**ĐẠI HỌC QUỐC GIA HÀ NỘI**



**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**---∞0∞---**

BÁO CÁO CUỐI KÌ

**Môn Học: Khai phá dữ liệu**

**Đề Tài: Spam Comments Detection**

Giảng viên hướng dẫn: **Lê Hoàng Sơn**

Sinh viên thực hiện: Vũ Quang Hiếu

Mai Bá Đức

Nguyễn Hà Nam

Nguyễn Hữu Nghĩa

**Hà Nội – 2024**

|  |  |  |
| --- | --- | --- |
| Phân chia công việc | Sinh viên thực hiện | Xác nhận |
| Trưởng nhóm | Nguyễn Hà Nam  210002160 |  |
| Huấn luyện mô hình học sâu | Mai Bá Đức  21000265 |  |
| Thu thập dữ liệu, làm sạch dữ liệu, huấn luyện mô hình học máy | Vũ Quang Hiếu  21002146 |  |
| Giao diện | Nguyễn Hữu Nghĩa  21002163 |  |

MỤC LỤC

[I. Giới thiệu bài toán 3](#_Toc167040014)

[II. Quá trình xử lí 3](#_Toc167040015)

[1. Thu thập dữ liệu 3](#_Toc167040016)

[2. Làm sạch dữ liệu 3](#_Toc167040017)

[3. Phân tích dữ liệu 4](#_Toc167040018)

[4. Huấn luyện mô hình 6](#_Toc167040019)

[4.1 Xử lí dữ liệu 6](#_Toc167040020)

[4.2 Mô hình học máy 6](#_Toc167040021)

[*4.2.1* Random Forest 6](#_Toc167040022)

[III. Kết quả và thảo luận 12](#_Toc167040023)

[1. Phương pháp đánh giá 12](#_Toc167040024)

[2. Kết quả 13](#_Toc167040025)

[2.1 Kết quả cho mô hình học máy 13](#_Toc167040026)

[2.2 Kết quả trong mô hình học sâu 16](#_Toc167040027)

[IV. Xây dựng ứng dụng 18](#_Toc167040028)

[1. Back-end 18](#_Toc167040029)

[2. Front-end 18](#_Toc167040030)

[2.1. Công nghệ sử dụng 18](#_Toc167040031)

[2.2. Cấu trúc và tính năng 18](#_Toc167040032)

[3. Khởi động ứng dụng 19](#_Toc167040033)

[V. Định hướng phát triển 21](#_Toc167040034)

[VI. Tài liệu tham khảo 21](#_Toc167040035)

# Giới thiệu bài toán

Ngày nay, đi kèm với sự phát triển của công nghệ và lượng người dùng tăng chóng mặt của các mạng xã hội là sự xuất hiện của các “comment spam”. Đem lại nhiều phiền toái cho người dùng, ảnh hưởng tiêu cực đến các mạng xã hội. Với ý tưởng lọc bớt các “comment spam” trên các mạng xã hội, chúng em đã xây dựng một ứng dụng dự đoán “comment spam” bằng các mô hình học máy và mô hình học sâu. Kết hợp với FastAPI và thư viện React tạo thành 1 ứng dụng phát hiện “comment spam”.

# Quá trình xử lí

Nhóm chúng em đã tiến hành một nghiên cứu nhằm phân tích dữ liệu và xây dựng các mô hình dự đoán hiệu quả. Quá trình của chúng tôi bao gồm thu thập và làm sạch dữ liệu, sau đó áp dụng các phương pháp phân tích và huấn luyện mô hình đa dạng, từ phương pháp Ensemble bao gồm Decision Tree, Logistic Regression đến Support Vector Machine, cùng với hai mô hình học sâu LSTM và GRU. Bằng cách này, chúng em hy vọng đem lại những hiểu biết và dự đoán chính xác trong bài toán đặt ra.

A diagram of a flowchart

Description automatically generated

## 1. Thu thập dữ liệu

Bộ dữ liệu: Gồm 5 bộ dataset ứng với 5000 bản ghi của 5 video trong 10 video có lượt xem lớn nhất năm 2015. Trong đó số bình luận được gán nhãn “Spam” là 2500 chiếm 50% còn lại là bình luận không bị coi là “Spam”. Tỉ lệ giữa bình luận spam và không spam tương đối cân bằng.

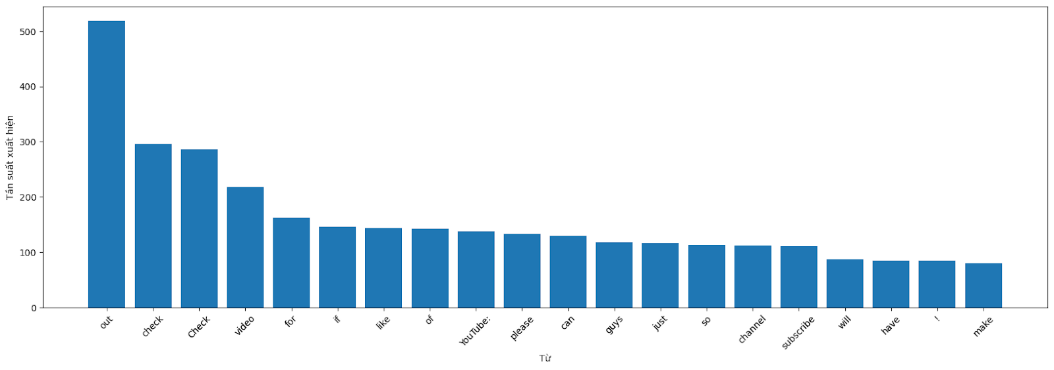
Mỗi bộ dataset bao gồm 5 thuộc tính: Comment ID, AUTHOR, DATE, CONTENT, CLASS. Trong đó Comment ID, AUTHOR, DATE lần lượt là id của bình luận, tên tài khoản bình luận và thời gian bình luận; CONTENT là nội dung người dùng bình luận, CLASS là nhãn của bình luận đó. CLASS ứng với 1 thể hiện rằng bình luận đó là spam, với 0 bình luận đó không phải spam.

## 2. Làm sạch dữ liệu

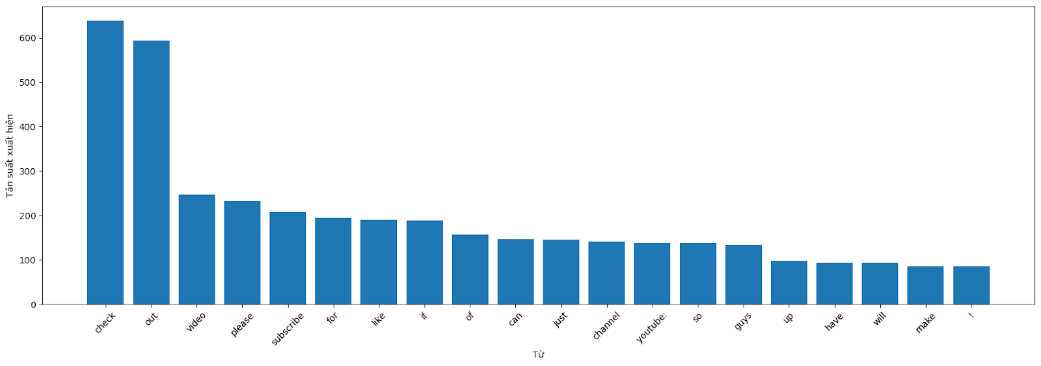
Việc làm sạch dữ liệu là bước đầu tiên trong mô hình đề xuất. Mục đích của nhiệm vụ này là loại bỏ các từ và ký tự không cần thiết khỏi các bình luận để cải thiện hiệu suất của mô hình học máy. Các thành phần cần loại bỏ ở bước này bao gồm:

* Loại bỏ các từ ‘stop word’ ra khỏi văn bản.
* Loại bất kỳ ký tự nào không phải là ký tự chữ cái, số hoặc dấu gạch dưới và cũng không phải là ký tự khoảng trắng, tức là các ký tự đặc biệt hoặc ký tự dấu câu.
* Loại bỏ các biểu tượng cảm xúc.

## 3. Phân tích dữ liệu



Plot 1: Đồ thị mật độ 20 từ xuất hiện nhiều nhất với các bình luận "Spam".



Plot 2: Đồ thị mật độ 20 từ xuất hiện nhiều nhất với các bình luận “Spam”.

A graph with numbers and lines

Description automatically generated

Plot 3: Đồ thị histogram của số lượng chữ cái trong 1 comment.

*A graph with a bar

Description automatically generated*

Plot 4: Đồ thị histogram của số lượng từ trong 1 comment.

Đồ thị 1,2 là các đồ thị đo tần suất xuất hiện của các từ trong tất cả 2034 bản ghi, trong đó đồ thị 1 không sử dụng hàm lower() để các chữ viết hoa thành viết thường, đồ thị 2 sử dụng lower để tất các các từ trong cùng một chữ trở thành chữ viết thường. Đồ thị 3,4 thể hiện tần suất xuất hiện của các từ trong 1076 bình luận được gán nhãn ‘Spam’.

Cả 4 đồ thị này đều có tỉ lệ xuất hiện rất bất thường, khi so sánh tần suất của 2 từ ‘check’ và ‘out’ với các từ còn lại; tần suất xuất hiện bắt đầu giảm mạnh ngay từ 2 điểm đầu.Khi so sánh đồ thị 1 với đồ thị 2, ta có thể thấy chỉ có thứ tự các từ là thay đổi; còn 20 từ xuất hiện nhiều nhất gần như không đổi, kết quả thu được tương tự khi ta so sánh đồ thị 3 với đồ thị 4. Khi so sánh đồ thị 1 với đồ thị 3; những từ xuất hiện nhiều bất thường như “check”, “out”, “video”, đều luôn thuộc top 3 từ xuất hiện nhiều nhất. Kết quả thu được cũng tương tự khi ta so sánh hai đồ thị 2 và 4. Từ đó có thể rút ra được rằng: Những bình luận chứa những từ xuất hiện với tần suất nhiều bất thường như “check”,”out”, “video” có khả năng rất cao là một bình luận “Spam”.

Plot 3 và 4 cho ta thấy được rằng đa số các comment đều chứa dưới 100 chữ và duới 25 từ. Cả 2 đồ thị đều có độ dốc cao. Nhưng đồ thị số lượng chữ thoải dần về sau, còn đồ thị số lượng từ lại có xu hướng tăng ở ngưỡng 75 từ, cho thấy có một số lượng các comment dùng các từ dài hoặc viết dính chữ vào nhau.

*A graph with different colored bars

Description automatically generatedA graph with different colored bars

Description automatically generated*

Plot 5: Top các bigram, trigram xuất hiện nhiều nhất trong các comment không Spam

* Nội dung: Các bigram, trigram xuất hiện trong các comment thông thường. Đa số đều liên quan đến nội dung của video, hoặc mang nội dung tích cực, có tính xây dựng cao. Nhưng vẫn có một số đoạn vô nghĩa không chứa nội dung.
* Hình dạng đồ thị: đồ thị có độ dốc cao, và càng xuống thấp đồ thị càng thoải. Cho thấy độ bất đối xứng trong phân bố của các cụm, nhóm từ.

*A graph of a number of colored bars

Description automatically generated with medium confidenceA graph of a graph showing different colored bars

Description automatically generated with medium confidence*

Plot 6: Top các bigram xuất hiện nhiều trong các comment Spam

* Nội dung: Các bigram, trigram xuất hiện trong các comment thông thường. Đa số đều không liên quan đến nội dung của video, hoặc mang nội dung tích không có tính xây dựng; Đa số nội dung đều lặp lại. Nhìn vào top 3, ta có thể dễ dàng thấy được các comment spam tập trung vào các từ ‘check’ ‘video’. Từ đấy có thể đưa ra một vài dự đoán về động cơ spam video là để tăng tương tác.
* Hình dạng đồ thị: Tương tự với các comment không spam, đồ thị của comment spam có độ dốc cao, và càng xuống thấp đồ thị càng thoải. Cho thấy độ bất đối xứng trong phân bố của các cụm, nhóm từ.  Và sự chênh lệch đáng ngờ của một số dữ liệu.

## 4. Huấn luyện mô hình

### 4.1 Xử lí dữ liệu

Word embedding là một kỹ thuật nhằm biểu diễn từ hoặc câu trong một văn bản bằng các vectơ số thực, mô tả trong một mô hình vectơ (hoặc Mô hình không gian vectơ). Các từ có ý nghĩa tương tự được ánh xạ gần nhau trong không gian vectơ. Word embedding rất hữu ích để biểu diễn các từ, để được đưa vào một mô hình Học sâu. Trong những năm gần đây, nó được coi là một trong những biểu diễn tốt nhất của từ trong Xử lý ngôn ngữ tự nhiên. Có một số mô hình nhúng từ được huấn luyện trước và sẵn sàng sử dụng, như Word2Vec và GloVe. Tuy nhiên, trong bài báo này, chúng em sử dụng GloVe cho mô hình học sâu và đối với mô hình học máy sử dụng phương pháp TF - IDF.

Trong phương pháp đề xuất, mục đích của nhúng từ là chuẩn bị dữ liệu văn bản cho mô hình học sâu. Hình 2 mô tả các bước thực hiện để hoàn thành quá trình nhúng từ. Khi xem xét rằng chúng ta có một dữ liệu đầu vào chứa một tập hợp gồm N bình luận, bước đầu tiên trong quá trình là biến đổi tập văn bản thành vectơ, bằng cách biến đổi mỗi văn bản thành một chuỗi số nguyên (mỗi số nguyên là chỉ mục của một token trong từ điển). Kết quả là một danh sách các chuỗi có kích thước biến đổi (vì các tin nhắn văn bản không có cùng độ dài). Tiếp theo, nhiệm vụ "Pad sequences" là làm cho các câu có độ dài đồng đều, bằng cách điền các câu nhỏ hơn "MAX\_SEQ\_LENGTH" bằng các giá trị trống và cắt bỏ các câu dài hơn "MAX\_SEQ\_LENGTH". Cuối cùng của quá trình này, dữ liệu văn bản đã sẵn sàng để huấn luyện với lớp nhúng từ và đưa chúng vào mô hình học sâu như đầu vào.

Hình 2: Quá trình Word Embedding


Hình 2: Quá trình Word Embedding

### 4.2 Mô hình học máy

### *4.2.1* Random Forest

Random forest là thuật toán supervised learning, có thể giải quyết cả bài toán hồi quy và phân loại. Với random forest, chúng ta sẽ xây dựng nhiều cây quyết định bằng decision tree, tuy nhiên mỗi cây quyết định sẽ có những yếu tố ngẫu nhiên và kết quả dự đoán được tổng hợp từ các cây quyết định.

##### a. Xây dựng thuật toán random forest

Giả sử bộ dữ liệu của chúng ta có n mẫu và mỗi mẫu có d thuộc tính (feature). Để xây dựng mỗi cây quyết định chúng ta sẽ làm như sau:

1. Lấy ngẫu nhiên n mẫu từ bộ dữ liệu với kĩ thuật [Bootstrapping](https://en.wikipedia.org/wiki/Bootstrapping_(statistics)), hay còn gọi là **random sampling with replacement**. Cụ thể khi lấy ra 1 mẫu dữ liệu, chúng ta không loại bỏ luôn mẫu đó khỏi không gian mẫu mà tiếp tục lặp lại cho tới khi có đủ n mẫu dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới được tạo ra có thể có những dữ liệu bị trùng nhau.
2. Sau khi sample được n dữ liệu từ bước 1, chúng ta sẽ chọn ngẫu nhiên ở k thuộc tính (k < n). Chúng ta thu được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.
3. Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Do tính chất ngẫu nhiên (random) trong quá trình lựa chọn mẫu dữ liệu nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest là tập hợp gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Khi dùng thuật toán Random Forest, chúng ta cần chú ý tới một số thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách.

##### b. Tại sao thuật toán Random Forest có thể tốt hơn so với một mô hình Decision tree thông thường

Trong thuật toán Decision Tree, khi xây dựng cây quyết định nếu để độ sâu tùy ý thì cây sẽ phân loại đúng hết các dữ liệu trong tập training, tuy nhiên lại không chắc thể hiện đúng trên tập test (high variance). Chính vì sự phức tạp trong mô hình có thể dẫn đến overfitting. Tuy nhiên, đối với thuật toán Random Forest gồm nhiều cây quyết định, mỗi cây quyết định đều có những yếu tố ngẫu nhiên:

1. Lấy ngẫu nhiên dữ liệu để xây dựng cây quyết định.
2. Lấy ngẫu nhiên các thuộc tính để xây dựng cây quyết định.

Do mỗi cây quyết định trong thuật toán Random Forest không dùng tất cả dữ liệu training, cũng như không dùng tất cả các thuộc tính của dữ liệu để xây dựng cây nên mỗi cây có thể sẽ dự đoán không tốt, khi đó mỗi mô hình cây quyết định không bị overfitting mà có thế bị underfitting, hay nói cách khác là mô hình có high bias. Tuy nhiên, kết quả cuối cùng của thuật toán Random Forest lại tổng hợp từ nhiều cây quyết định, thế nên thông tin từ các cây sẽ bổ sung thông tin cho nhau, dẫn đến mô hình có low bias và low variance, hay mô hình có kết quả dự đoán tốt.

#### 4.2.2. SVM

##### a. Giới thiệu về SVM

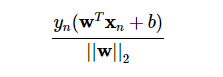
SVM là thuật toán học có giám sát, thường được dùng cho bài toán phân loại. SVM có khả năng phân loại dữ liệu có số chiều lớn. Ý tưởng của thuật toán là tìm một siêu phẳng để phân chia các điểm dữ liệu của các lớp khác nhau trong không gian.

Để tìm được siêu phẳng, chúng ta cần giải quyết bài toán tối ưu khoảng cách (margin) giữa các điểm dữ liệu của 2 lớp gần nhất. Và các điểm dữ liệu nằm gần siêu phẳng được gọi là các vector hỗ trợ.

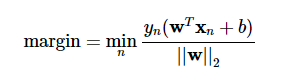
Đối với dữ liệu có phân bố tuyến tính, chúng ta có thể sử dụng mô hình linear SVM. Đối với dữ liệu có phân bố phi tuyến tính, chúng ta có thể sử dụng các mô hình SVM có kernel phi tuyến tính.

##### b. Xây dựng bài toán tối ưu

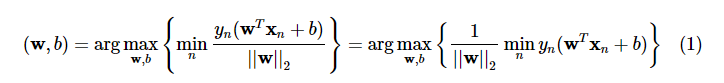
Giả sử các cặp dữ liệu của training set là (x1, y1), (x2, y2)…(xn, yn) với xi nằm trong không gian d chiều thể hiện đầu vào của dữ liệu và yi là nhãn của điểm dữ liệu đó. D là số chiều của dữ liệu và N là số điểm dữ liệu. Giả sử các nhãn yi được xác định bới yi = 1 hoặc yi = -1. Với một điểm quan sát bất kì, khoảng cách từ điểm đó tới siêu phẳng là:



Như vậy, margin được tính là khoảng cách gần nhất từ 1 điểm tới mặt phẳng đó.

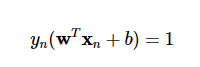


Bài toán tối ưu trong SVM là bài toán tìm **w** và **b** sao cho margin này đạt giá trị lớn nhất:



Quá trình giải bài toán này rất phức tạp, chính vì thế chúng ta sẽ đưa nó về bài toán đơn giản hơn.

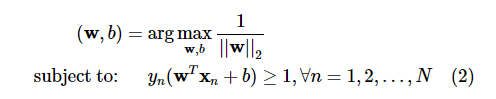
Nếu ta thay vector hệ số **w** bằng **kw**, **b** thành **kb** trong đó **k** là hằng số dương thì mặt phân chia không đổi, khoảng cách từ từng điểm đến mặt phân chia không đổi, tức margin không đổi. Dựa vào tính chất này, ta giả sử:



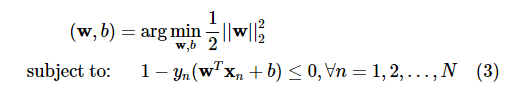
với những điểm nằm gần mặt phân chia nhất. Tổng quát với mọi n, ta có:



Ta có thể đưa bài toán tối ưu về bài toán tối ưu có ràng buộc:

****

Bằng vài phép biến đổi (nghịch đảo hàm mục tiêu, bình phương để có hàm khả vi, nhân với ½ để dễ đạo hàm hơn), ta có thể đưa về bài toán dưới đây:

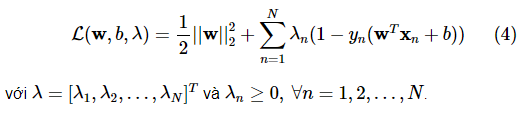


Xác định class cho điểm dữ liệu sau khi tìm được mặt phân cách **wTx + b = 0**:



##### c. Bài toán đối ngẫu SVM

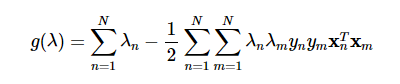
Bài toán tối ưu trên thỏa mãn điều kiện Slater, như vậy bài toán trên thỏa mãn strong duality. Ta sử dụng Lagrange cho bài toán (3):



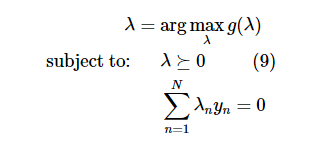
Hàm đối ngẫu Lagrange được định nghĩa như sau:



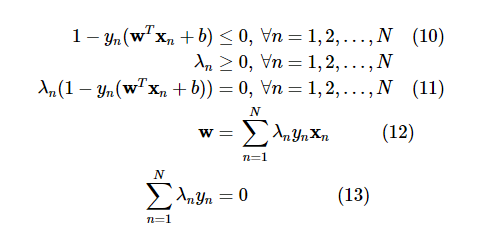
Ta có thể giải hàm trên sử dụng đạo hàm, chúng ta thu được:



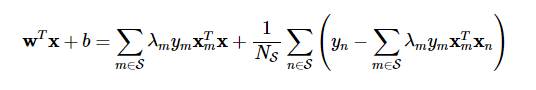
Với việc kết hợp hàm đối ngẫu Lagrange và các điều kiện ràng buộc của lambda, ta thu được bài toán đối ngẫu Lagrange:



Vì đây là một bài toán lồi và strong duality, nghiệm của bài toán sẽ thỏa mãn hệ KKT:



Bằng việc giải hệ tối ưu này, để xác định x thuộc class nào, chúng ta cần xác định dấu của biểu thức:



### 4.3 Mô hình học sâu

#### 4.3.1 Long short-term memory

Global Vector

Global vector là một trong những phương pháp mới để xây dựng vector, nó được xây dựng dựa trên Co-occurrence Matrix. GloVe có bản chất là xác suất, ý tưởng xây dựng phương pháp này đến từ tỉ số:

(1)

Trong đó là xác suất xuất hiện của từ trong ngữ cảnh của từ , tương tự với .

Công thức của :

Trong đó

: số lần xuất hiện của từ trong ngữ cảnh của từ .

: số lần xuất hiện của từ trong ngữ cảnh của toàn bộ các từ còn lại ngoại trừ .

(Các giá trị này được lấy từ Co-occurrence Matrix)

Ý tưởng của GloVe: độ tương tự ngữ nghĩa hai từ có thể xác định thông qua độ tương tự ngữ nghĩa giữa từ với mỗi từ những từ có tính xác định ngữ nghĩa tốt chính là những từ làm cho (1)1 hoặc xấp xỉ bằng 0.

GloVE khởi đầu bằng việc tìm một hàm sao cho nó ánh xạ từ các vector từ trong vùng không gian sang một giá trị tỉ lệ với (1). Sau nhiều bước đơn giản hóa với tối ưu, ta xét hàm minimum cost như sau:

Trong đó:

, : các vector từ

: các bias tương ứng

: mục tương ứng với cặp từ trong Co-occurrence Matrix.

Hàm được gọi là weighting function, được thêm vào để giảm bớt sự ảnh hưởng của các cặp từ xuất hiện quá thường xuyên, hàm thỏa mãn 3 tính chất:

* Có giới hạn tại 0
* Là hàm không giảm
* Có giá trị nhỏ khi rất lớn

Việc thực hiện minimun cost để tìm ra các vector từ bằng nhiều cách trong đó tiêu chuẩn nhất là sử dụng Gradient Descent.

## I. Học sâu

Trong bài nghiên cứu, chúng em sử dụng mô hình dựa trên học sâu để phát hiện nhận xét ‘spam’ từ các nhận xét trong một video trên ứng dụng Youtube. Để triển khai mô hình, chúng em thử nghiệm 2 kiến trúc học sâu khác nhau dựa trên việc kết hợp giữa LSTM (Long short-term memory) và CNN (Convolutional Neural Network). Hai mô hình đã mang lại cho chúng em kết quả tốt về hiệu suất.

Trong hình 1 và hình 2, chúng em mô phỏng kiến trúc của 2 mô hình CNN-LSTM và LSTM-CNN.

A diagram of a computer

Description automatically generatedA diagram of a computer

Description automatically generated

Hình 2: CNN-LSTM

Hình 1: LSTM-CNN

1. Convolution 1D

Convolution 1D

Lợi ích chính của lớp Convolution là để trích xuất các đặc điểm quan trọng từ dữ liệu văn bản. Điều này được thực hiện bằng cách áp dụng phép tích chập lên các vector từ được tạo ra bởi lớp Word Embedding. Với là vector từ có chiều của từ thứ trong nhận xét. Hãy cho là đầu vào của tin nhắn, với là độ dài của tin nhắn. Đối với mỗi vị trí trong nhận xét, xem xét một vector cửa sổ với vector từ liên tiếp, được biểu diễn như sau:

Một phép tích chập bao gồm một bộ lọc được áp dụng lên cửa sổ để tạo ra một bản đồ đặc điểm mới Mỗi phần tử đặc điểm cho vector cửa sổ được tính toán như sau:

Trong đó là tích vô hướng từng phần tử, là một thuật ngữ độ lệch, và là một hàm phi tuyến. Trong trường hợp của chúng em, chúng em đã sử dụng đơn vị tuyến tính Đơn vị Đơn vị (ReLU) như một hàm phi tuyến. Nó được định nghĩa như sau:

Hàm kích hoạt ReLU trả lại nếu giá trị là dương, nếu không trả lại 0. Đối với cấu hình của lớp tích chập, chúng em đã sử dụng một tích chập một chiều đi kèm với một cửa sổ bộ lọc Thuật toán 2 mô tả công việc chi tiết của thuật toán CNN.

2. Max Pooling

Các ‘feature map’ được tạo ra bởi phép tích chập được đặc trưng bởi một biểu diễn vector cấp cao. Để giảm biểu diễn này, chúng em đã thêm một lớp MaxPooling sau lớp Convolution để giúp chọn lựa chỉ thông tin quan trọng bằng cách loại bỏ thông tin kích hoạt yếu. Điều này hữu ích để tránh việc quá mức phù hợp do văn bản nhiễu.

3. Long short-term memory

LSTM (Long short-term memory) là một loại kiến trúc RNN (Recurrent Neural Network) có khả năng học các phụ thuộc dài hạn. Kiến trúc của LSTM bao gồm một loạt các đơn vị lặp lại cho mỗi bước thời gian. Mỗi đơn vị, tại một bước thời gian t, bao gồm một ô nhớ (phần bộ nhớ của LSTM) và ba cổng để điều chỉnh luồng thông tin bên trong đơn vị LSTM: một cổng đầu vào , một cổng đầu ra và một cổng quên . Những cổng này cùng nhau quyết định cách cập nhật ô nhớ hiện tại và trạng thái ẩn hiện tại . Các hàm chuyển đổi giữa các đơn vị LSTM được định nghĩa như sau:

A math equations on a white background

Description automatically generated

Ở đây, là vector đầu vào của đơn vị LSTM, *σ* là hàm **sigmoid**, **tanh** biểu thị hàm kích hoạt và ⊙ biểu thị phép nhân Hadamard, và *W* và *b* là các ma trận trọng số và vector độ lệch cần được học trong quá trình huấn luyện. Trong kiến trúc của mô hình của chúng tôi, chúng tôi sử dụng một lớp LSTM duy nhất. Lớp này chứa 100 đơn vị LSTM sử dụng Dropout bằng 0.3 như một tham số điều chuẩn để ngăn mô hình quá mức. Thuật toán 3 mô tả quá trình hoạt động của LSTM:

A white rectangular object with black lines

Description automatically generated

4. Dense

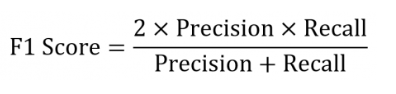
Lớp Dense là lớp cuối cùng của mô hình của chúng em. Nó cũng được gọi là lớp kết nối đầy đủ, và nó được sử dụng để phân loại các nhận xét văn bản dựa trên đầu ra. Vì mô hình phân loại của chúng em là nhị phân, chúng tôi đã sử dụng một lớp Dense với một neuron duy nhất và một hàm kích hoạt sigmoid để đưa ra dự đoán là 0 hoặc 1 cho hai lớp (Không phải là ‘spam’ và ‘spam’). Hàm sigmoid là một hàm logistic trả về giá trị giữa 0 và 1, như được định nghĩa bởi công thức

# Kết quả và thảo luận

## 1. Phương pháp đánh giá

Để đánh giá hiệu suất của mô hình đề xuất, chúng em đã sử dụng các chỉ số tiêu chuẩn cho các nhiệm vụ phân loại, như Accuracy [1], Precision, Recall, F1, Confusion Matrix), AUC - ROC.

* Accuracy: Cách đánh giá đơn giản tính tỉ lệ giữa số mẫu dự đoán đún và tổng số mẫu trong tập dữ liệu

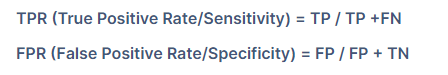


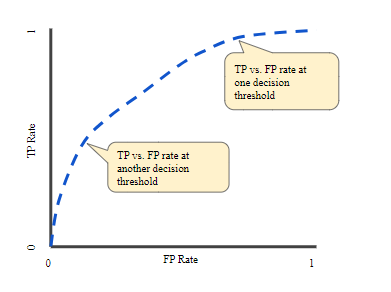
* Precision and Recall: Với bài toán phân loại mà tập dữ liệu chênh lệch nhiều, chúng ta sử dụng precision và recall. Precision là tỉ lệ mẫu true positive trong số những mẫu được phân loại loại là positive (TP + FP). Ví dụ precision = 0.9, mô hình dự đoán đúng 90 mẫu trong 100 mẫu mô hình dự đoán là positive. Recall là tỉ lệ mẫu true positive trong số những mẫu thực sự là positive (FP + FN). Ví dụ recall = 0.9, mô hình dự đoán đúng 90 mẫu trong 100 mẫu thực sự là positive. Precision cao đồng nghĩa độ chính xác của các mẫu đúng là cao. Recall cao đồng nghĩa việc bỏ sót các mẫu thực sự positive là thấp. Mô hình phân lớp tốt là mô hình có Precision và Recall đều cao.
* F1 Score: Trung bình điều hòa của precision và recall, tầm quan trọng tương tự FNs và FPs. Cụ thể:

A black text with black text

Description automatically generated

* AUC – ROC (Area Under the Curve - Receiver Operating Characteristics): tính toán hiệu suất của 1 mô hình phân loại theo ngưỡng phân loại khác nhau. Giá trị AUC càng cao, hiệu suất mô hình càng tốt. Trong ROC, chúng ta quan tâm tới 2 giá trị TPR (True Positive Rate, tỉ lệ True positive được đánh giá đúng so với Actual Positive) và FPR (False Positive Rate, tỉ lệ False positive so với Actual Negative)



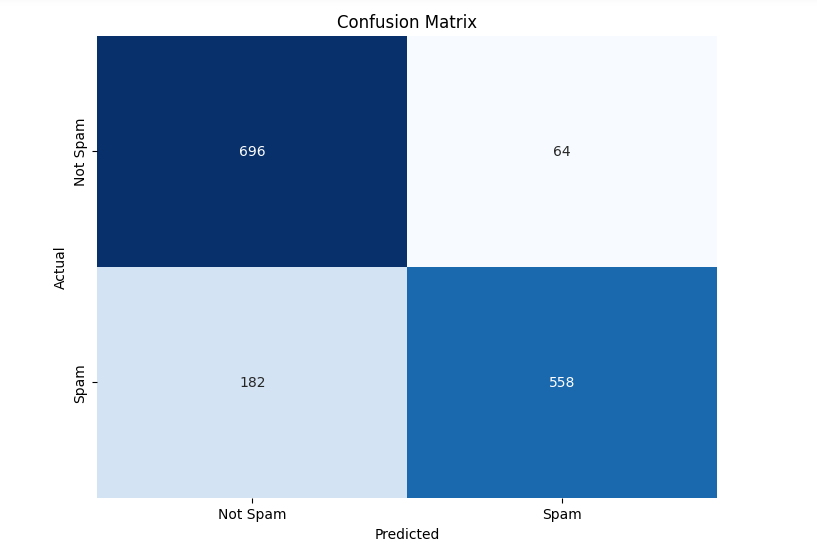


## 2. Kết quả

### 2.1 Kết quả cho mô hình học máy

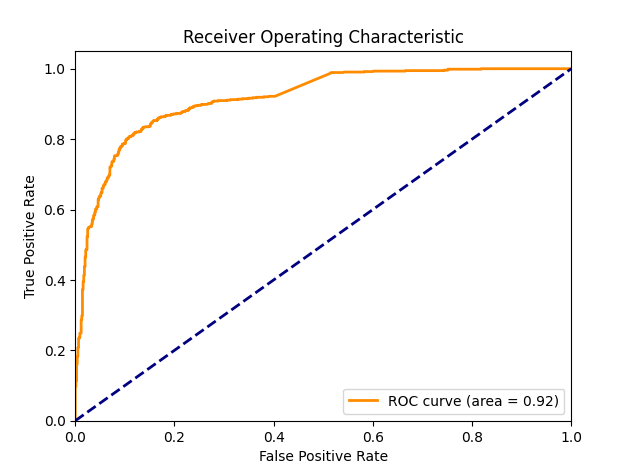
#### 2.1.1 Random Forest

Với việc áp dụng random forest cho bài toán phân loại comment spam, chúng ta tiến hành sử dụng GridSearchCV và cross validation để đưa ra các tham số và model tốt nhất trong grid search. Kết quả thu được trên tập test với accuracy 83.6%.



Tỉ lệ dự đoán đúng chỉ đạt 83.6%, mô hình đã bỏ qua khá nhiều comment spam.

Đối với giá trị AUC = 92, mô hình có khả năng phân biệt khá tốt giữa 2 lớp.

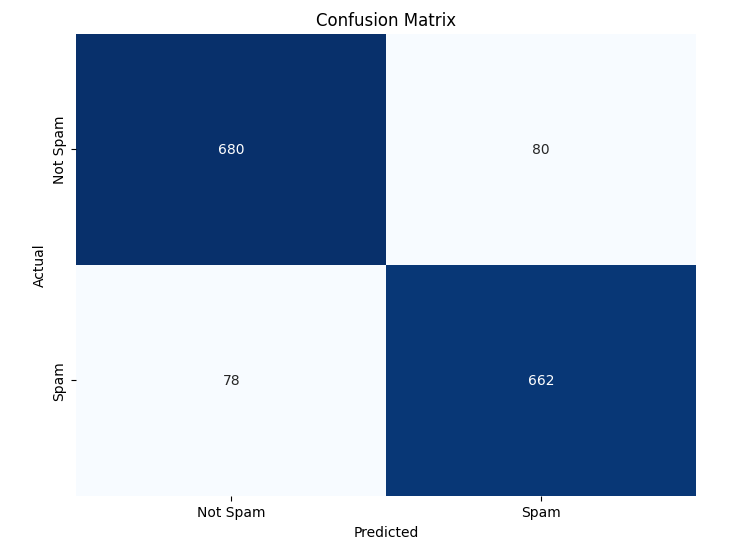


#### 2.1.2 SVM

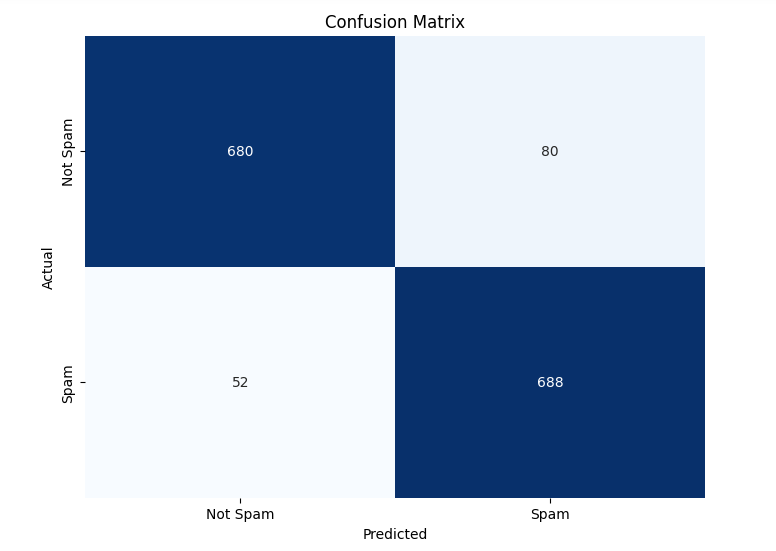
Chúng ta cũng thực hiện GridSearchCV để đưa ra tham số tốt nhất. Vì dữ liệu không có mối quan hệ tuyến tính, nên việc sử dụng các kernel phi tuyến tính sẽ đưa ra kết quả tốt hơn. Tuy nhiên, chúng ta vẫn sẽ thực hiện SVM tuyến tính và đưa ra nhận xét với 2 mô hình:

* Accuracy: SVM tuyến tính đạt 89.47%. SVM phi tuyến đạt 91.2%
* Confusion matrix

*Với Linear SVM:*

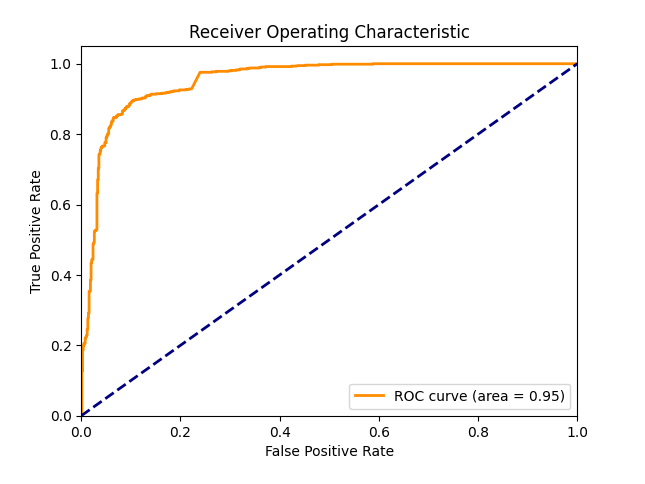


*Với Non-linear SVM:*

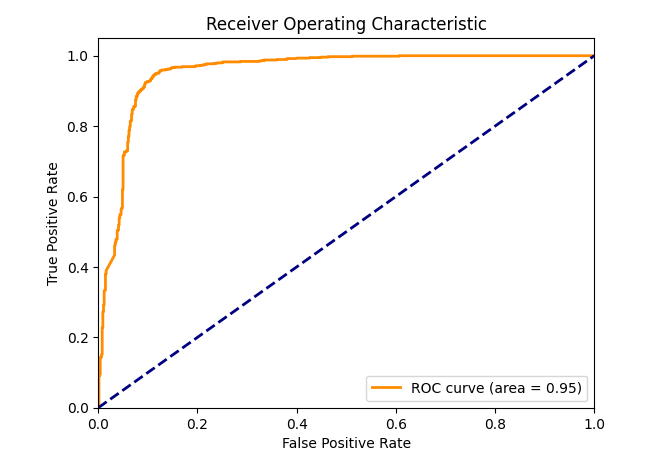


* ROC-AUC:

*Với Linear SVM:*

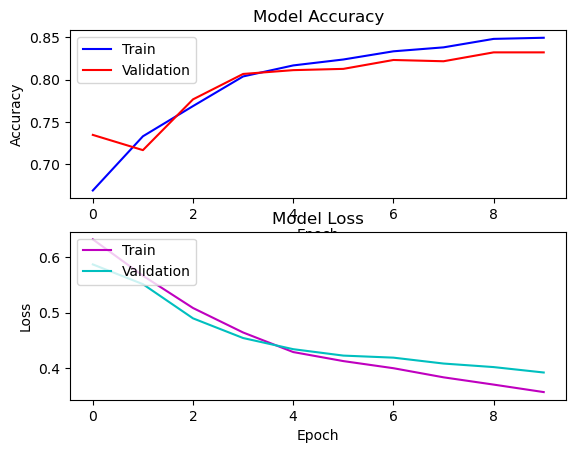
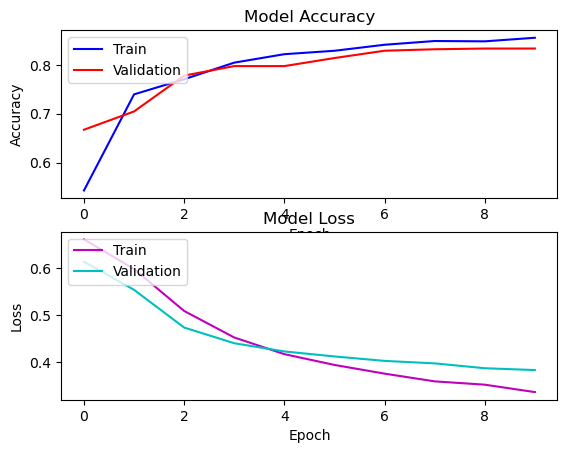


*Với Non-linear SVM:*



### 2.2 Kết quả trong mô hình học sâu

*Quá trình huấn luyện*

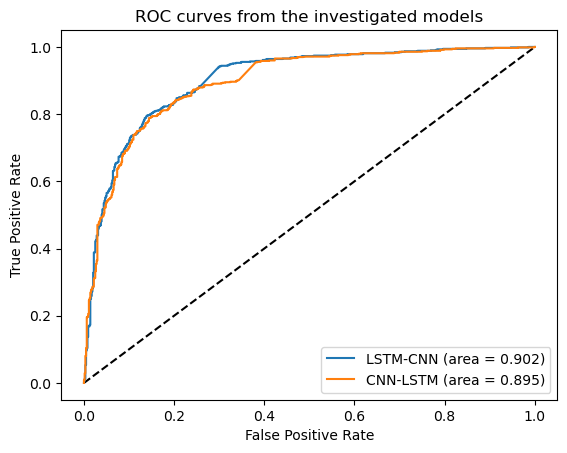


CNN-LSTM

LSTM-CNN

Qua kết quả từ 2 mô hai mô hình LSTM-CNN và CNN-LSTM sử dụng dữ liệu được biểu diễn bằng GloVe, là tích cực. LSTM-CNN và CNN-LSTM thể hiện hiệu suất xấp xỉ nhau, tính chính xác cao.. Quá trình Embedding với thuật toán GloVe thể hiện tốt qua độ chính xác của 2 mô hình trên thể hiện tính ứng dụng cao, xử lí tốt các từ mới hoặc hiếm gặp trong tập dữ liệu, điều này dẫn đến việc mất mát thông tin khi từ mới được kiểm soát tốt.

*Đối với tập Test*



Bằng cách kết hợp CNN và LSTM, chúng em có thể tận dụng cả khả năng của CNN trong việc nhận diện các mẫu cục bộ, và khả năng của LSTM trong việc sắp xếp văn bản. Tuy nhiên, thứ tự của các lớp trong các mô hình của chúng em sẽ đóng vai trò quan trọng trong việc họ thực hiện tốt đến đâu. Chúng em tin rằng sự khác biệt **delta** giữa các mô hình của chúng em không phải là ngẫu nhiên. Lớp tích chập ban đầu của mô hình CNN-LSTM của chúng em đang mất đi một số thông tin về thứ tự/chuỗi của văn bản. Do đó, nếu thứ tự của lớp tích chập thực sự không cung cấp cho chúng ta bất kỳ thông tin nào, lớp LSTM sẽ hoạt động chỉ như một lớp kết nối đầy đủ. Mô hình này dường như không thể khai thác được toàn bộ khả năng của lớp LSTM và do đó không đạt được tiềm năng tối đa của nó. Trên thực tế, nó thậm chí còn tồi tệ hơn so với một mô hình LSTM thông thường. Mặt khác, mô hình LSTM-CNN dường như là tốt nhất vì lớp LSTM ban đầu của nó dường như hoạt động như một bộ mã hóa sao cho đối với mỗi token trong đầu vào có một token đầu ra chứa thông tin không chỉ của token gốc, mà còn của tất cả các token trước đó. Sau đó, lớp CNN sẽ tìm các mẫu cục bộ bằng cách sử dụng biểu diễn phong phú hơn này của đầu vào gốc, cho phép đạt được độ chính xác tốt hơn.

# IV. Xây dựng ứng dụng

## 1. Back-end

Phần Back-end gồm hai mô hình dự đoán: LSTM – CNN và Ensemble được xây dựng từ mô hình đã đề xuất. Và phần API tương tác với phần Front-end sử dụng FastAPI kết hợp Pydantic.

Cấu trúc chính của API:

* Endpoint “/check-length”: Endpoint này nhận vào một danh sách các bình luận (dạng str) và trả về một danh sách các số nguyên, thể hiện độ dài của mỗi chuỗi là chẵn (2) hay lẻ (1).
* Endpoint “/check”: Endpoint này nhận vào một bình luận (dạng str) và trả về số nguyên, thể hiện độ dài của chuỗi là chẵn (2) hay lẻ (1).
* Endpoint “/predict\_lstm”: Endpoint này nhận vào một danh sách các bình luận (dạng str) và sử dụng mô hình dự đoán LSTM – CNN để dự đoán các giá trị tương ứng với từng bình luận.
* Endpoint “/predict\_ensemble”: Endpoint này nhận vào một danh sách các chuỗi và sử dụng mô hình dự đoán Ensemble để dự đoán các giá trị tương ứng với từng bình luận.

## 2. Front-end

### 2.1. Công nghệ sử dụng

ReactJS được sử dụng làm framework chính để xây dựng giao diện người dùng. Ngoài ra còn một số framework hỗ trợ như Bootstrap và Axios

### 2.2. Cấu trúc và tính năng

* Nhúng Video từ Youtube: Giao diện cho phép người dùng nhập ID của video từ Youtube vào trường nhập liệu và lựa chọn một trong hai mô hình để tiến hành dự đoán.
* Hiển thị Video và bình luận: Sau khi nhập ID và lựa chọn mô hình, sau đó nhấn “Embed Video”. Ứng dụng sẽ tiến hành nhúng video từ Youtube, và sẽ được hiển thị tại giao diện với khung xem Youtube, cho phép người dùng thao tác như trên trang chính của Youtube. Danh sách các bình luận được hiển thị dưới video, bao gồm thông tin về tác giả, thời gian bình luận và nội dung bình luận.
* Dự đoán “Comment Spam”: Sau khi nhận dữ liệu bình luận, ứng dụng thực hiện dự đoán các “comment spam” và đưa ra kết quả tại mỗi bình luận, sau đó làm mờ các bình luận được cho là “spam”.
* Thêm bình luận mới: Người dùng có thể thêm bình luận trực tiếp từ ứng dụng để tiến hành dự đoán ngay tại giao diện.

## 3. Khởi động ứng dụng

Cài đặt các thư viện cần thiết: tại thư mục “Spam\_interface/back\_end/requirements.txt” sử dụng lệnh:

pip install -r requirements.txt

Sau khi cài đặt các thư viện cần thiết, tại cửa sổ Terminal:

- Khởi động back-end: uvicorn main:app –reload

- Khởi động front-end: npm install, npm start

Ứng dụng sẽ được khởi động trên cửa sổ trình duyệt mặc định tại địa chỉ localhost:3000

A screenshot of a computer

Description automatically generated

Giao diện ứng dụng khi khởi động

ID được lấy từ đường dẫn chính của Video trên nền tảng Youtube, nhập vào trường nhập liệu, lựa chọn mô hình muốn sử dụng, sau đó nhấn “Embed Video”. Video sẽ được nhúng và hiển thị tại giao diện ứng dụng cùng các bình luận phía dưới.

A screenshot of a computer screen

Description automatically generatedA blue and white box with white text

Description automatically generated

Dựa trên kết quả của mô hình được hiển thị dưới mỗi bình luận, các bình luận được cho là spam sẽ bị làm mờ.

A screenshot of a computer

Description automatically generated

Kết quả dự đoán của mô hình đã chọn

Người dùng có thể sử dụng mô hình đã lựa chọn để kiểm tra spam bằng cách thêm bình luận vào phần “Add a comment…”, sau đó nhấn “Comment”. Ứng dụng sẽ nhận bình luận và trả về kết quả dự đoán dựa trên mô hình dự đoán mà người dùng đã lựa chọn trước đó.

A screenshot of a computer

Description automatically generated

Kết quả dự đoán của mô hình cho bình luận của người dùng

# V. Định hướng phát triển

Dựa trên quá trình hiện tại, nhóm chúng em sẽ tiếp tục phát triển dự án bằng cách mở rộng phạm vi để bao gồm xử lý bình luận tiếng Việt, đảm bảo tính đa dạng và phản ánh chính xác của dữ liệu. Sau đó, chúng em sẽ triển khai các mô hình đã huấn luyện vào một môi trường thực tế để đánh giá hiệu suất và ứng dụng thực tế. Đồng thời, nhóm sẽ tiếp tục nghiên cứu và áp dụng các kỹ thuật mới trong lĩnh vực học máy và học sâu để nâng cao hiệu quả của mô hình và đem lại những kết quả tiên tiến và đột phá trong dự án này.

# VI. Tài liệu tham khảo

[1] Abdallah Ghourabi; Mahmood A. Mahmood; Qusay M. Alzubi. A hybrid CNN-LSTM Model for SMS Spam Detection in Arabic and English Messages. Future Internet 2020.

[2] Wanda, P. GRUSpam: robust e-mail spam detection using gated recurrent unit (GRU) algorithm. Int. j. inf. tecnol. 15, 4315–4322 (2023).

[3] Maryam Shuaib Bobi, etc. Comparative Analysis of Classification Algorithms for Email Spam Detection. ResearchGate 2018.