alauthman [9] vào năm 2020 đã phát triển một mô hình GRU-RNN kết hợp với SVM để phát hiện các email được gửi bởi bot. Mô hình của ông đã đạt độ chính xác lên đến 98.7%, vượt qua các phương pháp truyền thống khác như logistic regression, Naive Bayes, và Random Forest, với độ chính xác lần lượt là 84.9%, 76.3%, 94.4%, và 95.7%.

3. Mục tiêu

Mục tiêu chính của nghiên cứu này là so sánh hiệu quả của các mô hình học sâu trong việc phát hiện thư rác. Cụ thể, nghiên cứu sẽ triển khai và thử nghiệm các mô hình trên cùng một tập dữ liệu chứa email đã được phân loại trước, từ đó đánh giá và so sánh dựa trên các tiêu chí quan trọng, bao gồm: độ chính xác (accuracy), độ nhạy (recall), giá trị dự đoán dương (precision), F1-score, và đặc biệt là AUC-ROC (Area Under the Receiver Operating Characteristic Curve) một chỉ số phổ biến trong phân loại nhị phân, đặc biệt hữu ích khi dữ liệu bị mất cân bằng.

Bên cạnh việc đánh giá hiệu suất kỹ thuật, nghiên cứu còn tập trung phân tích các ưu điểm và hạn chế của từng mô hình trong bối cảnh ứng dụng thực tế. Điều này nhằm xác định mô hình nào có tiềm năng ứng dụng cao nhất, không chỉ trong môi trường

nghiên cứu mà còn trong việc tích hợp vào các hệ thống thực tế để phát hiện và ngăn chặn thư rác một cách hiệu quả. Đồng thời, nghiên cứu cũng hy vọng sẽ đóng góp vào việc phát triển các hệ thống lọc thư rác tự động, nâng cao tính chính xác và hiệu quả trong việc bảo vệ người dùng khỏi các mối đe dọa từ thư rác.

4. Đối tượng và phạm vi nghiên cứu

4.1.Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài này là các mô hình học sâu, được áp dụng trong bài toán phân loại thư rác. Cụ thể, nghiên cứu tập trung vào bốn mô hình tiêu biểu: Feedforward neural network (FNN),Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), và Convolutional Neural Network (CNN). Các mô hình này được lựa chọn trên khả năng xử lý và phân tích dữ liệu văn bản hiệu quả, phù hợp với yêu cầu của bài toán phát hiện thư rác.

4.2.Phạm vi nghiên cứu

a. Thu thập và xử lý dữ liệu:

Nguồn dữ liệu: Sử dụng tập dữ liệu chứa các thư điện tử đã được gắn nhãn (bao gồm cả thư rác và thư hợp lệ), chẳng hạn như các bộ dữ liệu công khai như Enron Spam Dataset, SpamAssassin, hoặc các nguồn tương tự.

Tiền xử lý dữ liệu: Áp dụng các kỹ thuật như tokenization, stemming, lemmatization.

b. Huấn luyện và đánh giá mô hình:

Triển khai các mô hình: (CNN, RNN, LSTM, FNN).

Tiến hành đánh giá hiệu suất dựa trên các tiêu chí:

Độ chính xác (Accuracy)

Độ nhạy (Recall)

Giá trị dự đoán dương (Precision)

F1-score

Diện tích dưới đường cong ROC (AUC-ROC)

c. Mở rộng và kiểm tra tính tổng quát hóa

Kiểm tra hiệu suất của các mô hình khác trên các tập dữ liệu chưa từng thấy nhằm đánh giá khả năng tổng quát hóa.

Mô phỏng các tình huống thực tế, như phân loại thư rác với dữ liệu bị nhiễu (noise), để xác định tính ổn định và khả năng ứng dụng thực tiễn của các mô hình.

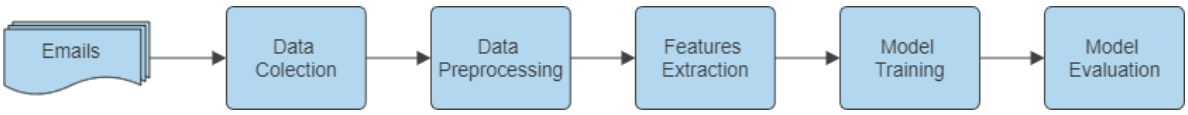
d. So sánh và đề xuất

So sánh kết quả của từng mô hình để xác định mô hình có hiệu suất cao nhất, phù hợp nhất với bài toán.

Ngoài việc đánh giá hiệu suất trên tập dữ liệu cố định, phạm vi nghiên cứu cũng đề xuất mở rộng để kiểm tra khả năng tổng quát hóa của các mô hình này trên các tập dữ liệu khác nhau hoặc trong các tình huống thực tế khác nhau. Điều này nhằm xác định mô hình nào có tiềm năng ứng dụng cao nhất trong việc phát hiện và ngăn chặn thư rác trong môi trường thực tế.

5. Phương pháp nghiên cứu

Quá trình xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) thường bao gồm 5 bước chính: Thu thập dữ liệu (Data Collection), Tiền xử lý dữ liệu (Data Preprocessing), Trích xuất đặc trưng (Feature Extraction), Huấn luyện mô hình (Model Training), và Đánh giá mô hình (Model Evaluation). Các bước này được minh họa trong Hình 1.



Hình 1. Lưu đồ quá trình xử lý ngôn ngữ tự nhiên.

5.1.Thu thập dữ liệu (Data Colection)

Tập dữ liệu được sử dụng trong nghiên cứu này là *Spam Mails* từ Kaggle [10], bao gồm 5171 email với 3672 email hợp lệ và 1499 email rác. Dữ liệu này thể hiện rõ tính mất cân bằng giữa hai lớp (lớp hợp lệ chiếm tỷ lệ lớn hơn nhiều so với lớp rác).

Để đảm bảo mô hình học sâu có thể xử lý tốt, nghiên cứu sử dụng kỹ thuật SMOTE (Synthetic Minority Over-sampling Technique) để cân bằng dữ liệu. SMOTE tạo ra các mẫu mới từ dữ liệu hiện có bằng cách nội suy giữa các điểm dữ liệu của lớp thiểu số, giúp cải thiện hiệu quả mô hình mà không làm mất dữ liệu gốc.

5.2.Tiền xử lý dữ liệu (Data Preprocessing)

Quá trình tiền xử lý dữ liệu là một bước quan trọng trong xử lý ngôn ngữ tự nhiên, đóng vai trò quan trọng trong việc chuẩn bị dữ liệu văn bản để các mô hình học sâu có thể xử lý hiệu quả.

A. Làm sạch dữ liệu (Data Cleaning)

Xóa bỏ các ký tự không cần thiết: Loại bỏ những ký tự không mang lại thông tin hữu ích hoặc không liên quan đến việc phân tích ngôn ngữ. Ví dụ, các dấu câu như "!" hay "#".

Xử lý khoảng trắng: Loại bỏ các khoảng trắng không cần thiết hoặc sắp xếp lại các khoảng trắng để đảm bảo tính toàn vẹn của văn bản.

Xử lý các ký tự số: Số liệu trong văn bản có thể được được thay thế, loại bỏ hoặc giữ nguyên tùy vào mục đích phân tích.

Xử lý các từ lập: Trong ngôn ngữ tự nhiên, văn bản có thể chứa các từ được nhấn mạnh bởi người gửi bằng cách lập từ, ví dụ như "Chàooooo" thì việc chuẩn hóa nó thành "chào" giúp cho mô hình hiểu rõ hơn.

Chuyển đổi văn bản thành chữ thường: Các từ được viết hoa cần được chuyển thành chữ thường để tránh việt phân biệt giữa chữ hoa và thường. Ví dụ, "FREE" và "free" sẽ được coi là một sau khi được chuyển đổi.

B. Loại bỏ từ dừng (Stopword Removal)

Từ dừng là những từ phổ biến và thường xuất hiện trong văn bản nhưng không mang nhiều thông tin ngữ nghĩa, như "là," "và," "nhưng," "của," v.v. Việc loại bỏ các từ này giúp giảm kích thước của dữ liệu mà vẫn giữ được thông tin quan trọng cho việc phân tích.

C. Phân loại từ (Tokenization)

Phân loại từ là quá trình ta chia văn bản thành các đơn vị nhỏ hơn, thường là chia câu thành từ hoặc cụm từ. Ví dụ, "Tôi thích nghe nhạc" sẽ được tách thành các token: ["Tôi", "thích", "nghe", "nhạc"]. Đây là bước cơ bản để chuẩn bị cho bước tiếp theo.

D. Bổ đề ngôn ngữ (Lemmatization)

Lemmatization là đưa các từ về định dạng gốc được thực hiện bằng cách sử dụng một bảng tra cứu các từ vựng gốc của các từ trong câu văn, và có thể có một số quy tắc để xử lý các từ mà ta chưa từng được nhìn thấy trước đây. Ví dụ, các từ "mouse", "mice" sẽ được đưa cùng về một dạng như nhau.

5.3. Trích xuất đặc trưng (Features Extraction)

Văn bản cần được chuyển đổi thành chuỗi số để các mô hình học sâu có thể xử lý. Nghiên cứu này sử dụng ***Word Embedding*** để biểu diễn văn bản dưới dạng vector

số, cụ thể là các công cụ ***TF-IDF***. Word Embedding giúp mô hình hiểu được mối quan hệ ngữ nghĩa giữa các từ trong văn bản, ví dụ: vector của từ "king" gần với vector của từ "queen".

Kích thước vector nhỏ hơn giúp giảm thiểu tài nguyên tính toán, đặc biệt khi sử dụng các mô hình học sâu như RNN, LSTM, hoặc CNN.

5.4.Huấn luyện mô hình (Model Training)

Bốn mô hình học sâu được triển khai và so sánh trong nghiên cứu này:

5.4.1. Convolutional Neural Network (CNN)

CNN thường được sử dụng trong xử lý ảnh, nhưng cũng có thể áp dụng cho dữ liệu văn bản. Mô hình này sử dụng các lớp tích chập để trích xuất đặc trưng từ dữ liệu văn bản, giúp nó nhận diện các mẫu quan trọng trong email.

CNN sẽ được huấn luyện trên các chuỗi văn bản đã được chuyển đổi thành ma trận số. Các siêu tham số như kích thước bộ lọc, số lượng bộ lọc, và kích thước bước nhảy (stride) sẽ được điều chỉnh để tối ưu hóa hiệu suất.

5.4.2. Recurrent Neural Network (RNN)

RNN là một loại mạng nơ-ron được thiết kế để xử lý các dữ liệu tuần tự, chẳng hạn như văn bản. RNN có khả năng ghi nhớ thông tin từ các bước trước đó, giúp nó xử lý các chuỗi dữ liệu hiệu quả.

RNN được thiết kế để xử lý dữ liệu tuần tự như văn bản. Mô hình ghi nhớ thông tin từ các bước trước, giúp xử lý hiệu quả chuỗi dữ liệu dài. RNN được tối ưu thông qua các siêu tham số như kích thước trạng thái ẩn và tỷ lệ dropout để ngăn chặn hiện tượng quá khớp (overfitting).

5.4.3. Long Short-term Memory (LSTM)

LSTM là biến thể của RNN, giải quyết vấn đề về ghi nhớ thông tin dài hạn. Với cấu trúc gồm các "cổng" (gates) như Forget Gate, Input Gate, và Output Gate, LSTM có khả năng xử lý các chuỗi dữ liệu phức tạp hơn.

Mô hình LSTM sẽ được huấn luyện tương tự như RNN, nhưng với cấu trúc đặc biệt của các tế bào LSTM. Các siêu tham số như số lượng đơn vị LSTM và tỷ lệ dropout sẽ được tối ưu hóa để đạt hiệu suất tốt nhất.

5.4.4. Feedforward Neural Network (FNN)

FNN là viết tắt của Feedforward Neural Network (Mạng nơ-ron truyền thẳng), một trong những kiến trúc cơ bản nhất của mạng nơ-ron nhân tạo. Đây là loại

mạng nơ-ron không có vòng lặp trong cấu trúc của nó, có nghĩa là thông tin di chuyển theo một hướng duy nhất, từ đầu vào qua các lớp ẩn đến đầu ra.

5.5.Đánh giá mô hình (Model Evaluation)

Các mô hình được đánh giá dựa trên các chỉ số quan trọng như:

Accuracy: Độ chính xác tổng thể.

Precision, Recall, F1-score: Đặc biệt quan trọng trong bài toán phân loại nhị phân (spam vs ham).

AUC-ROC: Đánh giá khả năng phân tách giữa hai lớp.

Mục tiêu là xác định mô hình hiệu quả nhất để áp dụng trong thực tế, đảm bảo tính chính xác và đáng tin cậy trong việc phân loại email.

6. Bố cục của bài niên luận cơ sở

Bài niên luận cơ sở này bao gồm 4 chương:

Chương 1: Phần giới thiệu. Giới thiệu chung mục đích, mục tiêu, đối tượng, phạm vi nghiên cứu, và phương pháp nghiên cứu về vấn đề phát hiện và phân loại thư rác.

Chương 2: Cơ sở lý thuyết. Giải thích các khái niệm quan trọng và cách hoạt động của các mô hình được sử dụng trong bài nghiên cứu.

Chương 3: Phân loại Email. Chương này sẽ đi sâu vào bài toán phát hiện thư rác bằng cách áp dụng các mô hình học sâu được đề cập đến trong các chương trước đó.

Chương 4: Kết quả nghiên cứu và hướng phát triển.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

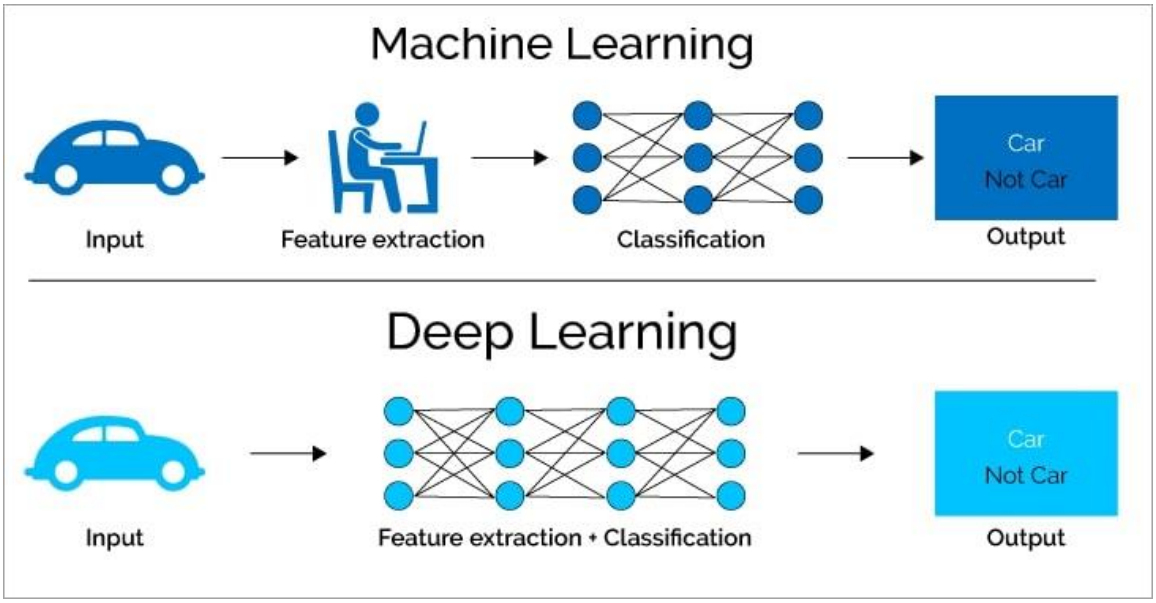
1. Thư rác

Thư rác là các email không mong muốn được gửi đến hàng loạt nhằm mục đích quảng cáo, phát tán thông tin không lành mạnh, hoặc đánh cắp thông tin cá nhân người dùng.

Thư rác có thể gây ra các hậu quả khó lường trước, tăng rủi ro an ninh mạng (phần mềm độc hại, lừa đảo qua email), gây thất thoát tài nguyên (bộ nhớ email server và băng thông mạng)

2. Máy học, học sâu và xử lý ngôn ngữ tự nhiên

Máy học (machine learning) là một nhánh của trí tuệ nhân tạo, tập trung vào việc tạo ra các hệ thống có khả năng tự học từ dữ liệu mà không cần lập trình chi tiết. Thay vì dựa vào các quy tắc được lập sẵn, các thuật toán học máy sử dụng dữ liệu huấn luyện để xây dựng mô hình nhằm dự đoán hoặc ra quyết định. Ví dụ, một hệ thống học máy có thể học cách phân loại thư điện tử thành thư rác (spam) hoặc thư hợp lệ một cách tự động. Máy học liên quan đến thống kê nhưng nhấn mạnh vào giải thuật và khả năng tính toán phức tạp.



Hình 2. Sự khác biệt giữa máy học và học sâu [11]

Học sâu (deep learning) là một nhánh của học máy, sử dụng các mô hình mạng nơ-ron nhân tạo với nhiều lớp ẩn để phân tích và học từ dữ liệu. Học sâu có khả năng

xử lý dữ liệu phức tạp và lớn, chẳng hạn như hình ảnh, âm thanh và văn bản, bằng cách tự động trích xuất các đặc trưng mà không cần can thiệp thủ công.

Các mô hình học sâu, đặc biệt là Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), và Convolutional Neural Networks (CNN), đã mang lại những bước đột phá trong nhiều lĩnh vực, bao gồm nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, và phát hiện thư rác (spam email). Sự phát triển của học sâu chủ yếu nhờ vào sức mạnh tính toán tăng cao và lượng dữ liệu lớn hơn, cho phép các mô hình này học được các đặc trưng sâu hơn và phức tạp hơn so với các thuật toán học máy truyền thống.

Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) là một nhánh của trí tuệ nhân tạo tập trung vào các ứng dụng trên ngôn ngữ của con người. Trong trí tuệ nhân tạo thì xử lý ngôn ngữ tự nhiên là một trong những phần khó nhất vì nó liên quan đến việc phải hiểu ý nghĩa ngôn ngữ-công cụ hoàn hảo nhất của tư duy và giao tiếp [12].

3. Mô hình

Mô hình máy học là một loại mô hình toán học, sau khi được "huấn luyện" trên một tập dữ liệu nhất định, có thể được sử dụng để đưa ra dự đoán hoặc phân loại trên dữ liệu mới. Trong quá trình huấn luyện, một thuật toán học sẽ điều chỉnh dần dần các tham số bên trong của mô hình nhằm giảm thiểu lỗi trong các dự đoán của nó. Thuật ngữ "mô hình" có thể đề cập đến nhiều cấp độ chi tiết khác nhau, từ một loại mô hình chung và các thuật toán học liên quan, đến một mô hình đã được huấn luyện hoàn chỉnh với tất cả các tham số đã được tối ưu [13].

Nhiều loại mô hình khác nhau đã được sử dụng và nghiên cứu trong các hệ thống học máy. Việc lựa chọn mô hình tốt nhất cho một tác vụ cụ thể được gọi là lựa chọn mô hình.

Chương 1 đã giới thiệu tổng quát về các mô hình được sử dụng trong bài nghiên cứu này, phần sau đây sẽ giải thích rõ hơn về cách hoạt động cũng như cơ sở lý thuyết của chúng.

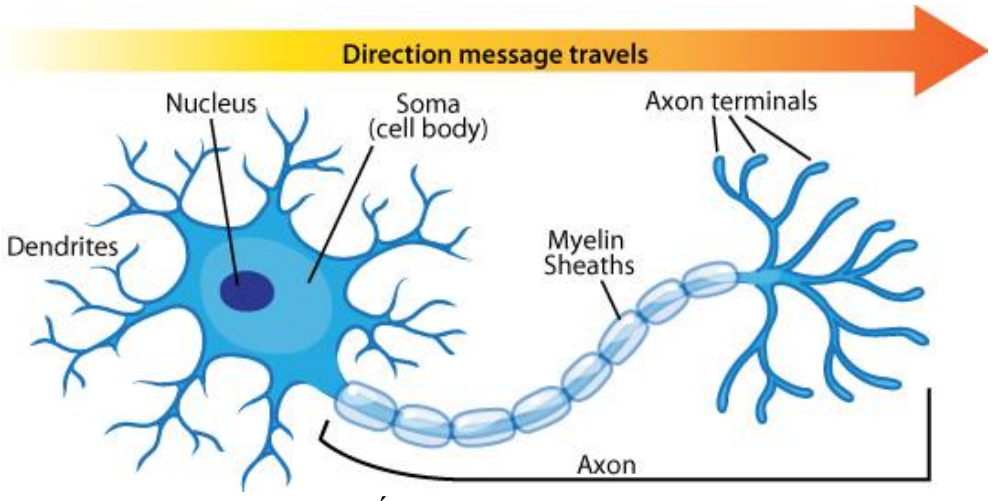
4. Mô hình mạng nơ-ron nhân tạo

4.1.Nơ-ron sinh học

Tế bào thần kinh, hay nơ-ron, là những tế bào đặc biệt có chức năng nhận, xử lý và truyền tải thông tin dưới dạng các xung điện sinh học. Chúng là đơn vị cấu tạo cơ bản của hệ thần kinh, đóng vai trò trung tâm trong việc điều khiển mọi hoạt động của

cơ thể, từ suy nghĩ, cảm xúc cho đến các phản xạ đơn giản. Ước tính có khoảng 100 tỷ nơ-ron thần kinh trong não người, chiếm 10% trong tổng só tế bào não.

Cấu tạo của nơ-ron được chia thành 3 phần chính: Thân tế bào (Soma), Sợi trục (axon), Sợi nhánh (dendrite). Thân tế bào là phàn trung tâm của nơ-ron, chứa nhân và các bào quan, nơi diễn ra các hoạt động sống của tế bào. Sợi nhánh là những sợi nhánh ngắn, phân nhánh nhiều, có chức năng nhận tín hiệu từ các nơ-ron khác hoặc từ các thụ thể cảm giác. Sợi trục là một sợi dìa, thường có bao myelin bao bọc có chức năng truyền tính hiệu đi xa đến các nơ-ron khác, cơ hoặc tuyến.



Hình 3. Cấu trúc nơ-ron sinh học [14]

Cách hoạt động của nơ-ron sinh học có thể hiểu đơn giản như sau:

Nơ-ron sinh học hoạt động như những đơn vị thông tin cơ bản trong hệ thần kinh, đóng vai trò như những "dây điện thông minh" truyền tải thông tin dưới dạng các xung điện sinh học. Khi một nơ-ron nhận được đủ kích thích, nó sẽ tạo ra một xung điện lan truyền dọc theo sợi trục đến các nơ-ron khác hoặc các cơ quan đích. Tại các điểm nối giữa các nơ-ron, gọi là khớp thần kinh, thông tin được truyền qua các chất dẫn truyền thần kinh. Quá trình này liên tục diễn ra, tạo thành một mạng lưới phức tạp cho phép chúng ta suy nghĩ, cảm nhận và tương tác với thế giới xung quanh. Mức độ bền vững của các liên kết này xác định mộ hệ số gọi là trọng số liên kết.

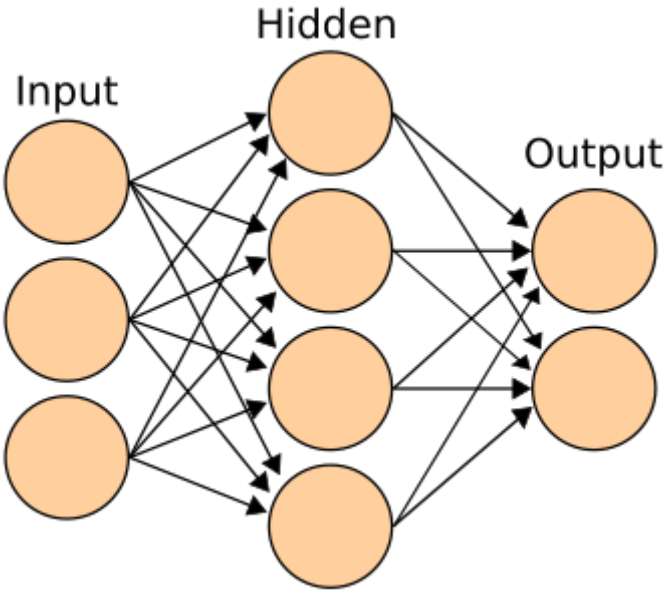
4.2.Nơ-ron nhân tạo

Nhằm mô phỏng các tế bào thần kinh và các khớp nối thần kinh của bộ não con người, mạng thần kinh nhân tạo (Artifical Neural Network) có các thành phần có vai trò tương tự như các nơ-ron nhân tạo và kết nối giữa chúng (kết nối này gọi là weights). Nơ-ron là một đơn vị tính toán có nhiều đầu vào và một đầu, mỗi đầu vào đến từ một khớp nối thần kinh (synapse). Đặc trưng của nơ-ron là một hàm kích hoạt phi tuyến

chuyển đổi một số tổ hợp tuyến tính của tất cả các tín hiệu đầu vào thành tín hiệu đầu ra.

Một nơ-ron nhân tạo là một đơn vị tính toán hay đơn vị xử lý thông tin cơ sở cho hoạt động của một mạng nơ-ron.

4.2.1. Các thành phần cơ bản của một mô hình nơ-ron nhân tạo



Hình 4. Cấu trúc mạng nơ-ron nhân tạo [15]

Nơ ron nhân tạo (Artifical Neuron):

Cấu trúc: Một nơ-ron nhân tạo bao gồm các đầu vào, trọng số tương ứng, một hàm tính tổng (tổng tín hiệu đầu vào), và một hàm kích hoạt. Đầu ra của nơ-ron được truyền tới các nơ-ron khác thông qua các liên kết có trọng số.

Tổng tín hiệu đầu vào (net input) là quá trình mà một nơ-ron nhận tất cả các tín hiệu từ các nơ-ron trước đó và cộng chúng lại. Mỗi tín hiệu từ một nơ-ron được nhân với trọng số của nó, rồi tất cả các giá trị này được cộng vào với nhau để tạo thành một giá trị tổng hợp. Được tính với công thức:

$$z = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + b$$

Trong đó, w_i là trọng số, x_i là các giá trị đầu vào, và b là bias.

Hàm kích hoạt (Activation Function): Sau khi tính tổng tín hiệu, hàm kích hoạt được sử dụng để quyết định xem nơ-ron có nên kích hoạt hay không. Các hàm kích hoạt phổ biến bao gồm:

Sigmoid: $\sigma(z) = \frac{1}{1+ e^{-z}}$

ReLU (Rectified Linear Unit): ReLU(z) = max(0,z)

Tanh: $tanh(z) = \frac{e^{z}- e^{-z}}{e^{z}+ e^{-z}}$

Lớp Neuron (Layers)

Lớp đầu vào (Input layer): Nhận dữ liệu đầu vào từ bên ngoài.

Lớp ẩn (Hidden layers): xử lý và học các đặc trưng từ dữ liệu đầu vào, mỗi nơ-ron trong lớp này kết nối với nơ-ron khác.

Lớp đầu ra (Output layer): Trả về kết quả dự đoán của mạng.

Nguyên lý hoạt động

Quá trình lan truyền (Forward Propagation): Dữ liệu được đưa từ lớp đầu vào, qua các lớp ẩn, và cuối cùng đến lớp đầu ra. Mỗi nơ-ron tính toán tổng tín hiệu đầu vào dựa trên trọng số và sau đó áp dụng hàm kích hoạt để tạo ra đầu ra.

Hàm mất mát (Loss Function): Đo lường sự khác biệt giữa đầu ra dự đoán của mạng và kết quả thực tế. Ví dụ:

Cross-Entropy Loss: Thường dùng cho các bài toán phân loại**.**

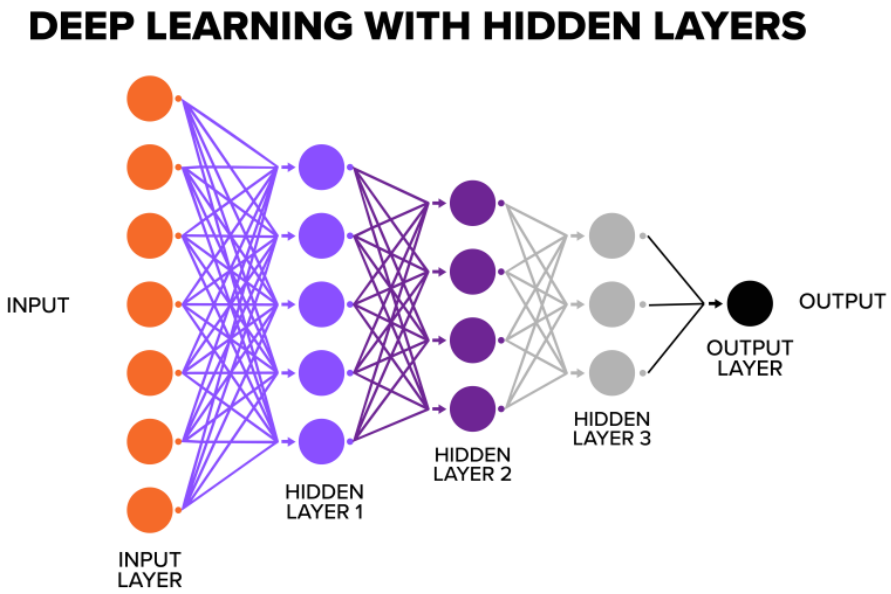
Mean Squared Error (MSE): Sử dụng trong các bài toán hồi quy**.**

Lan truyền ngược (Backpropagation): Đây là quá trình tính toán gradient của hàm mất mát đối với các trọng số bằng cách sử dụng quy tắc chuỗi, từ đó điều chỉnh các trọng số để giảm thiểu hàm mất mát. Công thức cập nhật trọng số trong Gradient Descent là:

$$w_{new} = w_{old} - \eta \cdot \nabla L(w)$$

Trong đó, η là tốc độ học (learning rate), $\nabla L(w)$ là gradient của hàm mất mát $L(w)$ đối với trọng số *w*.

4.3.Mạng nơ-ron sâu (Deep Neural Network – DNN)



Hình 5. Cấu trúc các mô hình học sâu [16]

Mạng nơ-ron sâu (DNN) là một dạng mạng nơ-ron nhân tạo có nhiều lớp ẩn (hidden layers) giữa lớp đầu vào (input layer) và lớp đầu ra (output layer). Sự tăng cường về số lượng lớp giúp DNN có khả năng học và biểu diễn các đặc trưng phức tạp hơn từ dữ liệu, đồng thời tạo ra các mô hình chính xác hơn cho nhiều bài toán khác nhau.

4.3.1. Cấu trúc

Lớp đầu vào (Input layer):

Nhận dữ liệu đầu vào từ bên ngoài. Mỗi nơ-ron trong lớp này tương ứng với một đặc trưng của dữ liệu. Trong phân loại email, các nơ-ron trong lớp đầu vào có thể đại diện cho tần suất xuất hiện của các từ trong email.

Lớp ẩn (Hidden layers):

Là nơi xử lý chính diễn ra trong DNN. Có thể có nhiều lớp ẩn, mỗi lớp gồm một số lượng nơ-ron lớn. Các nơ-ron trong lớp ẩn không chỉ truyền tín hiệu từ lớp trước mà còn thực hiện các phép tính để trích xuất và học các đặc trưng phức tạp hơn.

Lớp đầu ra (Output layer):

Cung cấp kết quả cuối cùng của mô hình, thường là các giá trị dự đoán hoặc xác suất. Số nơ-ron trong lớp này phụ thuộc vào bài toán. Ví dụ, trong bài toán phân loại thư

rác, có thể chỉ cần một nơ-ron với hàm kích hoạt sigmoid cho phân loại nhị phân hoặc nhiều nơ-ron với hàm softmax cho phân loại đa lớp.

4.3.2. Nguyên lý hoạt động

a. Lan truyền tiến (Forward Propagation):

- Dữ liệu được truyền từ lớp đầu vào qua các lớp ẩn đến lớp đầu ra.
- Mỗi nơ-ron trong lớp ẩn nhận tín hiệu từ các lớp nơ-ron ở lớp trước đó, tính toán tổng tín hiệu đầu vào (bao gồm trọng số và bias), sau đó áp dụng hàm kích hoạt để tạo đầu ra.
- Công thức tính tổng tín hiệu đầu vào cho một nơ-ron là:

$$z_j = \sum_{i=1}^{n} w_{ij} x_i + b_j$$

Trong đó:

z_j là tổng tín hiệu đầu vào cho nơ-ron thứ *j*.

w_{ij} là trọng số của kết nối từ nơ-ron *i* đến nơ-ron *j*.

x_i là đầu vào từ nơ-ron *i*.

b_j là bias cho nơ-ron *j*.

Đầu ra của nơ-ron *j* được tính bằng cách áp dụng hàm kích hoạt:

$$a_j = f(z_j)$$

Trong đó f có thể là các hàm kích hoạt như *sigmoid*, *ReLU*, hay *tanh*.

b. Lan truyền ngược (Backpropagation)

Sau khi tính toán đầu ra, mô hình sẽ tính toán độ sai lệch giữa đầu ra dự đoán và giá trị thực tế bằng cách sử dụng một hàm mất mát (loss function).

Gradient của hàm mất mát sẽ được tính toán đối với từng trọng số và bias bằng cách sử dụng quy tắc chuỗi (chain rule) để cập nhật trọng số. Các trọng số được cập nhật theo công thức:

$$w_{ij}^{new} = w_{ij}^{old} - \eta \frac{\partial L}{\partial w_{ij}}$$

Trong đó, η là tốc độ học (learning rate), $\frac{\partial L}{\partial w_{ij}}$ là gradient của hàm mất mát đối với trọng số w_{ij}.

4.3.3. Hàm kích hoạt (Activation Functions)

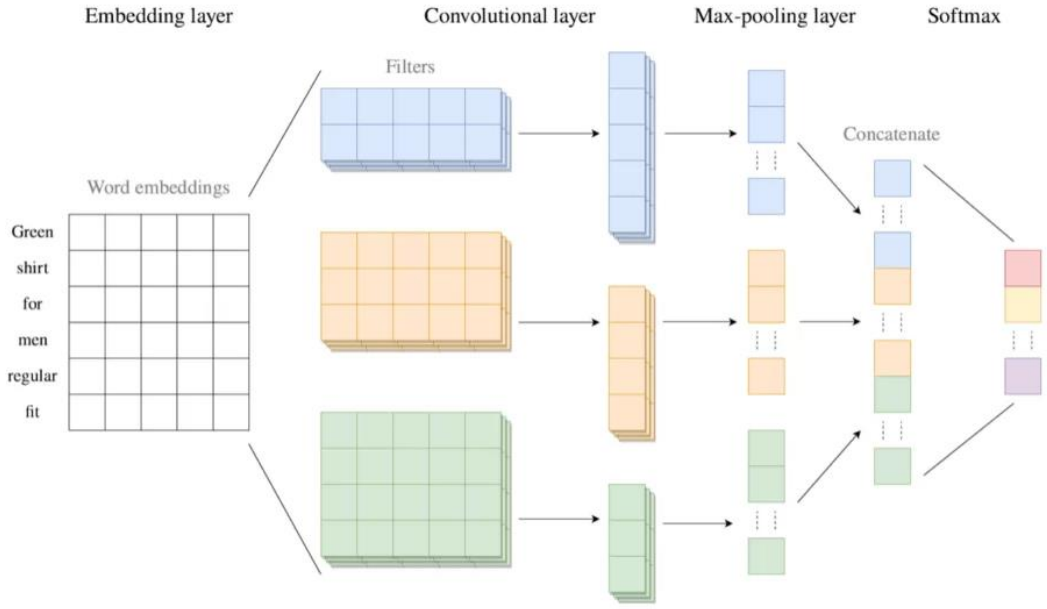
Hàm kích hoạt đóng vai trò quan trọng trong DNN, giúp tạo ra tính phi tuyếncho mô hình. Một số hàm kích hoạt phổ biến bao gồm:

Sigmoid: Thích hợp cho các bài toán phân loại nhị phân.

ReLU (Rectified Linear Unit): Giúp giải quyết vấn đề vanishing gradient và thường được sử dụng trong các lớp ẩn.

Tanh: Cung cấp đầu ra trong khoảng [-1, 1], thích hợp cho việc chuẩn hóa dữ liệu.

4.4.Mạng nơ-ron tích chập (Convolutional Neural Network – CNN)



Hình 6. Cấu trúc mô hình CNN [17]

Mạng tích chập (CNN) là một loại mạng nơ-ron nhân tạo được thiết kế đặc biệt cho việc xử lý dữ liệu có cấu trúc không gian, như hình ảnh và video. CNN đã chứng minh hiệu quả vượt trội trong các bài toán nhận diện hình ảnh và phân loại nhờ khả năng tự động trích xuất các đặc trưng quan trọng từ dữ liệu đầu vào.

4.4.1. Cấu trúc

CNN thường bao gồm các thành phần chính sau:

Lớp đầu vào (Input layer):

Nhận dữ liệu đầu vào, thường là hình ảnh có định dạng ma trận (ví dụ: 2D cho hình ảnh, 3D cho video).

Lớp tích chập(Convolutional layer):

Là thành phần chính của CNN, nơi các bộ lọc (filters hoặc kernels) được áp dụng để trích xuất các đặc trưng từ dữ liệu. Mỗi bộ lọc di chuyển qua hình ảnh, thực hiện phép tích chập (convolution) và tạo ra một bản đồ đặc trưng (feature map) cho mỗi đặc trưng mà nó phát hiện.

Lớp gộp (Pooling layer):

Thường xuất hiện sau lớp tích chập, lớp gộp giúp giảm kích thước của bản đồ đặc trưng, từ đó giảm số lượng tham số và độ phức tạp của mô hình. Phép gộp phổ biến nhất *là **gộp max (max pooling)***, trong đó giá trị lớn nhất trong mỗi vùng được giữ lại.

Lớp kết nối đầy đủ (Fully Connected Layer):

Lớp cuối cùng của mạng, nơi các đặc trưng đã được trích xuất được kết hợp để đưa ra dự đoán cuối cùng. Mỗi nơ-ron trong lớp này kết nối đầy đủ với tất cả nơ-ron ở lớp trước đó.

4.4.2. Nguyên lý hoạt động

a. Convolution (Tích Chập):

Phép tích chập: Khi một bộ lọc di chuyển qua hình ảnh, nó nhân các giá trị pixel trong vùng ảnh với các trọng số của bộ lọc, sau đó cộng lại để tạo ra một giá trị duy nhất cho mỗi vị trí.

Bản đồ đặc trưng: Kết quả của phép tích chập tạo ra bản đồ đặc trưng cho một đặc trưng nhất định, giúp mạng nắm bắt các thông tin như cạnh, đường nét, hình dạng.

b. Pooling (Gộp):

Giảm kích thước: Sau khi trích xuất đặc trưng, lớp gộp giúp giảm kích thước của bản đồ đặc trưng, giữ lại thông tin quan trọng và loại bỏ những thông tin không cần thiết.

Gộp max: Lấy giá trị lớn nhất từ một vùng nhỏ trong bản đồ đặc trưng để giảm kích thước mà không làm mất thông tin.

c. Forward Propagation (Lan truyền tiến):

Dữ liệu đi từ lớp đầu vào qua các lớp tích chập và gộp, sau đó qua các lớp fully connected để tạo ra dự đoán cuối cùng.

d. Backpropagation (Lan truyền ngược):

Giống như trong các mạng nơ-ron khác, CNN sử dụng quá trình lan truyền ngược để cập nhật trọng số dựa trên lỗi giữa đầu ra dự đoán và giá trị thực tế.

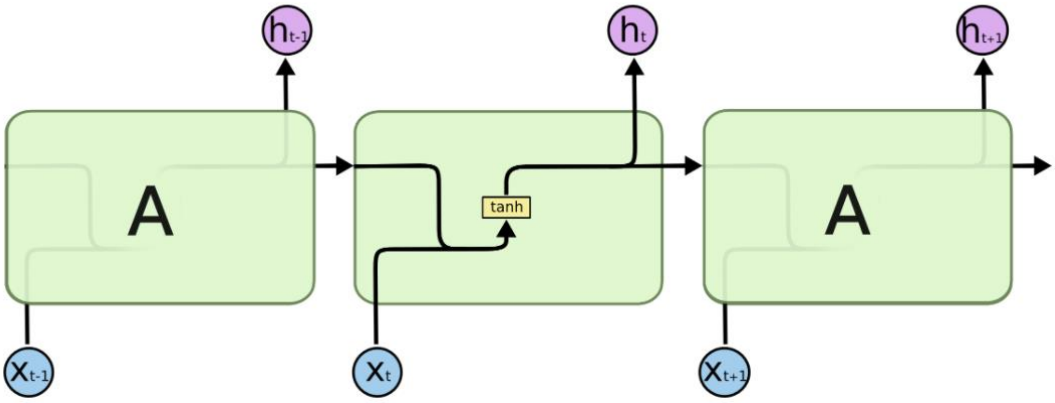
4.4.3. Hàm kích hoạt (Activation Functions)

ReLU (Rectified Linear Unit): Hàm kích hoạt phổ biến trong CNN, giúp giảm thiểu vấn đề vanishing gradient và cho phép mạng học các đặc trưng phi tuyến.

$$ReLU(x) = \max(0, x)$$

Softmax: Thường được sử dụng trong lớp đầu ra của CNN cho bài toán phân loại đa lớp, giúp chuyển đổi đầu ra thành xác suất.

4.5.Mạng nơ-ron hồi tiếp (Recurrent Neural Network – RNN)



Hình 7. Cấu trúc của một mô hình RNN [18]

RNN là một loại mạng nơ-ron đặc biệt được thiết kế để xử lý dữ liệu tuần tự hoặc chuỗi thời gian, chẳng hạn như văn bản, giọng nói, hoặc chuỗi sự kiện. Điểm đặc trưng của RNN là khả năng ghi nhớ thông tin từ các bước trước đó thông qua trạng thái ẩn (hidden state), cho phép mô hình nắm bắt được mối quan hệ giữa các bước trong chuỗi dữ liệu. Điều này giúp RNN trở nên đặc biệt hữu ích trong các bài toán yêu cầu phân tích ngữ cảnh hoặc các chuỗi dài hạn.

4.5.1. Cấu trúc

Mạng nơ-ron hồi tiếp khác với mạng nơ-ron truyền thẳng (feedforward neural network) ở chỗ nó có các kết nối hồi tiếp, nơi đầu ra từ nơ-ron ở bước thời gian hiện tại sẽ được sử dụng làm đầu vào cho các bước thời gian tiếp theo.

a. Các thành phần cơ bản của RNN:

Lớp đầu vào (Input Layer):

Mỗi bước trong chuỗi thời gian có một đầu vào, thường là các từ hoặc ký tự trong chuỗi văn bản, hoặc các giá trị trong chuỗi thời gian.

Lớp ẩn (Hidden Layer):

Đây là nơi thông tin từ bước thời gian hiện tại và từ trạng thái ẩn trước đó được kết hợp để tạo ra trạng thái ẩn mới và đầu ra tại mỗi bước. Tại mỗi bước thời gian, RNN tính toán trạng thái ẩn mới dựa trên đầu vào hiện tại và trạng thái ẩn từ bước trước đó:

$$h_t = f(W_h \cdot h_{t-1} + W_x \cdot x_t + b_h)$$

Trong đó:

h_t là trạng thái ẩn tại thời điểm *t*.

h_{t-1} là trạng thái ẩn từ bước trước.

x_t là đầu vào tại thời điểm *t*.

W_h và W_x là trọng số tương ứng.

b_h là bias.

f là hàm kích hoạt, thường là hàm *sigmoid* hoặc *tanh*.

Lớp đầu ra (Output Layer):

Sau khi xử lý dữ liệu qua các bước thời gian, RNN sẽ đưa ra dự đoán cho mỗi bước dựa trên trạng thái ẩn tại thời điểm đó hoặc dự đoán cuối cùng của toàn bộ chuỗi.

4.5.2. Nguyên lý hoạt động

a. Lan truyền tiến (Forward Propagation)

Trong RNN, đầu vào được xử lý theo thứ tự tuần tự. Ở mỗi bước thời gian *t*, RNN nhận đầu vào x_t và tính toán trạng thái ẩn h_t dựa trên thông tin từ bước thời gian trước đó h_{t-1} và đầu vào hiện tại x_t . Sau đó, mô hình sẽ dự đoán đầu ra y_t tại thời điểm *t* bằng cách sử dụng trạng thái ẩn h_t.

b. Lan truyền ngược qua thời gian (Backpropagation Through Time - BPTT)

RNN được huấn luyện bằng cách sử dụng quá trình lan truyền ngược qua thời gian (BPTT), trong đó gradient của hàm mất mát được tính toán và lan truyền ngược

qua chuỗi thời gian để điều chỉnh các trọng số. Tuy nhiên, trong quá trình lan truyền ngược qua nhiều bước thời gian, RNN thường gặp phải vấn đề vanishing gradient hoặc exploding gradient, làm cho việc học các phụ thuộc dài hạn trở nên khó khăn.

4.5.3. Vấn đề vanishing gradient trong RNN

Vấn đề vanishing gradient là một hạn chế lớn của RNN khi làm việc với các chuỗi dữ liệu dài. Khi gradient lan truyền ngược qua nhiều bước thời gian, nó có thể trở nên rất nhỏ, khiến cho mô hình không học được các mối quan hệ dài hạn trong dữ liệu. Điều này dẫn đến việc RNN không hiệu quả trong việc xử lý các bài toán yêu cầu lưu giữ thông tin trong thời gian dài, chẳng hạn như dịch ngôn ngữ hoặc phân tích chuỗi dài.

Để giải quyết vấn đề này, các biến thể như LSTM (Long Short-Term Memory) và GRU (Gated Recurrent Unit) đã được phát triển, giúp cải thiện khả năng ghi nhớ thông tin dài hạn.

4.5.4. Hàm kích hoạt (Activation Functions)

Trong RNN, các hàm kích hoạt thường được sử dụng để tính toán trạng thái ẩn và đầu ra:

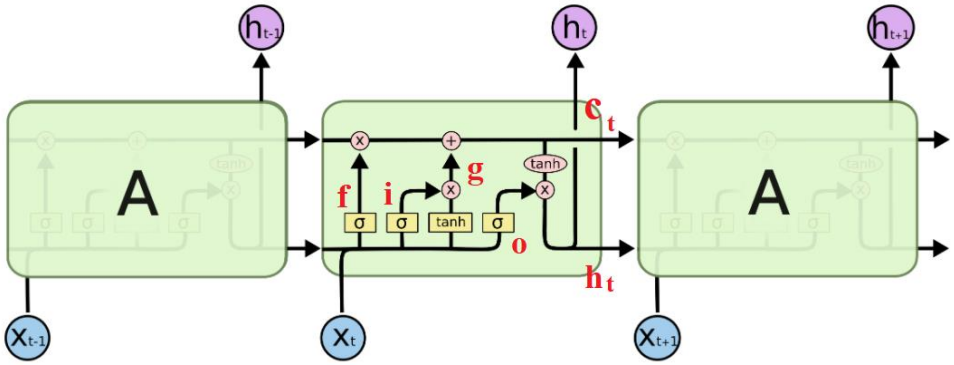
Tanh: Hàm tanh là hàm kích hoạt phổ biến trong RNN, chuyển đổi đầu ra thành một giá trị trong khoảng [-1, 1]. Điều này giúp trạng thái ẩn có thể nhận cả giá trị dương và âm, giúp cân bằng dữ liệu.

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

Sigmoid: Hàm sigmoid thường được sử dụng cho các bài toán phân loại, đầu ra của nó nằm trong khoảng [0, 1].

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

4.6.Bộ nhớ dài-ngắn hạn (Long Short-Term Memory – LSTM)



Hình 8. Cấu trúc của một mô hình LSTM [18]

LSTM là một loại mạng nơ-ron hồi tiếp (RNN), được thiết kế để khắc phục nhược điểm của RNN truyền thống trong việc ghi nhớ các thông tin dài hạn trong chuỗi dữ liệu. LSTM đặc biệt hiệu quả trong các bài toán liên quan đến dữ liệu tuần tự như văn bản, âm thanh, chuỗi thời gian và dịch ngôn ngữ, nhờ khả năng lưu giữ và xử lý thông tin từ nhiều bước trước đó.

4.6.1. Cấu trúc

LSTM có cấu trúc phức tạp hơn so với RNN thông thường nhờ vào việc sử dụng các "cổng" (gates) để kiểm soát luồng thông tin qua các ô nhớ (memory cells). Các cổng này giúp mô hình có khả năng quyết định thông tin nào nên được giữ lại, thông tin nào nên quên đi, và thông tin nào nên được thêm vào trong quá trình xử lý chuỗi dữ liệu.

LSTM bao gồm 3 loại cổng chính:

Cổng quên (Forget Gate): Quyết định thông tin nào từ trạng thái trước đó của ô nhớ cần được quên đi.

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

Trong đó:

f_t là đầu ra của cổng quên.

W_f và b_f lần lượt là trọng số và bias của cổng quên.

h_{t-1} là trạng thái ẩn từ bước trước, và x_t là đầu vào hiện tại.

σ là hàm *sigmoid*.

Cổng nhập (Input Gate): Quyết định thông tin mới nào sẽ được thêm vào ô nhớ.

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

Sau đó, một trạng thái ứng cử viên ($\tilde{C}_t$) sẽ được tính toán:

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_C)$$

Trong đó:

i_t là đầu ra của cổng nhập.

$\tilde{C}_t$ là trạng thái mới tiềm năng cho ô nhớ.

Hàm tanh đưa trạng thái mới về khoảng giá trị từ -1 đến 1.

Cổng đầu ra (Output Gate): Quyết định thông tin nào từ ô nhớ sẽ được đưa ra làm đầu ra của LSTM.

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

Trạng thái ẩn h_t tại thời điểm *t* sẽ được tính toán bằng cách kết hợp thông tin từ cổng đầu ra và trạng thái ô nhớ mới:

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

Cập nhật ô nhớ (Cell State Update): Trạng thái ô nhớ C_t được cập nhật theo công thức sau:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Trong đó, C_{t-1} là trạng thái ô nhớ từ bước trước đó.

4.6.2. Nguyên lý hoạt động

Forward Propagation (Lan truyền tiến): Dữ liệu được truyền qua từng bước thời gian, trong đó các trạng thái ô nhớ C_t và trạng thái ẩn h_t được cập nhật dựa trên đầu vào hiện tại và các trạng thái trước đó.

Backpropagation Through Time (BPTT - Lan truyền ngược theo thời gian): LSTM học bằng cách tính toán gradient đối với các trọng số thông qua quy trình backpropagation. Tuy nhiên, nhờ cấu trúc đặc biệt của LSTM với các cổng, nó giảm thiểu vấn đề vanishing gradient, giúp mô hình học các mối quan hệ dài hạn trong chuỗi.

4.7. Mạng Nơ-ron truyền thẳng (Feedforward Neural Network)

Cấu trúc cơ bản của FNN

Lớp đầu vào (Input Layer):

Nhận các đặc trưng từ dữ liệu đầu vào.

Ví dụ: Một mảng giá trị số hoặc vector từ dữ liệu (như vector hóa văn bản hoặc pixel ảnh).

Các lớp ẩn (Hidden Layers):

Bao gồm nhiều nơ-ron, mỗi nơ-ron nhận tín hiệu từ các nơ-ron ở lớp trước đó, thực hiện phép tính (thường là tích phân tuyến tính, sau đó áp dụng hàm kích hoạt).

Số lớp ẩn và số nơ-ron trong mỗi lớp là các siêu tham số bạn có thể điều chỉnh.

Lớp đầu ra (Output Layer):

Dựa vào bài toán cụ thể:

Phân loại: Số nơ-ron tương ứng với số lớp (kết hợp với hàm kích hoạt như *softmax* hoặc *sigmoid*).

Hồi quy: Một nơ-ron duy nhất (kết hợp với hàm kích hoạt tuyến tính hoặc không kích hoạt).

Quy trình hoạt động

Truyền thẳng (Forward Propagation): Dữ liệu được truyền qua từng lớp từ đầu vào đến đầu ra. Tại mỗi nơ-ron, tính toán được thực hiện theo công thức:

$$z = W.x + b$$

Trong đó:

- W : Ma trận trọng số.
- x : Đầu vào.
- b : Hệ số bias.
- z : Đầu ra trước khi áp dụng hàm kích hoạt.
- Hàm kích hoạt (Activation Function) như *ReLU, Sigmoid*, hoặc *Softmax* được áp dụng lên z.

- **Học (Learning):** Sử dụng thuật toán lan truyền ngược (*Backpropagation*) để điều chỉnh trọng số WWW và bias bbb, với mục tiêu giảm thiểu hàm mất mát (*loss function*).

1.Đặc điểm của FNN

- **Đơn giản**: Không có vòng lặp, nên không thể xử lý dữ liệu có mối liên hệ tuần tự như chuỗi thời gian (đó là nhiệm vụ của RNN).
- **Ứng dụng chính**:
    - Phân loại hình ảnh.
    - Nhận dạng giọng nói.
    - Dự đoán giá trị (hồi quy).

CHƯƠNG 3. PHÂN LOẠI THƯ RÁC

1. Tập dữ liệu thư rác

Tập dữ liệu Spam Mails Dataset được đăng tải trên Kaggle [10] sẽ được sử dụng để so sánh các mô hình DL trong bài nghiên cứu này.

Tập dữ liệu chứa: tổng 5171 emails

- 4993 email không trùng lập.
- Các email được chia thành 2 lớp:
    - Spam: được đánh số 1 và chiếm 29% tổng số email.
    - Ham: được đánh số 0 và chiếm 71% tổng số email.

Tập dữ liệu sẽ được chia thành 2 tập dữ liệu nhỏ nhờ thư viện *sklearn* với:

- Tập huấn luyện: chiếm 80%.
- Tập kiểm thử: chiếm 20%.

Tập dữ liệu có tỉ lệ Ham nhiều hơn gấp 2 lần Spam, ta có thể nói tập dữ liệu trên là không căn bằng và cần được qua xử lý đê model có thể hoạt động tốt hơn. Trong bài nghiên cứu này phương pháp sẽ được sử dụng là SMOTE vì tính dễ triển khai, tăng độ đa dạng cho tập dữ liệu và giảm nguy cơ overfitting trong quá trình huấn luyện.

2. Dữ liệu mẫu

Mẫu của một số email được lấy từ tập dữ liệu:

Bảng 1. Mẫu email có trong tập dữ liệu

STT	Nội dung	Nhãn	Số Nhãn
1	"Subject: enron methanol ; meter # : 988291\r\nthis is a follow up to the note i gave you on monday , 4 / 3 / 00 { preliminary\r\nflow data provided by daren } .\r\nplease override pop ' s daily volume { presently zero } to reflect daily\r\nactivity you can obtain from gas control .\r\nthis change is needed asap for economics purposes ."	ham	0
2	'Subject: hpl nom for january 9 , 2001\r\n(see attached file : hplnol 09 . xls)\r\n- hplnol 09 . xls'	ham	0

3	'Subject: photoshop , windows , office . cheap . main trending\r\nabasements darer prudently fortuitous undergone\r\nlighthearted charm orinoco taster\r\nrailroad affluent pornographic cuvier\r\nirvin parkhouse blameworthy chlorophyll\r\nrobed diagrammatic fogarty clears bayda\r\ninconveniencing managing represented smartness hashish\r\nacademies shareholders unload badness\r\ndanielson pure caffein\r\nspaniard chargeable levin\r\n'	spam	1
4	'Subject: ehronline web address change\r\nthis message is intended for ehronline users only .\r\ndue to a recent change to ehronline , the url (aka " web address ") for accessing ehronline needs to be changed on your computer . the change involves adding the letter " s " to the " http " reference in the url . the url for accessing ehronline should be : https : / / ehronline . enron . com .\r\nthis change should be made by those who have added the url as a favorite on the browser .'	ham	0
5	'Subject: noms / actual flow for 2 / 26\r\nwe agree\r\n- forwarded by melissa jones / texas utilities on\r\n02 / 27 / 2001\r\n10 : 33 am -\r\n" eileen ponton " on 02 / 27 / 2001 09 : 46 : 26 am\r\nto : david avila / lsp / enserch / us @ tu , charlie stone / texas utilities @ tu , melissa\r\njones / texas utilities @ tu , hpl . scheduling @ enron . com ,\r\nliz . bellamy @ enron . com\r\ncc :\r\nsubject : noms / actual flow for 2 / 26\r\ndate nom flow - mcf flow - mmbtu\r\n2 / 26 / 01 0 456 469\r\nbtu = 1 . 027'	ham	0

3. Kiến trúc

3.1. CNN

Bảng 2.Các thông số của mô hình CNN

Layer (Type)	Output Shape	Param #
conv1d (Conv1D)	(None, 4998, 32)	128
max_pooling1d (MaxPooling1D)	(None, 2499, 32)	0
dropout (Dropout)	(None, 2499, 32)	0
conv1d_1 (Conv1D)	(None, 2497, 64)	6,208
max_pooling1d_1 (MaxPooling1D)	(None, 1248, 64)	0
dropout_1 (Dropout)	(None, 1248, 64)	0
flatten (Flatten)	(None, 79872)	0
dense (Dense)	(None, 128)	10,223,744
dense_1 (Dense)	(None, 1)	129
Total params: 10,230,209 (39.03 MB) **Trainable params: 10,230,209 (39.03 MB)** **Non-trainable params: 0 (0.00 B)**		

3.2.RNN

Layer (Type)	Output Shape	Param #
reshape (Reshape)	(None, 100, 1)	0
conv1D (Conv1D)	(None, 98, 32)	128
max_pooling1d (MaxPooling1D)	(None, 49, 32)	0
flatten (Flatten)	(None, 1568)	0
dense (Dense)	(None, 64)	100, 416
dense_1 (Dense)	(None, 1)	65
Total params: 301,829 (1.15 MB) **Trainable params:** 100,609 (393.00 KB) **Non-trainable params:** 0 (0.00 B) **Optimizer params:** 201,220 (786.02 KB)		

3.3.LSTM

Layer (Type)	Output Shape	Param #
embedding_437 (Embedding)	(None, 100, 100)	4,224,300
lstm_437 (LSTM)	(None, 64)	42,240
Dense_437	(None, 49, 32)	65
Total params: 12,799,817 (48.83 MB) **Trainable params:** 4,266,605 (16.28 MB) **Non-trainable params:** 0 (0.00 B) **Optimizer params:** 8,533,212 (32.55 MB)		

3.4. FNN

Layer (Type)	Output Shape	Param #
dense_5 (Dense)	(None, 128)	640, 128
batch_normalization_4 (BatchNormalization)	(None, 128)	512
dropout_4 (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 128)	16,512
batch_normalization_5 (BatchNormalization)	(None, 128)	512

dropout_5 (Dropout)	(None, 128)	0
dense_7 (Dense)	(None, 64)	8,256
batch_normalization_6 (BatchNormalization)	(None, 64)	256
dropout_6 (Dropout)	(None, 64)	0
dense_8 (Dense)	(None, 32)	2,080
batch_normalization_7 (BatchNormalization)	(None, 32)	128
dropout_7 (Dropout)	(None, 32)	
dense_9 (Dense)	(None, 1)	33
Total params: 668,417 (2.55 MB) **Trainable params: 667,713 (2.55 MB)** **Non-trainable params: 704 (2.75 KB)** **Non-trainable params: 0 (0.00 B)**		

CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU

1. Thực nghiệm và kết quả

1.1. Môi trường

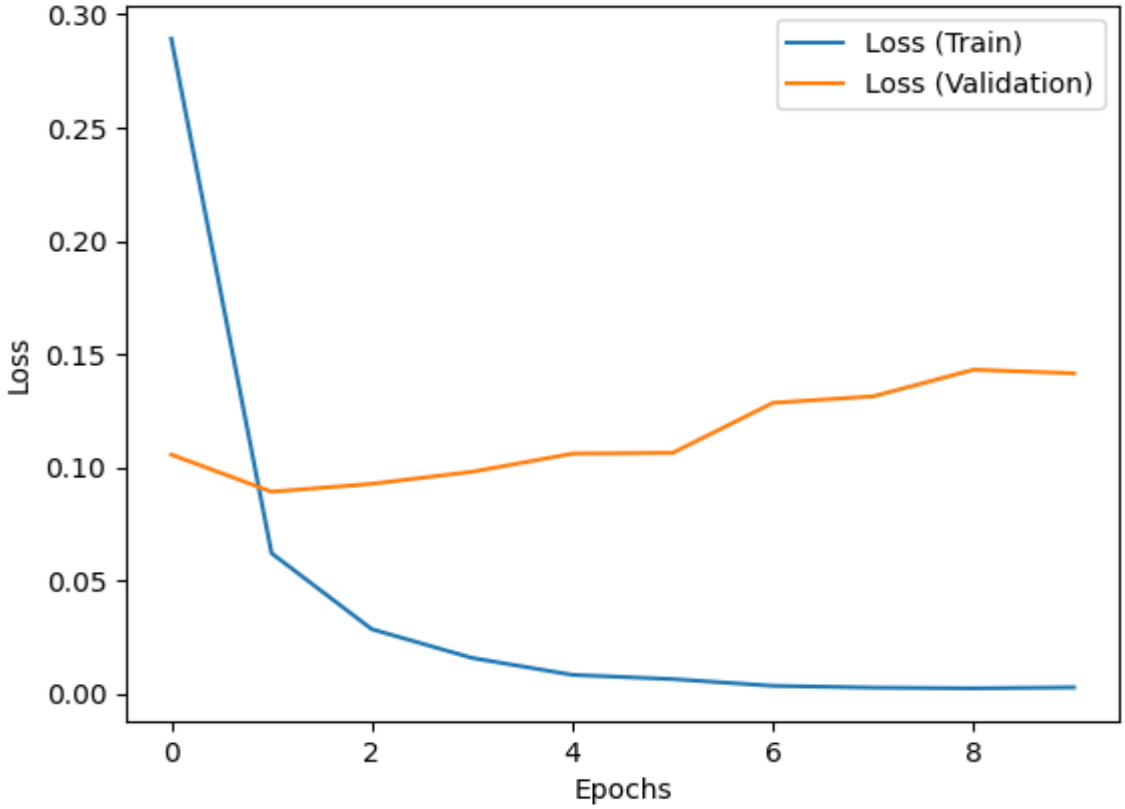
- Được phát triển trên môi trường Google colab (Python 3.5)
- Framework: dùng framework Tensorflow để phát triển các mô hình.

1.2. Thực nghiệm

- Tỷ lệ dùng training và testing: 80% và 20%.

1.2.1. CNN

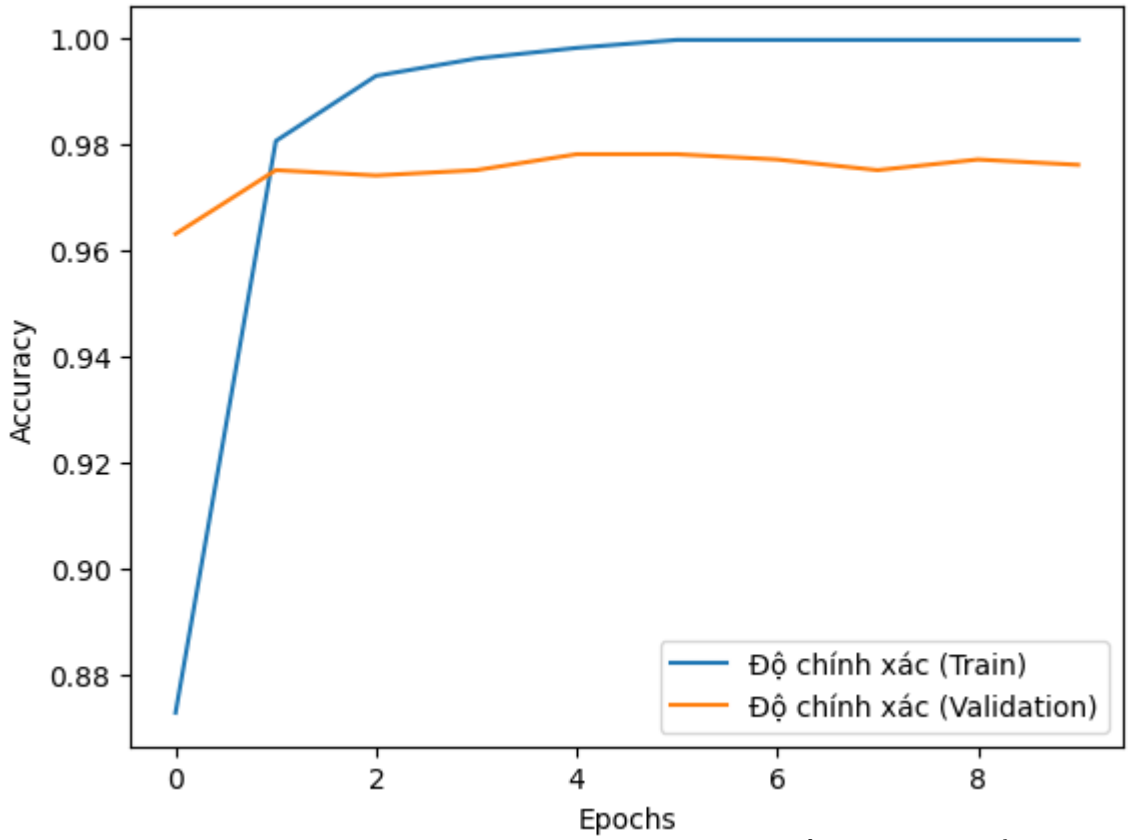
Quá trình giảm lỗi của mô hình trong quá trình huấn luyện và kiểm thử



Hình 9. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử

Độ lỗi của mô hình giảm khá nhanh chỉ qua vài vòng lập thì độ lỗi từ 3 xuống gần bằng 0 ở huấn luyện và chỉ ở mức 0.1 – 0.15 ở kiểm thử cho thấy độ hội tụ nhanh của mô hình.

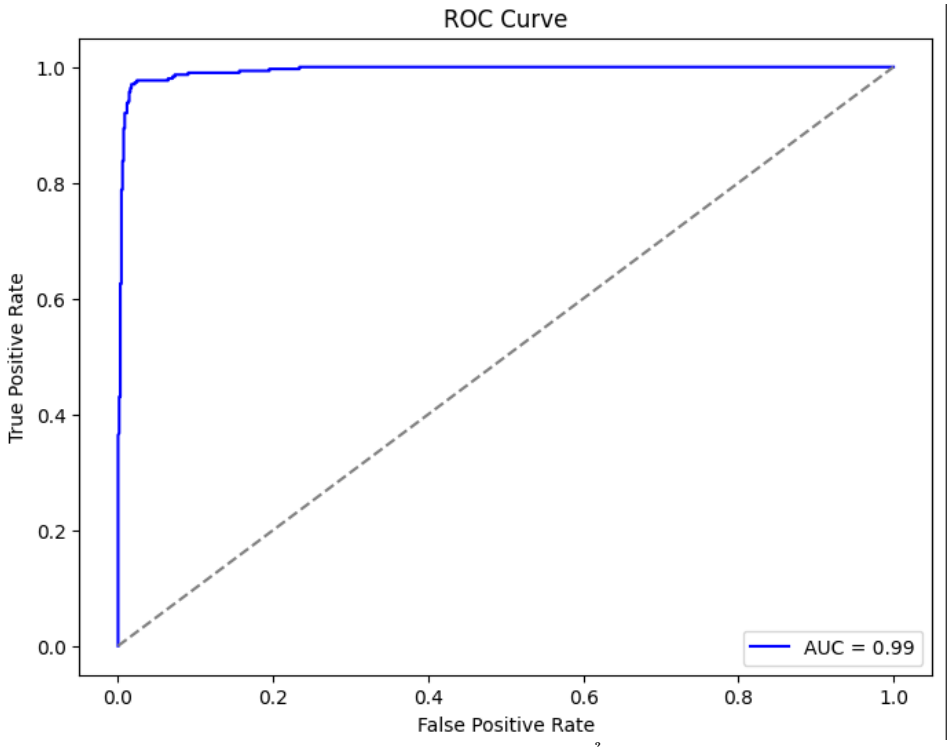
Quá trình tăng độ chính xác trong quá trình huấn luyện và kiểm thử



Hình 10. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử

Độ chính xáccủa mô hình giảm tốt chỉ trong vòng vài vòng lập thì độ chính xác từ 0.88 tăng đến 1 ở huấn luyện và ở mức 0.96 – 0.97 ở kiểm thử cho khả năng học khá nhanh.

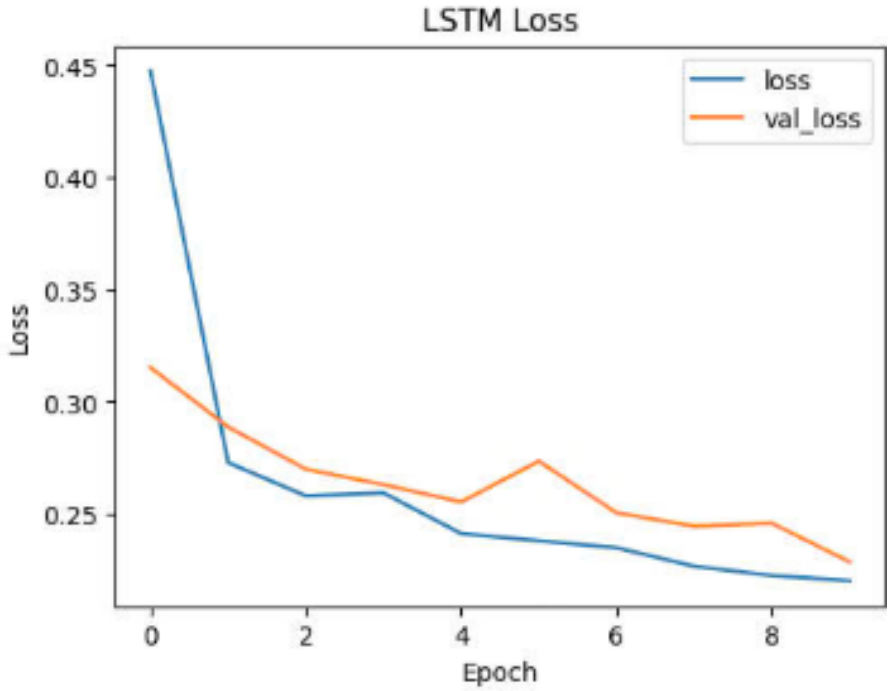
Kết quả: mô hình đạt tỉ lệ **97.6%** chính xác trên tập kiểm thử. ROC Curve và điểm AUC:



Hình 11. ROC Curve và điểm AUC

1.2.2. LSTM

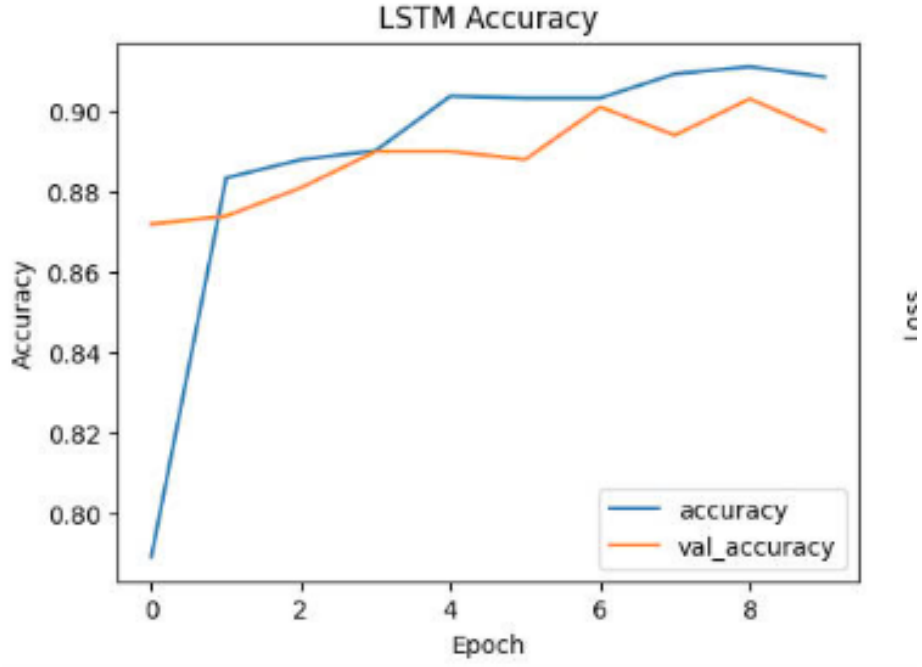
Quá trình giảm lỗi của mô hình trong quá trình huấn luyện và kiểm thử



Hình 12. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử

Độ lỗi của mô hình giảm khá nhanh chỉ qua vài vòng lập thì độ lỗi từ 4.5 xuống gần bằng 0.2 ở huấn luyện và chỉ ở mức 0.35 – 0.25 ở kiểm thử cho thấy độ hội tụ nhanh của mô hình.

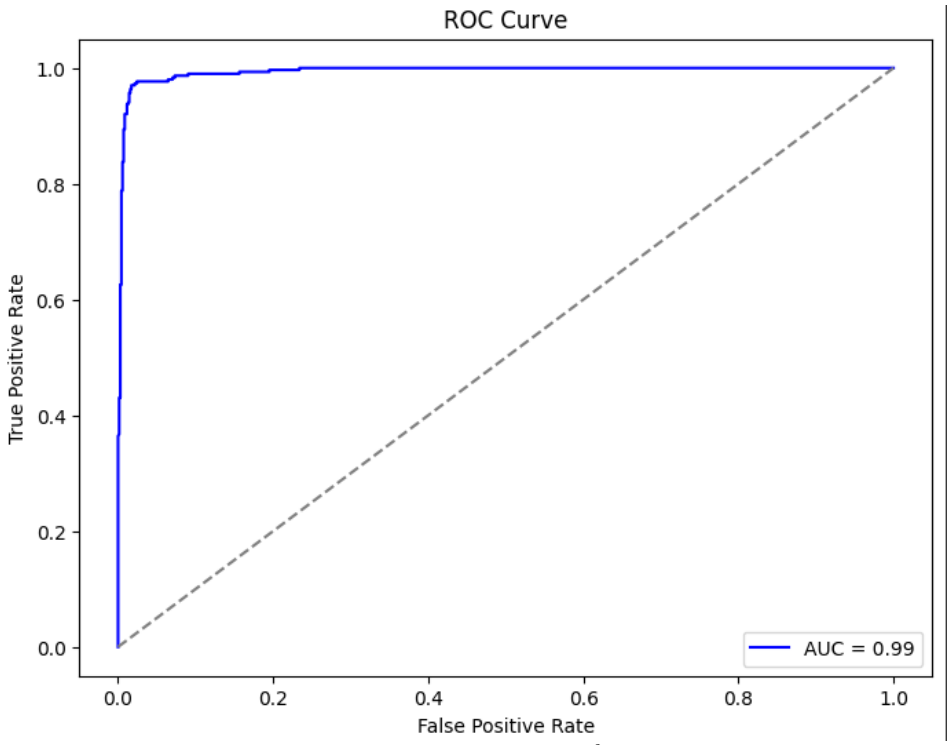
Quá trình tăng độ chính xác trong quá trình huấn luyện và kiểm thử



Hình 13. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử

Độ chính xáccủa mô hình giảm tốt chỉ trong vòng vài vòng lập thì độ chính xác từ 0.7 tăng đến 0.97 ở huấn luyện và ở mức 0.86– 0.90 ở kiểm thử cho khả năng học khá nhanh.

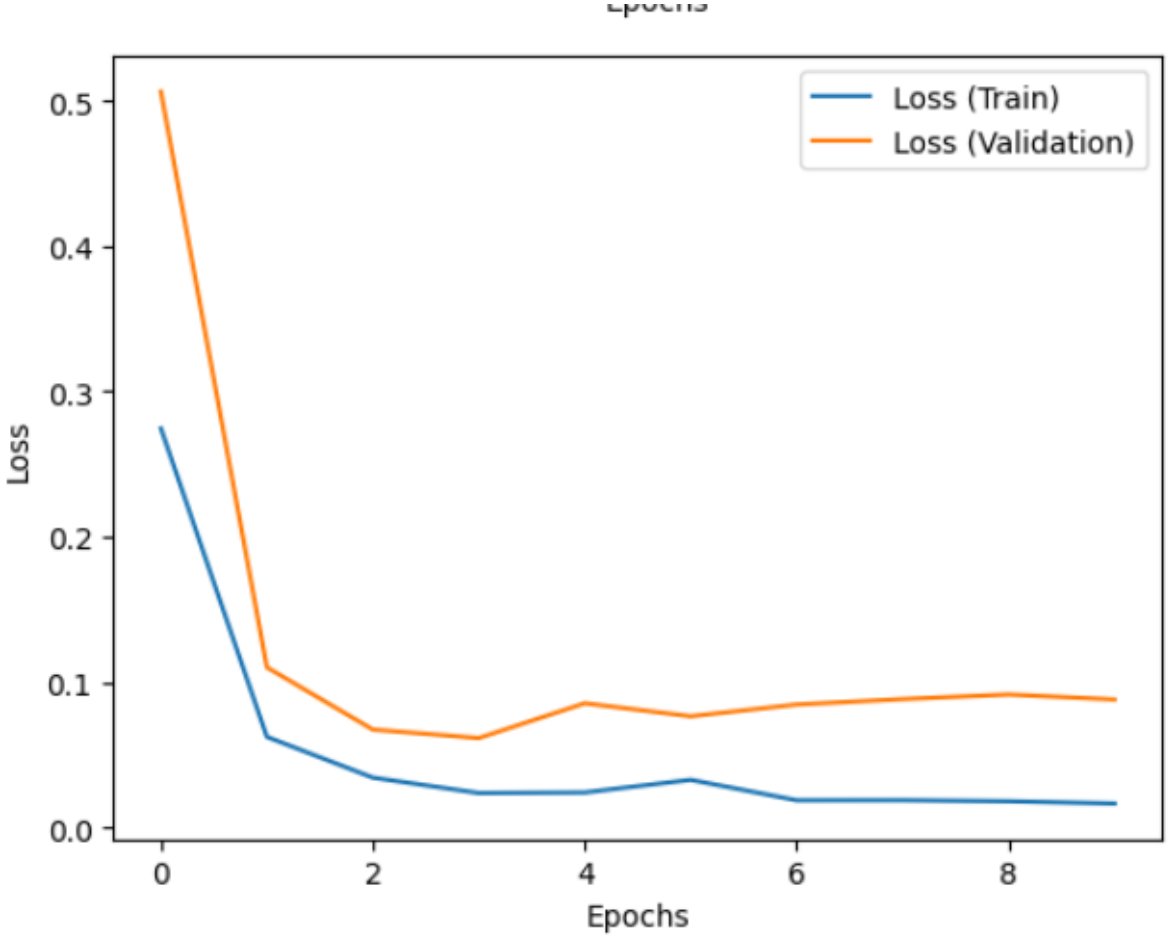
Kết quả: mô hình đạt tỉ lệ **89%** chính xác trên tập kiểm thử. ROC Curve và điểm AUC:



Hình 14. ROC Curve và điểm AUC

1.2.3. FNN

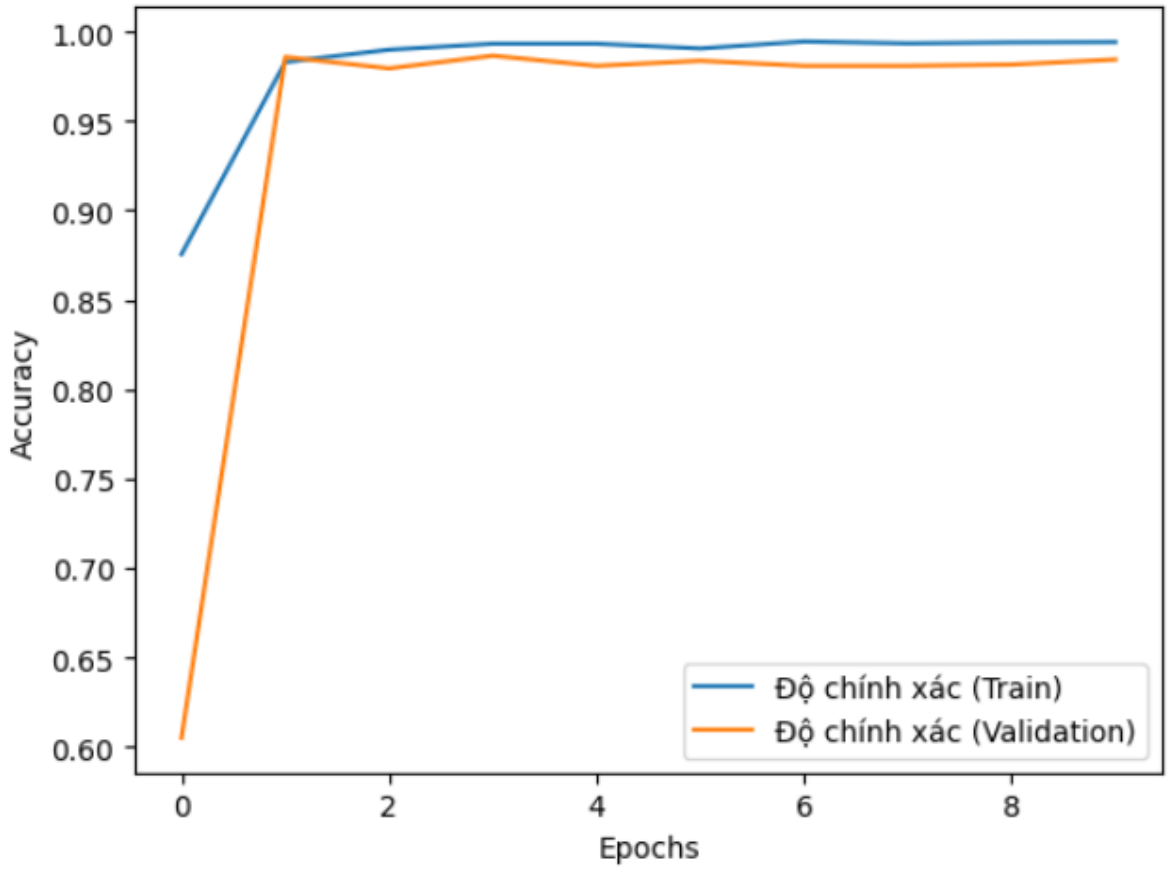
Quá trình giảm lỗi của mô hình trong quá trình huấn luyện và kiểm thử



Hình 15. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử

Độ lỗi của mô hình giảm khá nhanh chỉ qua vài vòng lập thì độ lỗi từ 0.3 xuống nhỏ hơn bằng 0.1 ở huấn luyện và chỉ ở mức 0.5 – 0.1 ở kiểm thử cho thấy độ hội tụ nhanh của mô hình.

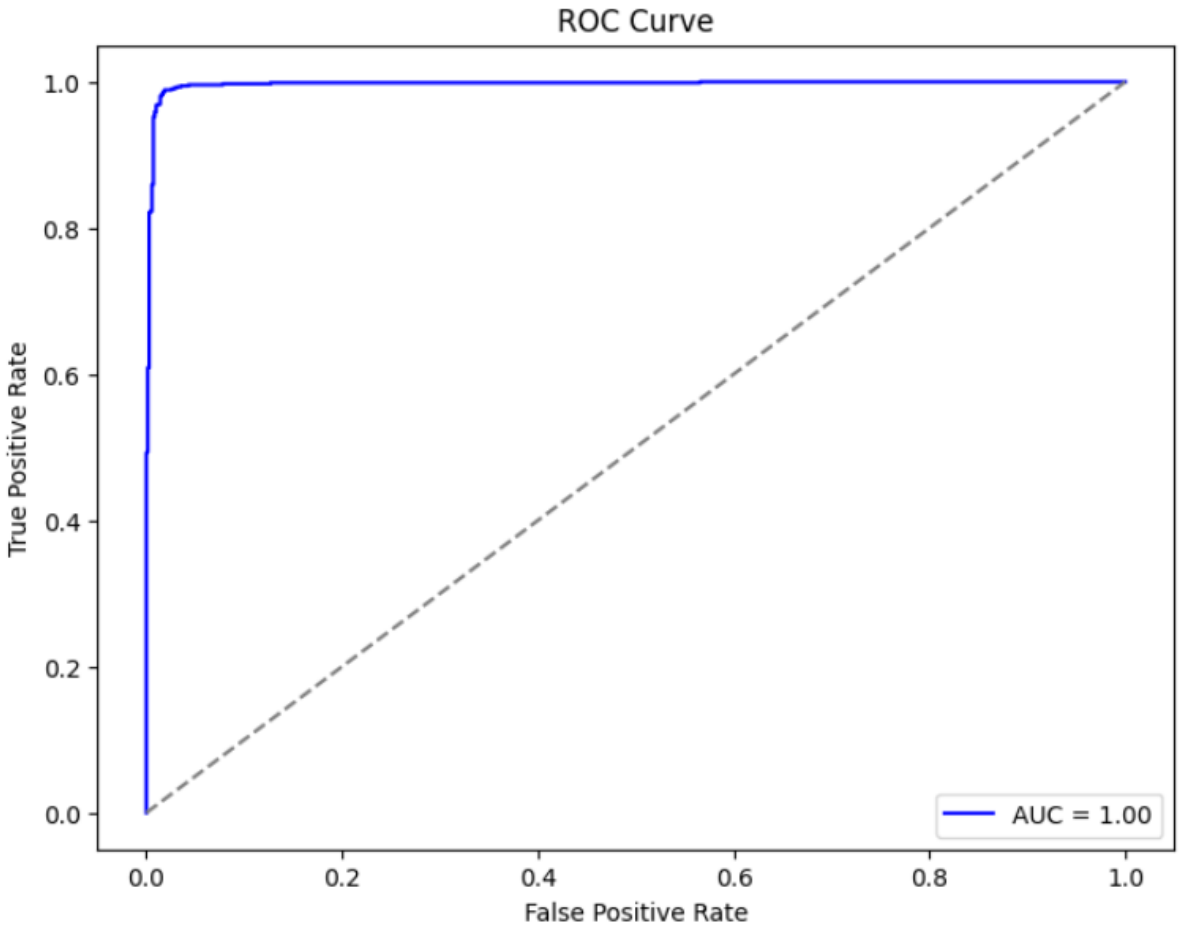
Quá trình tăng độ chính xác trong quá trình huấn luyện và kiểm thử



Hình 16. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử

Độ chính xáccủa mô hình giảm tốt chỉ trong vòng vài vòng lập thì độ chính xác từ 0.85 tăng đến 1 ở huấn luyện và ở mức 0.6– 0.98 ở kiểm thử cho khả năng học khá nhanh.

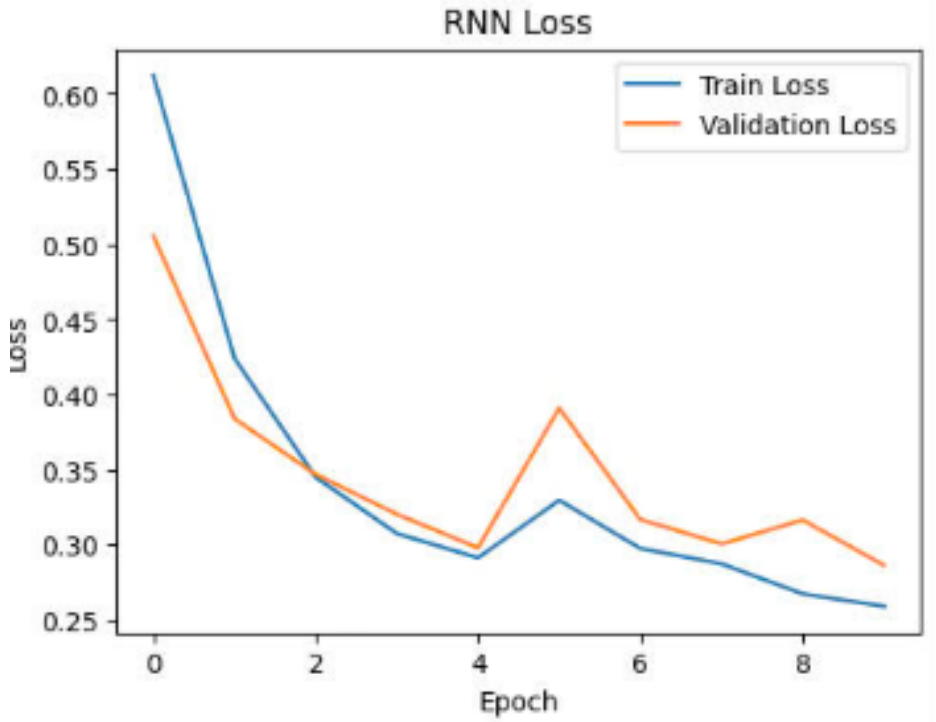
Kết quả: mô hình đạt tỉ lệ **98.44%** chính xác trên tập kiểm thử. ROC Curve và điểm AUC:



Hình 17. ROC Curve và điểm AUC

1.2.4. RNN

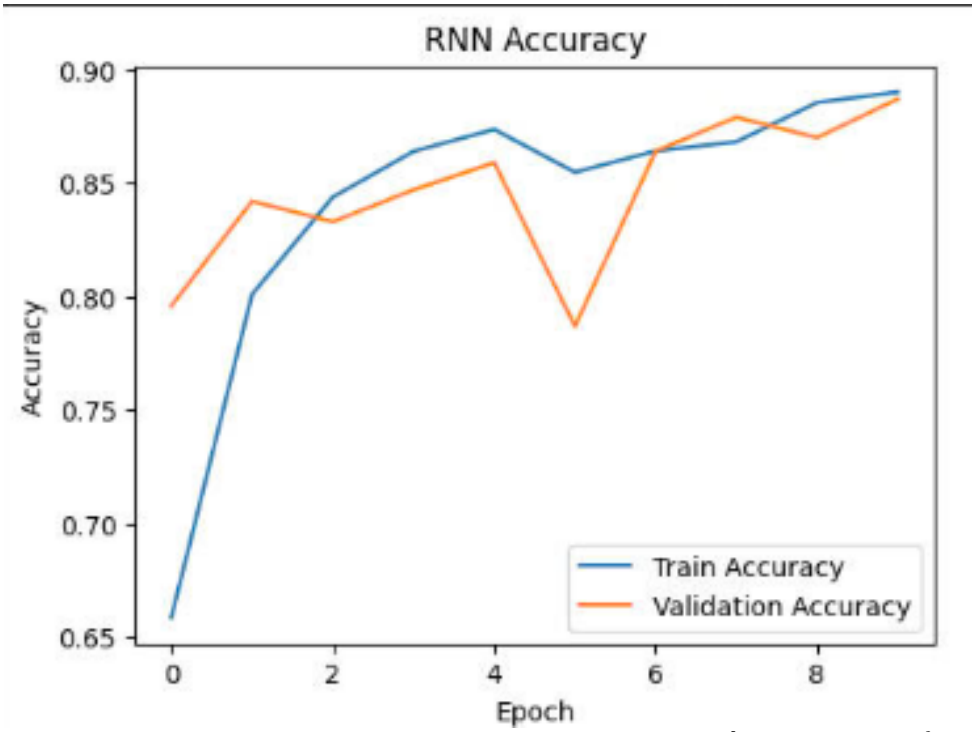
Quá trình giảm lỗi của mô hình trong quá trình huấn luyện và kiểm thử



Hình 18. Độ lỗi của mô hình trong quá trình huấn luyện và kiểm thử

Độ lỗi của mô hình giảm khá nhanh chỉ qua vài vòng lập thì độ lỗi từ 0.6 xuống nhỏ hơn bằng 0,3 ở huấn luyện và chỉ ở mức 0.6 – 0.25 ở kiểm thử cho thấy độ hội tụ khá nhanh của mô hình.

Quá trình tăng độ chính xác trong quá trình huấn luyện và kiểm thử



Hình 19. Độ chính xác của mô hình trong quá trình huấn luyện và kiểm thử

Độ chính xáccủa mô hình giảm tốt chỉ trong vòng vài vòng lập thì độ chính xác từ 0.65 tăng đến 0.9 ở huấn luyện và ở mức 0.8 – 0.9 ở kiểm thử cho khả năng học khá nhanh.

Kết quả: mô hình đạt tỉ lệ **89%** chính xác trên tập kiểm thử.

1.3. Kết quả

Mô hình	Độ chính xác	Điểm F1
FNN	98.44%	0.98(1), 0.98(0)
CNN	97.6%	
RNN	89%	0.92 (0), 0.83 (1)
LSTM	89%	0.92 (0), 0.83 (1)

Kết luận:

Nhìn chung, FNN là mô hình có hiệu suất cao nhất trong bài toán phân loại email rác, với độ chính xác và điểm F1 vượt trội hơn so với các mô hình khác. Mặc dù các mô hình như CNN, RNN, và LSTM có khả năng xử lý thông tin tuần tự tốt hơn, FNN vẫn cho thấy hiệu quả xuất sắc nhờ vào cấu trúc mạng phẳng, tính đơn giản và khả năng học các đặc trưng cục bộ trong dữ liệu. Điều này cho thấy rằng trong bài toán phân loại văn bản này, việc sử dụng mô hình FNN là một lựa chọn hợp lý và hiệu quả.

Hướng phát triển:

Các mô hình còn được triển khai sơ sài và sẽ được sửa đổi trong tương lai:

- Tối ưu hóa siêu tham số: Sử dụng Grid Search và Randomized Search để tối ưu các siêu tham số như tỷ lệ học, số lớp, số nút trong mỗi lớp.
- Ứng dụng mô hình hiện đại: Sử dụng BERT và các biến thể để cải thiện hiệu suất xử lý ngôn ngữ tự nhiên.
- Đào tạo trên dữ liệu lớn: Mở rộng và đào tạo mô hình trên dữ liệu đa dạng và lớn hơn.
- Tiền xử lý dữ liệu: Cải thiện các kỹ thuật tiền xử lý như sử dụng Word2Vec và GloVe.

TÀI LIỆU THAM KHẢO

[1] [Online] Number of sent and received e-mails per day worldwide from 2017 to 2026. Tại: https://www.statista.com/statistics/456500/daily-number-of-e-mails-worldwide/

[2] [Online] Global spam volume as percentage of total e-mail traffic from 2011 to 2023. Tại: https://www.statista.com/statistics/420400/spam-email-traffic-share-annual/

[3] Sanaa Kaddoura , Omar Alfandi, Nadia Dahmani. "*A Spam Email Detection Mechanism for English Language Text Emails Using Deep Learning Approach*". 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE).

[4] Isra'a AbdulNabi , Qussai Yaseen. *"Spam Email Detection Using Deep Learning"*. The 2nd International Workshop on Data-Driven Security (DDSW 2021) March 23 - 26, 2021.

[5] Souad Larabi-Marie-Sainte, Sanaa Ghouzali, Tanzila Saba, Linah Aburahmah, Rana Almohaini. "*Improving spam email detection using deep recurrent neural network*". Indonesian Journal of Electrical Engineering and Computer Science Vol. 25, No. 3, March 2022, pp. 1625~1633 ISSN: 2502-4752.

[6] S.M.Abdulhamid, M.Shuaib, O.Osho, I. Ismaila, and J.K.Alhassan, "*Comparative Analysis of Classification Algorithms for Email Spam Detection*," International Journal of Computer Network and Information Security, vol. 10, no.1, pp.60–67, Jan. 2018, doi: 10.5815/ijcnis.2018.01.07.

[7] Ala Mughaid, Shadi AlZu'bi, Adnan Hnaif, Salah Taamneh, Asma Alnajjar, Esraa Abu Elsoud. "*An intelligent cyber security phishing detection system using deep learning techniques*". Cluster Computing (2022) 25:3819–3828.

[8] Khan Farhan Rafat, Qin Xin, Abdul Rehman Javed, Zunera Jalil and Rana Zeeshan, Ahmad. "*Evading obscure communication from spam emails*". Mathematical Biosciences and Engineering (2021) Volume 19, Issue 2, 1926–1943.

[9] Mohammad Alauthman. "*Botnet Spam E-Mail Detection Using Deep Recurrent Neural Network*". International Journal of Emerging Trends in Engineering Research Volume 8. No. 5, May 2020. ISSN 2347 – 3983.

[10] [Kaggle] Spam Mails Dataset (enron-1 folder of Spam Dataset). Tại: https://www.kaggle.com/datasets/venky73/spam-mails-dataset/data

[11] [Online] [Image] Deep Learning vs. Machine Learning – What's The Difference?. Nguồn tại: https://www.softwaretestinghelp.com/data-mining-vs-machine-learning-vs-ai/

[12] [Wikipedia] Natural Language Processing.

Tại: https://en.wikipedia.org/wiki/Natural_language_processing

[13] [Wikipedia] Machine Learning.

Tại: https://en.wikipedia.org/wiki/Machine_learning

[14] [Online][Image] Nguồn tại: https://askabiologist.asu.edu/neuron-anatomy

[15] [Online][Image] Nguồn tại: https://en.wikipedia.org/wiki/Neural_network (machine_learning)

[16] [Online][Image] Nguồn tại: https://www.smartboost.com/blog/deep-learning-vs-neural-network-whats-the-difference/

[17] [Online][Image] Nguồn tại: https://machine-learning-company.nl/en /technical /convolutional-neural-network-text-classification-with-risk-assessment-eng/

[18] [Online][Image] Nguồn tại: https://viblo.asia/p/recurrent-neural-network-tu-rnn-den-lstm-gGJ597z1ZX2