**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

****

**ISO 9001:2015**

**DƯƠNG VĂN HIỆP**

**NHẬN DIỆN CẢM XÚC TỪ VĂN BẢN TIẾNG VIỆT BẰNG MÔ HÌNH NLP**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**VĨNH LONG, NĂM 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

****

**ISO 9001:2015**

**NHẬN DIỆN CẢM XÚC TỪ VĂN BẢN TIẾNG VIỆT BẰNG MÔ HÌNH NLP**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Sinh viên: **Dương Văn Hiệp**

Lớp: **DA21TTB**

MSSV: **110121209**

GVHD: **TS. Nguyễn Nhứt Lam**

**VĨNH LONG, NĂM 2025**

LỜI MỞ ĐẦU

Trong kỷ nguyên bùng nổ của công nghệ số và thương mại điện tử, khối lượng dữ liệu văn bản phát sinh mỗi ngày là vô cùng lớn, đặc biệt là các nội dung bình luận, đánh giá từ người dùng trên các nền tảng mua sắm trực tuyến. Những ý kiến này không chỉ phản ánh cảm xúc, trải nghiệm của người tiêu dùng mà còn là nguồn thông tin quý giá giúp doanh nghiệp hiểu rõ hơn về sản phẩm, dịch vụ và nhu cầu thị trường.

Tuy nhiên, việc xử lý và phân tích thủ công khối lượng dữ liệu lớn như vậy là điều gần như không khả thi. Chính vì thế, việc áp dụng các kỹ thuật Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) để tự động nhận diện cảm xúc từ văn bản tiếng Việt là một hướng nghiên cứu thiết thực và có ý nghĩa cả về mặt học thuật lẫn ứng dụng thực tế.

Đề tài “Nhận diện cảm xúc từ văn bản tiếng Việt bằng mô hình NLP” được thực hiện nhằm nghiên cứu, xây dựng và đánh giá mô hình phân loại cảm xúc trong văn bản tiếng Việt, với dữ liệu được thu thập từ các bình luận thực tế trên các trang thương mại điện tử. Các cảm xúc được chia thành ba nhóm chính: tích cực, tiêu cực và trung lập. Trong quá trình thực hiện, khóa luận ứng dụng các kỹ thuật tiền xử lý văn bản, biểu diễn đặc trưng và huấn luyện mô hình học máy để giải quyết bài toán một cách hiệu quả.

Nhầm vận dụng các kiến thức đã học vào thực tiễn, góp phần mở rộng hướng tiếp cận trong lĩnh vực xử lý ngôn ngữ tiếng Việt – một ngôn ngữ có đặc thù riêng và còn nhiều thách thức trong nghiên cứu NLP hiện nay.

LỜI CẢM ƠN

Trước hết, em xin chân thành cảm ơn quý thầy cô Khoa Công nghệ Thông tin, Trường Kỹ thuật và Công nghệ, Trường Đại học Trà Vinh, những người đã giảng dạy, truyền đạt kiến thức, kỹ năng và khơi dậy đam mê trong suốt quá trình em theo học ngành Công nghệ thông tin.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc đến thầy TS. Nguyễn Nhứt Lam, đã tận tình hướng dẫn, hỗ trợ và truyền đạt cho em những kiến thức quý báu trong suốt quá trình thực hiện khóa luận tốt nghiệp. Thầy không chỉ giúp em định hướng nghiên cứu rõ ràng, mà còn luôn tạo điều kiện thuận lợi và động viên tinh thần để em hoàn thành tốt công việc của mình.

Em rất mong nhận được sự góp ý của quý thầy cô và các bạn để khóa luận được hoàn thiện hơn.

Em xin chân thành cảm ơn!

|  |  |
| --- | --- |
|  | Sinh viên thực hiện  **Dương Văn Hiệp** |

NHẬN XÉT

(Của giảng viên hướng dẫn)

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

|  |  |
| --- | --- |
|  | **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)*  **Nguyễn Nhứt Lam** |

NHẬN XÉT

(Của giảng viên chấm)

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

|  |  |
| --- | --- |
|  | **Giảng viên chấm**  *(Ký và ghi rõ họ tên)* |

|  |  |
| --- | --- |
| UBND TỈNH VĨNH LONG | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **TRƯỜNG ĐẠI HỌC TRÀ VINH** | **Độc lập - Tự do - Hạnh phúc** |

**BẢN NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP**

*(Của giảng viên hướng dẫn)*

|  |  |
| --- | --- |
| Họ và tên sinh viên: Dương Văn Hiệp | MSSV: 110121209 |
| Ngành: Công nghệ thông tin | Khóa: 2021 |
| Tên đề tài: Nhận diện cảm xúc từ văn bản tiếng Việt bằng mô hình NLP | |
| Họ và tên Giảng viên hướng dẫn: Nguyễn Nhứt Lam | |
| Chức danh: Trưởng Khoa CNTT | Học vị: Tiến sĩ |

**NHẬN XÉT**

1. Nội dung đề tài:

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

2. Ưu điểm:

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

3. Khuyết điểm:

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

4. Điểm mới đề tài:

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

5. Giá trị thực trên đề tài:

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

6. Đề nghị sữa chửa bổ sung:

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

7. Đánh giá

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

|  |  |
| --- | --- |
|  | Vĩnh Long, *ngày tháng năm 2025*  **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)*  **Nguyễn Nhứt Lam** |

|  |  |
| --- | --- |
| UBND TỈNH VĨNH LONG | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **TRƯỜNG ĐẠI HỌC TRÀ VINH** | **Độc lập - Tự do - Hạnh phúc** |

**BẢN NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP**

*(Của giảng viên chấm)*

|  |  |
| --- | --- |
| Họ và tên người nhận xét:……………………………………………………………… | |
| Chức danh:………………………………………………... | Học vị:…………………… |
| Chuyên ngành:………………………………………………………………………….. | |
| Cơ quan công tác:………………………………………………………………………. | |
| Tên sinh viên: Dương Văn Hiệp | |
| Tên đề tài: Nhận diện cảm xúc từ văn bản tiếng Việt bằng mô hình NLP | |

**I. Ý KIẾN NHẬN XÉT**

1. Nội dung:

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

2. Điểm mới các kết quả của khóa luận:

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

3. Ứng dụng thực tế:

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**II. CÁC VẤN ĐỀ CẦN LÀM RÕ**

(Các câu hỏi của giáo viên phản biện)

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**III. KẾT LUẬN**

(Ghi rõ đồng ý hay không đồng ý cho bảo vệ đồ án khóa luận tốt nghiệp)

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

|  |  |
| --- | --- |
|  | Vĩnh Long, *ngày tháng năm 2025*  **Giảng viên chấm**  *(Ký và ghi rõ họ tên)* |

MỤC LỤC

CHƯƠNG 1. TỔNG QUAN 1

1.1. Tổng quan đề tài 1

1.2. Mục tiêu đề tài 2

1.3. Phạm vi nghiên cứu 3

1.4. Phương pháp nghiên cứu 3

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 6

2.1. Tổng quan về xử lý ngôn ngữ tự nhiên 6

2.1.1 Khái niệm 6

2.1.2 Quy trình xử lý ngôn ngữ tự nhiên 6

2.2. Phân tích cảm xúc 7

2.2.1 Khái niệm 7

2.2.2 Các phương pháp phân tích cảm xúc 8

2.3. Kỹ thuật biểu diễn văn bản 9

2.3.1 Kỹ thuật Bag of Word (BoW) 9

2.3.2 Kỹ thuật TF-IDF 10

2.4. Tổng quan về học máy 11

2.4.1 Khái niệm 11

2.4.2 Phân loại học máy 11

2.4.3 Một số thuật toán học máy phổ biến trong NLP 12

2.4.4 Ưu nhược điểm của học máy 13

2.5. Tổng quan về học sâu 14

2.5.1 Khái niệm 14

2.5.2 Một số kiến trúc học sâu tiêu biểu trong NLP 15

2.5.3 Ưu nhược điểm của học sâu 15

2.6. Thuật toán Naïve Bayes 17

2.6.1 Khái niệm 17

2.6.2 Định lý Bayes 17

2.6.3 Giả định “Naïve” – Độc lập có điều kiện 17

2.6.4 Ưu điểm và nhược điểm của Naïve Bayes 18

2.7. Mô hình PhoBERT 19

2.7.1 Khái niệm 19

2.7.2 Kiến trúc của PhoBERT 19

2.7.3 Đặc điểm của PhoBERT 21

2.7.4 Ứng dụng PhoBERT trong phân loại cảm xúc tiếng Việt 21

2.7.5 Ưu điểm và nhược điểm của PhoBERT 22

2.8. Tổng quan về FastAPI 23

2.8.1 Khái niệm 23

2.8.2 Đặc điểm nổi bật 24

2.8.3 Ưu nhược điểm của FastAPI 25

2.9. Tổng quan về NextJS 26

2.9.1 Tổng quan về NextJS 26

2.9.2 Ưu nhược điểm của NextJS 26

2.9.3 Cấu trúc của NextJS 27

2.9.4 Page Router và App Router trong NextJS 28

CHƯƠNG 3. HUẤN LUYỆN MÔ HÌNH 30

3.1. Thu thập và tiền xử lý dữ liệu 30

3.1.1 Thu thập dữ liệu 30

3.1.2 Làm sạch dữ liệu 36

3.1.3 Gán nhãn dữ liệu 39

3.2. Huấn luyện mô hình với Naïve Bayes 44

3.2.1 Xử lý dữ liệu và chia tập dữ liệu 44

3.2.2 Huấn luyện Naïve Bayes với BoW 46

3.2.3 Huấn luyện Naïve Bayes với TF-IDF 47

3.3. Huấn luyện mô hình với PhoBERT 49

3.4. So sánh kết quả giữa các phương pháp 53

CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG THỬ NGHIỆM 55

4.1. Mục tiêu xây dựng ứng dụng thử nghiệm 55

4.2. Kết quả sau khi xây dựng ứng dụng thử nghiệm 55

CHƯƠNG 5. KẾT LUẬN 60

5.1. Về mặt nghiên cứu 60

5.2. Về mặt ứng dụng 60

5.3. Về mặt học tập 61

DANH MỤC CÁC BẢNG, BIỂU ĐỒ, HÌNH

[Bảng 2.1 So sánh Page Router và App Router 28](#_Toc206216864)

[Bảng 3.1 Đối chiếu nhãn dữ liệu của các thành viên đã gán 41](#_Toc206216865)

[Bảng 3.2 Thống nhất nhãn dữ liệu 42](#_Toc206216866)

[Bảng 3.3 Đánh giá mô hình huấn luyện Naïve Bayes với trích xuất đặt trưng BoW 46](#_Toc206216867)

[Bảng 3.4 Đánh giá mô hình huấn luyện Naïve Bayes với trích xuất đặt trưng TF-IDF 48](#_Toc206216868)

[Bảng 3.5 Đánh giá mô hình huấn luyện với PhoBERT 52](#_Toc206216869)

[Bảng 3.6 So sánh kết quả dự đoán giữa các mô hình 53](#_Toc206216870)

[Biểu đồ 3.1 Phân bố số lượng của từng nhãn dữ liệu 43](#_Toc206216871)

[Biểu đồ 3.2 Tỷ lệ phân bố của từng nhãn dữ liệu 44](#_Toc206216872)

[Biểu đồ 3.3 Ma trận nhầm lẫn khi huấn luyện mô hình Naïve Bayes với trích xuất đặt trưng BoW 47](#_Toc206216873)

[Biểu đồ 3.4 Ma trận nhầm lẫn khi huấn luyện mô hình Naïve Bayes với trích xuất đặt trưng TF-IDF 48](#_Toc206216874)

[Biểu đồ 3.5 Training Loss over Epochs khi huấn luyện với mô hình PhoBERT 51](#_Toc206216875)

[Biểu đồ 3.6 Ma trận nhầm lẫn khi huấn luyện với mô hình PhoBERT 52](#_Toc206216876)

[Hình 2.1 Kiến trúc Transformer Encoder 20](#_Toc206216877)

[Hình 3.1 Dữ liệu thô sau khi thu thập 36](#_Toc206216878)

[Hình 4.1 Giao diện chính nhập văn bản dự đoán 56](#_Toc206216879)

[Hình 4.2 Giao diện khi dự đoán là tích cực 57](#_Toc206216880)

[Hình 4.3 Giao diện khi dự đoán là tiêu cực 57](#_Toc206216881)

[Hình 4.4 Giao diện khi dự đoán là trung lập 58](#_Toc206216882)

[Hình 4.5 Giao diện khi dự đoán một sản phẩm của tiki 59](#_Toc206216883)

DANH MỤC CÁC CỤM TỪ VIẾT TẮT

|  |  |
| --- | --- |
| BoW: | Bag of Word. |
| DL: | Deep Learning. |
| ML: | Machine Learning. |
| NLP: | Natural Language Processing. |
| TF-IDF: | Term Frequency - Inverse Document Frequency. |

# TỔNG QUAN

## Tổng quan đề tài

Trong thời đại số hóa hiện tại, đặc biệt là với sự phát triển mạnh mẽ của các nền tảng thương mại điện tử, người tiêu dùng đang có xu hướng chuyển từ hình thức mua sắm trực tiếp sang mua sắm trực tuyến về đa số hàng hóa tiêu dùng. Vì vậy, các cửa hàng, doanh nghiệp cũng đang mở rộng kinh doanh trực tuyến để đáp ứng được nhu cầu của khách hàng.

Qua đó, khi các cửa hàng, doanh nghiệp có một lượng lớn khách hàng và doanh số sản phẩm bán ra được, việc quản lý, chăm sóc lấy ý kiến đánh giá từ khách hàng là rất quan trọng. Nhưng vì khi có lượng lớn ý kiến đánh giá như vậy, việc phân tích theo dõi và tiếp nhận những ý kiến đó theo cách thủ công truyền thống là vô cùng khó khăn. Vì thế, cần một giải pháp thực tiễn nào đó để việc tiếp nhận được ý kiến của lượng lớn khách hàng một cách nhanh chống và hoàn toàn tự động.

Cùng với đó, việc phát triển của trí tuệ nhân tạo những năm gần đây mở ra một ý tưởng trong việc tự động hóa các tác vụ vốn trước đây cần đến sự can thiệp thủ công, trong đó có bài toán phân tích và phân loại cảm xúc từ phản hồi của khách hàng. Với sự hỗ trợ của các kỹ thuật xử lý ngôn ngữ tự nhiên kết hợp cùng các mô hình học máy và học sâu, hoàn toàn có thể xây dựng một hệ thống tự động có khả năng hiểu và phân loại cảm xúc lượng lớn ý kiến đánh giá của khách hàng một cách nhanh chóng, chính xác và hiệu quả.

Đặc biệt, trong bối cảnh ngôn ngữ tiếng Việt vẫn còn hạn chế về tài nguyên xử lý ngôn ngữ so với các ngôn ngữ lớn như tiếng Anh, việc phát triển các hệ thống phân tích cảm xúc dành riêng cho tiếng Việt mang ý nghĩa rất quan trọng cả về mặt ứng dụng lẫn nghiên cứu. Không chỉ giúp cửa hàng, doanh nghiệp nắm bắt nhanh ý kiến cảm nhận của khách hàng, cải thiện chất lượng sản phẩm, dịch vụ mà còn góp phần thúc đẩy sự phát triển của các công nghệ ngôn ngữ tự nhiên cho tiếng Việt.

“Phân loại cảm xúc tiếng Việt bằng mô hình NLP” không chỉ có giá trị trong phạm vi học thuật mà còn mang tính thực tiễn cao. Nó hướng đến việc ứng dụng công nghệ hiện đại nhằm giải quyết bài toán thực tế đang được quan tâm, đồng thời là cơ hội để người học tiếp cận và làm chủ các công cụ, kỹ thuật mới trong lĩnh vực trí tuệ nhân tạo, lĩnh vực đang ngày càng phát triển mạnh mẽ và đầy tiềm năng trong thời đại số hóa hiện nay.

## Mục tiêu đề tài

**Mục tiêu chính** của đề tài “Phân loại cảm xúc tiếng Việt bằng mô hình NLP” là nghiên cứu và xây dựng một hệ thống có khả năng tự động phân tích và phân loại cảm xúc từ các bình luận tiếng Việt, cụ thể là các bình luận người dùng trên các nền tảng thương mại điện tử. Hệ thống này cần có khả năng nhận diện và phân loại chính xác bình luận thành ba nhóm cảm xúc: tích cực, tiêu cực và trung lập, từ đó hỗ trợ các cửa hàng và doanh nghiệp hiểu rõ hơn về cảm nhận của khách hàng đối với sản phẩm, dịch vụ của mình.

**Về mục tiêu học tập, nghiên cứu:**

Nhằm củng cố và mở rộng kiến thức, kỹ năng chuyên môn trong lĩnh vực trí tuệ nhân tạo, đặc biệt là xử lý ngôn ngữ tự nhiên và học máy.

Hiểu và áp dụng kiến thức về xử lý ngôn ngữ tự nhiên tiếng Việt: Tìm hiểu sâu về đặc trưng ngôn ngữ tiếng Việt, bao gồm tách từ, chuẩn hóa văn bản, xử lý văn bản, đặc biệt là làm việc với ngôn ngữ thực tế trong các bình luận.

Làm quen và sử dụng các mô hình học máy và học sâu trong xử lý ngôn ngữ tự nhiên: Học cách sử dụng và so sánh hiệu quả giữa các thuật toán học máy truyền thống như Naïve Bayes và các mô hình học sâu tiên tiến như BERT, đặc biệt là các mô hình tiền huấn luyện dành riêng cho tiếng Việt như PhoBERT.

Cải thiện khả năng nghiên cứu khoa học và tư duy phản biện: Tìm hiểu, phân tích tài liệu nghiên cứu, đánh giá kết quả thực nghiệm, so sánh mô hình, và đưa ra giải pháp cải tiến là những bước quan trọng giúp rèn luyện tư duy khoa học và nâng cao năng lực nghiên cứu độc lập.

## Phạm vi nghiên cứu

Đề tài tập trung vào việc ứng dụng các kỹ thuật xử lý ngôn ngữ tự nhiên, học máy và học sâu để tự động phân loại cảm xúc trong các bình luận tiếng Việt. Cụ thể, phạm vi của đề tài được giới hạn trong ngữ cảnh phân tích cảm xúc của người dùng khi phản hồi sản phẩm trên các nền tảng thương mại điện tử phổ biến tại Việt Nam. Đây là môi trường dữ liệu thực tế, nơi người dùng thường xuyên để lại nhận xét, đánh giá về chất lượng sản phẩm, dịch vụ đã sử dụng. Việc lựa chọn dữ liệu từ các trang thương mại điện tử giúp đề tài tiếp cận sát với nhu cầu ứng dụng trong thực tiễn, đồng thời mang đến những thách thức thú vị trong quá trình xử lý ngôn ngữ tự nhiên tiếng Việt.

Dữ liệu của đề tài là các bình luận dạng văn bản, được trích xuất từ trang thương mại điện tử Tiki. Cảm xúc của các bình luận sẽ được phân loại thành ba nhóm cơ bản: tích cực, tiêu cực và trung lập. Đây là cách phân loại cảm xúc đơn giản nhưng hiệu quả, phù hợp với bài toán phân tích cảm xúc ở cấp độ ứng dụng. Việc gán nhãn được thực hiện một cách thủ công, tổng hợp từ nhiều người.

Đề tài áp dụng các bước tiền xử lý văn bản phù hợp với tiếng Việt, bao gồm: chuẩn hóa chữ viết, tách từ, loại bỏ ký tự đặc biệt, dấu câu. Sau khi tiền xử lý, dữ liệu sẽ được đưa vào các mô hình học máy và học sâu để huấn luyện. Các mô hình truyền thống như Naive Bayes được sử dụng như cơ sở so sánh, sau đó mô hình PhoBERT sẽ được áp dụng để nâng cao độ chính xác. Việc huấn luyện và đánh giá mô hình được thực hiện trên máy tính cá nhân cùng với nền tảng Google Colab.

## Phương pháp nghiên cứu

Để đạt được mục tiêu của đề tài, quá trình thực hiện sẽ được chia thành các bước cụ thể, tuần tự theo quy trình xử lý dữ liệu và phát triển mô hình học máy và học sâu trong lĩnh vực xử lý ngôn ngữ tự nhiên. Cụ thể, các phương pháp được áp dụng bao gồm:

**Khảo sát lý thuyết và tổng quan nghiên cứu:** Tìm hiểu các khái niệm liên quan đến phân tích cảm xúc, các cấp độ cảm xúc và các loại cảm xúc phổ biến. Nghiên cứu các phương pháp dựa trên học máy và học sâu. Khảo sát các công trình nghiên cứu trước đó liên quan đến phân tích cảm xúc trong tiếng Việt và các ngôn ngữ khác.

Thu thập và xử lý dữ liệu

Thu thập dữ liệu các bình luận của người dùng trên các nền tảng thương mại điện tử phổ biến tại Việt Nam là Tiki. Các bình luận này được viết bằng tiếng Việt, với văn tự nhiên, đôi khi có cả các từ viết tắt hoặc không có dấu câu. Điều này mang lại tính thực tế cao, phản ánh đúng với thực tế trong việc ứng dụng bài toán phân loại cảm xúc.

Sau khi thu thập dữ liệu, tiếp theo là xử lý và gán nhãn cảm xúc cho từng bình luận. Xử lý loại bỏ các ký tự đặc biệt, các biểu tượng cảm xúc, các dấu câu,… Để phục vụ cho mục tiêu nghiên cứu, mỗi bình luận sẽ được phân loại vào một trong ba nhãn cảm xúc: tích cực, tiêu cực, hoặc trung lập theo các tiêu chí riêng về các từ cảm xúc và được thực hiện bởi nhiều người, sau đó tổng hợp thống nhất về nhãn của dữ liệu.

**Xây dựng mô hình học máy và học sâu:** Thử nghiệm các mô hình học máy cổ điển để làm cơ sở so sánh, triển khai các mô hình học sâu phổ biến cho xử lý chuỗi văn bản, tinh chỉnh tham số và tối ưu mô hình.

**Huấn luyện và đánh giá mô hình:** Chia dữ liệu thành các tập huấn luyện, kiểm thử và kiểm tra theo tỷ lệ hợp lý, sử dụng các thước đo đánh giá như Accuracy, Precision, Recall, F1-score, Confusion Matrix để đo lường hiệu quả mô hình, so sánh kết quả giữa các mô hình học máy, học sâu và mô hình tiền huấn luyện để chọn ra phương án tối ưu.

**Triển khai ứng dụng thử nghiệm:** Phát triển một ứng dụng web đơn giản cho phép người dùng nhập vào đường dẫn của sản phẩm trên các sàn thương mại điện tử như Shopee, Tiki, Lazada, sau đó hệ thống sẽ tự động đánh giá các bình luận của sản phẩm đó và trả kết quả sau khi phân tích về cho người dùng, cụ thể ở đây là ứng dụng cho các cửa hàng và doanh nghiệp.

**Tổng hợp kết quả và đề xuất hướng phát triển:** Đưa ra nhận xét, phân tích những ưu, nhược điểm của các mô hình đã thử nghiệm. So sánh hiệu quả mô hình với các kết quả trong các công trình nghiên cứu khác. Đề xuất hướng phát triển tiếp theo như mở rộng phạm vi cảm xúc, áp dụng cho các lĩnh vực khác như y tế, giáo dục,...

# CƠ SỞ LÝ THUYẾT

## Tổng quan về xử lý ngôn ngữ tự nhiên

### Khái niệm

Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) là một lĩnh vực liên ngành giữa khoa học máy tính, trí tuệ nhân tạo và ngôn ngữ học, nhằm mục tiêu xây dựng các hệ thống có khả năng hiểu, diễn giải và tạo ra ngôn ngữ của con người một cách tự động. NLP đóng vai trò trung gian trong việc giao tiếp giữa con người và máy tính bằng ngôn ngữ tự nhiên, cho phép máy tính xử lý và phân tích lượng lớn văn bản, lời nói mà không cần đến các lệnh lập trình phức tạp từ phía người dùng.

Với sự phát triển mạnh mẽ của dữ liệu văn bản trên các nền tảng số như mạng xã hội, thương mại điện tử, diễn đàn trực tuyến,... NLP ngày càng trở thành một công cụ không thể thiếu trong việc phân tích dữ liệu phi cấu trúc. Các ứng dụng phổ biến của NLP có thể kể đến như: phân tích cảm xúc, chatbot, dịch máy, trích xuất thông tin, tóm tắt văn bản, nhận dạng thực thể, và nhiều ứng dụng khác.

Trong những năm gần đây, sự phát triển của các mô hình ngôn ngữ lớn như BERT, GPT,... đã mang lại bước tiến vượt bậc cho NLP, đặc biệt là trong các bài toán phức tạp như phân loại cảm xúc, dịch máy, và hiểu ngữ cảnh sâu. Đối với tiếng Việt, mô hình PhoBERT được huấn luyện riêng cho dữ liệu tiếng Việt đã chứng minh hiệu quả vượt trội trong nhiều tác vụ, bao gồm phân tích cảm xúc.

### Quy trình xử lý ngôn ngữ tự nhiên

Quy trình xử lý ngôn ngữ tự nhiên thường bao gồm nhiều giai đoạn:

**Thu thập và tiền xử lý dữ liệu văn bản:** Dữ liệu văn bản ban đầu thường chứa nhiều yếu tố không cần thiết như dấu câu, ký tự đặc biệt, từ dừng, từ viết tắt, lỗi chính tả,... nên cần được làm sạch và chuẩn hóa. Đối với tiếng Việt, bước tách từ là cực kỳ quan trọng do tiếng Việt là ngôn ngữ đơn âm tiết, không có dấu cách rõ ràng giữa các từ như tiếng Anh.

**Biểu diễn văn bản:** Sau khi tiền xử lý, văn bản cần được chuyển đổi thành dạng số để mô hình có thể xử lý được. Một số kỹ thuật phổ biến gồm có: Bag of Words (BoW), TF-IDF, word embeddings (Word2Vec, GloVe)…

**Xây dựng mô hình học máy và học sâu:** Tùy theo bài toán và yêu cầu về độ chính xác, người dùng có thể lựa chọn mô hình học máy truyền thống như Naive Bayes, SVM, Decision Tree,... hoặc các mô hình học sâu như CNN, RNN, LSTM, Transformer.

**Đánh giá và tối ưu mô hình:** Các mô hình sau khi huấn luyện cần được đánh giá bằng các chỉ số như độ chính xác (accuracy), độ bao phủ (recall), độ chính xác (precision), F1-score,... để so sánh và lựa chọn mô hình tốt nhất.

## Phân tích cảm xúc

### Khái niệm

Phân tích cảm xúc (Sentiment Analysis) hay còn gọi là phân loại cảm xúc, là một nhánh quan trọng trong xử lý ngôn ngữ tự nhiên (NLP), tập trung vào việc xác định thái độ, quan điểm hoặc cảm xúc của người viết đối với một chủ đề cụ thể trong văn bản. Mục tiêu chính của phân tích cảm xúc là xác định liệu văn bản thể hiện cảm xúc tích cực, tiêu cực hay trung lập, từ đó giúp các hệ thống hiểu được “ý định” hoặc “cảm nhận” của người dùng khi giao tiếp bằng ngôn ngữ tự nhiên.

Phân tích cảm xúc có rất nhiều ứng dụng trong thực tế, đặc biệt là trong lĩnh vực kinh doanh và chăm sóc khách hàng. Các doanh nghiệp có thể sử dụng phân tích cảm xúc để theo dõi phản hồi của người tiêu dùng, đánh giá mức độ hài lòng về sản phẩm, dịch vụ hoặc thương hiệu, từ đó đưa ra chiến lược cải thiện phù hợp. Ngoài ra, phân tích cảm xúc còn được ứng dụng trong phân tích dư luận xã hội, đánh giá hiệu quả chiến dịch truyền thông, quản lý danh tiếng và nhiều lĩnh vực khác.

Về mặt kỹ thuật, phân tích cảm xúc có thể được triển khai dưới nhiều mức độ khác nhau:

**Mức câu:** Xác định cảm xúc tổng thể của từng câu.

**Mức đoạn văn, bình luận:** Đánh giá cảm xúc chung của toàn bộ văn bản hoặc bài viết.

**Mức thực thể:** Phân tích cảm xúc hướng đến các khía cạnh cụ thể (ví dụ: người dùng thích "giá cả" nhưng không hài lòng với "chất lượng sản phẩm").

Đối với ngôn ngữ tiếng Việt, phân tích cảm xúc là một thách thức lớn do đặc điểm ngữ pháp đặc biệt, cách viết không chuẩn, sử dụng từ láy, tiếng lóng, viết tắt, cũng như hiện tượng từ đồng âm, dị nghĩa phổ biến. Do đó, việc áp dụng các mô hình hiện đại được thiết kế riêng cho tiếng Việt như PhoBERT sẽ mang lại kết quả tốt hơn so với việc sử dụng các mô hình học từ tiếng Anh hoặc ngôn ngữ khác.

Phân tích cảm xúc là một công cụ hữu ích giúp biến đổi dữ liệu văn bản phi cấu trúc thành thông tin có giá trị. Trong thời đại số hóa và dữ liệu lớn, việc áp dụng các kỹ thuật phân tích cảm xúc không chỉ mang lại lợi thế cạnh tranh cho doanh nghiệp mà còn mở ra nhiều hướng nghiên cứu và ứng dụng trong lĩnh vực trí tuệ nhân tạo và xử lý ngôn ngữ tự nhiên.

### Các phương pháp phân tích cảm xúc

**Phương pháp dựa trên từ điển cảm xúc:** Sử dụng một tập hợp từ điển chứa các từ có gán nhãn cảm xúc (ví dụ: “tuyệt vời” là tích cực, “tệ” là tiêu cực). Văn bản được phân tích dựa trên việc đếm và tổng hợp các từ có trong từ điển để đưa ra dự đoán về cảm xúc. Tuy đơn giản và dễ triển khai, phương pháp này thường thiếu hiệu quả trong việc xử lý các ngữ cảnh phức tạp hoặc câu mang tính mỉa mai, ẩn dụ.

**Phương pháp học máy truyền thống:** Bao gồm các mô hình như Naive Bayes, SVM, KNN,... Các mô hình này học từ dữ liệu huấn luyện đã được gán nhãn để rút ra quy tắc phân loại. Trước khi đưa vào mô hình, văn bản thường được chuyển thành dạng vector thông qua các kỹ thuật biểu diễn như TF-IDF hoặc Bag of Words.

**Phương pháp học sâu:** Các mô hình học sâu như CNN, RNN, LSTM hoặc Transformer (BERT, RoBERTa, PhoBERT) cho phép khai thác tốt hơn mối quan hệ ngữ cảnh và cấu trúc ngôn ngữ trong văn bản. Đặc biệt, các mô hình tiền huấn luyện như BERT hoặc PhoBERT có khả năng hiểu ngữ nghĩa theo ngữ cảnh sâu, giúp cải thiện đáng kể độ chính xác trong phân tích cảm xúc, đặc biệt đối với ngôn ngữ nhiều biến thể như tiếng Việt.

## Kỹ thuật biểu diễn văn bản

Trong xử lý ngôn ngữ tự nhiên, việc biểu diễn văn bản dưới dạng mà máy tính có thể hiểu và xử lý được là một bước quan trọng, đặc biệt trong các bài toán phân loại văn bản như phân loại cảm xúc. Văn bản ở dạng tự nhiên là một chuỗi ký tự hoặc từ ngữ, nhưng để xử lý bằng các thuật toán học máy hoặc học sâu, văn bản cần được chuyển đổi sang dạng số hoặc vector đặc trưng.

### Kỹ thuật Bag of Word (BoW)

Bag of Words là một trong những kỹ thuật biểu diễn văn bản đơn giản và phổ biến nhất trong lĩnh vực xử lý ngôn ngữ tự nhiên. Ý tưởng chính của BoW là đại diện cho một văn bản như một tập hợp các từ ngữ (không quan tâm đến thứ tự) và biểu diễn văn bản dưới dạng vector dựa trên tần suất xuất hiện của từng từ trong tập từ vựng.

Cụ thể, với một tập hợp các văn bản, trước tiên người ta xây dựng một tập từ vựng bao gồm tất cả các từ xuất hiện trong tập dữ liệu huấn luyện. Mỗi văn bản sau đó sẽ được biểu diễn bằng một vector có chiều bằng kích thước của tập từ vựng. Mỗi phần tử trong vector đại diện cho số lần xuất hiện của một từ cụ thể trong văn bản đó. Ví dụ, nếu từ "vui" xuất hiện ba lần trong văn bản, thì phần tử tương ứng với từ "vui" trong vector sẽ có giá trị là 3.

BoW có ưu điểm là dễ hiểu, dễ triển khai và hiệu quả trong các bài toán phân loại cơ bản. Tuy nhiên, kỹ thuật này tồn tại nhiều hạn chế. Trước hết, nó bỏ qua thứ tự từ trong câu, do đó không nắm bắt được ngữ nghĩa đầy đủ của văn bản. Ngoài ra, BoW không thể hiện được mối quan hệ ngữ cảnh giữa các từ, ví dụ như sự khác biệt giữa "không vui" và "rất vui" có thể bị mất đi khi chỉ xét đến tần suất của từ "vui".

Một vấn đề khác là độ hiệu quả tính toán: với các tập từ vựng lớn, vector BoW sẽ có kích thước rất cao và thường rất thưa, gây tốn bộ nhớ và ảnh hưởng đến tốc độ xử lý.

Mặc dù có nhiều điểm hạn chế, BoW vẫn là một kỹ thuật nền tảng quan trọng trong lĩnh vực NLP và thường được dùng như một bước khởi đầu để so sánh với các mô hình phức tạp hơn như TF-IDF hay Word Embedding. Trong các bài toán phân loại cảm xúc tiếng Việt, BoW có thể hoạt động tốt với dữ liệu nhỏ và khi kết hợp với các kỹ thuật tiền xử lý, chuẩn hóa từ hoặc phân tách từ chính xác.

### Kỹ thuật TF-IDF

TF-IDF (Term Frequency - Inverse Document Frequency) là một kỹ thuật biểu diễn văn bản nhằm đánh giá mức độ quan trọng của một từ trong một văn bản so với toàn bộ tập hợp các văn bản. Đây là sự cải tiến của mô hình Bag of Words, với mục tiêu không chỉ xét đến tần suất xuất hiện của từ trong văn bản, mà còn điều chỉnh dựa trên mức độ phổ biến của từ đó trên toàn bộ tập dữ liệu.

Cụ thể, TF-IDF là tích của hai thành phần:

**TF (Term Frequency):** đo tần suất xuất hiện của từ trong văn bản. Có thể tính bằng công thức:

TF(t, d) = (số lần từ t xuất hiện trong văn bản d) / (tổng số từ trong d)

**IDF (Inverse Document Frequency):** đo mức độ đặc trưng của từ, bằng cách giảm trọng số của các từ xuất hiện nhiều trong toàn bộ tập văn bản. Công thức thường dùng là:

    IDF(t) = log(N / (df(t) + 1))

Trong đó, N là tổng số văn bản trong tập dữ liệu, còn df(t) là số văn bản chứa từ t. Việc cộng thêm 1 vào mẫu số giúp tránh chia cho 0.

Khi nhân TF với IDF, ta được giá trị TF-IDF thể hiện độ quan trọng tương đối của từ trong một văn bản cụ thể. Các từ phổ biến như “và”, “là”, “của” có thể có TF cao nhưng do xuất hiện nhiều trong hầu hết các văn bản nên sẽ có IDF thấp, dẫn đến TF-IDF thấp. Ngược lại, các từ hiếm nhưng mang tính phân biệt cao (ví dụ: “phấn khích”, “tức giận”, “buồn bã”) sẽ có trọng số cao, từ đó giúp mô hình học được những đặc điểm đặc trưng của từng cảm xúc.

Ưu điểm chính của TF-IDF là khả năng làm nổi bật các từ khóa quan trọng và loại bỏ các từ mang tính chung chung. Trong các bài toán phân loại cảm xúc, TF-IDF giúp mô hình nhận diện được các từ thể hiện cảm xúc rõ ràng, từ đó nâng cao hiệu quả phân loại.

Tuy nhiên, tương tự như Bag of Words, TF-IDF không xét đến ngữ cảnh hoặc thứ tự từ trong câu. Điều này có thể gây khó khăn trong việc hiểu các cấu trúc ngữ nghĩa phức tạp như phủ định ("không vui") hoặc các cụm từ có sắc thái cảm xúc đặc biệt. Ngoài ra, biểu diễn văn bản dưới dạng TF-IDF vẫn tạo ra vector thưa và kích thước cao, gây tốn tài nguyên tính toán với các tập dữ liệu lớn.

Mặc dù vậy, TF-IDF vẫn là một kỹ thuật mạnh mẽ và thường được dùng làm baseline trong các mô hình phân loại văn bản, bao gồm cả phân loại cảm xúc tiếng Việt. Khi kết hợp với các thuật toán học máy truyền thống như Naive Bayes, SVM hay Logistic Regression, TF-IDF có thể đạt hiệu quả tốt trên các tập dữ liệu vừa và nhỏ.

## Tổng quan về học máy

### Khái niệm

Học máy (Machine Learning – ML) là một lĩnh vực thuộc Trí tuệ nhân tạo (AI) cho phép máy tính học từ dữ liệu mà không cần được lập trình một cách rõ ràng. Mục tiêu của học máy là xây dựng các mô hình có khả năng suy luận hoặc đưa ra dự đoán dựa trên các mẫu đã học được từ dữ liệu quá khứ.

### Phân loại học máy

Học máy có thể chia thành ba nhóm chính:

**Học có giám sát (Supervised Learning):** Là hình thức học dựa trên dữ liệu có nhãn. Mô hình được cung cấp các cặp dữ liệu đầu vào và đầu ra tương ứng để học cách ánh xạ giữa chúng. Ví dụ phân loại văn bản, hồi quy giá trị,...

**Học không giám sát (Unsupervised Learning):** Mô hình học từ dữ liệu chưa có nhãn, thường dùng trong các bài toán như phân cụm, giảm chiều dữ liệu,... Ví dụ nhóm các bình luận có nội dung tương tự nhau.

**Học tăng cường (Reinforcement Learning):** Mô hình học thông qua việc tương tác với môi trường và nhận phần thưởng (reward) hoặc hình phạt (penalty). Lĩnh vực này phổ biến trong robot, trò chơi, hoặc tối ưu hệ thống.

### Một số thuật toán học máy phổ biến trong NLP

Trong các bài toán xử lý ngôn ngữ tự nhiên, đặc biệt là phân loại văn bản và cảm xúc, học máy truyền thống vẫn đóng vai trò quan trọng nhờ khả năng huấn luyện nhanh, dễ triển khai và hiệu quả tốt khi dữ liệu không quá lớn.

Thuật toán học máy phổ biến trong bài toán xử lý ngôn ngữ tự nhiên:

**Naive Bayes** là một trong những thuật toán đơn giản và hiệu quả nhất trong phân loại văn bản. Dựa trên định lý Bayes và giả định độc lập giữa các đặc trưng, mô hình này tính xác suất một văn bản thuộc về một nhãn cụ thể dựa trên tần suất xuất hiện của các từ. Mặc dù giả định độc lập hiếm khi đúng trong thực tế, Naive Bayes vẫn cho kết quả đáng tin cậy trong nhiều ứng dụng thực tế, đặc biệt là với dữ liệu văn bản ngắn và đơn giản như bình luận hoặc đánh giá sản phẩm.

**Hồi quy logistic (Logistic Regression)** là một mô hình phân loại nhị phân truyền thống, hoạt động bằng cách học một hàm tuyến tính giữa các đặc trưng đầu vào và xác suất đầu ra thuộc một lớp cụ thể. Mặc dù mang tên "hồi quy", nhưng Hồi quy logistic thực chất là một mô hình phân loại, thường được mở rộng cho các bài toán đa lớp. Mô hình này có thể diễn giải tốt, dễ huấn luyện và hoạt động ổn định với các đặc trưng rời rạc như từ vựng trong văn bản.

**Support Vector Machine (SVM)** là một thuật toán mạnh mẽ và phổ biến trong phân loại văn bản, đặc biệt là trong các bài toán yêu cầu độ chính xác cao. SVM hoạt động bằng cách tìm một siêu phẳng tối ưu để phân tách các lớp dữ liệu. Trong không gian đặc trưng cao chiều như biểu diễn văn bản (TF-IDF hoặc Word Embedding), SVM thường cho kết quả rất tốt nhờ khả năng tối đa hóa biên phân cách giữa các lớp.

Ngoài ra, còn có một số thuật toán ít phổ biến hơn trong việc xử lý ngôn ngữ tự nhiên như K láng giềng gần nhất (K-Nearest Neighbors – KNN), Cây quyết định (Decision Tree) và Rừng ngẫu nhiên (Random Forest), do hiệu quả đem lại không cao mà chi phí tính toán cao khi dữ liệu càng lớn.

Các thuật toán học máy trên đều cần dữ liệu đầu vào là dạng số, cho nên khi xử lý ngôn ngữ tự nhiên, bước tiền xử lý dữ liệu và trích xuất đặc trưng là vô cùng quan trọng. Việc lựa chọn đúng phương pháp và kết hợp với thuật toán học máy phù hợp sẽ mang lại kết quả tốt cho mô hình bài toán cần huấn luyện.

### Ưu nhược điểm của học máy

#### Ưu điểm

**Tự động học từ dữ liệu:** Thay vì xây dựng các luật thủ công để phân loại, học máy cho phép mô hình học trực tiếp từ dữ liệu đầu vào và đưa ra quyết định dựa trên mẫu đã học. Điều này giúp tiết kiệm thời gian và công sức khi xử lý các bài toán có nhiều tình huống không thể lập trình tường minh.

**Tổng quát hóa tốt:** Với một tập dữ liệu đủ lớn và đa dạng, mô hình học máy có thể khái quát hóa và dự đoán chính xác cho các trường hợp chưa từng gặp, đặc biệt hiệu quả trong các bài toán như phân loại văn bản hay cảm xúc.

**Hiệu quả với dữ liệu vừa phải:** Các thuật toán như Naive Bayes, SVM hay Logistic Regression có thể đạt hiệu suất tốt ngay cả khi không cần lượng dữ liệu quá lớn.

**Tốc độ huấn luyện nhanh, dễ triển khai:** Các mô hình học máy có tốc độ huấn luyện nhanh, ít tốn tài nguyên tính toán, phù hợp với các bài toán vừa và nhỏ hoặc yêu cầu triển khai nhanh.

**Dễ giải thích và kiểm soát:** Một số mô hình học máy (Logistic Regression, Cây quyết định,…) có khả năng giải thích rõ ràng cách mô hình đưa ra dự đoán, hỗ trợ tốt cho các hệ thống cần tính minh bạch.

#### Nhược điểm

**Phụ thuộc nhiều vào trích xuất đặc trưng:** Hiệu quả của mô hình học máy phụ thuộc rất lớn vào cách biểu diễn dữ liệu đầu vào. Nếu đặc trưng không phản ánh đúng ngữ nghĩa hoặc không loại bỏ được nhiễu, mô hình dễ đưa ra kết quả sai lệch.

**Khó nắm bắt ngữ cảnh và ngữ nghĩa sâu:** Các mô hình học máy truyền thống thường dựa vào tần suất từ vựng (như TF-IDF), nên không thể hiểu được ngữ nghĩa sâu hoặc mối liên hệ dài giữa các từ trong văn bản, đặc biệt là với các ngôn ngữ giàu ngữ cảnh như tiếng Việt.

**Dễ bị ảnh hưởng bởi dữ liệu nhiễu hoặc mất cân bằng nhãn:** Khi tập dữ liệu chứa nhiều bình luận rác, lỗi chính tả, từ không dấu hoặc chênh lệch quá nhiều giữa các nhãn (ví dụ nhãn "tích cực" chiếm đa số), mô hình dễ bị học lệch và suy giảm độ chính xác.

**Không thích hợp với dữ liệu lớn hoặc quá phức tạp:** Với những bài toán yêu cầu hiểu sâu về ngôn ngữ hoặc xử lý dữ liệu rất lớn, các mô hình học máy truyền thống có thể không tốt về hiệu năng và độ chính xác.

## Tổng quan về học sâu

### Khái niệm

Học sâu (Deep Learning – DL) là một nhánh mở rộng của học máy, tập trung vào việc xây dựng và huấn luyện các mạng nơ-ron nhân tạo nhiều, nhằm mô phỏng cách thức hoạt động của bộ não con người trong việc xử lý dữ liệu và trích xuất thông tin. Điểm nổi bật của học sâu so với học máy truyền thống là khả năng tự động học biểu diễn đặc trưng từ dữ liệu thô mà không cần sự can thiệp thủ công từ con người.

Trong học sâu, các mô hình mạng nơ-ron bao gồm nhiều lớp ẩn (hidden layers), cho phép hệ thống học được các đặc trưng ở nhiều cấp độ khác nhau. Với dữ liệu văn bản, các lớp đầu tiên thường học thông tin cơ bản như từ vựng hoặc ký tự, trong khi các lớp sâu hơn có thể học được các cấu trúc ngữ pháp, ngữ nghĩa hoặc thậm chí cả sắc thái cảm xúc của câu. Chính nhờ khả năng học được biểu diễn ngữ nghĩa phức tạp, học sâu đã đạt được nhiều bước tiến vượt bậc trong các bài toán xử lý ngôn ngữ tự nhiên (NLP) như dịch máy, phân loại cảm xúc, tóm tắt văn bản, hoặc trả lời câu hỏi.

### Một số kiến trúc học sâu tiêu biểu trong NLP

**Mạng nơ-ron tích chập (Convolutional Neural Network – CNN):** Dù ban đầu được thiết kế cho xử lý ảnh, CNN cũng được sử dụng hiệu quả trong phân loại văn bản nhờ khả năng nhận diện các mẫu cục bộ trong chuỗi từ, ví dụ như cụm từ biểu thị cảm xúc.

**Mạng nơ-ron hồi tiếp (Recurrent Neural Network – RNN):** Đây là các mô hình phù hợp với dữ liệu chuỗi như văn bản, có khả năng ghi nhớ thông tin từ các bước trước đó trong câu. RNN được sử dụng rộng rãi trong các bài toán phân tích cảm xúc, phân tích ý định, hoặc sinh văn bản.

**Transformer:** Là kiến trúc mang tính đột phá trong NLP hiện đại, Transformer giúp xử lý ngữ cảnh dài tốt hơn nhờ cơ chế tự chú ý (self-attention). Các mô hình như BERT, GPT, RoBERTa, XLM-R,... đều dựa trên kiến trúc này. Đặc biệt, các mô hình tiền huấn luyện như BERT đã chứng minh khả năng hiểu ngữ nghĩa và cảm xúc sâu sắc từ văn bản đầu vào, kể cả trong tiếng Việt, nhờ được huấn luyện trên kho dữ liệu lớn.

### Ưu nhược điểm của học sâu

#### Ưu điểm

**Tự động trích xuất đặc trưng:** Một trong những điểm mạnh nổi bật nhất của học sâu là khả năng tự động học đặc trưng từ dữ liệu đầu vào, mà không cần phải xử lý thủ công các đặc trưng như trong học máy truyền thống. Điều này giúp giảm thiểu công sức tiền xử lý và loại bỏ sự phụ thuộc vào kiến thức chuyên môn trong việc lựa chọn đặc trưng.

**Hiểu ngữ nghĩa và ngữ cảnh tốt hơn:** Các kiến trúc học sâu như RNN, LSTM, và đặc biệt là Transformer (BERT, GPT,...) có khả năng nắm bắt quan hệ ngữ cảnh dài trong văn bản, hiểu được ý nghĩa câu và sắc thái cảm xúc phức tạp – điều mà các mô hình học máy truyền thống khó thực hiện.

**Hiệu suất vượt trội với dữ liệu lớn:** Khi được huấn luyện trên tập dữ liệu đủ lớn, mô hình học sâu thường đạt độ chính xác cao hơn đáng kể so với các mô hình học máy, nhờ khả năng học được các biểu diễn ngôn ngữ trừu tượng hơn.

**Khả năng mở rộng linh hoạt:** Học sâu có thể áp dụng cho nhiều bài toán NLP khác nhau như phân loại cảm xúc, sinh văn bản, dịch máy, phân tích thực thể,... chỉ cần điều chỉnh đầu ra và kiến trúc phù hợp.

#### Nhược điểm

**Đòi hỏi tài nguyên tính toán lớn:** Việc huấn luyện các mô hình học sâu yêu cầu phần cứng mạnh (như GPU), đặc biệt với các mô hình nhiều tham số như BERT hoặc GPT. Điều này làm tăng chi phí triển khai và khó tiếp cận với các dự án có nguồn lực hạn chế.

**Cần lượng dữ liệu lớn để đạt hiệu quả cao:** Mô hình học sâu hoạt động tốt nhất khi có một tập dữ liệu huấn luyện phong phú và đa dạng. Trong các trường hợp dữ liệu ít hoặc mất cân bằng nhãn (ví dụ: số lượng nhãn "trung lập" ít hơn nhiều so với "tích cực"), mô hình có thể bị lệch hoặc quá khớp.

**Khó giải thích và thiếu minh bạch:** Các mô hình học sâu thường khó lý giải cụ thể vì sao mô hình đưa ra một dự đoán nhất định. Điều này là một rào cản trong các hệ thống yêu cầu giải thích rõ ràng, như các ứng dụng trong y tế, pháp luật, hoặc tài chính.

**Thời gian huấn luyện lâu và khó điều chỉnh:** Quá trình huấn luyện mô hình học sâu thường kéo dài, đòi hỏi phải thử nghiệm nhiều lần với các siêu tham số khác nhau (learning rate, batch size, số lớp, v.v.). Việc tinh chỉnh mô hình hiệu quả cũng đòi hỏi kinh nghiệm và hiểu biết sâu về mạng nơ-ron.

## Thuật toán Naïve Bayes

### Khái niệm

Naïve Bayes là một trong những thuật toán học máy có giám sát đơn giản nhưng hiệu quả, đặc biệt phù hợp với các bài toán phân loại văn bản như phân loại cảm xúc, phân loại thư rác, hoặc phân loại chủ đề văn bản. Đây là thuật toán dựa trên định lý Bayes trong xác suất thống kê, kết hợp với giả định “ngây thơ” (naïve) rằng các đặc trưng là độc lập có điều kiện với nhau.

### Định lý Bayes

Thuật toán Naïve Bayes dựa trên định lý Bayes, định lý cung cấp một phương pháp để tính xác suất hậu nghiệm *P(C|X),* tức là xác suất để một mẫu dữ liệu *X* thuộc về lớp *C,* dựa trên xác suất tiên nghiệm của lớp *C* và xác suất có điều kiện *P(X|C)*:

Trong đó:

***C:*** là nhãn phân lớp (ví dụ: tích cực, tiêu cực, trung lập).

***X:*** là đặc trưng đầu vào (thường là một chuỗi các từ trong văn bản).

***P(C∣X):*** xác suất một văn bản có đặc trưng ***X*** thuộc lớp ***C***.

***P(X∣C):*** xác suất xuất hiện của đặc trưng ***X*** khi biết văn bản thuộc lớp ***C***.

***P(C):*** xác suất tiên nghiệm của lớp ***C***.

***P(X):*** xác suất xảy ra của đặc trưng ***X***.

### Giả định “Naïve” – Độc lập có điều kiện

Naive Bayes đơn giản hóa việc tính toán xác suất *P(X∣C)* bằng giả định rằng các đặc trưng *x1, x2,…xn* trong *X* là độc lập với nhau khi biết lớp *C*:

Giả định này làm giảm đáng kể độ phức tạp trong việc tính toán, đặc biệt khi số lượng đặc trưng lớn như trong phân tích văn bản, mỗi từ trong từ điển có thể được xem như một đặc trưng.

Khi áp dụng vào bài toán phân loại văn bản, đặc trưng thường là các từ hoặc n-gram trong văn bản. Mỗi văn bản được biểu diễn như một vector từ vựng với các giá trị là số lần xuất hiện, tần suất hoặc nhị phân. Do đó, mô hình Naive Bayes sẽ tính xác suất văn bản thuộc từng lớp dựa trên tần suất xuất hiện của các từ trong các văn bản đã biết nhãn.

### Ưu điểm và nhược điểm của Naïve Bayes

#### Ưu điểm

**Đơn giản và dễ triển khai:** Naive Bayes là thuật toán học máy có cấu trúc lý thuyết đơn giản. Việc cài đặt, huấn luyện và suy luận dựa trên mô hình này không đòi hỏi nhiều tài nguyên tính toán, giúp tiết kiệm thời gian và công sức trong giai đoạn phát triển.

**Tốc độ xử lý nhanh:** Cả quá trình huấn luyện và dự đoán của Naive Bayes đều có độ phức tạp thấp, chủ yếu dựa vào việc đếm tần suất và tính xác suất, giúp mô hình hoạt động rất nhanh, đặc biệt với các tập dữ liệu lớn hoặc yêu cầu phản hồi theo thời gian thực.

**Ít bị ảnh hưởng bởi dữ liệu nhiễu:** Do mô hình dựa trên xác suất tổng thể, nên những dữ liệu nhiễu hoặc ngoại lệ nhỏ ít ảnh hưởng nghiêm trọng đến kết quả phân loại.

**Hoạt động tốt với tập dữ liệu nhỏ:** Khác với các mô hình học sâu đòi hỏi tập dữ liệu lớn để học biểu diễn tốt, Naive Bayes có thể huấn luyện hiệu quả ngay cả khi dữ liệu huấn luyện còn hạn chế về số lượng.

#### Nhược điểm

**Giả định độc lập không thực tế:** Naive Bayes giả định rằng các đặc trưng là độc lập có điều kiện với nhau, điều này thường không đúng trong thực tế, đặc biệt trong ngôn ngữ tự nhiên, các từ có mối quan hệ về ngữ nghĩa và cú pháp. Sự vi phạm giả định này có thể làm giảm hiệu quả phân loại.

**Không xét đến thứ tự từ và ngữ cảnh:** Thuật toán không thể hiểu hoặc xử lý được mối quan hệ theo ngữ cảnh giữa các từ, cũng như không thể học được các cấu trúc cú pháp hoặc ngữ nghĩa phức tạp trong câu.

**Dễ bị sai lệch nếu có từ chưa từng xuất hiện:** Nếu một từ trong văn bản kiểm tra chưa từng xuất hiện trong tập huấn luyện (thuộc một lớp cụ thể), xác suất P(xi|C) sẽ bằng 0, khiến xác suất toàn bộ văn bản bằng 0. Tuy có thể khắc phục bằng một số kỹ thuật, nhưng nó vẫn là điểm yếu.

**Độ chính xác thấp hơn so với các mô hình hiện đại:** So với các thuật toán học sâu như LSTM, Transformer (BERT, GPT), hoặc các mô hình học máy mạnh như SVM, độ chính xác của Naive Bayes thường thấp hơn trong những bài toán phức tạp hoặc yêu cầu hiểu sâu về ngữ nghĩa.

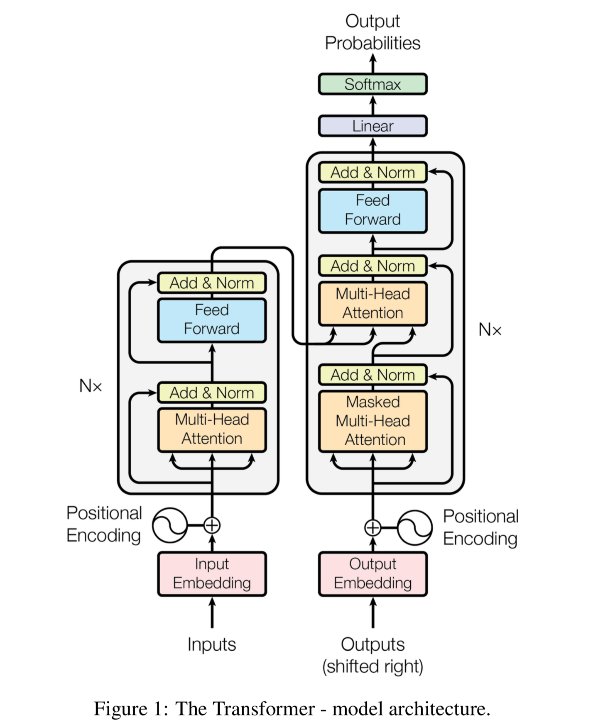
## Mô hình PhoBERT

### Khái niệm

PhoBERT là một mô hình ngôn ngữ tiền huấn luyện (pre-trained language model) dành riêng cho tiếng Việt, được phát triển dựa trên kiến trúc BERT (Bidirectional Encoder Representations from Transformers) của Google. Đây là bước tiến quan trọng trong việc áp dụng các kỹ thuật học sâu tiên tiến vào xử lý ngôn ngữ tự nhiên tiếng Việt, một ngôn ngữ phức tạp về ngữ pháp, cú pháp và có sự khác biệt rõ rệt so với tiếng Anh.

### Kiến trúc của PhoBERT

PhoBERT sử dụng kiến trúc Transformer Encoder tương tự như BERT gốc. Transformer là một mô hình mạng nơ-ron dựa trên cơ chế self-attention, cho phép mô hình học được mối quan hệ giữa các từ trong câu mà không cần dựa vào thông tin vị trí tuyệt đối như trong RNN hay CNN. Mô hình học các biểu diễn ngữ nghĩa của từ trong ngữ cảnh hai chiều, giúp nắm bắt tốt hơn ý nghĩa và cảm xúc trong câu.



Hình . Kiến trúc Transformer Encoder

Kiến trúc Transformer Encoder gồm nhiều lớp chồng lên nhau. Mỗi lớp gồm hai thành phần chính:

**Multi-Head Self-Attention:** Cho phép mô hình học được mối quan hệ giữa tất cả các từ trong câu, bất kể khoảng cách vị trí. Cơ chế này giúp PhoBERT nắm bắt ngữ cảnh toàn cục, thay vì chỉ dựa trên cửa sổ ngữ cảnh hẹp như RNN.

**Feed-Forward Neural Network:** Xử lý và biến đổi thông tin sau khi attention, giúp mô hình học được các biểu diễn đặc trưng phức tạp hơn.

Ngoài ra, mỗi lớp encoder đều có **Residual Connection** và **Layer Normalization** nhằm giữ ổn định quá trình huấn luyện và tránh mất mát thông tin khi truyền qua nhiều lớp.

PhoBERT sử dụng **positional embeddings** để mã hóa thông tin vị trí, giúp mô hình hiểu được cấu trúc câu, vì kiến trúc Transformer không tự động nhận biết được thứ tự của các từ. Các vector vị trí này được cộng trực tiếp với vector biểu diễn từ trước khi đưa vào các lớp attention.

PhoBERT có hai phiên bản chính:

**PhoBERT-base:** Có cấu hình tương đương BERT-base với 12 lớp Transformer, 768 chiều ẩn, và 12 heads.

**PhoBERT-large:** Tương đương BERT-large với 24 lớp Transformer, 1024 chiều ẩn, và 16 heads.

### Đặc điểm của PhoBERT

Điểm khác biệt quan trọng giữa PhoBERT và BERT gốc là ngôn ngữ và dữ liệu huấn luyện:

**Dữ liệu tiền huấn luyện:** PhoBERT được huấn luyện trên một tập dữ liệu văn bản tiếng Việt khổng lồ, khoảng 20GB văn bản sạch được trích xuất từ Common Crawl (gọi là Vietnamese Wikipedia + CC100-Viet). Giúp mô hình hiểu rõ hơn ngữ pháp, cách dùng từ, cấu trúc câu và sắc thái biểu cảm của tiếng Việt.

**Tiền xử lý bằng từ ghép Byte-Pair Encoding (BPE):** Không giống tiếng Anh, tiếng Việt có nhiều từ đa âm tiết được viết cách nhau như “máy tính”, “trí tuệ nhân tạo”. PhoBERT sử dụng kỹ thuật BPE giống RoBERTa để mã hóa từ, giúp tránh việc tách sai nghĩa của các từ tiếng Việt.

**Dựa trên kiến trúc RoBERTa:** PhoBERT thực chất không phải là một bản sao trực tiếp của BERT mà là bản mở rộng dựa trên RoBERTa, một cải tiến của BERT được huấn luyện với nhiều dữ liệu hơn và không sử dụng nhiệm vụ NSP (Next Sentence Prediction). Do đó, PhoBERT có khả năng biểu diễn ngữ nghĩa mạnh mẽ hơn.

### Ứng dụng PhoBERT trong phân loại cảm xúc tiếng Việt

Trong bài toán phân loại cảm xúc, mỗi văn bản đầu vào sẽ được mã hóa thành chuỗi token bằng tokenizer của PhoBERT, sau đó đưa vào mô hình để trích xuất vector biểu diễn. Vector này sẽ được đưa vào một lớp phân loại để dự đoán nhãn cảm xúc: tích cực, trung lập, hoặc tiêu cực.

**PhoBERT đặc biệt hiệu quả trong phân tích cảm xúc tiếng Việt do:**

Nắm bắt tốt mối quan hệ ngữ nghĩa phức tạp giữa các từ trong câu.

Nhận diện chính xác sắc thái biểu cảm dù ngôn từ có thể ngắn gọn, ẩn dụ hoặc có dấu hiệu mỉa mai.

Tăng đáng kể độ chính xác so với các mô hình truyền thống như Naive Bayes hoặc các mô hình word embedding đơn giản như Word2Vec, TF-IDF.

### Ưu điểm và nhược điểm của PhoBERT

#### Ưu điểm

**Được huấn luyện chuyên biệt cho tiếng Việt:** PhoBERT là mô hình đầu tiên được huấn luyện trên tập dữ liệu lớn gồm các văn bản tiếng Việt, giúp mô hình học được đặc trưng ngữ pháp, cú pháp, cấu trúc câu và sắc thái biểu cảm đặc thù của ngôn ngữ này. Điều này mang lại lợi thế lớn so với các mô hình đa ngôn ngữ trong các bài toán phân tích ngữ nghĩa tiếng Việt.

**Hiểu ngữ cảnh tốt nhờ kiến trúc Transformer:** PhoBERT sử dụng kiến trúc Transformer với cơ chế tự chú ý, cho phép mô hình hiểu được mối quan hệ giữa các từ trong cả hai chiều trái và phải của câu. Điều này rất quan trọng trong việc nắm bắt các biểu hiện cảm xúc thường được ngụ ý hoặc ẩn dụ trong văn bản tiếng Việt.

**Khả năng biểu diễn ngôn ngữ mạnh mẽ:** Các vector biểu diễn từ do PhoBERT tạo ra mang tính ngữ nghĩa cao, có thể được sử dụng hiệu quả trong nhiều tác vụ như phân loại cảm xúc, phân tích quan điểm, trích xuất thực thể, và nhiều ứng dụng NLP khác.

**Dễ dàng tinh chỉnh theo bài toán cụ thể:** PhoBERT có thể được fine-tune trên một tập dữ liệu nhỏ hơn cho các tác vụ cụ thể, giúp tăng độ chính xác trong những ứng dụng mục tiêu như phân loại cảm xúc trong đánh giá sản phẩm, ý kiến người dùng,…

#### Nhược điểm

**Yêu cầu tài nguyên tính toán lớn:** PhoBERT là mô hình học sâu với hàng triệu tham số, do đó việc huấn luyện và suy luận đòi hỏi phần cứng mạnh, đặc biệt là GPU với bộ nhớ lớn. Điều này có thể là rào cản đối với các hệ thống có tài nguyên hạn chế.

**Thời gian huấn luyện và suy luận chậm hơn:** So với các mô hình truyền thống như Naive Bayes, PhoBERT có tốc độ xử lý chậm hơn đáng kể, đặc biệt khi làm việc với tập dữ liệu lớn hoặc yêu cầu xử lý thời gian thực.

**Cần tiền xử lý dữ liệu phù hợp:** PhoBERT sử dụng tokenizer riêng dựa trên phương pháp BPE. Do đó, quá trình tiền xử lý dữ liệu đầu vào cần tuân thủ đúng định dạng yêu cầu của mô hình, nếu không sẽ dẫn đến sai lệch trong kết quả phân tích.

**Hiệu suất phụ thuộc vào kỹ thuật tinh chỉnh:** Nếu không tinh chỉnh đúng cách hoặc không chọn được siêu tham số phù hợp, mô hình PhoBERT có thể không phát huy tối đa hiệu quả. Việc tinh chỉnh cũng đòi hỏi kiến thức chuyên môn và kinh nghiệm trong lĩnh vực học sâu.

## Tổng quan về FastAPI

### Khái niệm

FastAPI là một framework hiện đại và hiệu suất cao được sử dụng để xây dựng các ứng dụng web và API bằng ngôn ngữ lập trình Python. Ra đời với mục tiêu tận dụng các tính năng mạnh mẽ của Python, FastAPI đã nhanh chóng trở thành một trong những công cụ phổ biến nhất trong việc triển khai các dịch vụ web hiện đại, đặc biệt trong lĩnh vực trí tuệ nhân tạo và học máy, nơi hiệu suất và khả năng mở rộng là yếu tố quan trọng.

FastAPI được phát triển bởi Sebastián Ramírez và lần đầu ra mắt vào năm 2018. Framework này được thiết kế với các tiêu chí: tốc độ, dễ sử dụng, hỗ trợ tốt cho kiểm thử, và khả năng tự động sinh tài liệu API.

FastAPI được xây dựng trên nền tảng của hai thư viện mạnh mẽ: **Starlette** thư viện hỗ trợ xây dựng các ứng dụng web bất đồng bộ hiệu suất cao và **Pydantic** thư viện để xác thực và phân tích dữ liệu đầu vào dựa trên type annotation. Sự kết hợp này cho phép FastAPI vừa đạt được hiệu năng cao, vừa dễ sử dụng và thân thiện với lập trình viên.

### Đặc điểm nổi bật

**Hiệu suất cao:** FastAPI có hiệu suất gần tương đương với NodeJS, nền tảng nổi tiếng về tốc độ, nhờ tận dụng cơ chế bất đồng bộ (asynchronous I/O) của Python thông qua async/await.

**Tự động sinh tài liệu API:** FastAPI hỗ trợ tự động sinh tài liệu API theo chuẩn OpenAPI (Swagger), giúp dễ dàng kiểm thử và tích hợp API. Tài liệu này được tạo dựa trên khai báo kiểu dữ liệu các tham số đầu vào và đầu ra, hoàn toàn tự động và đồng bộ với mã nguồn.

**Xác thực và phân tích dữ liệu mạnh mẽ với Pydantic:** Nhờ sử dụng thư viện Pydantic, FastAPI cho phép xác thực và chuyển đổi dữ liệu đầu vào dựa trên các kiểu dữ liệu rõ ràng, đồng thời hỗ trợ kiểm tra lỗi, cung cấp phản hồi chi tiết và giúp lập trình viên phát hiện lỗi sớm.

**Hỗ trợ type hints giúp tăng năng suất:** FastAPI tận dụng triệt để type hints trong Python để kiểm tra dữ liệu và sinh mã tài liệu, đồng thời tăng khả năng tương thích với các công cụ như VSCode, PyCharm, giúp lập trình viên có thể tự động gợi ý cú pháp và phát hiện lỗi sớm trong quá trình viết mã.

**Tương thích tốt với các thư viện Machine Learning:** FastAPI thường được sử dụng để triển khai các mô hình học máy, học sâu và xử lý ngôn ngữ tự nhiên, nhờ khả năng tích hợp mượt mà với các thư viện như TensorFlow, PyTorch, Scikit-learn, và Transformers.

### Ưu nhược điểm của FastAPI

#### Ưu điểm

**Hiệu suất cao và hỗ trợ bất đồng bộ:** FastAPI được xây dựng trên nền Starlette, một framework bất đồng bộ tốc độ cao, nhờ vậy có thể xử lý hàng nghìn request mỗi giây, rất phù hợp với các hệ thống có lưu lượng truy cập lớn hoặc yêu cầu thời gian phản hồi nhanh.

**Tự động sinh tài liệu API:** Một trong những điểm mạnh nổi bật của FastAPI là khả năng tự động sinh tài liệu API theo chuẩn OpenAPI và ReDoc. Điều này không chỉ giúp người phát triển dễ dàng kiểm thử API mà còn hỗ trợ việc tích hợp với các hệ thống bên ngoài một cách trực quan và hiệu quả.

**Hỗ trợ kiểm tra kiểu dữ liệu đầu vào:** Với sự hỗ trợ của thư viện Pydantic, FastAPI cho phép khai báo và kiểm tra kiểu dữ liệu một cách rõ ràng, giúp giảm thiểu lỗi trong quá trình xử lý yêu cầu, đồng thời tăng độ an toàn và tin cậy cho hệ thống.

**Phù hợp với các mô hình học máy:** FastAPI là sự lựa chọn lý tưởng cho việc triển khai các mô hình học máy như phân loại cảm xúc, nhận diện hình ảnh, chatbot,... vì nó hỗ trợ xử lý JSON, tích hợp dễ dàng với các thư viện như Scikit-learn, PyTorch, TensorFlow,…

#### Nhược điểm

**Cộng đồng người dùng chưa lớn:** Mặc dù FastAPI đang ngày càng phổ biến, nhưng so với các framework lâu đời như Django hay Flask, hệ sinh thái tài liệu, thư viện mở rộng và diễn đàn hỗ trợ còn hạn chế, đặc biệt với các tính năng phức tạp như quản lý cơ sở dữ liệu hoặc giao diện quản trị.

**Thiếu hệ thống quản trị tích hợp sẵn:** Khác với Django, FastAPI không tích hợp sẵn hệ thống quản trị, nên nếu cần xây dựng các công cụ quản trị nội bộ, nhà phát triển phải tự xây dựng từ đầu hoặc tích hợp thêm các giải pháp bên ngoài.

**Khó khăn khi phát triển ứng dụng quy mô lớn:** Khi xây dựng các ứng dụng phức tạp với nhiều tầng logic và mô-đun, FastAPI yêu cầu người phát triển phải tự tổ chức cấu trúc mã nguồn hợp lý. Nếu không có kinh nghiệm, điều này có thể dẫn đến khó bảo trì và mở rộng hệ thống.

**Đòi hỏi kiến thức về bất đồng bộ:** FastAPI khuyến khích sử dụng lập trình bất đồng bộ (async/await), tuy mang lại hiệu suất cao nhưng cũng có thể gây khó khăn cho người mới bắt đầu.

## Tổng quan về NextJS

### Tổng quan về NextJS

NextJS là một framework mã nguồn mở được phát triển dựa trên React, cho phép xây dựng các ứng dụng web với nhiều tính năng tối ưu hóa, bao gồm Server-Side Rendering (SSR) và Static Site Generation (SSG). Được phát triển bởi công ty Vercel, NextJS giúp tăng cường hiệu suất và khả năng SEO cho các ứng dụng web bằng cách cho phép máy chủ xử lý dữ liệu và tạo HTML trước khi gửi đến trình duyệt.

### Ưu nhược điểm của NextJS

#### Ưu điểm

**Tối ưu hóa SEO:** Hỗ trợ Server-Side Rendering (SSR) và Static Site Generation (SSG), giúp các trang web được lập chỉ mục dễ dàng hơn, cải thiện thứ hạng trên các công cụ tìm kiếm.

**Hiệu suất cao:** NextJS cho phép tạo ra các trang web tải nhanh và mượt mà nhờ các tính năng như SSG và SSR, mang lại trải nghiệm người dùng tốt hơn.

**Hệ thống Route tự động:** Tự động tạo các route dựa trên cấu trúc thư mục, giúp việc quản lý và phát triển ứng dụng trở nên dễ dàng hơn.

**Hỗ trợ CSS và JavaScript hiện đại:** Cho phép nhập tệp CSS trực tiếp trong tệp JavaScript và hỗ trợ các tính năng JavaScript hiện đại, giúp phát triển ứng dụng nhanh chóng và hiệu quả.

**Tích hợp API routes:** NextJS cho phép tạo các API routes để xử lý các yêu cầu HTTP, giúp xây dựng các API RESTful một cách dễ dàng.

#### Nhược điểm

**Giới hạn trong hệ thống định tuyến:** NextJS bị giới hạn việc chỉ sử dụng bộ định tuyến dựa trên cấu trúc tệp, không thể tùy chỉnh cách xử lý các route.

**Hệ sinh thái và thư viện còn hạn chế:** ít thư viện hơn so với các framework khác, có thể gây khó khăn khi tìm kiếm giải pháp cho các yêu cầu cụ thể.

### Cấu trúc của NextJS

Cấu trúc thư mục hiện tại của NextJS như sau:

my-nextjs-app/

|- .next/

|- node\_modules/

|- public/

|- src/

| |- app/

| |- components/

| |- styles/

| |- utils/

|- package.json

|- next.config.js

|- README.md

**Trong đó:**

**.next/:** Thư mục này được tạo ra sau khi build dự án, chứa các tệp liên quan đến quá trình biên dịch và tối ưu hóa.

**node\_modules/:** Chứa các module và thư viện mà dự án phụ thuộc, được quản lý bởi npm hoặc yarn.

**public/:** Lưu trữ các tệp tĩnh như hình ảnh, favicon, và các tài nguyên khác. Các tệp trong thư mục này có thể được truy cập trực tiếp thông qua đường dẫn URL.

**src/:** Thư mục gốc cho mã nguồn ứng dụng, giúp tổ chức code một cách rõ ràng và dễ quản lý.

**app/:** Chứa các thành phần ứng dụng và định tuyến. Trong NextJS, có thể tạo các trang bằng cách thêm các tệp vào thư mục này, NextJS sẽ tự động thiết lập các route dựa trên cấu trúc thư mục.

**components/:** Lưu trữ các component React dùng chung trong ứng dụng, giúp tái sử dụng và quản lý code hiệu quả.

**styles/:** Chứa các tệp CSS hoặc các giải pháp styling khác cho ứng dụng, giúp tách biệt phần giao diện và logic.

**utils/:** Lưu trữ các hàm tiện ích, helper functions, và logic chung được sử dụng trong nhiều phần của ứng dụng.

**package.json:** Tệp này chứa thông tin về dự án, bao gồm các gói phụ thuộc, script, và các cấu hình khác.

**next.config.js:** Tệp cấu hình cho NextJS, cho phép tùy chỉnh các thiết lập mặc định của framework theo nhu cầu dự án.

**README.md:** Tệp mô tả dự án, thường được sử dụng để cung cấp thông tin tổng quan, hướng dẫn cài đặt và sử dụng.

Ngoài ra, còn có thể tạo thêm cái thư mục khác tùy theo nhu cầu thực tế của từng dự án.

### Page Router và App Router trong NextJS

Trong Next.js, có hai hệ thống định tuyến chính: Pages Router và App Router.

Bảng . So sánh Page Router và App Router

|  |  |  |
| --- | --- | --- |
|  | **Page Router** | **App Router** |
| **Cấu trúc thư mục** | Sử dụng thư mục **pages/**, mỗi tệp tương ứng với một route. | Sử dụng thư mục **app/**, tạo các route lồng nhau thông qua cấu trúc thư mục. |
| **Thành phần** | Mặc định sử dụng thành phần phía client. | Mặc định sử dụng các thành phần phía server, giúp tối ưu hiệu suất. |
| **Gọi dữ liệu** | Sử dụng các hàm **getStaticProps**, **getServerSideProps,** **getInitialProps** để lấy dữ liệu. | Sử dụng hàm **fetch** trực tiếp trong các thành phần để lấy dữ liệu. |
| **Layouts** | Layouts tĩnh, không hỗ trợ lồng nhau một cách linh hoạt. | Hỗ trợ layouts động và có thể lồng nhau, cho phép cấu trúc giao diện phức tạp hơn. |
| **Server Components** | Không hỗ trợ. | Có hỗ trợ, tận dụng lợi thế của React Server Components. |

**Lựa chọn giữa Pages Router và App Router:**

**Dự án mới:** Nên sử dụng App Router để tận dụng các tính năng hiện đại và hiệu suất cao.

**Yêu cầu đơn giản:** Pages Router phù hợp với các dự án nhỏ, yêu cầu định tuyến đơn giản.

**Yêu cầu phức tạp:** App Router thích hợp cho các ứng dụng lớn với cấu trúc định tuyến và dữ liệu phức tạp.

# HUẤN LUYỆN MÔ HÌNH

## Thu thập và tiền xử lý dữ liệu

### Thu thập dữ liệu

Để phục vụ cho việc huấn luyện mô hình phân loại cảm xúc tiếng Việt, bước đầu tiên và quan trọng là tiến hành thu thập một tập dữ liệu thực tế, phản ánh đúng các biểu hiện cảm xúc của người dùng trong các tình huống giao tiếp. Trong đề tài này, dữ liệu được thu thập từ trang thương mại điện tử Tiki.vn, nơi khách hàng thường để lại các đánh giá và nhận xét về sản phẩm đã mua, một nguồn dữ liệu phong phú và mang tính cảm xúc rõ rệt, nhất là đối với ngôn ngữ tiếng Việt.

Quá trình thu thập dữ liệu được thực hiện bằng ngôn ngữ lập trình Python, với sự hỗ trợ của thư viện **requests** để gửi các yêu cầu HTTP đến API của trang Tiki và lấy dữ liệu trả về dưới dạng JSON. Dữ liệu sau đó được trích ra, chỉ lấy nội dung các bình luận, sau đó lưu trữ dưới dạng tệp .csv để thuận tiện cho việc xử lý sau này.

#### Các bước thực hiện

Quá trình thu thập dữ liệu được thực hiện theo các bước tuần tự, đảm bảo độ đầy đủ, phong phú và tính hiệu quả của tập dữ liệu. Cụ thể, các bước thực hiện được chia thành hai bước chính: thu thập danh sách sản phẩm từ các danh mục trên Tiki và thu thập các bình luận của người dùng từ từng sản phẩm đó. Dưới đây là mô tả chi tiết cho từng bước trong quy trình:

**Bước 1: Thu thập danh sách sản phẩm từ các danh mục**

Tiki tổ chức sản phẩm theo hệ thống phân cấp danh mục, với các danh mục lớn như "Sách", "Điện thoại - Máy tính bảng", "Thời trang", "Điện gia dụng",... và các danh mục con nằm bên trong. Việc lựa chọn đa dạng các danh mục giúp đảm bảo sự phong phú trong tập dữ liệu, bao phủ nhiều lĩnh vực, ngữ cảnh và hành vi tiêu dùng khác nhau.

Để truy vấn danh sách sản phẩm thuộc một danh mục cụ thể, Tiki có một API với phương thức GET, ta sử dụng một biến trong python để lưu lại để tiện sử dụng:

url\_category = "https://tiki.vn/api/personalish/v1/blocks/listings"

Khi gọi đến api này để lấy dữ liệu thì ta cần truyền vào các tham số sau:

category\_params = {

"limit": 40,

"include": "advertisement",

"aggregations": 2,

"version": "home-persionalized",

"category": 8322,

"page": 1,

"urlKey": "nha-sach-tiki"

}

Trong đó có các tham số quan trọng:

**limit:** là số lượng sản phẩm lấy trên mỗi trang.

**page:** là số thứ tự của trang hiện tại.

**category:** là id của danh mục.

Còn có một số tham số khác như: version, include, aggregations,…

Bên cạnh đó cần gửi kèm theo các headers quan trọng và là yêu cầu bắt buộc, bao gồm:

headers = {

"Content-Type": "application/json",

"Accept": "application/json, text/plain, \*/\*"

}

Vì ta cần thu thập dữ liệu từ nhiều danh mục khác nhau để tăng sự phong phú và đa dạng của dữ liệu, nên ta cần lưu id của các danh mục vào một danh sách để tiện truy vấn, thu thập id danh mục theo một cách thủ công từng danh mục dựa trên đường dẫn của tiki có dạng:

https://tiki.vn/url\_key/cxxx

Trong đó:

**url\_key:** là tham số urlKey trong API lấy sản phẩm từ danh mục ở trên đã giới thiệu.

**cxxx:** có nghĩa là category xxx và số xxx là id của danh mục đó.

Ví dụ danh mục “Nhà sách Tiki” sẽ có url\_key là “nha-sach-tiki” và id tương ứng là 8322 sẽ có dạng:

https://tiki.vn/nha-sach-tiki/c8322

Ta thu thập thủ công tuần tự qua các danh mục, sau đó lưu vào danh sách trong python với định dạng:

category\_ids = [

{"id": 8322, "url\_key": "sach"},

{"id": 1883, " url\_key": "nha-cua-doi-song"},

{"id": 1789, " url\_key": "dien-thoai-may-tinh-bang"},

{"id": 15078, " url\_key": "cham-soc-nha-cua"},

…

]

Vì đây là danh sách nên ta sử dụng vòng lặp để duyệt qua từng danh mục, sau đó sử dụng thư viện requests của python để gọi tới API, mục tiêu là lấy thông tin 40 sản phẩm trên trang đầu của từng danh mục:

for category in categories:

category\_params = {

"limit": 40, # lấy tối đa 40 sản phẩm

"include": "advertisement",

"aggregations": 2,

"version": "home-persionalized",

"category": category["id"], # id danh mục

"page": 1, # trang đầu tiền

"urlKey": category["url\_key"] # url\_key của danh mục

}

# sử dụng thư viện requests để gọi api với phương thức GET

response\_category = requests.get(url\_category, headers=headers, params=category\_params)

Sau khi gửi yêu cầu HTTP GET tới API, phản hồi trả về là dữ liệu JSON chứa thông tin của các sản phẩm, bao gồm: id, name, thumbnail\_url, rating\_average, review\_count,... Ở đây, quan trọng nhất là các Id sản phẩm và sẽ được lưu lại vào thành một danh sách để sử dụng ở bước tiếp theo:

for category in categories:

… # mã nguồn gọi API bên trên

if response\_category.status\_code == 200:

# trường data trong dữ liệu trả về là danh sách thông tin sản phẩm

product\_data = response\_category.json()["data"]

# Lấy danh sách product\_id

product\_ids = [item["id"] for item in product\_data]

**Bước 2: Thu thập bình luận từ các sản phẩm**

Sau khi thu được danh sách các product\_id, bước tiếp theo là thu thập các bình luận mà người dùng đã để lại cho từng sản phẩm. Mỗi sản phẩm trên Tiki thường có phần đánh giá bao gồm: số sao, nội dung nhận xét, thời gian đánh giá, và một số thông tin khác như hình ảnh, số lượt thích. Nhưng với phạm vi của đề tài là xử lý ngôn ngữ tự nhiên, nên ta chỉ quan tâm đến phần nội dung nhận xét.

Để truy vấn lấy danh sách các bình luận của một sản phẩm cụ thể, Tiki cung cấp một API với phương thức GET và ta sẽ lưu đường dẫn này vào một biến python để tiện sử dụng:

url\_reviews = "https://tiki.vn/api/v2/reviews"

Với các tham số cần truyền vào:

params = {

"limit": 20,

"page": 1,

"include": "comments",

"sort": "score|desc,id|desc,stars|all",

"product\_id": 252857829,

}

Trong đó:

**product\_id:** là id của sản phẩm.

**limit:** là số lượng bình luận mỗi lần truy vấn.

**page:** là số trang trong trường hợp sản phẩm có nhiều bình luận và được phân thành nhiều trang.

Tiếp tục sử dụng thư viện requests để gọi đến api, do ta gần gọi hết trang để lấy tất cả bình luận, nhưng ta không biết trước được tổng cộng có bao nhiêu trang, nên ta sử dụng vòng lập ***while True***, để lập qua các trang đến khi nào trang hiện tại là trang cuối (last\_page):

… # đoạn mã nguồn lấy id sản phẩm ở trên

for product\_id in product\_ids: # lập qua từng id sản phẩm

page = 1 # khởi tạo page đầu tiên

While True:

params = {

"limit": 20,

"page": page,

"include": "comments",

"sort": "score|desc,id|desc,stars|all",

"product\_id": product\_id,

}

response = requests.get(url\_reviews, headers=headers, params=params)

if response.status\_code == 200:

data = response.json()

Kết quả trả về các thông tin như ***paging, stars, rating\_average, data,…*** trong đó ***data*** là danh sách các bình luận:

reviews = data["data"]

for rw in reviews:

# lấy nội dung, đồng thời bỏ các ký tự xuống dòng

c = rw["content"].replace("\r","").replace("\n","")

# sau đó lưu vào tệp csv

with open(“data.csv", "a", newline="", encoding='utf-8-sig') as csvfile:

writer = csv.writer(csvfile)

writer.writerow([c])

# nếu đã đến trang cuối thì thoát vòng lập

if page == data["paging"]["last\_page"]:

break

# ngược lại đến trang tiếp theo

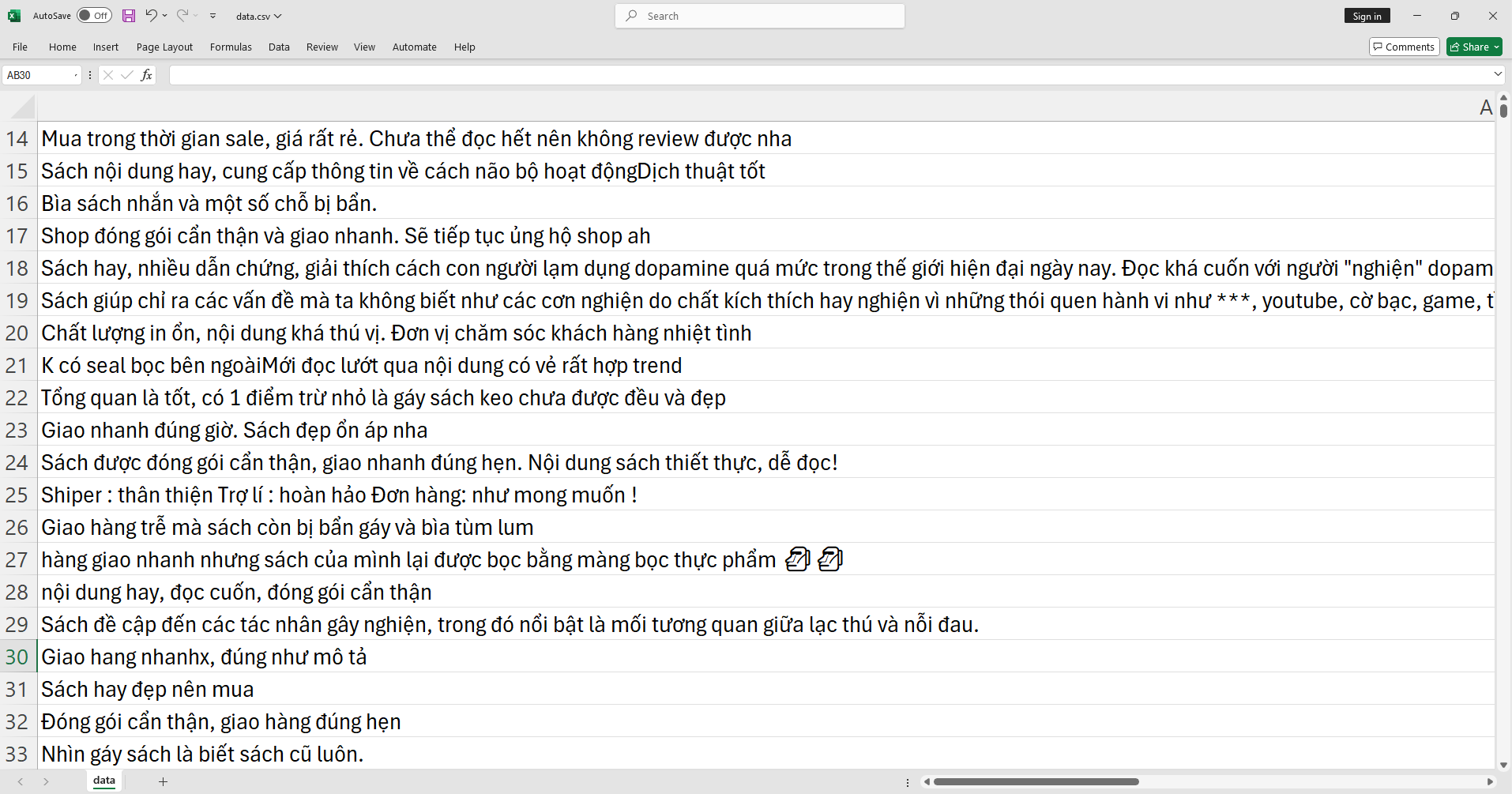
page += 1

else:

break

#### Kết quả thu thập

Sau khi thu thập ở các bước trên, ta được một tệp tin csv chứa dữ liệu nội dung các bình luận thô, các bình luận này chưa được làm sạch, còn chứa các ký tự đặc biệt, emoij, …



Hình . Dữ liệu thô sau khi thu thập

Số lượng bình luận thu thập với 21 danh mục lớn của Tiki và mỗi danh mục lấy 40 sản phẩm là 32.502 bình luận.

### Làm sạch dữ liệu

#### Mục tiêu

Dữ liệu đầu vào sau khi thu thập từ Tiki sẽ ở dạng thô, chứa nhiều yếu tố nhiễu như: Ký tự đặc biệt, dấu câu, không có nội dung, bình luận giống nhau về nội dung, lỗi mã hóa… Mục tiêu là làm sạch dữ liệu loại bỏ, xử lý các bình luận trên, nhằm đảm bảo chất lượng tập dữ liệu trước khi huấn luyện.

#### Các bước thực hiện

**Bước 1: Viết hàm Làm sạch dữ liệu cơ bản**

Sử dụng thư viện **re** để làm việc với biểu thức chính quy, phục vụ cho việc làm sạch chuỗi văn bản, ta viết một hàm xử lý trong python như sau:

def clean\_comment(comment: str) -> str:

# Bỏ ký tự đặc biệt, emoji, giữ lại chữ và số

comment = re.sub(r'[^\w\sÀ-ỹ]', '', comment, flags=re.UNICODE)

# Chuẩn hóa khoảng trắng

comment = re.sub(r'\s+', ' ', comment)

# trả kết quả về đồng thời chuyển thành chữ thường

return comment.strip().lower()

Hàm này dùng để làm sạch chuỗi văn bản đầu vào, tức là loại bỏ các ký tự không cần thiết như dấu câu, emoji, ký tự đặc biệt và chuẩn hóa văn bản về dạng chữ thường và sau đó trả kết quả về.

**Ví dụ sử dụng:**

clean\_comment(" Sản phẩm tốt lắm!!! 😍 ")

# Kết quả: "sản phẩm tốt lắm"

**Bước 2: Viết hàm kiểm tra tính hợp lệ**

Sau khi làm sạch, ta cần kiểm tra xem bình luận sau khi làm sạch có đủ điều kiện giữ lại để đưa vào mô hình hay không.

**Các tiêu chí kiểm tra:**

**Kiểm tra kiểu dữ liệu:** Nếu không phải chuỗi thì loại bỏ.

**Độ dài bình luận:** Nếu chuỗi rỗng hoặc chỉ có 1 ký tự thì loại bỏ. Điều này giúp loại bỏ các bình luận ngắn không đủ thông tin và không có ý nghĩa.

**Phát hiện ký tự lỗi mã hóa:** Nếu chuỗi chứa ký tự đặc biệt như ký tự Unicode lỗi thì loại bỏ.

def is\_valid\_comment(comment: str) -> bool:

if not isinstance(comment, str):

return False

if len(comment) <= 1:

return False

if re.search(r'[�]', comment): # Ký tự lỗi mã hóa

return False

return True

**Bước 3: Áp dụng các hàm vào làm sạch dữ liệu**

Sau khi có các hàm xử lý dữ liệu, ta sử dụng thư viện **pandas** của python để đọc dữ liệu đã thu thập và được lưu trong tệp tin ***data.csv*** để tiến hành xử lý:

import pandas as pd

df = pd.read\_csv("data.csv")

Sau đó ta từng bước xử lý dữ liệu qua các hàm đã có:

# B1: Làm sạch và chuẩn hóa bình luận

df['comment'] = df['comment'].astype(str).apply(clean\_comment)

Lúc này, bình luận sẽ được loại bỏ các ký tự đặc biệt, các khoảng trống dư thừa, bỏ các emoij và được chuyển tất cả về chữ thường. Tiếp theo, ta lược bỏ các bình luận không hợp lệ bằng cách sử dụng hàm **clean\_comment** đã viết:

# B2: Lược bỏ bình luận không hợp lệ

df = df[df['comment'].apply(is\_valid\_comment)]

Tiếp theo ta cần loại bỏ các bình luận trùng lặp, vì có rất nhiều bình luận giống nhau, nếu không được lược bỏ, sẽ gây nhiễu cho bộ dữ liệu, ta sử dụng hàm **drop\_duplicates** có sẵn trong thư viện pandas:

# B3: Loại bỏ bình luận trùng lặp

df = df.drop\_duplicates(subset='comment')

Sau khi qua những bước lược bỏ, sẽ có những dòng bị xóa đi, chỉ số dòng lúc này sẽ không còn liên tục, nên ta cần cài lại chỉ số dòng thông qua hàm **reset\_index** của thư viện pandas:

# B4: Reset chỉ số dòng

df = df.reset\_index(drop=True)

Cuối cùng, ta lưu lại dữ liệu đã được làm sạch vào tệp tin csv mới:

df[['comment']].to\_csv("data\_clean.csv", index=False)

Dữ liệu đã làm sạch lúc này còn 30 956 bình luận, tức là đã loại bỏ 1 546 bình luận so với 32 502 bình luận dữ liệu thô ban đầu thu thập được.

### Gán nhãn dữ liệu

#### Mục tiêu

Sau khi đã thu thập và làm sạch dữ liệu bình luận, việc gán nhãn là bước bắt buộc để biến dữ liệu thô thành dữ liệu huấn luyện có giám sát. Mỗi bình luận cần được gắn với một nhãn cảm xúc phù hợp, giúp mô hình học được mối liên hệ giữa đặc trưng văn bản và cảm xúc tương ứng.

Với mục tiêu của đề tài, việc phân loại cảm xúc được chia thành các cấp bậc cảm xúc gồm 3 nhóm chính:

**Tích cực (positive):** được gán nhãn là **1**, bình luận thể hiện sự hài lòng, đánh giá tốt về sản phẩm hoặc dịch vụ như “chất lượng tốt”, “hàng đẹp”, “giao nhanh”, “rất ưng ý”.

**Tiêu cực (negative):** được gán nhãn là **-1**, bình luận thể hiện sự không hài lòng, có thái độ phàn nàn, chê bai hoặc tức giận như “hàng bị lỗi”, “giao hàng trễ”, “sản phẩm không đúng mô tả”.

**Trung lập (neutral):** được gán nhãn là **0**, bình luận có nội dung khách quan, mang tính mô tả hoặc xác nhận giao dịch, không biểu hiện rõ cảm xúc như “đã nhận hàng”, “đúng mô tả”, cùng với các bình luận vừa có ý khen ở một gốc độ và vừa góp ý, chê bai ở một gốc độ khác của sản phẩm.

#### Quá trình gán nhãn

Quá trình gán nhãn được tiến hành thủ công, chia thành các bước cụ thể như sau:

**Bước 1:** **Đọc và đánh giá nội dung từng bình luận**

Được thực hiện bởi một nhóm ba người, mỗi người sẽ lần lượt đọc và đánh giá từng bình luận một cách thủ công, dựa vào nội dung văn bản để xác định sắc thái cảm xúc mà người dùng thể hiện trong bình luận đó.

Trong việc này, yêu cầu sự tập trung và hiểu biết ngôn ngữ, vì mỗi bình luận có thể có độ dài khác nhau, văn phong không đồng nhất và chứa các yếu tố ngữ nghĩa phức tạp. Việc đọc phải kết hợp giữa hiểu nghĩa đen và suy luận ý nghĩa ngữ cảnh, bởi một số bình luận sử dụng các hình thức diễn đạt ẩn dụ, nói giảm nói tránh hoặc hài hước châm biếm.

**Ví dụ:**

**“Shop gói hàng không kỹ lưỡng.”**: thể hiện cảm xúc tiêu cực, dù không dùng từ ngữ tiêu cực, chê bai trực tiếp.

**“Chất lượng tốt, giá hợp lý, 10 điểm.”**: thể hiện rõ cảm xúc tích cực.

**Bước 2: Xác định cảm xúc và gán nhãn**

Dựa trên nội dung bình luận, mỗi người gán nhãn sẽ xác định xem cảm xúc thể hiện trong văn bản thuộc loại nào trong ba nhóm: tích cực, tiêu cực hay trung lập:

**Tích cực:** Những bình luận biểu thị sự hài lòng, cảm xúc vui vẻ, đánh giá cao sản phẩm. Thường xuất hiện các từ khóa như: “tốt”, “đẹp”, “ưng ý”, “đáng tiền”, “hài lòng”, “tuyệt vời”,…

**Tiêu cực:** Là các phản hồi tiêu cực, chê bai hoặc bất mãn. Từ khóa thường gặp: “tệ”, “hỏng”, “không giống mô tả”, “giao hàng chậm”, “rất thất vọng”,…

**Trung lập:** Những bình luận mang tính mô tả, không có yếu tố cảm xúc mạnh, ví dụ như: “Đã nhận hàng”, “Chưa sử dụng”, “Sản phẩm đúng như hình”,… hoặc là các nhận xét có phần góp ý như: “sản phẩm có một vài chỗ chưa được tốt lắm mong shop cải thiện”.

Sau khi đã xác định được cảm xúc, thì người gán nhãn sẽ tiến hành ghi nhãn tương ứng cho từng bình luận vào cột **label** trong tệp tin data\_clean.csv đã được thu thập và làm sạch trước đó, với 1 là bình luận đó thuộc cảm xúc tích cực, -1 có nghĩa là bình luận tiêu cực, cuối cùng là 0 thể hiện là bình luận trung lập. Thao tác này được thực hiện trên công cụ **Microsoft Excel**, vì công cụ này có thể thực hiện xem và chỉnh sửa tệp tin csv một cách dễ dàng.

**Bước 3: Kiểm tra và thống nhất nhãn dữ liệu**

Sau khi quá trình gán nhãn ban đầu được thực hiện bởi ba người khác nhau, việc kiểm tra và thống nhất lại nhãn là một bước quan trọng nhằm đảm bảo chất lượng và độ tin cậy của tập dữ liệu huấn luyện. Trong thực tế, do cảm xúc là yếu tố mang tính chủ quan, mỗi người có thể có cách cảm nhận và đánh giá khác nhau về cùng một bình luận, dẫn đến các nhãn không đồng nhất. Do đó, cần có một cơ chế để phát hiện, so sánh và thống nhất các trường hợp không khớp nhãn.

Khi từng người hoàn thành phần gán nhãn của mình, các tệp dữ liệu được tổng hợp lại. Với mỗi bình luận, nhóm thực hiện đối chiếu các nhãn được gán từ các thành viên khác nhau. Những bình luận có nhãn trùng khớp ở tất cả thành viên thì được giữ nguyên, trong khi đó các bình luận có sự khác biệt sẽ được liệt kê vào một danh sách cần thống nhất lại.

Bảng . Đối chiếu nhãn dữ liệu của các thành viên đã gán

|  |  |  |  |
| --- | --- | --- | --- |
| ***Bình luận*** | ***Nhãn của thành viên 1*** | ***Nhãn của thành viên 2*** | ***Nhãn của thành viên 3*** |
| Chất lượng khá ổn, nhưng giá hơi cao. | 1 (Tích cực) | 0 (Trung lập) | 0 (Trung lập) |
| Sản phẩm vỡ khi nhận. | -1 (Tiêu cực) | -1 (Tiêu cực) | 0 (Trung lập) |

**Thảo luận thống nhất nhãn**

Đối với các bình luận có nhãn khác nhau, các thành viên sẽ thảo luận để đánh giá lại nội dung và tìm ra cảm xúc chủ đạo được thể hiện. Quy trình thực hiện theo các bước sau:

**Đọc kỹ nội dung bình luận:** Mọi người sẽ cùng đọc lại và đưa ra lập luận cho nhãn mình đã gán.

**Phân tích sắc thái cảm xúc chính:** Nếu bình luận có cả yếu tố tích cực và tiêu cực, nhóm sẽ thảo luận xem yếu tố nào chiếm ưu thế hoặc mục đích chính của bình luận là gì, xem xét xem có thể đưa vào bình luận trung lập hay không.

**Thống nhất theo quy tắc chung:**

Nếu bình luận có cả khen và chê nhưng chê rõ ràng hơn thì gán nhãn -1, nghĩa là tiêu cực và ngược lại gán nhãn 1 nghĩa là tích cực.

Nếu bình luận chỉ mô tả mà không kèm cảm xúc rõ ràng thì gán 0, nghĩa là trung lập.

Nếu không xác định rõ, ưu tiên đưa về nhãn 0 (trung lập) để giảm nhiễu cho mô hình.

Bảng . Thống nhất nhãn dữ liệu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Bình luận*** | ***Nhãn của thành viên 1*** | ***Nhãn của thành viên 2*** | ***Nhãn của thành viên 3*** | ***Nhãn thống nhất*** |
| Chất lượng khá ổn, nhưng giá hơi cao. | 1 (Tích cực) | 0 (Trung lập) | 0 (Trung lập) | 0 (Trung lập) |
| Sản phẩm vỡ khi nhận. | -1 (Tiêu cực) | -1 (Tiêu cực) | 0 (Trung lập) | -1 (Tiêu cực) |

Sau khi đạt được sự thống nhất, nhãn cuối cùng sẽ được ghi nhận vào tập dữ liệu chính thức. Việc thực hiện bước kiểm tra và thống nhất nhãn từ nhiều người khác nhau không chỉ giúp nâng cao chất lượng của tập dữ liệu, mà còn góp phần tăng độ tin cậy của mô hình khi huấn luyện. Dữ liệu đầu vào càng chính xác thì khả năng mô hình học được các đặc trưng cảm xúc đúng càng cao, từ đó cải thiện hiệu suất phân loại trên dữ liệu thực tế.

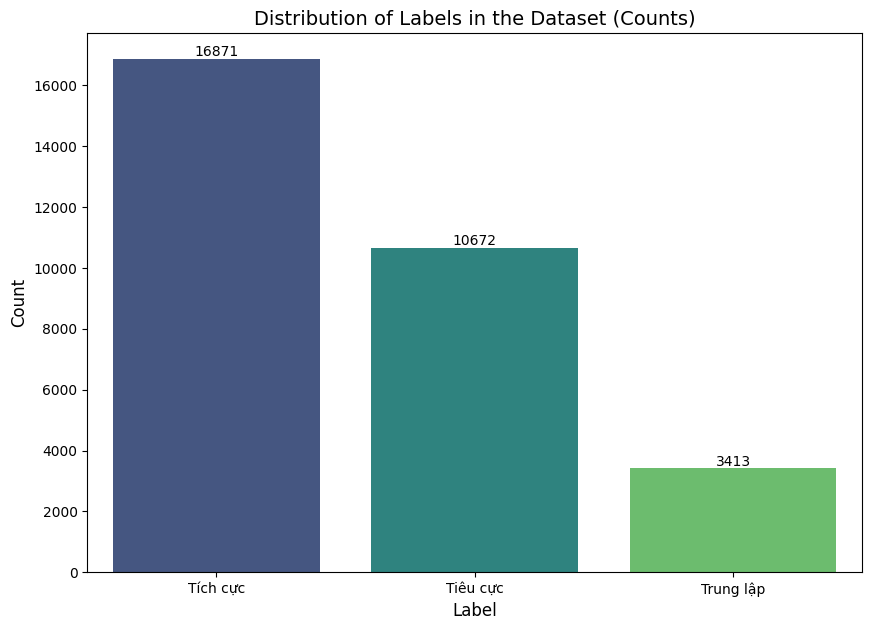
#### Kết quả sau khi gán nhãn

Sau khi rà soát và thống nhất nhãn, phân bố số lượng bình luận theo từng loại cảm xúc như sau:

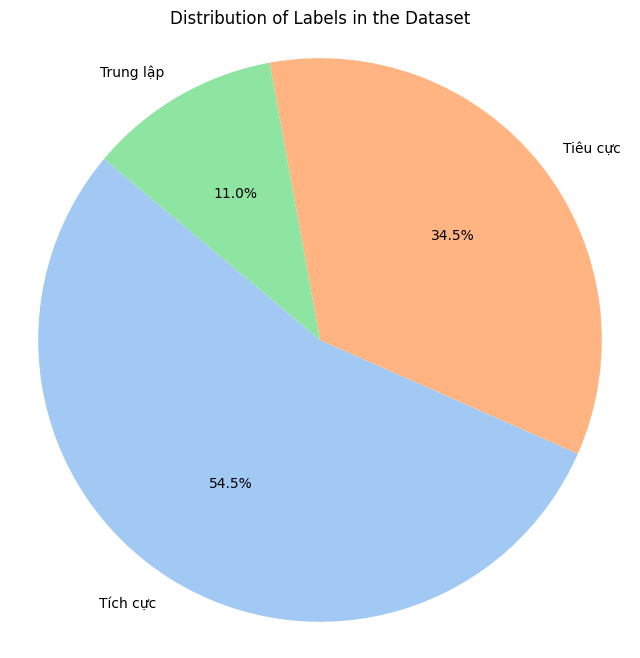
**Tích cực:** có 16 871 bình luận, chiếm 54.5% tổng số dữ liệu. Nhóm này chủ yếu gồm các bình luận thể hiện sự hài lòng về chất lượng sản phẩm, tốc độ giao hàng, giá cả hợp lý, hoặc dịch vụ hỗ trợ khách hàng tốt.

**Tiêu cực:** có 10 672 bình luận, chiếm 34.5%. Đây là các phản hồi thể hiện sự không hài lòng như sản phẩm bị lỗi, giao hàng trễ, đóng gói không cẩn thận, hoặc thái độ phục vụ chưa tốt.

**Trung lập:** có 3 413 bình luận, chiếm 11%. Nhóm này bao gồm các nhận xét mang tính mô tả, không thể hiện rõ cảm xúc, ví dụ như “Đã nhận hàng”, “Đúng như hình”, hoặc “Chưa sử dụng nên chưa biết thế nào”.



Biểu đồ . Phân bố số lượng của từng nhãn dữ liệu



Biểu đồ . Tỷ lệ phân bố của từng nhãn dữ liệu

Việc phân bố các nhãn trong tập dữ liệu được đánh giá chỉ tương đối và không đồng đều. Điều này phản ánh khá trung thực hành vi và cảm xúc thực tế của người tiêu dùng trên nền tảng thương mại điện tử. Tuy nhiên, cũng nhận thấy rằng xu hướng người dùng thường chỉ phản hồi khi họ cảm thấy quá hài lòng hoặc quá không hài lòng, khiến dữ liệu trung lập có phần ít hơn khá nhiều.

## Huấn luyện mô hình với Naïve Bayes

### Xử lý dữ liệu và chia tập dữ liệu

Dù dữ liệu đã được làm sạch ở các bước trước đó, nhưng để tránh lỗi trong quá trình huấn luyện mô hình thì ta cần một số bước xử lý trước khi đưa vào thuật toán với các mục tiêu: không có dữ liệu bị thiếu, chỉ chứa các dòng dữ liệu với nhãn hợp lệ, không có câu quá dài. Mã nguồn xử lý trong python:

**B1: Xóa dữ liệu bị thiếu**

df = df.dropna()

Hàm ***dropna*** sẽ loại bỏ hoàn toàn những dòng nào chứa ít nhất một cột bị thiếu giá trị. Đảm bảo mỗi dòng đều đầy đủ dữ liệu, ví dụ: Nếu dữ liệu ***comment*** bị thiếu thì sẽ không có gì để phân tích, còn nếu cột ***label*** bị thiếu thì không thể huấn luyện mô hình vì thiếu nhãn.

**B2: Chỉ giữ lại những dòng có nhãn hợp lệ**

df = df[df['label'].isin([-1, 0, 1])]

Hàm **isin** sẽ lọc ra những dòng mà cột ***label*** có giá trị nằm trong danh sách [-1, 0, 1], đây mới là những nhãn hợp lệ.

**B3: Loại bỏ bình luận quá dài**

df = df[df['comment'].astype(str).apply(len) <= 512]

Sẽ có một số bình luận quá dài gây ra lỗi trong quá trình huấn luyện, nên ta cần lọc lấy ra những bình luận có độ dài giới hạn không quá 512 ký tự.

**B4: Tách từ tiếng Việt**

Trong tiếng Việt, từ thường được viết liền nhau không có dấu phân cách giữa các từ đơn thuộc về cùng một cụm từ. Vì vậy, việc tách từ không chỉ là tách theo dấu cách mà cần xử lý ngữ nghĩa.

def tokenize(text):

return underthesea.word\_tokenize(text, format="text")

Hàm trên thực hiện tách từ tiếng Việt, dùng thư viện ***underthesea***, một thư viện xử lý ngôn ngữ tự nhiên tiếng Việt rất phổ biến. Hàm ***word\_tokenize*** có chức năng tách câu tiếng Việt thành các từ hoặc cụm từ có nghĩa, ví dụ: “sản phẩm đẹp quá” sẽ được tách thành các từ hoặc cụm từ “sản phẩm”, “đẹp”, “quá”.

Sao đó ta áp dụng hàm vào bộ dữ liệu với đoạn mã nguồn sau:

df['comment\_preprocessed'] = df['comment'].apply(tokenize)

**B5: Chia tập dữ liệu**

Chia tập dữ liệu là bước cực kỳ quan trọng trong quy trình huấn luyện mô hình, ta chia tập dữ liệu thành tập huấn luyện và tập kiểm tra:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df['text\_preprocessed'], df['label'], test\_size=0.2, stratify=df['label'])

Ta chia tập dữ liệu với test\_size=0.2, có nghĩa là 20% bộ dữ liệu chia cho tập kiểm thử và 80% dữ liệu còn lại chia cho tập huấn luyện. Hàm cũng đảm bảo giữ nguyên tỷ lệ nhãn giữa hai tập để đảm bảo mô hình không bị học lệch.

### Huấn luyện Naïve Bayes với BoW

Ở bước huấn luyện mô hình, ta sử dụng **pipeline**, một công cụ giúp xây dựng chuỗi các bước xử lý dữ liệu và huấn luyện mô hình một cách gọn gàng, tự động và dễ tái sử dụng. Với mã nguồn khởi tạo trong python như sau:

pipeline = Pipeline([

    ('vectorizer', CountVectorizer()),

    ('nb', MultinomialNB())

])

Sau đó, ta tiến hành đưa bộ dữ liệu vào mô hình để huấn luyện:

pipeline.fit(X\_train, y\_train)

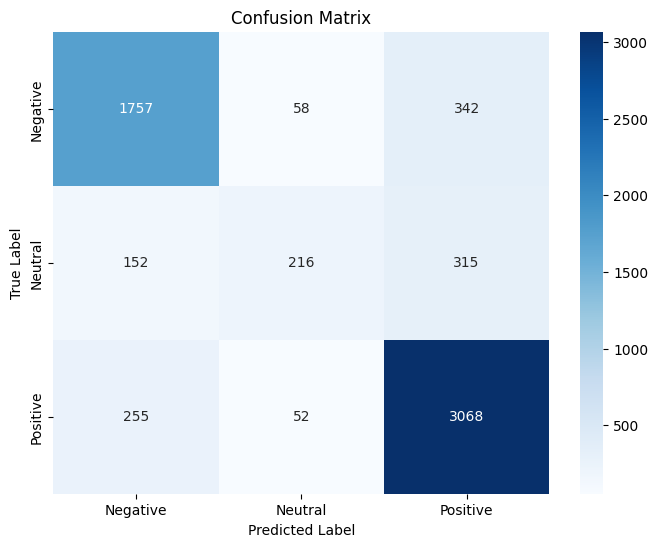
**Ta tiến hành đánh giá về mô hình sau khi huấn luyện**

Bảng . Đánh giá mô hình huấn luyện Naïve Bayes với trích xuất đặt trưng BoW

|  |  |  |  |
| --- | --- | --- | --- |
| ***Label*** | ***Precision*** | ***Recall*** | ***F1-score*** |
| **-1 (Tiêu cực)** | 0.81 | 0.81 | 0.81 |
| **0 (Trung lập)** | 0.66 | 0.32 | 0.43 |
| **1 (Tích cực)** | 0.82 | 0.91 | 0.86 |

Accuracy của mô hình là **81.11%**, một điểm số tương đối cao, cho thấy mô hình học được khá tốt, nhưng vì dữ liệu bị mất cân bằng, nên accuracy có thể không nói lên chính xác về độ hiệu quả của mô hình. Ta cần xét thêm các chỉ số khác.

Với lớp trung lập, điểm Recall là **0.32**, ta thấy mô hình chỉ phát hiện được 32% trong số các bình luận trung lập thật sự. Điều này là một vấn đề lớn và hiển nhiên xảy ra vì: Lớp trung lập quá ít mẫu dữ liệu; Đặc trưng của lớp không rõ ràng (câu đều có yếu tố tích cực và tiêu cực kết hợp với nhau nên mô hình dễ nhầm lẫn).



Biểu đồ . Ma trận nhầm lẫn khi huấn luyện mô hình Naïve Bayes với trích xuất đặt trưng BoW

Nhìn vào ma trận nhầm lẫn, ta dễ dàng thấy lớp trung lập rất dễ bị nhầm lẫn sang các lớp khác, cụ thể nhầm lẫn sang tiêu cực là 152, nhầm lẫn sang tích cực là 315 và dự đoán đúng là 216 trên tổng số 683, một số liệu rất đáng lo ngại và cần phải được khắc phục.

### Huấn luyện Naïve Bayes với TF-IDF

Tương tự, ta sử dụng **pipline** để xây dựng chuỗi các bước xử lý dữ liệu và huấn luyện mô hình:

pipeline = Pipeline([

    ('tfidf', TfidfVectorizer()),

    ('nb', MultinomialNB())

])

**Tiến hành huấn luyện**

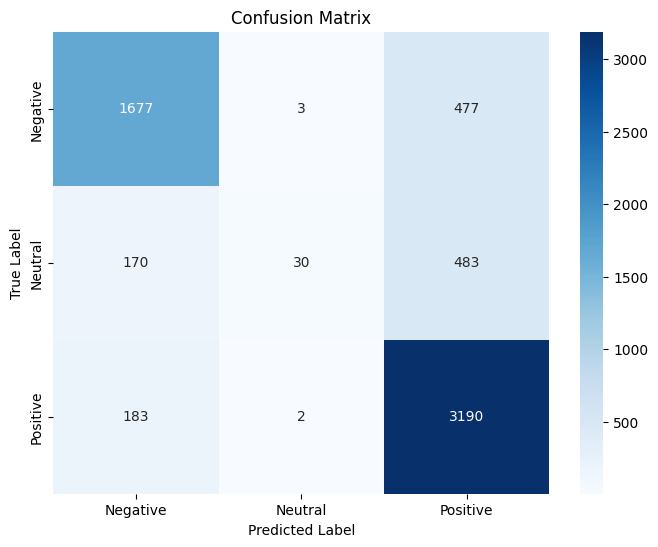
pipeline.fit(X\_train, y\_train)

**Đánh giá mô hình sau khi huấn luyện**

Bảng . Đánh giá mô hình huấn luyện Naïve Bayes với trích xuất đặt trưng TF-IDF

|  |  |  |  |
| --- | --- | --- | --- |
| ***Label*** | ***Precision*** | ***Recall*** | ***F1-score*** |
| **-1 (Tiêu cực)** | 0.83 | 0.78 | 0.80 |
| **0 (Trung lập)** | 0.86 | 0.04 | 0.08 |
| **1 (Tích cực)** | 0.77 | 0.95 | 0.85 |

Với accuracy **78.79%**, mô hình đạt hiệu quả thấp hơn khi huấn luyện với trích xuất đặc trưng BoW đáng kể. Lớp trung lập bị loại bỏ gần như hoàn toàn với điểm Recall chỉ 0.04 có nghĩa chỉ đúng 4% số mẫu trung lập thực tế, F1-score là 0.08 cực kỳ thấp, mô hình gần như bỏ qua hoặc nhầm toàn bộ lớp trung lập.



Biểu đồ . Ma trận nhầm lẫn khi huấn luyện mô hình Naïve Bayes với trích xuất đặt trưng TF-IDF

Dựa trên ma trận nhầm lẫn, lớp tiêu cực dự đoán đúng 1 677 trên tổng số 2 157 mẫu, độ chính xác khoảng 77.8%, giảm đáng kể so với trước là 90.6%. Mô hình dự đoán nhầm sang tích cực 477 mẫu, cao bất thường, mô hình mất khả năng phân biệt rõ ràng tiêu cực với tích cực, điều này rất đáng lo ngại.

Lớp trung lập chỉ dự đoán đúng 30 trên 683 mẫu là cực kỳ thấp, mô hình được xem như hoàn toàn mất khả năng nhận diện lớp trung lập.

## Huấn luyện mô hình với PhoBERT

Sau khi qua các bước xử lý dữ liệu như khi huấn luyện với Naïve Bayes, ta tiến hành các bước chuẩn bị để huấn luyện với bộ dữ liệu:

phobert\_tokenizer = AutoTokenizer.from\_pretrained("vinai/phobert-base", use\_fast=False)

AutoTokenizer là một lớp thuộc thư viện transformers, giúp tự động tải tokenizer phù hợp với mô hình đang sử dụng (ở đây là "vinai/phobert-base").

Ta định nghĩa hàm tokenize\_data để mã hóa dữ liệu theo định dạng của mô hình PhoBERT, để mô hình có thể hiểu được:

def tokenize\_data(example):

return phobert\_tokenizer(example['comment'], truncation=True,

padding='max\_length', max\_length=512)

Chuyển đổi DataFrame thành HuggingFace Dataset để tiện dùng cho training với transformers:

dataset = Dataset.from\_pandas(df)

Sau đó áp dụng Tokenizer cho toàn bộ tập dữ liệu:

dataset = dataset.map(tokenize\_data)

Tiếp theo ta cũng tiến hành chia tập dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ số lượng tập kiểm tra là 20%:

dataset = dataset.train\_test\_split(test\_size=0.2)

**Huấn luyện mô hình**

Trước tiên, ta tiến hành tải và nạp mô hình PhoBERT, với tham số num\_labels là 3, đây là số lượng lớp phân loại trong bộ dữ liệu, vì bộ dữ liệu ta có 3 lớp là tích cực, tiêu cực và trung lập:

model = AutoModelForSequenceClassification.from\_pretrained("vinai/phobert-base", num\_labels=3)

Sau đó ta tiến hành thiết lập cấu hình cho quá trình huấn luyện mô hình với lớp ***TrainingArguments***, đây là một lớp trong thư viện Hugging Face Transformers:

training\_args = TrainingArguments(

output\_dir="./phobert\_sentiment",

learning\_rate=2e-5,

per\_device\_train\_batch\_size=16,

per\_device\_eval\_batch\_size=16,

num\_train\_epochs=5,

weight\_decay=0.01,

save\_total\_limit=1,

logging\_steps=500,

)

Với ý nghĩa các tham số:

**output\_dir:** Thư mục lưu trữ kết quả huấn luyện, bao gồm: checkpoint mô hình, tệp tin cấu hình và các thông tin log. Ở đây, kết quả sẽ được lưu trong thư mục ./phobert\_sentiment.

**learning\_rate:** Tốc độ học xác định mức độ điều chỉnh trọng số mô hình sau mỗi bước lan truyền ngược. Ở đây, giá trị 2e-5 (0.00002) là khá nhỏ, phù hợp khi tinh chỉnh các mô hình ngôn ngữ lớn giống như đề tài đang làm để tránh làm mất kiến thức đã học sẵn.

**per\_device\_train\_batch\_size:** Số lượng mẫu huấn luyện xử lý trên mỗi bước huấn luyện.

**per\_device\_eval\_batch\_size:** Số lượng mẫu dùng trong mỗi lần ở giai đoạn đánh giá.

**num\_train\_epochs:** Số lần quét toàn bộ tập dữ liệu huấn luyện. Ở đây giá trị là 5, nghĩa là mô hình sẽ học qua toàn bộ dữ liệu 5 lần.

**weight\_decay:** Tham số điều chuẩn để tránh overfitting, bằng cách giảm dần trọng số mô hình trong quá trình huấn luyện. Giá trị 0.01 thường được dùng để giữ cân bằng giữa độ chính xác và khả năng tổng quát hóa.

**save\_total\_limit:** Giới hạn số lượng checkpoint lưu lại. Ở đây, chỉ giữ 1 checkpoint mới nhất.

**logging\_steps:** Khoảng cách số bước huấn luyện sau mỗi lần ghi log. Giá trị ở đây là 500, nghĩa là cứ sau 500 bước huấn luyện, hệ thống sẽ ghi lại thông tin tiến trình một lần.

Sau khi cấu hình xong, ta tiến hành huấn luyện mô hình:

trainer = Trainer(

model=model,

args=training\_args,

train\_dataset=dataset['train'],

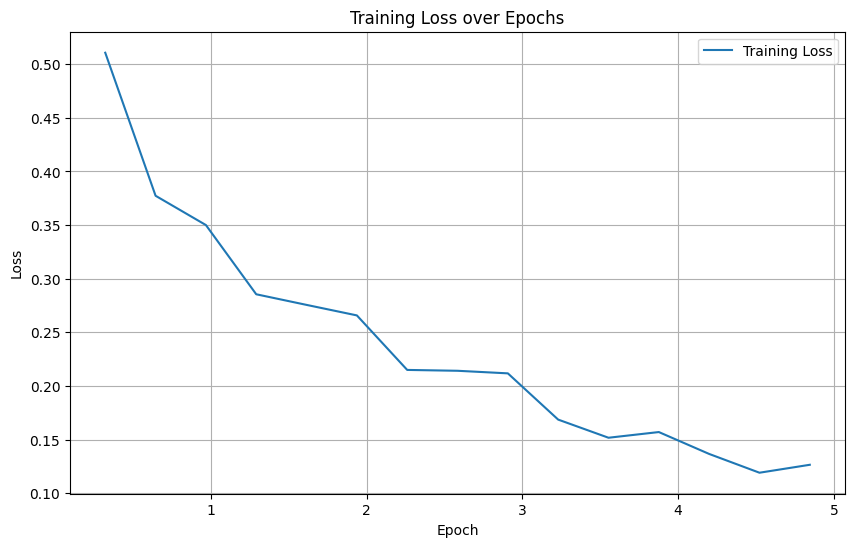
eval\_dataset=dataset['test'],

)

trainer.train()

Quá trình huấn luyện được thực hiện trên công cụ Google Colab, với phiên bản Google Colab Pro, với loại GPU A100.

Sau khi huấn luyện thành công, ta có biểu đồ Training Loss over Epochs như bên dưới:



Biểu đồ . Training Loss over Epochs khi huấn luyện với mô hình PhoBERT

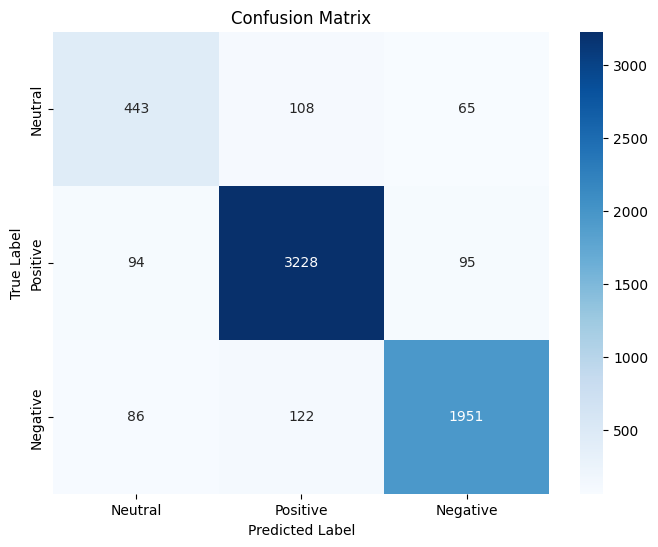
Qua biểu đồ cho thấy mô hình đang học tốt và ổn định. Loss giảm mạnh ở giai đoạn đầu, sau đó giảm chậm dần và hội tụ ở mức thấp, khoảng 0,12. Điều này chứng tỏ mô hình đã học được các đặc trưng quan trọng từ dữ liệu.

Bảng . Đánh giá mô hình huấn luyện với PhoBERT

|  |  |  |  |
| --- | --- | --- | --- |
| ***Label*** | ***Precision*** | ***Recall*** | ***F1-score*** |
| **-1 (Tiêu cực)** | 0.92 | 0.90 | 0.91 |
| **0 (Trung lập)** | 0.71 | 0.72 | 0.72 |
| **1 (Tích cực)** | 0.93 | 0.94 | 0.94 |

Kết quả đánh giá trên tập kiểm tra cho thấy mô hình đạt hiệu suất cao với hai lớp “Tích cực” và “Tiêu cực”. Đối với lớp “Trung lập”, các chỉ số Precision, Recall và F1-score thấp hơn đáng kể so với hai lớp còn lại. Điều này cho thấy mô hình gặp khó khăn khi phân biệt câu trung lập với câu tích cực hoặc tiêu cực. Tuy nhiên, so với khả năng phân loại lớp “Trung lập” của mô hình khi huấn luyện với Naïve Bayes trước đó là có cải thiện và hiệu quả hơn đáng kể.

Mô hình đạt Accuracy là 0.9079, tương đương với 90,79% dự đoán đúng tập kiểm thử.



Biểu đồ . Ma trận nhầm lẫn khi huấn luyện với mô hình PhoBERT

Biểu đồ ma trận nhầm lẫn cho thấy mô hình nhận diện tốt hai lớp “Tích cực” và “Tiêu cực” như đã nói trước đó, khi số lượng dự đoán đúng lần lượt đạt 3.228 và 1.951, trong khi số lượng dự đoán nhầm sang các lớp khác tương đối thấp. Đối với lớp “Trung lập”, số mẫu dự đoán đúng là 443, nhưng vẫn tồn tại tỷ lệ nhầm lẫn đáng kể, đặc biệt là sang lớp “Tích cực” với 108 mẫu.

## So sánh kết quả giữa các phương pháp

So sánh kết quả dự đoán giữa các phương pháp thông qua các chỉ số đánh giá gồm Precision, Recall, F1-score và Accuracy. Đây là bốn chỉ số quan trọng phản ánh khả năng phân loại chính xác, khả năng bao phủ đầy đủ dữ liệu, sự cân bằng giữa độ chính xác và độ bao phủ, cũng như tỷ lệ dự đoán đúng tổng thể của mô hình.

Bảng . So sánh kết quả dự đoán giữa các mô hình

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mô hình \ Macro avg | ***Precision*** | ***Recall*** | ***F1-score*** | ***Accuracy*** |
| Naïve Bayes + IF-IDF | 0.82 | 0.59 | 0.58 | 0.79 |
| Naïve Bayes + BoW | 0.77 | 0.68 | 0.71 | 0.81 |
| PhoBERT | 0.86 | 0.86 | 0.86 | 0.91 |

Nhìn vào kết quả tổng thể, có thể nhận thấy rằng PhoBERT vượt trội hơn hẳn hai mô hình Naïve Bayes về tất cả các chỉ số. Cụ thể, PhoBERT đạt Precision và Recall đều ở mức 0.86, giúp F1-score cũng đạt 0.86, và Accuracy lên tới 0.91. Điều này cho thấy PhoBERT không chỉ dự đoán đúng với độ chính xác cao mà còn duy trì khả năng nhận diện đầy đủ tất cả các nhãn cảm xúc, đảm bảo tính cân bằng giữa các lớp dữ liệu. Đây là đặc điểm nổi bật của các mô hình ngôn ngữ tiền huấn luyện sâu, vốn đã được học trên kho dữ liệu lớn và hiểu rõ ngữ cảnh tiếng Việt.

Trong khi đó, hai biến thể của Naïve Bayes cho kết quả thấp hơn khá nhiều. Mô hình Naïve Bayes + TF-IDF có Precision tương đối cao (0.82) nhưng Recall thấp (0.59), dẫn đến F1-score chỉ đạt 0.58 và Accuracy ở mức 0.79. Điều này phản ánh rằng mặc dù mô hình này dự đoán đúng khá nhiều trường hợp khi đã nhận diện được, nhưng lại bỏ sót nhiều mẫu thuộc các lớp đúng, làm giảm hiệu quả tổng thể.

Mô hình Naïve Bayes + BoW có sự cân bằng hơn giữa Precision (0.77) và Recall (0.68), giúp F1-score tăng lên 0.71 và Accuracy đạt 0.81. Điều này cho thấy BoW cải thiện được khả năng bao phủ dữ liệu so với TF-IDF, mặc dù độ chính xác (Precision) giảm nhẹ.

Tổng thể, kết quả cho thấy PhoBERT là mô hình có hiệu suất vượt trội nhất trong ba phương pháp, đặc biệt khi yêu cầu bài toán đòi hỏi sự cân bằng giữa độ chính xác và khả năng bao phủ các lớp dữ liệu. Trong khi đó, Naïve Bayes (với TF-IDF hoặc BoW) vẫn có thể được sử dụng cho các ứng dụng đơn giản, yêu cầu ít tài nguyên tính toán, nhưng khó đạt được chất lượng cao như các mô hình ngôn ngữ sâu.

# XÂY DỰNG ỨNG DỤNG THỬ NGHIỆM

## Mục tiêu xây dựng ứng dụng thử nghiệm

Sau khi hoàn thành quá trình huấn luyện mô hình cho bài toán, bước quan trọng tiếp theo là triển khai mô hình thử nghiệm dưới dạng dịch vụ API và xây dựng giao diện người dùng. Mục tiêu của bước này là đưa mô hình từ môi trường nghiên cứu sang ứng dụng thử nghiệm thực tế, cho phép người dùng dễ dàng nhập dữ liệu văn bản và nhận kết quả phân tích cảm xúc. Việc triển khai theo kiến trúc frontend – backend tách biệt giúp hệ thống linh hoạt, dễ mở rộng và dễ dàng tích hợp với các nền tảng khác.

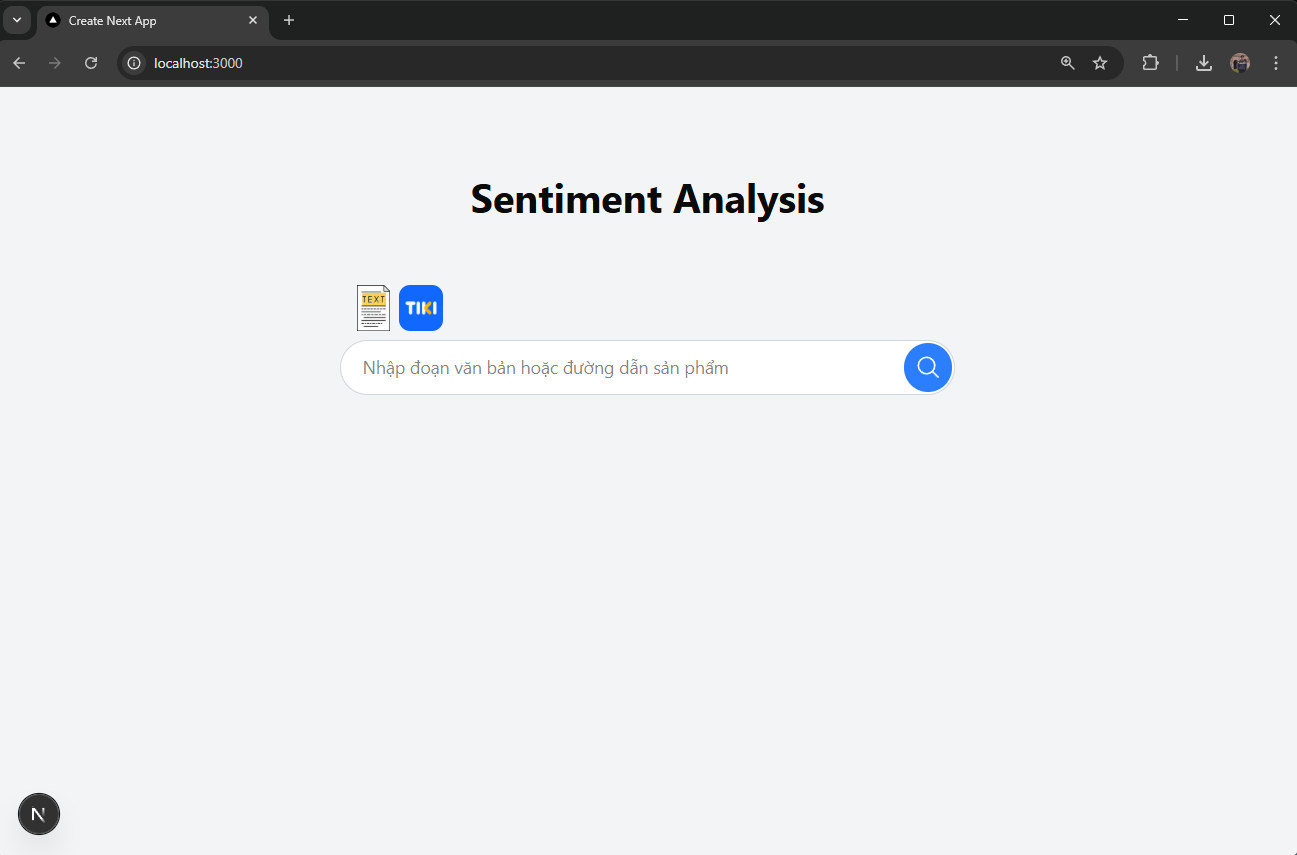
Ở phía backend, mô hình PhoBERT được đóng gói và triển khai thành API sử dụng FastAPI. Đây là framework Python hiện đại, tối ưu hiệu năng, hỗ trợ lập trình bất đồng bộ, cho phép xử lý lượng lớn yêu cầu đồng thời với độ trễ thấp, tích hợp sẵn thư viện tự động sinh tài liệu API dưới dạng Swagger UI giúp kiểm thử nhanh chóng.

Ở phía frontend, hệ thống sử dụng NextJS để xây dựng giao diện web. Next.js là framework React mạnh mẽ, cải thiện tốc độ tải trang và tối ưu SEO. Giao diện được thiết kế đơn giản, trực quan, cho phép người dùng nhập đoạn văn hoặc đường liên kết sản phẩm, gửi yêu cầu đến API FastAPI, và hiển thị kết quả phân loại cảm xúc (tích cực, trung lập, tiêu cực) cùng với xác suất dự đoán.

Việc kết hợp FastAPI cho backend và NextJS cho frontend mang lại một hệ thống vừa mạnh mẽ ở khả năng xử lý dữ liệu, vừa thân thiện với người dùng, đáp ứng tốt yêu cầu của một ứng dụng thử nghiệm nhận diện cảm xúc tiếng Việt trong khuôn khổ đề tài.

## Kết quả sau khi xây dựng ứng dụng thử nghiệm

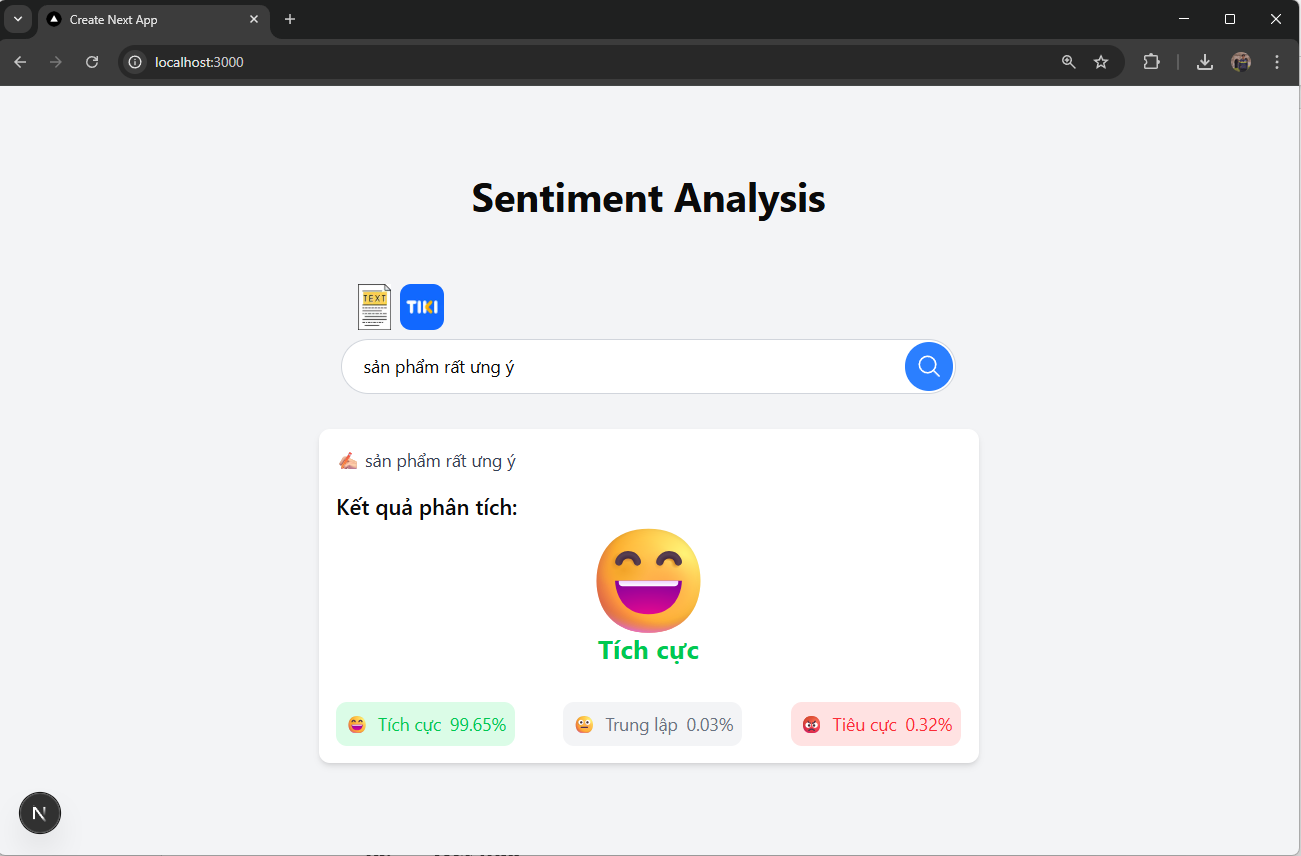
Tại giao diện chính của hệ thống phân tích cảm xúc, người dùng có thể nhập một đoạn văn bản hoặc dán liên kết đến trang sản phẩm trên Tiki để tiến hành phân tích cảm xúc. Với thiết kế đơn giản, tập trung vào trải nghiệm người dùng, với thanh nhập liệu đặt ở trung tâm màn hình và biểu tượng tìm kiếm giúp thao tác trở nên trực quan hơn.



Hình . Giao diện chính nhập văn bản dự đoán

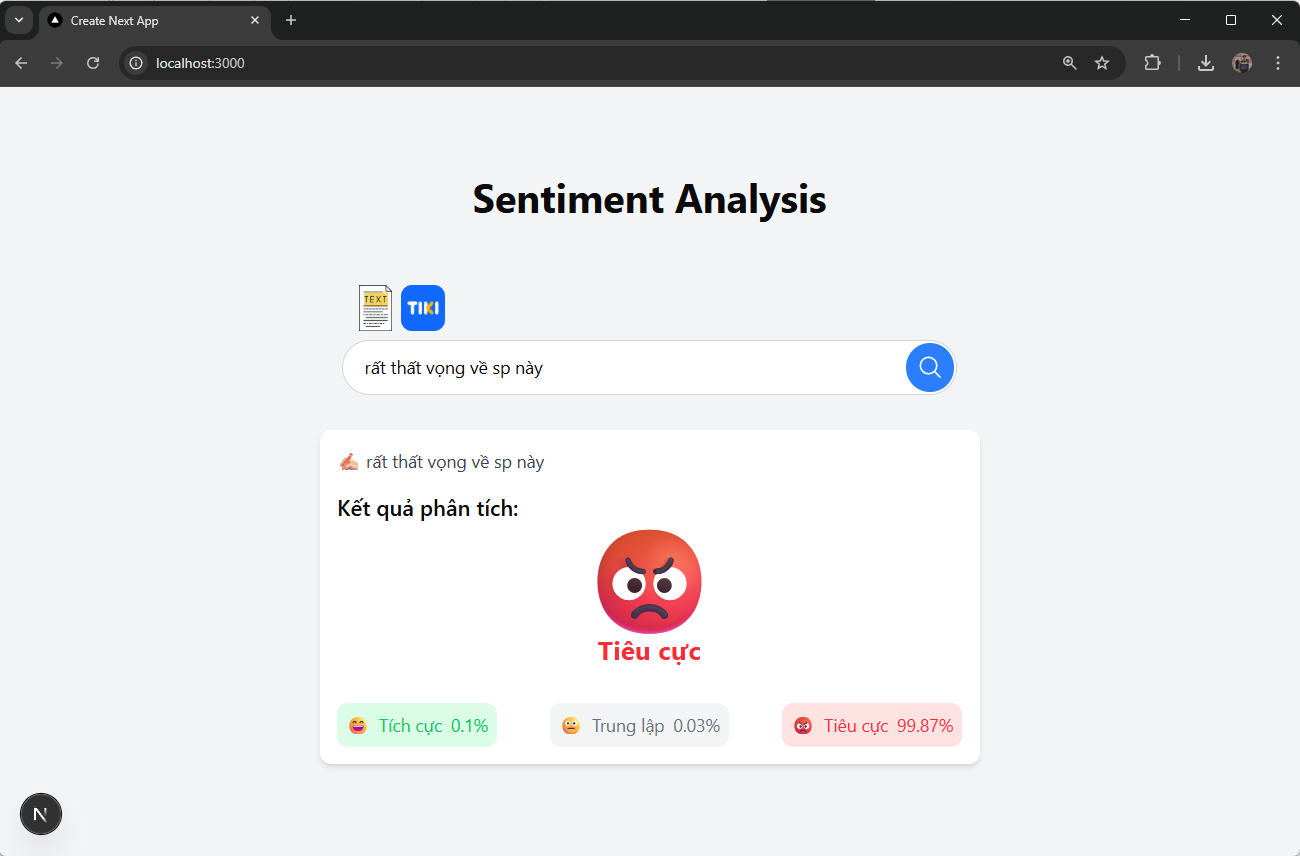
Nếu người dùng nhập trực tiếp một đoạn văn bản, hệ thống sẽ gửi dữ liệu này đến FastAPI, nơi mô hình PhoBERT đã được huấn luyện sẽ xử lý và trả về kết quả phân loại cảm xúc ngay lập tức. Ngược lại, nếu người dùng nhập vào một liên kết sản phẩm từ Tiki, ứng dụng sẽ tự động thu thập danh sách các bình luận của sản phẩm đó. Sau khi dữ liệu bình luận được tải về, hệ thống sẽ lần lượt dự đoán cảm xúc của từng bình luận, sau đó tổng hợp và hiển thị kết quả phân tích dưới dạng thống kê trực quan.

Thiết kế này cho phép người dùng linh hoạt lựa chọn cách thức nhập dữ liệu, đồng thời tận dụng tối đa khả năng của mô hình PhoBERT để phân tích cảm xúc cả ở mức văn bản đơn lẻ lẫn tập hợp nhiều bình luận. Điều này giúp hệ thống trở nên hữu ích hơn trong các tình huống đánh giá chất lượng sản phẩm, theo dõi phản hồi khách hàng và nghiên cứu dữ liệu từ mạng xã hội, thương mại điện tử.



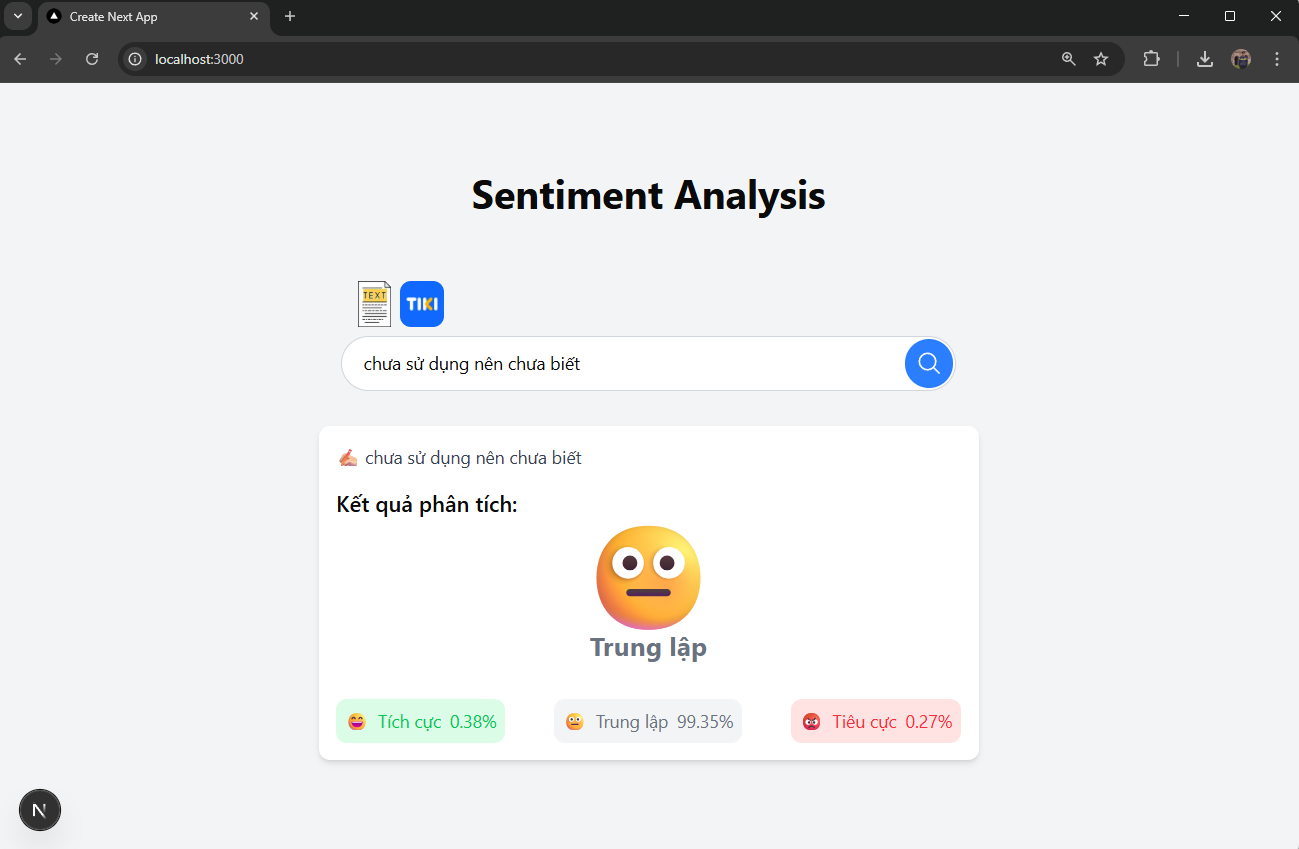
Hình . Giao diện khi dự đoán là tích cực

Giao diện của ứng dụng phân tích cảm xúc khi hệ thống dự đoán kết quả là tích cực. Ví dụ như trên, người dùng nhập câu “sản phẩm rất ưng ý” vào ô tìm kiếm, sau đó hệ thống gửi dữ liệu đến backend nơi mô hình PhoBERT đã được huấn luyện sẽ xử lý và trả về kết quả phân loại. Giao diện hiển thị biểu tượng cảm xúc tươi cười kèm nhãn “Tích cực” màu xanh lá, giúp người dùng dễ dàng nhận biết kết quả phân tích. Đồng thời, hệ thống cũng cung cấp tỉ lệ xác suất cho từng nhãn cảm xúc.



Hình . Giao diện khi dự đoán là tiêu cực

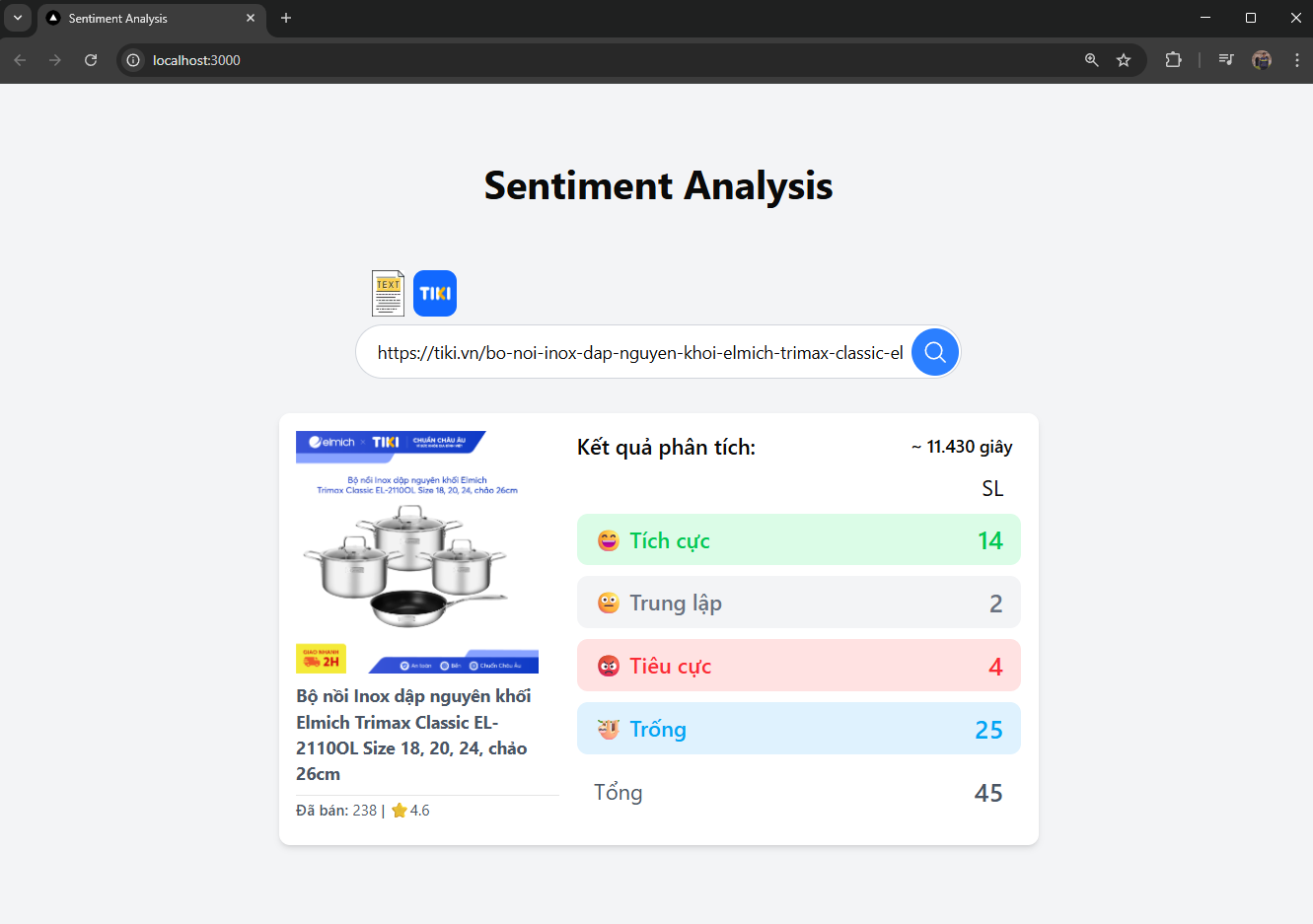
Giao diện ứng dụng khi hệ thống dự đoán kết quả phân tích cảm xúc là tiêu cực. Ví dụ này, người dùng nhập câu “rất thất vọng về sp này” vào ô nhập liệu. Tương tự như trên, dữ liệu sau đó được gửi đến API FastAPI, nơi mô hình PhoBERT đã huấn luyện xử lý và trả về kết quả dự đoán. Giao diện hiển thị biểu tượng khuôn mặt giận dữ kèm nhãn “Tiêu cực” màu đỏ nổi bật, giúp người dùng nhanh chóng nhận biết cảm xúc được phân loại



Hình . Giao diện khi dự đoán là trung lập

Giao diện ứng dụng khi hệ thống dự đoán kết quả phân tích cảm xúc là trung lập. Trong ví dụ này, người dùng nhập câu “chưa sử dụng nên chưa biết” vào ô tìm kiếm. Về luồng xử lý cũng tương tự như các ví dụ trên, sau đó giao diện hiển thị biểu tượng khuôn mặt trung tính cùng nhãn “Trung lập” với màu vàng nhạt cùng với tỷ lệ xác suất dự đoán của từng nhãn ở phía dưới.

Cuối cùng là giao diện khi hệ thống dự đoán kết quả phân tích cảm xúc của các bình luận của một sản phẩm trên sàn thương mại điện tử Tiki. Người dùng chỉ cần nhập liên kết của sản phẩm vào ô tìm kiếm, hệ thống sẽ tự động tải dữ liệu về sản phẩm đó cùng với các bình luận, sau đó lần lượt gửi các bình luận về backend để xử lý. Khi xử lý xong, giao diện sẽ hiện kết quả tổng quan về số lượng các nhãn đã phân tích được, cùng với tổng số lượng các bình luận và tổng thời gian phân tích như hình bên dưới:



Hình . Giao diện khi dự đoán một sản phẩm của tiki

Việc hiển thị đầy đủ các xác suất dự đoán mang lại sự minh bạch, hỗ trợ người dùng đánh giá mức độ tin cậy của kết quả. Thiết kế giao diện trực quan, sử dụng màu sắc và biểu tượng cảm xúc rõ ràng cho từng loại cảm xúc, giúp trải nghiệm trở nên sinh động và dễ hiểu.

# KẾT LUẬN

## Về mặt nghiên cứu

Đề tài “Nhận diện cảm xúc từ văn bản tiếng Việt bằng mô hình NLP” đã tập trung nghiên cứu và huấn luyện thành công mô hình có khả năng phân loại cảm xúc trong văn bản tiếng Việt, với ba nhóm cảm xúc chính: tích cực, tiêu cực và trung lập. Dữ liệu đầu vào được lấy từ các bình luận thực tế trên các trang thương mại điện tử, phản ánh tính ứng dụng và độ phức tạp cao trong ngôn ngữ tự nhiên.

**Về nghiên cứu, đề tài đã đạt được một số kết quả đáng chú ý:**

Tìm hiểu và áp dụng hiệu quả các kỹ thuật tiền xử lý văn bản tiếng Việt như: chuẩn hóa, loại bỏ từ, tách từ, và mã hóa đặc trưng bằng TF-IDF và BoW.

So sánh và đánh giá các mô hình học máy như Naive Bayes và mô hình ngôn ngữ hiện đại như PhoBERT, cho thấy tiềm năng vượt trội của các mô hình pretrained khi xử lý ngôn ngữ tiếng Việt.

Phân tích hiệu suất mô hình dựa trên các chỉ số đánh giá phổ biến như Accuracy, Precision, Recall và F1-score, từ đó rút ra nhận định khách quan về độ chính xác cũng như điểm mạnh, điểm yếu của từng phương pháp.

Nhìn chung, kết quả nghiên cứu cho thấy khả năng áp dụng của các mô hình NLP hiện đại vào bài toán phân tích cảm xúc tiếng Việt là hoàn toàn khả thi, thậm chí đạt độ chính xác cao trong một số trường hợp. Tuy nhiên, vẫn còn một số hạn chế cần khắc phục như: sự mất cân bằng trong phân bố nhãn dữ liệu, độ đa dạng ngữ nghĩa hay cấu trúc phức tạp của tiếng Việt.

## Về mặt ứng dụng

Đề tài cho thấy tiềm năng ứng dụng rộng rãi trong thực tiễn, đặc biệt là trong bối cảnh số hóa và sự phát triển mạnh mẽ của các nền tảng trực tuyến tại Việt Nam. Việc tự động phân tích cảm xúc người dùng từ các bình luận, đánh giá trên các trang thương mại điện tử có thể mang lại nhiều giá trị thực tiễn, **cụ thể như**:

Hỗ trợ doanh nghiệp trong việc tổng hợp và đánh giá ý kiến khách hàng một cách nhanh chóng, từ đó cải thiện sản phẩm, dịch vụ phù hợp hơn với nhu cầu thị trường.

Tăng cường trải nghiệm người dùng bằng cách phát hiện và xử lý các đánh giá tiêu cực hoặc phản hồi bất thường một cách chủ động.

Ứng dụng trong quản trị thương hiệu, truyền thông và chăm sóc khách hàng tự động, nhờ vào khả năng phát hiện nhanh các xu hướng cảm xúc trong cộng đồng.

Tích hợp vào hệ thống đánh giá nội dung mạng xã hội, diễn đàn hoặc ứng dụng thương mại điện tử để hỗ trợ phân loại, kiểm duyệt nội dung và đề xuất thông minh.

Tuy vẫn còn một số giới hạn như sự phức tạp trong ngôn ngữ tiếng Việt và tính chủ quan trong cảm xúc, nhưng kết quả đạt được từ khóa luận đã cho thấy tiềm năng lớn của các mô hình NLP trong việc khai thác thông tin cảm xúc từ văn bản tiếng Việt. Điều này mở ra nhiều hướng phát triển và ứng dụng trong kỷ nguyên dữ liệu và trí tuệ nhân tạo hiện nay.

## Về mặt học tập

Quá trình thực hiện đề không chỉ giúp tiếp cận và áp dụng kiến thức chuyên ngành vào một bài toán thực tiễn, mà còn mang lại nhiều giá trị về mặt học tập và rèn luyện kỹ năng cá nhân.

**Thông qua đề tài này, có cơ hội:**

Củng cố và mở rộng kiến thức chuyên môn về xử lý ngôn ngữ tự nhiên, học máy và các kỹ thuật phân tích dữ liệu văn bản.

Tiếp cận và sử dụng thành thạo các thư viện, công cụ hiện đại như Scikit-learn, Transformers, cùng với các mô hình ngôn ngữ tiên tiến như PhoBERT.

Hiểu rõ quy trình xây dựng một hệ thống phân tích cảm xúc từ khâu thu thập dữ liệu, tiền xử lý, huấn luyện mô hình, đánh giá kết quả cho đến bước ứng dụng thực tế.

Rèn luyện kỹ năng tự học, tự nghiên cứu tài liệu chuyên ngành, một yếu tố quan trọng trong lĩnh vực công nghệ thông tin hiện nay cùng với đó giúp nâng cao kỹ năng lập trình, tư duy giải quyết vấn đề và phân tích dữ liệu.

TÀI LIỆU THAM KHẢO

PHỤ LỤC