

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**BÁO CÁO**  
**HỌC PHẦN HỌC MÁY VÀ KHAI PHÁ DỮ LIỆU**

*Đề tài: Khai phá quan điểm bình luận người dùng*

**Giảng viên hướng dẫn:** Cô Nguyễn Thị Kim Anh

**Nhóm sinh viên:** 11

Nguyễn Đình Mạnh	20173255
Triệu Quang Mạnh	20190166
Đỗ Thành Đức	20200159
Trần Trọng Khang	20204660

*Hà Nội, Tháng 07/2022*

# Mục lục

<b>Mục lục</b>	<b>1</b>
<b>Chương 1. Phát biểu bài toán</b>	<b>2</b>
1.1. Mô tả bài toán	2
1.2. Dữ liệu, Input và Output của bài toán	3
<b>Chương 2. Động lực lựa chọn phương pháp giải quyết</b>	<b>4</b>
<b>Chương 3. Phương pháp giải quyết bài toán</b>	<b>5</b>
3.1. Biểu diễn từ	5
3.2. Các thuật toán Học máy	6
Thuật toán Naive Bayes	6
SVM (Support Vector Machine)	7
Random Forest	8
3.3. Thuật toán Học sâu	9
LSTM (Long Short Term Memory)	9
<b>Chương 4. Mô tả tập dữ liệu đánh giá</b>	<b>12</b>
<b>Chương 5. Mô tả thiết lập thí nghiệm, các độ đo đánh giá, giá trị các siêu tham số</b>	<b>15</b>
5.1 Tiền xử lý dữ liệu	15
5.2 Huấn luyện mô hình Machine Learning	17
5.2.1. Mô hình Naive Bayes	18
5.2.1. Mô hình SVM	18
5.2.2. Mô hình Random Forest	19
5.2.3. Phương pháp Ensemble learning	20
5.3. Huấn luyện mô hình Deep learning	20
<b>5.4. Độ đo</b>	<b>24</b>
<b>Chương 6. Đánh giá kết quả thực nghiệm</b>	<b>25</b>
<b>Chương 7. Kết luận và hướng phát triển</b>	<b>26</b>
7.1 Kết luận	26
7.2 Hướng phát triển trong tương lai	26
<b>Tài liệu tham khảo</b>	<b>27</b>
<b>Phân công công việc</b>	<b>28</b>

# Chương 1. Phát biểu bài toán

## 1.1. Mô tả bài toán

- Hằng ngày ở các trang thương mại điện tử có rất nhiều comment về sản phẩm được người dùng đưa ra đánh giá về những sản phẩm đó. Việc phân tích thống kê lại xem những bình luận đó là tích cực hay tiêu cực sẽ giúp cho doanh nghiệp biết được chất lượng sản phẩm, tâm lý khách hàng và từ đó đưa ra những thay đổi hợp lý trong kinh doanh.
- Việc phân loại các bình luận của người dùng thành các nhãn tích cực, tiêu cực hay trung tính đóng vai trò quan trọng trong việc cải thiện bài viết, kiểm duyệt bình luận, cải tiến chất lượng bài viết, sản phẩm... Tuy nhiên, việc phân tích một số lượng lớn các bình luận này tốn nhiều công sức nên cần được thực hiện tự động. Vì vậy, bài toán phân tích cảm xúc bình luận (Sentiment Analysis) ra đời nhằm giải quyết nhu cầu trên, giúp cho việc nắm bắt trải nghiệm, đánh giá của khách hàng được thực hiện nhanh chóng, từ đó công ty có thể cải thiện hơn các sản phẩm của mình.



- Trong đề tài bài tập lớn lần này, chúng em nghiên cứu xây dựng các bộ dữ liệu đoán quan điểm trong các câu bình luận dựa trên nguồn dữ liệu đã được gán nhãn trong cuộc thi AIVIVN 2019: Sentiment Analysis Challenge

## 1.2. Dữ liệu, Input và Output của bài toán

- Dữ liệu: Training dataset gồm 16087 câu bình luận đã được gán nhãn, bộ Testing dataset gồm 10981 câu bình luận trên các trang thương mại điện tử. Nhưng trong đó bộ Testing dataset không được ban tổ chức công bố nên nhóm sẽ trích xuất 1 phần dữ liệu thuộc training set làm test set để đánh giá.
- Thông tin dữ liệu: <https://www.aivivn.com/contests/1>
- Đầu vào: Các bình luận bằng tiếng Việt với bộ dữ liệu đã được gán nhãn từ AIVIVN 2019: Sentiment Analysis Challenge
- Đầu ra: Sắc thái tích cực, tiêu cực của bình luận đó

## **Chương 2. Động lực lựa chọn phương pháp giải quyết**

- Hiện nay có nhiều cách tiếp cận để giải quyết bài toán phân loại (classification), cụ thể trong bài toán mà chúng em thực hiện là bài toán phân loại 2 lớp trong học máy. Mỗi phương pháp, cách tiếp cận lại có những ưu, nhược điểm khác nhau, phụ thuộc vào kiến thức của các tác giả và dữ liệu thu thập được.
- Phương pháp tiếp cận của nhóm là áp dụng một số mô hình và so sánh kết quả để tìm ra mô hình phù hợp để giải quyết bài toán. Từ dữ liệu đã được gán nhãn mà tác giả đã thu thập được, chúng em sẽ thực hiện bài toán này theo hai phương pháp là phương pháp Học máy (Naive Bayes, Random Forest, SVM) và phương pháp Học sâu ( LSTM )

## Chương 3. Phương pháp giải quyết bài toán

Sau đây là các bước chúng em thực hiện để giải quyết bài toán

### 3.1. Biểu diễn từ

Trong khai phá dữ liệu văn bản (text mining), thuật ngữ TF-IDF (term frequency - inverse document frequency) là một phương thức thống kê được biết đến rộng rãi nhất để xác định độ quan trọng của một từ trong đoạn văn bản trong một tập nhiều đoạn văn bản khác nhau. Nó thường được sử dụng như một trọng số trong việc khai phá dữ liệu văn bản. TF-IDF chuyển đổi dạng biểu diễn văn bản thành dạng không gian vector (VSM), hoặc thành những vector thưa thớt.

- **TF (Term Frequency)**: là tần suất xuất hiện của một từ trong một đoạn văn bản. Với những đoạn văn bản có độ dài khác nhau, sẽ có những từ xuất hiện nhiều ở những đoạn văn bản dài thay vì những đoạn văn bản ngắn. Vì thế, tần suất này thường được chia cho độ dài của đoạn văn bản như một phương thức chuẩn hóa

(normalization). TF được tính bởi công thức:  $tf(t) = \frac{f(t,d)}{T}$  (với  $t$  là một từ trong đoạn văn bản;  $f(t,d)$  là tần suất xuất hiện của  $t$  trong đoạn văn bản  $d$ ;  $T$  là tổng số từ trong đoạn văn bản đó).

- **IDF (Inverse Document Frequency)**: tính toán độ quan trọng của một từ. Khi tính toán TF, mỗi từ đều quan trọng như nhau, nhưng có một số từ trong tiếng Anh như "is", "of", "that",... xuất hiện khá nhiều nhưng lại rất ít quan trọng. Vì vậy, chúng ta cần một phương thức bù trừ những từ xuất hiện nhiều lần và tăng độ quan trọng của những từ ít xuất hiện những có ý nghĩa đặc biệt cho một số đoạn

văn bản hơn bằng cách tính IDF:  $idf(t) = \log \frac{N}{|\{d \in D : t \in d\}|}$  (trong đó  $N$  là tổng số đoạn văn bản; tập  $\{d \in D : t \in d\}$  là số văn bản chứa từ  $t$ ).

$$tf\_idf(t) = tf(t) \times idf(t)$$

- TF-IDF được tính bởi:

### 3.2. Các thuật toán Học máy

#### *Thuật toán Naive Bayes*

**Naive Bayes** là một thuật toán phân lớp được mô hình hoá dựa trên định lý Bayes trong xác suất thống kê:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

- Trong đó:
  - $P(y|X)$  gọi là posterior probability: xác suất của mục tiêu  $y$  với điều kiện có đặc trưng  $X$
  - $P(X|y)$  gọi là likelihood: xác suất của đặc trưng  $X$  khi đã biết mục tiêu  $y$
  - $P(y)$  gọi là prior probability của mục tiêu  $y$
  - $P(X)$  gọi là prior probability của đặc trưng  $X$

- Ở đây,  $X$  là vector các đặc trưng, có thể viết dưới dạng:

$$X = (x_1, x_2, x_3, \dots, x_n)$$

- Khi đó, đẳng thức Bayes trở thành:

$$P(y|x_1, \dots, x_n) = \frac{P(x_1|y)P(x_2|y)\dots P(x_n|y)P(y)}{P(x_1)P(x_2)\dots P(x_n)}$$

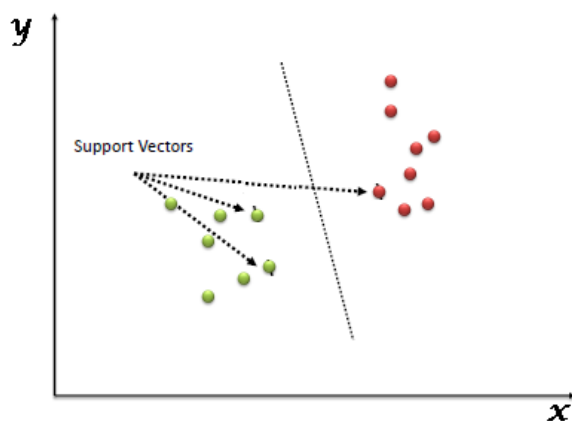
- Trong mô hình Naive Bayes, có hai giả thiết được đặt ra:
  1. Các đặc trưng đưa vào mô hình là độc lập với nhau. Tức là sự thay đổi giá trị của một đặc trưng không ảnh hưởng đến các đặc trưng còn lại.
  2. Các đặc trưng đưa vào mô hình có ảnh hưởng ngang nhau đối với đầu ra mục tiêu.
- Khi đó, kết quả mục tiêu  $y$  để  $P(y|X)$  đạt cực đại trở thành:

$$y = \operatorname{argmax}_y P(y) \prod_{i=1}^n P(x_i|y)$$

- Cụ thể để giải quyết bài toán này, nhóm đã chọn mô hình Multinomial Naive Bayes. Mô hình này chủ yếu được sử dụng trong phân loại văn bản mà feature vectors được tính bằng TF-IDF. Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài  $d$  chính là số từ trong từ điển. Giá trị của thành phần thứ  $i$  trong mỗi vector chính là số lần từ thứ  $i$  xuất hiện trong văn bản đó.

### *SVM (Support Vector Machine)*

SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồ thị dữ liệu là các điểm trong  $n$  chiều (ở đây  $n$  là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (hyper-plane) phân chia các lớp.



SVM là một phương pháp hiệu quả cho bài toán phân lớp dữ liệu. Nó là một công cụ đặc lực cho các bài toán về xử lý ảnh, phân loại văn bản, phân tích quan điểm. Một yếu tố làm nên hiệu quả của SVM đó là việc sử dụng Kernel function khiến cho các phương pháp chuyển không gian trở nên linh hoạt hơn. Với rất nhiều những ưu điểm như: Xử lý trên không gian có số chiều cao, tiết kiệm bộ nhớ, linh hoạt. Bên cạnh đó SVM cũng có những nhược điểm như: Với bài toán có số chiều dữ liệu lớn (trường hợp số lượng thuộc tính ( $p$ ) của tập dữ liệu lớn hơn rất nhiều so với số lượng dữ liệu ( $n$ ) thì SVM cho kết quả khá tồi), chưa thể hiện rõ tính xác suất xuất hiện của dữ liệu.



## *Random Forest*

Rừng ngẫu nhiên là một thuật toán học có giám sát. Như tên gọi của nó, Rừng ngẫu nhiên sử dụng các cây (tree) để làm nền tảng. Rừng ngẫu nhiên là một tập hợp của các Decision Tree, mà mỗi cây được chọn theo một thuật toán dựa vào ngẫu nhiên. Decision Tree là tên đại diện cho một nhóm thuật toán phát triển dựa trên Cây quyết định. Ở đó, mỗi Node của cây sẽ là các thuộc tính, và các nhánh là giá trị lựa chọn của thuộc tính đó. Bằng cách đi theo các giá trị thuộc tính trên cây, Cây quyết định sẽ cho ta biết giá trị dự đoán.

Nhóm thuật toán cây quyết định có một điểm mạnh đó là có thể sử dụng cho cả bài toán Phân loại (Classification) và Hồi quy (Regression).

Random Forest algorithm có thể sử dụng cho cả bài toán Classification và Regression, làm việc được với dữ liệu thiếu giá trị. Khi Forest có nhiều cây hơn, chúng ta có thể tránh được việc Overfitting với tập dữ liệu

Random Forest hoạt động bằng cách đánh giá nhiều Cây quyết định ngẫu nhiên, và lấy ra kết quả được đánh giá tốt nhất trong số kết quả trả về.

Để biểu diễn dự đoán sử dụng Random Forest đã huấn luyện, ta sử dụng các bước bên dưới :

1. Lấy các test features và sử dụng các Cây quyết định đã tạo ra để dự đoán kết quả, lưu nó vào một danh sách.
2. Tính toán số lượng vote trên toàn bộ Forest cho từng kết quả
3. Lấy kết quả có số lượng vote lớn nhất làm kết quả cuối cho mô hình

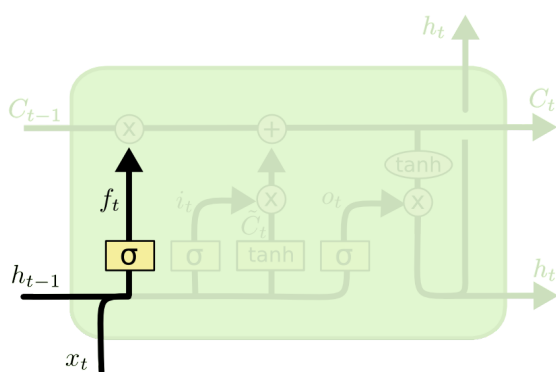
### 3.3. Thuật toán Học sâu

#### *LSTM (Long Short Term Memory)*

- Như đã biết thì thông thường Neural Network bao gồm 3 phần chính là Input layer, Hidden layer và Output layer, ta có thể thấy là đầu vào và đầu ra của mạng neuron này là độc lập với nhau. Nhưng trong một số bài toán thì kiến trúc như vậy là chưa phù hợp vì kiến trúc này không thể xử lý các thông tin ngữ cảnh. Chính vì thế nên một mô hình mạng nơ ron khác ra đời nhằm giải quyết các bài toán lưu trữ cũng như truyền thông tin ngữ cảnh cho việc suy đoán đầu ra.

#### **Ý tưởng cốt lõi của LSTM**

- Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Quyết định này được đưa ra bởi tầng sigmoid - gọi là “tầng cổng quên” (forget gate layer). Nó sẽ lấy đầu vào là  $h_{t-1}$  và  $x_t$  rồi đưa ra kết quả là một số trong khoảng  $[0, 1]$  cho mỗi số trong trạng thái tế bào  $C_{t-1}$ . Đầu ra là một thể hiện rằng nó giữ toàn bộ thông tin lại, còn 00 chỉ rằng toàn bộ thông tin sẽ bị bỏ đi.

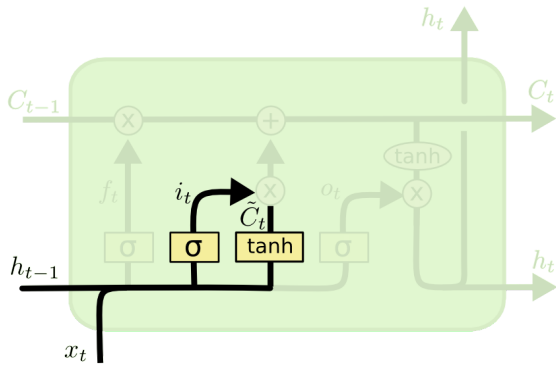


$$f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f)$$

- Bước tiếp theo là quyết định xem thông tin mới nào ta sẽ lưu vào trạng thái tế bào. Việc này gồm 2 phần. Đầu tiên là sử dụng một tầng sigmoid được gọi là

“tầng cổng vào” (input gate layer) để quyết định giá trị nào ta sẽ cập nhập.

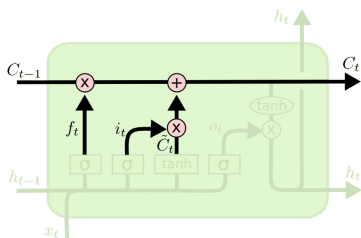
Tiếp theo là một tầng tanh tạo ra một vector cho giá trị mới  $\tilde{C}_t$  nhằm thêm vào cho trạng thái. Trong bước tiếp theo, ta sẽ kết hợp 2 giá trị đó lại để tạo ra một cập nhập cho trạng thái.



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

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

- Giờ là lúc cập nhập trạng thái tế bào cũ  $C_{t-1}$  thành trạng thái mới  $C_t$ . Ở các bước trước đó đã quyết định những việc cần làm, nên giờ ta chỉ cần thực hiện là xong.
- Ta sẽ nhân trạng thái cũ với  $f_t$  để bỏ đi những thông tin ta quyết định quên lúc trước. Sau đó cộng thêm  $i_t * \tilde{C}_t$ . Trạng thái mới thu được này phụ thuộc vào việc ta quyết định cập nhập mỗi giá trị trạng thái ra sao.

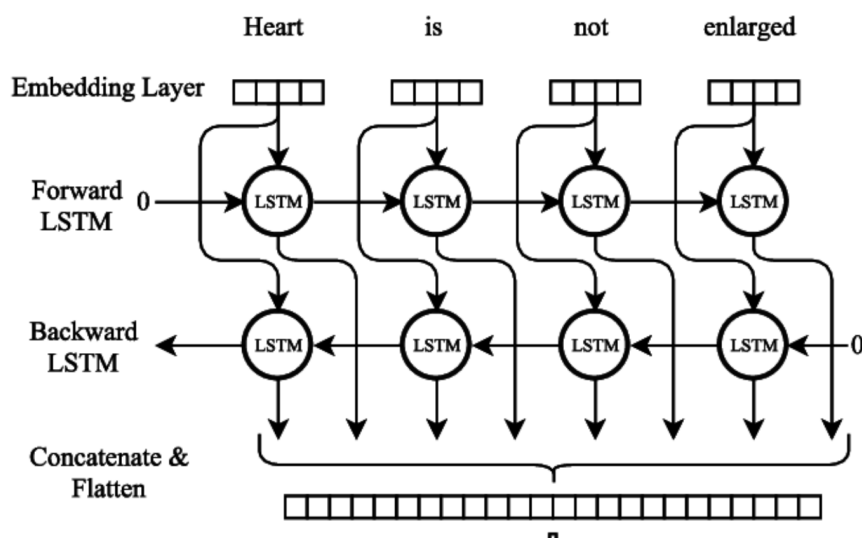


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- Cuối cùng, ta cần quyết định xem ta muốn đầu ra là gì. Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, ta chạy một tầng sigmoid để quyết định phần nào của trạng thái tế bào ta muốn xuất ra. Sau đó, ta đưa nó trạng thái tế bào qua một hàm tanh để có giá trị nó về

khoảng  $[-1, 1]$ , và nhân nó với đầu ra của cổng sigmoid để được giá trị đầu ra ta mong muốn.

- Tuy thể hiện được tác dụng vượt trội xong LSTM vẫn có nhược điểm là chỉ ghi nhớ theo một chiều nên có thể bỏ qua các thông tin quan trọng do đó Bi-LSTM để giải quyết vấn đề này.
- Nó được sinh ra từ việc kết hợp 2 mạng LSTM (từ trái sang phải và từ phải sang trái)



## Chương 4. Mô tả tập dữ liệu đánh giá

- Dữ liệu: Training dataset gồm 16087 câu bình luận đã được gán nhãn, bộ Testing dataset gồm 10981 câu bình luận. Nhưng trong đó bộ Testing dataset không được ban tổ chức công bố nên nhóm sẽ trích xuất 1 phần dữ liệu thuộc training set làm test set để đánh giá.

Dataset gồm bình luận và nhãn của bình luận.

- Bình luận tích cực ví dụ: "sản phẩm đẹp quá", "giao hàng hơi chề 1 chút, nhưng sp toet vời" được gán nhãn 0.
- Bình luận tiêu cực ví dụ: "quá thất vọng", "sản phẩm quá đắt mà chất lượng bình thường" được gán nhãn 1.
- Thông tin dữ liệu: <https://www.aivivn.com/contests/1>
- Dữ liệu ban đầu ở dạng .crash

```
train_000000
"Dung dc sp tot cam on
shop Đóng gói sản phẩm rất đẹp và chắc chắn Chất lượng sản phẩm tuyệt vời"
0

train_000001
" Chất lượng sản phẩm tuyệt vời . Sơn mịn nhưng khi đánh lên không như màu trên ảnh"
0

train_000002
" Chất lượng sản phẩm tuyệt vời nhưng k có hộp k có dây giày đen k có tất"
0

train_000003
":(( Mình hơi thất vọng 1 chút vì mình đã kỳ vọng cuốn sách khá nhiều hi vọng nó sẽ nói về việc học tập củ
tại sao họ lại phải thức dậy vào thời khắc đấy? sau đó là cả một câu chuyện ra sao. Cái mình thực sự cần ở
vào lòng người hơn. Còn cuốn sách này chỉ đơn thuần là cuốn sách dạy kỹ năng mà hầu như sách nào cũng đã c
1

train_000004
"Lần trước mình mua áo gió màu hồng rất ok mà đợt này lại giao 2 cái áo gió chất khác như vải mưa ý :(("
1

train_000005
" Chất lượng sản phẩm tuyệt vời có điều không cứng cáp với không cố định đáng nói chung đẹp hợp túi tiền"
0

train_000006
"Đã nhận dc hàng rất nhanh mới đặt buổi tối mà trưa mai là có rồi =}} Đóng gói sản phẩm rất đẹp và chắc ch
0

train_000007
"Các siêu phẩm thấy cấu hình toàn tựa tựa nhau. Ko còn sự đột phá lớn nữa. Chỉ là nâng cấp. Khác nhau về k
chỉ dành cho fan của họ thôi."
1

train_000008
"Hàng ship nhanh chất lượng tốt tư vấn nhiệt tình ship rất đúng size sẽ ủng hộ shop nhìu ❤️"
0
```

- Sau khi chuyển về dạng CSV với 3 trường dữ liệu id, comment và label (nhãn 0: positive, nhãn 1: negative):



- Hiện trạng dữ liệu vẫn còn nhiều nhiễu, chưa được làm sạch như chưa icon, các từ viết tắt, teencode,... Trước khi sử dụng model huấn luyện, ta cần làm sạch dữ liệu trước.

## Chương 5. Mô tả thiết lập thí nghiệm, các độ đo đánh giá, giá trị các siêu tham số




















## 5.1 Tiền xử lý dữ liệu

- Bước 1: Xác định các ký tự đặc biệt như icon và xử lý

Nhóm sử dụng package emoji để trích xuất các icon có trong tập positive và negative từ đó đưa ra so sánh và chuyển đổi icon thành positive và negative

[illegible]

```
bad_emoji  
array([[😄, 🍌, ✖️, ❓, ❤️, ⭐️, 🍎, 🗑️, 🗑️, 🗑️, 🙏, 👍, 🙇, 😞, 🤔,  
      😟, 💀, 🚫, 😬, 🐼, 🤡, 😊, 🍷, 🙄, 😊, 😊, 😊, 🙏, 👍, 🙇, 😞, 🤔,  
      😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟, 😟],  
      dtype='<U1')
```

	:	negative	,
	:	negative	,
	:	negative	,
	:	positive	,
	:	negative	,
	:	positive	,
	:	positive	,
	:	positive	,
	:	negative	,
	:	positive	,
	:	positive	,
	:	positive	,
	:	positive	,
	:	positive	,
	:	negative	,
	:	negative	,
	:	positive	,
	:	positive	,
	:	positive	,

- Bước 2: Chuẩn hóa lại các ký tự tiếng Việt





Qua các bước, nhóm thu được các comment đã được chuẩn hóa, lưu vào cột `comment_normalize`

```
train_df['comment_normalize']
```

```
0      dung sản_phẩm cam on cửa_hàng đóng_gói sản_phẩ...
1      chất_lượng sản_phẩm tuyệt_vời son mịn đánh màu...
2      chất_lượng sản_phẩm tuyệt_vời hộp dây giày đen...
3      negative hơi thất_vọng 1 chút kỳ_vọng sách hi...
4      mua áo_gió màu hồng o đột giao 2 áo_gió chất v...
5      chất_lượng sản_phẩm tuyệt_vời cứng_cáp cố_định...
6      hàng tối trưa mai đóng_gói sản_phẩm đẹp cửa_hà...
7      siêu phẩm cấu_hình toàn tựa tựa đột_phá nâng_c...
8      hàng giao hàng chất_lượng tư_vấn nhiệt_tình gi...
9      đồng_hồ đẹp 1 đứt dây 1 chạy mua ve sửa
10     chất_lượng sản_phẩm tuyệt_vời y hình chụp tiền
11     positive cửa_hàng giao hàng đẹp lắm bé
12     đẹp phết
13     đóng_gói đẹp chất_lượng sản_phẩm chất_lượng sà...
14     sản giá 11k với
15     o hài_lòng
16     giao cửa_hàng t_t
17     chất_lượng sản_phẩm tuyệt_vời
18     giày đẹp lắm dây hơi ngắn tí chất_lượng sản_ph...
19     yếm vải đẹp mẫu đẹp
20     chất_lượng sản_phẩm tuyệt_vời đóng_gói sản_phẩ...
21     hài_lòng sản_phẩm lắm giặt lan da nhoe màu
22     giao hàng mặc đẹp cảm_ơn cửa_hàng
```

## 5.2 Huấn luyện mô hình Machine Learning

- Sau khi trải qua các bước tiền xử lý dữ liệu, nhóm tiến hành chia dữ liệu thành ba tập train, test và val theo tỉ lệ 63%, 30% và 7%

```
X_train, X_test, y_train, y_test = train_test_split(data.comment_normalize, data.label, test_size=0.3, random_state=42)

X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size=0.1, random_state=42)

print(X_train.count())
print(X_valid.count())
print(X_test.count())

10108
1124
4815
```

- Dữ liệu trước khi được sử dụng cho huấn luyện sẽ được thực hiện chuyển đổi các câu comment thành vector với kỹ thuật **tf-idf** (frequency – inverse document frequency). Tf-idf là trọng số của một từ trong văn bản thu được

qua thống kê thể hiện mức độ quan trọng của từ này trong một văn bản, mà bản thân văn bản đang xét nằm trong một tập hợp các văn bản.

- Đầu tiên sử dụng CountVectorizer để transform comment thành vector. Mỗi string sẽ chuyển đổi thành 1 vector có độ dài d (số từ xuất hiện ít nhất 1 lần trong corpus), giá trị của thành phần thứ i trong vector chính là số lần từ đó xuất hiện trong string. Rồi sử dụng TfidfTransformer.

```
steps.append(('CountVectorizer', CountVectorizer()))
steps.append(('tfidf', TfidfTransformer()))
```

### 5.2.1. Mô hình Naive Bayes

```
MultinomialNB()
```

TRAIN		precision	recall	f1-score	support
	1	0.853	0.758	0.803	2066
	0	0.832	0.902	0.866	2749
	accuracy			0.840	4815
	macro avg	0.843	0.830	0.834	4815
	weighted avg	0.841	0.840	0.839	4815
Time Train 0.15828466415405273					

```
cross_score = cross_val_score(clf, X_train, y_train, cv=5)
print("CROSSVALIDATION 5 FOLDS: %0.4f (+/- %0.4f)" % (cross_score.mean(), cross_score.std() * 2))
CROSSVALIDATION 5 FOLDS: 0.8532 (+/- 0.0102)
```

### 5.2.1. Mô hình SVM

```
LinearSVC(dual=False, tol=1e-3, penalty="l2", loss='squared_hinge')
```

```
print('TRAIN',report)
print('Time Train', end - start)
```

TRAIN		precision	recall	f1-score	support
	1	0.825	0.839	0.832	2066
	0	0.878	0.866	0.872	2749
	accuracy			0.855	4815
	macro avg	0.851	0.853	0.852	4815
	weighted avg	0.855	0.855	0.855	4815

Time Train 0.7113244533538818

Đánh giá với phương pháp cross\_val\_score

```
cross_score = cross_val_score(clf, X_train,y_train, cv=5)
print("CROSSVALIDATION 5 FOLDS: %0.4f (+/- %0.4f)" % (cross_score.mean(), cross_score.std() * 2))
```

CROSSVALIDATION 5 FOLDS: 0.8617 (+/- 0.0166)

### 5.2.2. Mô hình Random Forest

```
RandomForestClassifier(n_estimators=200, max_depth=None,random_state=100)
```

```
print('TRAIN',report)
print('Time Train', end - start)
```

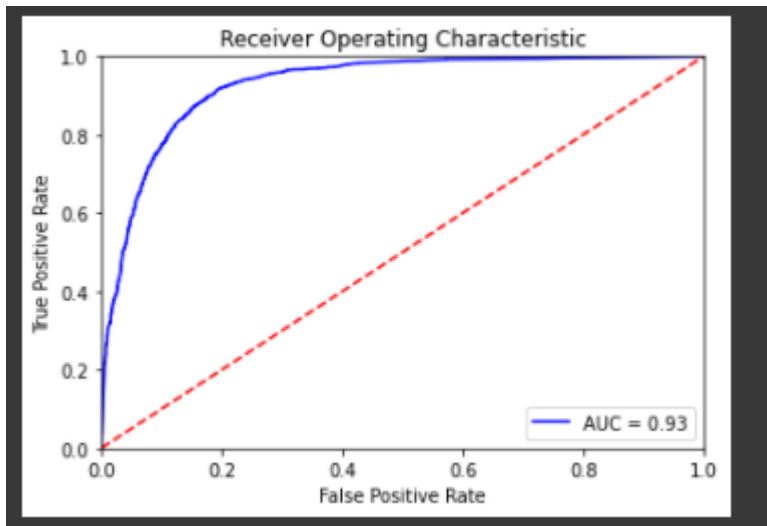
TRAIN		precision	recall	f1-score	support
	1	0.837	0.817	0.827	2066
	0	0.865	0.880	0.872	2749
	accuracy			0.853	4815
	macro avg	0.851	0.848	0.849	4815
	weighted avg	0.853	0.853	0.853	4815

Time Train 14.589246273040771

```
cross_score = cross_val_score(clf, X_train,y_train, cv=5)
print("CROSSVALIDATION 5 FOLDS: %0.4f (+/- %0.4f)" % (cross_score.mean(), cross_score.std() * 2))
```

CROSSVALIDATION 5 FOLDS: 0.8541 (+/- 0.0116)

Đồ thị ROC curve với metric AUC đạt tới 0.93



### 5.2.3. Phương pháp Ensemble learning

```
ensemble_clf = VotingClassifier(
    estimators=[('nb', MultinomialNB()),
                ('linearSVM', LinearSVC(dual=False, tol=1e-3, penalty="l2", loss='squared_hinge')),
                ('rdf', RandomForestClassifier(n_estimators=200, max_depth=None, random_state=100))],
    voting='hard')
```

TRAIN	precision	recall	f1-score	support
1	0.832	0.852	0.842	2066
0	0.887	0.870	0.878	2749
accuracy			0.863	4815
macro avg	0.859	0.861	0.860	4815
weighted avg	0.863	0.863	0.863	4815

Time Train 14.996941566467285

```
cross_score = cross_val_score(clf, X_train, y_train, cv=5)
print("CROSSVALIDATION 5 FOLDS: %0.4f (+/- %0.4f)" % (cross_score.mean(), cross_score.std() * 2))
```

CROSSVALIDATION 5 FOLDS: 0.8698 (+/- 0.0153)

## 5.3. Huấn luyện mô hình Deep learning

- Nhóm tiến hành chia dữ liệu thành ba tập train, test và val theo tỉ lệ 63%, 30% và 7%

```

X_train, X_test, y_train, y_test = train_test_split(data.comment_normalize, data.label, test_size=0.3, random_state=42)

X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size=0.1, random_state=42)

print(X_train.count())
print(X_valid.count())
print(X_test.count())

10108
1124
4815

```

- Các bước áp dụng model của nhóm vào bài toán là:
  - Vector hóa từ
  - Xây dựng mô hình
  - Huấn luyện mô hình
- Nhóm sử dụng Tokenizer để tách comment thành các từ, xây dựng bộ từ điển với vocab\_size = 10000. Mỗi comment sẽ được vector hóa thành 1 vector có độ dài maxlen = 200.

```

tokenizer = text.Tokenizer(num_words=config.vocab_size, split=' ')
tokenizer.fit_on_texts(X_train)
X_train = tokenizer.texts_to_sequences(X_train)
X_valid = tokenizer.texts_to_sequences(X_valid)
X_test = tokenizer.texts_to_sequences(X_test)
X_train = sequence.pad_sequences(X_train, maxlen=config.maxlen)
X_valid = sequence.pad_sequences(X_valid, maxlen=config.maxlen)
X_test = sequence.pad_sequences(X_test, maxlen=config.maxlen)

```

Ví dụ 1 vector comment

```

array([[ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        10,  9, 129, 341, 11], dtype=int32)

```

## Xây dựng mô hình

```
Model: "sequential_7"
```

Layer (type)	Output Shape	Param #
embedding_7 (Embedding)	(None, 200, 50)	500000
conv1d_6 (Conv1D)	(None, 198, 10)	1510
max_pooling1d_6 (MaxPooling 1D)	(None, 99, 10)	0
lstm_6 (LSTM)	(None, 10)	840
dense_6 (Dense)	(None, 1)	11

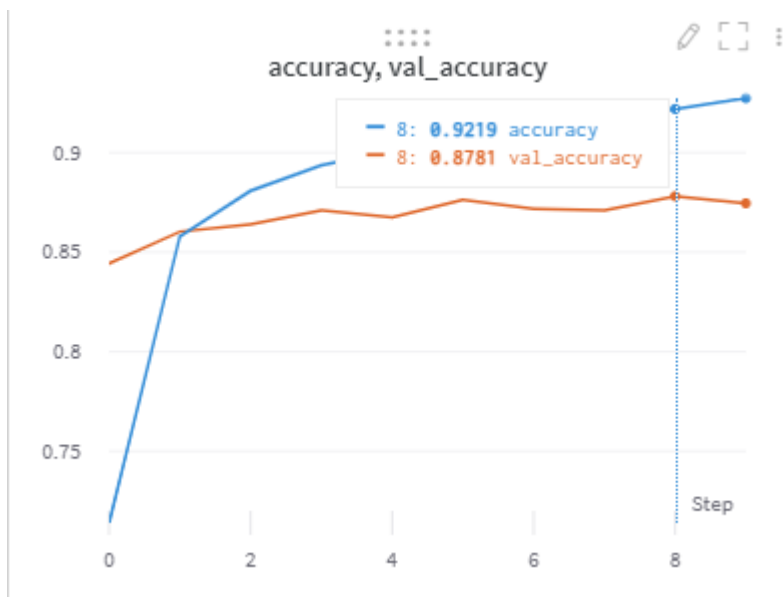
```
=====  
Total params: 502,361  
Trainable params: 502,361  
Non-trainable params: 0
```

- Kiến trúc của mô hình bao gồm 4 tầng chính là:
  - Embedding layer
  - Conv với kernel\_size = 3
  - Max\_pooling
  - LSTM
  - Dense
- Huấn luyện mô hình với
  - epoch: 10
  - batch\_size: 32
  - loss\_function: binary\_crossentropy
  - optimizer: rmsprop
- Quá trình huấn luyện:

```

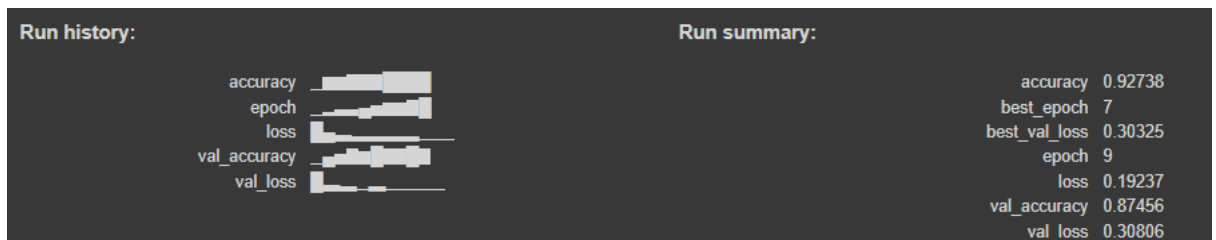
Epoch 1/10
316/316 [=====] - 16s 46ms/step - loss: 0.5540 - accuracy: 0.7140 - val_loss: 0.3818 - val_accuracy: 0.8443
Epoch 2/10
316/316 [=====] - 14s 45ms/step - loss: 0.3426 - accuracy: 0.8579 - val_loss: 0.3291 - val_accuracy: 0.8603
Epoch 3/10
316/316 [=====] - 14s 45ms/step - loss: 0.2946 - accuracy: 0.8809 - val_loss: 0.3157 - val_accuracy: 0.8639
Epoch 4/10
316/316 [=====] - 15s 46ms/step - loss: 0.2675 - accuracy: 0.8937 - val_loss: 0.3067 - val_accuracy: 0.8710
Epoch 5/10
316/316 [=====] - 14s 46ms/step - loss: 0.2490 - accuracy: 0.9009 - val_loss: 0.3118 - val_accuracy: 0.8674
Epoch 6/10
316/316 [=====] - 14s 45ms/step - loss: 0.2346 - accuracy: 0.9081 - val_loss: 0.3053 - val_accuracy: 0.8763
Epoch 7/10
316/316 [=====] - 14s 45ms/step - loss: 0.2223 - accuracy: 0.9148 - val_loss: 0.3045 - val_accuracy: 0.8719
Epoch 8/10
316/316 [=====] - 14s 45ms/step - loss: 0.2109 - accuracy: 0.9189 - val_loss: 0.3032 - val_accuracy: 0.8710
Epoch 9/10
316/316 [=====] - 14s 44ms/step - loss: 0.2013 - accuracy: 0.9219 - val_loss: 0.3057 - val_accuracy: 0.8781
Epoch 10/10
316/316 [=====] - 14s 45ms/step - loss: 0.1924 - accuracy: 0.9274 - val_loss: 0.3081 - val_accuracy: 0.8746

```



Độ chính xác trên tăng lên đến 87% sau 9 epochs! Tuy nhiên sau đó model liên tục giảm dần, còn độ chính xác tập train thì vẫn tiếp tục tăng. Do vậy có thể model đã bị "overfitting" sau epoch thứ 9.

Model hoạt động tốt nhất ở epoch thứ 9





## 5.4. Độ đo

- Để đánh giá hiệu quả mô hình, nhóm sử dụng chỉ số Accuracy và các chỉ số liên quan như Precision, Recall, F1
- Precision được định nghĩa là tỉ lệ số điểm Positive mô hình dự đoán đúng trên tổng số điểm mô hình dự đoán là Positive. Recall được định nghĩa là tỉ lệ số điểm Positive mô hình dự đoán đúng trên tổng số điểm thật sự là Positive (hay tổng số điểm được gán nhãn là Positive ban đầu).
- Precision càng cao, tức là số điểm mô hình dự đoán là positive đều là positive càng nhiều. Precision = 1, tức là tất cả số điểm mô hình dự đoán là Positive đều đúng, hay không có điểm nào có nhãn là Negative mà mô hình dự đoán nhầm là Positive.
- Recall càng cao, tức là số điểm là positive bị bỏ sót càng ít. Recall = 1, tức là tất cả số điểm có nhãn là Positive đều được mô hình nhận ra.
- F1-score là trung bình điều hòa (harmonic mean) của precision và recall (giả sử hai đại lượng này khác 0). F1-score được tính theo công thức:

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

## Chương 6. Đánh giá kết quả thực nghiệm

STT	Mô hình	Accuracy	Time Training
1	Naive Bayes	0.853	0.158
2	SVM	0.862	0.711
3	Random Forest	0.854	14.5
4	Ensemble	0.871	14.9
5	LSTM	0.874	146

## **Chương 7. Kết luận và hướng phát triển**

### **7.1 Kết luận**

- Các mô hình càng phức tạp thì có độ hiệu quả càng cao, nhưng kèm theo đó thời gian training và thời gian dự đoán lâu hơn.
- Nhìn kết quả thực nghiệm, ta có thể thấy khi sử dụng thêm các mô hình phức tạp như LSTM cũng không nâng cao độ chính xác nhiều. Điều đó nói rằng nếu muốn tăng độ chính xác cho bài toán, ta cần tập trung vào cải thiện chất lượng của dữ liệu hơn là mô hình, bằng cách tăng cường thêm dữ liệu hay cải tiến tiền xử lý dữ liệu,...

### **7.2 Hướng phát triển trong tương lai**

- Áp dụng thêm một số mô hình phân loại sâu hơn như CNN, BiLSTM, PhoBert,...
- Làm sạch dữ liệu hơn nữa

## Tài liệu tham khảo

- <https://machinelearningcoban.com/>
- <https://viblo.asia/p/phan-tich-phan-hoi-khach-hang-hieu-qua-voi-machine-learningvietnamese-sentiment-analysis-Eb85opXOK2G>
- <https://github.com/NLP-Projects/Vietnamese-Sentiment-Analysis>

## Phân công công việc

STT	Công việc	Thành viên	Mức độ hoàn thành
1	Tìm hiểu đề tài	Cả nhóm	100%
2	Tìm hiểu tập dữ liệu và cách đánh giá mô hình	Cả nhóm	100%
3	Khám phá tập dữ liệu	Triệu Quang Mạnh	90%
4	Tiền xử lý dữ liệu	Nguyễn Đình Mạnh	90%
5	Tìm hiểu và thử nghiệm mô hình Naive Bayes	Trần Trọng Khang	100%
6	Tìm hiểu và thử nghiệm mô hình SVM	Đỗ Thành Đức	100%
7	Tìm hiểu và thử nghiệm mô hình Random Forest	Triệu Quang Mạnh	100%
8	Tìm hiểu và thử nghiệm phương pháp Ensemble Learning	Nguyễn Đình Mạnh	100%
9	Tìm hiểu và thử nghiệm mô hình LSTM	Nguyễn Đình Mạnh	100%
10	Hoàn thiện báo cáo bản mềm	Nguyễn Đình Mạnh	90%
11	Hoàn thiện Slide báo cáo	Trần Trọng Khang, Đỗ Thành Đức	90%
12	Trình bày thuyết trình về đề tài	Trần Trọng Khang, Đỗ Thành Đức	90%