

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN KỸ THUẬT DỮ LIỆU



PHẠM HỮU DŨNG – 21133022

HOÀNG MẠNH ĐỨC – 21133027

NGUYỄN PHƯƠNG KHOA – 21133048

Đề tài:

**XÂY DỰNG HỆ THỐNG PHÂN TÍCH CẢM XÚC KHÁCH HÀNG BẰNG XỬ
LÝ NGÔN NGỮ TỰ NHIÊN ĐỂ GỢI Ý SẢN PHẨM**

KHÓA LUẬN TỐT NGHIỆP

GIẢNG VIÊN HƯỚNG DẪN
PGS.TS HOÀNG VĂN DŨNG

KHÓA 2021 – 2025

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên 1: **Nguyễn Phương Khoa**

MSSV 1: **21133048**

Họ và tên Sinh viên 2: **Hoàng Mạnh Đức**

MSSV 2: **21133027**

Họ và tên Sinh viên 3: **Phạm Hữu Dũng**

MSSV 3: **21133022**

Chuyên ngành: **Kỹ thuật dữ liệu**

Tên đề tài: **Xây dựng hệ thống phân tích cảm xúc khách hàng bằng xử lý ngôn ngữ tự nhiên để gợi ý sản phẩm**

Họ và tên giáo viên hướng dẫn: **PGS.TS. Hoàng Văn Dũng**

NHẬN XÉT

1. Về nội dung đề tài và khối lượng thực hiện:
2. Ưu điểm:
3. Khuyết điểm:
4. Đề nghị cho bảo vệ hay không?
5. Đánh giá loại:
6. Điểm:

TP. Hồ Chí Minh, tháng.... năm 2025

Giảng viên hướng dẫn

(Ký & ghi rõ họ tên)

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên 1: **Nguyễn Phương Khoa**

MSSV 1: **21133048**

Họ và tên Sinh viên 2: **Hoàng Mạnh Đức**

MSSV 2: **21133027**

Họ và tên Sinh viên 3: **Phạm Hữu Dũng**

MSSV 3: **21133022**

Chuyên ngành: **Kỹ thuật dữ liệu**

Tên đề tài: **Xây dựng hệ thống phân tích cảm xúc khách hàng bằng xử lý ngôn ngữ tự nhiên để gợi ý sản phẩm**

Họ và tên giáo viên phản biện:

NHẬN XÉT

1. Về nội dung đề tài và khối lượng thực hiện:
2. Ưu điểm:
3. Khuyết điểm:
4. Đề nghị cho bảo vệ hay không?
5. Đánh giá loại:
6. Điểm:

TP. Hồ Chí Minh, tháng ... năm 2025

Giảng viên phản biện

(Ký & ghi rõ họ tên)

Trường ĐH Sư Phạm Kỹ Thuật TP.HCM

Khoa: Công Nghệ Thông Tin

ĐỀ CƯƠNG KHÓA LUẬN TỐT NGHIỆP

Họ và tên Sinh viên 1: **Nguyễn Phương Khoa**

MSSV 1: **21133048**

Họ và tên Sinh viên 2: **Hoàng Mạnh Đức**

MSSV 2: **21133027**

Họ và tên Sinh viên 3: **Phạm Hữu Dũng**

MSSV 3: **21133022**

Thời gian làm khóa luận: **Từ ngày ../../2025**

Đến: ../.. / 2025

Chuyên ngành: **Kỹ thuật dữ liệu**

Tên đề tài: **Xây dựng hệ thống phân tích cảm xúc khách hàng bằng xử lý ngôn ngữ tự nhiên để gợi ý sản phẩm**

Họ và tên giáo viên hướng dẫn: **PGS.TS. Hoàng Văn Dũng**

Nhiệm vụ của luận văn:

1. Lý thuyết:

2. Thực Hành:

LỜI CẢM ƠN

Đầu tiên, nhóm em xin chân thành gửi lời cảm ơn với thầy Hoàng Văn Dũng - Giảng viên phụ trách hướng dẫn khóa luận tốt nghiệp – Trường Đại học Sư phạm kỹ thuật TP.HCM.

Trong thời gian làm khóa luận, nhóm em nhận được sự nhiều sự giúp đỡ từ Thầy. Thầy đã cung cấp cho chúng em đầy đủ kiến thức, chỉ bảo và đóng góp những ý kiến quý báu giúp chúng em có thêm hiểu biết để hoàn thành được báo cáo tiểu luận chuyên ngành. Thầy đã cung cấp tài liệu và hướng dẫn tận tình cho chúng em trong suốt quá trình nghiên cứu. Thầy luôn nhiệt tình vui vẻ chỉ dạy, chia sẻ giúp chúng em quá trình thực hiện bài báo cáo. Chúng em rất trân quý sự tâm huyết và trách nhiệm của Thầy trong công việc giảng dạy và truyền đạt kiến thức.

Trong quá trình thực hiện khóa luận tốt nghiệp, dựa trên kiến thức được Thầy cung cấp qua các thảo luận, kết hợp với việc tự tìm hiểu những công cụ và kiến thức mới, nhóm đã cố gắng thực hiện tiểu luận một cách tốt nhất. Tuy nhiên, bài khóa luận còn chưa được hoàn thiện và có thể còn có sai sót.

Nhóm rất mong nhận được sự góp ý từ Thầy nhằm rút ra những kinh nghiệm quý báu và hoàn thiện vốn kiến thức để nhóm có thể hoàn thành những nghiên cứu, dự án khác trong tương lai.

Nhóm chúng em xin chân thành cảm ơn Thầy!

Mục Lục

DANH MỤC HÌNH VẼ.....	1
DANH MỤC CÁC TỪ VIẾT TẮT	3
PHẦN 1: MỞ ĐẦU	4
1.1. Giới thiệu đề tài	4
1.2. Mục đích nghiên cứu.....	4
1.3. Cách tiếp cận và phương pháp nghiên cứu.....	5
1.3.1. Đối tượng nghiên cứu.....	5
1.3.2. Phạm vi nghiên cứu	5
1.3.3. Tình hình nghiên cứu.....	5
1.4. Kết quả dự kiến đạt được	6
PHẦN 2: NỘI DUNG.....	8
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT	8
1.1. Tổng quan về phân tích cảm xúc.....	8
1.1.1. Phân tích cảm xúc là gì?.....	8
1.1.2. Các phương pháp phân tích cảm xúc	9
1.1.3. Phân tích cảm xúc hoạt động như thế nào ?	10
1.2. Xử lý ngôn ngữ tự nhiên (Natural Language Processing).....	11
1.2.1. Định nghĩa.....	11
1.2.2. Các kỹ thuật xử lý ngôn ngữ tự nhiên trong phân tích cảm xúc.....	12
1.2.3. Ứng dụng trong phân tích cảm xúc.....	15
1.2.4. Áp dụng các mô hình học sâu	17
1.2.5. Thuật ngữ	18
1.3 . Hệ thống gợi ý.....	20
1.3.1. Giới thiệu	20
1.3.2. Vai trò.....	21
1.3.3. Các phương pháp xây dựng hệ thống gợi ý	22
1.4. Lý thuyết các mô hình học máy	28
1.4.1. Mô hình Logistic Regression	28
1.4.2. Mô hình Random Forest Classifier	30
1.4.3. Mô hình Long Short Term Memory	33
1.4.4. Mô hình PhoBert	38
CHƯƠNG 2: XÂY DỰNG HỆ THỐNG VÀ GIẢI PHÁP	42
2.1. Mô tả xây dựng kiến trúc hệ thống	42
2.2. Thu thập và tiền xử lý dữ liệu.....	45
2.3. Xử lý ngôn ngữ tự nhiên để phân loại cảm xúc	52

2.4. Xây dựng mô hình học máy	56
2.4.1. Các phương pháp đánh giá	56
2.4.2. Mô hình truyền thống	57
2.4.3. Mô hình học sâu	60
2.5. Xây dựng hệ thống phân tích cảm xúc	66
2.5.1. Dự đoán cảm xúc với mô hình Random Forest với TF-IDF.....	66
2.5.2. Dự đoán cảm xúc với PhoBERT Fine-Tuned.....	68
2.5.3. Phát hiện sarcasm trong bình luận	71
2.6. Xây dựng hệ thống gợi ý.....	72
2.7. Giao diện hệ thống.....	81
CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ	86
3.1. Dữ liệu thực nghiệm.....	86
3.2. Môi trường thực nghiệm	87
3.2.1. Thư viện sử dụng.....	87
3.2.2. Cấu hình máy thực nghiệm	90
3.3. Mô hình thực nghiệm	90
3.4. Kết quả thực nghiệm.....	91
PHẦN 3: KẾT LUẬN	93
3.1. Kết quả đạt được	93
3.2. Hạn chế	93
3.3. Hướng phát triển	94
DANH MỤC TÀI LIỆU THAM KHẢO	95
PHỤ LỤC	97

DANH MỤC HÌNH VẼ

Hình 1: Minh họa quá trình phân tích cảm xúc trong một đoạn văn bản.	8
Hình 2: Mô tả tần suất xuất hiện của các từ.	13
Hình 3: Hình mô tả TF của các từ.	13
Hình 4: Hình mô tả IDF của các từ.	14
Hình 5: Hình mô tả tích TF và IDF của các từ.	14
Hình 6: Mô tả chuẩn hoá L2 của các từ.	15
Hình 7: Sơ đồ biểu diễn các thuật toán chính trong hệ thống gợi ý.	22
Hình 8: Sơ đồ minh họa quá trình lọc cộng tác để đưa ra gợi ý cá nhân hóa.	24
Hình 9: Sơ đồ minh họa quá trình lọc nội dung để đưa ra gợi ý cá nhân hóa.	26
Hình 10: Sơ đồ Phương pháp Gợi ý Lai (Hybrid Recommendations)	27
Hình 11: Cách xây dựng các cây quyết định.	30
Hình 12: Kết quả tổng hợp từ các cây quyết định.	31
Hình 13: Sơ đồ kiến trúc module trong mạng LSTM chứa 4 tầng ẩn.	34
Hình 14: Đường đi của ô trạng thái (cell state) trong mạng LSTM.	34
Hình 15: Một cổng của hàm sigmoid trong LSTM.	35
Hình 16: Tầng cổng quên (forget gate layer).	36
Hình 17: Cập nhật giá trị cho ô trạng thái bằng cách kết hợp 2 kết quả từ tầng cổng vào và tầng ẩn.	36
Hình 18: Ô trạng thái mới.	37
Hình 19: Điều chỉnh thông tin ở đầu ra thông qua hàm tanh.	37
Hình 20: Sơ đồ mô tả quy trình thu thập dữ liệu.	42
Hình 21: Sơ đồ quy trình thực hiện đề tài.	43
Hình 22: Mô tả dữ liệu sau khi thu thập tên sản phẩm, giá tiền, rating, số bình luận, link sản phẩm trên web.	46
Hình 23: Mô tả 10 dòng đầu của tập dữ liệu sau khi thu thập từ trang web của Thế Giới Di Động.	50
Hình 24: Mô tả 10 dòng đầu của tập dữ liệu sau khi tiền xử lý.	51
Hình 25: Mô tả 10 dòng đầu của tập dữ liệu sau khi xử lý ngôn ngữ tự nhiên và gán nhãn.	55
Hình 26: Giao diện khi phân tích cảm xúc của Random Forest.	68
Hình 27: Giao diện khi phân tích cảm xúc liên quan tới 3 khía cạnh của PhoBERT Fine tuned.	70
Hình 28: Giao diện khi phân tích cảm xúc liên quan tới 2 khía cạnh của PhoBERT Fine tuned.	71
Hình 29: Giao diện phát hiện bình luận ẩn ý sarcasm(khiêu khích).	72
Hình 30: Giao diện ban đầu của trang web gợi ý sản phẩm.	73
Hình 31: Giao diện khi gợi ý sản phẩm với bình luận tiêu cực.	78
Hình 32: Giao diện khi gợi ý sản phẩm với bình luận tiêu cực và giá cao.	79
Hình 33: Giao diện khi gợi ý sản phẩm với bình luận tích cực.	80

Hình 34: Giao diện trang home	81
Hình 35: Giao diện trang chọn sản phẩm	81
Hình 36: Giao diện trang nhập thông tin.....	82
Hình 37: Giao diện gợi ý sản phẩm.....	82
Hình 38: Giao diện phân tích cảm xúc.....	83
Hình 40: Biểu đồ thể hiện các kết quả cảm xúc theo bình luận của từng sản phẩm	84
Hình 41: Bộ lọc các bình luận theo các khía cạnh và cảm xúc cho từng sản phẩm.	85
Hình 42: Bảng kết quả thực nghiệm.....	91

DANH MỤC CÁC TỪ VIẾT TẮT

Tên viết tắt	Viết tắt tiếng anh	Nghĩa tiếng việt
SA	Sentiment Analysis	Phân tích cảm xúc
NLP	Natural Language Processing	Xử lý ngôn ngữ tự nhiên
VLSP	Vietnamese Language and Speech Processing	Xử lý ngôn ngữ và giọng nói tiếng Việt
MEM	Maximum Entropy Model	Mô hình Entropy cực đại
Bert	Bidirectional Encoder Representations from Transformers	Biểu diễn Mã hóa Hai chiều từ Transformer.
AI	Artificial Intelligence	Trí tuệ nhân tạo
RS	Recommender System	Hệ thống gợi ý
MTL	Multi-Task Learning	Học đa nhiệm

PHẦN 1: MỞ ĐẦU

1.1. Giới thiệu đề tài

Trong thời đại công nghệ 4.0, việc thu thập và phân tích dữ liệu từ các nền tảng trực tuyến đang ngày càng trở nên quan trọng, đặc biệt trong việc giúp doanh nghiệp hiểu rõ và nâng cao trải nghiệm của người dùng. Một trong những phương pháp phổ biến hiện nay là Phân tích cảm xúc (Sentiment Analysis), một kỹ thuật thuộc lĩnh vực Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP). Kỹ thuật này giúp nhận biết và phân loại cảm xúc của người dùng dựa trên những văn bản như đánh giá, bình luận hoặc phản hồi.

Với sự hỗ trợ của trí tuệ nhân tạo, phân tích cảm xúc giúp doanh nghiệp hiểu khách hàng tốt hơn, từ đó cải thiện trải nghiệm sử dụng sản phẩm và cung cấp các sản phẩm gợi ý phù hợp hơn. Điều này không chỉ làm tăng sự hài lòng của khách hàng mà còn giúp xây dựng lòng trung thành lâu dài.

Đề tài của nhóm em, “Xây dựng hệ thống phân tích cảm xúc khách hàng bằng xử lý ngôn ngữ tự nhiên để gợi ý sản phẩm”, tập trung vào việc tạo ra một hệ thống phân tích cảm xúc tự động. Hệ thống sẽ thu thập dữ liệu từ các nguồn như mạng xã hội, trang thương mại điện tử và ứng dụng di động để hiểu rõ cảm xúc và nhu cầu của người dùng. Dựa trên kết quả đó, doanh nghiệp có thể đưa ra các đề xuất sản phẩm phù hợp hơn, giúp khách hàng cảm thấy hài lòng hơn khi sử dụng dịch vụ.

1.2. Mục đích nghiên cứu

Mục tiêu chính của nghiên cứu là sẽ áp dụng hệ thống phân tích cảm xúc khách hàng dựa trên xử lý ngôn ngữ tự nhiên nhằm cho người dùng sẽ biết được những sản phẩm nào tốt nhất hoặc được đánh giá cao và sau đó sẽ đề xuất các sản phẩm phù hợp với sở thích và ý định của người dùng như là nhận diện và phân loại cảm xúc của người dùng từ các phản hồi hay đánh giá, phân tích xu hướng theo cảm xúc và đề xuất các sản phẩm để cải thiện nhu cầu của khách hàng.

Một trong những thách thức lớn phải đối diện trong việc triển khai hệ thống phân tích cảm xúc khách hàng dựa trên NLP là sự đa dạng và phức tạp trong ngôn ngữ của người dùng hiện nay. Ngôn ngữ tự nhiên chứa nhiều yếu tố tác động như lỗi chính tả, từ

viết tắt, tiếng lóng, hoặc ngữ cảnh mơ hồ, làm cho việc xử lý và nhận diện cảm xúc trở nên khó khăn. Ngoài ra, sự khác biệt văn hóa và cách biểu đạt cảm xúc giữa các nhóm người dùng cũng tạo ra các thách thức trong việc xây dựng mô hình đủ linh hoạt và chính xác để áp dụng trên diện rộng.

Nghiên cứu này của nhóm chúng em sẽ tập trung các phương pháp xử lý và xây dựng để xử lý ngôn ngữ tự nhiên một cách hiệu quả và tối đa và tăng độ chính xác của các mô hình học máy hay học sâu nhằm đem lại sự chính xác nhất có thể cho việc nhận diện được cảm xúc của khách hàng. Từ đó cung cấp thông tin tốt nhất cho việc đem lại cho gợi ý cho người dùng.

1.3. Cách tiếp cận và phương pháp nghiên cứu

1.3.1. Đối tượng nghiên cứu

Đối tượng nghiên cứu chính trong đề tài này là các đánh giá, nhận xét, phản hồi của các khách hàng đối với các sản phẩm từ các nền tảng thương mại điện tử.

1.3.2. Phạm vi nghiên cứu

Thu thập dữ liệu từ các nguồn và phân tích dữ liệu phản hồi, đánh giá sản phẩm của khách hàng từ các nền tảng thương mại điện tử như là Thế Giới Di Động và Điện Máy Xanh. Sau đó áp dụng các kỹ thuật NLP hiện đại như PhoBERT..., để phân loại cảm xúc của người dùng, từ đó xây dựng hệ thống gợi ý sản phẩm dựa trên ba tiêu chí gồm: tương đồng về nội dung, theo cảm xúc người dùng và hiển thị theo loại sản phẩm theo ý định của người dùng.

1.3.3. Tình hình nghiên cứu

Từ những năm 2000 trở lại đây, phân tích ý kiến (opinion mining) và phân tích cảm xúc (sentiment analysis) đã trở thành những lĩnh vực nghiên cứu hấp dẫn, thu hút nhiều sự quan tâm và có nhiều ứng dụng thực tiễn. Khái niệm phân tích cảm xúc lần đầu tiên được đề xuất trong nghiên cứu của Nasukawa và Yi, trong khi phân tích ý kiến được giới thiệu qua nghiên cứu của Dave, Lawrence và Pennock. Đặc biệt, công trình của Pang và Lillian Lee [6] được coi là nền móng cho sự phát triển của lĩnh vực này, góp phần quan trọng trong việc định hình và mở rộng nghiên cứu. Các ứng dụng của phân

tích cảm xúc rất đa dạng, từ phân tích đánh giá phim, bình luận sản phẩm tiêu dùng, và nhiều lĩnh vực khác.

Để giải quyết các vấn đề trong phân tích cảm xúc, nhiều nghiên cứu đã áp dụng các phương pháp như học máy, thống kê, và kỹ thuật dựa trên luật, kết hợp với việc khai thác dữ liệu ngôn ngữ. Nhờ vậy, lĩnh vực này đã đạt được những bước tiến lớn trong việc hiểu và phân tích cảm xúc trong ngôn ngữ, trở thành một chủ đề ngày càng được quan tâm và phát triển.

Với tiếng Việt, các nghiên cứu trong lĩnh vực phân tích cảm xúc vẫn đang được tiếp tục phát triển. Ví dụ, Kieu và Pham [7] đã trình bày một phương pháp dựa trên hệ thống luật để phân loại cảm xúc trên các đánh giá sản phẩm máy tính. Duyen cùng các cộng sự [8] đã áp dụng các thuật toán học máy như SVM, MEM vào phân tích cảm xúc của đánh giá khách sạn từ Agoda, trong khi Van và nhóm nghiên cứu [9] đã sử dụng SVM để xử lý bình luận trên Facebook. Một số nghiên cứu khác, như của Tran và Phan [10], đã tích hợp ngữ cảnh vào câu để cải thiện hiệu quả phân tích cảm xúc.

Những nỗ lực này đã đóng góp vào sự phát triển của phân tích cảm xúc tiếng Việt. Tuy nhiên, lĩnh vực này vẫn còn gặp nhiều thách thức, chưa có sự hệ thống hóa và định hướng rõ ràng, với phần lớn nghiên cứu chỉ dừng lại ở cấp độ thạc sĩ, tiến sĩ, hoặc các bài báo khoa học mang tính thăm dò. Đáng chú ý, các bộ dữ liệu phân tích cảm xúc tiếng Việt đã dần được xây dựng để phục vụ cộng đồng nghiên cứu. Năm 2016, cộng đồng VLSP đã tổ chức cuộc thi phân tích cảm xúc dựa trên phản hồi mua hàng của người dùng, một sáng kiến được dẫn dắt bởi Huyen và các đồng nghiệp. Gần đây, AIVIVN – nền tảng hỗ trợ các cuộc thi học máy – cũng đã công bố bộ dữ liệu phân loại sắc thái bình luận trên các trang thương mại điện tử, đóng góp tích cực vào lĩnh vực này.

1.4. Kết quả dự kiến đạt được

Nhờ sự kết hợp giữa các phương pháp phân tích cảm xúc dựa trên mô hình ngôn ngữ tiên tiến như PhoBERT và các thuật toán gợi ý sản phẩm dựa trên nội dung (Content-Based) thì đề tài của nhóm chúng em "Xây dựng hệ thống phân tích cảm xúc khách hàng bằng xử lý ngôn ngữ tự nhiên để gợi ý sản phẩm" được dự đoán mang lại nhiều kết quả tích cực và hữu ích.

Dự kiến, hệ thống sẽ cải thiện đáng kể độ chính xác trong việc nhận diện cảm xúc (tích cực, tiêu cực, trung lập) từ phản hồi khách hàng, từ đó gợi ý sản phẩm phù hợp với nhu cầu và sở thích cá nhân. Hệ thống không chỉ giúp nâng cao trải nghiệm người dùng mà còn thúc đẩy sự tương tác thông qua các đề xuất cá nhân hóa. Ngoài ra, nhờ khai thác dữ liệu và phân loại cảm xúc, hệ thống có thể dự đoán xu hướng thị trường và hỗ trợ các chiến lược kinh doanh hiệu quả hơn.

Bên cạnh đó, việc sử dụng các thuật toán gợi ý theo dự kiến sẽ đa dạng hóa danh sách sản phẩm đề xuất, giúp người dùng khám phá những sản phẩm mới mẻ mà họ chưa từng nghĩ đến. Điều này không chỉ tăng giá trị trải nghiệm mà còn cải thiện tỷ lệ chuyển đổi hóa cho các nền tảng thương mại điện tử.

Tuy nhiên, để đạt được những kết quả này, đề tài đòi hỏi sự đầu tư đáng kể trong việc xây dựng tập dữ liệu huấn luyện, tối ưu hóa mô hình ngôn ngữ và thuật toán gợi ý, cũng như đảm bảo tính bảo mật và riêng tư trong việc xử lý dữ liệu người dùng. Sự thành công của hệ thống sẽ phụ thuộc vào khả năng nghiên cứu và triển khai thực tế trên dữ liệu thực, cũng như việc liên tục kiểm tra, đánh giá để cải thiện hiệu suất và độ tin cậy từ phản hồi của người dùng.

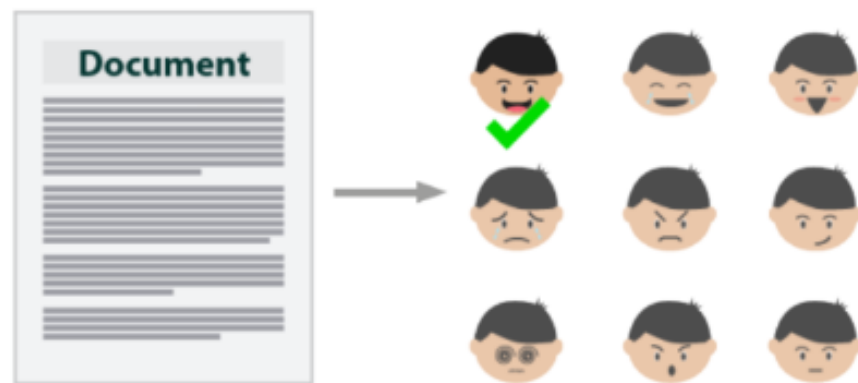
PHẦN 2: NỘI DUNG

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về phân tích cảm xúc

1.1.1. Phân tích cảm xúc là gì?

Phân tích cảm xúc (SA) là một nhánh quan trọng trong lĩnh vực Xử lý ngôn ngữ tự nhiên (NLP), với mục tiêu nhận diện và đánh giá cảm xúc của con người thông qua văn bản, như bình luận, câu nói, hay đánh giá. Phân tích cảm xúc giúp phân loại các trạng thái cảm xúc thành tích cực, tiêu cực hoặc trung tính. Với sự phát triển mạnh mẽ của mạng xã hội, các kỹ thuật NLP ngày càng được ứng dụng rộng rãi để trích xuất và phân tích thông tin từ hàng triệu bài viết, qua đó đánh giá mức độ quan tâm của người dùng đối với các sự kiện hoặc nội dung được chia sẻ. Những kết quả này cung cấp cho doanh nghiệp và tổ chức cái nhìn sâu sắc hơn về ý kiến khách hàng, từ đó hỗ trợ cải tiến sản phẩm, dịch vụ và xây dựng chiến lược truyền thông hiệu quả hơn.



Hình 1: Minh họa quá trình phân tích cảm xúc trong một đoạn văn bản.

Ngoài giá trị nghiên cứu học thuật, phân tích cảm xúc còn đóng vai trò quan trọng trong các ngành công nghiệp, đặc biệt là lĩnh vực dịch vụ, nơi việc hiểu rõ hành vi và cảm xúc của khách hàng là yếu tố then chốt để nâng cao trải nghiệm người dùng. Tuy nhiên, việc phân tích cảm xúc không hề đơn giản do ngôn ngữ tự nhiên thường mang tính nhập nhằng về ngữ nghĩa. Thêm vào đó, người dùng còn sử dụng từ viết tắt, tiếng lóng và các biểu tượng cảm xúc như "=))", ":(", ">,<" để bộc lộ trạng thái, gây khó khăn cho việc xử lý dữ liệu một cách chính xác.

Hiện nay, bài toán phân tích cảm xúc thường được thực hiện ở ba cấp độ chính: cấp độ câu (sentence-level), cấp độ văn bản (document-level) và cấp độ khía cạnh (aspect-level). Ở cấp độ câu, mục tiêu là phân loại cảm xúc của từng câu văn thành tích cực, tiêu cực hoặc trung tính. Cấp độ văn bản đi xa hơn, tập trung vào đánh giá cảm xúc tổng thể của một đoạn văn hoặc tài liệu gồm nhiều câu. Trong khi đó, cấp độ khía cạnh đi sâu vào việc phân tích cảm xúc theo từng khía cạnh cụ thể được đề cập trong văn bản. Trong phạm vi đề án này, nhóm sẽ tập trung vào phân tích cảm xúc ở cấp độ câu, vì đây là phương pháp đơn giản hơn nhưng vẫn đảm bảo cung cấp thông tin hữu ích và tối ưu hóa kết quả.

1.1.2. Các phương pháp phân tích cảm xúc

Các phương pháp phân tích cảm xúc sẽ được chia thành các nhóm như sau :

- Phương pháp dựa trên từ điển cảm xúc (Lexicon-Based Approach): Phương pháp này sử dụng các từ điển cảm xúc đã được xây dựng sẵn, trong đó các từ hoặc cụm từ được gán điểm số cảm xúc (tích cực, tiêu cực, trung tính). Khi phân tích văn bản, hệ thống so khớp các từ trong văn bản với từ điển để xác định cảm xúc tổng thể. Phương pháp này được sử dụng phổ biến trong các bài toán phân tích dữ liệu mạng xã hội, đặc biệt là trên các nền tảng như Twitter hoặc Facebook, nơi cần xử lý văn bản ngắn gọn.

- Phương pháp dựa trên học máy (Machine Learning-Based Approach) : Phương pháp học máy dựa trên việc sử dụng dữ liệu được gán nhãn trước (tích cực, tiêu cực, trung tính) để huấn luyện các mô hình phân loại. Sau khi được huấn luyện, mô hình có thể dự đoán cảm xúc từ văn bản chưa biết dựa trên các đặc trưng được trích xuất, như Bag of Words (BoW), TF-IDF, hoặc Word Embeddings. Phương pháp này thường được sử dụng trong việc phân tích phản hồi khách hàng, đánh giá các bài viết blog, hoặc các bình luận trực tuyến. Nó cũng là nền tảng cho các hệ thống phân loại email và tự động hóa quy trình dịch vụ khách hàng.

- Phương pháp học sâu (Deep Learning-Based Approach) : Học sâu sử dụng các mô hình mạng thần kinh như RNN, LSTM, GRU, hoặc Transformers để phân tích cảm xúc. Mô hình này có khả năng tự động học các đặc trưng từ dữ liệu mà không cần quá trình tiền xử lý phức tạp. Phương pháp này được áp dụng rộng rãi trong các hệ thống

phức tạp, như chatbot AI thông minh, phân tích dữ liệu lớn trên các nền tảng truyền thông xã hội, và hỗ trợ dịch vụ khách hàng. Các mô hình như phoBERT hoặc GPT thường được sử dụng để hiểu ngữ cảnh văn bản và tạo phản hồi thông minh.

- Phân tích cảm xúc đa chiều (Aspect-Based Sentiment Analysis - ABSA): Phân tích cảm xúc đa chiều tập trung vào việc phân loại cảm xúc theo từng khía cạnh cụ thể trong văn bản, thay vì phân loại cảm xúc chung cho cả văn bản. Ví dụ, một đánh giá về sản phẩm có thể có cảm xúc tích cực về chất lượng nhưng tiêu cực về giá cả. ABSA thường được sử dụng trong thương mại điện tử để phân tích các yếu tố cụ thể của sản phẩm (như chất lượng, giá cả, hoặc dịch vụ). Điều này giúp các doanh nghiệp hiểu rõ hơn về từng khía cạnh quan trọng với khách hàng.

- Kết hợp các phương pháp (Hybrid Approach): Phương pháp này kết hợp các đặc điểm của cả từ điển cảm xúc và học máy hoặc học sâu. Từ điển cảm xúc được sử dụng để hỗ trợ trong việc xử lý ngôn ngữ và trích xuất đặc trưng, sau đó được đưa vào các mô hình học máy hoặc học sâu để cải thiện độ chính xác.

1.1.3. Phân tích cảm xúc hoạt động như thế nào ?

Một cách tiếp cận phổ biến trong phân tích cảm xúc là tạo ra một từ điển chứa thông tin về các từ hoặc cụm từ mang ý nghĩa tích cực và tiêu cực. Từ điển này có thể được xây dựng thủ công hoặc tự động bằng các phương pháp thuật toán. Quá trình gán nhãn cảm xúc cho từ điển hoặc tập dữ liệu thường được thực hiện bằng tay, sau đó các mô hình học máy được huấn luyện để phân loại các từ hoặc cụm từ mới dựa trên các đặc trưng đã được trích xuất.

Bên cạnh việc sử dụng từ điển, một phương pháp khác là phân tích cảm xúc trên toàn bộ câu hoặc văn bản thay vì tập trung vào từng từ riêng lẻ. Cách tiếp cận này cố gắng xác định cảm xúc tổng thể của tài liệu, nhưng có thể gặp khó khăn khi cảm xúc chỉ xuất hiện trong một số câu hoặc cụm từ cụ thể.

Ngoài ra, phân tích cảm xúc còn có thể khai thác ý kiến từ các nguồn trên web. Phương pháp này nhằm trích xuất, tóm tắt và theo dõi các thông tin chủ quan, giúp ích trong việc

theo dõi xu hướng, cải thiện chiến lược marketing hoặc nâng cao trải nghiệm khách hàng thông qua phân tích đánh giá trực tuyến và phương tiện truyền thông xã hội.

Các kỹ thuật học máy trong phân tích cảm xúc được chia thành hai nhóm chính là mô hình truyền thống và mô hình học sâu.

- Mô hình truyền thống: Các thuật toán như Random Forest, máy vector hỗ trợ (SVM) và Logistic Regression được sử dụng phổ biến. Những mô hình này dựa trên các đặc trưng từ vựng, từ điển cảm xúc, hoặc các yếu tố ngữ pháp như tính từ và trạng từ. Hiệu quả của chúng phụ thuộc lớn vào việc lựa chọn và xử lý đặc trưng đầu vào.
- Mô hình học sâu: Các phương pháp hiện đại như Bộ nhớ dài ngắn hạn (LSTM), Gated Recurrent Unit (GRU) và Convolutional Neural Network (CNN) mang lại hiệu suất cao hơn bằng cách học tự động các đặc trưng từ dữ liệu. Chúng có thể phân tích cảm xúc ở nhiều cấp độ khác nhau, từ toàn bộ văn bản, các cụm từ, đến những khía cạnh cụ thể trong tài liệu. Những mô hình này ngày càng phổ biến trong các ứng dụng phân tích cảm xúc nhờ khả năng xử lý dữ liệu phức tạp một cách hiệu quả.

1.2. Xử lý ngôn ngữ tự nhiên (Natural Language Processing)

1.2.1. Định nghĩa

Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) là một lĩnh vực thuộc trí tuệ nhân tạo (AI) và khoa học máy tính, nghiên cứu về cách thức để máy tính có thể hiểu, phân tích và tương tác với ngôn ngữ của con người. Mục tiêu chính của NLP là giúp máy móc có khả năng "đọc", "hiểu" và "sử dụng" ngôn ngữ tự nhiên một cách hiệu quả, tương tự như cách con người giao tiếp trong đời sống hàng ngày. Để đạt được điều này, các chuyên gia phát triển các phương pháp giúp máy tính nhận diện và xử lý cấu trúc ngữ pháp, ý nghĩa của từ ngữ, cũng như hiểu được ngữ cảnh của các câu từ trong văn bản hoặc lời nói.

Xử lý ngôn ngữ tự nhiên (NLP) bao gồm nhiều nhiệm vụ khác nhau, chẳng hạn như phân tích cảm xúc, dịch ngôn ngữ, nhận diện thực thể và trả lời câu hỏi. Để thực

hiện những nhiệm vụ này, các kỹ thuật học máy, học sâu (deep learning) và các mô hình thống kê được áp dụng để giúp máy tính hiểu và xử lý văn bản tự nhiên. Nhờ vào đó, máy móc có thể thực hiện các tác vụ phức tạp liên quan đến ngôn ngữ và cải thiện khả năng tương tác giữa con người và máy tính.

Với sự tiến bộ của NLP, ngày càng có nhiều ứng dụng thực tế trong cuộc sống, từ hỗ trợ khách hàng qua các chatbot thông minh, dịch ngôn ngữ một cách tự động, cho đến phân tích và khai thác thông tin từ các dữ liệu văn bản lớn. Những ứng dụng này không chỉ giúp cải thiện dịch vụ mà còn nâng cao chất lượng sản phẩm và trải nghiệm của người dùng.

1.2.2. Các kỹ thuật xử lý ngôn ngữ tự nhiên trong phân tích cảm xúc

Tiền xử lý là bước quan trọng đầu tiên trong xử lý ngôn ngữ tự nhiên (NLP). Trong bước này, mỗi từ sẽ được phân tích và các ký tự không phải là chữ (như dấu câu) sẽ được tách ra khỏi các từ. Trong tiếng Anh và nhiều ngôn ngữ khác, các từ được phân tách bằng dấu cách. Tuy nhiên, trong tiếng Việt thì dấu cách được sử dụng để phân tách các tiếng (âm tiết) chứ không phải là từ. Tương tự với các ngôn ngữ như tiếng Trung, tiếng Hàn, tiếng Nhật, phân tách từ trong tiếng Việt là một công việc không hề đơn giản. Các kỹ thuật cơ bản trong bước này bao gồm: Tách từ, Loại bỏ từ dừng hay Ghi đè từ.

Biểu diễn dữ liệu văn bản trong NLP là quá trình chuyển đổi văn bản thành một dạng mà máy tính có thể hiểu và xử lý. Các phương pháp biểu diễn dữ liệu văn bản phổ biến bao gồm: Bag of Words hay Text Vectorization.

Hai cách thông thường để biểu diễn văn bản dưới dạng vector là Count Vector và TF-IDF.

- Count Vector:

Bước 1: Tất cả các từ duy nhất được đặt trong các cột và tất cả tài liệu được đặt trong các hàng.

Bước 2: Tần suất xuất hiện của các từ trong các tài liệu được đặt tại các ô tại sự giao nhau.

	this	is	the	first	document	second	and	third	one
This is the first document	1	1	1	1	1				
This document is the second document	1	1	1		2	1			
And this is the third one	1	1	1				1	1	1
Is this the first document	1	1	1	1	1				

Hình 2: Mô tả tần suất xuất hiện của các từ.

- **TF-IDF (Term Frequency-Inverse Document Frequency):** Là một kỹ thuật phổ biến trong Xử lý Ngôn ngữ Tự nhiên và khai phá văn bản, được sử dụng để đánh giá tầm quan trọng của một từ trong một tài liệu thuộc một tập hợp tài liệu. TF-IDF giúp chuyển đổi văn bản thành các số liệu mà máy có thể xử lý, đặc biệt hữu ích cho các bài toán như phân loại văn bản, tìm kiếm tài liệu, và trích xuất thông tin.

Cách thức hoạt động của TF-IDF: gồm TF và IDF

Term Frequency - (Tần suất xuất hiện của từ trong tài liệu): TF đo lường số lần một từ xuất hiện trong một tài liệu cụ thể. Nếu một từ xuất hiện nhiều lần trong một tài liệu, điều đó cho thấy từ này quan trọng trong tài liệu đó.

Bước 1: Tính toán Count Vector.

Bước 2: Tính toán TF.

$$TF = \frac{1 + \log(\text{Số lượng từ khóa})}{\log(\text{số lượng từ})}$$

	this	is	the	first	document	second	and	third	one
This is the first document	0.2	0.2	0.2	0.2	0.2	0	0	0	0
This document is the second document	0.167	0.167	0.167	0	0.3333	0.17	0	0	0
And this is the third one	0.167	0.167	0.167	0	0	0	0.17	0.17	0.17
Is this the first document	0.2	0.2	0.2	0.2	0.2	0	0	0	0

Hình 3: Hình mô tả TF của các từ.

Inverse Document Frequency - (Tần suất nghịch của từ trong tập tài liệu): IDF đo lường mức độ phổ biến của từ trên toàn bộ tập tài liệu. Nếu một từ xuất hiện trong nhiều tài liệu, nó có thể ít quan trọng hơn, vì nó không đặc trưng cho một tài liệu cụ thể.

Bước 3: Tính toán IDF: Nó đo lường tỷ lệ giữa tổng số lượng tài liệu với số lượng các tài liệu có chứa từ khóa đó

$$IDF = 1 + \log \frac{\text{tổng số văn bản} + 1}{\text{số lượng tài liệu chứa từ} + 1}$$

	this	is	the	first	document	second	and	third	one
number of documents with term t in it	4	4	4	2	3	1	1	1	1

	this	is	the	first	document	second	and	third	one
IDF	1.00	1.00	1.00	1.51	1.22	1.91	1.91	1.91	1.91

Hình 4: Hình mô tả IDF của các từ.

Nếu một từ có tần suất quan sát cao trong toàn bộ tập dữ liệu, điều này có nghĩa rằng từ tương ứng này ảnh hưởng đến toàn bộ tập dữ liệu. Trong trường hợp này, tiến hành một quá trình chuẩn hóa đối với tần suất xuất hiện cả bên trong các từ và trong toàn bộ tập dữ liệu.

Bước 4: Tính toán TF*IDF.

	this	is	the	first	document	second	and	third	one
This is the first document	0.2	0.2	0.2	0.2	0.2446	0	0	0	0
This document is the second document	0.167	0.167	0.167	0	0.4077	0.3194	0	0	0
And this is the third one	0.167	0.167	0.167	0	0	0	0.32	0.32	0.32
Is this the first document	0.2	0.2	0.2	0.2	0.2446	0	0	0	0

Hình 5: Hình mô tả tích TF và IDF của các từ.

Bước 5: Chuẩn hóa L2

Tìm căn bậc hai của tổng bình phương của các hàng, và sau đó chia các ô tương ứng cho giá trị được tìm thấy.

Công thức: $\sqrt{\sum x_i^2}$

	this	is	the	first	document	second	and	third	one
This is the first document	0.384	0.384	0.384	0.58	0.4697	0	0	0	0
This document is the second document	0.281	0.281	0.281	0	0.6876	0.5386	0	0	0
And this is the third one	0.281	0.267	0.267	0	0	0	0.51	0.51	0.15 1
Is this the first document	0.384	0.384	0.384	0.58	0.4697	0	0	0	0

	The square root of the sum of the squares
This is the first document	0.520717
This document is the second document	0.592932
And this is the third one	0.623977
Is this the first document	0.520717

Hình 6: Mô tả chuẩn hoá L2 của các từ.

Chuẩn hóa L2 (hoặc chuẩn hóa Euclidean) sửa lại cho các từ không thể hiển thị tác động của nó do sự xuất hiện của các giá trị bị thiếu trong một số hàng.

Mục tiêu: Khi tính TF-IDF, giá trị sẽ cao nếu từ đó xuất hiện nhiều lần trong một tài liệu (TF cao) nhưng lại xuất hiện trong ít tài liệu khác trong tập dữ liệu (IDF cao). Điều này cho phép mô hình nhấn mạnh những từ đặc trưng và loại bỏ những từ phổ biến không mang nhiều thông tin (như "the", "is", "in").

Ví dụ: Nếu từ “tốt” xuất hiện rất nhiều trong các bình luận về sản phẩm, nó có thể có giá trị thấp trong TF-IDF so với một từ ít xuất hiện như “tiện lợi”.

Word Embeddings :là cách giúp mô hình hiểu được ngữ nghĩa của từ và mối quan hệ ngữ nghĩa giữa các từ. Các kỹ thuật như Word2Vec và GloVe đã được sử dụng để cải thiện hiệu quả của việc phân loại chủ đề.

Ví dụ: Từ "công nghệ" có thể có vector gần với các từ như "kỹ thuật" và "đổi mới" trong không gian vector, phản ánh mối quan hệ ngữ nghĩa giữa chúng.

1.2.3. Ứng dụng trong phân tích cảm xúc

Trong thực tế, phân tích cảm xúc ngày càng được áp dụng rộng rãi trong các lĩnh vực như kinh doanh, marketing, chăm sóc khách hàng và nghiên cứu thị trường, mang lại những lợi ích thiết thực cho các doanh nghiệp. Các công ty hiện nay thường sử dụng các công cụ phân tích cảm xúc để quét và đánh giá các bình luận, đánh giá trực tuyến, bài viết trên mạng xã hội hay email từ khách hàng. Điều này giúp họ nhanh chóng nắm

bắt được phản hồi từ khách hàng, hiểu rõ mức độ hài lòng hoặc không hài lòng về sản phẩm hay dịch vụ và từ đó đưa ra các chiến lược cải tiến hiệu quả.

Một trong những ứng dụng chính của NLP trong phân tích cảm xúc là phân tích đánh giá sản phẩm, khi các công ty có thể sử dụng NLP để phân tích hàng triệu đánh giá và bình luận về sản phẩm trên các trang web thương mại điện tử như Thế Giới Di Động hay Shopee. Việc này giúp họ hiểu rõ hơn về cảm nhận của khách hàng, từ đó cải thiện chất lượng sản phẩm và đáp ứng nhu cầu của người tiêu dùng.

Ngoài ra, NLP còn được áp dụng trong giám sát mạng xã hội, nơi các doanh nghiệp theo dõi phản ứng của khách hàng đối với các chiến dịch marketing, sự kiện hay hình ảnh thương hiệu trên các nền tảng như Twitter hay Facebook. Việc phân tích cảm xúc trong các bài đăng này giúp nhận diện nhanh chóng các phản hồi tích cực hay tiêu cực, từ đó đưa ra các biện pháp điều chỉnh kịp thời.

NLP cũng đóng vai trò quan trọng trong chăm sóc khách hàng tự động, đặc biệt là trong các chatbot hoặc trợ lý ảo, giúp nhận diện cảm xúc của khách hàng trong cuộc trò chuyện và cải thiện trải nghiệm người dùng thông qua phản hồi nhanh chóng và phù hợp. Điều này không chỉ giảm thiểu thời gian chờ đợi mà còn giúp nâng cao sự hài lòng của khách hàng.

Bên cạnh đó, phân tích cảm xúc còn hỗ trợ dự báo xu hướng tiêu dùng, khi các công ty có thể phát hiện những xu hướng mới trong hành vi người tiêu dùng thông qua các đánh giá và bình luận, từ đó xây dựng các chiến lược marketing hiệu quả hơn và đáp ứng nhu cầu khách hàng một cách chính xác.

Cuối cùng, NLP cũng được sử dụng để phân tích thông tin từ các bài viết và nghiên cứu, giúp các tổ chức hiểu rõ hơn về cách công chúng phản ứng với các vấn đề xã hội hoặc chính trị, từ đó cung cấp thông tin quan trọng cho các quyết định kịp thời. Nhờ vào sự tiến bộ vượt bậc của các mô hình NLP hiện đại, phân tích cảm xúc giờ đây không chỉ chính xác hơn mà còn nhanh chóng và hiệu quả hơn, mở ra nhiều cơ hội ứng dụng phong phú trong các lĩnh vực từ kinh doanh đến nghiên cứu xã hội.

1.2.4. Áp dụng các mô hình học sâu

Trong những năm gần đây, việc áp dụng các mô hình học sâu (Deep Learning) vào phân tích cảm xúc đã trở thành xu hướng mạnh mẽ. Những mô hình này mang lại khả năng phân tích chính xác hơn nhiều so với các phương pháp truyền thống nhờ khả năng tự động học các đặc điểm của dữ liệu mà không cần sự can thiệp quá nhiều từ con người. Khi áp dụng vào phân tích cảm xúc, các mô hình học sâu giúp máy tính hiểu được cảm xúc người dùng từ các bình luận, bài viết, hoặc đánh giá, từ đó phân loại cảm xúc đó là tích cực, tiêu cực hay trung lập.

Một trong những mô hình phổ biến trong phân tích cảm xúc là LSTM (Long Short-Term Memory), đây là một loại mạng nơ-ron hồi tiếp có khả năng ghi nhớ thông tin từ các câu văn dài. LSTM rất hiệu quả khi phải xử lý các văn bản có ngữ cảnh dài và phức tạp, giúp phân tích chính xác hơn cảm xúc được thể hiện qua nhiều câu. Mô hình này có thể giữ lại thông tin quan trọng từ đầu câu cho đến cuối câu, điều này cực kỳ hữu ích khi phải phân tích các cảm xúc phức tạp.

Bên cạnh LSTM, một mô hình khác cũng rất hữu ích trong phân tích cảm xúc là CNN (Convolutional Neural Networks). Mặc dù CNN thường được dùng trong nhận diện hình ảnh, nhưng trong xử lý ngôn ngữ, CNN có thể giúp phân tích các đặc trưng cục bộ của một câu, tức là nó tìm ra các mẫu cảm xúc trong các đoạn văn ngắn. CNN khá hiệu quả trong những tình huống văn bản đơn giản hoặc khi phân tích những câu ngắn, nơi thông tin cảm xúc thường rõ ràng và dễ nhận diện.

Tuy nhiên, bước đột phá lớn trong xử lý ngôn ngữ tự nhiên là sự ra đời của mô hình Transformer, đặc biệt là BERT (Bidirectional Encoder Representations from Transformers). BERT giúp máy tính hiểu ngữ cảnh của một từ trong câu bằng cách nhìn nhận cả từ phía trước và phía sau từ đó, thay vì chỉ phân tích theo chiều một chiều như các mô hình trước đây. Điều này giúp BERT phân tích ngữ nghĩa và cảm xúc của câu chính xác hơn, nhất là với các câu có cấu trúc phức tạp hoặc có các từ mang nghĩa bóng.

Đối với ngữ cảnh tiếng Việt, PhoBERT là một phiên bản đặc biệt của BERT được phát triển để tối ưu hóa cho ngôn ngữ này. PhoBERT được huấn luyện trên một bộ dữ liệu lớn của tiếng Việt, giúp mô hình này hiểu được các từ ngữ, cấu trúc câu và đặc trưng

ngữ nghĩa của tiếng Việt. Nhờ PhoBERT, việc phân tích cảm xúc trong các văn bản tiếng Việt trở nên chính xác hơn, kể cả với các từ viết tắt, từ lóng, hay biểu cảm cảm xúc như "=)", ":(", ">,<", thường thấy trong các bình luận trên mạng xã hội.

Sử dụng PhoBERT trong phân tích cảm xúc giúp cải thiện độ chính xác khi nhận diện cảm xúc của người dùng, đặc biệt là trong các tình huống không chính thức hoặc không chuẩn mực như trên mạng xã hội. PhoBERT có khả năng nhận diện cảm xúc một cách tự nhiên và chính xác, kể cả những cảm xúc khó nhận ra hoặc ẩn giấu trong câu văn.

Tóm lại, các mô hình học sâu, đặc biệt là PhoBERT, đã mang lại bước tiến lớn trong việc phân tích cảm xúc. Những mô hình này không chỉ giúp máy tính hiểu được cảm xúc người dùng mà còn tự động hóa quá trình phân tích từ các nền tảng trực tuyến, giúp doanh nghiệp hiểu rõ hơn về khách hàng và đưa ra những cải tiến phù hợp.

1.2.5. Thuật ngữ

Tiền xử lý (Preprocessing): Là bước đầu tiên trong xử lý ngôn ngữ tự nhiên, tiền xử lý giúp làm sạch và chuẩn bị dữ liệu cho các bước phân tích tiếp theo. Trong tiền xử lý, các từ sẽ được phân tích, loại bỏ các ký tự không cần thiết như dấu câu, và các từ dừng (stop words) sẽ được loại bỏ. Đặc biệt, đối với tiếng Việt, việc phân tách từ không hề đơn giản do tiếng Việt không sử dụng dấu cách để phân tách từ mà chỉ phân tách các âm tiết, đòi hỏi các kỹ thuật phân tách từ chuyên biệt.

Túi từ - Bag of Words (BoW): Là một trong những phương pháp phổ biến để biểu diễn văn bản dưới dạng vector. Trong phương pháp này, mỗi từ duy nhất trong tập dữ liệu được xem như một đặc trưng (feature), và tần suất xuất hiện của chúng trong tài liệu được sử dụng làm giá trị của các đặc trưng đó.

Count Vector: Là một kỹ thuật trong đó mỗi từ trong văn bản được biểu diễn dưới dạng một số đếm tương ứng với tần suất xuất hiện của từ đó trong các tài liệu. Đặc điểm của Count Vector là không tính đến sự quan trọng hay sự xuất hiện của từ trong các tài liệu khác, mà chỉ đơn giản là số lần xuất hiện của từ.

TF-IDF (Term Frequency-Inverse Document Frequency): Là một kỹ thuật biểu diễn văn bản được sử dụng để xác định tầm quan trọng của một từ trong một tài liệu cụ thể. TF-IDF kết hợp giữa TF (Tần suất xuất hiện của từ trong tài liệu) và IDF (Tần suất nghịch của từ trong tập tài liệu), giúp giảm trọng số của những từ xuất hiện quá thường xuyên trong toàn bộ tập tài liệu, từ đó làm nổi bật những từ đặc trưng hơn. Đây là kỹ thuật quan trọng khi cần phân loại văn bản hoặc tìm kiếm tài liệu.

Word Embeddings: Là một kỹ thuật biểu diễn các từ dưới dạng vector trong không gian nhiều chiều. Word Embeddings giúp mô hình hiểu được mối quan hệ ngữ nghĩa giữa các từ, giúp cải thiện việc phân loại và phân tích văn bản. Các mô hình như Word2Vec và GloVe sử dụng các kỹ thuật học sâu để xây dựng các vector từ, sao cho các từ có nghĩa tương tự sẽ có các vector gần nhau trong không gian.

TF (Term Frequency): Là tần suất xuất hiện của một từ trong một tài liệu cụ thể. TF được sử dụng để đo lường tầm quan trọng của từ trong tài liệu, với giả thuyết rằng các từ xuất hiện nhiều lần trong một tài liệu là những từ quan trọng.

IDF (Inverse Document Frequency): Là một chỉ số đo lường mức độ quan trọng của một từ trong toàn bộ tập tài liệu. IDF giúp giảm trọng số của những từ xuất hiện trong hầu hết các tài liệu, vì những từ này không có tính đặc trưng cao.

Chuẩn hóa L2 - L2 Normalization: Là kỹ thuật chuẩn hóa để đảm bảo rằng các giá trị trong vector không bị ảnh hưởng bởi các giá trị lớn hoặc thiếu dữ liệu. Quá trình này giúp giảm sự thiên lệch trong dữ liệu và làm cho các vector dễ dàng so sánh với nhau.

Stopwords (Từ dừng): Là những từ không mang nhiều thông tin ý nghĩa trong việc phân tích cảm xúc, chẳng hạn như "và", "của", "là". Trong quá trình tiền xử lý, các từ dừng thường bị loại bỏ để tập trung vào những từ quan trọng hơn trong việc phân tích.

Tokenization (Tách từ): Là quá trình chia văn bản thành các đơn vị nhỏ hơn, gọi là "tokens". Những tokens này có thể là các từ, câu hoặc các ký tự phụ thuộc vào phương pháp tách từ được sử dụng.

Vectorization (Biểu diễn dưới dạng vector): Là quá trình chuyển đổi văn bản thành các đại diện số mà máy tính có thể xử lý. Các kỹ thuật vector hóa như Count Vector, TF-IDF và Word Embeddings giúp chuyển đổi văn bản thành các vector số để áp dụng vào các mô hình học máy.

1.3 . Hệ thống gợi ý

1.3.1. Giới thiệu

Hệ thống gợi ý (RS) là một tập hợp hay công cụ và phần mềm được thiết kế để cung cấp đề xuất các mặt hàng (items) cho người dùng (users). Chúng được thiết kế để hỗ trợ người dùng đưa ra quyết định trong nhiều lĩnh vực khác nhau, chẳng hạn như lựa chọn sách, âm nhạc, phim ảnh, hoặc sản phẩm mua sắm trực tuyến. Hệ thống này đã trở thành một phần không thể thiếu trong các nền tảng công nghệ, từ Spotify gợi ý bài hát, Lazada hay Thế giới di động đề xuất sản phẩm hay giới thiệu sản phẩm liên quan, Disney + gợi ý phim truyền hình, đến các nền tảng tin tức trực tuyến như Zing News.

RS hoạt động bằng cách dự đoán "điểm đánh giá" (rating) hoặc "mức độ yêu thích" (preference) của người dùng đối với một mặt hàng cụ thể [12]. Điều này được thực hiện thông qua việc phân tích dữ liệu thu thập được từ hành vi và tương tác của người dùng. Nhiều công ty công nghệ lớn đã triển khai các hệ thống đề xuất để nâng cao trải nghiệm người dùng và tối ưu hóa lợi nhuận, ví dụ: Amazon sử dụng RS để đề xuất sản phẩm phù hợp với lịch sử mua sắm của khách hàng hay là Facebook gợi ý kết bạn, các trang yêu thích, hoặc nhóm phù hợp với người dùng.

Sự phát triển của hệ thống gợi ý bắt đầu từ quan sát rằng con người thường dựa vào đề xuất từ người khác khi đưa ra quyết định hàng ngày. Để mô phỏng hành vi này, các hệ thống đề xuất đã sử dụng thuật toán để tận dụng đề xuất của cộng đồng người dùng và tạo ra đề xuất cho người dùng đang hoạt động.

RS đã chứng minh giá trị của mình trong việc giải quyết vấn đề quá tải thông tin trên web và trong các dịch vụ kinh doanh điện tử. Chúng giúp người dùng tìm kiếm và khám phá mặt hàng mới, giảm thiểu rủi ro quyết định sai lầm. Đồng thời, RS có khả năng cá nhân hóa đề xuất theo nhu cầu và ngữ cảnh cụ thể của người dùng, tạo ra một trải nghiệm tìm kiếm độc đáo và hữu ích.

1.3.2. Vai trò

Hệ thống gợi ý (RS) đóng vai trò vô cùng quan trọng trong việc nâng cao trải nghiệm người dùng và hỗ trợ doanh nghiệp phát triển bền vững. Đối với người dùng, hệ thống này mang lại sự tiện lợi và cá nhân hóa cao bằng cách đưa ra các đề xuất phù hợp với sở thích, nhu cầu và hành vi mua sắm của họ. Điều này giúp người dùng dễ dàng tìm thấy sản phẩm mong muốn trong vô vàn lựa chọn, giảm thời gian tìm kiếm, đồng thời tăng mức độ hài lòng và khuyến khích họ sử dụng hệ thống thường xuyên hơn.

Đối với doanh nghiệp, hệ thống gợi ý là công cụ mạnh mẽ giúp tối ưu hóa doanh thu thông qua các chiến lược up-selling (gợi ý sản phẩm cao cấp hơn) và cross-selling (đề xuất các sản phẩm liên quan hoặc bổ trợ). Nhờ đó, giá trị trung bình của mỗi đơn hàng được gia tăng, đồng thời mang lại trải nghiệm mua sắm phong phú và toàn diện hơn cho khách hàng. Những đề xuất này không chỉ đáp ứng nhu cầu hiện tại mà còn mở rộng nhận thức của khách hàng về các sản phẩm khác mà họ có thể quan tâm.

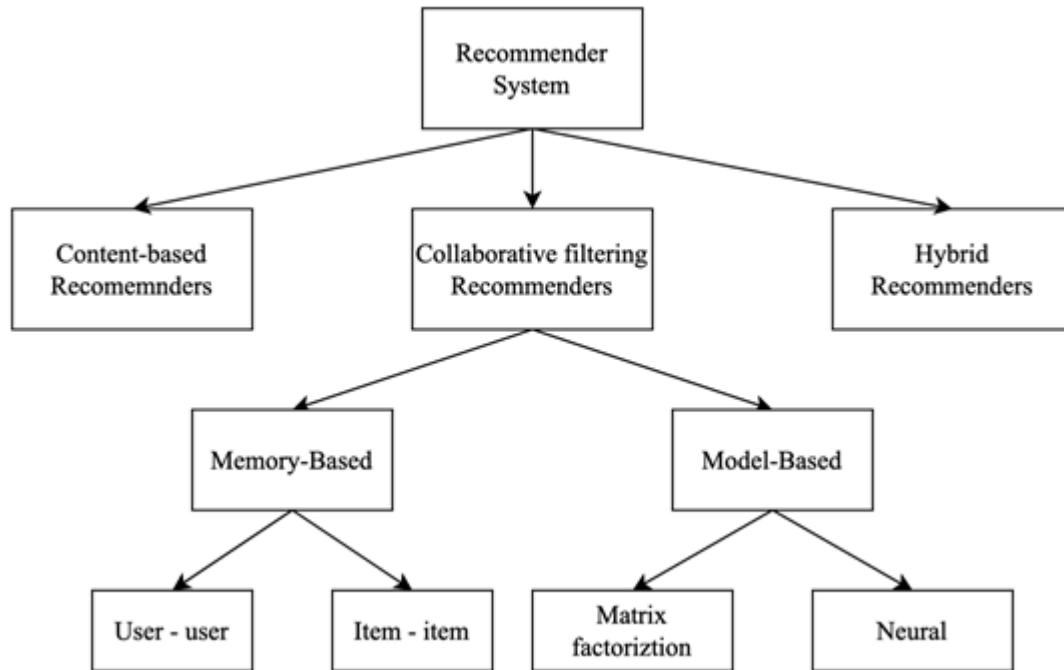
Ngoài ra, hệ thống gợi ý còn đóng vai trò quan trọng trong việc chuyển đổi khách hàng tiềm năng thành khách hàng thực sự. Bằng cách cung cấp các đề xuất chính xác, chương trình khuyến mãi hấp dẫn và trải nghiệm mua sắm độc đáo, hệ thống không chỉ thu hút khách hàng mới mà còn góp phần xây dựng mối quan hệ lâu dài. Điều này giúp tăng mức độ trung thành, giữ chân khách hàng hiện tại và tạo ra một tệp khách hàng bền vững cho doanh nghiệp.

Bên cạnh việc cải thiện trải nghiệm và tối ưu hóa doanh thu, hệ thống gợi ý còn là nguồn dữ liệu quý giá cho doanh nghiệp. Các thông tin về sở thích và hành vi người dùng được thu thập và phân tích để hỗ trợ các quyết định chiến lược, từ việc tối ưu hóa sản phẩm đến phát triển chiến lược tiếp thị hiệu quả hơn.

Trong bối cảnh người dùng phải đối mặt với tình trạng quá tải thông tin, hệ thống gợi ý hoạt động như một trợ lý thông minh, giúp họ nhanh chóng tìm thấy sản phẩm phù hợp mà không mất nhiều thời gian. Điều này không chỉ giúp giải quyết vấn đề thông tin dư thừa mà còn làm tăng hiệu quả tương tác giữa người dùng và hệ thống..

1.3.3. Các phương pháp xây dựng hệ thống gợi ý

Hệ thống gợi ý, hay còn gọi là Recommender System, là một giải pháp thông minh nhằm cung cấp những gợi ý phù hợp dựa trên dữ liệu về hành vi và sở thích của người dùng. Trong lĩnh vực này, có ba phương pháp xây dựng chính: lọc cộng tác (Collaborative Filtering), lọc nội dung (Content-Based Filtering), và phương pháp lai (Hybrid Method). Dưới đây là chi tiết từng phương pháp.



Hình 7: Sơ đồ biểu diễn các thuật toán chính trong hệ thống gợi ý.

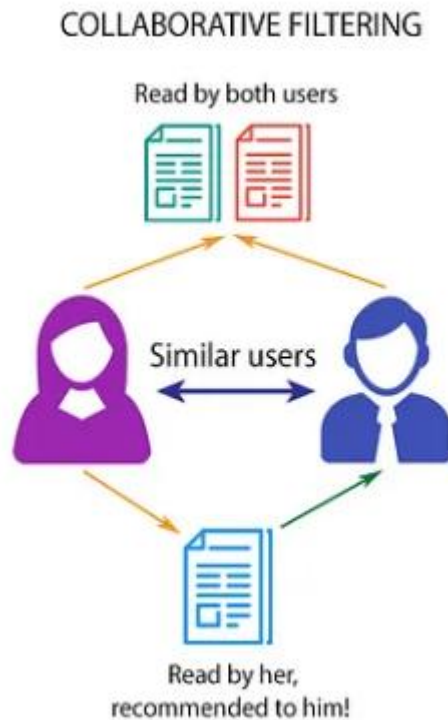
1.3.3.1. Lọc cộng tác (Collaborative Filtering)

Phương pháp lọc cộng tác (Collaborative Filtering) là một trong những cách tiếp cận phổ biến nhất để xây dựng hệ thống gợi ý. Dựa trên dữ liệu tương tác giữa người dùng và sản phẩm, phương pháp này nhằm tìm ra các mẫu tương đồng để đưa ra các gợi ý phù hợp. Có hai hướng tiếp cận chính trong lọc cộng tác, bao gồm lọc cộng tác dựa trên bộ nhớ (Memory-Based Collaborative Filtering) và lọc cộng tác dựa trên mô hình (Model-Based Collaborative Filtering).

Phương pháp lọc cộng tác dựa trên bộ nhớ khai thác trực tiếp dữ liệu tương tác lịch sử để tính toán sự tương đồng giữa người dùng hoặc giữa sản phẩm. Hướng tiếp cận này được chia thành hai dạng: lọc theo người dùng (User-Based Collaborative Filtering)

và lọc theo sản phẩm (Item-Based Collaborative Filtering). Với lọc theo người dùng, hệ thống tìm kiếm những người dùng có hành vi tương tự với người dùng hiện tại, từ đó gợi ý các sản phẩm mà những người này ưa thích. Trong khi đó, lọc theo sản phẩm tập trung xác định mối quan hệ giữa các sản phẩm, chẳng hạn như các sản phẩm thường được mua cùng nhau. Tuy nhiên, cách tiếp cận này có nhược điểm khi dữ liệu thưa thớt hoặc đối mặt với vấn đề "khởi tạo lạnh" (Cold-Start Problem), khi thông tin về người dùng mới hoặc sản phẩm mới còn hạn chế. Dù vậy, phương pháp này đơn giản, dễ triển khai và hiệu quả trong các hệ thống có quy mô dữ liệu vừa và nhỏ.

Ngược lại, phương pháp lọc cộng tác dựa trên mô hình sử dụng các thuật toán học máy để học từ dữ liệu và xây dựng mô hình dự đoán. Một số kỹ thuật phổ biến bao gồm phân rã ma trận (Matrix Factorization) như SVD (Singular Value Decomposition) hoặc ALS (Alternating Least Squares), giúp giảm kích thước ma trận người dùng-sản phẩm và phát hiện các yếu tố tiềm ẩn. Ngoài ra, mô hình yếu tố tiềm ẩn (Latent Factor Models) và các mô hình học sâu (Deep Learning Models) cũng được sử dụng để nắm bắt các mẫu phức tạp hơn trong dữ liệu. Phương pháp này mang lại độ chính xác cao hơn, đặc biệt trên các tập dữ liệu lớn và phức tạp, nhưng lại đòi hỏi tài nguyên tính toán lớn và thời gian huấn luyện đáng kể.



Hình 8: Sơ đồ minh họa quá trình lọc cộng tác để đưa ra gợi ý cá nhân hóa.

1.3.3.2. Lọc nội dung (Content-Based Filtering)

Phương pháp lọc nội dung (Content-Based Filtering) là một cách tiếp cận dựa trên việc phân tích đặc điểm và thông tin chi tiết của sản phẩm (Item Features) để đưa ra các gợi ý phù hợp với sở thích cá nhân của người dùng. Thay vì phụ thuộc vào dữ liệu tương tác giữa các người dùng, phương pháp này tập trung hoàn toàn vào dữ liệu của từng cá nhân và đặc điểm của sản phẩm mà họ quan tâm.

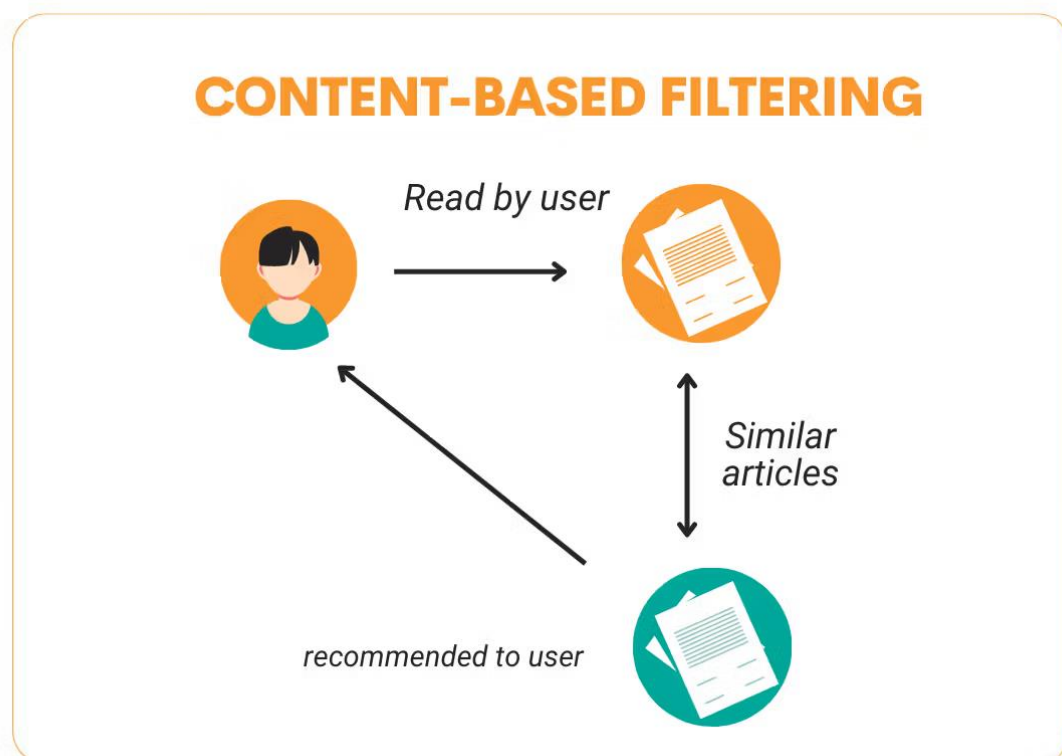
Cụ thể, hệ thống sẽ phân tích những sản phẩm mà người dùng đã yêu thích hoặc tương tác trước đó, từ đó xây dựng hồ sơ sở thích (User Profile) dựa trên các thuộc tính nổi bật của những sản phẩm này. Chẳng hạn, nếu một người dùng thường xuyên đánh giá cao hoặc mua các sản phẩm "điện thoại thông minh" với tính năng "chụp ảnh đẹp", hệ thống sẽ tự động gợi ý các sản phẩm có cùng đặc điểm, chẳng hạn như các mẫu điện thoại mới có chất lượng camera vượt trội.

Phương pháp này đặc biệt hiệu quả trong những trường hợp dữ liệu về sản phẩm được mô tả chi tiết và đầy đủ, ví dụ như danh mục sản phẩm bao gồm thông tin về giá, tính năng kỹ thuật, hoặc các mô tả chi tiết khác. Một ưu điểm nổi bật là nó có thể đưa ra các gợi ý mang tính cá nhân hóa cao, vì các sản phẩm được gợi ý hoàn toàn dựa trên sở

thích cụ thể của người dùng mà không bị ảnh hưởng bởi dữ liệu từ những người dùng khác.

Tuy nhiên, phương pháp lọc nội dung cũng gặp phải một số hạn chế. Một trong những vấn đề chính là chuyên môn hóa quá mức (Over-Specialization Problem), xảy ra khi hệ thống chỉ gợi ý những sản phẩm rất giống với những gì người dùng đã quan tâm trước đó, dẫn đến việc thiếu sự đa dạng trong các gợi ý. Ví dụ, nếu người dùng đã mua nhiều sách về "khoa học viễn tưởng", hệ thống có thể chỉ gợi ý thêm các sách cùng thể loại, mà không mở rộng sang các thể loại khác như "trinh thám" hay "lịch sử" mà người dùng có thể chưa từng khám phá.

Ngoài ra, phương pháp này cũng khó hoạt động hiệu quả với những sản phẩm hoặc danh mục không có đủ thông tin chi tiết về các thuộc tính. Trong những trường hợp như vậy, việc xây dựng các gợi ý phù hợp trở nên thách thức. Tuy nhiên, khi được áp dụng trong các lĩnh vực có dữ liệu rõ ràng và đầy đủ, lọc nội dung vẫn là một công cụ mạnh mẽ để tăng cường trải nghiệm cá nhân hóa cho người dùng.



Hình 9: Sơ đồ minh họa quá trình lọc nội dung để đưa ra gợi ý cá nhân hóa

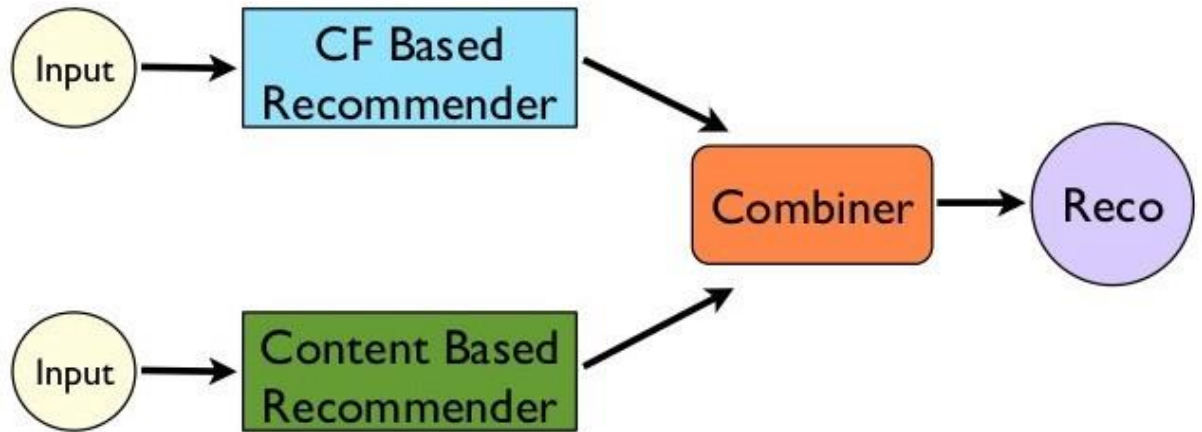
1.3.3.3. Phương pháp lai (Hybrid Method)

Phương pháp lai (Hybrid Method) kết hợp các kỹ thuật lọc cộng tác và lọc nội dung để tận dụng ưu điểm của cả hai phương pháp, nhằm tối ưu hóa kết quả gợi ý sản phẩm. Mục tiêu chính của phương pháp này là khắc phục những hạn chế của từng phương pháp riêng lẻ, như vấn đề "khởi tạo lạnh" (Cold-Start Problem) trong lọc cộng tác và "chuyên môn hóa quá mức" (Over-Specialization) trong lọc nội dung. Bằng cách kết hợp, phương pháp lai giúp tăng cường độ chính xác trong việc dự đoán sở thích của người dùng và nâng cao khả năng cá nhân hóa, từ đó mang đến những gợi ý phù hợp và thú vị hơn.

Có nhiều cách kết hợp các phương pháp lọc trong hệ thống lai. Một trong những cách phổ biến là kết hợp tuần tự (Sequential Hybrid), trong đó kết quả của một phương pháp sẽ làm đầu vào cho phương pháp còn lại. Ví dụ, kết quả từ lọc nội dung có thể được sử dụng để bổ sung cho lọc cộng tác. Một phương pháp kết hợp khác là kết hợp song song (Parallel Hybrid), trong đó cả hai phương pháp lọc được áp dụng đồng thời và kết quả cuối cùng được hợp nhất để đưa ra gợi ý. Ngoài ra, còn có phương pháp kết hợp dựa trên trọng số (Weighted Hybrid), trong đó mỗi phương pháp được gán một trọng số tùy theo hiệu quả của nó đối với tập dữ liệu cụ thể, giúp tăng cường sự linh hoạt và hiệu quả của hệ thống.

Phương pháp lai không chỉ giúp giải quyết các vấn đề như khởi tạo lạnh và chuyên môn hóa quá mức, mà còn cung cấp một giải pháp linh hoạt, cho phép hệ thống hoạt động hiệu quả hơn trong các tình huống khác nhau. Các nền tảng lớn như Netflix và YouTube đã áp dụng phương pháp lai để tối ưu hóa các gợi ý cho người dùng, cải thiện trải nghiệm và tăng mức độ hài lòng của người dùng và YouTube đã áp dụng phương pháp lai để tối ưu hóa các gợi ý cho người dùng, cải thiện trải nghiệm và tăng mức độ hài lòng của người dùng.

Hybrid Recommendations



Hình 10: Sơ đồ Phương pháp Gợi ý Lai (Hybrid Recommendations)

1.3.3.4. Các kỹ thuật hiện đại trong xây dựng hệ thống gợi ý

Ngoài các phương pháp truyền thống, sự phát triển của công nghệ đã đưa vào các kỹ thuật tiên tiến như:

- Học sâu (Deep Learning): Sử dụng mạng nơ-ron sâu (Deep Neural Networks) để khám phá các mẫu phức tạp trong dữ liệu lớn, từ đó cải thiện chất lượng gợi ý.
- Xử lý ngôn ngữ tự nhiên (NLP): Kết hợp phân tích cảm xúc (SA) từ đánh giá của người dùng để tăng độ chính xác của gợi ý.
- Đặc biệt, việc ứng dụng PhoBERT – mô hình ngôn ngữ tối ưu dành cho tiếng Việt – đã giúp nâng cao khả năng phân tích dữ liệu văn bản tiếng Việt, từ đó cải thiện đáng kể hiệu suất của hệ thống gợi ý trên các nền tảng Việt Nam.

1.4. Lý thuyết các mô hình học máy

1.4.1. Mô hình Logistic Regression

Hồi quy Logistic là một mô hình thống kê được sử dụng để phân loại nhị phân, tức là dự đoán một đối tượng thuộc vào một trong hai nhóm. Mô hình này hoạt động dựa trên hàm sigmoid, giúp chuyển đổi đầu vào thành xác suất thuộc một trong hai lớp nhị phân. Hàm sigmoid được biểu diễn như sau:

$$S(z) = \frac{1}{1 + e^{-z}}$$

Khi áp dụng vào Hồi quy Logistic, đầu vào là ma trận dữ liệu (X) và trọng số (w), đầu ra là xác suất dự đoán cho một lớp nhị phân. Mục tiêu của quá trình huấn luyện là tìm ra bộ trọng số (w) sao cho đầu ra của hàm sigmoid gần với kết quả thực tế nhất.

Để tối ưu hóa mô hình, ta sử dụng hàm mất mát (Loss Function) để đánh giá hiệu suất dự đoán. Một trong những hàm mất mát phổ biến là hàm Cross-Entropy (hay còn gọi là Log Loss), giúp đo lường mức độ lỗi của mô hình khi dự đoán [15].

Hàm mất mát Cross-Entropy được định nghĩa như sau:

$$L = -\frac{1}{N} \sum_{i=1}^N (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

Trong đó:

- (y_i) là nhãn thực tế (0 hoặc 1),
- (p_i) là xác suất dự đoán của mô hình.

Đối với bài toán phân loại cảm xúc có nhiều hơn hai nhãn như trong đề tài này (gồm 4 nhãn: *None*, *Positive*, *Neutral*, *Negative*), mô hình Logistic Regression cần được mở rộng để xử lý nhiều lớp. Có hai phương pháp phổ biến là One-vs-Rest (OvR): huấn luyện một mô hình phân biệt mỗi lớp với phần còn lại và Softmax Regression (Multinomial

Logistic Regression): mở rộng trực tiếp bằng cách thay thế hàm sigmoid bằng hàm softmax, đảm bảo tổng xác suất các lớp bằng 1.

Hàm Softmax:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

$$\sum_{i=1}^K \text{softmax}(z_i) = 1$$

Áp dụng vào bài toán phân loại cảm xúc đa nhiệm. Trong đề tài này, Logistic Regression được sử dụng để dự đoán cảm xúc người dùng đối với 3 khía cạnh trong một bình luận: Price, Quality, và Service, mỗi khía cạnh có 4 nhãn cảm xúc: None, Positive, Neutral, Negative.

Theo quy trình, sau khi tiền xử lý văn bản (Các bình luận được chuẩn hóa và biến đổi thành dạng vector bằng kỹ thuật TF-IDF); trong bước huấn luyện, để đánh giá sai số giữa đầu ra mô hình và nhãn thực tế trong quá trình huấn luyện, ta sử dụng hàm mất mát Categorical Cross-Entropy, được định nghĩa như sau:

$$\mathcal{L} = - \sum_{i=1}^N \sum_{k=0}^3 y_{ik} \log(\hat{p}_{ik})$$

Trong đó:

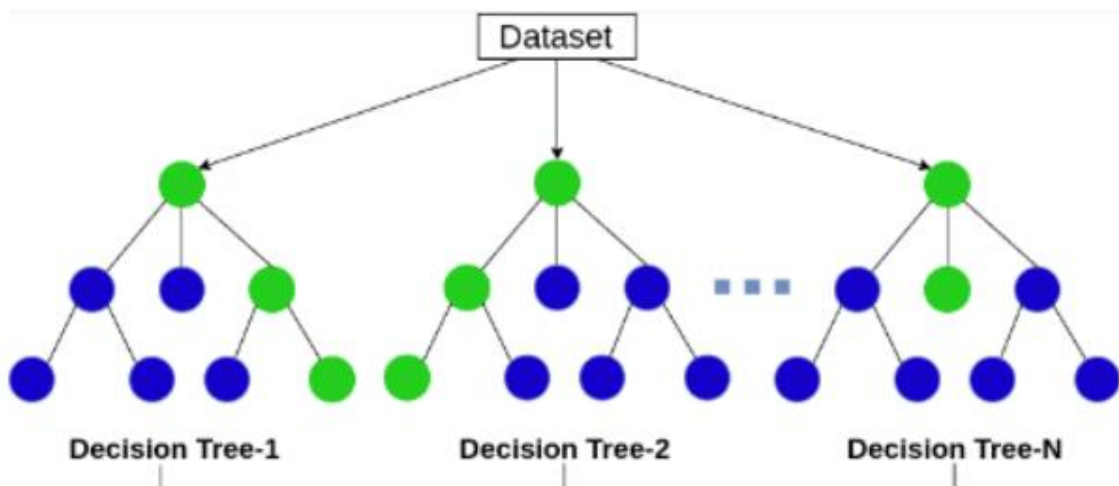
- y_{ik} là nhãn thực tế (one-hot) cho mẫu i thuộc lớp k .
- \hat{p}_{ik} là xác suất mô hình dự đoán mẫu i thuộc lớp k .
- N là số lượng mẫu trong tập huấn luyện.
- Các lớp $k = 0, 1, 2, 3$ tương ứng với: *None, Positive, Neutral, Negative*.

Với mỗi khía cạnh, mô hình trả ra xác suất cho 4 lớp và lớp có xác suất cao nhất là kết quả.

1.4.2. Mô hình Random Forest Classifier

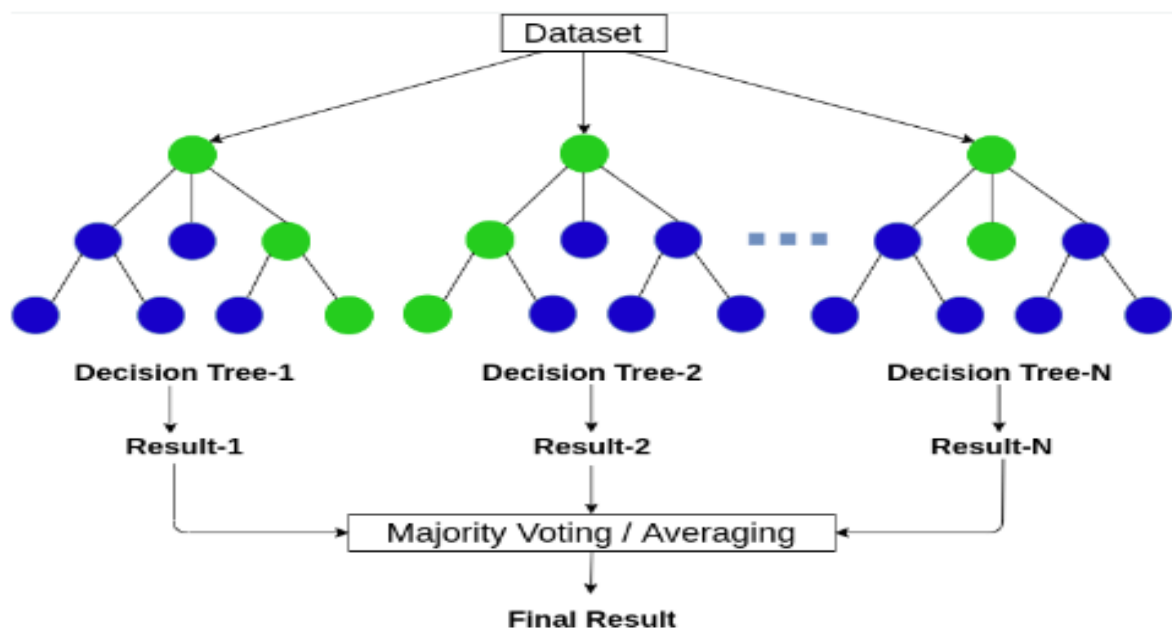
Thuật toán Random Forest hoạt động bằng cách tạo ra nhiều cây quyết định (Decision Tree), giống như một khu rừng có nhiều cây. Từ “Random” (ngẫu nhiên) có nghĩa là mỗi cây sẽ được huấn luyện trên một phần dữ liệu khác nhau và chọn ngẫu nhiên một số đặc trưng (feature) để học, nên các cây sẽ không giống nhau. Khi cần dự đoán, các cây sẽ đưa ra ý kiến riêng của mình, và thuật toán sẽ tổng hợp lại các ý kiến đó. Ví dụ như lấy ý kiến đa số trong bài toán phân loại. Nhờ cách làm này, kết quả cuối cùng thường chính xác hơn và ít bị sai lệch do một cây duy nhất gây ra.

Ở bước huấn luyện thì chúng ta sẽ xây dựng nhiều cây quyết định, các cây quyết định có thể khác nhau:



Hình 11: Cách xây dựng các cây quyết định.

Sau đó ở bước dự đoán, với một dữ liệu mới, thì ở mỗi cây quyết định chúng ta sẽ đi từ trên xuống theo các node điều kiện để được các dự đoán, sau đó kết quả cuối cùng được tổng hợp từ kết quả của các cây quyết định.



Hình 12: Kết quả tổng hợp từ các cây quyết định.

Cách hoạt động của Random Forest Classifier:

Giả sử bộ dữ liệu có n dữ liệu (sample) và mỗi dữ liệu có d thuộc tính (feature).

Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kỹ thuật Bootstrapping, hay còn gọi là random sampling with replacement. Tức khi lấy sample được 1 dữ liệu thì sẽ không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kỹ thuật này thì tập n dữ liệu mới có thể có những dữ liệu bị trùng nhau.

1. Sau khi sample được n dữ liệu từ bước 1 thì sẽ chọn ngẫu nhiên ở k thuộc tính ($k < n$). Sau đó sẽ có được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.
2. Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính

khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

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

Ví dụ: Đầu tiên ta sẽ trích xuất đặc trưng từ văn bản bằng phương pháp TF-IDF. Sau đó mỗi đánh giá trở thành một vector đặc trưng (feature vector).

Tiếp theo, ta xây dựng mô hình Random Forest để xử lý bài toán phân loại đa nhãn với ba nhãn đầu ra: Price, Quality, và Service. Mỗi cây trong rừng sẽ là một cây quyết định thực hiện phân chia dữ liệu dựa trên các đặc trưng đầu vào với mục tiêu giảm độ bất thuần (Impurity).

Công thức:

Gini Impurity (Gini): $Gini(S) = 1 - \sum_{i=1}^n (p_i)^2$

Trong đó:

- p_i : xác suất của lớp i.
- n : Số lớp cảm xúc (ở đây là 4 lớp: none, positive, neutral, negative).

Giả sử tại một nút có 3 mẫu: 1 mẫu positive, 1 mẫu neutral, 1 mẫu none thì

$$Gini(S) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{1}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 0.666$$

Chọn đặc trưng phân chia tốt nhất:

- Thông tin thu được (Information Gain):

$$IG = Entropy(T) - \sum_{v \in A} \frac{|T_v|}{|T|} \cdot Entropy(T_v)$$

Trong đó:

- T: Tập dữ liệu.
- A: Thuộc tính đang xem xét.
- T_v : Tập dữ liệu con tương ứng với giá trị v của thuộc tính A.

Cuối cùng ta tạo Random Forest để xây dựng nhiều cây quyết định với các tập dữ liệu con và chọn ngẫu nhiên một tập con các đặc trưng để giảm overfitting. Ta sẽ lấy số phiếu bầu từ các cây (Majority Voting):

$$y_{predict} = mode(y_{tree1}, y_{tree2}, \dots, y_{treeM})$$

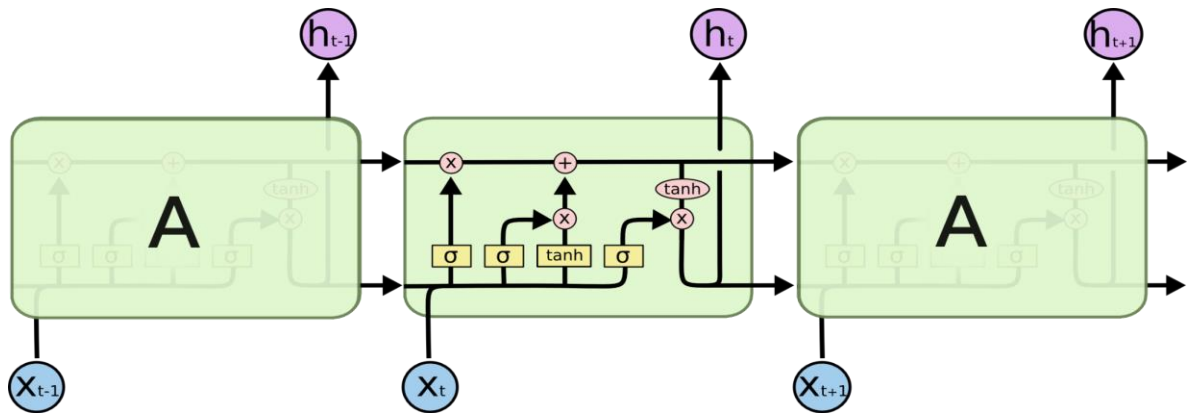
Với M là số cây trong rừng và y là các nhãn đầu ra cho mỗi nhãn là price, quality, service. Sau đó dựa vào số cây đồng thuận thì đưa ra được kết quả phân tích cảm xúc để tăng độ chính xác.

1.4.3. Mô hình Long Short Term Memory

Long short-term memory (LSTM) là một kiến trúc đặc biệt của RNN có khả năng học được sự phụ thuộc trong dài hạn (long-term dependencies) được giới thiệu bởi Hochreiter & Schmidhuber (1997) [13]. Kiến trúc này đã được phổ biến và sử dụng rộng rãi cho tới ngày nay. LSTM đã tỏ ra khắc phục được rất nhiều những hạn chế của RNN trước đây về triệt tiêu đạo hàm. Tuy nhiên cấu trúc của chúng có phần phức tạp hơn mặc dù vẫn giữ được tư tưởng chính của RNN là sao chép các kiến trúc theo dạng chuỗi.

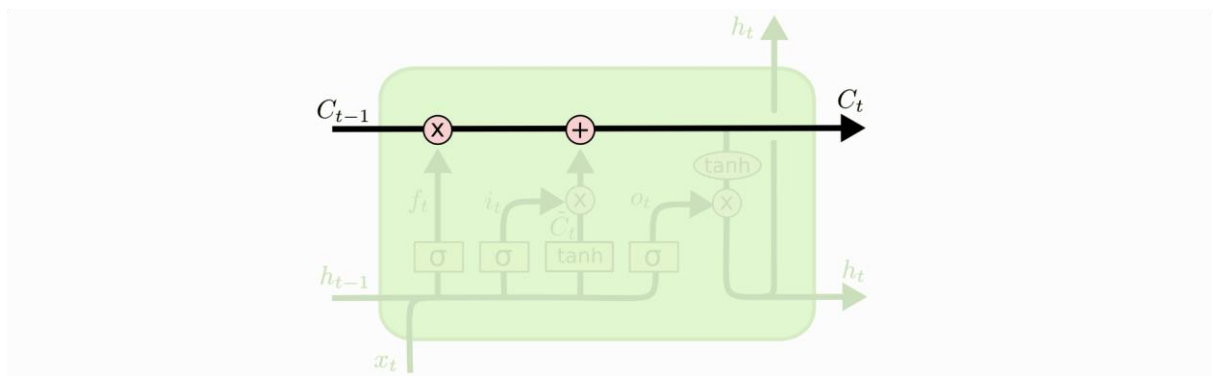
Một đơn vị **LSTM** thường được cấu thành từ một cell và ba gates:

- Input gate
- Output gate
- Forget gate.



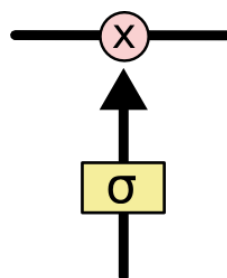
Hình 13: Sơ đồ kiến trúc module trong mạng LSTM chứa 4 tầng ẩn.

- Ô trạng thái là một dạng băng chuyền chạy thẳng xuyên suốt toàn bộ chuỗi với chỉ một vài tương tác tuyến tính nhỏ giúp cho thông tin có thể truyền dọc theo đồ thị mạng nơ ron ổn định.



Hình 14: Đường đi của ô trạng thái (cell state) trong mạng LSTM.

- LSTM có khả năng xóa và thêm thông tin vào ô trạng thái và điều chỉnh các luồng thông tin này thông qua các cấu trúc gọi là cổng.



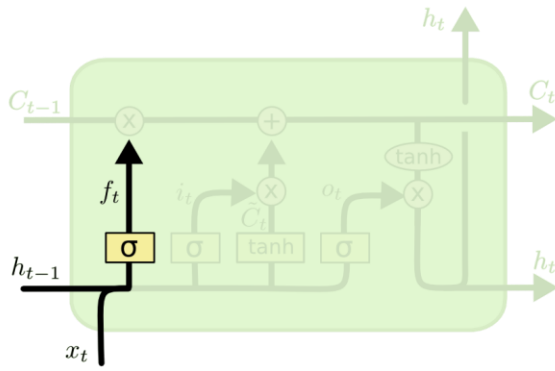
Hình 15: Một cổng của hàm sigmoid trong LSTM.

- Cổng là cơ chế đặc biệt để điều chỉnh luồng thông tin đi qua. Chúng được tổng hợp bởi một tầng ẩn của hàm activation sigmoid và với một toán tử nhân như đồ thị.
- Hàm sigmoid sẽ cho đầu ra là một giá trị xác suất nằm trong khoảng từ 0 đến 1, thể hiện rằng có bao nhiêu phần thông tin sẽ đi qua cổng. Giá trị bằng 0 ngụ ý rằng không cho phép thông tin nào đi qua, giá trị bằng 1 sẽ cho toàn bộ thông tin đi qua.

Thứ tự các bước thực hiện của LSTM :

- Cổng là cơ chế đặc biệt để điều chỉnh luồng thông tin đi qua. Chúng được tổng hợp bởi một tầng ẩn của hàm activation sigmoid và với một toán tử nhân như đồ thị.
- Bước đầu tiên trong LSTM sẽ quyết định xem thông tin nào chúng ta sẽ cho phép đi qua ô trạng thái (cell state). Nó được kiểm soát bởi hàm sigmoid trong một tầng gọi là tầng quên (forget gate layer). Đầu tiên nó nhận đầu vào là 2 giá trị h_{t-1} và x_t và trả về một giá trị nằm trong khoảng 0 và 1 cho mỗi giá trị của ô trạng thái C_{t-1} . Nếu giá trị bằng 1 thể hiện ‘giữ toàn bộ thông tin’ và bằng 0 thể hiện ‘bỏ qua toàn bộ chúng’.

Ví dụ : "Ban đầu tôi thấy sản phẩm rất tốt, nhưng sau một tuần sử dụng, tôi phát hiện nhiều lỗi." Ban đầu, từ "rất tốt" biểu thị cảm xúc tích cực, được ghi nhớ trong trạng thái của mô hình. Tuy nhiên, khi từ "nhưng" xuất hiện, nó báo hiệu một sự thay đổi ngữ cảnh. Lúc này, tầng quên sẽ giảm trọng số của thông tin tích cực trước đó để chuẩn bị cho dữ liệu cảm xúc mới, chẳng hạn như "nhiều lỗi". Cuối cùng, trạng thái bộ nhớ sẽ phản ánh cảm xúc tổng thể là tiêu cực, nhờ việc tầng quên loại bỏ thông tin không còn phù hợp và tập trung vào những phần dữ liệu quan trọng mới.

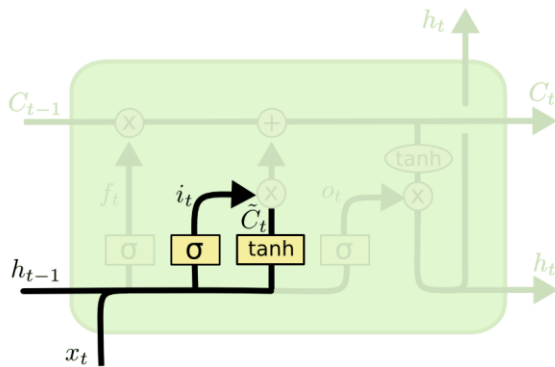


$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Hình 16: Tầng cổng quên (forget gate layer).

Bước tiếp theo chúng ta sẽ quyết định loại thông tin nào sẽ được lưu trữ trong ô trạng thái. Bước này bao gồm 2 phần. Phần đầu tiên là một tầng ẩn của hàm sigmoid được gọi là tầng cổng vào (input gate layer) quyết định giá trị bao nhiêu sẽ được cập nhật. Tiếp theo, tầng ẩn hàm tanh sẽ tạo ra một vector của một giá trị trạng thái mới \underline{c}_t mà có thể được thêm vào trạng thái. Tiếp theo kết hợp kết quả của 2 tầng này để tạo thành một cập nhật cho trạng thái.

Trong bài toán phân tích cảm xúc, tương tự như việc thay thế loại chủ ngữ trong ví dụ của mô hình ngôn ngữ, chúng ta cũng cần thay thế cảm xúc cũ trong trạng thái khi có một cảm xúc mới quan trọng hơn xuất hiện.



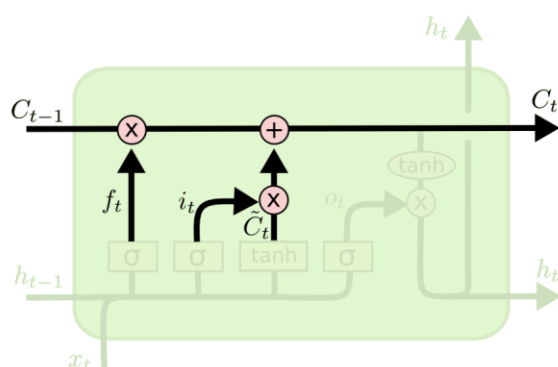
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

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

Hình 17: Cập nhật giá trị cho ô trạng thái bằng cách kết hợp 2 kết quả từ tầng cổng vào và tầng ẩn.

Đây là thời điểm để cập nhật một ô trạng thái cũ, C_{t-1} sang một trạng thái mới C_t . Những bước trước đó đã quyết định làm cái gì, và tại bước này chỉ cần thực hiện nó.

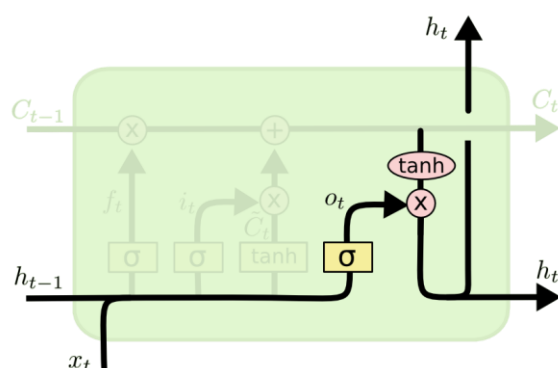
Chúng ta nhân trạng thái cũ với f_t tương ứng với việc quên những thứ quyết định được phép quên sớm. Phần tử đề cử $i_t * \underline{c}_t$ là một giá trị mới được tính toán tương ứng với bao nhiêu được cập nhật vào mỗi giá trị trạng thái.



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

Hình 18: Ô trạng thái mới.

Cuối cùng cần quyết định xem đầu ra sẽ trả về bao nhiêu. Kết quả ở đầu ra sẽ dựa trên ô trạng thái, nhưng sẽ là một phiên bản được lọc. Đầu tiên, chúng ta chạy qua một tầng sigmoid nơi quyết định phần nào của ô trạng thái sẽ ở đầu ra. Sau đó, ô trạng thái được đưa qua hàm tanh (để chuyển giá trị về khoảng -1 và 1) và nhân nó với đầu ra của một cổng sigmoid, do đó chỉ trả ra phần mà chúng ta quyết định.



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

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

Hình 19: Điều chỉnh thông tin ở đầu ra thông qua hàm tanh.

Ví dụ :

Đầu vào từng bước: "Dịch vụ" → "rất" → "tốt" → "nhưng" → "sản phẩm" → "thì" → "kém".

Cổng quên: Loại bỏ thông tin không quan trọng như từ "rất", "nhưng", "thì".

Cổng nhập: Ghi nhớ từ "tốt" (cảm xúc tích cực) và "kém" (cảm xúc tiêu cực).

Cổng đầu ra: Kết hợp thông tin để đưa ra dự đoán dựa trên trọng số ưu tiên cho từ "kém".

1.4.4. Mô hình PhoBERT

PhoBERT là mô hình học sâu dựa trên kiến trúc BERT, được phát triển đặc biệt cho tiếng Việt. Mô hình này được thiết kế để giải quyết các bài toán trong lĩnh vực Xử lý ngôn ngữ tự nhiên (NLP) cho tiếng Việt, chẳng hạn như phân tích cảm xúc, phân loại văn bản, nhận diện thực thể, và các tác vụ NLP khác. PhoBERT giúp cải thiện hiệu suất các ứng dụng NLP nhờ vào khả năng hiểu sâu về ngữ nghĩa và ngữ cảnh của ngôn ngữ tiếng Việt [14].

Kiến trúc của PhoBERT:

PhoBERT hoạt động dựa trên kiến trúc Transformer, vốn nổi bật với khả năng xử lý văn bản theo ngữ cảnh. Thay vì chỉ phân tích mỗi từ một cách độc lập, PhoBERT sử dụng phương pháp mã hóa hai chiều (bidirectional encoding), giúp mô hình nắm bắt được mối quan hệ giữa các từ trong câu. Cách tiếp cận này giúp PhoBERT hiểu rõ hơn về ý nghĩa của từng từ trong ngữ cảnh của câu và đoạn văn, từ đó cải thiện độ chính xác trong việc xử lý các bài toán ngôn ngữ.

PhoBERT được huấn luyện trên một tập dữ liệu lớn tiếng Việt thông qua phương pháp unsupervised learning (học không giám sát). Quá trình này giúp mô hình học được các mối quan hệ ngữ nghĩa giữa các từ và cấu trúc ngữ pháp của tiếng Việt mà không cần nhãn. Sau khi hoàn thành quá trình huấn luyện, PhoBERT có thể áp dụng vào nhiều bài toán NLP khác nhau, đặc biệt trong ngữ cảnh tiếng Việt.

Tại sao cần Fine-tuning PhoBERT ?

Fine-tuning PhoBERT là quá trình điều chỉnh mô hình PhoBERT đã được huấn luyện sẵn trên dữ liệu tiếng Việt để nó có thể thực hiện một bài toán cụ thể, chẳng hạn như phân tích cảm xúc. Dù PhoBERT đã học được cách biểu diễn văn bản và hiểu ngữ nghĩa tiếng Việt, nhưng nó cần được tinh chỉnh để có thể dự đoán chính xác nhãn (như POSITIVE, NEUTRAL, NEGATIVE) cho từng văn bản. Quá trình fine-tune giúp mô hình học cách tối ưu hóa các trọng số của nó sao cho phù hợp với tập dữ liệu đã được gán nhãn, từ đó cải thiện độ chính xác trong việc phân loại cảm xúc.

Quá trình Fine-tuning PhoBERT:

- Chọn mô hình PhoBERT đã được huấn luyện trước: PhoBERT đã được huấn luyện trên một tập dữ liệu tiếng việt lớn và bao quát, vì vậy có thể sử dụng mô hình này làm nền tảng cho các tác vụ NLP khác.

- Chuẩn bị dữ liệu đặc thù: Trước khi fine-tune, cần chuẩn bị dữ liệu đầu vào phù hợp với mục tiêu phân tích cảm xúc theo khía cạnh. Cụ thể, mỗi bình luận sẽ được gán nhãn cảm xúc riêng biệt cho từng khía cạnh gồm: **Service Sentiment**(Cảm xúc về dịch vụ), **Price Sentiment**(Cảm xúc về giá cả), **Quality Sentiment**(Cảm xúc về chất lượng). Mỗi khía cạnh có thể mang một trong bốn nhãn: None, Positive, Neutral, hoặc Negative. Dữ liệu cũng được token hóa và chuẩn hóa trước khi đưa vào mô hình.

- Thiết kế mô hình tinh chỉnh: Quá trình fine-tuning sẽ huấn luyện lại PhoBERT với tập dữ liệu có nhãn. Để tối ưu hóa hiệu quả của PhoBERT trong bài toán multi-task, mô hình được mở rộng với các kỹ thuật quan trọng: **Attention Pooling**: Là cơ chế giúp mô hình học trọng số cho từng token trong câu, nhằm tập trung vào các phần quan trọng. Mục đích là nâng cao độ chính xác trong việc trích xuất đặc trưng ngữ nghĩa, tốt hơn so với các phương pháp pooling truyền thống như mean hay CLS token; **Adapter Layer**: Là một mạng con nhẹ được chèn vào giữa các tầng transformer, cho phép mô hình học thêm thông tin mới mà không làm thay đổi trọng số gốc. Mục đích là tiết kiệm tài nguyên và giảm nguy cơ overfitting, đặc biệt hữu ích khi dữ liệu huấn luyện có quy mô nhỏ; **Multi-Task Learning**: Là kỹ thuật huấn luyện mô hình để dự đoán nhiều tác vụ cùng lúc – ở đây là 3 khía cạnh cảm xúc (giá, chất lượng, dịch vụ), thay vì phải tách ra làm 3 mô hình riêng biệt. Điều này giúp chia sẻ đặc trưng và tăng tính khái quát cho mô hình.

- Đánh giá và tối ưu hóa: Sau khi fine-tune, mô hình được đánh giá trên tập dữ liệu kiểm tra theo từng khía cạnh. Các chỉ số như **Macro F1-score**, **Precision**, **Recall** được sử dụng để đo lường hiệu suất mô hình trong việc nhận diện cảm xúc ở từng khía cạnh. Từ đó, có thể phát hiện ra các vấn đề như mất cân bằng nhãn (ví dụ: nhãn "None" xuất hiện quá nhiều) hoặc mô hình học chưa tốt ở khía cạnh dịch vụ, từ đó điều chỉnh lại dữ liệu hoặc cấu hình huấn luyện..

Cách hoạt động của PhoBERT trong quá trình Fine-tuning

Trong quá trình fine-tuning, PhoBERT sẽ được áp dụng vào bài toán cụ thể và điều chỉnh các trọng số sao cho phù hợp với dữ liệu của bài toán. PhoBERT có khả năng sử dụng các trọng số đã học trong quá trình huấn luyện ban đầu và điều chỉnh chúng để đáp ứng các yêu cầu của tác vụ mới.

Chẳng hạn, trong bài toán phân tích cảm xúc, PhoBERT sẽ học cách phân biệt các cảm xúc khác nhau như tích cực, tiêu cực, hay trung tính dựa trên ngữ cảnh trong câu. Mô hình sẽ sử dụng thông tin từ các từ xung quanh để xác định cảm xúc của người viết. Đặc biệt, PhoBERT có thể nắm bắt được các ngữ cảnh phức tạp trong tiếng Việt, như các từ đa nghĩa hay cấu trúc câu đặc trưng, điều này giúp cải thiện khả năng phân tích cảm xúc chính xác hơn.

Ưu điểm của PhoBERT khi Fine-tuning

Khả năng xử lý ngữ nghĩa tốt: PhoBERT có thể hiểu được ngữ cảnh của từ và cụm từ trong câu, giúp cải thiện độ chính xác trong các tác vụ phân tích cảm xúc và các bài toán NLP khác.

Tính linh hoạt cao: PhoBERT có thể dễ dàng được fine-tune cho nhiều tác vụ khác nhau mà không cần phải huấn luyện lại từ đầu. Điều này giúp tiết kiệm thời gian và tài nguyên.

Hiệu quả cao trong ngữ cảnh tiếng Việt: Mô hình PhoBERT được huấn luyện đặc biệt cho tiếng Việt, giúp mô hình hiểu sâu về các đặc điểm ngữ pháp và ngữ nghĩa của ngôn ngữ này. Điều này mang lại lợi thế lớn trong việc xử lý các tác vụ NLP tiếng Việt.

Nhờ vào quá trình fine-tuning, PhoBERT có thể đạt được hiệu quả cao trong các bài toán xử lý ngôn ngữ tự nhiên, đặc biệt là trong các ứng dụng tiếng Việt, giúp nâng cao chất lượng của các hệ thống tự động như phân tích cảm xúc, phân loại văn bản, và gợi ý sản phẩm.

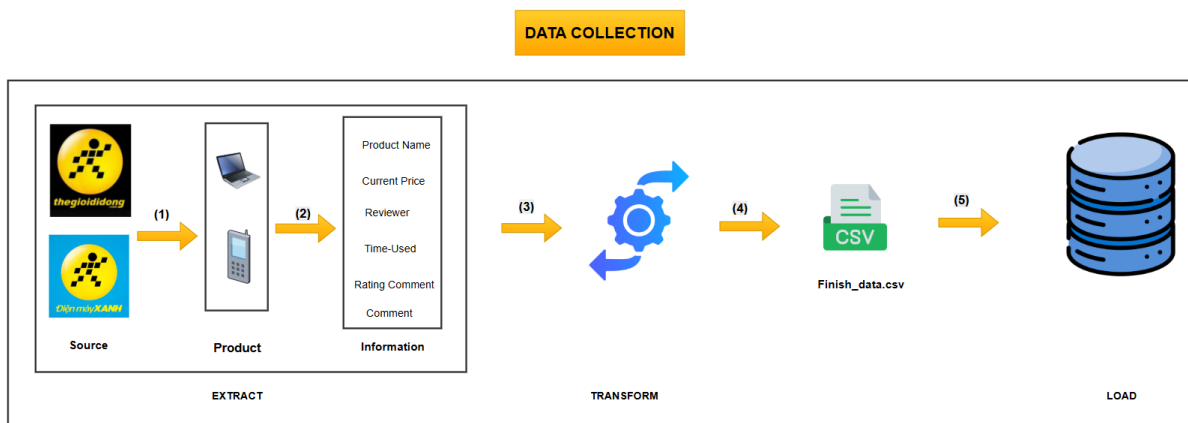
Nhược điểm của PhoBERT khi Fine-tuning

Giống như bất kỳ mô hình AI nào, PhoBERT có thể phản ánh các thiên lệch có trong dữ liệu huấn luyện, dẫn đến kết quả không công bằng hoặc phân biệt đối xử trong các ứng dụng yêu cầu sự công bằng và minh bạch.

Việc sử dụng hiệu quả PhoBERT đòi hỏi tài nguyên tính toán đáng kể và chuyên môn kỹ thuật, điều này có thể là rào cản đối với những người không có quyền truy cập vào phần cứng mạnh mẽ hoặc kiến thức chuyên môn.

CHƯƠNG 2: XÂY DỰNG HỆ THỐNG VÀ GIẢI PHÁP

2.1. Mô tả xây dựng kiến trúc hệ thống



Hình 20: Sơ đồ mô tả quy trình thu thập dữ liệu.

- Đầu tiên, Extract :

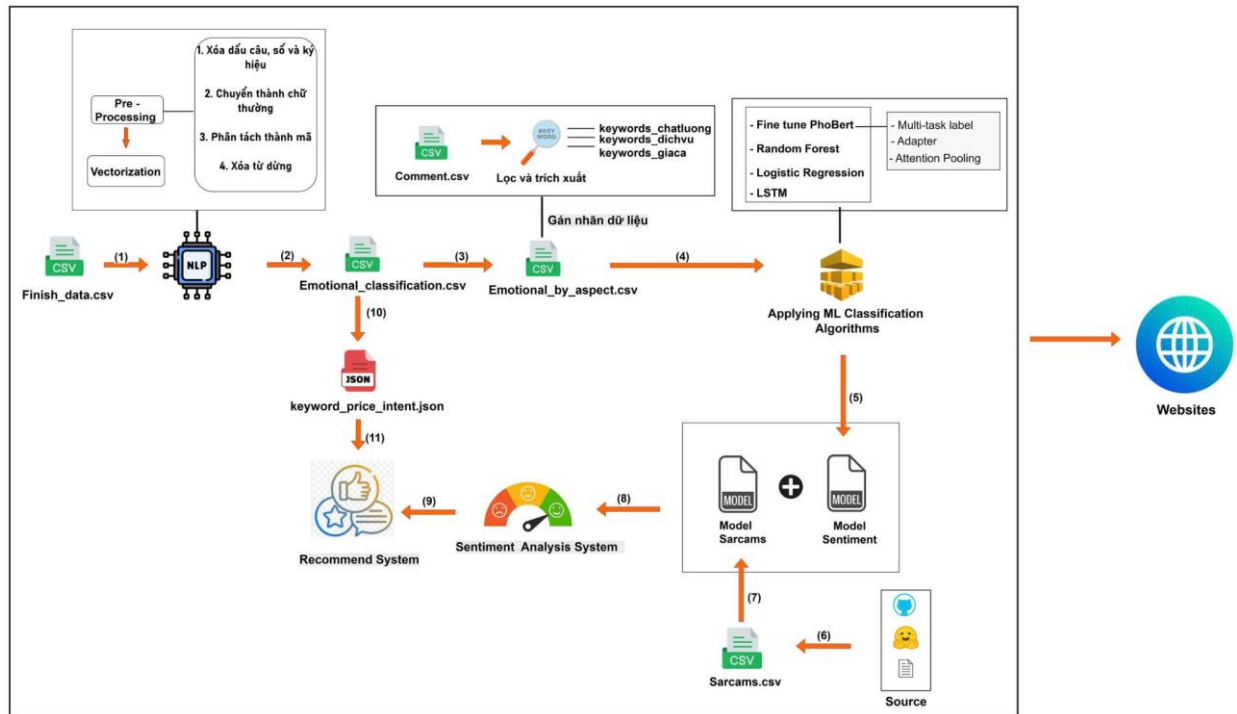
- (1) : Thực hiện thu thập dữ liệu từ trang Websites thương mại điện tử của Thế Giới Di Động, Điện Máy Xanh các thông tin liên quan của tất cả thiết bị laptop và điện thoại.
- (2) : Duyệt qua từng sản phẩm và trích xuất các thông tin là tên sản phẩm, giá, tên tài khoản đánh giá, bình luận của người dùng, thời gian sử dụng của người dùng, chỉ số đánh giá theo định dạng file là csv.

- Transform :

- (3) : Thực hiện chuyển đổi hay xử lý file csv vừa lấy từ quá trình extract trên bao gồm việc làm sạch dữ liệu như kiểm tra giá trị null, xử lý giá trị bị thiếu, chuyển đổi thời gian sử dụng sang số ngày, tính toán và thêm cột rating trung bình cho từng sản phẩm, hợp nhất dữ liệu từ hai trang web.
- (4) : Sau khi xử lý, dữ liệu sẽ hợp nhất thành file có tên là Finish_data.csv

- Load :

- (5) : Sau khi dữ liệu đã được chuẩn bị và chuẩn hóa, nó sẽ được tải vào cơ sở dữ liệu SQL Server với các thông số tương ứng từ file trên.



Hình 21: Sơ đồ quy trình thực hiện đề tài.

(1) : Dữ liệu sau khi đã được thu thập như cách mô tả ở trên sẽ được đưa vào xử lý ngôn ngữ tự nhiên. Xử lý ngôn ngữ tự nhiên: Dữ liệu được thu thập được xử lý qua các bước của xử lý ngôn ngữ tự nhiên gồm:

- Tiền xử lý : Dữ liệu văn bản được làm sạch và chuẩn hóa với các bước:

- + Loại bỏ dấu câu, số và ký hiệu.
- + Chuyển đổi văn bản thành chữ thường.
- + Tách từ thông qua các đoạn mã .
- + Xóa từ dừng.

- Biểu diễn dữ liệu : Chuyển dữ liệu văn bản thành dạng số thông qua phương pháp TF-IDF.

(2) : Sau khi thực hiện xử lý ngôn tự nhiên xong thì nó sẽ được lưu thành file `Emotional_classification.csv` với các cột comment đã được xử lý.

(3) : Từ file `Emotional_classification.csv` vừa được xử lý ở trên ta tiếp tục sẽ thực hiện rút gọn chỉ còn cột comment và lưu vào `Comment.csv` để thực hiện lọc và trích xuất ra 3 file txt có tên là `Keywords_chatluong`, `Keywords_giaca`,

- Keywords_dichvu và tiến hành gán nhãn dữ liệu bằng mô hình phobert để thành file đã được gán nhãn cảm xúc cho từng loại có tên là Emotional_by_aspect.csv
- (4) : Áp dụng tập dữ liệu Emotional_by_aspect.csv để train các mô hình. Các mô hình thử nghiệm bao gồm 4 mô hình là PhoBert Fine tune, Random Forest, Logistic Regression và LSTM. Trong đó, các kỹ thuật áp dụng cho việc fine tune PhoBert là thay đổi trích xuất đặc trưng và phân lớp so với mô hình gốc. Cụ thể ở bước trích xuất đặc trưng, áp dụng Attention Pooling để giúp mô hình tập trung vào các phần quan trọng trong bình luận (ví dụ: "giá", "dịch vụ", "tệ", "quá đắt"...) và Adapter để học thêm đặc trưng mới nhưng giữ nguyên trọng số gốc của PhoBERT giúp quá trình fine-tune nhanh, nhẹ hơn. Về phần phân lớp, phân lớp được thay đổi bằng multi-task learning(học đa nhiệm) với 3 nhãn cùng lúc là Price Sentiment, Quality Sentiment, Service Sentiment, mỗi nhãn có một classifier riêng biệt và tương tự cho các model Random Forest, Logistic Regression và LSTM sử dụng các phương pháp thay đổi trọng số để tính toán độ chính xác để đưa ra model có độ chính xác cao nhất.
- (5) : Sau khi hoàn thành bước 4, chúng em lựa chọn một mô hình PhoBERT đã được fine-tune để sử dụng. Đồng thời, các mô hình học máy truyền thống cũng được huấn luyện và đánh giá dựa trên độ chính xác. Trong số đó, mô hình Random Forest cho kết quả tốt nhất. Vì vậy, chúng tôi quyết định sử dụng hai mô hình chính trong hệ thống phân tích cảm xúc là: PhoBERT đã fine-tune và Random Forest. Mục đích là để so sánh hiệu quả giữa mô hình deep learning và mô hình truyền thống trong việc phân tích cảm xúc.
- (6) : Ta thu thập thông tin từ các nguồn gồm GitHub,Hugging Face hay Paper News chúng em sẽ tập hợp các thông tin về các câu mỉa mai hay còn gọi là sarcasm hay gặp trong lĩnh vực mua bán và lưu vào file có tên là Sarcasm.csv.
- (7) : Từ file Sarcasm.csv tiếp tục sẽ train thành các nhãn gồm Sarcasm_Service, Sarcasm_Price và Sarcasm_Quality cho từng loại cụ thể và lưu nó vào model Sarcasm.
- (8) : Kết hợp từ Model Sentiment và Model Sarcasm để tiến hành phân tích cảm xúc cho giọng điệu theo ngữ cảnh cho từng trường hợp cụ thể (có kết hợp thêm phần Retrain để cập nhật những comment mới vào model để tăng độ chính xác cao cho model trên sẽ được minh họa cụ thể ở trên web)

(9) : Áp dụng phương pháp content-based cộng thêm các điều kiện và kết quả từ phân tích cảm xúc ở trên (Rating, Service_sentiments, Price_sentiments, Quality_sentiments) để đưa ra kết quả gợi ý sản phẩm phù hợp với mức giá hay rating cụ thể.

(10) : Từ tập Emotional_classification.csv, tiếp tục dùng code để trích xuất ra các keyword về giá cụ thể gồm: giá thấp, giá cao, giá vừa phải, giá quá đắt,..

(11) : Từ file json ở trên có tên là keyword_price_intent.json sẽ được dùng để áp dụng vào cho hệ thống gợi ý về giá phù hợp cho từng loại sản phẩm.

→ Cuối cùng, triển khai hệ thống trên website bằng Framework Flask.

2.2. Thu thập và tiền xử lý dữ liệu

Thu thập dữ liệu:

Nhóm chúng em sẽ thực hiện cào các sản phẩm bao gồm laptop và điện thoại từ hai trang web nổi tiếng là Thế Giới Di Động và Điện Máy Xanh

- Truy cập vào trang web cần thu thập và tạo file csv để lưu trữ dữ liệu:

```
driver = webdriver.Chrome()
wait = WebDriverWait(driver, 10)

url = "https://www.thegioididong.com/dtdd"
driver.get(url)
time.sleep(3)

csv_file = '../data/data_List_Phone.csv'
save_to_csv([], csv_file)
```

- Lấy toàn bộ thông tin tên sản phẩm, giá tiền, rating, số bình luận, link sản phẩm trên web và lưu vào file csv :

```
# Lấy toàn bộ sản phẩm từ danh sách
phone_elements = driver.find_elements(By.CSS_SELECTOR, 'ul.listproduct
li.item.ajaxed.__cate_42')

for phone in phone_elements:
    try:
```

```

vote_txt = phone.find_element(By.CSS_SELECTOR, 'div.vote-txt')
if vote_txt.text.strip() != "":
    ten_SanPham = phone.find_element(By.CSS_SELECTOR,
        'a.main-contains h3').text
    giaTien_raw = phone.find_element(By.CSS_SELECTOR,
        'strong.price').text
    giaTien = convert_price_to_int(giaTien_raw)

    rating = float(vote_txt.find_element(By.TAG_NAME,
        'b').text)

    full_text = vote_txt.text
    comment = int(full_text.split('(')[-1].strip(' '))
    link_Phone = phone.find_element(By.CSS_SELECTOR,
        'a.main-contains').get_attribute('href')
    phone_data = {
        'ten_SanPham': ten_SanPham,
        'giaTien': giaTien,
        'rating': rating,
        'comment': comment,
        'link_Phone': link_Phone
    }
    save_to_csv([phone_data], csv_file)
driver.quit()

```

- File data sau khi chạy sẽ có dạng:

A	B	C	D	E	F	G	H	I	J	K	L
ten_SanPham	giaTien	rating	comment	link_Phone							
iPhone 16	34990000	3.1	31	https://www.dienmayxanh.com/dien-thoai/iphone-16-pro-max							
iPhone 16	28990000	2.6	9	https://www.dienmayxanh.com/dien-thoai/iphone-16-pro							
Xiaomi Red	5290000	3.2	224	https://www.dienmayxanh.com/dien-thoai/xiaomi-redmi-note-13							
realme C6	4290000	3.9	11	https://www.dienmayxanh.com/dien-thoai/realme-c65s							
realme C6	4290000	3.9	7	https://www.dienmayxanh.com/dien-thoai/realme-c65-8-128							
Samsung C	39490000	4.9	25	https://www.dienmayxanh.com/dien-thoai/samsung-galaxy-z-fold6							
Samsung C	9690000	4.3	21	https://www.dienmayxanh.com/dien-thoai/samsung-galaxy-a55-5g-128gb							
Samsung C	25990000	5	15	https://www.dienmayxanh.com/dien-thoai/samsung-galaxy-z-flip6							
vivo Y28	5790000	3.8	17	https://www.dienmayxanh.com/dien-thoai/vivo-y28							

Hình 22: Mô tả dữ liệu sau khi thu thập tên sản phẩm, giá tiền, rating, số bình luận, link sản phẩm trên web

- Tiếp theo, chúng em sẽ đọc file csv trên và duyệt qua từng link sản phẩm trong cột

link_Phone:

```
df = pd.read_csv('../data/data_List_Phone.csv')
product_info = {
    row['link_Phone']: {
        'phone_Name': row['ten_SanPham'],
        'phone_Price': row['giaTien']
    }
    for _, row in df.iterrows()
}
```

- Duyệt qua từng link trong cột link_Phone của file csv trước đó và thêm url /danh-gia để đưa tới trang chứa comment của sản phẩm:

```
for link in product_info.keys():
    if pd.notna(link):
        phone_name = product_info[link]['phone_Name']
        phone_price = product_info[link]['phone_Price']
        page_Index = 1
        driver = webdriver.Chrome()
        while True:
            modified_link = f"{link}/danh-gia?page={page_Index}"
            driver.get(modified_link)
            time.sleep(3)
```

- Duyệt qua từng thẻ chứa bình luận, thời gian bình luận, tên người dùng của các sản phẩm và thu thập:

```
# Duyệt qua từng thẻ <li class="par"> trong comment-list
    comments = driver.find_elements(By.CSS_SELECTOR,
    "li.par")
    temp_data = []
    for comment in comments:
        try:
            user_name =
```

```

comment.find_element(By.CSS_SELECTOR,
"div.cmt-top p.cmt-top-name").text

    if user_name == "":
        user_name = "N/A"
except NoSuchElementException:
    user_name = "N/A"
try:
    user_rating =
len(comment.find_elements(By.CSS_SELECTOR,
"div.cmt-intro div.cmt-top-star
i.iconcmt-starbuy"))
except NoSuchElementException:
    user_rating = 0
try:

    user_comment =
comment.find_element(By.CSS_SELECTOR,
"div.cmt-content p.cmt-txt").text
    if user_comment == "":
        user_comment = "N/A"
except NoSuchElementException:
    user_comment = "N/A"
try:

    user_time_comment =
    comment.find_element(By.CSS_SELECTOR,
        "div.cmt-command span.cmt.d.dot-line").text
        if user_time_comment ==
"":

        user_time_comment = "N/A"
    else:
        words = user_time_comment.split()
        if len(words) < 2:
            user_time_comment = "N/A"
        else:
            user_time_comment = "
".join(words[-2:])
except NoSuchElementException:

```

```
user_time_comment = "N/A"
```

Giải thích:

- Đầu tiên, chương trình sử dụng `driver.find_elements(By.CSS_SELECTOR, "li.par")` để tìm tất cả các thẻ `` có class `"par"`. Mỗi thẻ này tương ứng với một bình luận của người dùng trên trang web.

Với mỗi bình luận tìm được, ta tiếp tục trích xuất các thông tin chi tiết như sau:

- Tên người dùng được lấy từ thẻ `<p>` có class `"cmt-top-name"` nằm trong `<div class="cmt-top">`.
- Số sao đánh giá được xác định bằng cách đếm số lượng thẻ `<i>` có class `"iconcmt-starbuy"` trong `<div class="cmt-top-star">`; mỗi thẻ `<i>` đại diện cho một ngôi sao mà người dùng đánh giá.
- Nội dung bình luận được trích xuất từ thẻ `<p>` có class `"cmt-txt"` nằm trong `<div class="cmt-content">`, đây chính là đoạn văn mà người dùng đã viết.
- Thời gian đăng bình luận được lấy từ thẻ `` có class `"cmttd dot-line"` bên trong `<div class="cmt-command">`, thể hiện thời điểm người dùng đăng bình luận (ví dụ: "3 ngày trước", "1 tuần trước",...).
- Thu thập tất cả bình luận, thời gian bình luận, tên người dùng, rating của các sản phẩm và lưu vào file csv mới:

```
df_temp = pd.DataFrame(temp_data)
df_result = pd.concat([df_result, df_temp],
                       ignore_index=True)
df_result.to_csv(r'../data/phone_data.csv',
                 index=False, encoding='utf-8')
```

- File csv sau khi thu thập hoàn thiện sẽ có dạng như sau:

	A	B	C	D	E	F
1	Product Name	Current Price	Reviewer	Rating	Comment	Time Used
2	Xiaomi Redmi Note 13	5290000	Hải	5	H thay kính cam hết nhieu ạ	120
3	Xiaomi Redmi Note 13	5290000	Quách Đoàn	3	Mình vừa dùng dc tầm 2 tháng. Ma giờ dt. đơ giật lác. C	60
4	Xiaomi Redmi Note 13	5290000	Đỗ Văn Chien	3	Máy quá tệ Sài tôn pin quá à	1
5	Xiaomi Redmi Note 13	5290000	Thơm	4	Mình mới mua được nửa tháng bị lỗi sóng di động khi	14
6	Xiaomi Redmi Note 13	5290000	Nguyễn Văn Thích	4	Em muốn bán lại nhưng ko có phụ kiện thì mất bao nh	14
7	Xiaomi Redmi Note 13	5290000	Sơn	3	lúc mình mua shop bảo có hỗ trợ NFC về coi không co	5
8	Xiaomi Redmi Note 13	5290000	Đinh	1	Điện thoại giảm bắt sóng 4g yếu, lỗi từ lum mới bảo	210
9	Xiaomi Redmi Note 13	5290000	Trần Quang Duy	1	Sau một ngày sử dụng thì thấy máy rất pin nhanh hết t	1
10	Xiaomi Redmi Note 13	5290000	Phan Minh Lực	1	Pin tụt rất nhanh mặc dù pin là 5000, thông số camer	7

Hình 23: Mô tả 10 dòng đầu của tập dữ liệu sau khi thu thập từ trang web của Thế Giới Di Động.

Dữ liệu gồm các thông tin như tên sản phẩm, giá, tên người dùng, rating, bình luận, thời gian bình luận. Cào tương tự với các sản phẩm laptop.

Tiền xử lý dữ liệu:

- Dữ liệu sau khi thu thập về sẽ có 2 file là laptop và điện thoại

Kiểm tra giá trị null và thay thế các giá trị null trong cả 2 tập dữ liệu bằng các giá trị phổ biến:

```
df_laptop.isnull().sum()
```

```
mean_price = df_laptop['Current Price'].mean()
most_common_product_name = df_laptop['Product Name'].mode()[0]
most_common_time_used = df_laptop['Time Used'].mode()[0]
most_common_reviewer = df_laptop['Reviewer'].mode()[0]

df_laptop['Product Name'] = df_laptop['Product
Name'].fillna(most_common_product_name)
df_laptop['Time Used'] = df_laptop['Time
Used'].fillna(most_common_time_used)
df_laptop['Reviewer'] =
df_laptop['Reviewer'].fillna(most_common_reviewer)
df_laptop['Current Price'] = df_laptop['Current
Price'].fillna(mean_price)
most_common_comment = df_laptop['Comment'].mode()[0] if not
df_laptop['Comment'].mode().empty else 'Unknown'

df_laptop['Comment'] =
```

```
df_laptop['Comment'].fillna(most_common_comment)
df_laptop.to_csv("../data/laptop_data.csv", index=False, encoding='utf-8')
```

- Nối dataframe của laptop và điện thoại lại với nhau:

```
df_combined = pd.concat([df_laptop, df_dienthoai], ignore_index=True)
```

- Đổi ngày sử dụng trong cột Time_Used thành số(ngày)

```
def convert_time_to_days(time_str):
    if pd.isna(time_str):
        return np.nan
    time_str = time_str.lower().strip()
    if 'ngày' in time_str:
        return int(time_str.replace('ngày', '').strip())
    elif 'tuần' in time_str:
        return int(time_str.replace('tuần', '').strip()) * 7
    elif 'tháng' in time_str:
        return int(time_str.replace('tháng', '').strip()) * 30
    elif 'năm' in time_str:
        return int(time_str.replace('năm', '').strip()) * 365
    else:
        return np.nan
```

- Thêm cột ‘Rating Average’ vào tập dữ liệu bằng cách tính rating trung bình của từng sản phẩm:

```
df['Rating Average'] = df.groupby('Product
Name')['Rating'].transform('mean').round(1)
```

- Dữ liệu sau khi tiền xử lý sẽ được lưu vào file csv có dạng sau:

	A	B	C	D	E	F	G
1	Product Name	Current Price	Reviewer	Rating	Rating Average	Comment	Time Used
2	Laptop Acer Aspire 3 A315	16173535	huyền trang	3	4.2	em muốn c	180
3	Laptop Acer Aspire 3 A315	16173535	Trịnh Thanh Tú	4	4.2	Em có mua	15
4	Laptop Acer Aspire 3 A315	16173535	NGUYỄN THỊ TH	5	4.2	Mình chỉ sử	30
5	Laptop Acer Aspire 3 A315	16173535	bùi tiên	3	4.2	tôi không b	365
6	Laptop HP Gaming VICTUS	18490000	Nguyễn Chí Vỹ	4	4	Lap dùng ổ	15
7	Laptop HP Gaming VICTUS	18490000	Cường	2	4	Không nên	7
8	Laptop HP Gaming VICTUS	18490000	Nguyễn Chí Tân	5	4	Tốt đáng m	7
9	Laptop HP Gaming VICTUS	18490000	Nguyễn Xuân T	5	4	giá rẻ và ch	7
10	Laptop HP Gaming VICTUS	18490000	Trần Thị Ngọc K	4	4	máy tốt nh	90

Hình 24: Mô tả 10 dòng đầu của tập dữ liệu sau khi tiền xử lý.

Dữ liệu sau khi tiền xử lý sẽ có 7 cột thể hiện tên sản phẩm, giá, tên người dùng, rating, rating trung bình, bình luận, thời gian sử dụng(ngày).

2.3. Xử lý ngôn ngữ tự nhiên để phân loại cảm xúc

Phần này sẽ mô tả quá trình thực hiện xử lý ngôn ngữ tự nhiên dùng để dùng trong quy trình phân loại cảm xúc.

```
def clean_text(text):
    if not isinstance(text, str):
        text = ""

    text = re.sub(r'\d+', '', text)
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = text.lower()
    return text

df_data['Cleaned Comment'] = df_data['Comment'].apply(clean_text)
```

Mục đích của đoạn mã trên là làm sạch dữ liệu văn bản như loại bỏ các yếu tố không cần thiết như số, dấu câu, và chuẩn hóa về chữ thường trong cột Comment của DataFrame sau đó lưu kết quả đã làm sạch vào một cột mới tên là Cleaned Comment.

```
def load_stopwords(file_path):
    with open(file_path, 'r', encoding='utf-8') as f:
        stopwords = f.read().splitlines()
    return set(stopwords)

stopwords_file = "../NLP/vietnamese-stopwords.txt"
vietnamese_stopwords = load_stopwords(stopwords_file)

def remove_stopwords(text):
    tokens = text.split() # Chia từ
    filtered_tokens = [word for word in tokens if word not in
                       vietnamese_stopwords]
    return ' '.join(filtered_tokens)

df_data['Filtered Comment'] = df_data['Tokenized
Comment'].apply(remove_stopwords)
```

Đoạn mã trên nhằm mục đích loại bỏ các từ dừng (stopword) trong văn bản tiếng Việt để cải thiện chất lượng dữ liệu trước khi sử dụng cho các mô hình học máy. Đầu tiên, hàm `load_stopwords(file_path)` được sử dụng để tải danh sách các từ dừng từ file `vietnamese-stopwords.txt` được tải về từ Vinai trên Github, và lưu chúng vào một tập hợp (set). Tiếp theo, hàm `remove_stopwords(text)` chia văn bản thành các từ (tokenizer) và loại bỏ những từ có trong danh sách từ dừng, sau đó ghép các từ còn lại thành một chuỗi văn bản mới. Cuối cùng, đoạn mã áp dụng hàm này lên từng bình luận trong cột 'Tokenized Comment' của dataframe `df_data`, tạo ra một cột mới 'Filtered Comment' chứa các bình luận đã được lọc bỏ từ dừng. Việc này giúp làm sạch dữ liệu và nâng cao hiệu quả của các mô hình phân tích ngôn ngữ tự nhiên.

```
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(df_data['Filtered
Comment'])
print("TF-IDF matrix shape:", tfidf_matrix.shape)
```

Đoạn mã trên sử dụng kỹ thuật TF-IDF để chuyển đổi văn bản thành các vector số. Cụ thể, đối tượng `TfidfVectorizer` được tạo ra để áp dụng lên cột "Filtered Comment" trong dataframe `df_data`, từ đó tạo ra ma trận TF-IDF. Ma trận này thể hiện mức độ quan trọng của các từ trong các tài liệu, giúp mô hình học máy có thể hiểu và xử lý văn bản. Cuối cùng, `tfidf_matrix.shape` in ra kích thước của ma trận TF-IDF, cho biết số lượng từ và tài liệu trong tập dữ liệu.

Phần tiếp theo sẽ mô tả quá trình thực hiện gán nhãn cho mô hình PhoBert :

```
# === Map nhãn từ mô hình ===
label_map = {
    "NEG": "Negative",
    "NEU": "Neutral",
    "POS": "Positive"
}

# === Hàm dự đoán cảm xúc ===
def predict_sentiment(text):
    if not text.strip():
        return "None", 0.0
    result = sentiment_pipeline_phobert(text)[0]
```

```

label = label_map[result['label']]
score = result['score']
return label, score

```

Xây dựng hàm dự đoán cảm xúc với các nhãn ‘Negative’, ‘Neutral’, ‘Positive’. Nếu văn bản rỗng hoặc không liên quan đến khía cạnh thì hàm trả về ‘None’. Ngược lại, văn bản được đưa vào pipeline phân tích cảm xúc ‘sentiment_pipeline_phobert’, mô hình sẽ trả về nhãn cảm xúc tương ứng và độ tin cậy.

```

# === Load từ khóa từng khía cạnh ===
def load_keywords(file_path):
    with open(file_path, "r", encoding="utf-8") as f:
        return [line.strip().lower() for line in f if line.strip()]

service_keywords = load_keywords("keywords_dichvu.txt")
price_keywords = load_keywords("keywords_giaca.txt")
quality_keywords = load_keywords("keywords_chatluong.txt")

# === Tìm từ khóa liên quan trong bình luận ===
def extract_keywords(text, keywords):
    return [kw for kw in keywords if kw in text.lower()]

```

Load và áp dụng các file chứa từ khóa của 3 khía cạnh là Dịch vụ, Giá cả, Chất lượng. Dựa vào từ khóa, mô hình sẽ phân biệt được khía cạnh và phân loại cảm xúc đúng cho từng khía cạnh.

```

# === Phân tích cảm xúc theo từng khía cạnh ===
def analyze_comment_by_aspect(comment):
    aspects = {
        'Service': extract_keywords(comment, service_keywords),
        'Price': extract_keywords(comment, price_keywords),
        'Quality': extract_keywords(comment, quality_keywords),
    }

    results = {}
    for aspect, keywords in aspects.items():
        if keywords:
            joined_keywords = " ".join(keywords)
            label, _ = predict_sentiment(joined_keywords)
            results[aspect] = label
        else:

```

```
results[aspect] = "None"
return results
```

Hàm `analyze_comment_by_aspect` dùng để phân tích cảm xúc của bình luận theo từng khía cạnh (Dịch vụ, Giá cả, Chất lượng) bằng cách trích xuất từ khóa liên quan và dự đoán cảm xúc cho mỗi khía cạnh. Sau đó tiếp tục phân tích từng comment có liên quan đến các khía cạnh trên và lưu lại vào dataframe.

Cuối cùng, sau khi hoàn tất quá trình phân tích cảm xúc các kết quả sẽ được lưu vào file `Emotional_by_aspect.csv` được mô tả như sau:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	Product Name	Current Price	Reviewer	Rating	Rating Average	Comment	Time Used	Cleaned Comment	Tokenized Comment	Filtered Comment	Service Sentiment	Price Sentiment	Quality Sentiment	
1	Laptop Acer Aspire 3 A315 16173535	16173535	huyền trang	3	4.2	em muốn cà	180	em muốn cài sang ti	em muốn cài sang tiếng v	cài tiếng việt máy hong hỏ	None	None	Negative	
2	Laptop Acer Aspire 3 A315 16173535	16173535	Trịnh Thanh Tú	4	4.2	Em có mua n	15	em có mua máy này	em có mua máy này ở	thế mua máy thế giới di động t	None	None	Negative	
3	Laptop Acer Aspire 3 A315 16173535	16173535	NGUYỄN THỊ T	5	4.2	Mình chỉ sử c	30	mình chỉ sử dụng m	mình chỉ sử dụng mục đ	sử dụng mục đích học tập	Positive	Positive	Positive	
4	Laptop Acer Aspire 3 A315 16173535	16173535	bùi tiên	3	4.2	tôi không bật	365	tôi không bật được n	tôi không bật được micro	không bật micro laptop xin	None	None	None	
5	Laptop HP Gaming VICTUS 18490000	18490000	Nguyễn Chí Vỹ	4	4	Lap dùng ổn	15	lap dùng ổn khi chơi	lap dùng ổn khi chơi game	lap dùng ổn game tác vụ và	None	None	Neutral	
6	Laptop HP Gaming VICTUS 18490000	18490000	Cường	2	4	Không nên m	7	không nên mua con	không nên mua con này	không mua này dùng sác kh	None	None	Negative	
7	Laptop HP Gaming VICTUS 18490000	18490000	Nguyễn Chí Tân	5	4	Tốt đáng mua	7	tốt đáng mua	tốt đáng mua	tốt đáng mua	None	None	Positive	
8	Laptop HP Gaming VICTUS 18490000	18490000	Nguyễn Xuân T	5	4	giá rẻ và chấ	7	giá rẻ và chất lượng t	giá rẻ và chất lượng tốt	giá rẻ và chất lượng tốt	None	Positive	Positive	
9	Laptop HP Gaming VICTUS 18490000	18490000	Trần Thị Ngọc K	4	4	máy tốt nh	90	máy tốt nhân viên tậ	máy tốt nhân viên tận t	máy tốt nhân viên tận t	Positive	None	Positive	

Hình 25: Mô tả 10 dòng đầu của tập dữ liệu sau khi xử lý ngôn ngữ tự nhiên và gán nhãn.

Tập dữ liệu sau khi hoàn tất sẽ bao gồm 13 cột là: Product Name (Tên sản phẩm), Current Price (Giá hiện tại), Reviewer (Người đánh giá), Rating (Điểm đánh giá của người dùng), Rating Average (Điểm trung bình của sản phẩm), Comment (Bình luận gốc của người dùng), Time Used (Thời gian sử dụng sản phẩm), Cleaned Comment (Bình luận sau khi làm sạch), Tokenized Comment (Bình luận đã được tách từ/token hóa), Filtered Comment (Bình luận sau khi loại bỏ từ dừng), Service Sentiment (Cảm xúc về dịch vụ), Price Sentiment (Cảm xúc về giá cả), Quality Sentiment (Cảm xúc về chất lượng).

Nhóm đã sử dụng mô hình PhoBERT để gán nhãn cho tập dữ liệu, tuy nhiên, do hạn chế của mô hình, một số nhãn được dự đoán chưa chính xác. Để cải thiện chất lượng dữ liệu, nhóm đã tiến hành kiểm tra và điều chỉnh thủ công các nhãn bị phân loại sai, đảm bảo độ chính xác cao nhất cho tập dữ liệu và đưa ra tập dữ liệu cuối là tập dữ liệu có độ chính xác tốt nhất. Việc này giúp tạo nền tảng dữ liệu vững chắc, góp phần nâng cao hiệu suất và độ tin cậy khi áp dụng các mô hình học máy sau này.

2.4. Xây dựng mô hình học máy

2.4.1. Các phương pháp đánh giá

Accuracy: thể hiện độ chính xác của mô hình. Tuy nhiên, khi dữ liệu không cân bằng (ví dụ: một lớp chiếm ưu thế), accuracy có thể không phản ánh chính xác hiệu suất của mô hình. Ví dụ, nếu mô hình chỉ dự đoán lớp phổ biến nhất, accuracy có thể cao, nhưng mô hình không thực sự hữu ích, được tính bằng tỷ lệ tổng số dự đoán đúng so với tổng số mẫu.

Công thức:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Trong bài toán nhiều nhãn (Price, Quality, Service), Accuracy được tính riêng cho từng nhãn.

Joint Accuracy (Độ chính xác liên nhiệm): Tỷ lệ số mẫu mà mô hình dự đoán đúng tất cả các nhãn con (ví dụ: Price, Quality, Service) trên tổng số mẫu:

$$\text{Joint Accuracy} = \frac{1}{N} \sum_{i=1}^N \mathbf{1} \left(\hat{y}_i^{(1)} = y_i^{(1)} \wedge \hat{y}_i^{(2)} = y_i^{(2)} \wedge \hat{y}_i^{(3)} = y_i^{(3)} \right)$$

Trong đó:

- $\hat{y}_i^{(j)}$ là nhãn dự đoán cho nhãn con thứ j của mẫu i.
- $y_i^{(j)}$ là nhãn thực tế,
- $\mathbf{1}(\cdot)$ là hàm chỉ thị (bằng 1 nếu đúng, 0 nếu sai).
- N là tổng số mẫu.

F1-Score: là một chỉ số tổng hợp, đặc biệt hữu ích khi cần một thước đo duy nhất cho mô hình phân loại, nơi mà cả precision và recall đều quan trọng. F1-Score thấp cho

thấy mô hình có thể tốt ở một trong hai (precision hoặc recall), nhưng không hoàn hảo ở cả hai.

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

Macro F1-Score: Tính trung bình F1-Score cho từng lớp, không tính đến phân bố dữ liệu:

$$\text{Macro F1} = \frac{1}{K} \sum_{k=1}^K F1_k$$

Trong đó:

- K là số lớp (ví dụ: None, Positive, Neutral, Negative).
- $F1_k$ là F1-score cho lớp k .

Average Macro F1: trung bình cộng của Macro F1 cho từng nhãn con (Price, Quality, Service):

$$\text{Avg Macro F1} = \frac{1}{3} (\text{Macro F1}_{\text{Price}} + \text{Macro F1}_{\text{Quality}} + \text{Macro F1}_{\text{Service}})$$

2.4.2. Mô hình truyền thống

```
df=pd.read_csv("Emotional_by_aspect.csv")
df['Tokenized Comment'].fillna('', inplace=True)
sentiment_map = {'None': 0, 'Positive': 1, 'Neutral': 2, 'Negative':3}
sentiment_columns = ['Service Sentiment', 'Price Sentiment', 'Quality Sentiment']
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(df['Filtered Comment'])
X_seq = pad_sequences(tokenizer.texts_to_sequences(df['Filtered Comment']), maxlen=100)
# TF-IDF vectorizer for ML models
vectorizer_tfidf = TfidfVectorizer()
X_tfidf = vectorizer_tfidf.fit_transform(df['Filtered Comment'])
```

- Chuẩn bị đặc trưng cho mô hình học máy : Sử dụng TF-IDF (TfidfVectorizer) để biến đổi cột Tokenized Comment thành ma trận đặc trưng dạng số, phục vụ cho các mô hình học máy truyền thống về sau.

- Chuẩn bị dữ liệu cho mô hình học sâu : Sử dụng Tokenizer (num_words=5000) để giữ lại 5,000 từ phổ biến nhất, chuyển các bình luận thành chuỗi số nguyên với texts_to_sequences, sau đó chuẩn hóa độ dài mỗi chuỗi về 100 từ bằng pad_sequences, tạo ra ma trận X_seq làm đầu vào cho các mô hình học sâu về sau như LSTM.

```
# ===== TF-IDF + Logistic Regression =====
X_tfidf = vectorizer_tfidf.transform(df_filtered[
'Filtered Comment'])
X_train_tfidf, X_test_tfidf, y_train, y_test =
train_test_split(X_tfidf, y, test_size=0.2, random_state=42)
```

- Chia dữ liệu thành tập train-test với tỷ lệ **80/20**.

2.4.2.1. Mô hình Logistic Regression

```
log_reg = MultiOutputClassifier(LogisticRegression(max_iter=1000))
log_reg.fit(X_train_tfidf, Y_train)
Y_pred_log = log_reg.predict(X_test_tfidf)
```

Áp dụng mô hình Logistic Regression trên đặc trưng TF-IDF. Mô hình Logistic Regression được huấn luyện với max_iter=1000 và dùng để dự đoán nhãn cảm xúc trên tập kiểm tra và cho ra kết quả bên dưới :

```
--- Logistic Regression ---
Joint Accuracy: 0.8333
Accuracy - Price: 0.8333
Accuracy - Quality: 0.8889
Accuracy - Service: 0.9444
Macro F1 - Price: 0.6516
Macro F1 - Quality: 0.604
Macro F1 - Service: 0.4857
Avg Macro F1: 0.5804
```

2.4.2.2. Mô hình Random Forest Classifier

```
base_rf = RandomForestClassifier(
    random_state=42,
    class_weight='balanced',
    n_jobs=-1)
```

```

)
multi_rf = MultiOutputClassifier(base_rf, n_jobs=-1)
param_dist = {
    'estimator__n_estimators': randint(200, 600),
    'estimator__max_depth': [10, 20, 30, 40, 50, None],
    'estimator__min_samples_split': [2, 5, 10],
    'estimator__min_samples_leaf': [1, 2, 4],
    'estimator__max_features': ['sqrt', 'log2', None],
    'estimator__bootstrap': [True, False]
}
random_search = RandomizedSearchCV(
    multi_rf,
    param_distributions=param_dist,
    n_iter=50, # Tăng số phép thử
    cv=3,
    scoring=scorer,
    verbose=2,
    n_jobs=-1,
    random_state=42
)
random_search.fit(X_train_tfidf, Y_train)
best_rf = random_search.best_estimator_
Y_pred_rf = best_rf.predict(X_test_tfidf)

```

Với đoạn mã trên, ta tạo một mô hình Random Forest đa đầu ra bằng cách kết hợp RandomForestClassifier với MultiOutputClassifier, đồng thời sử dụng RandomizedSearchCV để tìm kiếm tổ hợp siêu tham số tối ưu. Mô hình sử dụng random_state=42 để đảm bảo tính tái lập của kết quả, và được huấn luyện với tập hợp các siêu tham số như số lượng cây (estimators từ 200 đến 600), độ sâu cây (max_depth), và chiến lược lấy mẫu (bootstrap). Việc tối ưu hóa này giúp ta nâng cao độ chính xác và khả năng khái quát hóa của mô hình. Mô hình sau đó được sử dụng để dự đoán nhãn cảm xúc theo ba khía cạnh (giá cả, chất lượng, dịch vụ), và cho phép ta đánh giá và so sánh hiệu quả của Random Forest so với mô hình Logistic Regression trước đó thông qua các chỉ số như Accuracy, Macro F1-score và Joint Accuracy trong bài toán phân loại cảm xúc đa nhãn. Kết quả đạt được sau khi train mô hình Random Forest:

```

--- Optimized Random Forest ---
Joint Accuracy: 0.7778
Accuracy - Price: 0.9444
Accuracy - Quality: 0.9444
Accuracy - Service: 0.8889
Macro F1 - Price: 0.9113
Macro F1 - Quality: 0.9496
Macro F1 - Service: 0.4706
Avg Macro F1: 0.7772

```

2.4.3. Mô hình học sâu

2.4.3.1. Mô hình Long Short Term Memory

```

# ===== LSTM =====
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(df_filtered['Filtered Comment'])
X_seq =
pad_sequences(tokenizer.texts_to_sequences(df_filtered['Filtered
Comment']), maxlen=70)

X_train_seq, X_test_seq, Y_train_seq, Y_test_seq =
train_test_split(X_seq, Y, test_size=0.2, random_state=42)

input_layer = Input(shape=(70,))
embedding = Embedding(input_dim=5000, output_dim=128)(input_layer)
lstm_out = LSTM(128, return_sequences=True)(embedding)
lstm_out = Dropout(0.5)(lstm_out)
lstm_out = LSTM(64)(lstm_out)
lstm_out = Dropout(0.5)(lstm_out)
out_price = Dense(4, activation='softmax', name='price')(lstm_out)
out_quality = Dense(4, activation='softmax', name='quality')(lstm_out)
out_service = Dense(4, activation='softmax', name='service')(lstm_out)

model = Model(inputs=input_layer, outputs=[out_price, out_quality,
out_service])
model.compile(
    loss={
        'price': 'sparse_categorical_crossentropy',
        'quality': 'sparse_categorical_crossentropy',
        'service': 'sparse_categorical_crossentropy'
    },
    optimizer='adam',

```

```

metrics={
    'price': 'accuracy',
    'quality': 'accuracy',
    'service': 'accuracy'
} )
model.fit(
    X_train_seq,
    [Y_train_seq['Price_Label'], Y_train_seq['Quality_Label'],
    Y_train_seq['Service_Label']],
    epochs=30,
    batch_size=32,
    validation_data=(X_test_seq, [Y_test_seq['Price_Label'],
    Y_test_seq['Quality_Label'], Y_test_seq['Service_Label']])
)

```

Ta khởi tạo mô hình LSTM cho bài toán phân loại cảm xúc đa nhiệm với ba khía cạnh. Mô hình bắt đầu bằng lớp embedding: sử dụng 5000 từ phổ biến, mỗi từ được ánh xạ thành vector 128 chiều, và câu được chuẩn hóa về độ dài 70 từ.

Tiếp theo là hai tầng LSTM : tầng một LSTM đầu tiên có 128 đơn vị, kết hợp Dropout 50% để giảm overfitting và tầng thứ hai LSTM thứ hai có 64 đơn vị, tổng hợp toàn bộ câu, cũng kèm Dropout.

Đầu ra gồm ba lớp Dense, mỗi lớp có 4 nơon tương ứng với các nhãn cảm xúc: None, Tích cực, Trung tính, Tiêu cực, sử dụng softmax.

Mô hình dùng hàm mất mát `sparse_categorical_crossentropy` do nhãn là số nguyên, tối ưu bằng Adam, và đánh giá bằng độ chính xác cho từng nhãn. Cuối cùng, mô hình được huấn luyện trong 30 epoch với batch size 32 để so sánh hiệu quả với các mô hình khác. Sau khi huấn luyện mô hình LSTM, ta đạt được kết quả sau:

```

--- LSTM ---
Joint Accuracy: 0.6111
Accuracy - Price: 0.8333
Accuracy - Quality: 0.6667
...
Macro F1 - Price: 0.6516
Macro F1 - Quality: 0.2667
Macro F1 - Service: 0.4857
Avg Macro F1: 0.468

```

2.4.3.2. Mô hình Fine-tuning PhoBert

Sau khi thực hiện dùng phoBert để gán nhãn và hiệu chỉnh thô từ phần 2.3 nhóm chúng em sẽ thực hiện tinh chuẩn phoBert để đem lại độ chính xác cao nhất cho việc đánh giá cảm xúc. Việc tinh chỉnh mô hình sẽ tập trung vào phần trích xuất đặc trưng và phân lớp.

```

# Load Dataset with correct column names
df_data = pd.read_csv("Emotional_by_aspect.csv")
label_mapping = {'None': 0, 'Positive': 1, 'Neutral': 2, 'Negative': 3}
df_data['Price_Label'] = df_data['Price Sentiment'].map(label_mapping).fillna(0)
df_data['Quality_Label'] = df_data['Quality Sentiment'].map(label_mapping).fillna(0)
df_data['Service_Label'] = df_data['Service Sentiment'].map(label_mapping).fillna(0)

texts = df_data['Tokenized Comment'].tolist()
price_labels = df_data['Price_Label'].tolist()
quality_labels = df_data['Quality_Label'].tolist()
service_labels = df_data['Service_Label'].tolist()

```

Chuẩn bị dữ liệu đầu vào cho mô hình học sâu đa tác vụ (multi-task learning) với ba nhãn cảm xúc: Price, Quality và Service.

```

# Tokenizer and Dataloader
tokenizer = AutoTokenizer.from_pretrained("vinai/phobert-base")

```

```
train_texts, val_texts, train_p, val_p, train_q, val_q, train_s, val_s
= train_test_split(
    texts, price_labels, quality_labels, service_labels,
    test_size=0.2, random_state=42)
```

Chia dữ liệu thành tập train-test với tỷ lệ **80/20**. Sau bước này, dữ liệu được dùng để tạo Dataset và DataLoader phục vụ huấn luyện và đánh giá mô hình.

```
# Attention pooling layer
class AttentionPooling(nn.Module):
    def __init__(self, hidden_size):
        super(AttentionPooling, self).__init__()
        self.attention = nn.Sequential(
            nn.Linear(hidden_size, 128),
            nn.Tanh(),
            nn.Linear(128, 1)
        )

    def forward(self, hidden_states, mask):
        scores = self.attention(hidden_states).squeeze(-1)
        scores = scores.masked_fill(mask == 0, -1e9)
        weights = torch.softmax(scores, dim=1).unsqueeze(-1)
        return torch.sum(hidden_states * weights, dim=1)
```

Xây dựng class Attention Pooling. Ý tưởng là thay vì lấy trung bình hoặc max pool đơn giản, ta học được một trọng số attention để "tập trung" vào những phần quan trọng hơn của chuỗi đầu vào, sau đó lấy weighted sum (tổng có trọng số) của các vector đó.

```
# PhoBERT MTL with attention pooling and adapter
class PhoBERTMultiTaskWithAttentionAdapter(nn.Module):
    def __init__(self, pretrained_model_name, num_labels=4):
        super().__init__()
        self.bert = AutoModel.from_pretrained(pretrained_model_name)
        self.pooling = AttentionPooling(self.bert.config.hidden_size)

        self.adapter = nn.Sequential(
            nn.Linear(self.bert.config.hidden_size, 256),
            nn.ReLU(),
            nn.Linear(256, self.bert.config.hidden_size)
        )

        self.dropout = nn.Dropout(0.3)
```

```

        self.classifier_price =
nn.Linear(self.bert.config.hidden_size, num_labels)
        self.classifier_quality =
nn.Linear(self.bert.config.hidden_size, num_labels)
        self.classifier_service =
nn.Linear(self.bert.config.hidden_size, num_labels)

    def forward(self, input_ids, attention_mask):
        outputs = self.bert(input_ids=input_ids,
attention_mask=attention_mask)
        pooled = self.pooling(outputs.last_hidden_state,
attention_mask)
        adapted = self.adapter(pooled)
        final_output = self.dropout(adapted)

    return {
        'price': self.classifier_price(final_output),
        'quality': self.classifier_quality(final_output),
        'service': self.classifier_service(final_output),
    }

```

Xây dựng class PhoBERTMultiTask With Attention Adapter, ý nghĩa của class này là một kiến trúc đa nhiệm (multi-task learning) sử dụng PhoBERT làm bộ mã hóa ngôn ngữ để xử lý cùng lúc ba nhiệm vụ phân loại liên quan đến giá cả, chất lượng và dịch vụ. Việc sử dụng attention pooling giúp mô hình tự học trọng số tập trung vào những phần quan trọng nhất trong chuỗi đầu vào thay vì chỉ lấy đại diện cố định như token CLS, từ đó cải thiện khả năng biểu diễn. Adapter được thêm vào nhằm điều chỉnh đặc trưng sau pooling, giúp mô hình tùy biến hiệu quả cho từng nhiệm vụ mà không cần tinh chỉnh toàn bộ PhoBERT, giúp tiết kiệm tài nguyên tính toán và tăng tốc quá trình huấn luyện. Dropout được áp dụng để giảm nguy cơ overfitting.

Tổng thể, kiến trúc này tận dụng sức mạnh của mô hình ngôn ngữ lớn kết hợp với kỹ thuật attention và đa nhiệm, giúp cải thiện hiệu suất dự đoán và tận dụng triệt để mối quan hệ giữa các nhiệm vụ liên quan. Sau khi áp dụng các kỹ thuật được mở rộng trên, mô hình PhoBERT Fine tune đã cho ra các chỉ số kết quả sau:

```

--- Validation Results ---
Avg Validation Loss: 0.3846

[Price]
      precision    recall  f1-score   support

    None         0.99      0.99      0.99     1308
  Positive         0.85      0.90      0.88         63
    Neutral         0.62      0.62      0.62         24
   Negative         0.92      0.85      0.88         13

 accuracy                   0.98     1408
 macro avg         0.85      0.84      0.84     1408
weighted avg         0.98      0.98      0.98     1408

[Quality]
      precision    recall  f1-score   support

    None         0.85      0.77      0.81         78
  Positive         0.97      0.97      0.97        700
    Neutral         0.83      0.84      0.83        203
   Negative         0.94      0.95      0.94        427

 accuracy                   0.93     1408
 macro avg         0.89      0.88      0.89     1408
weighted avg         0.93      0.93      0.93     1408

[Service]
      precision    recall  f1-score   support

    None         0.98      1.00      0.99     1315
  Positive         0.92      0.79      0.85         58
    Neutral         0.50      0.08      0.14         12
   Negative         0.73      0.48      0.58         23

 accuracy                   0.97     1408
 macro avg         0.78      0.59      0.64     1408
weighted avg         0.97      0.97      0.97     1408

Joint Accuracy (cả 3 task đều đúng): 0.8942

Accuracy từng task:
- Price: 0.9780
- Quality: 0.9311
- Service: 0.9709
Accuracy trung bình (cả mô hình): 0.9600

Macro F1 từng task:
- Price: 0.8431
- Quality: 0.8863
- Service: 0.6398

Average Macro F1 (cả mô hình): 0.7897

```


2.5. Xây dựng hệ thống phân tích cảm xúc

Trong phần này, nhóm đã xây dựng một hệ thống phân tích cảm xúc bằng cách áp dụng các mô hình học máy và mô hình ngôn ngữ tiên tiến, nhằm tối ưu hóa khả năng phân loại cảm xúc. Cụ thể, hệ thống sử dụng hai phương pháp chính là Random Forest Classifier và PhoBERT Fine-Tuned để đảm bảo hiệu suất và độ chính xác cao nhất.

Kết quả của phần phân tích cảm xúc này là kết quả của 2 mô hình học máy **Random Forest Classifier** và **PhoBERT Fine-Tuned**. Nhóm đưa ra kết quả hai mô hình để nhằm mục đích so sánh kết quả giữa hai mô hình giúp đánh giá hiệu suất và xác định mô hình nào hoạt động tốt hơn trên tập dữ liệu hiện tại. Điều này hỗ trợ nhóm lựa chọn mô hình phù hợp nhất hoặc kết hợp cả hai để tối ưu hóa hệ thống phân tích cảm xúc.

2.5.1. Dự đoán cảm xúc với mô hình Random Forest với TF-IDF

Trước hết, hệ thống sẽ nạp vectorizer TF-IDF đã được huấn luyện trước đó. Vectorizer này đã học cách chuyển đổi các từ trong bình luận của người dùng thành các con số đại diện đặc trưng (vector TF-IDF) dựa trên tập dữ liệu bình luận trước. Việc nạp vectorizer được thực hiện bằng dòng lệnh:

```
vectorizer = joblib.load("./models/tfidf_vectorizer.pkl")
```

Dòng code sẽ tải lại đối tượng vectorizer đã lưu trữ, để có thể sử dụng ngay cho việc biến đổi các văn bản mới thành dạng vector số học.

TF-IDF Vectorizer phân tích tần suất các từ trong câu, sau đó chuyển thành một vector số học để dễ dàng đưa vào mô hình học máy. Điều này giúp mô hình hiểu được nội dung văn bản một cách có hệ thống.

Tiếp theo, hệ thống sẽ nạp mô hình Random Forest đã được huấn luyện từ (2.4.2.2), với nhiệm vụ cụ thể là phân loại cảm xúc của người dùng đối với từng khía cạnh riêng biệt của sản phẩm như "giá cả" (Price), "chất lượng" (Quality) hay "dịch vụ" (Service). Mỗi mô hình được huấn luyện độc lập trên một tập dữ liệu đã được gán nhãn theo khía cạnh tương ứng, nhằm đảm bảo rằng mô hình có thể hiểu rõ ngữ cảnh và đặc điểm ngôn ngữ liên quan đến từng loại cảm xúc cụ thể.

```
rf_model_multi = joblib.load("/models/random_forest_multi.pkl")
```

Các mô hình này đã học cách nhận diện các dấu hiệu trong văn bản liên quan đến cảm xúc về giá cả, chất lượng, dịch vụ.

Sau khi có vectorizer và mô hình, bước tiếp theo là chuyển đổi văn bản đầu vào thành vector TF-IDF để mô hình có thể xử lý. Hàm transform nhận vào một danh sách chứa câu cần phân tích và trả về ma trận đặc trưng:

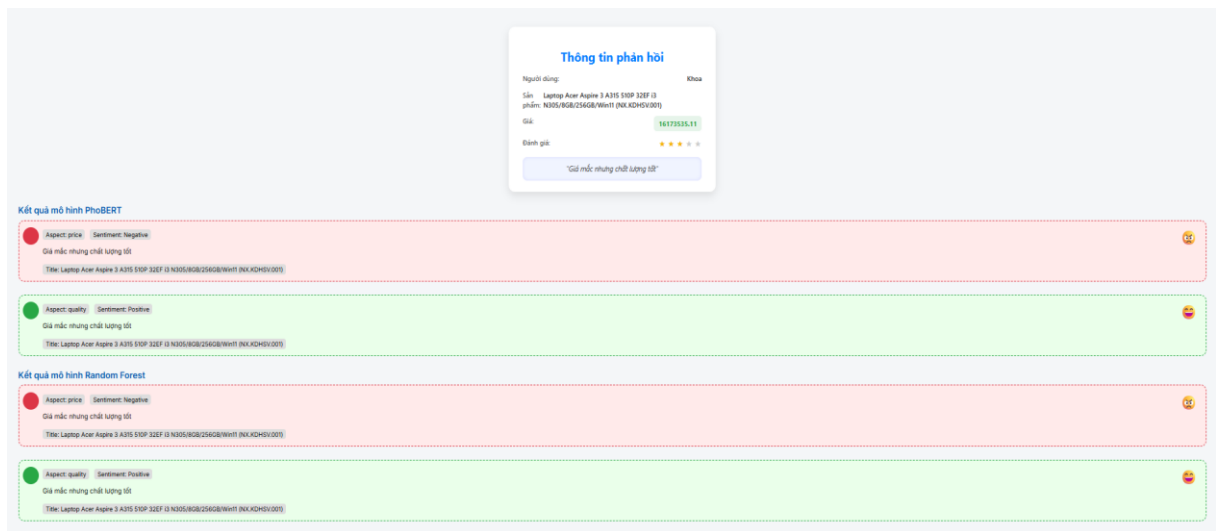
```
features = vectorizer.transform([text])
```

Cuối cùng, vector TF-IDF này được đưa vào mô hình Random Forest để dự đoán nhãn cảm xúc của câu bình luận, ví dụ như tích cực (positive), trung tính (neutral), hay tiêu cực (negative):

```
price_pred = rf_model_price.predict(features)[0]  
quality_pred = rf_model_quality.predict(features)[0]  
service_pred = rf_model_service.predict(features)[0]
```

Kết quả trả về là nhãn cảm xúc dự đoán cho khía cạnh giá cả, giúp hệ thống hiểu được cảm xúc của người dùng về giá sản phẩm.

Ví dụ: Khi bình luận ‘Giá mắc nhưng chất lượng tốt’ thì hệ thống sẽ nhận diện khía cạnh liên quan là Giá cả và Chất lượng tương ứng với cảm xúc Negative ở Price(Giá cả) và Positive ở Chất lượng(Quality).



Hình 26: Giao diện khi phân tích cảm xúc của Random Forest.

2.5.2. Dự đoán cảm xúc với PhoBERT Fine-Tuned

Ở phần này, sẽ sử dụng, tích hợp mô hình PhoBERT được tinh chỉnh để giải quyết ba tác vụ phân loại cảm xúc cùng lúc – tương ứng với ba khía cạnh chính: Giá cả (Price), Chất lượng (Quality), và Dịch vụ (Service) đã được huấn luyện ở phần (2.4.3.2).

Để sử dụng mô hình đã huấn luyện cho dự đoán mới, sẽ thực hiện các bước như sau:

```
# Khởi tạo lại kiến trúc mô hình
model = PhoBERTMultiTaskWithAttentionAdapter()
```

Ở bước này, nhóm khởi tạo lại mô hình với kiến trúc y hệt như khi huấn luyện. Điều này là bắt buộc vì các trọng số được nạp vào sau đó phải khớp hoàn toàn với cấu trúc mạng.

```
#Nạp trọng số đã huấn luyện
model.load_state_dict(torch.load("E:/Khóa
Luận/phobert_multitask_with_attention_adapter_state_dict3.pt",
map_location=device))
```

Dòng lệnh trên giúp tải toàn bộ tham số đã được mô hình học trong quá trình huấn luyện trước đó từ file .pt vào mô hình mới khởi tạo. Tham số map_location=device giúp đảm bảo mô hình được đưa đúng vào thiết bị xử lý hiện tại (CPU hoặc GPU).

```
model.to(device)
model.eval()
```

Sau khi tải trọng số, mô hình được chuyển sang thiết bị (device) để xử lý và được thiết lập chế độ eval() nhằm sử dụng cho suy luận (dự đoán), thay vì huấn luyện. Điều này giúp vô hiệu hóa các cơ chế như dropout để đầu ra ổn định hơn.

Khi người dùng nhập một câu bình luận, mô hình sẽ xử lý nội dung này thông qua một hàm tiện ích được định nghĩa sẵn (ví dụ predict_sentiment(text)) mà bên trong đã bao gồm các bước token hóa, chuẩn hóa, padding, và đưa dữ liệu vào mô hình.

Ví dụ:

```
price_sentiment, quality_sentiment, service_sentiment =
predict_sentiment("Sản phẩm giao nhanh, nhân viên thân thiện nhưng giá
hơi cao.")
```

- Dữ liệu đầu vào

Hàm predict_sentiment() sẽ xử lý toàn bộ pipeline đầu vào. Văn bản đầu vào được mã hóa bằng AutoTokenizer từ mô hình PhoBERT. Tokenizer trả về các tensor input_ids và attention_mask với độ dài cố định (max_length=128). Điều này đảm bảo tính đồng nhất đầu vào khi đưa vào mô hình.

```
encoding = tokenizer(
    text,
    truncation=True,
    padding='max_length',
    max_length=128,
    return_tensors='pt'
)

input_ids = encoding['input_ids'].to(device)
attention_mask = encoding['attention_mask'].to(device)
```

- Dự đoán cảm xúc

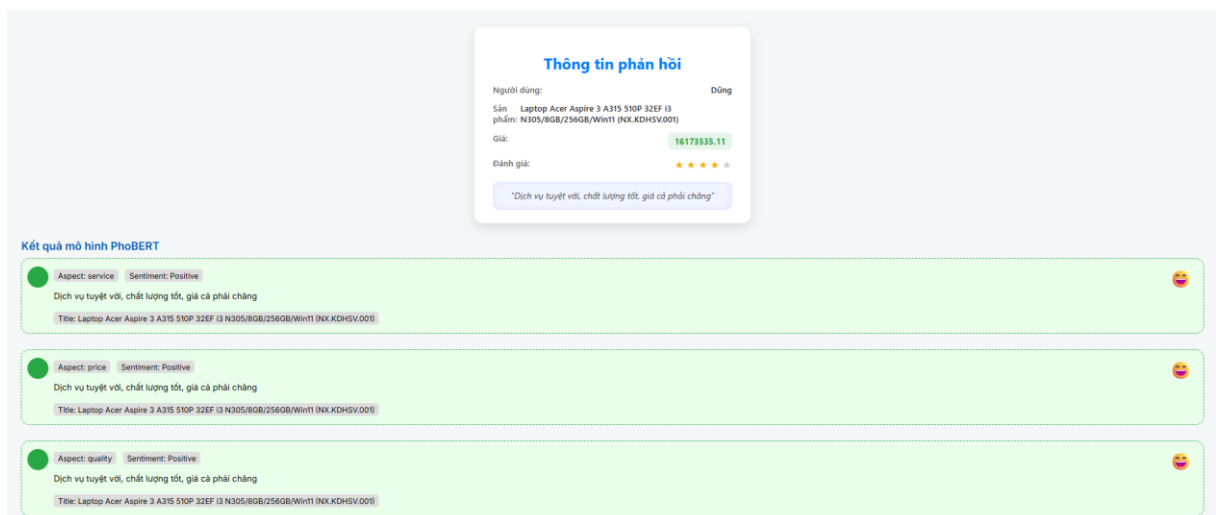
Mô hình đầu ra một dictionary gồm ba tensor dự đoán tương ứng ba khía cạnh:

```
"price": self.classifier_price(adapted_output),  
"quality": self.classifier_quality(adapted_output),  
"service": self.classifier_service(adapted_output),
```

Sau khi thực hiện `torch.argmax` trên từng tensor, hệ thống thu được nhãn cảm xúc đầu ra cho từng khía cạnh.

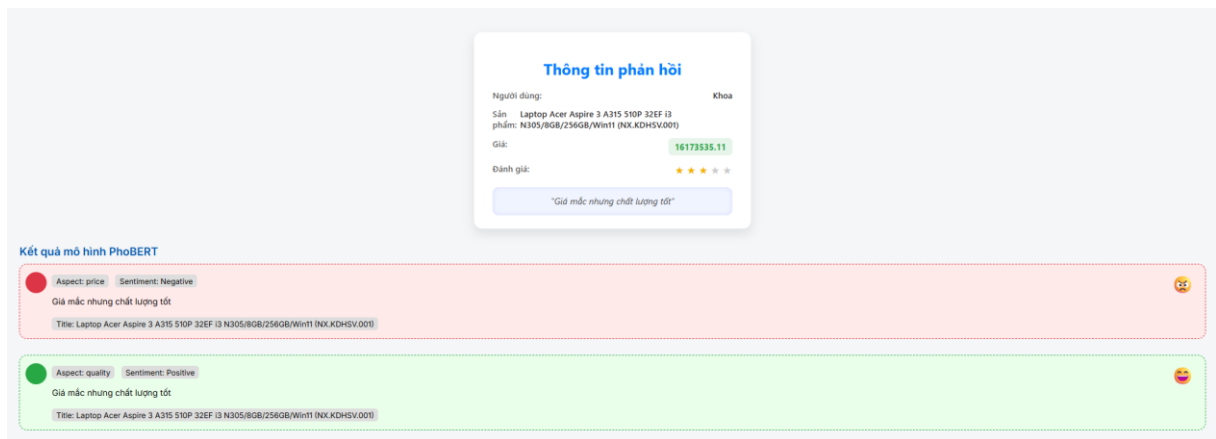
Bằng cách kết hợp PhoBERT với attention và adapter trong một mô hình đa tác vụ, hệ thống không chỉ tận dụng được kiến thức ngôn ngữ mạnh mẽ từ mô hình tiền huấn luyện, mà còn có khả năng phân tích cảm xúc theo ba khía cạnh cùng lúc, một cách chính xác, hiệu quả và tối ưu hóa tài nguyên huấn luyện. Cách tiếp cận này góp phần nâng cao khả năng hiểu ngữ cảnh và ý định của người dùng trong các bài đánh giá sản phẩm.

Ví dụ, khi bình luận ‘Dịch vụ tuyệt vời, chất lượng tốt, giá cả phải chăng’ thì hệ thống sẽ nhận diện các khía cạnh liên quan là Dịch vụ, Giá cả và Chất lượng và phân tích cảm xúc của bình luận này với cảm xúc Positive (Tích cực).



Hình 27: Giao diện khi phân tích cảm xúc liên quan tới 3 khía cạnh của PhoBERT Fine tuned.

Ví dụ: Khi bình luận ‘Giá mắc nhưng chất lượng tốt’ thì hệ thống sẽ nhận diện khía cạnh liên quan là Giá cả và Chất lượng tương ứng với cảm xúc Negative ở Price(Giá cả) và Positive ở Chất lượng(Quality).



Hình 28: Giao diện khi phân tích cảm xúc liên quan tới 2 khía cạnh của PhoBERT Fine tuned.

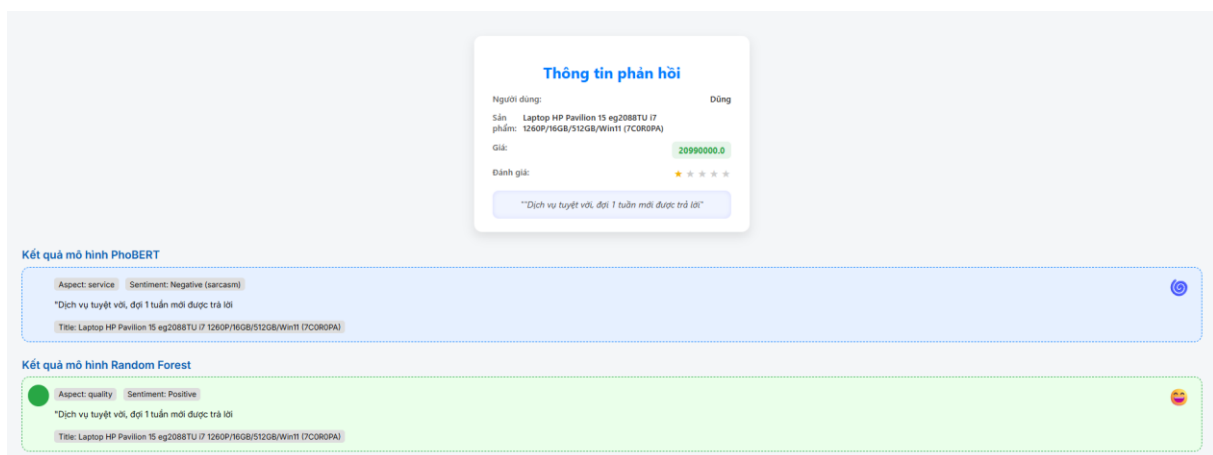
2.5.3. Phát hiện sarcasm trong bình luận

Mô hình phát hiện sarcasm được xây dựng dựa trên kiến trúc PhoBERT, với một lớp phân loại đa nhãn (multi-label classifier) ở phía trên. Khi gọi mô hình, bình luận đầu vào sẽ được chuyển thành các tensor `input_ids` và `attention_mask` bằng tokenizer của PhoBERT. Sau đó, mô hình trả về một vector logits biểu diễn xác suất của sarcasm cho từng khía cạnh.

```
encoding = tokenizer(
    text,
    truncation=True,
    padding='max_length',
    max_length=128,
    return_tensors='pt' )
input_ids = encoding['input_ids'].to(device)
attention_mask = encoding['attention_mask'].to(device)
with torch.no_grad():
    logits = sarcasm_model(input_ids,
attention_mask).sigmoid().cpu().numpy()[0]
    preds = (logits >= 0.5).astype(int)
    return {
        "sarcasm_service": int(preds[0]),
        "sarcasm_price": int(preds[1]),
        "sarcasm_quality": int(preds[2]) }
```

Đầu tiên, câu bình luận được chuyển đổi thành các tensor `input_ids` và `attention_mask` để phù hợp với định dạng đầu vào của mô hình PhoBERT. Khi đưa vào mô hình sarcasm, nó sẽ trả về các giá trị logits biểu thị mức độ bình luận có tính châm biếm ở từng khía cạnh. Những giá trị này được chuyển sang dạng xác suất từ 0 đến 1 bằng hàm sigmoid. Cuối cùng, dựa trên ngưỡng 0.5, nếu xác suất của một khía cạnh nào đó lớn hơn hoặc bằng 0.5 thì bình luận được coi là có sarcasm ở khía cạnh đó, ngược lại thì không có.

Sau khi phát hiện sarcasm, hệ thống sẽ kết hợp kết quả này với nhãn cảm xúc từ mô hình PhoBERT đa nhiệm để điều chỉnh nhãn cuối cùng. Nếu bình luận bị đánh dấu là sarcasm ở một khía cạnh nào đó và nhãn cảm xúc ban đầu là tích cực hoặc trung tính, thì nhãn đó sẽ được chuyển thành tiêu cực. Điều này giúp phản ánh đúng ý nghĩa tiêu cực của bình luận châm biếm, tránh sai lệch trong phân tích cảm xúc. Quá trình này được thực hiện trong hàm `predict_sentiment_smart(text)`, làm cho hệ thống phân tích cảm xúc chính xác và thực tế hơn khi xử lý các bình luận đa dạng, đặc biệt là trên môi trường trực tuyến.

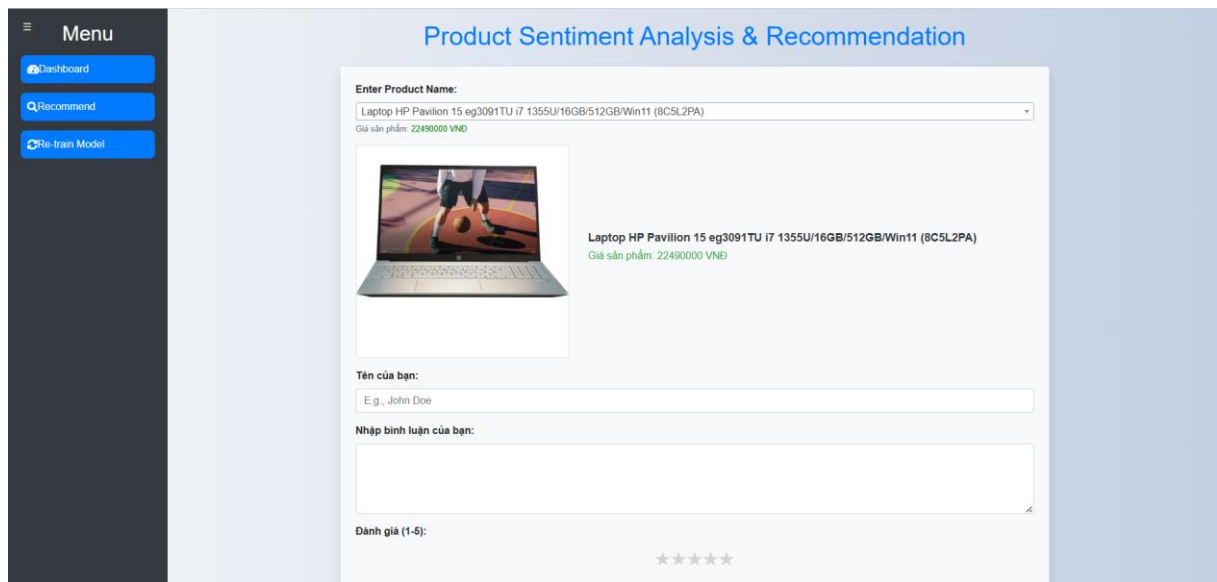


Hình 29: Giao diện phát hiện bình luận ẩn ý sarcasm(khiêu khích).

2.6. Xây dựng hệ thống gợi ý

Trong phần này, nhóm đã xây dựng một hệ thống gợi ý sản phẩm dựa trên phương pháp lọc nội dung (Content-Based Filtering). Hệ thống phân tích ý định người dùng, cảm xúc, và sự tương đồng về đặc điểm sản phẩm. Kết quả được triển khai với mục tiêu cung cấp các đề xuất phù hợp nhất với nhu cầu và sở thích của người dùng.

Giao diện tổng thể của trang web được thiết kế như sau:



Hình 30: Giao diện ban đầu của trang web gợi ý sản phẩm.

Cách hoạt động và các hàm chính của hệ thống được trình bày như sau:

- Phân tích ý định người dùng (infer_user_intent):

```
def load_keywords():
    with open('web_sentiment/Keyword/keyword_occurrences.json', 'r',
              encoding='utf-8') as file:
        keywords = json.load(file)
    return keywords
# Phân loại ý định người dùng
def infer_user_intent(comment):
    keywords = load_keywords()
    comment = comment.lower()
    comment = re.sub(r'^\w\s', '', comment) # Xoá dấu câu

    for intent, words in keywords.items():
        for word in words:
            if word.lower() in comment:
                return intent
    return "balanced"
```

Hàm infer_user_intent được thiết kế để phân tích ý định của người dùng dựa trên bình luận hoặc phản hồi của họ. Hệ thống sẽ tải danh sách từ khóa được phân loại theo các ý định khác nhau (như giá cao, giá thấp hoặc cân bằng) từ tệp keywords.json.

Sau đó, bình luận của người dùng sẽ được kiểm tra xem có chứa từ khóa nào trong danh sách không. Nếu tìm thấy từ khóa phù hợp, ý định tương ứng (như "high_price_keywords", "low_price_keywords", hoặc "balanced") sẽ được trả về. Trong trường hợp không có từ khóa phù hợp, hệ thống sẽ gán ý định mặc định là "balanced".

- Xử lý dữ liệu sản phẩm (process_data):

```
def process_data(df):
    df = df.copy()
    df['user_comment'] = df['user_comment'].fillna('')
    df['combined_sentiments'] = (
        df['Service_Sentiment'].fillna('') + " " +
        df['Price_Sentiment'].fillna('') + " " +
        df['Quality_Sentiment'].fillna('') )
    df['current_price'] =
df['current_price'].fillna(df['current_price'].mean())
    df['average_rating'] =
df['average_rating'].fillna(df['average_rating'].mean())
    scaler = MinMaxScaler()
    df[['normalized_price', 'normalized_rating']] =
scaler.fit_transform(
    df[['current_price', 'average_rating']])
    df['combined_features'] = (
        df['user_comment'] + " " +
        df['combined_sentiments'] + " " +
        df['product_name']
    )
    tfidf_vectorizer = TfidfVectorizer(stop_words='english')
    tfidf_matrix =
tfidf_vectorizer.fit_transform(df['combined_features'])
    quantitative_features = df[['normalized_price',
'normalized_rating']].to_numpy()
    combined_matrix = np.hstack([tfidf_matrix.toarray(),
quantitative_features])
    combined_matrix = np.nan_to_num(combined_matrix)
    cosine_sim = cosine_similarity(combined_matrix)
```

Hàm process_data thực hiện xử lý dữ liệu sản phẩm để chuẩn hóa các đặc điểm và tính toán sự tương đồng giữa các sản phẩm. Đầu tiên, dữ liệu về giá cả (current_price) và đánh giá (average_rating) của sản phẩm được chuẩn hóa về khoảng [0, 1] bằng cách sử dụng MinMaxScaler. Tiếp theo, các đặc điểm của sản phẩm, bao gồm nhận xét của

người dùng (user_comment), cảm xúc (sentiments), và tên sản phẩm (product_name), được kết hợp lại thành một cột combined_features. Để chuyển các đặc điểm văn bản này thành dạng số, TfidfVectorizer được áp dụng để tạo ra ma trận TF-IDF, phản ánh mức độ quan trọng của các từ khóa trong các nhận xét. Sau đó, ma trận TF-IDF và các đặc điểm định lượng (giá và đánh giá) được kết hợp thành một ma trận tổng hợp. Cuối cùng, hàm cosine_similarity được sử dụng để tính toán mức độ tương đồng giữa các sản phẩm dựa trên ma trận tổng hợp này. Kết quả trả về bao gồm dataframe chứa dữ liệu sản phẩm đã xử lý và ma trận tương đồng giữa các sản phẩm, giúp hệ thống có thể tính toán và đưa ra các gợi ý sản phẩm dựa trên mức độ tương đồng giữa chúng.

- Gợi ý sản phẩm (get_recommendations):

```
def get_recommendations(product_name, df, user_intent,
                        service_sentiment, price_sentiment,
                        quality_sentiment, user_rating):
    if "laptop" in product_name.lower():
        df_filtered = df[df["product_name"].str.contains("laptop",
case=False, na=False)]
        product_type = 'laptop'
    else:
        df_filtered = df[~df["product_name"].str.contains("laptop",
case=False, na=False)]
        product_type = 'phone'

    df_filtered = df_filtered.reset_index(drop=True)
    df_filtered_proc, cosine_sim = process_data(df_filtered)
    filtered_index =
df_filtered_proc[df_filtered_proc['product_name'].str.lower() ==
product_name.lower()].index.tolist()
    if not filtered_index:
        return df_filtered_proc.sample(3)[['product_name',
'user_rating', 'current_price', 'average_rating']]
    idx = filtered_index[0]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = [score for score in sim_scores if score[0] != idx]
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    unique_products = set()
    product_indices = []
    base_product = df_filtered_proc.iloc[idx]
    product_price = base_product['current_price']
```

```

product_rating = base_product['average_rating']

def add_if_valid(pname, pid):
    if pname not in unique_products:
        unique_products.add(pname)
        product_indices.append(pid)
for score in sim_scores:
    i = score[0]
    row = df_filtered_proc.iloc[i]
    pname = row['product_name']
    avg_rating = row['average_rating']
    curr_price = row['current_price']

    if service_sentiment == 1:
        passed &= ((user_rating < 5 and 4.0 <= avg_rating <= 4.9)
or (user_rating == 5 and avg_rating >= 4.0))
    elif service_sentiment in [2, 3]:
        passed &= (avg_rating > product_rating)

    if price_sentiment == 1:
        passed &= (4.0 <= avg_rating <= 4.9)
    elif price_sentiment == 2:
        passed &= (avg_rating > user_rating)

    if quality_sentiment == 1:
        passed &= ((user_rating < 5 and 4.0 <= avg_rating <= 4.9)
or (user_rating == 5 and avg_rating >= 4.0))
    elif quality_sentiment in [2, 3]:
        passed &= (avg_rating > product_rating)

    if passed:
        add_if_valid(pname, i)
    if len(product_indices) >= 10:
        break
# GỢI Ý GIÁ RẺ HƠN nếu chê mắc
    if price_sentiment == 3 and user_intent ==
"too_expensive_keywords":
        product_indices = [
            i for i in product_indices
            if df_filtered_proc['current_price'].iloc[i] <
product_price
        ]
    return df_filtered_proc[['product_name', 'user_rating',
'current_price', 'average_rating', 'image_url']].iloc[product_indices]

```

Sau khi phân tích cảm xúc bằng hai mô hình, hệ thống sẽ ghi nhận kết quả từ mô hình Fine-tuned PhoBERT làm kết quả chính vì mô hình này cho độ chính xác cao nhất. Dựa trên kết quả cảm xúc và ý định của người dùng, hệ thống sẽ gợi ý tối đa 10 sản phẩm tương tự. Cụ thể:


- Với cảm xúc về chất lượng (quality_sentiment): Cảm xúc tích cực(Positive) thì hệ thống sẽ ưu tiên gợi các sản phẩm có đánh giá trung bình cao (từ 4.0 trở lên) và tương đương hoặc tốt hơn so với rating mà người dùng dành cho sản phẩm gốc. Cảm xúc trung tính (Neutral) hoặc tiêu cực (Negative), hệ thống ưu tiên sản phẩm có đánh giá trung bình cao hơn, đặc biệt tập trung vào giá trị thực tế và chất lượng
- Với cảm xúc về giá cả của sản phẩm (price_sentiment): Cảm xúc tích cực (Positive) thì hệ thống sẽ ưu tiên gợi các sản phẩm có đánh giá trung bình cao (từ 4.0 trở lên) và tương đương hoặc tốt hơn so với rating mà người dùng dành cho sản phẩm gốc. Cảm xúc trung tính (Neutral) thì hệ thống ưu tiên sản phẩm có đánh giá trung bình cao hơn. Đặc biệt nếu người dùng chê giá mắc (ý định là “too_expensive_keywords” và cảm xúc giá là tiêu cực), hệ thống sẽ lọc ra các sản phẩm có giá thấp hơn sản phẩm gốc.
- Với cảm xúc về dịch vụ(service_sentiment): Logic lọc tương tự như cảm xúc về chất lượng.

Tất cả sản phẩm gợi ý được chọn lọc dựa trên mức độ tương đồng nội dung, kết hợp với các yếu tố như giá hiện tại, đánh giá trung bình và loại sản phẩm. Kết quả phân tích cảm xúc và danh sách gợi ý sẽ được lưu trữ vào cơ sở dữ liệu để phục vụ cho việc tra cứu và cải tiến hệ thống sau này.

Ví dụ, khi bình luận là ‘Sản phẩm tệ, dễ hư hỏng về sau’, rating 2 sao, hệ thống sẽ phân tích cảm xúc của bình luận là Negative(tiêu cực) ở khía cạnh Quality(chất lượng) và gợi ý các sản phẩm có rating trung bình cao hơn và nội dung tương đồng:

Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)

Giá sản phẩm: 22490000 VNĐ



Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)

Giá sản phẩm: 22490000 VNĐ

Tên của bạn:

Khoa

Nhập bình luận của bạn:

Sản phẩm tệ, dễ hư hỏng về sau


Đánh giá (1-5):

★ ★ ★ ★ ★

Recommend

Phân tích cảm xúc của bình luận này

Recommended Products:




Laptop Asus Vivobook 15 X1504VA i7 1355U/16GB/512GB/Win11 (NJ023W)

18,690,000 VNĐ

★ 3.7

Mua ngay




Laptop HP Pavilion 15 eg2088TU i7 1260P/16GB/512GB/Win11 (7C0R0PA)

21,190,000 VNĐ

★ 4.5

Mua ngay




DELL INSPIRON 15 3530

Laptop Dell Inspiron 15 3530 i7 1355U/16GB/512GB/2GB MX550/OfficeH S/Win11 (N3530I716W1)

23,990,000 VNĐ

★ 3.7

Mua ngay



Laptop Dell Inspiron 14 7430 2-in-1 i7 1355U/16GB/512GB/Touch/Pen/ (i7U165W11SLU)

27,990,000 VNĐ

★ 4.7

Mua ngay


Hình 31: Giao diện khi gợi ý sản phẩm với bình luận tiêu cực.

Khi thay đổi bình luận là ‘Sản phẩm giá cao’, rating 2 sao, hệ thống sẽ gợi ý các sản phẩm có giá rẻ hơn và có rating trung bình cao hơn, nội dung tương đồng:

Enter Product Name:

Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)

Giá sản phẩm: 22490000 VNĐ



Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)
Giá sản phẩm: 22490000 VNĐ

Tên của bạn:

Khoa

Nhập bình luận của bạn:

Sản phẩm giá cao


Đánh giá (1-5):

★★★★☆


Recommend

Phân tích cảm xúc của bình luận này


Recommended Products:




Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)
22,290,000 VNĐ
★ 3.0
Mua ngay



Laptop Asus Vivobook 15 X1504VA i7 1355U/16GB/512GB/Win11 (NJ023W)
18,690,000 VNĐ
★ 3.7
Mua ngay



Laptop HP Pavilion 15 eg2088TU i7 1260P/16GB/512GB/Win11 (7C0R0PA)
21,190,000 VNĐ
★ 4.5
Mua ngay



DELL INSPIRON 15 3530
Laptop Dell Inspiron 15 3530 i7 1355U/16GB/512GB/2GB MX550/OfficeHS/Win11 (N3530I716W1)
23,990,000 VNĐ
★ 3.7
Mua ngay

Hình 32: Giao diện khi gợi ý sản phẩm với bình luận tiêu cực và giá cao.

Khi bình luận ‘Sản phẩm tuyệt vời’, rating 4 sao, hệ thống sẽ phân loại là POSITIVE và gợi ý các sản phẩm có rating trung bình tương tự và cao hơn, nội dung tương đồng:

Enter Product Name:

Laptop HP Gaming VICTUS 15 fa1139TX i5 12450H/16GB/512GB/4GB RTX2050/144Hz/Win11 (8Y6W3PA)

Giá sản phẩm: 18490000 VNĐ

Đường nét cuốn hút chuẩn gaming

- Dày 23.5 mm
- Khối lượng 2.29 kg
- Đèn bàn phím đơn sắc

Laptop HP Gaming VICTUS 15 fa1139TX i5 12450H/16GB/512GB/4GB RTX2050/144Hz/Win11 (8Y6W3PA)

Giá sản phẩm: 18490000 VNĐ

Tên của bạn:

Khoa

Nhập bình luận của bạn:

Sản phẩm tuyệt vời

Đánh giá (1-5):

★★★★☆

Recommend

Phân tích cảm xúc của bình luận này

Recommended Products:

Đường nét cuốn hút chuẩn gaming

- Dày 23.5 mm
- Khối lượng 2.29 kg
- Đèn bàn phím đơn sắc

Laptop HP Gaming VICTUS 15 fa1139TX i5 12450H/16GB/512GB/4GB RTX2050/144Hz/Win11 (8Y6W3PA)

19,490,000 VNĐ

★ 4.0

Mua ngay

msi
GF63 Thin 12U

Đường nét cuốn hút chuẩn gaming

- Dày 23.5 mm
- Khối lượng 2.29 kg
- Đèn bàn phím đơn sắc

Laptop MSI Gaming GF63 Thin 12UCX i5 12450H/16GB/512GB/4GB RTX2050/144Hz/Win11 (873VN)

15,490,000 VNĐ

★ 4.0

Mua ngay

Đường nét cuốn hút chuẩn gaming

- Dày 23.5 mm
- Khối lượng 2.29 kg
- Đèn bàn phím đơn sắc

Laptop Lenovo Gaming LOQ 15IAX9 i5 12450HX/16GB/512GB/6GB RTX3050/144Hz/Win11 (83GS000JVN)

21,690,000 VNĐ

★ 4.2

Mua ngay

acer
NITRO

Đường nét cuốn hút chuẩn gaming

- Dày 23.5 mm
- Khối lượng 2.29 kg
- Đèn bàn phím đơn sắc

Laptop Acer Gaming Nitro 5 Tiger AN515 58 769J i7 12700H/8GB/512GB/4GB RTX3050/144Hz/Win11 (NH.QFHSV.003)

21,490,000 VNĐ

★ 4.5

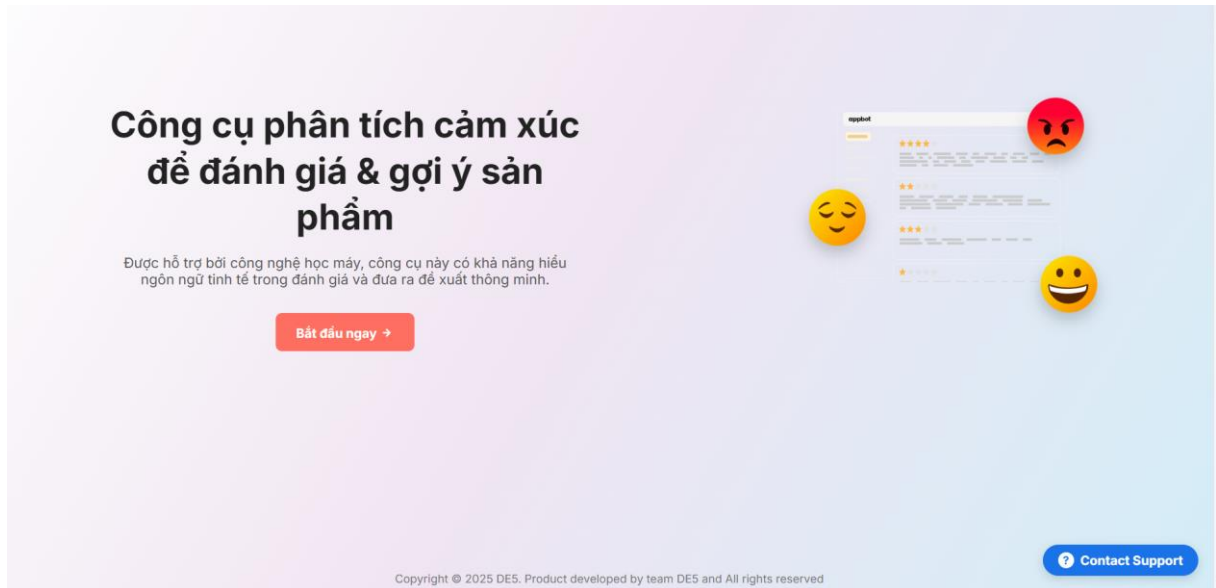
Mua ngay

Hình 33: Giao diện khi gợi ý sản phẩm với bình luận tích cực.

80

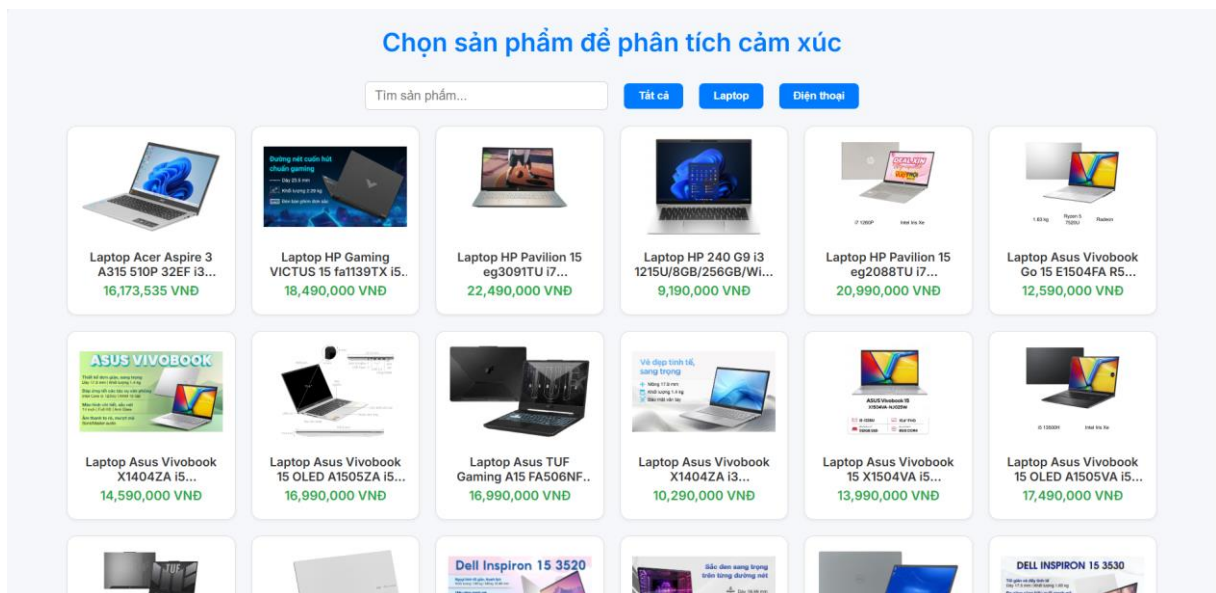
2.7. Giao diện hệ thống

Khi chạy hệ thống, giao diện trang home sẽ hiển thị để người dùng bắt đầu chương trình:



Hình 34: Giao diện trang home

Sau khi bắt đầu chương trình, giao diện sẽ hiển thị một danh sách các sản phẩm nhằm giúp người dùng lựa chọn sản phẩm mà họ muốn mua để thực hiện việc bình luận và phân tích cảm xúc. Người dùng có thể cuộn và chọn sản phẩm phù hợp để tiếp tục quy trình phân tích cảm xúc cho sản phẩm đó:



Hình 35: Giao diện trang chọn sản phẩm


Sau khi người dùng chọn một sản phẩm bất kỳ từ danh sách, hệ thống sẽ hiển thị giao diện để nhập thông tin phản hồi về sản phẩm. Giao diện này bao gồm: Thông tin

sản phẩm, tên người dùng, bình luận, đánh giá và hai lựa chọn: nút “Recommend” để gửi đánh giá và đề xuất sản phẩm và nút “Phân tích cảm xúc của bình luận này” giúp phân tích nội dung bình luận nhằm xác định cảm xúc.

Product Sentiment Analysis & Recommendation

Product Name:

Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)
Giá sản phẩm: 22450000 VND



Laptop HP Pavilion 15 eg3091TU i7 1355U/16GB/512GB/Win11 (8C5L2PA)
Giá sản phẩm: 22450000 VND

Tên của bạn:
E.g. John Doe

Nhập bình luận của bạn:

Đánh giá (1-5):
☆☆☆☆☆

Recommend

Phân tích cảm xúc của bình luận này

Hình 36: Giao diện trang nhập thông tin

Sau khi điền đủ thông tin và chọn “Recommend”, hệ thống sẽ gợi ý các sản phẩm bên dưới, người dùng có thể cuộn ngang sang phải để xem toàn bộ danh sách sản phẩm gợi ý:

Đức

Nhập bình luận của bạn:


sản phẩm khá tốt

Đánh giá (1-5):
☆☆☆☆☆


Recommend

Phân tích cảm xúc của bình luận này


Recommended Products:




Laptop Dell Inspiron 14 7430
2-in-1 i7
1355U/16GB/512GB/Touch/Pen/IO
(I7U165W11SLU)
27,390,000 VND
☆ 4.7
Mua ngay



Laptop Lenovo LOQ Gaming
15AX9 i5
12450HX/16GB/512GB/6GB
RTX3050/144Hz/Win11
(83GS000JVN)
21,690,000 VND
☆ 4.3
Mua ngay



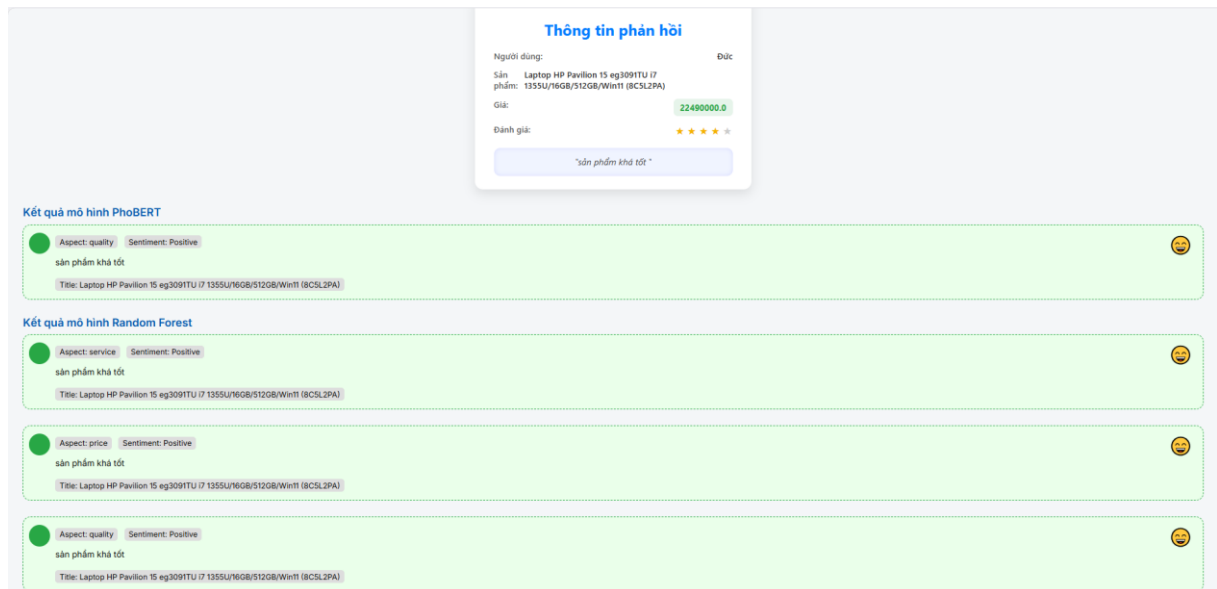
Laptop Acer Gaming Nitro 5
Tiger AN515 58 768J i7
12700H/16GB/512GB/4GB
RTX3050/144Hz/Win11
(NH.QFHSV.003)
21,490,000 VND
☆ 4.5
Mua ngay



Laptop Lenovo Gaming LOQ
15AX9 i5
12450HX/16GB/512GB/6GB
RTX3050/144Hz/Win11
(83GS000JVN)
21,690,000 VND
☆ 4.3
Mua ngay

Hình 37: Giao diện gợi ý sản phẩm

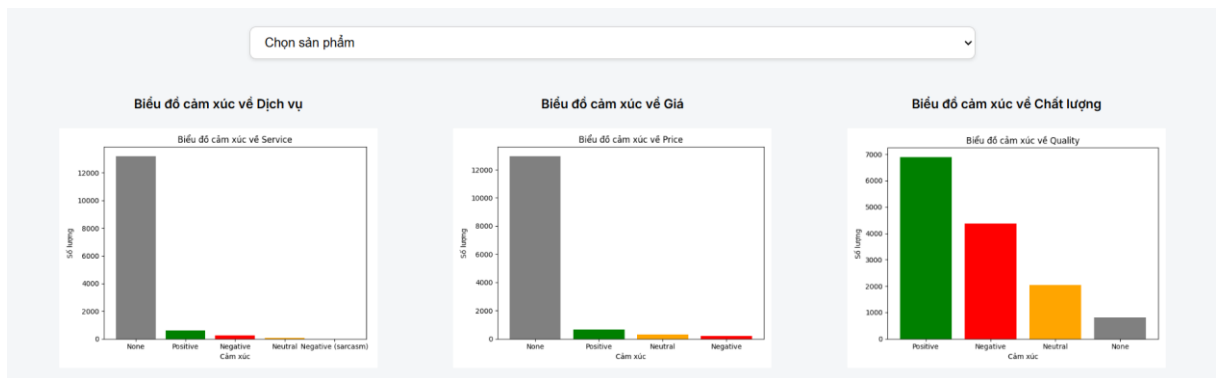
Nếu chọn nút “Phân tích cảm xúc của bình luận này”, hệ thống sẽ hiển thị giao diện phân tích cảm xúc bao gồm thông tin phản hồi và kết quả phân tích của hai mô hình PhoBERT và Random Forest:



Hình 38: Giao diện phân tích cảm xúc

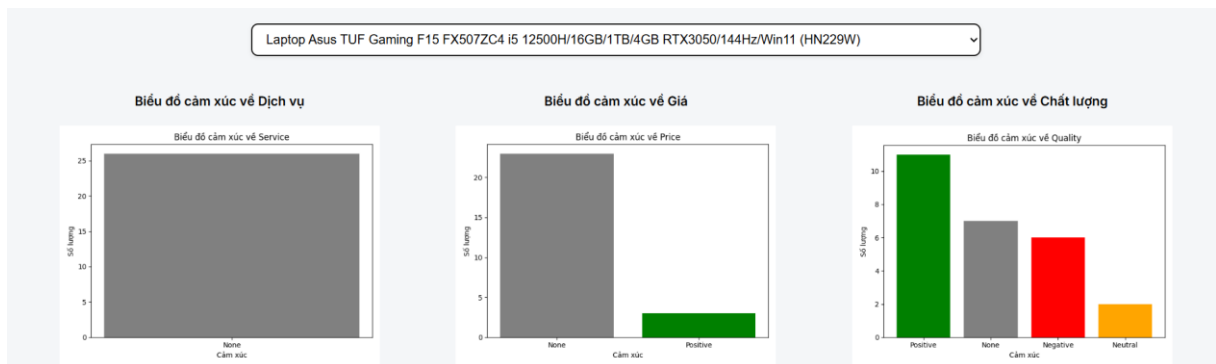
Để giúp người dùng dễ dàng quan sát và đánh giá tổng quan cảm xúc của khách hàng về các khía cạnh của sản phẩm, hệ thống cung cấp các biểu đồ trực quan hóa dữ liệu theo từng khía cạnh cụ thể bao gồm: dịch vụ (Service), giá cả (Price) và chất lượng (Quality).

Biểu đồ tổng quan (Toàn bộ sản phẩm): Biểu đồ này thể hiện số lượng các loại cảm xúc (tích cực, tiêu cực, trung lập) tương ứng với mỗi khía cạnh của tất cả sản phẩm trong hệ thống. Người dùng có thể thấy được xu hướng chung của khách hàng đối với dịch vụ, giá cả và chất lượng của các sản phẩm đã được đánh giá.



Hình 39: Biểu đồ ban đầu thể hiện các cảm xúc của toàn bộ sản phẩm.

Biểu đồ theo từng sản phẩm: Khi người dùng chọn một sản phẩm cụ thể từ danh sách, hệ thống sẽ hiển thị lại các biểu đồ cảm xúc, nhưng chỉ áp dụng cho sản phẩm đã chọn. Nhờ đó, người dùng có thể phân tích chi tiết mức độ hài lòng của khách hàng theo từng khía cạnh, hỗ trợ việc ra quyết định mua hàng hoặc cải thiện sản phẩm phù hợp hơn.



Hình 40: Biểu đồ thể hiện các kết quả cảm xúc theo bình luận của từng sản phẩm


Bên cạnh việc trực quan hóa dữ liệu bằng biểu đồ, hệ thống còn cung cấp chức năng lọc bình luận theo từng khía cạnh và cảm xúc cụ thể. Người dùng có thể chọn sản phẩm mong muốn, sau đó lựa chọn khía cạnh (dịch vụ, giá cả, chất lượng) và loại cảm xúc (tích cực, tiêu cực, trung lập) để xem chi tiết các bình luận tương ứng.

Giao diện lọc bình luận được thiết kế trực quan, dễ sử dụng. Sau khi chọn bộ lọc phù hợp và nhấn nút "Lọc bình luận", hệ thống sẽ hiển thị toàn bộ các bình luận của người dùng theo đúng điều kiện đã chọn, kèm theo thông tin sản phẩm (tên, hình ảnh và giá bán hiện tại).

Lọc bình luận theo cảm xúc và khía cạnh

Chọn khía cạnh:

Chọn cảm xúc:



Laptop Asus Vivobook 15 X1504VA i5 1335U/8GB/512GB/Win11 (NJ025W)
Giá: 13.990.000đ

Quality - neutral

- mua dc tuần thì máy bị lỗi cảm ứng chuột và đã dc đổi mới
- mua dc tuần thì máy bị lỗi cảm ứng chuột và đã dc đổi mới

Quality - negative

- phát hiện ra máy bị lỗi đem tới cửa hàng gửi ngày bảo máy k bị gì hết đem về sử dụng vẫn bị lỗi
- phát hiện ra máy bị lỗi đem tới cửa hàng gửi ngày bảo máy k bị gì hết đem về sử dụng vẫn bị lỗi

Quality - positive

- tốt
- okie
- sạc nhanh đẹp đ
- máy chạy rất mượt và thích hợp cho nhiều mục đích
- mới thì dùng ok để dùng thêm thời gian xem thế nào
- máy sử dụng rất ok quản lý và nhân viên rất thân thiện dễ thương

Hình 41: Bộ lọc các bình luận theo các khía cạnh và cảm xúc cho từng sản phẩm.

Việc lọc chi tiết này giúp người dùng hoặc nhà quản trị dễ dàng đánh giá chất lượng sản phẩm từ nhiều góc nhìn khác nhau, phát hiện sớm các vấn đề tiêu cực, cũng như nắm bắt được điểm mạnh cần phát huy.

CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ

3.1. Dữ liệu thực nghiệm

Finish_data.csv là tập dữ liệu được Crawl từ các trang web Điện máy xanh và thế giới di động.

Emotional_classification.csv là tập dữ liệu đã được xử lý ngôn ngữ tự nhiên.

Emotional_by_aspect.csv là tập dữ liệu cuối cùng đã được gán nhãn từ mô hình PhoBert.

Sarcasm_comment.csv là tập dữ liệu tổng hợp các câu bình luận mang tính mỉa mai nhằm training cho mô hình phát hiện sarcasm.

Tập dữ liệu này chứa các thông tin về các sản phẩm và các đánh giá của người dùng.

Mô tả:

Tập dữ liệu gồm 15 cột và 14101 hàng.

Bảng mô tả dữ liệu **Emotional_by_aspect.csv** :

Tên cột	Mô tả	Kiểu dữ liệu
Product Name	Tên sản phẩm (unique)	nvarchar(255)
Product Name_ID	Số thứ tự tên sản phẩm	nvarchar(255)
Current Price	Giá sản phẩm (unique)	decimal(18, 2)
Reviewer	Tên người dùng	nvarchar(255)
Reviewer_ID	Số thứ tự tên người dùng	nvarchar(255)
Rating	Đánh giá của người dùng (thuộc đoạn [0,5])	int
Rating Average	Đánh giá trung bình của sản phẩm	float
Comment	Bình luận của người dùng	nvarchar(MAX)
Time Used	Số ngày sử dụng	int
Cleaned Comment	Comment của người dùng sau khi quá trình xử lý dữ liệu	nvarchar(MAX)

Tokenized Comment	Comment của người dùng tách mã	nvarchar(MAX)
Filtered Comment	Comment của người dùng sau khi loại bỏ từ dừng	nvarchar(MAX)
Service Sentiment	Phân loại cảm xúc theo dịch vụ	nvarchar(55)
Price Sentiment	Phân loại cảm xúc theo giá	nvarchar(55)
Quality Sentiment	Phân loại cảm xúc theo chất lượng	nvarchar(55)

3.2. Môi trường thực nghiệm

3.2.1. Thư viện sử dụng

Mô tả thư viện cho việc crawl dữ liệu :

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.keys import Keys
import time
import pandas as pd
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import NoSuchElementException
import os
from bs4 import BeautifulSoup
```

- + Thư viện **Selenium**: Tự động hóa trình duyệt web, tương tác với các phần tử trang web (click, nhập dữ liệu, cuộn trang) và chờ tải nội dung.
- + Thư viện **Time**: Tạm dừng chương trình với từng thời gian cụ thể hoặc đo thời gian thực hiện quy trình.
- + Thư viện **Pandas**: Xử lý và lưu trữ dữ liệu thu thập được dưới dạng các DataFrame , xuất ra tệp CSV hoặc Excel.
- + Thư viện **OS**: Quản lý tệp và thư mục trên hệ thống (tạo, kiểm tra, xóa).

+ Thư viện **BeautifulSoup**: Phân tích và trích xuất dữ liệu từ mã nguồn HTML của trang web.

Mô tả thư viện cho việc xử lý ngôn ngữ tự nhiên:

```
import re
import string
from pyvi import ViTokenizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer
from transformers import AutoTokenizer,
AutoModelForSequenceClassification, pipeline
```

+ Thư viện **Regular Expressions** : giúp xử lý chuỗi bằng cách sử dụng biểu thức chính quy như là loại bỏ các ký tự không mong muốn.

+ Thư viện **String** : giúp cung cấp các hằng số và phương thức xử lý chuỗi và kết hợp với thư viện re để tiến hành làm sạch dữ liệu.

+ Thư viện **Natural Language Toolkit** cung cấp công cụ xử lý ngôn ngữ tự nhiên gồm: Stopwords ,Tokenize và WordNet Lemmatizer.

+ Thư viện **Scikit-learn** : giúp trích xuất các đặc trưng từ văn bản để sử dụng trong các mô hình Machine Learning gồm: CountVectorizer và TfidfVectorizer

+ Thư viện **Transformers**: Cung cấp các mô hình học sâu hiện đại cho xử lý ngôn ngữ tự nhiên gồm: Pipeline để triển khai các tác vụ NLP phổ biến như phân loại,..., AutoTokenizer và Auto Model For Sequence Classification giúp cho tải mô hình học sâu dùng cho phân loại văn bản.

Mô tả thư viện cho việc thực hiện mô hình học máy:

```
import pickle
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import classification_report, accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
```

- + Thư viện **Pickle** : giúp lưu trữ và tải dữ liệu/mô hình dưới dạng nhị phân.
- + Thư viện **Scikit-learn** gồm:
 - `train_test_split`: Chia dữ liệu thành tập huấn luyện và kiểm tra.
 - `classification_report`, `accuracy_score`: Đánh giá mô hình đưa ra các độ chính xác hay các chỉ số có liên quan.
 - Logistic Regression: Mô hình phân loại tuyến tính.
 - `RandomForestClassifier`: Thuật toán phân loại dựa trên cây quyết định ngẫu nhiên.
- + Thư viện **tensorflow.keras** : giúp xây dựng và huấn luyện các mô hình học sâu gồm:
 - `Tokenizer`: Biểu diễn văn bản dưới dạng chuỗi số nguyên.
 - `pad_sequences`: Làm đầy/truncate chuỗi để đảm bảo cùng độ dài.
 - `Sequential`: Xây dựng mô hình tuần tự theo từng lớp một.
 - `Embedding`: Biểu diễn từ dưới dạng vector nhúng.
 - `LSTM`,: Các lớp mạng hồi tiếp .
 - `Dense`: Lớp kết nối đầy đủ.
 - `Dropout`: Giảm overfitting bằng cách ngắt kết nối ngẫu nhiên.

Mô tả thư viện cho việc thực hiện tinh chỉnh PhoBert:

```
import torch
import torch.nn as nn
from torch.utils.data import DataLoader, Dataset
from transformers import AutoModel, AutoTokenizer, AdamW
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from tqdm import tqdm
```

- + Thư viện **torch** giúp xây dựng và huấn luyện mô hình học sâu cụ thể là cung cấp các hàm và cấu trúc tensor để tính toán
- + Thư viện **torch.utils.data** gồm `dataset` là lớp cơ sở tạo tập dữ liệu hoàn chỉnh và `dataloader` tải dữ liệu từ dataset trên theo batch.

- + Thư viện **transformers** giúp cung cấp các mô hình hiện đại phoBert và nhiều công cụ xử lý ngôn ngữ tự nhiên gồm AutoKenizen, AutoModel và AdamW là bộ tối ưu hóa để điều chỉnh trọng số.
- + Thư viện **tqdm** giúp hiển thị thanh tiến trình cho các batch trong tập huấn luyện, giúp giám sát tiến độ của quá trình.

3.2.2. Cấu hình máy thực nghiệm

3.3. Mô hình thực nghiệm

Ở đây nhóm chúng em thực hiện trên bốn mô hình chính gồm Logistic Regression, Random Forest Classifier, Long Short Term Memory và PhoBert.

Đầu tiên với mô hình học máy Logistic Regression nhóm thấy mô hình này đơn giản và nhanh chóng, nhưng nó không xử lý tốt dữ liệu không cân bằng và các lớp ít xuất hiện, mô hình gặp khó khăn trong việc phân loại các bình luận tiêu cực, đặc biệt khi dữ liệu không cân bằng.

Đối với mô hình Random Forest Classifier nhóm thấy mô hình có ưu điểm rõ ràng trong việc xử lý các vấn đề với dữ liệu không cân bằng và các lớp có số lượng mẫu ít. Đem lại kết quả tốt và độ chính xác khá tốt với tập dữ liệu mà nhóm thực nghiệm và vượt trội hơn so với mô hình Logistic Regression.

Đối với mô hình Long Short Term Memory nhóm thấy LSTM rất mạnh mẽ trong việc học từ dữ liệu chuỗi, nhưng yêu cầu tài nguyên tính toán lớn và thời gian huấn luyện lâu hơn. Trong tập dữ liệu thực nghiệm này, LSTM có thể xử lý ngữ cảnh tốt hơn.

Cuối cùng đối với mô hình phoBert nhóm thấy so với các mô hình truyền thống như Logistic Regression hay LSTM, PhoBERT thể hiện vượt trội nhờ khả năng tận dụng ngữ cảnh tốt hơn thông qua cơ chế attention và pre-training trên ngôn ngữ tiếng Việt. Tốc độ xử lý rất tốt với dữ liệu thực nghiệm của nhóm.

3.4. Kết quả thực nghiệm

Model	Joint Accuracy	Accuracy - Price	Accuracy - Quality	Accuracy - Service	Macro F1 - Price	Macro F1 - Quality	Macro F1 - Service	Avg Macro F1
Logistic Regression	83%	83%	89%	94%	65%	60%	49%	58%
Random Forest	80%	94%	94%	88%	91%	94%	47%	77%
LSTM	61%	83%	66%	94%	65%	26%	49%	47%
Fin-PhoBERT	89%	98%	93%	98%	84%	88%	64%	79%

Hình 42: Bảng kết quả thực nghiệm.

- **Logistic Regression:** Mô hình đơn giản, dễ triển khai và đạt Joint Accuracy 83%, thể hiện hiệu suất khá ổn định trên cả ba khía cạnh. Tuy nhiên, Macro F1-score trung bình chỉ đạt 0.58, cho thấy khả năng phân biệt các cảm xúc chưa đều, đặc biệt yếu ở khía cạnh Service ($F1 = 0.48$) mô hình này phù hợp với các hệ thống đơn giản, không yêu cầu hiểu sâu ngữ cảnh.
- **Random Forest:** Với khả năng xử lý tốt các quan hệ phi tuyến tính, mô hình Random Forest đạt Joint Accuracy 80%, nhưng bù lại có Macro F1 trung bình 0.77, cao hơn hẳn Logistic Regression. Mô hình thể hiện rõ ưu thế ở khía cạnh Price ($F1 = 0.91$) và Quality ($F1 = 0.95$), tuy nhiên vẫn còn hạn chế ở Service ($F1 = 0.47$). Precision cao cho thấy độ tin cậy trong dự đoán, nhưng Recall chưa đủ mạnh để bao phủ hết các nhãn hiếm. Đây là mô hình khá cân bằng giữa hiệu suất và tính khả dụng.
- **LSTM:** Có khả năng học chuỗi và ngữ cảnh tốt, nhưng trong thực tế đạt Joint Accuracy chỉ 61%, thấp nhất trong các mô hình. Macro F1 trung bình là 0.47, chủ yếu do hiệu suất rất thấp ở Quality ($F1 = 0.26$). Dù có kết quả khá tốt ở Price và Service, mô hình LSTM gặp khó khăn khi lớp dữ liệu không cân bằng. LSTM không phải lựa chọn lý tưởng trong trường hợp này.
- **PhoBERT (Fine-tuned):** Là mô hình mạnh mẽ nhất trong tất cả, PhoBERT đạt Joint Accuracy lên đến 89% và Accuracy trung bình 96% trên cả 3 task. Macro

F1-score trung bình là 0.79, với hiệu suất cao đồng đều ở cả Price ($F1 = 0.84$), Quality ($F1 = 0.88$) và Service ($F1 = 0.64$). Nhờ được huấn luyện sẵn trên dữ liệu tiếng Việt và fine-tuning kỹ lưỡng, PhoBERT thể hiện khả năng nhận diện cảm xúc chính xác và hiệu năng vượt trội. Đây là lựa chọn lý tưởng cho các hệ thống yêu cầu độ chính xác cao và xử lý ngôn ngữ tự nhiên chuyên sâu.

Kết Luận:

Sau quá trình thực nghiệm và so sánh, nhóm quyết định chọn PhoBERT (Fine-tuned) làm mô hình chính cho hệ thống vì hiệu suất vượt trội, và sử dụng Random Forest làm mô hình phụ để đối chiếu nhằm so sánh hiệu quả của hai mô hình. Trong ba mô hình truyền thống (Random Forest, Logistic Regression, LSTM), nhóm chọn Random Forest vì đây là lựa chọn cân bằng nhất giữa độ chính xác, khả năng khái quát, chi phí tính toán và độ ổn định vượt trội hơn Logistic Regression (quá đơn giản) và LSTM (nặng và kém hiệu quả trong bài toán này).

PHẦN 3: KẾT LUẬN

3.1. Kết quả đạt được

Sau khi hoàn thành dự án, nhóm chúng em đã tiếp thu được những kiến thức cốt lõi về phân tích cảm xúc và hệ thống gợi ý, đặc biệt là khả năng áp dụng các mô hình học máy truyền thống và học sâu, nổi bật là PhoBERT, vào bài toán thực tiễn. Chúng em cũng hiểu rõ hơn về cách các sàn thương mại điện tử triển khai hệ thống gợi ý sản phẩm hiệu quả. Thông qua việc trực quan hóa dữ liệu trên giao diện web, nhóm đã khám phá được nhiều insight giá trị từ dữ liệu của Thế Giới Di Động và Điện Máy Xanh, mở ra tiềm năng áp dụng kinh nghiệm này để đề xuất các chiến lược phát triển sản phẩm và tăng trưởng doanh số cho doanh nghiệp trong tương lai.

3.2. Hạn chế

Trong quá trình thực hiện hệ thống phân tích cảm xúc và gợi ý sản phẩm, nhiều thách thức đã xuất hiện ở các giai đoạn khác nhau. Đầu tiên, dữ liệu cảm xúc được thu thập từ nhiều nguồn như các trang thương mại điện tử, mỗi nguồn có định dạng khác nhau, khiến việc hợp nhất và xử lý dữ liệu trở nên phức tạp. Ngoài ra, dữ liệu thường chứa nhiều nhiễu, bao gồm các từ ngữ không liên quan, biểu tượng cảm xúc, hoặc các yếu tố phi cấu trúc, làm giảm độ chính xác khi phân tích.

Ngôn ngữ tiếng Việt cũng là một thách thức lớn. Tiếng Việt mang tính mơ hồ, đa nghĩa và nhạy cảm với ngữ cảnh, khiến việc phân loại cảm xúc trở nên phức tạp. Bên cạnh đó, ngôn ngữ địa phương hoặc tiếng lóng thường xuất hiện trong các đánh giá từ người dùng, làm tăng thêm độ khó trong việc xử lý dữ liệu văn bản và xây dựng mô hình phân tích hiệu quả.

Trong giai đoạn gợi ý sản phẩm, việc kết hợp phân tích cảm xúc với các phương pháp gợi ý truyền thống như Collaborative Filtering hoặc Content-based Filtering gặp nhiều khó khăn. Mô hình không chỉ cần phân tích được cảm xúc mà còn phải hiểu và tích hợp thông tin từ nhiều nguồn khác nhau để đưa ra các gợi ý phù hợp.

Bên cạnh các thách thức kỹ thuật, yêu cầu về bảo vệ dữ liệu cá nhân và tuân thủ các quy định, như Quy định chung về bảo vệ dữ liệu (GDPR), cũng đặt ra áp lực lớn. Việc xử lý và lưu trữ dữ liệu khách hàng cần đảm bảo an toàn, minh bạch và không vi phạm

quyền riêng tư, đòi hỏi áp dụng các biện pháp bảo mật chặt chẽ trong toàn bộ quy trình.

3.3. Hướng phát triển

Hệ thống hỗ trợ phân tích bình luận từ khách hàng bằng nhiều ngôn ngữ khác nhau, tiếng Anh và các ngôn ngữ khác, nhằm đáp ứng nhu cầu đa dạng của người dùng.

Bên cạnh phân tích văn bản, hệ thống còn có thể tích hợp khả năng phân tích cảm xúc từ hình ảnh do người dùng tải lên, bao gồm ảnh sản phẩm kèm đánh giá hoặc biểu cảm khuôn mặt trong video và ảnh, giúp mở rộng phạm vi phân tích và cung cấp cái nhìn toàn diện hơn về cảm xúc khách hàng.

Tích hợp thêm Chatbox để đa dạng hơn về cung cấp thông tin cho người dùng đối với từng loại sản phẩm.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Lê Sĩ Lắc, "A Research on Sentiment Analysis", 2021.
- [2] Agustini, "Sentiment Analysis on Social Media using Machine Learning-Based Approach", June 2021.
- [3] Shashank Kalluri, "Deep Learning Based Sentiment Analysis", January 2023.
- [4] Doaa Mohey El-Din Mohamed Hussein, "Analyzing Scientific Papers Based on Sentiment Analysis", June 2016.
- [5] Ngoc C. Le, Nguyen The Lam, Son Hong Nguyen, Duc Thanh Nguyen, "On Vietnamese Sentiment Analysis: A Transfer Learning Method", July 2020.
- [6] Bo Pang and Lillian Lee, "Opinion mining and Sentiment analysis", 2007.
- [7] Binh Thanh Kieu and Son Bao Pham, "Sentiment analysis for Vietnamese", 2010.
- [8] Nguyen Thi Duyen, Ngo Xuan Bach and Tu Minh Phuong, "An Empirical Study on Sentiment analysis for Vietnamese", 2014.
- [9] Vi Ngo Van, Minh Hoang Van, Tam Nguyen Thanh, "Sentiment analysis for Vietnamese using Support Vector Machines with application on Facebook comments", Proc. VLSP 2016.
- [10] Thien Khai Tran and Tuoi Thi Phan, "Computing Sentiment Scores of Verb Phrases for Vietnamese", 2016.
- [11] [Hochreiter & Schmidhuber\(1997\)](https://www.bioinf.jku.at/publications/older/2604.pdf) ,”LONG SHORT-TERM MEMORY”,SEP 1997, <https://www.bioinf.jku.at/publications/older/2604.pdf>
- [12] Viblo, Tổng quan về Recommender System (Recommender System cơ bản - Phần 1), 2020. Link:<https://viblo.asia/p/tong-quan-ve-recommender-system-recommender-system-co-ban-phan-1-924IJGBb5PM>
- [13] Phạm Đình Khánh, Lý thuyết về mạng LSTM, 2019. Link: https://phamdinhhkhanh.github.io/2019/04/22/Ly_thuyet_ve_mang_LSTM.html
- [14] Phạm Đình Khánh, PhoBERT và Fairseq: Hiểu và ứng dụng, 2020. Link: https://phamdinhhkhanh.github.io/2020/06/04/PhoBERT_Fairseq.html
- [15] Chung Pham Van, Logistic Regression: Bài toán cơ bản trong Machine Learning, 2020. Link: <https://viblo.asia/p/logistic-regression-bai-toan-co-ban-trong-machine-learning-924IJ4rzKPM>

- [16] Lê Quang Hòa - Sentiment Analysis Using BERT (Hugging Face), 2020. Link: <https://www.slideshare.net/slideshow/sentiment-analysis-using-bert-hugging-face/269451286>
- [17] Viblo - BERT, RoBERTa, PhoBERT, BERTweet: Ứng dụng state-of-the-art pre-trained model cho bài toán phân loại văn bản*, 2020. Link: <https://viblo.asia/p/bert-roberta-phobert-bertweet-ung-dung-state-of-the-art-pre-trained-model-cho-bai-toan-phan-loai-van-ban-4P856PEWZY3>
- [18] GitHub - *Sentiment and Sarcasm Detection using Multi-Task Learning (MTL)*, 2023. Link: <https://github.com/yikyang99/sentiment-sarcasm-using-MTL>
- [19] Hugging Face - *SarcasmNet Dataset for Sarcasm Detection*. Link: <https://huggingface.co/datasets/SarcasmNet/sarcasm>
- [20] Papers with Code - *iSarcasm: A Dataset for Sarcasm Detection in Arabic*. Link: <https://paperswithcode.com/dataset/isarcasm>
- [21] GeeksforGeeks. (n.d.). *F1 Score in Machine Learning*. Retrieved June 5, 2025, from <https://www.geeksforgeeks.org/f1-score-in-machine-learning>
- [22] Trần Văn, H. (n.d.). *Multi-label classification*. Retrieved June 5, 2025, from <https://huytranvan2010.github.io/Multi-label-classification/>

PHỤ LỤC

Phụ lục A: Mã nguồn và mô hình huấn luyện

- Toàn bộ mã nguồn và mô hình huấn luyện trong quá trình thực hiện đề tài được lưu trữ tại liên kết sau:

Google Drive:

<https://drive.google.com/file/d/1YmirX2UzzXW7Vmph85urbHRFtV1jOlah/view>

Gồm các tệp:

- Thư mục cấu hình là chứa các tệp cấu hình hệ thống, thông số kết nối, biến môi trường
- Mô hình học máy chứa mã huấn luyện mô hình học máy/mô hình deep learning để phân tích cảm xúc hoặc gợi ý sản phẩm.
- Xử lý ngôn ngữ tự nhiên gồm các công cụ và mô hình NLP như PhoBERT, TF-IDF, Tokenizer ,...
- Mô hình gán nhãn và fine-tuning PhoBert
- Dữ liệu chứa dữ liệu gốc và dữ liệu đã xử lý từ hệ thống Điện Máy Xanh và Thế giới di động.
- Trích xuất dữ liệu gồm các đoạn mã để thu thập và trích xuất dữ liệu từ các nguồn khác nhau (CSDL, API...).
- Tiền xử lý và phân tích: gồm tiền xử lý, trực quan hóa và kiểm tra dữ liệu, script chuẩn hóa văn bản hoặc dữ liệu dạng bảng.
- Tải dữ liệu các module để tải dữ liệu vào hệ thống (ví dụ: vào CSDL hoặc dashboard).

Phụ lục B: File Sao Lưu Hệ Thống

- File này chứa **bản sao lưu toàn bộ hệ thống** bao gồm:
 - Thông tin các bảng trong Database gồm: Devices và Products

Google Drive:

<https://drive.google.com/file/d/18YpaY9EI0xJOP2EZVv0koh77Jo4-jhr8/view>