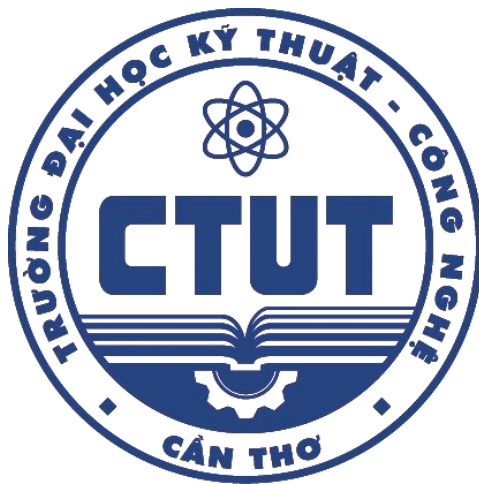


**TRƯỜNG ĐẠI HỌC KỸ THUẬT – CÔNG NGHỆ CẦN THƠ**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN KHOA HỌC MÁY TÍNH 4**

**NGHIÊN CỨU KỸ THUẬT TRÍCH XUẤT ĐẶC  
TRƯNG TRONG NHẬN DẠNG VĂN BẢN**

Giảng viên hướng dẫn:

**TS: Hà Lê Ngọc Dung**

Sinh viên thực hiện:

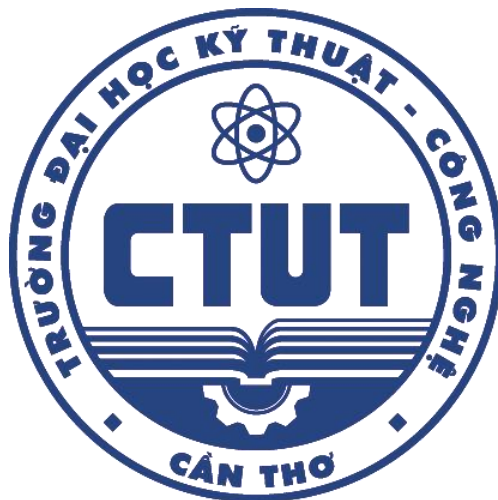
**Từ Thái Bảo**

**MSSV: 1900222**

**Lớp: KHMT0119**

**Cần Thơ – 5/2023**

**TRƯỜNG ĐẠI HỌC KỸ THUẬT – CÔNG NGHỆ CẦN THƠ**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN KHOA HỌC MÁY TÍNH 4**

**NGHIÊN CỨU KỸ THUẬT TRÍCH XUẤT ĐẶC  
TRÙNG TRONG NHẬN DẠNG VĂN BẢN**

Giảng viên hướng dẫn:

**TS: Hà Lê Ngọc Dung**

Sinh viên thực hiện:

**Từ Thái Bảo**

**MSSV: 1900222**

**Lớp: KHMT0119**

**Cần Thơ – 5/2023**

## **NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN HƯỚNG DẪN**

**Đồ án Khoa học máy tính 4**

**Đề tài: NGHIÊN CỨU KỸ THUẬT TRÍCH XUẤT ĐẶC TRƯNG TRONG NHẬN DẠNG VĂN BẢN**

**Sinh viên thực hiện:** Từ Thái Bảo

**MSSV:** 1900222

**Ngành:** Khoa học máy tính - 2019

**Giảng viên hướng dẫn:** TS Hà Lê Ngọc Dung

Nhận xét, đánh giá:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

....., ngày .... tháng .... năm 2023

**Giảng viên hướng dẫn**

## NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN PHẢN BIỆN

**Đồ án Khoa học máy tính 4**

**Đề tài: NGHIÊN CỨU KỸ THUẬT TRÍCH XUẤT ĐẶC TRƯNG TRONG NHẬN DẠNG VĂN BẢN**

**Sinh viên thực hiện:** Từ Thái Bảo

**MSSV:** 1900222

**Ngành:** Khoa học máy tính - 2019

**Giảng viên phản biện:**.....

Nhận xét, đánh giá:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

....., ngày .... tháng .... năm 2023

## LỜI CAM ĐOAN

Em xin cam đoan rằng đồ án này là công trình nghiên cứu của chính bản thân em thực hiện dựa trên những kiến thức đã được học, nghiên cứu và tìm hiểu một số đề tài và các phương án đi trước, không sao chép từ bất cứ công trình đã có trước đó. Mọi thứ được dựa trên sự cố gắng cũng như sự nỗ lực của bản thân.

Những phần có sử dụng tài liệu tham khảo có trong đồ án đã được liệt kê và nêu rõ ra tại phần tài liệu tham khảo.

*Cần Thơ, ngày 6 tháng 5 năm 2023*

Sinh viên thực hiện



Từ Thái Bảo

## LỜI CẢM ƠN

Được sự giúp đỡ nhiệt tình của quý thầy cô, bộ môn Khoa học máy tính, khoa Công nghệ thông tin, trường Đại học Kỹ thuật - Công nghệ Cần Thơ, những người đã dìu dắt em tận tình, đã truyền đạt cho em những kiến thức và những bài học quý giá trong suốt thời gian em theo học tại trường. Sau gần ba tháng nghiên cứu em đã hoàn thành đề tài: **“NGHIÊN CỨU KỸ THUẬT TRÍCH XUẤT ĐẶC TRƯNG TRONG NHẬN DẠNG VĂN BẢN”**.

Em chân thành gửi lời cảm ơn đến tất cả các thầy cô trong khoa Công Nghệ Thông Tin đã hỗ trợ chúng em hoàn thành tốt đồ án lần này. Đặc biệt nhất là em muốn dành lời cảm ơn đến cô Hà Lê Ngọc Dung, người đã tận tâm hướng dẫn và trực tiếp giúp đỡ em trong suốt quá trình nghiên cứu đồ án. Với sự chỉ bảo nhiệt tình của cô, em đã có định hướng tốt trong việc triển khai và thực hiện các yêu cầu trong quá trình làm đồ án.

Để có thể hoàn thành tốt được đề tài, ngoài này sự nỗ lực học hỏi của bản thân, còn có sự hướng dẫn tận tình từ quý thầy cô và sự giúp đỡ tận tình từ gia đình và bạn bè đã luôn tạo cho em những điều kiện tốt nhất trong suốt quá trình học tập và nghiên cứu.

Xin chân thành cảm ơn!

## TÓM TẮT ĐỒ ÁN

Đây là đồ án phục vụ cho việc nghiên cứu của sinh viên tìm hiểu về “**Nghiên cứu kỹ thuật trích xuất đặc trưng trong nhận dạng văn bản**” tập trung vào việc nghiên cứu các kỹ thuật trích xuất đặc trưng từ văn bản để giúp máy tính hiểu và nhận dạng văn bản một cách chính xác.

Đồ án tập trung vào các phương pháp trích xuất đặc trưng từ văn bản. Đồ án cũng tập trung vào việc áp dụng các kỹ thuật này để giải quyết các vấn đề nhận dạng văn bản thực tế.

Sâu hơn, đồ án tập trung vào các phương pháp trích xuất đặc trưng phổ biến như Bag of Words, TF-IDF và Word2Vec, cùng với việc ứng dụng chúng vào bài toán phân loại văn bản. Kết quả đạt được từ đồ án sẽ giúp hiểu rõ hơn về cách máy tính "đọc" và xử lý thông tin từ văn bản, đồng thời cung cấp cho họ kiến thức cơ bản để tiếp cận các bài toán phân loại văn bản phức tạp hơn trong tương lai.

Bố cục của đề tài nghiên cứu bao gồm 5 phần:

- **Chương I: Tổng quan về nhận dạng văn bản**
- **Chương II. Một số phương pháp trích xuất đặc trưng trong nhận dạng văn bản**
- **Chương III: Ứng dụng các phương pháp trích xuất đặc trưng vào bài toán phân loại văn bản**
- **Chương IV: Kết luận**
- **Tài liệu tham khảo**

## MỤC LỤC

NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN HƯỚNG DẪN.....	i
NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN PHẢN BIỆN.....	ii
LỜI CAM ĐOAN.....	iii
LỜI CẢM ƠN.....	iv
TÓM TẮT ĐỒ ÁN.....	v
MỤC LỤC .....	vi
DANH MỤC TỪ VIẾT TẮT, THUẬT NGỮ .....	vii
DANH MỤC BẢNG BIỂU.....	viii
DANH MỤC HÌNH ẢNH.....	ix
CHƯƠNG I: TỔNG QUAN VỀ NHẬN DẠNG VĂN BẢN.....	1
1.1. Tổng quan.....	1
1.2. Nhận dạng văn bản.....	1
CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP TRÍCH SUẤT ĐẶC TRƯNG TRONG NHẬN DẠNG VĂN BẢN.....	3
2.1. Phương pháp BoW .....	3
2.2. Phương pháp TF – IDF .....	4
2.3. Phương pháp World2Vec.....	7
2.3.1. Mô hình CBOW.....	8
2.3.2. Mô hình Skip-gram.....	9
CHƯƠNG III: ỨNG DỤNG CÁC PHƯƠNG PHÁP TRÍCH SUẤT ĐẶC TRƯNG VÀO BÀI TOÁN PHÂN LOẠI VĂN BẢN.....	12
3.1. Chuẩn bị dữ liệu .....	12
3.2. Trích xuất đặc trưng. ....	13
3.3. Xây dựng mô hình.....	16
3.4. Đánh giá kết quả.....	17
3.5. Triển khai ứng dụng.....	18
CHƯƠNG IV: KẾT LUẬN.....	22
4.1. Kết quả đạt được .....	22
4.2. Hạn chế.....	22
4.3. Hướng phát triển .....	22
TÀI LIỆU THAM KHẢO .....	23
PHỤ LỤC CODE .....	24



## DANH MỤC TỪ VIẾT TẮT, THUẬT NGỮ

STT	Từ viết tắt	Tên đầy đủ	Dịch nghĩa
1	BoW	Bag of Words	Phương pháp trích suất đặc trưng văn bản
2	TF-IDF	Term Frequency-Inverse Document Frequency	Phương pháp trích suất đặc trưng văn bản
3	UIT-VSFC	Vietnamese Students' Feedback Corpus for Sentiment Analysis	Bộ dữ liệu văn bản

## DANH MỤC BẢNG BIỂU

Bảng 2. 1: Ví dụ về tập dữ liệu văn bản của bộ dữ liệu UIT-VSFC .....	4
Bảng 2. 2: Biểu diễn tập dữ liệu văn bản bằng mô hình BoW .....	4
Bảng 2. 3: Biểu diễn Doc1 và Doc2 theo số lần xuất hiện của từ .....	6
Bảng 2. 4: Biểu diễn Doc1 và Doc2 theo TF .....	6
Bảng 2. 5: Biểu diễn Doc1 và Doc 2 theo IDF.....	6
Bảng 2. 6: Biểu diễn văn bản Doc1 và Doc2 theo TF-IDF .....	7
Bảng 3. 1: Thống kê số lượng dữ liệu chi tiết .....	13
Bảng 3. 2: Các tham số SVM đối với từng phương pháp .....	17
Bảng 3. 3: Kết quả của các tiêu chí đánh giá.....	18

## DANH MỤC HÌNH ẢNH

Hình 2. 1: Ví dụ về biểu diễn vector có cùng kích thước.....	8
Hình 2. 2: Mô hình CBOW dạng tổng quát.....	8
Hình 2. 3: Minh họa đầu vào và đầu ra của mô hình CBOW .....	9
Hình 2. 4: Mô hình Skip-gram dạng tổng quát.....	10
Hình 2. 5: Minh họa đầu vào và đầu ra của mô hình Skip-gram.....	10
Hình 3. 1: Trực quan hoá về tập dữ liệu UIT-VSFC.....	12
Hình 3. 2: Tập dữ liệu sau khi đã xoá nhãn Trung Tính .....	13
Hình 3. 3: Vector hoá và kích thước của tập train BoW .....	14
Hình 3. 4: Vector hoá và kích thước của tập train TF-IDF .....	14
Hình 3. 5: Kích thước vector sau khi sử dụng SVD.....	15
Hình 3. 6: Vector hoá 1 từ trong Word2vec .....	16
Hình 3. 7: Similar word2vec với Skipgram và CBOW .....	16
Hình 3. 8: Kiến trúc của mô hình Deep Learning .....	17
Hình 3.9: Confusion Matrix đối với từng phương pháp.....	18
Hình 3. 10: Giao diện của ứng dụng phân loại văn bản .....	19
Hình 3. 11: Giao diện của chương trình khi dự đoán nhãn là tích cực.....	20
Hình 3. 12: Giao diện của chương trình khi dự đoán nhãn là tiêu cực.....	21

# CHƯƠNG I: TỔNG QUAN VỀ NHẬN DẠNG VĂN BẢN

## 1.1. Tổng quan

Phân loại văn bản là tiến trình xếp các tài liệu văn bản vào trong một hoặc nhiều phân loại hoặc lớp các tài liệu tương tự xác định trước. Quá trình phân loại văn bản nhằm mục đích xác định một văn bản cho trước thuộc lớp ngữ nghĩa được xác định trước. Bài toán nhận dạng văn bản, thực chất có thể xem là bài toán phân lớp. Phân loại văn bản tự động là việc gán các nhãn phân loại lên một văn bản mới dựa trên mức độ tương tự của văn bản đó so với các văn bản đã được gán nhãn trong tập huấn luyện.

Nhiệm vụ của nhận dạng văn bản là xếp các tài liệu văn bản vào trong các phân loại của các tài liệu với các yêu cầu cao hơn để thu nhận nhanh hơn các tài liệu đó và cung cấp các lĩnh vực trong đó người dùng có thể khảo sát sâu hơn các tài liệu tương tự. Trước đây, các hệ thống thu nhận thông tin sử dụng các biểu đồ phân loại truyền thống trong khi hầu hết các giải thuật phân nhóm sử dụng mô hình không gian vector để hình thức hoá các nhóm tài liệu.

Thông thường phân loại văn bản thường được sử dụng dưới hai hình thức:

- Phân loại theo chủ đề: đây là cách phân loại dựa vào chủ đề mà văn bản có thể thuộc vào. Tập văn bản được phân thành các chủ đề khác nhau.

Ví dụ: Giáo dục, Thể Thao, Du Lịch, Sức Khỏe,...

- Phân loại theo ngữ nghĩa: đây là cách phân loại dựa vào ngữ nghĩa của văn bản để phân loại chúng.

Ví dụ: Phân tích cảm xúc, Spam hay không spam, được đề nghị hay không đề nghị,...

Các nhà nghiên cứu đã thực hiện nhiều kỹ thuật học máy như và khai phá dữ liệu đã được áp dụng vào bài toán phân loại văn bản, chẳng hạn: phương pháp quyết định dựa vào bộ phân loại: Naive Bayes, Decision Tree, KNN, Neural network,... Các phương pháp thu được các kết quả khả quan. Các nhà nghiên cứu tập trung vào tăng độ chính xác và hiệu năng như giảm chiều dữ liệu, sử dụng các tập dữ liệu nhỏ,... Trên thế giới đã có nhiều công trình nghiên cứu đạt những kết quả khả quan, nhất là đối với phân loại văn bản tiếng Anh. Tuy vậy, các nghiên cứu và ứng dụng đối với văn bản tiếng Việt còn nhiều hạn chế do khó khăn về tách từ và câu. Nên các bước phân loại văn bản cho các ngôn ngữ khác sẽ biến đổi về mức độ chính xác do khó đồng nhất mọi ngôn ngữ.

## 1.2. Nhận dạng văn bản

Bài toán nhận dạng văn bản được xem dưới góc nhìn toán học như sau:

- $D$  là tập các văn bản:  $D=\{d_1, d_2, \dots, d_n\}$
- $C$  là các chủ đề:  $C=\{c_1, c_2, \dots, c_n\}$

Mục tiêu của bài toán là tìm hàm  $f_{DC}$  được định nghĩa như sau:

$$fDC = \{0,1\} \rightarrow Boolean$$

Với:

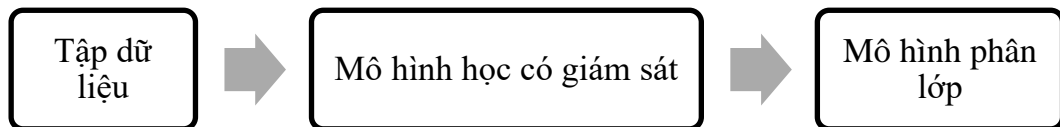
- $f(d, c) = 1$ , nếu  $d_j$  thuộc lớp  $c_i$ .
- $f(d, c) = 0$ , nếu  $d_j$  không thuộc lớp  $c_i$ .

Vấn đề nhận dạng văn bản theo phương pháp thống kê dựa trên kiểu học có giám sát được đặc tả bao gồm 2 giai đoạn : Giai đoạn huấn luyện và giai đoạn phân lớp

– **Giai đoạn huấn luyện.**

Bước đầu tiên, bắt đầu với tập huấn luyện, mỗi phân tử trong tập huấn luyện được gán vào một hoặc nhiều lớp sẽ thể hiện bằng một mô hình mã hoá. Thông thường, mỗi phân tử trong tập huấn luyện được thể hiện theo dạng vector biểu diễn cho văn bản trong tập huấn luyện.

Giai đoạn huấn luyện trong nhận dạng văn bản được tổng quát hoá như hình sau:

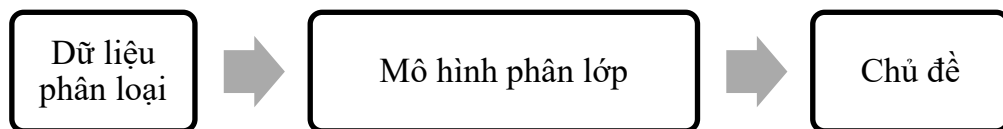


Đầu vào : Dữ liệu huấn luyện và Mô hình học có giám sát.

Đầu ra : Mô hình phân lớp.

– **Giai đoạn phân loại.**

Sau khi đã hoàn thành giai đoạn huấn luyện, mô hình phân loại sẽ được áp dụng cho các văn bản mới cần phân loại.



Đầu vào : Dữ liệu cần phân loại và Mô hình phân lớp.

Đầu ra : Chủ đề của văn bản.

## CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP TRÍCH SUẤT ĐẶC TRƯNG TRONG NHẬN DẠNG VĂN BẢN

Mỗi đối tượng nói chung đều có những đặc trưng riêng, đặc trưng chính là yếu tố giúp phân biệt đối tượng này với đối tượng khác. Tóm lại đặc trưng là các yếu tố xác định nên đối tượng. Một đối tượng chỉ được xác định khi có đủ số đặc trưng xác định nên nó.

Trích xuất đặc trưng là tìm ra điểm đặc trưng của đối tượng so với đối tượng khác tùy theo mục đích. Là quá trình chọn lọc một tập con chứa các thuộc tính liên quan để sử dụng trong quá trình xây dựng mô hình.

Trích suất đặc trưng trong văn bản là một phương pháp được sử dụng để tóm tắt và giới thiệu các thông tin quan trọng và ý nghĩa trong một văn bản. Thông thường, trích suất đặc trưng được sử dụng để trích xuất những câu hoặc đoạn văn trong một tài liệu mà chứa đựng nhiều thông tin quan trọng và cần được truyền đạt.

### 2.1. Phương pháp BoW

BoW là viết tắt của Bag of Words là một dạng biểu diễn đơn giản được sử dụng trong xử lý ngôn ngữ tự nhiên và tìm kiếm thông tin. Trong mô hình này, một văn bản được biểu diễn như là túi của các từ của nó, không quan tâm đến ngữ pháp và thậm chí cả thứ tự.

Quá trình trích xuất đặc trưng của một văn bản bao gồm tách từ và đếm số lần xuất hiện của các từ trong văn bản. Mô hình BoW chỉ tập hợp tất cả các từ dạng một từ duy nhất, không chứa các cụm từ gồm nhiều từ ghép lại. Như vậy, mô hình BoW là mô hình biểu diễn văn bản như vector tần số xuất hiện của từ trong văn bản, được sử dụng phổ biến hiện nay trong vấn đề phân lớp văn bản

Trong đó, từ điển được tạo thành từ tập tất cả các từ trong tập dữ liệu. Mỗi tài liệu (có thể là câu, đoạn hoặc văn bản) trong tập dữ liệu được biểu diễn dưới dạng vector đặc trưng, vector này có số chiều bằng với số từ có trong từ điển. Ví dụ, nếu tập dữ liệu có  $n$  từ thì vector của mỗi tài liệu trong tập dữ liệu sẽ có  $n$  chiều, mỗi từ khác nhau trong văn bản sẽ là một đặc trưng và tần số xuất hiện của nó trong văn bản là giá trị của đặc trưng tương ứng. Trị thành phần của vector là tần số xuất hiện của từ trong tài liệu.

Do dữ liệu văn bản ở đầu vào ở dạng không cấu trúc, trong khi các giải thuật máy học ở giai đoạn tiếp theo sau thường chỉ có thể xử lý được dữ liệu dạng cấu trúc bảng (mỗi dòng là một phần tử dữ liệu, cột là chiều hay thuộc tính). Để giải quyết vấn đề này, mô hình BoW cho phép biểu diễn tập dữ liệu văn bản về cấu trúc bảng.

STT	Nội dung	Lớp
1	thầy giảng bài hay , có nhiều bài tập ví dụ ngay trên lớp.	Tích cực
2	em sẽ nợ môn này , nhưng em sẽ học lại ở các học kỳ kế tiếp.	Trung tính

...	.....	...
n	thời lượng học quá dài , không đảm bảo tiếp thu hiệu quả .	Tiêu cực

**Bảng 2. 1: Ví dụ về tập dữ liệu văn bản của bộ dữ liệu UIT-VSFC**

Bước tiền xử lý này bao gồm việc phân tích từ vựng và tách các từ trong nội dung của tập văn bản, sau đó chọn tập hợp các từ có ý nghĩa quan trọng dùng để phân lớp, biểu diễn dữ liệu văn bản về dạng bảng để từ đó các giải thuật máy học có thể học để phân lớp. Ở bước phân tích từ vựng, công việc có thể là quy về từ gốc của các biến thể từ, có thể xóa bỏ các từ không có ý nghĩa cho việc phân lớp như các mạo từ, từ nối,... Tiếp đến là tách các từ, đưa vào từ điển. Một văn bản được biểu diễn dạng vector (có n thành phần, chiều) mà giá trị thành phần thứ j là tần số xuất hiện từ thứ j trong văn bản. Nếu xét tập T gồm m văn bản và từ điển có n từ vựng, thì T có thể được biểu diễn thành bảng D kích thước  $m \times n$ , dòng thứ i của bảng là vector biểu diễn văn bản thứ i tương ứng.

Xem ví dụ tập dữ liệu văn bản trong **Bảng 2.1**, sau khi tiền xử lý biểu diễn với mô hình BoW thu được bảng dữ liệu D có cấu trúc như **Bảng 2.2**, từ bảng này mô hình máy học có thể xử lý vấn đề phân lớp.

STT	1 (thầy)	2 (giảng)	3 (bài)	...	n (quả)	Lớp
1	1	1	1	...	0	Tích cực
2	0	0	0	...	0	Trung tính
...	...	...	...	...	...	...
3	0	0	0	...	1	Tiêu cực

**Bảng 2. 2: Biểu diễn tập dữ liệu văn bản bằng mô hình BoW**

Nhược điểm của mô hình BoW nằm ở chỗ không xác định đến sự đồng nghĩa của từ, điều này dẫn đến làm giảm hiệu quả dự đoán của mô hình máy học phân lớp văn bản và có thể cho kết quả với độ chính xác không cao. Bên cạnh đó, với kích thước của bộ từ điển lớn. Kích thước của các vector theo kích thước của từ điển. Để khắc phục, ta thường thực hiện việc rút gọn chiều dữ liệu. Phương pháp rút gọn có thể là lựa chọn những từ quan trọng nhất để có thể phân biệt văn bản này với văn bản khác, hay phương pháp giảm chiều.

## 2.2. Phương pháp TF – IDF

TF – IDF là viết tắt của Term Frequency – Inverse Document Frequency. Là một phương thức thống kê được biết đến rộng rãi để xác định độ quan trọng của một từ trong đoạn văn bản trong một tập nhiều đoạn văn bản khác nhau.

TF-IDF xác định 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. Giá trị TF-IDF cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu.

TF (Term Frequency) – Tần suất xuất hiện của từ là số lần từ xuất hiện trong văn bản. Vì các văn bản có thể có độ dài ngắn khác nhau nên một số từ có thể xuất hiện nhiều lần trong một văn bản dài hơn là một văn bản ngắn. Như vậy, TF thường được chia cho độ dài văn bản (tổng số từ trong một văn bản).

$$TF(t, d) = \frac{f(t, d)}{\sum \{f(w, d) \mid w \in d\}}$$

Trong đó:

- $TF(t, d)$ : là tần suất xuất hiện của từ  $t$  trong văn bản  $d$ .
- $f(td)$ : là số lần xuất hiện của từ trong văn bản  $d$ .
- $\sum \{f(w, d) \mid w \in d\}$ : Tổng số từ xuất hiện trong văn bản  $d$ .

IDF (Inverse Document Frequency) – Nghịch đảo tần suất của văn bản, giúp đánh giá tầm quan trọng của một từ. Khi tính tần số xuất hiện TF thì các từ đều được coi là quan trọng như nhau. Tuy nhiên có một số từ thường được sử dụng nhiều nhưng không quan trọng để thể hiện ý nghĩa của đoạn văn. Vì vậy ta cần giảm đi mức độ quan trọng của những từ đó bằng cách sử dụng IDF:

$$IDF(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

Trong đó:

- $IDF(t, D)$ : là giá trị IDF của từ  $t$  trong tập văn bản  $D$ .
- $|D|$ : Tổng số văn bản trong tập  $D$ .
- $|\{d \in D : t \in d\}|$ : Thể hiện số văn bản  $D$  có chứa từ  $t$ .

Cơ số logarit không làm thay đổi giá trị IDF của từ mà chỉ thu hẹp khoảng giá trị của từ đó. Việc sử dụng logarit nhằm giúp giá trị TF-IDF của một từ nhỏ hơn, do công thức tính TF-IDF của một từ trong một văn bản là tích của TF và IDF của từ đó. Công thức tính TF-IDF được xác định như sau:

$$TF-IDF(t, d, D) = TF(t, d) * IDF(t, D)$$

Những từ có giá trị TF-IDF cao là những từ xuất hiện nhiều trong văn bản này, và xuất hiện ít trong các văn bản khác. Việc này giúp lọc ra những từ phổ biến và giữ lại những từ có giá trị cao chính là các từ khóa của văn bản đó.

Ví dụ: cho 3 văn bản sau:

**Doc1:** thầy giảng bài hay, có nhiều bài tập ví dụ ngay trên lớp.

**Doc2:** thời lượng học quá dài, không đảm bảo tiếp thu hiệu quả.



**Bước 1: Tính TF –Tuần số xuất hiện của một từ trong một văn bản**

<b>Doc1</b>	thầy	giảng	bài	hay	có	nhiều	tập	ví	dụ	ngay	trên	lớp
<b>TF</b>	1	1	2	1	1	1	1	1	1	1	1	1

<b>Doc2</b>	thời	lượng	học	quá	dài	không	đảm	bảo	tiếp	thu	hiệu	quả.
<b>TF</b>	1	1	1	1	1	1	1	1	1	1	1	1

**Bảng 2. 3: Biểu diễn Doc1 và Doc2 theo số lần xuất hiện của từ**

Do tổng số từ trong 3 văn bản khác nhau. Trong đó văn bản nào có kích thước lớn thì tần số xuất hiện của một từ có thể sẽ nhiều hơn so với các văn bản có kích thước nhỏ hơn. Để giải quyết việc này cần chuẩn hóa tần số xuất hiện của một từ trong văn bản dựa theo kích thước của văn bản đó. Để làm việc này ta chia số lần xuất hiện của một từ cho tổng số từ trong văn bản. Cụ thể sau khi chuẩn hóa được như sau:

<b>Doc1</b>	thầy	giảng	bài	hay	có	nhiều	tập	ví	dụ	ngay	trên	lớp
<b>TF</b>	0.07	0.07	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07

<b>Doc2</b>	thời	lượng	học	quá	dài,	không	đảm	bảo	tiếp	thu	hiệu	quả.
<b>TF</b>	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08

**Bảng 2. 4: Biểu diễn Doc1 và Doc2 theo TF****Bước 2: Tính IDF – tần số nghịch của một từ trong một văn bản**

Ta biết rằng trong văn bản có một số từ xuất hiện nhiều nhưng vai trò quyết định đến độ liên quan giữa hai văn bản lại không cao. Để giảm việc này cần tính IDF cho từng từ trong tập văn bản của chúng ta (ở đây gồm 2 văn bản Doc1 và Doc2).

$$\begin{aligned} \text{IDF}(\text{thầy}) &= \log(\text{Tổng số văn bản} / \text{số văn bản chứa từ "thầy"}) \\ &= \log(2/1) = 0.301 \end{aligned}$$

Tương tự ta có bảng IDF cho các từ thuộc tập văn bản ta đang xét là Doc1 và Doc2

<b>Từ</b>	<b>IDF</b>
thầy	0.3
giảng	0.3
bài	0.3
....	...
hiệu	0.3
quả	0.3

**Bảng 2. 5: Biểu diễn Doc1 và Doc 2 theo IDF**

### Bước 3: Tính TF\*IDF

Mục đích của ta là tìm xem Doc3 liên quan nhất tới Doc1 hay Doc2, có bảng sau:

Doc3: Mùa thu lá vàng rơi.

Doc1	thầy	giảng	bài	hay	có	nhiều	tập	ví	dụ	ngay	trên	lớp
TF	0.07	0.07	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
IDF	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
TF-IDF	0.021	0.021	0.045	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021

Doc2	thời	lượng	học	quá	dài,	không	đảm	bảo	tiếp	thu	hiệu	quả.
TF	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08
IDF	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
TF-IDF	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024

*Bảng 2. 6: Biểu diễn văn bản Doc1 và Doc2 theo TF-IDF*

### 2.3. Phương pháp Word2Vec

Để khắc phục nhược điểm của phương pháp thống kê là số chiều của một vector quá lớn (bằng độ dài của từ điển, có thể đến cả triệu từ) và không quan tâm đến ngữ nghĩa của văn bản. Tác giả Tomas Mikolov và các cộng sự đã công bố phương pháp Word2Vec trong bài báo “Distributed Representations of Words and Phrases and their Compositionality”. Đây là thuật toán theo phương pháp dự đoán (Prediction-based embedding), dự đoán học biểu diễn vector từ thông qua những từ ngữ cảnh xung quanh nhằm cải thiện khả năng dự đoán ý nghĩa các từ.

Word2Vec là phương pháp tiêu chuẩn để biểu diễn văn bản đó là biểu diễn các văn bản theo vector. Trong đó, các từ/cụm từ thuộc kho tài liệu ngôn ngữ được ánh xạ thành những vector trên hệ không gian số thực.

Word2vec là một mạng neural 2 lớp với duy nhất 1 tầng ẩn, lấy đầu vào là một ngữ liệu lớn và sinh ra không gian vector (với số chiều khoảng vài trăm), với mỗi từ duy nhất trong ngữ liệu được gắn với một vector tương ứng trong không gian. Các vector từ được xác định trong không gian vector sao cho những từ có chung ngữ cảnh trong ngữ liệu được đặt gần nhau trong không gian.

Nếu ta gán nhãn các thuộc tính cho một vector từ giả thiết, thì các vector được biểu diễn theo Word2Vec sẽ có dạng như sau:

		Vua	Hoàng hậu	Phụ nữ	Công chúa
Hoàng gia	<input type="text"/>	<input type="text" value="0.99"/>	<input type="text" value="0.99"/>	<input type="text" value="0.02"/>	<input type="text" value="0.98"/>

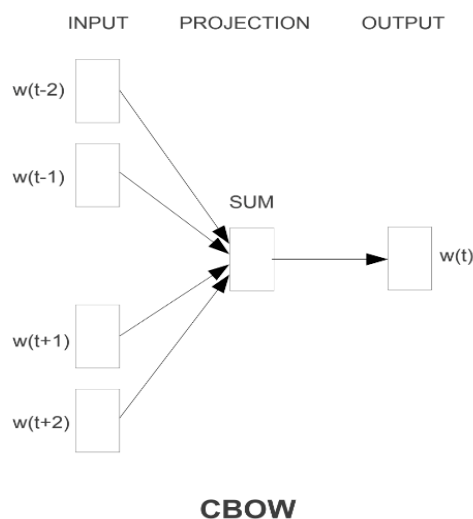
Nam tính		0.99	0.05	0.01	0.02
Nữ tính		0.05	0.93	0.99	0.94
Tuổi		0.7	0.6	0.5	0.1

**Hình 2. 1: Ví dụ về biểu diễn vector có cùng kích thước**

Mô hình Word2vec được hình thành từ hai thành phần đó là CBOW và Skip-gram.

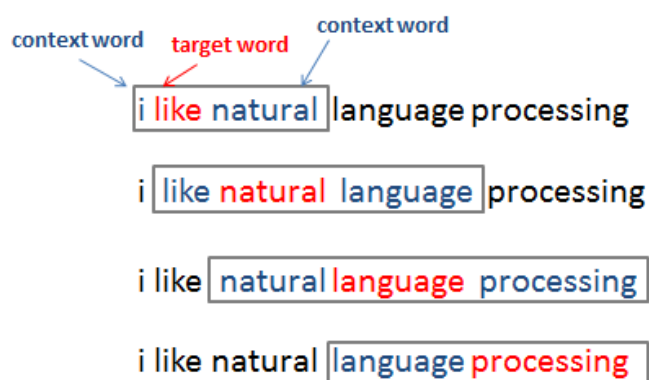
### 2.3.1. Mô hình CBOW

Mô hình CBOW, sử dụng từ ngữ cảnh (từ lân cận) để dự đoán từ đích. Trong thành phần này, mô hình cố gắng dự đoán từ hiện tại bằng cách sử dụng các từ lân cận trong văn bản đầu vào. Cụ thể, CBOW sẽ lấy các từ lân cận của từ hiện tại và đưa chúng vào một lớp ẩn. Lớp ẩn này sẽ tính toán các vector biểu diễn của các từ lân cận này và đưa ra vector biểu diễn của từ hiện tại.



**Hình 2. 2: Mô hình CBOW dạng tổng quát**

Ý tưởng của mô hình là dự đoán từ mục tiêu dựa vào các từ ngữ cảnh xung quanh nó trong một phạm vi nhất định. Cho từ mục tiêu  $w_c$  tại vị trí  $c$  trong câu văn bản, khi đó đầu vào là các từ ngữ cảnh ( $w_{c-m}, \dots, w_{c-1}, w_{c+1}, \dots, w_{c+m}$ ) xung quanh từ  $w_c$  trong phạm vi  $m$ .



Hình 2. 3: Minh họa đầu vào và đầu ra của mô hình CBOW

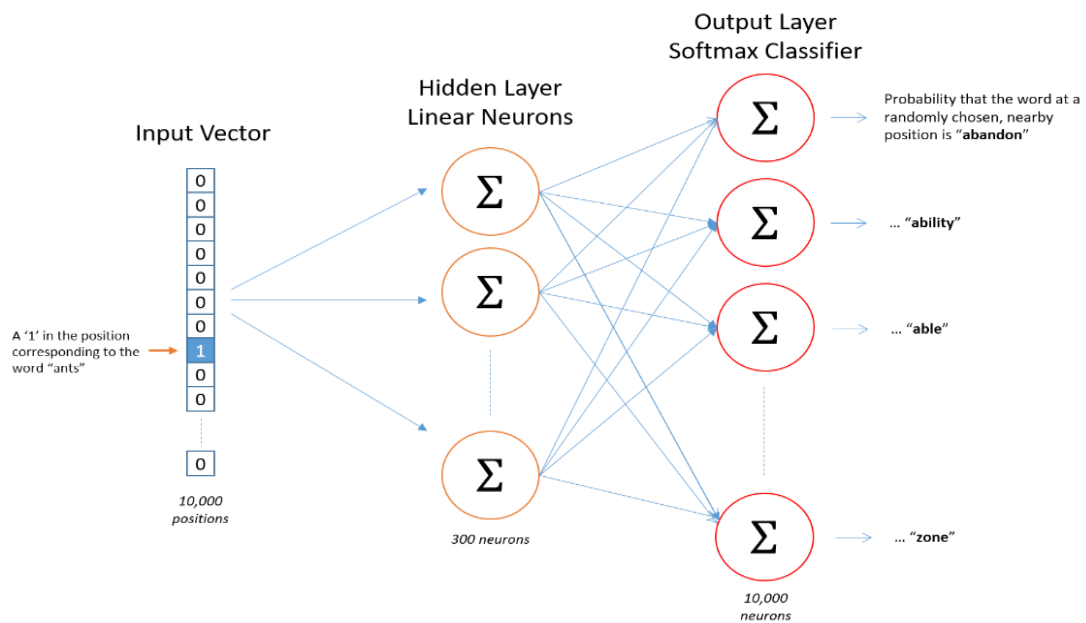
### Chi tiết về mô hình:

- Xây dựng bộ từ vựng.
- Biểu diễn mỗi từ thành các one-hot-vector.
- Input của mô hình CBOW là một tập hợp các từ lân cận của từ cần dự đoán.
- Mô hình CBOW sử dụng tầng ẩn với số lượng neuron là kích thước của embedding. Hàm kích hoạt trên tầng ẩn là hàm sigmoid hoặc tanh.
- Output của mô hình CBOW là một vector duy nhất, có kích thước bằng kích thước của bộ từ vựng, thể hiện xác suất của mỗi từ được dự đoán là từ cần tìm.
- Hàm kích hoạt trên tầng output là softmax.
- Trong quá trình huấn luyện, input của mô hình CBOW là một vector có kích thước bằng số lượng các từ lân cận của từ cần dự đoán, output là một one-hot-vector.
- Trong quá trình đánh giá sau khi huấn luyện, đầu ra của mô hình CBOW là một vector chứa embedding của từ cần tìm.

### 2.3.2. Mô hình Skip-gram

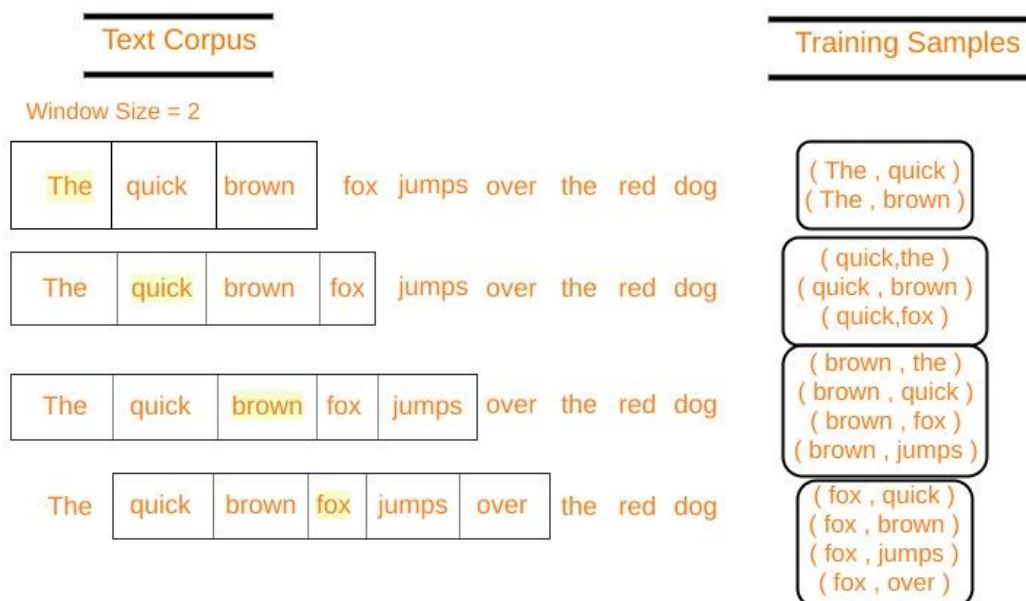
Mô hình Skip-gram sử dụng một từ để dự đoán ngữ cảnh mục tiêu hay các từ xung quanh xem xét những từ ngữ cảnh xung quanh sẽ được đánh giá tốt hơn so với những từ trong ngữ cảnh nhưng ở vị trí xa hơn. Cụ thể, skip-gram sẽ lấy một từ hiện tại và đưa nó vào một lớp ẩn. Lớp ẩn này sẽ tính toán vector biểu diễn của từ hiện tại. Sau đó, mô hình sẽ sử dụng vector biểu diễn này để dự đoán các vector biểu diễn của các từ lân cận.

Cho một từ cụ thể ở giữa câu, nhìn vào những từ ở gần và chọn ngẫu nhiên. Mạng neural sẽ cho ta biết xác suất của mỗi từ trong từ vựng về việc trở thành từ gần đó mà ta chọn. Dưới đây là mô hình kiến trúc của mạng Skip-gram và cách xây dựng Dữ liệu huấn luyện mô hình word embeddings.



Hình 2. 4: Mô hình Skip-gram dạng tổng quát

Giả sử khi xây dựng training data với windows size = 2. Ở đây windows được hiểu như một cửa sổ trượt qua mỗi từ. Windows size = 2 tức là lấy 2 từ bên trái và bên phải mỗi từ trung tâm.



Hình 2. 5: Minh họa đầu vào và đầu ra của mô hình Skip-gram

Chi tiết về mô hình

- Xây dựng bộ từ vựng.
- Biểu diễn mỗi từ thành các one-hot-vector.
- Đầu ra là một vector duy nhất, có kích thước bằng kích thước của bộ từ vựng, thể hiện xác suất của mỗi từ được là lân cận của từ đầu vào.
- Không có hàm kích hoạt trên tầng ẩn.

- Hàm kích hoạt trên tầng output là softmax.
- Trong quá trình huấn luyện, input là 1 one-hot-vector, output cũng là 1 one-hot-vector.
- Trong quá trình đánh giá sau khi huấn luyện, đầu ra phải là 1 phân bố xác suất.

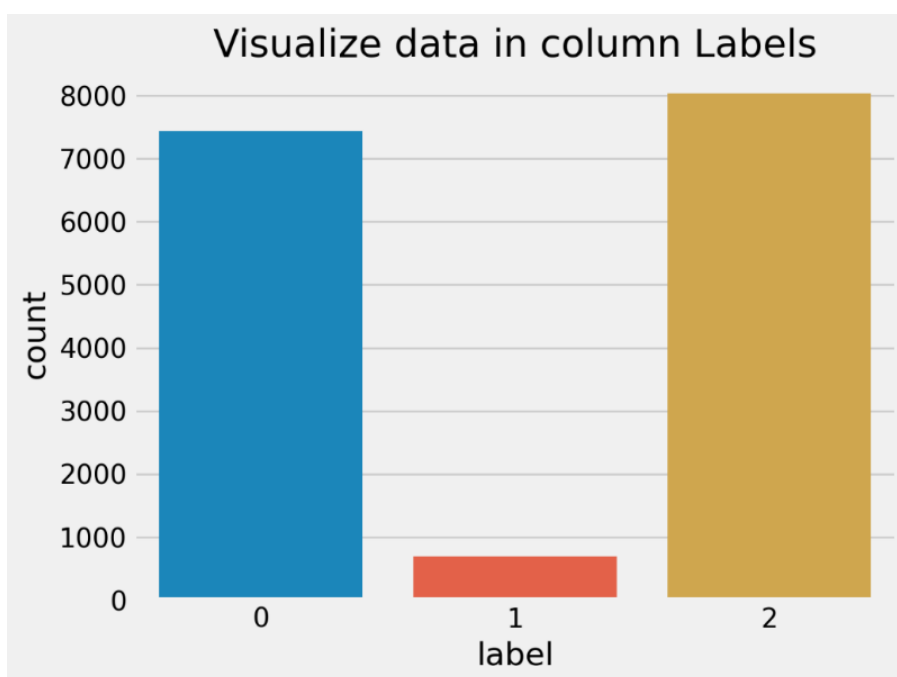
## CHƯƠNG III: ỨNG DỤNG CÁC PHƯƠNG PHÁP TRÍCH SUẤT ĐẶC TRƯNG VÀO BÀI TOÁN PHÂN LOẠI VĂN BẢN

### 3.1. Chuẩn bị dữ liệu

Bộ dữ liệu Vietnam Students' Feedback Corpus (UIT-VSFC) là một tập hợp các phản hồi của sinh viên về các khóa học và giảng viên tại trường Đại học Công nghệ Thông tin - Đại học Quốc gia Thành phố Hồ Chí Minh (UIT). Là một nguồn quan trọng cho nghiên cứu liên ngành liên quan đến việc kết hợp hai lĩnh vực nghiên cứu giữa phân tích tình cảm và giáo dục.

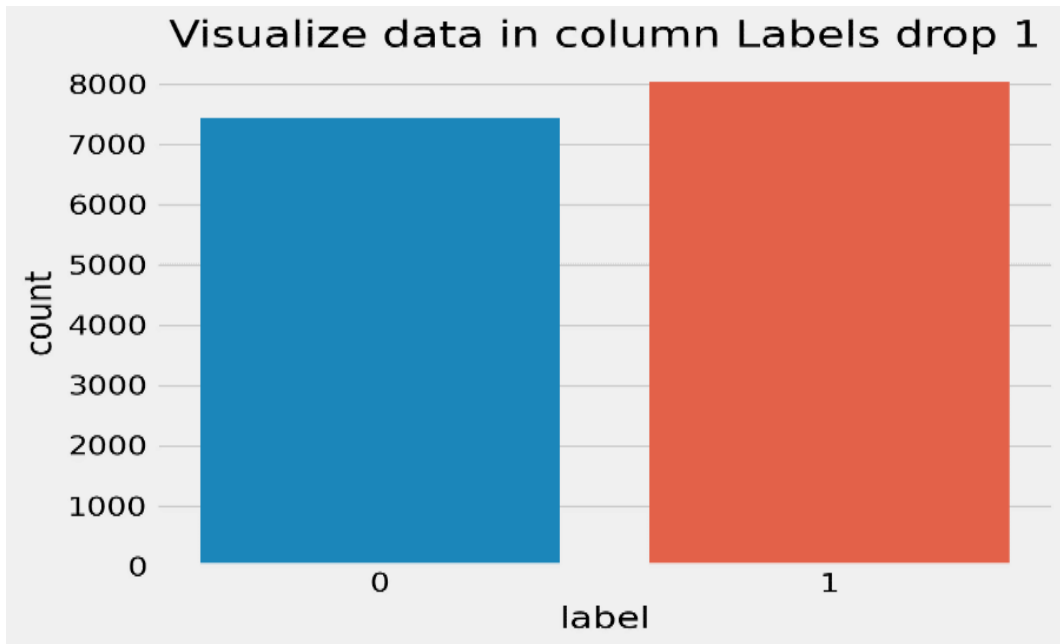
Bộ dữ liệu này bao gồm hơn 16,000 mẫu phản hồi được thu thập từ các sinh viên của UIT được phân vào 3 nhóm 0, 1, 2 tương ứng với Tiêu cực, Trung tính và Tích cực, Bộ dữ liệu được viết bằng tiếng Việt và chứa các đánh giá về chất lượng giảng dạy, nội dung khóa học, cách thức giảng dạy và các khía cạnh khác liên quan đến trải nghiệm học tập của sinh viên.

UIT-VSFC là bộ dữ liệu đầu tiên và lớn nhất của Việt Nam về phản hồi của sinh viên về giáo dục và đã được phân tích để đưa ra những thông tin hữu ích cho việc nâng cao chất lượng giáo dục và cải thiện trải nghiệm học tập của sinh viên.



Hình 3. 1: Trực quan hoá về tập dữ liệu UIT-VSFC

Do số lượng dữ liệu nhóm 1 là nhóm Trung tính quá ít có thể dẫn đến ảnh hưởng về độ chính xác của mô hình. Nên sẽ thực hiện xóa bỏ những dữ liệu trên tập dữ liệu thuộc nhóm 1. Sau khi xóa sẽ chuyển nhóm 2 là Tích cực về nhóm 1. Cuối cùng dữ liệu sẽ còn 2 nhóm 0,1 ứng với Tiêu cực và Tích cực.



*Hình 3. 2: Tập dữ liệu sau khi đã xoá nhãn Trung Tính*

Để thực nghiệm các phương pháp phân loại văn bản, sẽ thực hiện tiền xử lý dữ liệu văn lên tập dữ liệu UIT-VSFC. Quá trình xử lý dữ liệu bao gồm các thao tác:

- Chuyển văn bản sang dạng chữ in thường.
- Tách từ sử dụng hàm `word_tokenize` trong thư viện `underthesea` của python.
- Loại bỏ các stopwords tức là những từ không mang nhiều ý nghĩa cho câu (a lô, a ha, ai , ai ai,...)
- Xóa bỏ các kí tự đặc biệt, chỉ giữ chữ cái, số, dấu chấm, dấu phẩy, dấu hỏi chấm, dấu chấm than.

Số từ	Số văn bản huấn luyện	Số văn bản đánh giá	Số văn bản kiểm tra
149863	10969	1511	3000

*Bảng 3. 1: Thống kê số lượng dữ liệu chi tiết*

### 3.2. Trích xuất đặc trưng.

Trong thực nghiệm này sử dụng 3 phương pháp trích xuất đặc trưng văn bản: phương pháp BoW, Phương pháp TF-IDF, phương pháp Word2Vec để lấy ra các vector.

#### Phương pháp BoW:

Khi sử dụng phương pháp này, từ điển được xác định trước và mỗi từ trong từ điển được đánh một chỉ số duy nhất. Sau đó, mỗi văn bản được biểu diễn dưới dạng một vector với chiều dài bằng số từ trong từ điển và các giá trị trong vector tương ứng với tần suất xuất hiện của từ trong văn bản. Khi sử dụng phương pháp này, sẽ thu được một ma trận mà trong đó, mỗi hàng sẽ đại diện cho một văn bản, mỗi cột đại diện cho một



từ có trong từ điển, và mỗi ô (cell) sẽ chứa tần suất xuất hiện của từ trong văn bản tương ứng.

```
Vector hoá tập Train BoW [1 1 1 ... 1 1 1]
-----
Kích thước vector (10969, 2406)
```

**Hình 3. 3: Vector hoá và kích thước của tập train BoW**

Số từ trong phương pháp BoW là 2406 từ không trùng nhau gọi là từ điển. Sau khi mô hình hóa, mỗi văn bản là một vector trọng số các từ, trong đó các trọng số là chỉ số BoW như đã trình bày.

Như vậy tập ngữ liệu được mô hình hóa như là một ma trận chứa BoW của các từ và có kích thước 10969 x 2406 phần tử. Kích thước của 1 vector khá lớn, đối với những bộ dữ liệu có nhiều từ điển hơn thì kích thước của 1 vector có thể lên đến hàng triệu thuộc tính.

**Phương pháp TF-IDF:** Tương tự như phương pháp BoW, phương pháp này sẽ thu được một ma trận mà trong đó, mỗi hàng sẽ đại diện cho một văn bản, mỗi cột đại diện cho một từ có trong từ điển, và mỗi ô (cell) sẽ chứa tần suất xuất hiện của từ trong văn bản tương ứng. Số từ trong phương pháp TF-IDF là 2406. Nên ma trận trên tập train sẽ có kích thước 10969 x 2406.

```
Vector hoá tập Train TF-IDF [0.45295835 0.48467215 0.41156385 ... 0.26838392 0.25948468 0.20639552]
-----
Kích thước vector (10969, 2406)
```

**Hình 3. 4: Vector hoá và kích thước của tập train TF-IDF**

Để khắc phục được nhược điểm là kích thước của 1 vector có thể lên đến hàng triệu thuộc tính trong phương pháp BoW và phương pháp TF-IDF. Sẽ sử dụng phương pháp SVD để lọc ra những thuộc tính nổi bật của các vector. Loại bỏ các thuộc tính không nổi bật hoặc tần suất xuất hiện thấp. giúp giảm kích thước của ma trận mà không làm mất đi quan hệ tuyến tính giữa các phần tử của nó. Phương pháp này giúp chuyển đổi ma trận ban đầu thành một ma trận mới có kích thước nhỏ hơn, đồng thời giữ lại những thông tin quan trọng nhất của ma trận gốc.

Với phương pháp SVD, ta có thể áp dụng để giảm số lượng thuộc tính của các vector được tạo ra từ phương pháp BoW và TF-IDF. Những thuộc tính không quan trọng hoặc tần suất xuất hiện thấp sẽ được loại bỏ, giúp giảm kích thước của vector và tăng tốc độ xử lý dữ liệu. Việc loại bỏ những thuộc tính không quan trọng này cũng giúp cho mô hình học tập tốt hơn và chính xác hơn trong việc phân loại và xử lý dữ liệu. Trong

thực nghiệm này sẽ thực hiện giữ lại 1000 thuộc tính nổi bật. Hiện tại ma trận sẽ có kích thước 10969 x 1000.

```
Kích thước vector TF- IDF ban đầu: (10969, 2406)
-----
Kích thước vector TF- IDF + SVD: (10969, 1000)

Kích thước vector TF- IDF ban đầu: (10969, 2406)
-----
Kích thước vector TF- IDF + SVD: (10969, 1000)
```

**Hình 3. 5: Kích thước vector sau khi sử dụng SVD**

### **Phương pháp Word2Vec:**

Trong phương pháp này, sẽ chuyển mỗi từ trong từ điển về một vector 100 chiều. Để thực hiện phương pháp này tôi dùng thư viện gensim với mô hình Skipgram và CBOW để học biểu diễn vector cho các từ. Trong phương pháp Word2vec, sẽ sử dụng một mô hình neural network đơn giản để học cách biểu diễn các từ dưới dạng vector. Có hai kiểu mô hình Word2vec phổ biến: Skipgram và CBOW. Skipgram sẽ cố gắng dự đoán các từ xung quanh từ đầu vào, trong khi CBOW sẽ cố gắng dự đoán từ đầu ra từ một số từ đầu vào. Sử dụng các tham số sao để huấn luyện Word2vec: vector\_size=100, negative=5, window=2, min\_count=2, workers=cores, alpha=0.065, min\_alpha=0.065.

Sau khi huấn luyện mô hình Word2vec, chúng ta có thể sử dụng các vector biểu diễn cho các từ để huấn luyện các mô hình Deep Learning như mô hình LSTM, CNN hoặc các mô hình neural network khác.

```
array([-2.36130968e-01, 1.51865458e+00, 3.90912071e-02, -6.47129238e-01,
       3.95269305e-01, 2.36238301e-01, 1.00375068e+00, -4.78306234e-01,
       -5.26681483e-01, -6.26474023e-01, -5.00228584e-01, 2.61381954e-01,
       6.15740299e-01, -1.78991067e+00, 1.42292589e-01, -1.65004715e-01,
       1.28027213e+00, -3.15176547e-01, -9.62247103e-02, -2.07302541e-01,
       3.06555510e-01, 3.64028305e-01, -4.71231282e-01, 8.83896947e-01,
       1.03606904e+00, -6.69953600e-02, -2.81340808e-01, 4.70504284e-01,
       -4.33924720e-02, 2.36919865e-01, -6.06413543e-01, -2.77972728e-01,
       7.52993464e-01, -1.24011397e+00, 3.74837577e-01, 2.39646626e+00,
       -8.28102827e-01, 1.69910848e-01, 2.68751591e-01, -3.26928020e-01,
       1.92323968e-01, 4.55610096e-01, -3.48844111e-01, 1.52058378e-01,
       -7.82458037e-02, -1.94296494e-01, 1.12737797e-01, -4.13304955e-01,
       -7.59598315e-01, 3.01975049e-02, 1.36429036e+00, 7.11591780e-01,
       4.30603087e-01, 4.66274232e-01, -2.40892872e-01, 4.14078295e-01,
       1.56921804e-01, 3.87404919e-01, 5.24007499e-01, -8.21064413e-01,
       3.51086259e-01, 2.36065701e-01, 1.42162189e-01, -3.42677772e-01,
       7.39648104e-01, -9.26412344e-01, 7.89233327e-01, -5.23562789e-01,
       -1.04665637e+00, -2.21141011e-01, 5.47989190e-01, -4.42659527e-01,
       -5.60853720e-01, -1.35036543e-01, 1.50999576e-02, -2.18030542e-01,
       6.98899031e-01, 2.65954673e-01, 8.20910871e-01, -6.20242894e-01,
       -3.86930704e-01, 8.09942856e-02, 1.01670516e+00, -3.86442900e-01,
       -1.12725282e-03, 3.12398523e-01, 8.89553785e-01, -1.38606966e-01,
       -1.22030005e-01, -1.63038671e-02, 4.33075368e-01, 6.46911979e-01,
       -7.59048104e-01, 2.72605360e-01, -1.11590691e-01, 4.00531322e-01,
       -1.17718220e+00, 7.42888808e-01, 9.73147452e-01, -5.04885018e-01,
       6.16169460e-02, 2.68340707e-01, 4.05866235e-01, 3.79808128e-01,
       -1.32646278e-01, 7.74637237e-02, 1.43229023e-01, 2.03027755e-01,
       -6.37472123e-02, 6.99065998e-02, 3.50750536e-01, -1.53752789e-01,
       3.46801281e-01, -8.99076313e-02, 2.26748019e-01, 3.25216129e-02,
       2.43232682e-01, 3.93894240e-02, -2.18116045e-01, -1.42101631e-01,
       -3.96238089e-01, 3.94358069e-01, 3.10194716e-02, -3.41631144e-01,
       2.28431359e-01, 2.31401935e-01, 1.58176556e-01, 3.12889889e-02,
       -1.49778321e-01, 1.97697759e-01, -1.96413264e-01, -2.48990238e-01,
       3.65522414e-01, -3.36974353e-01, -2.02654749e-02, 6.82471991e-01,
```

Kích thước vector Word2Vec (200,)

**Hình 3. 6: Vector hoá 1 từ trong Word2vec**

Khi mô hình Word2Vec được huấn luyện, các từ được biểu diễn bằng các vector trong không gian đa chiều. Các từ có ý nghĩa tương tự thường có xu hướng có các vector gần nhau trong không gian này

```
model_ug_cbow.wv.most_similar("cô")
```

```
[('thầy', 0.8796814680099487),
 ('viên', 0.40515556931495667),
 ('giảng', 0.3942835032939911),
 ('tui', 0.34046387672424316),
 ('giảng', 0.3124541938304901),
 ('colonhihi', 0.31141629815101624),
 ('dạy', 0.29825031757354736),
 ('hay', 0.28326719999313354),
 ('ơ', 0.2800177335739136),
 ('hỏi', 0.27910083532333374)]
```

```
model_ug_sg.wv.most_similar("cô")
```

```
[('thầy', 0.8586903810501099),
 ('boss', 0.5332469940185547),
 ('tánh', 0.5327700972557068),
 ('i', 0.5308171510696411),
 ('colonhihi', 0.530480146408081),
 ('tui', 0.5266602039337158),
 ('giảng', 0.49574220180511475),
 ('nà', 0.4906882047653198),
 ('cắm', 0.4819173514842987),
 ('ơ', 0.47725653648376465)]
```

**Hình 3. 7: Similar word2vec với Skipgram và CBOW**

### 3.3. Xây dựng mô hình

Đối với pháp pháp BoW và phương pháp TF-IDF sẽ sử dụng Máy vector hỗ trợ SVM để thực hiện huấn luyện.

Để chọn ra các tham số tốt nhất phù hợp với từng phương pháp trích xuất đặc trưng khác nhau. Sử dụng GridSearchCV để tìm ra các tham số tối ưu. Sau khi tiến hành tìm tham số tối ưu. Quá trình chạy mô hình với tham số cho kết quả tốt nhất tham số của SVM đối với từng phương pháp trích xuất đặc trưng như sau:

Phương pháp trích xuất đặc trưng	Thuật toán phân lớp SVM		
	C	Gamma	Kernel
BoW	10	0.01	rbf
TF-IDF	1	1	rbf

*Bảng 3. 2: Các tham số SVM đối với từng phương pháp*

Đối với phương pháp Word2Vec sẽ sử dụng kết hợp giữa 2 phương pháp Deep Learning đó là Long short-term memory (LSTM) và Convolutional Neural Network (CNN). Tiến hành huấn luyện với 30 epoch và backsize là 40. Kiến trúc của mô hình được mô tả như hình dưới.

```
model_lstm_cnn = Sequential([
    Embedding(vocab_size, embedding_dim, weights=[embedding_matrix],
              input_length=max_sequence_length, trainable=True),
    LSTM(300, return_sequences=True, dropout=0.25, recurrent_dropout=0.1),
    Conv1D(filters=512, kernel_size=7, padding='same', activation='relu', strides=1),
    MaxPooling1D(),
    Conv1D(filters=256, kernel_size=5, activation='relu', padding='same', strides=1),
    MaxPooling1D(),
    Conv1D(filters=128, kernel_size=3, activation='relu', padding='same', strides=1),
    MaxPooling1D(),
    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.2),
    Dense(2, activation='sigmoid')
])
```

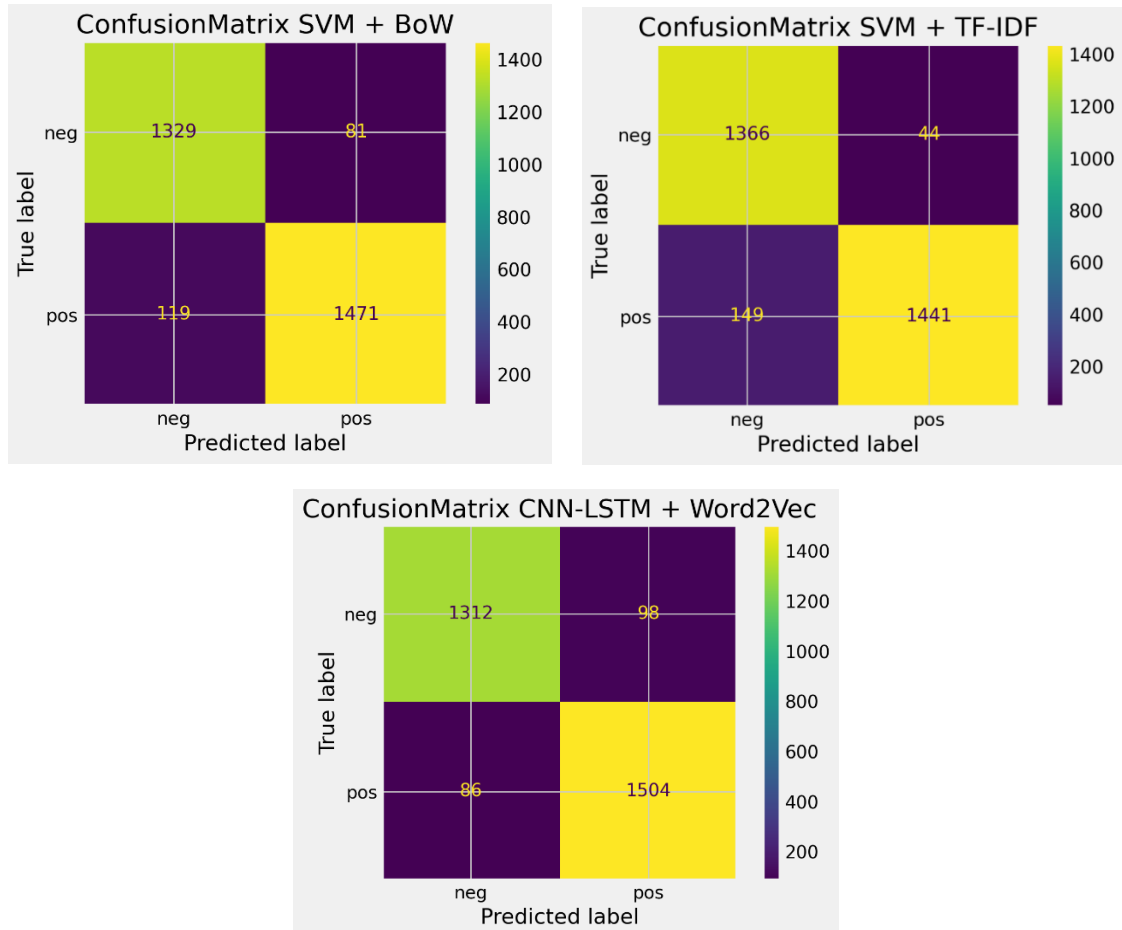
*Hình 3. 8: Kiến trúc của mô hình Deep Learning*

### 3.4. Đánh giá kết quả

Tiến hành huấn luyện tất cả phương pháp nêu trên. Sau khi huấn luyện hoàn thành. Đánh giá các mô hình bằng các tiêu chí đánh giá như Accuracy. Precision, Recall, F1-score đánh giá dựa vào Confusion Matrix. Cụ thể là để đánh giá một bộ phân loại hai lớp tạm gọi là dương và âm:

- Số đúng dương (TP- True positive): số phần tử dương được phân loại dương
- Số sai âm (FN - False negative): số phần tử dương được phân loại âm
- Số đúng âm (TN- True negative): số phần tử âm được phân loại âm
- Số sai dương (FP - False positive): số phần tử âm được phân loại dương

- Độ chính xác (Precision) =  $TP / (TP + FP)$
- Độ bao phủ (Recall) =  $TP / (TP + FN)$
- Độ đo F1 =  $2 * Precision * Recall / (Precision + Recall)$



*Hình 3.9: Confusion Matrix đối với từng phương pháp*

Phương pháp huấn luyện	Tiêu chí đánh giá			
	Accuracy	Precision	Recall	F1-score
<b>SVM + BoW</b>	93 %	93 %	93 %	93 %
<b>SVM + TF-IDF</b>	93.5 %	94 %	94 %	94 %
<b>Word2Vec + LSTM-CNN</b>	94 %	94 %	95 %	94 %

*Bảng 3. 3: Kết quả của các tiêu chí đánh giá*

Kết quả thể hiện các tiêu chí đánh giá của các mô hình trên tập kiểm thử được thống kê trong Bảng 3.3. Các phương pháp đều cho kết quả tốt và đồng đều trong các tiêu chí đánh giá.

### 3.5. Triển khai ứng dụng

Khi các mô hình đã được huấn luyện hoàn hảo, công việc cuối cùng là triển khai nó thành ứng dụng. Sử dụng thư viện mã nguồn mở của Python là Streamlit để triển khai trên nền web.

Do khi huấn luyện các mô hình có kích thước khá lớn. Nên khi đóng gói ứng dụng không thể tải các tài nguyên lên Cloud. Nên chỉ hiển thị được lên trang web cục bộ.

Giao diện của ứng dụng phân loại văn bản sẽ có 2 tùy chọn đầu vào đó là Text và Audio. Sau khi nhận được đầu vào đó. Sẽ thực hiện các bước tiền xử lý văn bản và trích xuất đặc trưng để huấn luyện mô hình.

Cuối cùng sẽ tạo một sự kiện khi người dùng nhấn vào button ***“Dự đoán cảm xúc”*** thì chương trình sẽ thực hiện dự đoán với đầu vào người dùng đã nhập. Kết quả sẽ hiển thị với dự đoán là lần lượt các phương pháp SVM + BoW, SVM + TF-IDF, Word2Vec + LSTM-CNN.

Giao diện chính của chương trình sẽ như **hình 3.9**

## Sentiment Classification Student App



Lựa chọn thành phần Input

☒ Text


☐ Audio

Nhập văn bản cần dự đoán tại đây:

Dự đoán cảm xúc

**Hình 3. 10: Giao diện của ứng dụng phân loại văn bản**

# Sentiment Classification Student App



Lựa chọn thành phần Input

☒ Text

☐ Audio

Nhập văn bản cần dự đoán tại đây:

thầy dạy chi tiết giảng giải kỹ

Dự đoán cảm xúc

Văn bản vừa nhập:

thầy dạy chi tiết giảng giải kỹ

Prediction

SVM + BoW dự đoán: Tích cực 😊

SVM + TF-IDF dự đoán: Tích cực 😊

LSTM + Word2Vec dự đoán: Tích cực 😊

**Hình 3. 11:** Giao diện của chương trình khi dự đoán nhận là tích cực

# Sentiment Classification Student App



Lựa chọn thành phần Input

☒ Text  
☐ Audio

Nhập văn bản cần dự đoán tại đây:

sử dụng thời gian không hiệu quả đi quá muộn và về quá trễ 8h30 vào lớp 12h về

Dự đoán cảm xúc

Văn bản vừa nhập:

sử dụng thời gian không hiệu quả đi quá muộn và về quá trễ 8h30 vào lớp 12h về

Prediction

SVM + BoW dự đoán: Tiêu cực 😞

TF-IDF dự đoán: Tiêu cực 😞

LSTM + Word2Vec dự đoán: Tiêu cực 😞

**Hình 3. 12:** Giao diện của chương trình khi dự đoán nhận là tiêu cực



## CHƯƠNG IV: KẾT LUẬN

### 4.1. Kết quả đạt được

Qua quá trình tìm hiểu, nghiên cứu về một số phương pháp trích suất đặc trưng trong nhận dạng văn bản, đã đạt được kết quả đáng khả quan. Cụ thể, đã tìm hiểu và áp dụng các phương pháp trích suất đặc trưng như TF-IDF, BoW, Word2Vec.

### 4.2. Hạn chế

Tuy nhiên, hạn chế của phương pháp trích suất đặc trưng là có thể bỏ qua một số thông tin quan trọng trong văn bản nếu không được thiết lập đúng tham số. Ngoài ra, các phương pháp này cũng có thể bị ảnh hưởng bởi các ngôn ngữ địa phương.

### 4.3. Hướng phát triển

Để phát triển hơn nữa trong lĩnh vực này, có thể nghiên cứu sâu hơn về các phương pháp trích suất đặc trưng hiện có, thử nghiệm và so sánh hiệu quả của chúng trên các bộ dữ liệu khác nhau. Ngoài ra, có thể khám phá và áp dụng các phương pháp mới nhất trong lĩnh vực học máy để nâng cao độ chính xác và tốc độ xử lý của quá trình nhận dạng văn bản. Bên cạnh đó, cần phát triển các ứng dụng thực tế để sử dụng trong các lĩnh vực khác nhau như tin tức, quảng cáo, phân tích ý kiến của khách hàng, và nhiều lĩnh vực khác.

## TÀI LIỆU THAM KHẢO

- [1] Trần Cao Đệ và Phạm Nguyên Khang. (2012). Phân loại văn bản với máy học vector hỗ trợ và cây quyết định. Tạp chí Khoa học, 21a, 52-63. Trường Đại học Cần Thơ.
- [2] Đỗ Thanh Nghị và Phạm Nguyên Khang. (2013). Phân loại văn bản: Mô hình túi từ và tập hợp mô hình máy học tự động. Trường Đại học Cần Thơ.
- [3] Đỗ Thanh Nghị và Trần Cao Đệ. (2014). Kết hợp ngữ nghĩa với mô hình túi từ để cải tiến giải thuật k láng giềng trong phân lớp văn bản ngắn. Tạp chí Khoa học. Trường Đại học Cần Thơ.
- [4] Đặng Văn Nam. (2020). Nghiên cứu mô hình học máy Naïve Bayes trong phân lớp văn bản; Ứng dụng phân lớp cho tập dữ liệu các nhận xét trên Twitter. Hội nghị Toàn quốc Khoa học Trái đất và Tài nguyên với phát triển bền vững (ERSD 2020), Trường Đại học Mở - Địa chất.
- [5] Nguyễn Đình Mạnh. (2020). Nghiên cứu các phương pháp tính toán độ tương tự của văn bản luật tiếng Việt. Luận văn thạc sĩ Khoa học Máy tính, Hà Nội.

## PHỤ LỤC CODE

```
1. import streamlit as st
2. import joblib
3. import string
4. import pandas as pd
5. import numpy as np
6. from underthesea import word_tokenize
7. import speech_recognition as sr
8. from sklearn.decomposition import TruncatedSVD
9. from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
10. from PIL import Image
11. from gensim.models import KeyedVectors
12. from keras.preprocessing.text import Tokenizer
13. from keras.utils import pad_sequences
14. from keras.models import load_model
15.
16.
17.
18. model_ug_cbow = KeyedVectors.load('w2v_model_ug_cbow.word2vec')
19. model_ug_sg = KeyedVectors.load('w2v_model_ug_sg.word2vec')
20.
21. # Load Mô Hình
22. Svm_TF_IDF = joblib.load('svm_tfidf.pkl')
23. Svm_BoW = joblib.load('svm_BoW.pkl')
24. print(1)
25. # Load Vector
26. vec_TF_IDF = joblib.load('tfidf_vec.pkl')
27. vec_BoW = joblib.load('BoW_vec.pkl')
28.
29. # Load Vector SVD
30. svd_TF_IDF = joblib.load('svd_tfidf.pkl')
31. svd_BoW = joblib.load('svd_BoW.pkl')
32.
33. # tokenizer
34. tokenizer = joblib.load('tokenizer.joblib')
35.
36. # CNN- LSTM
37. LSTM_CNN = load_model('model_lstm_cnn_33.h5')
38.
39.
40. print("Load Thành Công")
41. def preprocess_text(text):
42.     text = text.translate(str.maketrans('', '', string.punctuation))
43.     tokenize = word_tokenize(text)
44.     filename = 'stopwords.csv'
45.     data = pd.read_csv(filename, names=['word'])
46.     list_stopwords = data['word']
47.     myarray = np.asarray(list_stopwords)
48.     words = [word for word in tokenize if word.lower() not in myarray]
49.     sentence = ' '.join(words)
50.     return [sentence]
51.
52.
53. def transform_doc (doc):
54.     doc = preprocess_text(doc)
55.     # vec tf
56.     vec_doc_tf = vec_TF_IDF.transform(doc)
57.     vec_doc_tf= svd_TF_IDF.transform(vec_doc_tf)
58.     # vector bow
59.     vec_doc_bow = vec_BoW.transform(doc)
60.     vec_doc_bow= svd_BoW.transform(vec_doc_bow)
61.     return vec_doc_tf, vec_doc_bow
62.
63. def predict_tf_bow( vec_doc_tf, vec_doc_bow):
64.     y_pred_tf = Svm_TF_IDF.predict(vec_doc_tf)
65.     y_pred_bow = Svm_BoW.predict(vec_doc_bow)
```

```

66.     return y_pred_tf, y_pred_bow
67.
68. # embeddings
69. embeddings_index = {}
70. for w in list(model_ug_cbow.wv.index_to_key):
71.     embeddings_index[w] = np.append(model_ug_cbow.wv[w], model_ug_sg.wv[w])
72. num_words = 10000
73. embedding_matrix = np.zeros((num_words, 200))
74. for word, i in tokenizer.word_index.items():
75.     if i >= num_words:
76.         continue
77.     embedding_vector = embeddings_index.get(word)
78.     if embedding_vector is not None:
79.         embedding_matrix[i] = embedding_vector
80.
81.
82. def w2v (doc):
83.     doc = preprocess_text(text)
84.     doc_sequence = tokenizer.texts_to_sequences([doc])
85.     doc_padded = pad_sequences(doc_sequence, maxlen=150)
86.     predictions = (LSTM_CNN.predict(doc_padded) > 0.5).astype("float32")
87.     a = np.argmax(predictions)
88.     return a
89.
90.
91. st.title("Sentiment Classification Student App")
92. st.write('\n')
93.
94. image = Image.open('what-is-sentiment-analysis-cover.jpg')
95. show = st.image(image, use_column_width=True)
96.
97. option = st.radio(
98.     "Lựa chọn thành phần Input",
99.     ("Text", "Audio")
100. )
101.     if option == "Text":
102.         with st.form(key='emotion_clf_form'):
103.             text = st.text_area("Nhập văn bản cần dự đoán tại đây: ")
104.             submit = st.form_submit_button(label='Dự đoán cảm xúc')
105.
106.             if submit:
107.                 st.success('Văn bản vừa nhập: ')
108.                 st.write(text)
109.                 st.success("Prediction")
110.
111.                 vec_tf, vec_bow = transform_doc(text)
112.                 vec_tf, vec_bow = predict_tf_bow(vec_tf, vec_bow)
113.
114.                 vec_w2v = w2v (text)
115.
116.                 st.spinner('Classifying ...')
117.                 if vec_tf[0] and vec_bow[0] and vec_w2v == 1:
118.                     st.write("SVM + BoW dự đoán: Tích cực 🤩")
119.                     st.write("SVM + TF-IDF dự đoán: Tích cực 🤩")
120.                     st.write("LSTM + Word2Vec dự đoán: Tích cực 🤩")
121.
122.                     st.balloons()
123.
124.                 else:
125.                     st.write("SVM + BoW dự đoán: Tiêu cực 😞")
126.                     st.write("TF-IDF dự đoán: Tiêu cực 😞")
127.                     st.write("LSTM + Word2Vec dự đoán: Tiêu cực 😞")
128.
129.             else:
130.                 with st.form(key='emotion_clf_form'):
131.                     r = sr.Recognizer()
132.                     with sr.Microphone() as source:

```

```

133.         st.write("Hãy nói gì đó...")
134.         audio = r.listen(source)
135.         try:
136.             text = r.recognize_google(audio, language="vi-VN")
137.             st.write("Bạn vừa nói: ", text)
138.         except sr.UnknownValueError:
139.             st.write("Không thể nhận dạng giọng nói!")
140.         except sr.RequestError as e:
141.             st.write("Lỗi kết nối tới Google Speech Recognition service;
{0}".format(e))
142.
143.         submit = st.form_submit_button(label='Dự đoán cảm xúc')
144.
145.         if submit:
146.             st.success("Prediction")
147.
148.             vec_tf, vec_bow = transform_doc(text)
149.             vec_tf, vec_bow = predict_tf_bow(vec_tf, vec_bow)
150.
151.             vec_w2v = w2v (text)
152.
153.             st.spinner('Classifying ...')
154.             if vec_tf[0] and vec_bow[0] and vec_w2v == 1:
155.                 st.write("SVM + BoW dự đoán: Tích cực 🤩")
156.                 st.write("SVM + TF-IDF dự đoán: Tích cực 🤩")
157.                 st.write("LSTM + Word2Vec dự đoán: Tích cực 🤩")
158.                 st.balloons()
159.
160.             else:
161.                 st.write("SVM + BoW dự đoán: Tiêu cực 😞")
162.                 st.write("TF-IDF dự đoán: Tiêu cực 😞")
163.                 st.write("LSTM + Word2Vec dự đoán: Tiêu cực 😞")

```