

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

PHẠM ĐỨC DUY

ĐATN ĐẠI HỌC NGÀNH KHOA HỌC MÁY TÍNH

**PHÂN LOẠI CẢM XÚC NGƯỜI DÙNG MẠNG X THEO THỜI
GIAN THỰC BẰNG APACHE SPARK MLLIB**

GVHD: TS. Nguyễn Mạnh Cường

Sinh viên: Phạm Đức Duy

Mã số sinh viên: 2021602655

KHOA HỌC MÁY TÍNH

Hà Nội – Năm 2025

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

=====*=====



ĐỒ ÁN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH

**ĐỀ TÀI: PHÂN LOẠI CẢM XÚC NGƯỜI DÙNG
MẠNG X THEO THỜI GIAN THỰC BẰNG APACHE
SPARK MLLIB**

Giảng viên hướng dẫn : TS. Nguyễn Mạnh Cường

Sinh viên thực hiện : Phạm Đức Duy

Mã sinh viên : 2021602655

Lớp : KHMT01 – K16

Hà Nội, 5/2025

MỤC LỤC

DANH MỤC NHỮNG TỪ VIẾT TẮT	3
DANH MỤC BẢNG BIỂU	4
DANH MỤC HÌNH ẢNH.....	5
LỜI CẢM ƠN	8
LỜI NÓI ĐẦU.....	9
CHƯƠNG 1: PHÁT BIỂU BÀI TOÁN	11
1.1. Tổng quan về xử lý ngôn ngữ tự nhiên	11
1.1.1. Định nghĩa	11
1.1.2. Thực trạng	12
1.1.3. Tác vụ.....	13
1.2. Tổng quan về phân tích cảm xúc.....	23
1.3. Phát biểu bài toán	24
1.3.1. Định nghĩa bài toán.....	24
1.3.2. Phân tích yêu cầu.....	24
1.3.3. Phạm vi và giới hạn của bài toán	25
1.3.4. Khó khăn và thách thức	26
1.3.5. Các ứng dụng của bài toán	26
CHƯƠNG 2: MỘT SỐ KỸ THUẬT GIẢI QUYẾT BÀI TOÁN	28
2.1. Phương hướng tiếp cận bài toán.....	28
2.2. Hướng tiếp cận học máy	28
2.2.1. Logistic regression.....	28
2.2.2. SVM (Support Vector Machine)	31
2.3. Hướng tiếp cận học sâu.....	33
2.3.1. CNN (Convolutional Neural Network)	33
2.3.2. LSTM (Long Short-Term Memory).....	37
2.4. Các nghiên cứu giải quyết bài toán tiêu biểu.....	40
CHƯƠNG 3: APACHE SPARK MLLIB VÀ CÁC CÔNG CỤ HỖ TRỢ TRIỂN KHAI.....	42
3.1. Tổng quan về Apache Spark MLlib	42
3.2. Ưu điểm và lý do lựa chọn Spark Mllib	43
3.3. Các thuật toán được hỗ trợ trong Spark Mllib	44

3.3.1. Thuật toán phân loại	44
3.3.2. Các kỹ thuật tiền xử lý và trích xuất đặc trưng	45
3.3.3. Đánh giá mô hình	45
3.3.4. Tuning siêu tham số.....	45
3.4. Mô hình Logistic Regression với Spark MLlib	46
3.5. Các công nghệ hỗ trợ triển khai	48
3.5.1. Docker	48
3.5.2. Apache Kafka	49
3.5.3. Spark Streaming	52
3.5.4. MongoDB.....	53
3.5.5. Django	55
CHƯƠNG 4: THỰC NGHIỆM VÀ XÂY DỰNG SẢN PHẨM DEMO	58
4.1. Dữ liệu thực nghiệm	58
4.2. Tiền xử lý dữ liệu	58
4.3. Huấn luyện mô hình	59
4.4. Các kết quả thực nghiệm.....	60
4.5. Phân tích thiết kế hệ thống.....	63
4.5.1. Kiến trúc hệ thống	63
4.5.2. Biểu đồ use case	65
4.5.3. Đặc tả use case.....	65
4.5.4. Phân tích các use case	67
4.6. Xây dựng phần mềm demo	69
4.6.1. Cài đặt môi trường.....	69
4.6.2. Triển khai	71
4.6.3. Kết quả	74
KẾT LUẬN	80
TÀI LIỆU THAM KHẢO	81

DANH MỤC NHỮNG TỪ VIẾT TẮT

API	Application Programming Interface
AUC	Area Under the Curve
BiLSTM	Bidirectional Long Short-Term Memory
BSON	Binary Javascript Object Notation
CNN	Convolutional Neural Network
CSRF	Cross-site request forgery
ETL	Extract Transform Load
GPT	Generative Pre-training Transformer
JSON	JavaScript Object Notation
LSTM	Long Short-Term Memory
MVC	Model-View-Controller
MVT	Model-View-Template
NER	Named Entity Recognition
NLG	Natural Language Generation
NLP	Natural Language Processing
NLU	Natural Language Understanding
ORM	Object-Relational Mapping
POS	Part-Of-Speech
RDD	Resilient Distributed Dataset
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TF-IDF	Term Frequency – Inverse Document Frequency
URL	Uniform Resource Locator
WSGI	Web Server Gateway Interface
XSS	Cross-site scripting

DANH MỤC BẢNG BIỂU

Bảng 4. 1 Metrics của các mô hình.....	61
Bảng 4. 2 Đặc tả use case Dashboard	65
Bảng 4. 3 Đặc tả use case Classify	66

DANH MỤC HÌNH ẢNH

Hình 1. 1 Xử lý ngôn ngữ tự nhiên	11
Hình 1. 2 Tokenization (Mã hoá từ)	14
Hình 1. 3 Word segmentation (Tách từ liên tục)	14
Hình 1. 4 Sentence breaking (Tách câu)	15
Hình 1. 5 Stop word removal (Loại bỏ từ dừng)	15
Hình 1. 6 Phân biệt Stemming và lemmatization (Chuyển từ về gốc)	16
Hình 1. 7 Morphological segmentation (Phân tích hình thái từ).....	16
Hình 1. 8 Part-of-speech tagging (Gán nhãn từ loại).....	17
Hình 1. 9 Parsing (Phân tích cú pháp)	17
Hình 1. 10 Named Entity Recognition – NER (Nhận diện thực thể có tên).....	18
Hình 1. 11 Coreference resolution (Xác định đồng tham chiếu)	18
Hình 1. 12 Word sense disambiguation (Phân biệt nghĩa từ)	19
Hình 1. 13 Bag-of-words models (Mô hình túi từ).....	19
Hình 1. 14 TF-IDF (Term Frequency – Inverse Document Frequency)	20
Hình 1. 15 Word Embedding.....	21
Hình 1. 16 Sentiment Analysis (Phân tích cảm xúc)	22
Hình 1. 17 Topic Modeling (Mô hình chủ đề)	22
Hình 1. 18 Phân tích cảm xúc	23
Hình 2. 1 Minh họa thuật toán Logistic Regression	29
Hình 2. 2 Minh họa thuật toán SVM.....	31
Hình 2. 3 Minh họa thuật toán CNN	35
Hình 2. 4 Minh họa thuật toán LSTM.....	38
Hình 3. 1 Minh họa Spark Mllib.....	42
Hình 3. 2 Code khởi tạo Spark session và đọc dữ liệu	46
Hình 3. 3 Code tiền xử lý dữ liệu bằng VectorAssembler.....	47
Hình 3. 4 Code khởi tạo mô hình Logistic Regression nhị phân	47
Hình 3. 5 Code khởi tạo mô hình Logistic Regression đa lớp	47
Hình 3. 6 Docker.....	49

Hình 3. 7 Cấu trúc Kafka.....	50
Hình 3. 8 Ứng dụng của Kafka	51
Hình 3. 9 Spark Streaming	52
Hình 3. 10 MongoDB.....	54
Hình 3. 11 Cấu trúc Mongo DB.....	55
Hình 3. 12 Django	56
Hình 3. 13 Các thành phần cơ bản của Django	57
Hình 4. 1 Mẫu dữ liệu huấn luyện và kiểm thử.....	58
Hình 4. 2 Dữ liệu sau khi được làm sạch.	59
Hình 4. 3 Code Pipeline xử lý và huấn luyện.....	59
Hình 4. 4 Code Grid search cho mô hình Logistic Regression.....	60
Hình 4. 5 Confusion matrix của Logistic Regression.....	61
Hình 4. 6 Confusion matrix của Random Forest.....	62
Hình 4. 7 Confusion Matrix của Naive Bayes.....	63
Hình 4. 8 Kiến trúc hệ thống	64
Hình 4. 9 Biểu đồ use case tổng quát.....	65
Hình 4. 10 Biểu đồ trình tự use case Dashboard	67
Hình 4. 11 Biểu đồ phân tích use case Dashboard	68
Hình 4. 12 Biểu đồ trình tự use case Classify	68
Hình 4. 13 Biểu đồ phân tích use case Classify	68
Hình 4. 14 Cấu hình Zookeeper và Fafka	69
Hình 4. 15 Container chứa Zookeeper và Kafka	69
Hình 4. 16 Tạo topic trong Kafka	70
Hình 4. 17 Database và Collection để lưu dữ liệu.....	70
Hình 4. 18 Các thư viện cần thiết	71
Hình 4. 19 Code Producer	71
Hình 4. 20 Code Consumer	72
Hình 4. 21 Code hàm hiển thị thống kê dữ liệu trong view	73
Hình 4. 22 Code hàm phân loại dữ liệu	73

Hình 4. 23 Code hàm phân loại dữ liệu trong view.....	74
Hình 4. 24 Hình ảnh dữ liệu được thu thập bởi consumer.....	75
Hình 4. 25 Hình ảnh dữ liệu được xử lý và phân loại từ topic.....	75
Hình 4. 26 Hình ảnh kết quả trong cơ sở dữ liệu	76
Hình 4. 27 Giao diện Dashboard	77
Hình 4. 28 Giao diện phân loại dữ liệu	78
Hình 4. 29 Giao diện phân loại dữ liệu khi nhập văn bản	78
Hình 4. 30 Giao diện phân loại dữ liệu khi không nhập văn bản	79

LỜI CẢM ƠN

Lời đầu tiên, em muốn gửi lời cảm ơn tới trường Đại học Công nghiệp Hà Nội nói chung và trường Công nghệ thông tin và Truyền thông nói riêng đã tạo môi trường và điều kiện học tập tốt nhất cho các sinh viên như em. Nhờ sự quan tâm của nhà trường trong suốt quá trình 4 năm học tập mà những sinh viên như chúng em mới dần trưởng thành, học hỏi được nhiều kiến thức, bài học bổ ích, là tiền đề cho chúng em phát triển bản thân và tích cực tham gia vào thị trường lao động sau này.

Tiếp theo chắc chắn phải kể đến sự định hướng và nhiệt tình hướng dẫn của thầy Nguyễn Mạnh Cường đối với em cũng như các bạn sinh viên trong khóa luận tốt nghiệp này. Sự quan tâm, tận tình của thầy là nguồn động lực thúc đẩy chúng em tạo ra được những đồ án tốt nhất – thành quả đánh dấu sự trưởng thành và hoàn thiện của bản thân chúng em trước khi có thể tốt nghiệp ra trường.

Em xin trân trọng cảm ơn!

Sinh viên thực hiện

Phạm Đức Duy

LỜI NÓI ĐẦU

Trong những năm gần đây, việc ứng dụng các công nghệ trí tuệ nhân tạo nói chung và các kỹ thuật phân tích cảm xúc nói riêng đang nở rộ tại Việt Nam. Trí tuệ nhân tạo đã được nghiên cứu và ứng dụng mạnh mẽ trong phân tích dữ liệu mạng xã hội như hỗ trợ phân tích xu hướng thị trường, nhận diện khủng hoảng truyền thông, hay theo dõi phản ứng của người dùng với các sự kiện. Tuy nhiên, các hỗ trợ này hầu như nhắm tới các doanh nghiệp lớn hoặc các tổ chức có quy mô lớn, trong khi các doanh nghiệp vừa và nhỏ đang chiếm một số lượng rất lớn tại Việt Nam lại gặp khó khăn trong việc tiếp cận những công nghệ này.

Đề tài "**Phân loại cảm xúc người dùng mạng X theo thời gian thực bằng Apache Spark MLLib**" tập trung vào việc xây dựng một hệ thống phân loại cảm xúc người dùng theo thời gian thực. Thực tế cho thấy việc phân tích cảm xúc người dùng mạng xã hội hiện chủ yếu dựa trên các công cụ thủ công hoặc phải thuê các chuyên gia phân tích dữ liệu, dẫn đến tình trạng nhiều doanh nghiệp gặp phải các phản hồi tiêu cực trên mạng X nhưng chưa biết cách phản ứng kịp thời và gặp nhiều khó khăn trong việc theo dõi thái độ của khách hàng.

Mục tiêu chính của nghiên cứu hướng tới việc phát huy thế mạnh của công nghệ xử lý dữ liệu lớn trong lĩnh vực này, góp phần xây dựng một hệ thống hỗ trợ các doanh nghiệp trong việc theo dõi và phản ứng kịp thời với phản hồi của khách hàng trên mạng xã hội. Đồng thời, việc thực hiện đề tài sẽ góp phần thúc đẩy trào lưu nghiên cứu, sản xuất các sản phẩm phần mềm trí tuệ nhân tạo và mở ra nhiều hướng nghiên cứu, ứng dụng phân tích cảm xúc, đặc biệt trong lĩnh vực xử lý dữ liệu mạng xã hội theo thời gian thực.

Để thực hiện thành công đề tài, nghiên cứu yêu cầu việc thực hiện khảo sát và phân tích bài toán phân loại cảm xúc người dùng mạng X, thu thập và tiền xử lý bộ dữ liệu huấn luyện cho các mô hình phân loại trên Spark MLLib, nghiên cứu chuyên sâu về kiến trúc hệ thống streaming, và cuối cùng là xây dựng chương trình demo.

Nội dung quyển báo cáo đồ án tốt nghiệp sẽ bao gồm các chương như sau:

Chương 1: Phát biểu bài toán

Chương đầu tiên đặt nền tảng cho nghiên cứu bằng cách giới thiệu về xử lý ngôn ngữ tự nhiên và vai trò của nó trong thế giới hiện đại. Từ đó, tôi đi sâu vào lĩnh vực phân loại cảm xúc. Cuối cùng, tôi xác định và phát biểu bài toán cụ thể.

Chương 2: Một số kỹ thuật giải quyết bài toán

Sau khi đã phát biểu và xác định rõ ràng được yêu cầu bài toán, tôi trình bày một số kỹ thuật giải quyết bài toán hiện có cùng các ưu và nhược điểm của chúng, cũng như các nghiên cứu nổi bật đã đạt được thành công nhất định từ những kỹ thuật đó.

Chương 3: Apache Spark Mllib và các công cụ hỗ trợ triển khai

Tại chương 3, tôi tập trung vào công cụ hiện đại Apache Spark MLlib và tiềm năng của nó trong xử lý dữ liệu lớn thời gian thực. Đặc biệt, tôi trình bày chi tiết việc triển khai Logistic Regression với Spark MLlib, cùng các công nghệ hỗ trợ như Docker, Kafka và MongoDB để xây dựng một hệ thống phân tích cảm xúc hoàn chỉnh.

Chương 4: Thực nghiệm và xây dựng sản phẩm demo

Trong chương 4, tôi trình bày quá trình thực nghiệm với mô hình đề xuất, so sánh kết quả với các phương pháp khác và xây dựng ứng dụng demo phân loại cảm xúc theo thời gian thực.

Phản kết luận:

Cuối cùng trong phản kết luận, tôi tổng hợp các kết quả đạt được, các hướng phát triển và mở rộng để tài nghiên cứu trong tương lai, bao gồm cả việc thích ứng mô hình cho các ngôn ngữ khác và các ứng dụng thực tiễn trong nhiều lĩnh vực.

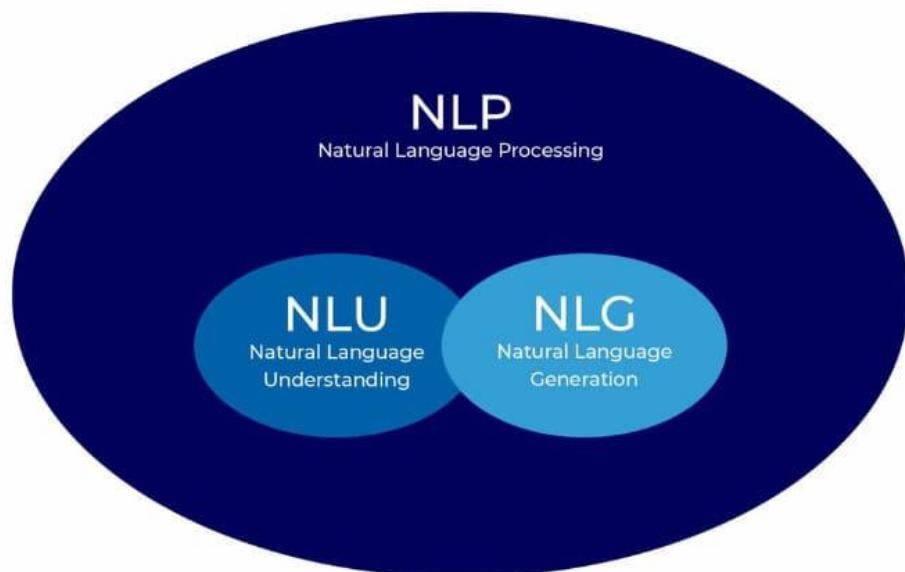
Tổng kết lại, đề tài mong muốn đóng góp giải pháp phân tích cảm xúc mạng X theo thời gian thực cho các doanh nghiệp vừa và nhỏ tại Việt Nam với chi phí hợp lý. Kết quả nghiên cứu vừa có giá trị học thuật về xử lý dữ liệu lớn, vừa mang ý nghĩa thực tiễn giúp doanh nghiệp cải thiện khả năng phản ứng truyền thông và mở ra tiềm năng phát triển công nghệ AI trong nước.

CHƯƠNG 1: PHÁT BIỂU BÀI TOÁN

1.1. Tổng quan về xử lý ngôn ngữ tự nhiên

1.1.1. Định nghĩa

Xử lý ngôn ngữ tự nhiên hay Natural Language Processing (NLP) là một lĩnh vực trong trí tuệ nhân tạo (AI) nhằm biến máy tính thành một công cụ có thể hiểu và diễn giải ngôn ngữ con người dưới dạng văn bản hoặc giọng nói giống như cách con người giao tiếp với nhau. Qua đó, hỗ trợ con người trong việc tự động hóa các nhiệm vụ hay công việc liên quan đến ngôn ngữ hàng ngày, đặc biệt là những công việc đòi hỏi phân tích văn bản với khối lượng lớn và yêu cầu hiệu suất cao. Để thực hiện điều đó, các thuật toán và mô hình học sâu, đặc biệt là những mô hình dựa trên cơ chế Transformer và các kiến trúc ngôn ngữ lớn đã ra đời nhằm trích xuất, xử lý, phân tích nội dung và tạo ra các phép toán phù hợp với các bộ dữ liệu ngôn ngữ mà con người đưa vào.



Hình 1. 1 Xử lý ngôn ngữ tự nhiên

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

Ngoài ra, NLP thường được chia thành hai nhánh chính:

- Hiểu ngôn ngữ tự nhiên (NLU): Tập trung vào việc phân tích và hiểu văn bản, giúp máy tính nắm bắt được ý nghĩa thực sự của ngôn ngữ con người. NLU sử dụng các kỹ thuật để xác định ngữ cảnh, ý định và các yếu tố ngữ nghĩa trong văn bản.
- Sinh ngôn ngữ tự nhiên (NLG): Liên quan đến việc tạo ra văn bản có ý nghĩa từ dữ liệu có cấu trúc. NLG chuyển đổi thông tin máy tính thành ngôn ngữ tự nhiên mà con người có thể dễ dàng đọc và hiểu được.

1.1.2. Thực trạng

Hiện nay, xử lý ngôn ngữ tự nhiên đang được ứng dụng vào rất nhiều lĩnh vực, đặc biệt là lĩnh vực giao tiếp khách hàng, dịch thuật và trợ lý ảo. Những lĩnh vực này đều đang phục vụ tích cực cho việc tự động hóa giao tiếp, xử lý thông tin, giải quyết các vấn đề phức tạp về ngôn ngữ mà con người khó thực hiện được với khối lượng lớn và nâng cao chất lượng sống của con người.

Trên thế giới hiện nay, các nước như Mỹ, Trung Quốc, Nhật Bản, Hàn Quốc hay Singapore là các nước đi đầu về xử lý ngôn ngữ tự nhiên và đang liên tục chạy đua trong việc chiếm ưu thế trong lĩnh vực trí tuệ nhân tạo ngôn ngữ. Tiêu biểu nhất là Mỹ, với các mô hình ngôn ngữ lớn như GPT, PaLM, và Claude đang dẫn đầu cuộc cách mạng trí tuệ nhân tạo. Các ứng dụng như chatbot thông minh, hệ thống dịch thuật tự động, và trợ lý ảo đang ngày càng phổ biến và trở thành một phần không thể thiếu trong cuộc sống hàng ngày của người dùng.

Tại Việt Nam, xử lý ngôn ngữ tự nhiên được ứng dụng rộng rãi nhất trong lĩnh vực dịch vụ khách hàng, giáo dục và thương mại điện tử. Điều này thể hiện rõ nhất qua các dự án ứng dụng của VinBigdata trong xây dựng các công cụ xử lý tiếng Việt, hay các chatbot hỗ trợ khách hàng của các doanh nghiệp thương mại điện tử như Tiki, Shopee. Qua đó có thể thấy xử lý ngôn ngữ tự nhiên đang thực sự bùng nổ tại nước ta trong những năm vừa qua như thế nào.

Tuy nhiên, hiện tại ngành công nghiệp xử lý ngôn ngữ tự nhiên cho tiếng Việt đang có dấu hiệu thách thức khi không có các nguồn dữ liệu phong phú như các ngôn ngữ phổ biến khác. Đa phần là hướng tới tối ưu hóa mô hình sẵn có hoặc

tạo ra các thuật toán phù hợp cho việc tiền xử lý và sử dụng dữ liệu tiếng Việt. Điều này diễn ra là do sự thiếu hụt trầm trọng nguồn dữ liệu chất lượng cao, các vấn đề đặc thù về ngữ pháp và ngữ nghĩa của tiếng Việt, hay chi phí cho việc huấn luyện mô hình riêng là quá lớn. Ngoài ra, số lượng chuyên gia và tài nguyên xử lý còn thiếu hụt cũng là một vấn đề cực kỳ nghiêm trọng.

Từ đó, một hướng đi mới đã được tạo ra đó là việc tích hợp xử lý ngôn ngữ tự nhiên vào các dạng mô hình trí tuệ nhân tạo đa phương thức. Tiêu biểu nhất là các mô hình như GPT-4V, Gemini hay Claude, đây là ứng dụng bắt đầu cho sự bùng nổ cho con sốt mô hình đa phương thức hay AI đa phương thức. Dưới góc nhìn của xử lý ngôn ngữ tự nhiên, ta thấy được tiềm năng to lớn trong việc sử dụng các kỹ thuật xử lý ngôn ngữ kết hợp với xử lý hình ảnh, âm thanh để tạo ra các hệ thống hiểu và giao tiếp toàn diện hơn. Hay việc các mô hình ngôn ngữ lớn hiện tại có thể sử dụng đầu vào là dữ liệu đa dạng để phục vụ phân tích, tạo sinh theo yêu cầu của người dùng. Những tiến ích đều đang giúp tối ưu hóa các công việc của con người hàng ngày, mang đến trải nghiệm thân thiện và hỗ trợ tối đa nhu cầu tìm hiểu và sáng tạo của con người.

1.1.3. Tác vụ

Các tác vụ NLP bao phủ nhiều khía cạnh khác nhau của việc xử lý văn bản và giọng nói, từ phân tích hình thức đến hiểu ngữ nghĩa. Dưới đây là các tác vụ NLP quan trọng nhất:

- **Tokenization (Mã hóa từ):** Đây là bước đầu tiên trong xử lý ngôn ngữ tự nhiên tiếng Việt, giúp chia văn bản thành các đơn vị nhỏ hơn gọi là “tokens”, thường là từ hoặc cụm từ. Ví dụ, câu “Tôi yêu tiếng Việt” sẽ được chia thành các token: [“Tôi”, “yêu”, “tiếng”, “Việt”].



Hình 1. 2 Tokenization (Mã hóa từ)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Word segmentation (Tách từ liên tục):** Word segmentation giúp hệ thống nhận biết đâu là ranh giới giữa các từ. Ví dụ, một người quét một tài liệu viết tay vào máy tính, Word segmentation có thể phân tích trang và nhận ra rằng các từ được chia bởi khoảng trắng. Đây là kỹ thuật đặc biệt quan trọng với các ngôn ngữ không dùng dấu cách rõ ràng như tiếng Trung hoặc tiếng Việt viết liền, giúp. Ví dụ, từ chuỗi “hoctiengviet”, hệ thống cần phân đoạn chính xác thành “học tiếng Việt”.

Such an incomprehensible mess that it feels less like bad cinema
 EDU1 EDU2

than like being stuck in a dark pit having a nightmare about bad cinema .
 EDU3 EDU4

(a) A sentence (negative) from the MRPC test dataset.

A classy item by a legend who may have nothing left to prove
 EDU1 EDU2

but still has the chops and drive to show how it is done .
 EDU3 EDU4

(b) A sentence (very positive) from the SST-5 test dataset.

Hình 1. 3 Word segmentation (Tách từ liên tục)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Sentence breaking (Tách câu):** Tác vụ này đặt ranh giới giữa các câu trong một đoạn văn bản dài. Ví dụ, đoạn “Trời mưa. Tôi mang ô.” sẽ được xác định là hai câu riêng biệt nhờ dấu chấm phân chia các câu.

Sentence Segmentation

Hello world. This blog post is about sentence segmentation. It is not always easy to determine the end of a sentence. One difficulty of segmentation is periods that do not mark the end of a sentence. An ex. is abbreviations.



- Hello world.
- This blog post is about sentence segmentation.
- It is not always easy to determine the end of a sentence.
- One difficulty of segmentation is periods that do not mark the end of a sentence.
- An ex. is abbreviations.

Hình 1. 4 Sentence breaking (Tách câu)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Stop word removal (Loại bỏ từ dừng):** Những từ phổ biến như “là”, “của”, “và” thường không mang nhiều thông tin ý nghĩa nên có thể được loại bỏ để giảm thời gian xử lý và tăng hiệu quả phân tích. Tuy nhiên, trong một số mô hình hiện đại, những từ này vẫn được giữ lại vì chúng có thể ảnh hưởng đến ngữ cảnh tổng thể.

Natural Language Processing with Python! Stop Words

[“**This**”, “**is**”, “**a**”, “**test**”]
✓ X X ✓

Hình 1. 5 Stop word removal (Loại bỏ từ dừng)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Stemming và lemmatization (Chuyển từ về gốc):** Hai kỹ thuật này giúp đưa từ về dạng gốc để dễ dàng phân tích và so sánh. Ví dụ, từ “đang học”,

“học tập”, “đã học” đều có thể được đưa về từ gốc là “học”. Stemming cắt bỏ phần đuôi từ, còn lemmatization phân tích kỹ hơn để giữ nguyên nghĩa chính xác.

Stemming vs Lemmatization



Hình 1. 6 Phân biệt Stemming và lemmatization (Chuyển từ về gốc)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Morphological segmentation (Phân tích hình thái từ):** Tác vụ này chia từ thành các phần nhỏ hơn gọi là hình thái tố (morphemes). Mỗi hình thái tố có vai trò riêng trong việc hình thành ngữ nghĩa và cấu trúc ngữ pháp của từ. Ví dụ, từ “untestably” sẽ được chia thành [[un[[test]able]]ly], trong đó thuật toán nhận ra “un”, “test”, “able” và “ly” là các hình thái tố.

Morphological Segmentation



- Breaks words into **morphemes**

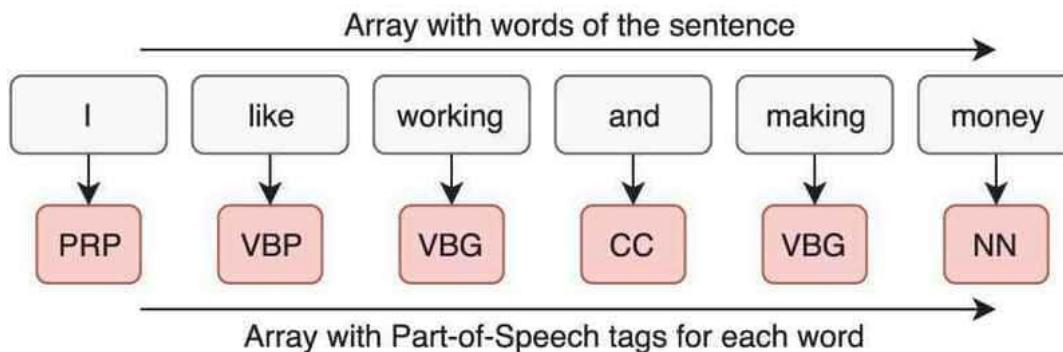
governments \Rightarrow govern – ment – s
lm\$pxtm \Rightarrow l – m\$p – t – m
(according to their families)

- Key component in many NLP applications
- Particularly important for morphologically-rich languages (e.g., Arabic, Hebrew, ...)

Hình 1. 7 Morphological segmentation (Phân tích hình thái từ)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

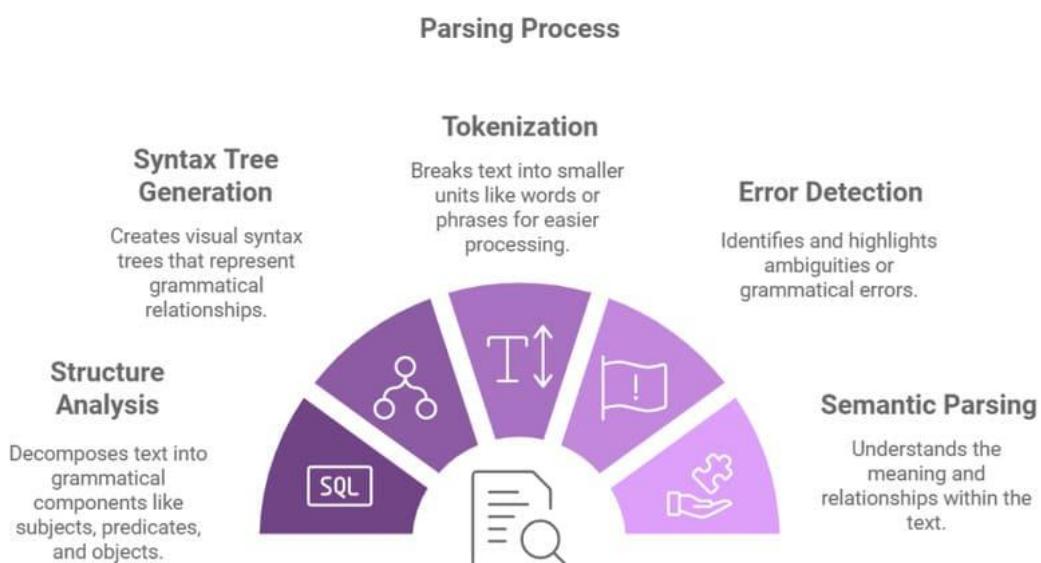
- **Part-of-speech tagging (Gán nhãn từ loại):** Hệ thống xác định vai trò ngữ pháp của từng từ trong câu, như danh từ, động từ, tính từ,... Ví dụ, từ “chạy” là động từ trong câu “Tôi chạy bộ”, nhưng “chạy” có thể là danh từ trong cụm “cuộc chạy đua”. Việc xác định đúng từ loại giúp máy hiểu ngữ cảnh chính xác hơn.



Hình 1. 8 Part-of-speech tagging (Gán nhãn từ loại)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Parsing (Phân tích cú pháp):** Cú pháp là sự sắp xếp của các từ trong một câu để tạo ra ý nghĩa ngữ pháp. Dựa trên cấu trúc ngữ pháp của câu, Parsing xác định chủ ngữ, vị ngữ, tân ngữ,... Ví dụ, trong câu “Con mèo đuổi chuột”, parsing giúp hệ thống NLP hiểu “con mèo” là chủ ngữ và “chuột” là tân ngữ của động từ “đuổi”.



Hình 1. 9 Parsing (Phân tích cú pháp)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

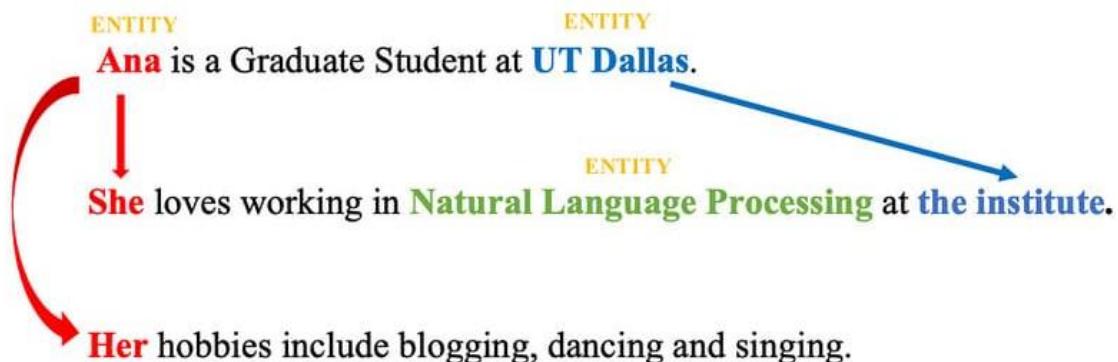
- **Named Entity Recognition – NER (Nhận diện thực thể có tên):** NER giúp xác định và phân loại các thực thể như tên người, địa điểm, tổ chức. Ví dụ, trong câu “Nguyễn Văn A làm việc tại Hà Nội”, hệ thống sẽ nhận biết “Nguyễn Văn A” là tên người, “Hà Nội” là địa danh.



Hình 1. 10 Named Entity Recognition – NER (Nhận diện thực thể có tên)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Coreference resolution (Xác định đồng tham chiếu):** Tác vụ này xác định các từ trong câu hoặc đoạn văn liệu có đề cập đến cùng một thực thể hay không. Ví dụ, trong hai câu “Lan đi học. Cô ấy mang theo sách vở”, hệ thống cần hiểu rằng “Cô ấy” và “Lan” là cùng một người.



Hình 1. 11 Coreference resolution (Xác định đồng tham chiếu)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Word sense disambiguation (Phân biệt nghĩa từ):** Nhiều từ có thể mang nhiều nghĩa, và nhiệm vụ của hệ thống là xác định nghĩa đúng dựa trên ngữ cảnh.

cảnh. Ví dụ, trong câu: “Tôi đang tìm cách làm bánh mì.”, từ “cách” mang nghĩa là phương pháp nhưng trong câu “Nhà tôi cách trường 2km.”, từ “cách” lại mang nghĩa là khoảng cách. Hệ thống NLP cần hiểu ngữ cảnh để không nhầm lẫn hai nghĩa này của từ “cách”.

- ▶ S: (n) **mouse**
 - ▷ S: (n) rodent, gnawer
 - ▶ S: (n) shiner, black eye, **mouse**
 - ▷ S: (n) bruise, contusion
 - ▶ S: (n) **mouse**
 - ▷ S: (n) person, individual, someone, somebody, mortal, soul
 - ▶ S: (n) **mouse**, computer mouse
 - ▷ S: (n) electronic device
- "a **mouse** takes much more room than a trackball"*
-

Hình 1. 12 Word sense disambiguation (Phân biệt nghĩa từ)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Bag-of-words models (Mô hình túi từ):** Đây là phương pháp biểu diễn văn bản như một tập hợp các từ không có thứ tự, chỉ xét đến tần suất xuất hiện. Ví dụ, hai câu “Trời hôm nay rất đẹp” và “Hôm nay trời rất đẹp” sẽ được coi là tương đương trong mô hình này, vì chúng chứa cùng một tập từ với cùng tần suất, dù thứ tự từ khác nhau.

Document D1	<i>The child makes the dog happy</i> the: 2, dog: 1, makes: 1, child: 1, happy: 1					
Document D2	<i>The dog makes the child happy</i> the: 2, child: 1, makes: 1, dog: 1, happy: 1					
	↓					
	child	dog	happy	makes	the	BoW Vector representations
D1	1	1	1	1	2	[1,1,1,1,2]
D2	1	1	1	1	2	[1,1,1,1,2]

Hình 1. 13 Bag-of-words models (Mô hình túi từ)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **TF-IDF (Term Frequency – Inverse Document Frequency):** Đánh trọng số cho từ quan trọng dựa trên mức độ xuất hiện trong tài liệu và trong toàn bộ bộ dữ liệu. Giả sử có 3 tài liệu: “Cà phê là thức uống yêu thích của tôi.”, “Cà phê là thức uống yêu thích của tôi.” và “Cà phê và trà đều rất ngon.”, từ “cà phê” xuất hiện trong tài liệu 1 và tài liệu 3, nhưng không có trong tài liệu 2, nên từ “cà phê” sẽ có trọng số cao hơn trong tài liệu 1 và tài liệu 3 so với tài liệu 2.

$\text{tf}(t, d)$

	blue	bright	can	see	shining	sky	sun	today
1	1/2	0	0	0	0	1/2	0	0
2	0	1/3	0	0	0	0	1/3	1/3
3	0	1/3	0	0	0	1/3	1/3	0
4	0	1/6	1/6	1/6	1/6	0	1/3	0

$\text{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
	0.602	0.125	0.602	0.602	0.602	0.301	0.125	0.602

$\xrightarrow{\quad}$

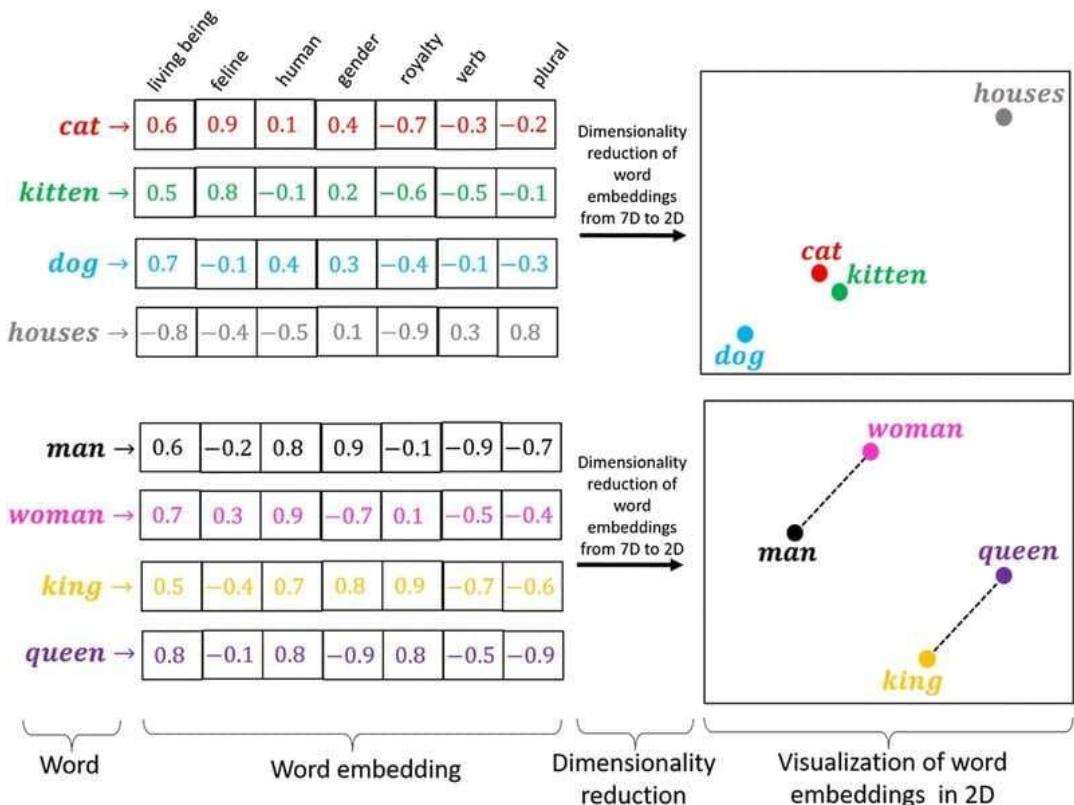
$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
1	0.301	0	0	0	0	0.151	0	0
2	0	0.0417	0	0	0	0	0.0417	0.201
3	0	0.0417	0	0	0	0.100	0.0417	0
4	0	0.0209	0.100	0.100	0.100	0	0.0417	0

Hình 1. 14 TF-IDF (Term Frequency – Inverse Document Frequency)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Word Embeddings:** Biểu diễn từ dưới dạng vector số, giúp máy tính hiểu mối quan hệ ngữ nghĩa giữa các từ. Giả sử ta có các từ: “công việc”, “nghề nghiệp”, “làm việc”, “trường học”. Các từ này có thể được biểu diễn dưới dạng vector số, chẳng hạn như: “công việc” = [0.21, 0.33, 0.45, ...], “nghề nghiệp” = [0.20, 0.34, 0.46, ...]. Máy tính có thể nhận ra rằng “công việc” và “nghề nghiệp” có mối quan hệ ngữ nghĩa gần gũi, do đó chúng sẽ có các vector gần nhau trong không gian vector



Hình 1. 15 Word Embedding

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

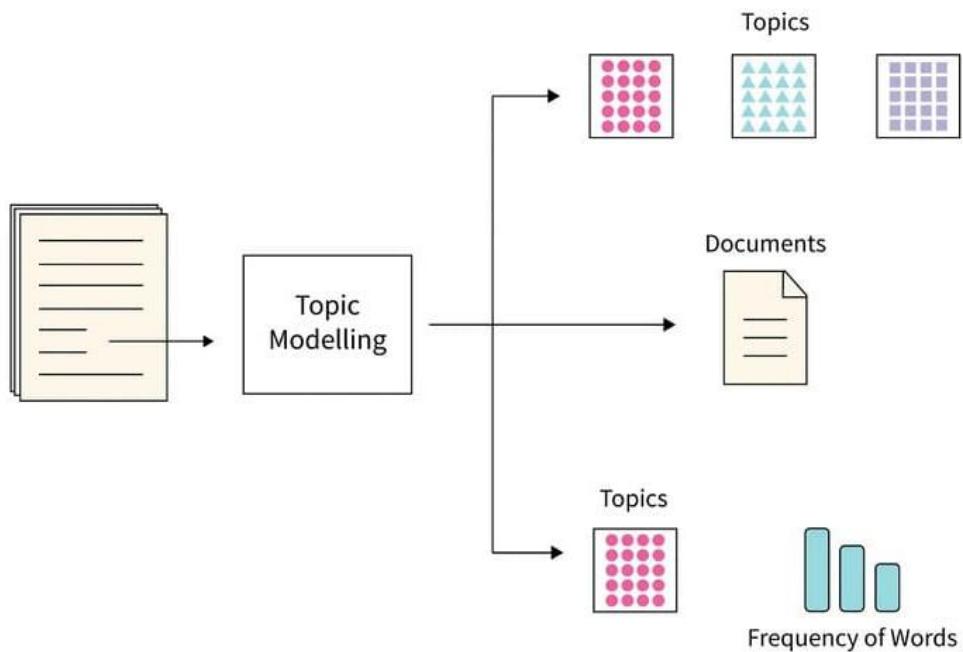
- **Sentiment Analysis (Phân tích cảm xúc):** Xác định cảm xúc hoặc thái độ trong văn bản, giúp đánh giá liệu văn bản mang tính tích cực, tiêu cực hay trung lập. Ví dụ: Câu: “Món ăn này thật tuyệt vời, tôi rất thích!”, phân tích cảm xúc sẽ xác định đây là một câu tích cực, vì từ “tuyệt vời” và “thích” thể hiện sự hài lòng và cảm giác tốt đẹp.



Hình 1. 16 Sentiment Analysis (Phân tích cảm xúc)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

- **Topic Modeling (Mô hình chủ đề):** Nhận diện các chủ đề chính trong một tập hợp văn bản lớn, giúp tổ chức và phân loại tài liệu dựa trên các chủ đề. Ví dụ, các bài viết về công nghệ có thể bao gồm từ “máy tính”, “phần mềm”, “trí tuệ nhân tạo” còn các bài viết về thể thao có thể bao gồm từ “bóng đá”, “giải đấu”, “vận động viên”.

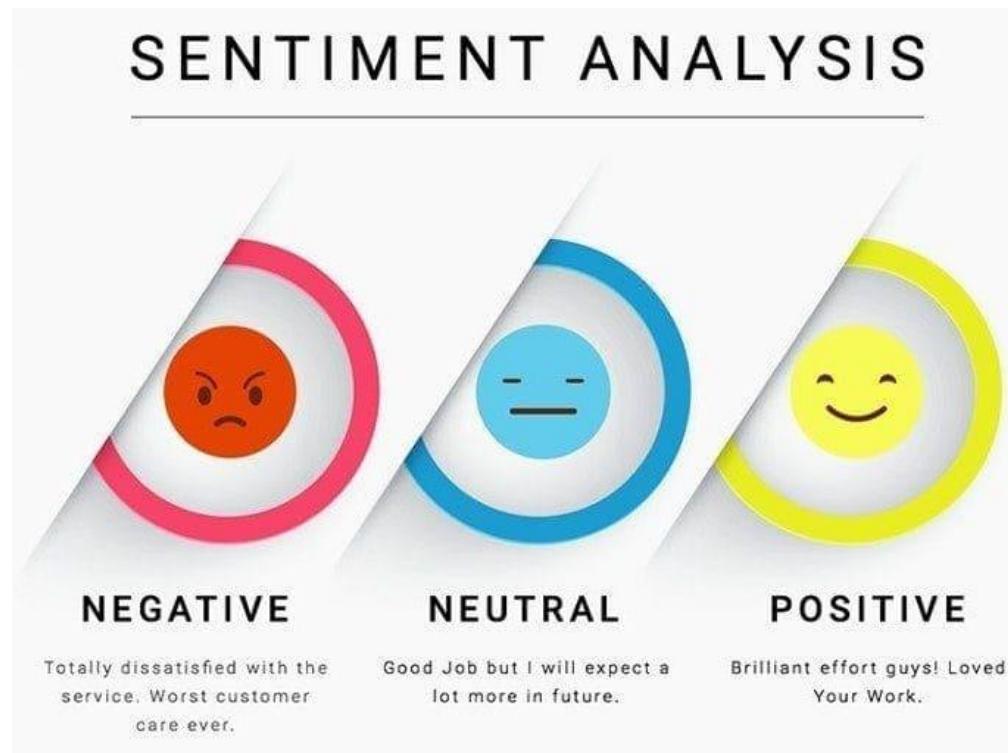


Hình 1. 17 Topic Modeling (Mô hình chủ đề)

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [1])

1.2. Tổng quan về phân tích cảm xúc

Phân tích cảm xúc (Sentiment Analysis) là quy trình kết hợp các công nghệ xử lý ngôn ngữ tự nhiên, học máy và phân tích dữ liệu nhằm nhận diện và phân loại sắc thái tình cảm (tích cực, tiêu cực hoặc trung lập) được thể hiện trong nội dung văn bản. Mục đích chính của phân tích cảm xúc là nắm bắt chính xác tâm trạng, quan điểm và thái độ của người dùng hoặc cộng đồng đối với các sản phẩm, dịch vụ hoặc đề tài cụ thể.



Hình 1. 18 Phân tích cảm xúc

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [2])

Phân tích cảm xúc được chia thành 4 loại chính bao gồm:

- **Phân tích cảm xúc chi tiết:** Phân loại cảm xúc theo nhiều cấp độ (rất tích cực, tích cực, trung tính, tiêu cực, rất tiêu cực) thay vì chỉ phân loại đơn giản; ví dụ như đánh giá sản phẩm 5 sao ("tuyệt vời") so với 1 sao ("hoàn toàn thất vọng").
- **Phân tích cảm xúc theo khía cạnh:** Không chỉ xác định cảm xúc tổng thể mà còn phân tích cảm xúc đối với từng khía cạnh cụ thể của đối tượng; như đánh giá "Thức ăn rất ngon (tích cực) nhưng dịch vụ quá chậm (tiêu cực)".

- **Phát hiện cảm xúc cụ thể:** Xác định các loại cảm xúc chi tiết (vui, buồn, giận dữ, lo lắng, sợ hãi) thay vì chỉ xác định cảm xúc tích cực/tiêu cực; ví dụ nhận diện cảm xúc "phản khích" từ câu "Tôi thực sự phản khích khi nhận được tin này!"
- **Phân tích ý định:** Phát hiện mục đích hoặc ý định của người dùng thông qua nội dung văn bản; thường được ứng dụng trong chatbot và trợ lý ảo để xác định hành động người dùng muốn thực hiện.

1.3. Phát biểu bài toán

1.3.1. Định nghĩa bài toán

Bài toán phân loại cảm xúc người dùng mạng X theo thời gian thực thuộc dạng bài toán phân tích cảm xúc (sentiment analysis) trong lĩnh vực xử lý ngôn ngữ tự nhiên. Về bản chất, đây là một bài toán phân loại văn bản (text classification) với đặc thù xử lý dữ liệu streaming. Đầu vào của bài toán là dữ liệu văn bản từ các bài đăng (tweets) đã được tiền xử lý, và đầu ra là nhãn cảm xúc (tích cực, tiêu cực, trung tính, không liên quan) tương ứng.

1.3.2. Phân tích yêu cầu

Cho một tập dữ liệu đầu vào $D = \{t_1, t_2, \dots, t_n\}$ gồm n bài đăng từ mạng X, mỗi bài đăng t_i là một chuỗi văn bản tiếng Anh. Mục tiêu là xây dựng một hàm phân loại $f: t_i \rightarrow c_j$, trong đó $c_j \in C = \{\text{tích cực}, \text{tiêu cực}, \text{trung lập}, \text{không liên quan}\}$ là tập các nhãn cảm xúc.

Đầu vào của bài toán:

- Dữ liệu văn bản từ mạng X, được thu thập thông qua file csv.
- Mỗi bản ghi đầu vào có dạng tweet nguyên bản, ví dụ:
 - "Rocket League, Sea of Thieves or Rainbow Six Siege? I love playing all three on stream but which is the best? #stream #twitch #RocketLeague #SeaOfThieves #RainbowSixSiege #follow"
 - "my ass still knee-deep in Assassins Creed Odyssey with no way out anytime soon lmao"

Đầu ra của bài toán:

- Mỗi tweet sẽ được phân loại vào một trong bốn nhãn:
 - Tích cực (Positive): biểu thị quan điểm tích cực
 - Tiêu cực (Negative): biểu thị quan điểm tiêu cực
 - Trung lập (Neutral): không rõ ràng tích cực hay tiêu cực
 - Không liên quan (Irrelevant): nội dung không phù hợp để phân tích cảm xúc
- Kết quả đầu ra sẽ được lưu trữ trong cơ sở dữ liệu MongoDB và hiển thị trên web với các thành phần:
 - Bảng dữ liệu chi tiết từng tweet và nhãn tương ứng
 - Biểu đồ thống kê phân bố cảm xúc
 - Biểu đồ cột thể hiện tần suất từ xuất hiện theo từng loại cảm xúc

1.3.3. Phạm vi và giới hạn của bài toán

Bài toán phân loại cảm xúc người dùng mạng X nếu xét một cách tổng quát sẽ có phạm vi rất lớn do sự đa dạng của ngôn ngữ, văn hóa và chủ đề trên mạng X. Không chỉ vậy, cách biểu đạt cảm xúc luôn thay đổi với sự phát triển của mạng xã hội, bao gồm việc sử dụng emoji, hashtag, từ viết tắt và biểu hiện mỉa mai, châm biếm gây khó khăn trong việc phân loại chính xác.

Trên thế giới, bài toán phân loại cảm xúc đã được giải quyết dưới nhiều hình thức khác nhau như phân tích theo chủ đề, phân tích đa ngôn ngữ, phân tích theo ngữ cảnh... Tuy nhiên, để giải quyết bài toán một cách hiệu quả nhất, phương pháp được lựa chọn cần phù hợp với nguồn lực và yêu cầu cụ thể. Do đó, tôi sẽ thực hiện bài toán trong phạm vi sau:

- Chỉ xử lý dữ liệu văn bản tiếng Anh từ mạng X, không bao gồm hình ảnh hoặc video đính kèm.
- Sử dụng mô hình Logistic Regression trên Spark MLlib vì hiệu quả cao trong phân loại văn bản và khả năng xử lý theo thời gian thực.
- Phân loại cảm xúc thành ba nhóm chính: tích cực, tiêu cực, trung lập và không liên quan.

- Tiến hành thực nghiệm trên bộ dữ liệu khoảng 74,000 bài đăng đã được gán nhãn để huấn luyện mô hình.
- Xây dựng kiến trúc hệ thống dựa trên nền tảng Docker, Kafka và Spark Streaming để đảm bảo khả năng xử lý theo thời gian thực.

1.3.4. Khó khăn và thách thức

Đối với con người, việc nhận biết cảm xúc trong văn bản không phải là việc khó khăn, nhưng đối với một hệ thống nhân tạo thì việc phân tích cảm xúc đòi hỏi phải giải quyết rất nhiều vấn đề như:

- **Sự xuất hiện hoặc thiếu một số thành phần:** Các yếu tố biểu đạt cảm xúc có thể xuất hiện hoặc không xuất hiện trong văn bản làm cho bài toán phân tích cảm xúc trở nên khó khăn hơn nhiều.
- **Tư thế, góc độ biểu đạt:** Cách diễn đạt cảm xúc có thể thay đổi rất nhiều tùy thuộc vào góc nhìn của người viết. Chẳng hạn như diễn đạt trực tiếp, gián tiếp, ngũ ý...
- **Sự biến dạng của đối tượng:** Biến dạng của cảm xúc trong văn bản cũng có thể làm ảnh hưởng đến việc phân tích chính xác cảm xúc đó.
- **Sự che khuất:** Cảm xúc thật có thể bị che khuất bởi các biểu đạt mỉa mai, châm biếm hoặc phức tạp.
- **Sự phức tạp của ngũ cảnh:** Ngũ cảnh phức tạp cũng sẽ khiến cho việc phân tích cảm xúc trở nên khó khăn.
- **Điều kiện của văn bản:** Văn bản được viết trong các điều kiện khác nhau về ngôn ngữ, văn hóa, lĩnh vực... ảnh hưởng rất nhiều đến chất lượng của việc phân tích cảm xúc.

1.3.5. Các ứng dụng của bài toán

Hệ thống phân loại cảm xúc người dùng mạng X theo thời gian thực có nhiều ứng dụng tiềm năng trong các lĩnh vực:

- **Quản lý khủng hoảng truyền thông:** Phát hiện sớm các phản hồi tiêu cực về thương hiệu hoặc sản phẩm, cho phép doanh nghiệp phản ứng nhanh

chóng trước khi vấn đề lan rộng. Đặc biệt hữu ích trong các tình huống khủng hoảng truyền thông khi mỗi giây đều quan trọng.

- **Đo lường hiệu quả chiến dịch marketing:** Theo dõi phản ứng của người dùng đối với các chiến dịch quảng cáo, ra mắt sản phẩm mới hoặc sự kiện marketing. Doanh nghiệp có thể đánh giá tác động theo thời gian thực và điều chỉnh chiến lược nếu cần.
- **Phân tích phản hồi sản phẩm/dịch vụ:** Thu thập và phân tích ý kiến người dùng về sản phẩm hoặc dịch vụ, giúp doanh nghiệp hiểu rõ điểm mạnh và điểm yếu để cải tiến liên tục.
- **Nghiên cứu thị trường và cạnh tranh:** Theo dõi xu hướng thị trường, phân tích cảm xúc người dùng đối với sản phẩm của đối thủ cạnh tranh, và xác định cơ hội mới trên thị trường.
- **Dự báo xu hướng:** Kết hợp phân tích cảm xúc với kỹ thuật dự báo để nhận diện các xu hướng mới nổi trong ngành hoặc thị trường.
- **Cải thiện dịch vụ khách hàng:** Xác định và phản hồi nhanh chóng các khiếu nại hoặc câu hỏi của khách hàng trên mạng X, cải thiện trải nghiệm và sự hài lòng của khách hàng.
- **Tối ưu hóa nội dung truyền thông:** Phân tích loại nội dung nào tạo ra phản ứng tích cực nhất từ người dùng, giúp doanh nghiệp tối ưu hóa chiến lược nội dung.
- **Theo dõi sự kiện:** Theo dõi phản ứng của công chúng đối với các sự kiện lớn như hội nghị, triển lãm, hay sự kiện thể thao, giúp ban tổ chức điều chỉnh và cải thiện trải nghiệm người tham dự.
- **Phân tích chính trị và xã hội:** Đánh giá phản ứng công chúng đối với các chính sách, sự kiện chính trị hoặc xã hội, cung cấp thông tin giá trị cho các nhà hoạch định chính sách và tổ chức xã hội.

CHƯƠNG 2: MỘT SỐ KỸ THUẬT GIẢI QUYẾT BÀI TOÁN

2.1. Phương hướng tiếp cận bài toán

Nghiên cứu này tiếp cận bài toán phân tích cảm xúc thông qua một quy trình có cấu trúc, bắt đầu từ việc thu thập và tiền xử lý dữ liệu từ nền tảng mạng xã hội X, sau đó ứng dụng các kỹ thuật học máy để phân loại cảm xúc từ nội dung các bài đăng.

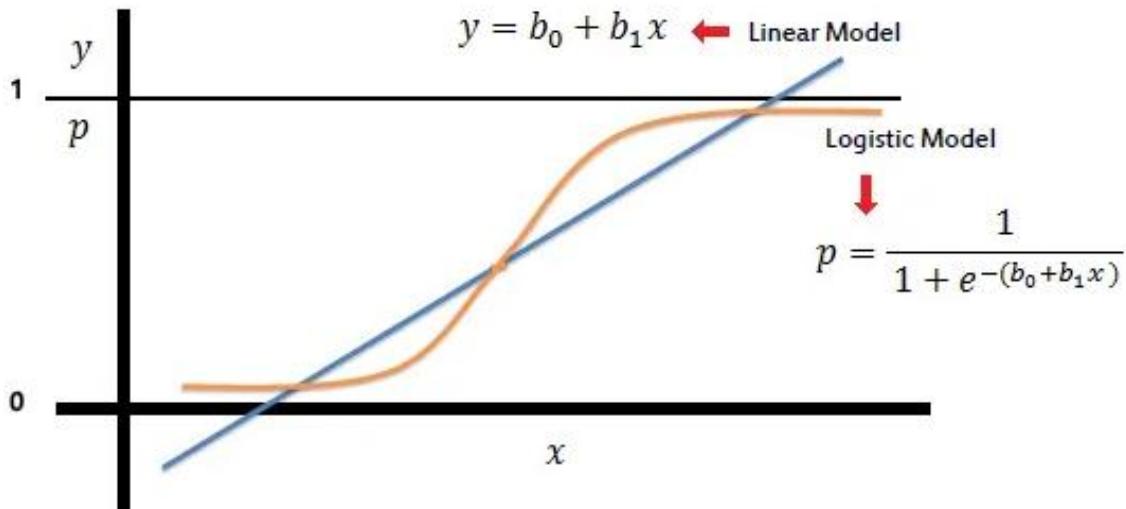
Trong phân tích cảm xúc, văn bản thường được phân loại thành các nhóm cảm xúc như tích cực (positive), tiêu cực (negative) và trung tính (neutral). Để giải quyết bài toán này, nghiên cứu xem xét hai hướng tiếp cận chính:

- **Hướng tiếp cận học máy truyền thống:** Sử dụng các thuật toán học máy cổ điển như Logistic Regression và Support Vector Machine (SVM) để xây dựng mô hình phân loại cảm xúc dựa trên các đặc trưng được trích xuất từ văn bản.
- **Hướng tiếp cận học sâu:** Áp dụng các mô hình mạng nơ-ron sâu như CNN (Convolutional Neural Network) và Long Short-Term Memory (LSTM) để tự động học các biểu diễn phức tạp và trừu tượng từ dữ liệu văn bản.

2.2. Hướng tiếp cận học máy

2.2.1. Logistic regression

Logistic regression hay còn gọi là hồi quy logistic là một phương pháp thống kê mạnh mẽ dùng cho các bài toán phân loại nhị phân, nơi biến phụ thuộc chỉ nhận một trong hai giá trị (thường là 0 hoặc 1). Khác với hồi quy tuyến tính thông thường, hồi quy Logistic áp dụng chuyển đổi phi tuyến để dự đoán xác suất một quan sát thuộc về một nhóm cụ thể.



Hình 2. 1 Minh họa thuật toán Logistic Regression

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [3])

Hồi quy Logistic hoạt động dựa trên hàm Sigmoid, được biểu diễn như sau:

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

Hàm Sigmoid nhận đầu vào là một giá trị z bất kỳ, và trả về đầu ra là một giá trị xác suất nằm trong khoảng [0, 1]. Khi áp dụng vào mô hình Hồi quy Logistic với đầu vào là ma trận dữ liệu X và trọng số w, ta có z = Xw.

Việc huấn luyện của mô hình là tìm ra bộ trọng số w sao cho đầu ra dự đoán của hàm Sigmoid gần với kết quả thực tế nhất. Để làm được điều này, ta sử dụng hàm mất mát (Loss Function) để đánh giá hiệu năng của mô hình. Mô hình càng tốt khi hàm mất mát càng nhỏ.

Hàm mất mát (Loss Function) là một hàm số được sử dụng để đo lường mức độ lỗi mà mô hình của chúng ta tạo ra khi dự đoán các kết quả từ dữ liệu đầu vào. Trong bài toán Hồi quy Logistic, chúng ta sử dụng hàm mất mát Cross-Entropy (còn gọi là Log Loss) để đánh giá hiệu năng của mô hình.

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

$$L(w) = -\frac{1}{n} \sum_{i=1}^n [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

Trong đó:

- π : số lượng mẫu dữ liệu trong tập huấn luyện.
- y_i : giá trị thực tế của đầu ra thứ i.
- p_i : xác suất dự đoán thuộc lớp 1 của mô hình cho đầu vào thứ i.

Hàm Cross-Entropy đo lường khoảng cách giữa hai phân phối xác suất y_i và p_i . Khi mô hình dự đoán chính xác, tức là nếu $y_i = 1$ thì p_i càng gần 1, và nếu $y_i = 0$ thì p_i càng gần 0, sau đó hàm mất mát sẽ tiến gần về 0.

Trong quá trình huấn luyện, chúng ta tìm cách cập nhật bộ trọng số w sao cho giá trị hàm mất mát Cross-Entropy đạt giá trị nhỏ nhất, dẫn đến một mô hình dự đoán tốt nhất.

Để tìm giá trị tối ưu cho bộ trọng số w, chúng ta có thể sử dụng kỹ thuật Gradient Descent. Tại mỗi bước lập, chúng ta cập nhật w theo phương từng ứng với đạo hàm của hàm mất mát L(w) theo w.

Các loại mô hình hồi quy logistic chính:

- **Hồi quy logistic nhị phân:** Trong phương pháp này, biến phản hồi hoặc biến phụ thuộc có tính chất lưỡng phân—nghĩa là nó chỉ có hai kết quả có thể (ví dụ: 0 hoặc 1). Phổ biến trong các bài toán như phân loại email rác/không rác hoặc xác định khói u lành/ác tính. Đây là loại phổ biến nhất trong hồi quy logistic.
- **Hồi quy logistic đa thức:** Trong loại mô hình hồi quy logistic này, biến phụ thuộc có ba hoặc nhiều kết quả có thể; tuy nhiên, các giá trị này không có thứ tự cụ thể. Ví dụ: dự đoán thể loại phim người xem sẽ chọn dựa trên tuổi, giới tính và tình trạng hẹn hò.
- **Hồi quy logistic thứ tự:** Loại mô hình hồi quy logistic này được áp dụng khi biến phản hồi có ba hoặc nhiều kết quả có thể, nhưng trong trường hợp này, các giá trị này có thứ tự xác định. Ví dụ về các phản hồi có thứ tự bao gồm thang điểm từ A đến F hoặc thang đánh giá từ 1 đến 5.

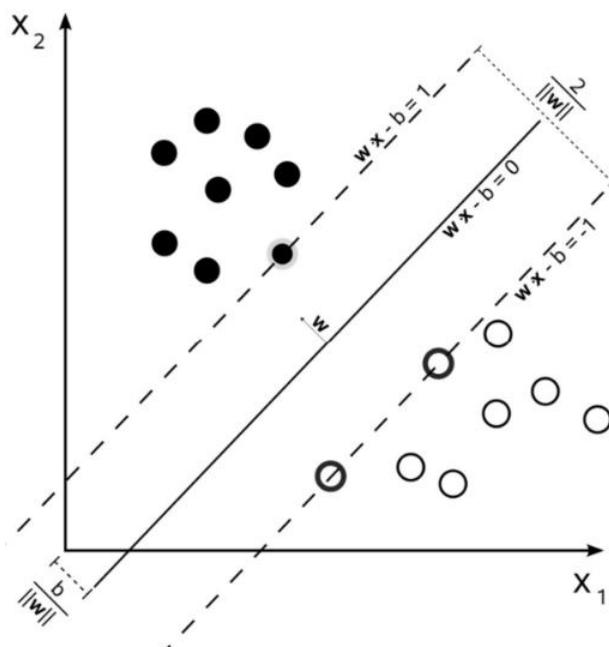
Ưu và nhược điểm trong bài toán phân loại cảm xúc:

- **Ưu điểm:**
 - Mô hình đơn giản, dễ triển khai với cơ sở toán học rõ ràng và dễ hiểu.

- Huấn luyện nhanh, hiệu quả với tập dữ liệu văn bản lớn.
- Cho phép xác định mức độ đóng góp của từng đặc trưng văn bản vào quá trình phân loại cảm xúc.
- Dễ dàng cập nhật và bảo trì, không đòi hỏi tài nguyên tính toán cao.
- **Nhược điểm:**
 - Khả năng mô hình hóa mối quan hệ phi tuyến giữa các từ/cụm từ trong văn bản bị hạn chế.
 - Không hiệu quả trong việc nắm bắt ngữ cảnh và trình tự của từ trong câu văn.
 - Phụ thuộc nhiều vào chất lượng kỹ thuật trích xuất đặc trưng thủ công.
 - Dễ bị overfitting với các đặc trưng hiếm gặp trong văn bản.

2.2.2. SVM (Support Vector Machine)

SVM là một thuật toán học có giám sát (supervised learning), nó có thể sử dụng cho cả bài toán phân lớp hoặc hồi quy. Mục tiêu của SVM là tìm ra một siêu phẳng trong không gian N chiều (ứng với N đặc trưng) chia dữ liệu thành hai phần tương ứng với lớp của chúng. Nói theo ngôn ngữ của đại số tuyến tính, siêu phẳng này phải có lề cực đại và phân chia hai bao lồi và cách đều chúng.



Hình 2. 2 Minh họa thuật toán SVM

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [4])

Sau đây, tôi xin giới thiệu sơ lược về mô hình SVM [13, 14]. Xét bài toán phân lớp nhị phân. Cho trước một tập dữ liệu huấn luyện gồm n mẫu:

$$X = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\} \in R^{n \times d+1}$$

trong đó, x_i là một véc tơ trong không gian R^d và $y_i \in \{-1, 1\}$ là tập các nhãn lớp. Một siêu phẳng phân tách tập X thành hai miền có dạng:

$$\langle w, x \rangle + b = 0 \text{ với } \begin{cases} w \in R^d \\ b \in R \end{cases}$$

Mục tiêu của bài toán huấn luyện SVM là tìm ra một siêu phẳng phân tách "tốt nhất" tập X theo nghĩa là lề của siêu phẳng (margin) đạt cực đại. Để tìm được bộ (w, b) như vậy, ta giải bài toán tối ưu sau:

$$\min_{w, b, \xi} \left\{ \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n \xi_i \right\}$$

sao cho thỏa mãn:

$$\Omega: \begin{cases} (w, b, \xi) \in R^d \times R \times R_+^n \\ y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i, \forall 1 \leq i \leq n \end{cases}$$

trong đó, $\langle \cdot, \cdot \rangle$ là một tích vô hướng được định nghĩa trong không gian R^n , ξ_i là các biến slack được thêm vào để nói lỏng điều kiện phân lớp và C là tham số điều chỉnh. Thay vì giải bài toán trên, ta thường xem xét bài toán đối ngẫu của nó như sau:

$$\min_{\alpha} \frac{1}{2} \alpha^T H \alpha - \hat{1} \alpha$$

trong đó thỏa mãn $\Delta: \begin{cases} y^T \alpha = 0 \\ 0 \leq \alpha_i \leq C, i = 1, \dots, n \end{cases}$. Trong đó $y = (y_1, y_2, \dots, y_n)$, $\hat{1}$ là véc tơ với toàn bộ các thành phần đều bằng 1 và H là một ma trận đối称 được xác định bởi:

$$H_{i,j} = y_i y_j \langle \phi(x_i), \phi(x_j) \rangle = y_i y_j K(x_i, x_j)$$

Ở đây, $\phi(\cdot)$ là một ánh xạ từ không gian ban đầu (input space) sang không gian đặc trưng (feature space) có số chiều cao hơn nhằm xử lý trường hợp dữ liệu

không phân tách tuyến tính. Hàm K(.) được gọi là hàm nhân (kernel function) được định nghĩa:

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

Ưu và nhược điểm trong bài toán phân loại cảm xúc:

- **Ưu điểm:**
 - SVM dựa trên cơ sở toán học chặt chẽ, hoạt động tốt khi dữ liệu văn bản được tiền xử lý tốt (như loại bỏ stopwords, chuẩn hóa, vector hóa từ).
 - SVM có khả năng xử lý dữ liệu phi tuyến trong không gian văn bản nhờ các hàm nhân kernel, giúp phân loại cảm xúc phức tạp.
 - Hiệu quả với không gian đặc trưng có số chiều cao (phổ biến trong xử lý ngôn ngữ tự nhiên).
 - Có khả năng tổng quát hóa tốt, giảm thiểu overfitting với dữ liệu văn bản mới
- **Nhược điểm:**
 - SVM không phù hợp khi huấn luyện với những tập dữ liệu văn bản quá lớn, tốc độ huấn luyện chậm với corpus lớn.
 - Kém hiệu quả trong những trường hợp dữ liệu văn bản chất lượng kém, nhiễu hoặc không được tiền xử lý đầy đủ.
 - Khó khăn trong việc xử lý trực tiếp văn bản đa ngôn ngữ hoặc có nhiều biến thể ngôn ngữ.
 - Yêu cầu lựa chọn hàm kernel phù hợp, việc điều chỉnh tham số có thể phức tạp cho người mới.

2.3. Hướng tiếp cận học sâu

2.3.1. CNN (Convolutional Neural Network)

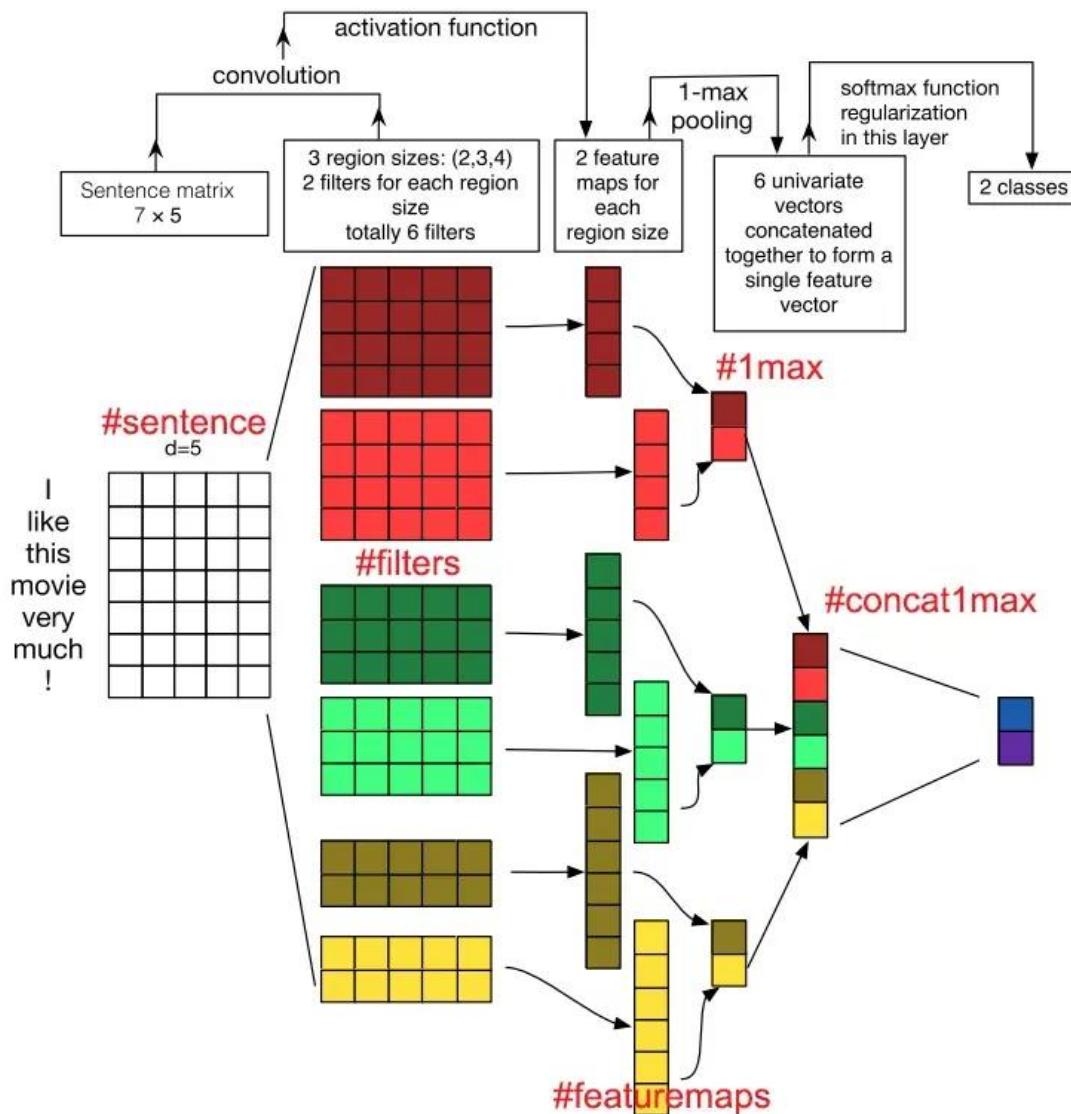
CNN là một dạng mạng nơ-ron nhân tạo được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc lưới như hình ảnh, nhưng cũng rất hiệu quả trong việc xử lý dữ liệu văn bản. Mạng CNN có khả năng tự động học các đặc trưng cục bộ và phân cấp từ dữ liệu, giúp bắt được các mẫu ngữ nghĩa và ngữ cảnh trong văn bản.

Trong lĩnh vực phân tích cảm xúc, CNN được ứng dụng hiệu quả để nhận diện các mẫu cảm xúc trong văn bản thông qua việc phát hiện các đặc trưng ngôn ngữ quan trọng như cụm từ, n-gram và cấu trúc câu. Điểm mạnh của CNN là khả năng nhận diện các mẫu cục bộ trong dữ liệu văn bản mà không cần tiền xử lý phức tạp, cũng như khả năng học đặc trưng tự động từ dữ liệu huấn luyện.

Các thành phần của CNN bao gồm:

- **Tầng đầu vào (Input layer):** Thường là các vector biểu diễn từ (word embedding) của văn bản đầu vào. Mỗi từ trong câu được biểu diễn bằng một vector có số chiều cố định, và toàn bộ văn bản được biểu diễn dưới dạng ma trận 2 chiều (số từ × số chiều embedding).
- **Tầng tích chập (Convolutional layer):** Áp dụng các filter (kernel) trượt trên ma trận đầu vào để phát hiện các mẫu cục bộ. Mỗi filter có kích thước cố định (ví dụ: $3 \times d$, $4 \times d$, $5 \times d$ với d là chiều của word embedding) và trượt dọc theo chiều của văn bản. Các filter này học cách nhận diện các n-gram và cụm từ biểu thị cảm xúc.
- **Hàm kích hoạt (Activation function):** Thường sử dụng hàm ReLU sau mỗi phép tích chập để đưa tính phi tuyến vào mô hình. ReLU giúp mô hình học được các mẫu phức tạp và ngăn chặn vấn đề gradient biến mất.
- **Tầng gộp (Pooling layer):** Thường sử dụng max-pooling để trích xuất các đặc trưng quan trọng nhất từ kết quả của tầng tích chập. Tầng gộp giúp giảm kích thước dữ liệu, giữ lại thông tin quan trọng nhất và tăng tính bất biến với vị trí của các đặc trưng trong văn bản.
- **Tầng dropout:** Đây là cơ chế chống overfitting bằng cách ngẫu nhiên tắt một số nơ-ron trong quá trình huấn luyện, buộc mô hình phải học các đặc trưng mạnh mẽ và đa dạng hơn.
- **Tầng fully-connected (Dense layer):** Kết nối mọi nơ-ron từ tầng trước với mọi nơ-ron ở tầng này. Tầng này thực hiện phép phân loại dựa trên các đặc trưng đã được trích xuất.

- **Tầng đầu ra (Output layer):** Sử dụng hàm kích hoạt softmax (cho bài toán đa lớp) hoặc sigmoid (cho bài toán nhị phân) để xác định xác suất thuộc về mỗi lớp cảm xúc (như tích cực, tiêu cực, trung tính).



Hình 2.3 Minh họat thuật toán CNN

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [5])

CNN trong phân tích cảm xúc hoạt động theo các bước sau:

- Đầu tiên, văn bản được chuyển đổi thành ma trận câu (sentence matrix) qua biểu diễn vector từ (word embedding).
- Sử dụng nhiều filter với kích thước khác nhau (region sizes) để phát hiện các mẫu n-gram.

- Phép tính tích chập được thực hiện giữa filter và vùng tương ứng trên ma trận đầu vào theo công thức: $c_i = f(\sum_{j=1}^m w_j \cdot x_{i+m-1,j} + b)$ trong đó:
 - x_{i+m-1} là vùng từ thứ i đến $i+m-1$ trên ma trận đầu vào
 - w_j là trọng số của filter
 - b là bias
 - f là hàm kích hoạt (thường là ReLU)
 - c_i là giá trị đặc trưng được tạo ra
- Phép max-pooling được áp dụng để chọn giá trị lớn nhất từ mỗi feature map: $\hat{c} = \max c_1, c_2, \dots, c_{n-h+1}$ với n là độ dài văn bản và h là chiều cao của filter.
- Các đặc trưng sau khi qua max-pooling được nối lại với nhau thành một vector đặc trưng.
- Vector đặc trưng này được đưa qua tầng fully-connected và cuối cùng là tầng đầu ra để dự đoán nhãn cảm xúc.

Trong quá trình huấn luyện, CNN sử dụng thuật toán lan truyền ngược (backpropagation) kết hợp với các phương pháp tối ưu hóa như Adam hoặc SGD để điều chỉnh các trọng số nhằm giảm thiểu hàm mất mát (thường là cross-entropy).

Ưu và nhược điểm trong bài toán phân loại cảm xúc:

- **Ưu điểm:**
 - Khả năng học tự động các đặc trưng cục bộ từ văn bản mà không cần feature engineering phức tạp.
 - Hiệu quả trong việc phát hiện các mẫu n-gram quan trọng cho cảm xúc, bất kể vị trí của chúng trong văn bản.
 - Tốc độ xử lý nhanh hơn so với RNN/LSTM, có thể song song hóa các phép tính.
 - Khả năng bắt được các mẫu ngữ nghĩa phức tạp trong các cụm từ ngắn.
- **Nhược điểm:**

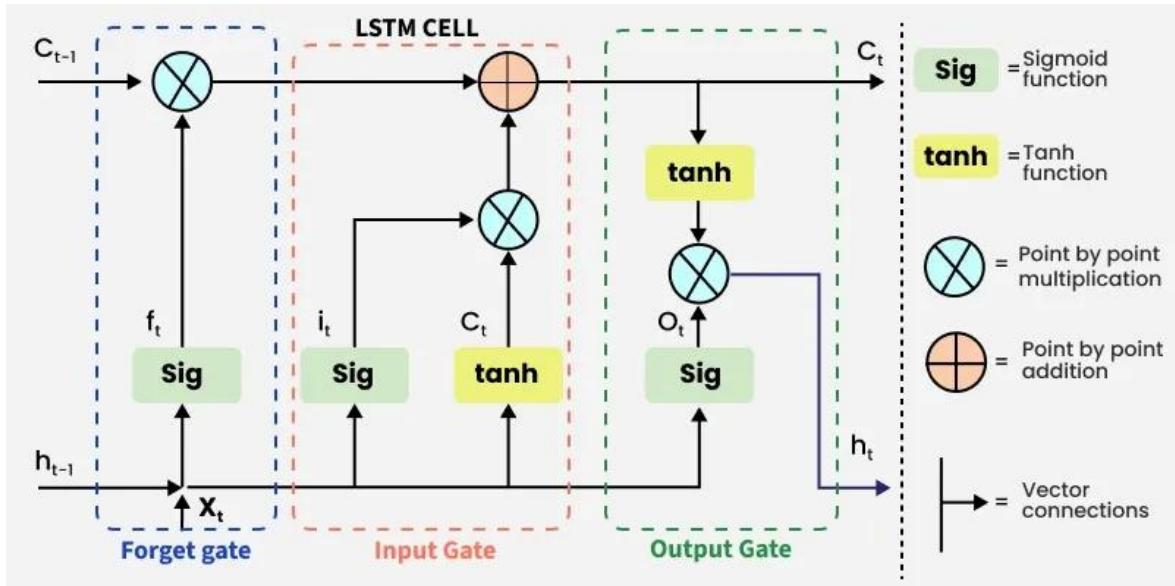
- Khó bắt được mối quan hệ phụ thuộc dài (long-range dependencies) giữa các từ xa nhau trong văn bản.
- Không tận dụng được thông tin về thứ tự từ một cách tự nhiên như RNN/LSTM.
- Cần lượng dữ liệu huấn luyện lớn để đạt hiệu quả tốt.
- Khó khăn trong việc tinh chỉnh siêu tham số như số lượng filter, kích thước filter.

2.3.2. LSTM (Long Short-Term Memory)

LSTM hay Long Short-Term Memory, là một loại mạng nơron hồi tiếp (Recurrent Neural Network - RNN) được thiết kế để xử lý và nhớ thông tin trong các chuỗi dữ liệu dài. Các mô hình RNN truyền thống gặp khó khăn trong việc xử lý các thông tin lâu dài do hiện tượng vanishing gradient (gradient biến mất). Tuy nhiên, nhờ vào cấu trúc đặc biệt của mình, LSTM có khả năng duy trì thông tin trong một khoảng thời gian dài và giải quyết vấn đề này.

Cấu trúc chính của một LSTM bao gồm các thành phần sau:

- **Cổng đầu vào (Input Gate):** Quyết định những thông tin nào từ đầu vào hiện tại sẽ được lưu trữ trong trạng thái tế bào (cell state).
- **Cổng quên (Forget Gate):** Quyết định những thông tin nào sẽ được xóa khỏi trạng thái tế bào.
- **Cổng đầu ra (Output Gate):** Quyết định những thông tin nào từ trạng thái tế bào sẽ được phát ra như là đầu ra của LSTM.



Hình 2.4 Minh họa thuật toán LSTM

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [6])

LSTM bắt đầu tại cổng quên (Forget Gate) với đầu vào x_t (hiện tại) và h_{t-1} (đầu ra trước đó) được kết hợp với ma trận trọng số và độ lệch, sau đó đưa qua hàm kích hoạt. Kết quả trả về sẽ là giá trị nhị phân: 0 (loại bỏ thông tin) hoặc 1 (giữ lại thông tin). Cổng quên hoạt động theo công thức:

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

Trong đó:

- W_f biến diễn ma trận trọng số liên quan đến cổng quên.
- $[h_{t-1}, x_t]$ biểu thị sự nối kết giữa đầu vào hiện tại và trạng thái ẩn trước đó.
- b_f độ lệch với cổng quên.
- σ là hàm kích hoạt sigmoid.

Tiếp theo, LSTM xử lý thông tin mới thông qua cổng đầu vào (Input Gate). Cổng này quyết định thông tin nào sẽ được cập nhật vào trạng thái tế bào. Quá trình này bắt đầu bằng việc sử dụng hàm sigmoid để điều chỉnh thông tin từ đầu vào x_t và trạng thái ẩn trước đó h_{t-1} , xác định giá trị nào sẽ được cập nhật:

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

Sau đây, tạo ra các giá trị ứng viên mới bằng hàm tanh, tạo vector có giá trị từ -1 đến +1:

$$\hat{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

Cuối cùng, thông tin mới được tích hợp vào trạng thái tế bào thông qua công thức:

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t$$

Trong đó:

- W_i và W_c là các ma trận trọng số cho cổng đầu vào.
- b_i và b_c là các độ lệch tương ứng.
- i_t kiểm soát thông tin nào được cập nhật.
- \hat{C}_t chứa các giá trị ứng viên mới.
- C_t là trạng thái tế bào mới sau khi cập nhật

Nhiệm vụ cuối cùng trong quá trình xử lý thông tin của mạng LSTM được thực hiện bởi cổng đầu ra (Output Gate). Sau khi thông tin đã được xử lý qua cổng quên và cổng đầu vào, cổng đầu ra sẽ quyết định những thông tin nào từ trạng thái tế bào cần được trích xuất làm kết quả đầu ra.

Quy trình này bắt đầu bằng việc sử dụng hàm sigmoid để điều chỉnh thông tin từ đầu vào x_t và trạng thái ẩn trước đó h_{t-1} :

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

Kết quả sau đây được nhân với trạng thái tế bào hiện tại sau khi đã áp dụng hàm tanh:

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

Trong đó:

- W_o biểu diễn ma trận trọng số của cổng đầu ra
- b_o là độ lệch tương ứng
- o_t điều khiển mức độ thông tin được đưa ra từ trạng thái tế bào
- h_t là trạng thái ẩn mới và cũng là đầu ra tại thời điểm hiện tại t

Ưu và nhược điểm trong bài toán phân loại cảm xúc:

- **Ưu điểm:**

- Có khả năng nắm bắt phụ thuộc dài hạn trong văn bản, hiểu được ngữ cảnh toàn bộ câu hoặc đoạn.
- Giải quyết vấn đề gradient biến mất thông qua cơ chế cồng, phù hợp cho văn bản dài.
- Ghi nhớ thông tin quan trọng và lọc bỏ thông tin không cần thiết qua các cơ chế cồng.
- **Nhược điểm:**
 - Huấn luyện chậm do tính tuần tự trong xử lý.
 - Cấu trúc phức tạp với nhiều tham số cần điều chỉnh.
 - Tiêu tốn nhiều tài nguyên tính toán, đặc biệt với văn bản dài.
 - Có thể gặp khó khăn khi xử lý các văn bản quá dài do giới hạn bộ nhớ thực tế.

2.4. Các nghiên cứu giải quyết bài toán tiêu biểu

Trong những năm gần đây, bài toán phân tích cảm xúc trên mạng xã hội đã được nhiều nhà nghiên cứu trên thế giới quan tâm. Các phương pháp học máy truyền thống như Naive Bayes, SVM và Logistic Regression đã từng được sử dụng rộng rãi do khả năng đơn giản hóa pipeline xử lý và hiệu quả tương đối ổn định trên dữ liệu văn bản ngắn như Twitter [7][8]. Tuy nhiên, các mô hình này yêu cầu nhiều kỹ thuật trích xuất đặc trưng thủ công như TF-IDF hoặc Bag-of-Words, đồng thời khó mở rộng cho bài toán thời gian thực.

Với sự phát triển của học sâu, các mô hình như CNN và LSTM đã cho thấy hiệu quả vượt trội trong việc trích xuất ngữ cảnh và mối quan hệ dài hạn giữa các từ. Ví dụ, mô hình CNN của Dos Santos và Gatti (2014) đạt kết quả tốt trong việc phân tích văn bản ngắn trên Twitter [9], trong khi LSTM giúp cải thiện độ chính xác rõ rệt nhờ khả năng ghi nhớ chuỗi [10][11]. Những nghiên cứu trên bộ dữ liệu Sentiment140 và SemEval đã chứng minh các mô hình học sâu này đạt độ chính xác trên 85% [12][13].

Tuy nhiên, điểm yếu của học sâu là đòi hỏi tài nguyên tính toán lớn và độ trễ xử lý cao, không phù hợp trong bối cảnh xử lý dữ liệu luồng. Để khắc phục,

nhiều nghiên cứu đã tích hợp công nghệ Big Data như Apache Spark với các kỹ thuật học máy. Wang et al. (2016) [14] đã xây dựng hệ thống phân tích cảm xúc thời gian thực với Spark Streaming và Logistic Regression, xử lý hàng nghìn tweet mỗi giây với độ chính xác 83%. Gokulakrishnan et al. (2018) [15] đề xuất framework phân tán sử dụng Kafka và Spark, cho phép xử lý song song tới 5000 tweet/s, đạt độ chính xác 87%.

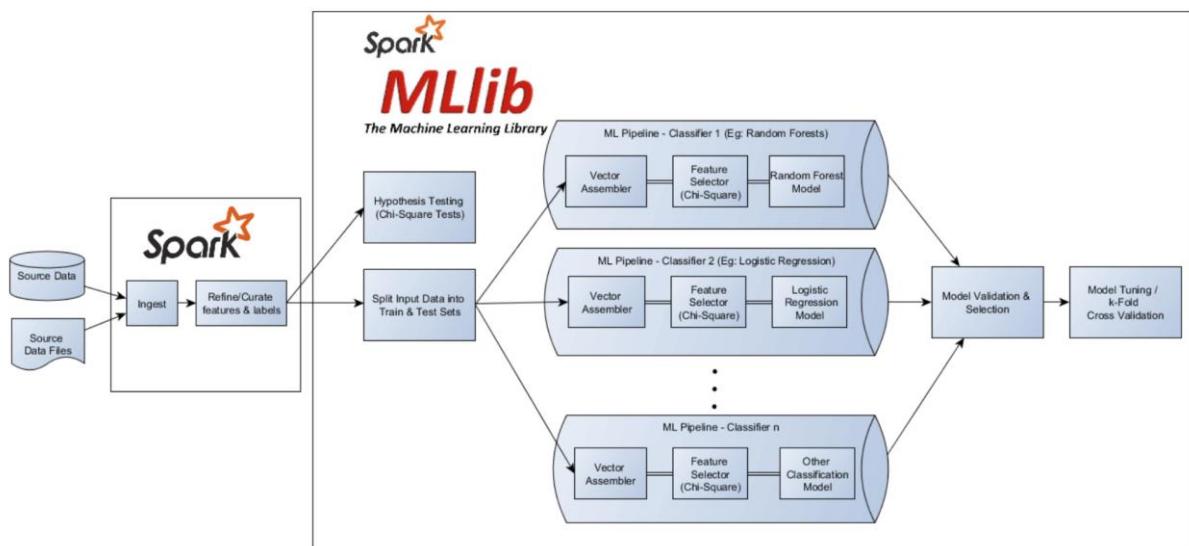
Rodriguez et al. (2020) [16] tiếp tục mở rộng bằng cách tối ưu các thuật toán như Random Forest, Naive Bayes trong Spark MLlib, cho thấy mô hình Random Forest đạt độ chính xác 88% trên dữ liệu lớn. Mới đây, Singh et al. (2022) [17] đề xuất hệ thống học online tự thích ứng với dòng dữ liệu mới, giúp cải thiện độ chính xác và phản ứng với sự thay đổi của cảm xúc theo thời gian thực.

CHƯƠNG 3: APACHE SPARK MLLIB VÀ CÁC CÔNG CỤ HỖ TRỢ TRIỂN KHAI

3.1. Tổng quan về Apache Spark MLlib

Apache Spark MLlib là thư viện học máy phân tán được phát triển như một phần của hệ sinh thái Apache Spark. MLlib được thiết kế đặc biệt để xử lý dữ liệu lớn và tính toán phân tán, cho phép các thuật toán học máy chạy hiệu quả trên các tập dữ liệu có kích thước lớn nằm trên nhiều máy tính.

MLlib cung cấp một bộ API thống nhất cho nhiều ngôn ngữ lập trình như Scala, Java, Python và R. Thư viện này bao gồm nhiều thuật toán học máy phổ biến và tiện ích cho phép xây dựng, đánh giá và điều chỉnh các pipelines học máy trên quy mô lớn.



Hình 3. 1 Minh họa Spark Mllib

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [18])

MLlib được xây dựng trên nền tảng của RDD (Resilient Distributed Datasets) - một cấu trúc dữ liệu phân tán cơ bản trong Spark, cho phép các phép toán song song trên các tập dữ liệu lớn. Ngoài ra, MLlib còn hỗ trợ DataFrame API cấp cao hơn, giúp đơn giản hóa việc tiền xử lý dữ liệu và xây dựng các pipeline học máy.

Ngoài ra, Mllib còn được tích hợp chặt chẽ với các thành phần khác của hệ sinh thái Spark như Spark SQL, Spark Streaming và GraphX, tạo nên một nền tảng toàn diện cho việc xử lý dữ liệu lớn và học máy.

3.2. Ưu điểm và lý do lựa chọn Spark Mllib

Có nhiều lý do khiến Apache Spark Mllib trở thành lựa chọn phù hợp cho bài toán phân loại cảm xúc người dùng mạng X theo thời gian thực:

- **Khả năng xử lý dữ liệu lớn:** Mạng X tạo ra lượng dữ liệu khổng lồ mỗi ngày, với hàng triệu bài đăng được chia sẻ. Spark Mllib được thiết kế để xử lý hiệu quả các tập dữ liệu lớn nhò vào kiến trúc phân tán của nó. Điều này cho phép hệ thống mở rộng theo chiều ngang bằng cách thêm nhiều node vào cluster khi khối lượng dữ liệu tăng lên.
- **Xử lý theo thời gian thực:** Spark Streaming, một thành phần của hệ sinh thái Spark, cho phép xử lý dữ liệu theo micro-batch, tức là dữ liệu được xử lý theo từng đợt nhỏ, gần như thời gian thực. Điều này lý tưởng cho các ứng dụng phân tích cảm xúc trên mạng xã hội, nơi cần phản ứng nhanh với các xu hướng hoặc sự kiện đang diễn ra.
- **Tích hợp liền mạch với hệ sinh thái Big Data:** Spark Mllib tích hợp dễ dàng với các công nghệ Big Data khác như Hadoop, Kafka và MongoDB. Điều này tạo điều kiện cho việc xây dựng một pipeline xử lý dữ liệu hoàn chỉnh từ việc thu thập dữ liệu từ mạng X, xử lý, phân tích, đến lưu trữ và hiển thị kết quả.
- **Hiệu suất cao:** Spark Mllib cung cấp hiệu suất cao hơn nhiều so với các framework xử lý dữ liệu lớn truyền thống như Hadoop MapReduce. Việc tính toán trong bộ nhớ (in-memory computing) và tối ưu hóa thực thi giúp Spark Mllib đạt tốc độ nhanh hơn 100 lần so với MapReduce trong một số trường hợp.
- **API đa ngôn ngữ và dễ sử dụng:** Spark Mllib cung cấp API cho nhiều ngôn ngữ lập trình phổ biến, bao gồm Python, Scala, Java và R. Điều này cho phép các nhà phát triển làm việc với ngôn ngữ họ quen thuộc.

- **Hỗ trợ cho các thuật toán phân loại văn bản:** MLlib cung cấp hỗ trợ mạnh mẽ cho các thuật toán phân loại văn bản, bao gồm các kỹ thuật tiền xử lý như tokenization, stopword removal, TF-IDF, và Word2Vec. Các thuật toán phân loại như Logistic Regression, SVM, và Naive Bayes được tối ưu hóa cho dữ liệu văn bản phân tán.
- **Khả năng cập nhật mô hình liên tục:** Đối với bài toán phân tích cảm xúc theo thời gian thực, việc cập nhật mô hình định kỳ để thích ứng với sự thay đổi của ngôn ngữ và xu hướng là rất quan trọng. Spark MLlib hỗ trợ việc cập nhật mô hình lặp đi lặp lại, cho phép hệ thống liên tục cải thiện độ chính xác của việc phân loại cảm xúc.

3.3. Các thuật toán được hỗ trợ trong Spark MLlib

Spark MLlib cung cấp một bộ thuật toán học máy toàn diện, đặc biệt hữu ích cho các bài toán phân loại cảm xúc. Dưới đây là các thuật toán chính được hỗ trợ trong MLlib:

3.3.1. Thuật toán phân loại

Spark MLlib hỗ trợ nhiều thuật toán phân loại, bao gồm:

- **Logistic Regression:** Thuật toán này là sự lựa chọn phổ biến cho phân loại nhị phân và đa lớp. MLlib cung cấp cả logistic regression với L1 và L2 regularization, giúp tránh overfitting và cải thiện khả năng tổng quát hóa.
- **Support Vector Machines (SVM):** Thuật toán mạnh mẽ cho các bài toán phân loại tuyến tính và phi tuyến. SVM trong MLlib sử dụng thuật toán SGD (Stochastic Gradient Descent) để huấn luyện trên dữ liệu phân tán.
- **Naive Bayes:** Một thuật toán đơn giản nhưng hiệu quả cho phân loại văn bản, đặc biệt là khi số lượng chiều của dữ liệu lớn. MLlib hỗ trợ cả Multinomial và Bernoulli Naive Bayes.
- **Decision Trees và Random Forests:** Các thuật toán này phù hợp cho dữ liệu có cấu trúc phức tạp. Random Forests trong MLlib là phiên bản phân tán của thuật toán truyền thống, cho phép xử lý tập dữ liệu lớn.

- **Gradient-Boosted Trees:** Thuật toán ensemble học tuần tự, xây dựng mô hình dự đoán bằng cách kết hợp nhiều mô hình cây quyết định đơn giản.
- **Multilayer Perceptron (MLP):** Một loại mạng neural đơn giản cho phân loại và hồi quy. MLlib hỗ trợ các kiến trúc MLP tùy chỉnh với nhiều lớp ẩn.

3.3.2. Các kỹ thuật tiền xử lý và trích xuất đặc trưng

Spark MLlib cung cấp nhiều công cụ cho việc tiền xử lý và trích xuất đặc trưng từ dữ liệu văn bản:

- **Tokenization:** Chuyển đổi văn bản thành các token riêng biệt.
- **Stopword Removal:** Loại bỏ các từ phổ biến không mang nhiều ý nghĩa.
- **n-gram:** Tạo ra các chuỗi từ liên tiếp để nắm bắt context.
- **TF-IDF (Term Frequency-Inverse Document Frequency):** Đo lường tầm quan trọng của từ trong văn bản.
- **Word2Vec:** Chuyển đổi từ thành các vector số trong không gian nhiều chiều, nắm bắt các mối quan hệ ngữ nghĩa.
- **CountVectorizer:** Chuyển đổi tập văn bản thành vector tần suất từ.
- **Feature Scaling:** Chuẩn hóa các đặc trưng để cải thiện hiệu suất của thuật toán.

3.3.3. Đánh giá mô hình

MLlib cung cấp các công cụ để đánh giá hiệu suất của mô hình phân loại:

- **Binary Classification Evaluator:** Tính toán các chỉ số như precision, recall, F1-score và AUC-ROC cho phân loại nhị phân.
- **Multiclass Classification Evaluator:** Đánh giá hiệu suất của các mô hình phân loại đa lớp.
- **Cross-validation:** Hỗ trợ đánh giá chéo để ước tính hiệu suất mô hình một cách chính xác hơn.

3.3.4. Tuning siêu tham số

MLlib cung cấp công cụ để tối ưu hóa siêu tham số của mô hình:

- **GridSearch:** Tìm kiếm lưới để tìm bộ siêu tham số tối ưu.

- **CrossValidator:** Kết hợp đánh giá chéo với tìm kiếm lưới để chọn mô hình tốt nhất.
- **TrainValidationSplit:** Một phiên bản đơn giản hơn của CrossValidator, sử dụng một tập validation duy nhất thay vì đánh giá chéo.

3.4. Mô hình Logistic Regression với Spark MLlib

Thuật toán hồi quy logistic (Logistic Regression) trong Apache Spark MLlib là một trong những phương pháp được ứng dụng rộng rãi trong bài toán phân loại nhị phân hoặc đa lớp. Đây là một biến thể đặc biệt của mô hình Tuyến tính tổng quát (Generalized Linear Models), ước lượng xác suất đầu ra cho các nhãn phân loại dựa trên hàm sigmoid (nhị phân) hoặc softmax (đa lớp).

Trong Spark MLlib, lớp LogisticRegression hỗ trợ hai biến thể chính:

- **Hồi quy logistic nhị phân** (binomial): Phân loại 2 nhãn.
- **Hồi quy logistic đa lớp** (multinomial): Phân loại nhiều nhãn.

Việc chọn biến thể được xác định bằng tham số family, hoặc Spark tự động suy luận dựa vào dữ liệu huấn luyện.

Chúng ta bắt đầu với việc tạo Spark session để giao tiếp với Spark và đọc dữ liệu (giả sử dữ liệu đã được clean):

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.classification import LogisticRegression

# Khởi tạo Spark session
spark = SparkSession.builder.appName("SentimentClassification").getOrCreate()

# Đọc dữ liệu CSV
df = spark.read.csv("path/to/training_data.csv", header=True, inferSchema=True)

# Các cột đặc trưng và nhãn
feature_cols = ['feature1', 'feature2', 'feature3']
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")
data = assembler.transform(df).select("features", "label")
```

Hình 3. 2 Code khởi tạo Spark session và đọc dữ liệu

Sau khi đã khởi tạo Spark session và đọc dữ liệu, các cột đặc trưng sẽ được tổng hợp và chuyển về dạng vector bằng VectorAssembler:

```
# Các cột đặc trưng và nhãn
feature_cols = ['feature1', 'feature2', 'feature3']
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")
data = assembler.transform(df).select("features", "label")
```

Hình 3. 3 Code tiền xử lý dữ liệu bằng VectorAssembler

Cuối cùng, chúng ta tiến hành tạo mô hình Logistic Regression và huấn luyện mô hình:

- Biến thể nhị phân:

```
# Khởi tạo mô hình Logistic Regression với biến thể nhị phân
lr = LogisticRegression(maxIter=20, regParam=0.3, elasticNetParam=0.8)
lr_model = lr.fit(data)
print("Coefficients:", lr_model.coefficients)
print("Intercept:", lr_model.intercept)
```

Hình 3. 4 Code khởi tạo mô hình Logistic Regression nhị phân

- Biến thể đa lớp:

```
# Khởi tạo mô hình Logistic Regression với biến thể đa lớp
mlr = LogisticRegression(maxIter=20, regParam=0.3, elasticNetParam=0.8, family="multinomial")
mlr_model = mlr.fit(data)
print("Coefficient Matrix:\n", mlr_model.coefficientMatrix)
print("Intercept Vector:", mlr_model.interceptVector)
```

Hình 3. 5 Code khởi tạo mô hình Logistic Regression đa lớp

Sau khi huấn luyện mô hình, ta có thể truy xuất hệ số (coefficients) và hệ số chêch (intercept) để hiểu rõ hơn về cách mô hình đưa ra quyết định:

- **Coefficients:**

- Là vector trọng số ứng với từng đặc trưng đầu vào.
- Biểu thị mức độ ảnh hưởng của mỗi đặc trưng đến xác suất dự đoán của mô hình.
- Hệ số dương → đặc trưng góp phần làm tăng xác suất phân lớp về phía lớp dương (ví dụ: tích cực).
- Hệ số âm → đặc trưng làm giảm xác suất.

- **Intercept:**

- Là giá trị ngưỡng (bias) giúp dịch chuyển hàm sigmoid.

- Trong không gian tuyến tính, nó xác định nơi đường phân cách cắt trực.
- Ngoài ra, sau khi huấn luyện, ta còn có thể truy xuất các chỉ số đánh giá mô hình như Accuracy, F1-Score, Precision và Recall. Đặc biệt, đối với biến thể nhị phân, có thể vẽ đường cong ROC và tính giá trị AUC để đánh giá khả năng phân biệt của mô hình một cách trực quan và chi tiết hơn

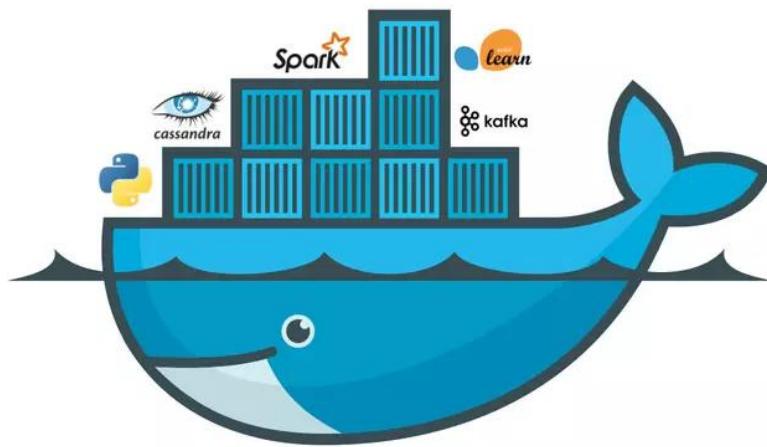
3.5. Các công nghệ hỗ trợ triển khai

Để có thể triển khai dự án phân loại cảm xúc người dùng mạng X theo thời gian thực, chúng ta cần sử dụng kết hợp nhiều công nghệ hiện đại để tạo thành một kiến trúc xử lý dữ liệu phân tán hiệu quả. Dưới đây là tổng quan về các công nghệ chính được sử dụng trong kiến trúc hệ thống.

3.5.1. Docker

Docker là một nền tảng phần mềm cho phép bạn xây dựng, kiểm tra và triển khai ứng dụng một cách nhanh chóng. Docker đóng gói phần mềm vào các đơn vị tiêu chuẩn hóa gọi là container, chứa mọi thứ mà phần mềm cần để chạy, bao gồm thư viện, công cụ hệ thống, mã và thời gian chạy. Các khái niệm cơ bản của Docker bao gồm:

- **Container:** Là một đơn vị phần mềm nhẹ, độc lập và có thể thực thi được, bao gồm mọi thứ cần thiết để chạy ứng dụng.
- **Image:** Là một gói không thay đổi chứa tất cả các phụ thuộc và cấu hình cần thiết để chạy ứng dụng.
- **Dockerfile:** Là một tệp văn bản chứa các hướng dẫn để xây dựng Docker image.
- **Docker Hub:** Là một kho lưu trữ trực tuyến nơi bạn có thể tìm và chia sẻ các container image.
- **Docker Compose:** Là công cụ cho phép định nghĩa và chạy các ứng dụng Docker đa container.



Hình 3. 6 Docker

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [19])

Lợi ích của Docker:

- **Nhất quán môi trường:** Loại bỏ vấn đề code hoạt động tốt trên máy phát triển nhưng lại gặp lỗi khi triển khai nơi khác.
- **Nhỏ và nhanh:** Container chia sẻ kernel của hệ điều hành máy chủ.
- **Cô lập:** Các ứng dụng chạy độc lập không ảnh hưởng lẫn nhau.
- **Khả năng mở rộng:** Dễ dàng mở rộng và quản lý các dịch vụ.
- **Triển khai đơn giản:** Tăng tốc quá trình phát triển, kiểm tra và triển khai.

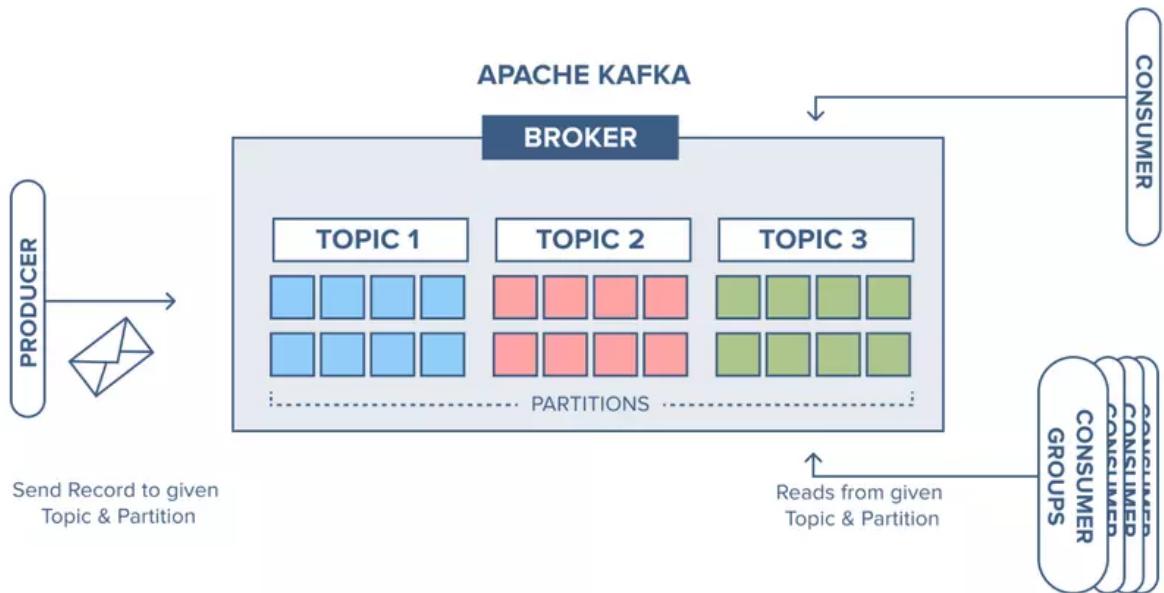
Docker không chỉ đơn thuần là một công cụ kỹ thuật mà còn là một yếu tố chiến lược giúp đơn giản hóa quy trình phát triển, triển khai và mở rộng ứng dụng.

3.5.2. Apache Kafka

Apache Kafka là một nền tảng phân tán xử lý luồng dữ liệu (distributed streaming platform) được phát triển bởi LinkedIn vào năm 2011, sau đó được đóng góp cho Apache Software Foundation và trở thành một dự án mã nguồn mở. Các đặc điểm chính của Kafka gồm có:

- **Kiến trúc phân tán:** Kafka chạy dưới dạng cluster trên một hoặc nhiều máy chủ, có khả năng mở rộng dễ dàng.
- **Xử lý theo thời gian thực:** Cho phép xử lý dữ liệu theo thời gian thực với hiệu suất cao, độ trễ thấp.

- **Độ bền cao:** Dữ liệu được nhân bản (replicated) và lưu trữ trên đĩa, giúp chống lại sự cố hệ thống.
- **Khả năng mở rộng:** Có thể mở rộng từ vài máy chủ đến hàng trăm máy chủ mà không bị gián đoạn.

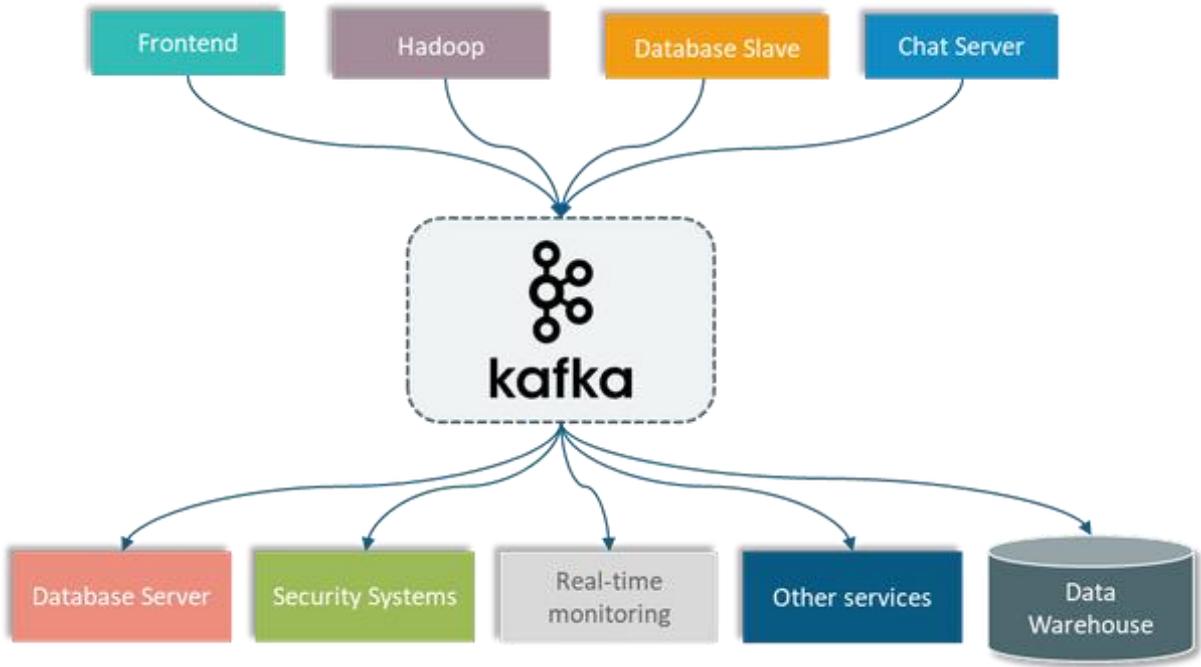


Hình 3. 7 Cấu trúc Kafka

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [20])

Cấu trúc của Kafka:

- **Broker:** Các máy chủ Kafka trong cluster.
- **Topic:** Một luồng dữ liệu với tên cụ thể.
- **Partition:** Mỗi topic được chia thành nhiều phần nhỏ.
- **Producer:** Ứng dụng gửi dữ liệu vào Kafka.
- **Consumer:** Ứng dụng đọc dữ liệu từ Kafka.
- **Consumer Group:** Nhóm các consumer làm việc cùng nhau.
- **ZooKeeper:** Dịch vụ điều phối cluster (trong các phiên bản mới, Kafka đang dần loại bỏ phụ thuộc vào ZooKeeper).



Hình 3. 8 Ứng dụng của Kafka

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [21])

Một số ứng dụng của Kafka:

- **Xử lý luồng sự kiện:** Thu thập và xử lý dữ liệu từ nhiều nguồn.
- **Phân tích dữ liệu:** Kết hợp với các công cụ phân tích như Spark, Flink.
- **Tích hợp hệ thống:** Kết nối các hệ thống riêng biệt thông qua luồng dữ liệu.
- **Giám sát và theo dõi:** Thu thập log và metric từ nhiều nguồn.
- **Xử lý dữ liệu theo thời gian thực:** Ứng dụng trong phát hiện gian lận, để xuất sản phẩm, v.v.

Ưu điểm:

- **Hiệu suất cao:** Có thể xử lý hàng triệu tin nhắn mỗi giây.
- **Khả năng mở rộng tốt:** Dễ dàng thêm/bớt các node.
- **Độ tin cậy cao:** Dữ liệu được nhân bản và bền vững.
- **Hỗ trợ nhiều ngôn ngữ lập trình:** Java, Python, Go, .NET, và nhiều ngôn ngữ khác.

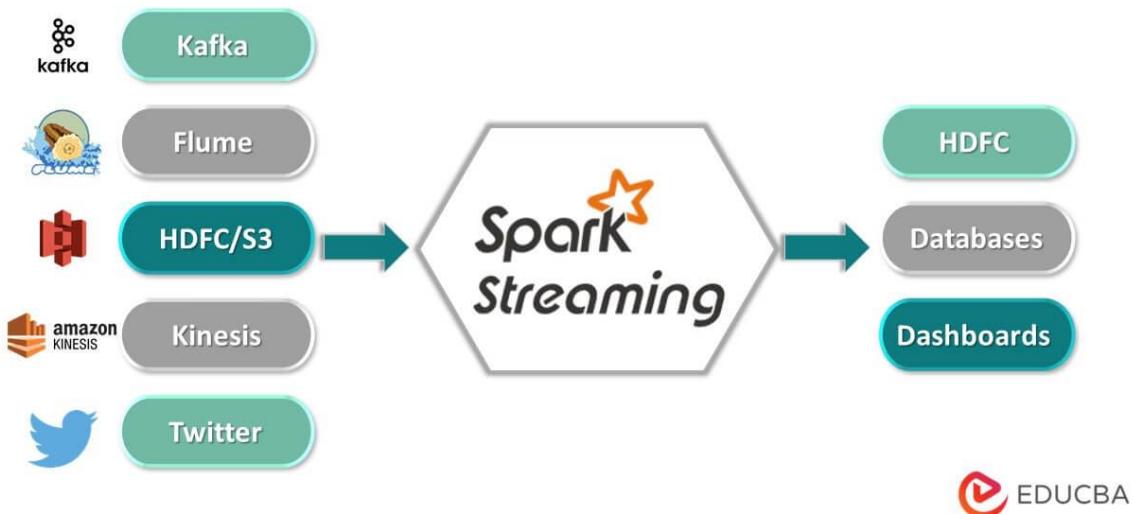
Apache Kafka đã trở thành một thành phần quan trọng trong kiến trúc dữ liệu hiện đại, đặc biệt trong các hệ thống xử lý dữ liệu lớn và ứng dụng thời gian thực.

3.5.3. Spark Streaming

Spark Streaming là một phần mở rộng của Apache Spark, được thiết kế để xử lý luồng dữ liệu (stream data) theo thời gian thực. Đây là một công nghệ mạnh mẽ cho phép xử lý dữ liệu liên tục từ nhiều nguồn khác nhau như Kafka, Flume, Kinesis hoặc TCP socket. Spark Streaming gồm có các đặc điểm chính sau:

- Mô hình xử lý micro-batch:** Spark Streaming chia luồng dữ liệu vào thành các batch nhỏ và xử lý chúng theo khoảng thời gian (batch interval).
- Tích hợp với Spark Core:** Được xây dựng trên nền tảng Spark nên có thể sử dụng các tính năng như RDD và các thuật toán phân tích phức tạp.
- Khả năng chịu lỗi cao:** Phục hồi tự động khi xảy ra lỗi và đảm bảo mỗi đơn vị dữ liệu được xử lý đúng một lần (exactly-once semantics).
- Mở rộng dễ dàng:** Có thể mở rộng đến hàng trăm node để xử lý hàng triệu sự kiện mỗi giây.

Spark Streaming



Hình 3. 9 Spark Streaming

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [22])

Ứng dụng của Spark Streaming:

- **Phân tích dữ liệu thời gian thực:** Phát hiện bất thường, phân tích cảm xúc, xu hướng thị trường.
- **Xử lý các sự kiện IoT:** Phân tích dữ liệu từ các thiết bị IoT.
- **Giám sát hệ thống:** Theo dõi và phân tích log, metric từ các hệ thống.
- **Phát hiện gian lận:** Phát hiện các hoạt động đáng ngờ trong thời gian thực.
- **ETL thời gian thực:** Trích xuất, biến đổi và tải dữ liệu từ nhiều nguồn.

Khác với Kafka tập trung vào việc lưu trữ và truyền tải dữ liệu, Spark Streaming tập trung vào xử lý và phân tích. Spark Streaming là lựa chọn phổ biến cho các ứng dụng yêu cầu xử lý dữ liệu theo thời gian thực kết hợp với các phân tích phức tạp, đặc biệt khi đã có sẵn hệ sinh thái Spark trong tổ chức.

3.5.4. MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL hướng tài liệu (document-oriented database), được phát triển bởi MongoDB Inc. từ năm 2007. Đây là một trong những cơ sở dữ liệu NoSQL phổ biến nhất hiện nay. MongoDB gồm có các đặc điểm chính sau:

- **Mô hình dữ liệu document-based:** Dữ liệu được lưu trữ dưới dạng các tài liệu BSON (Binary JSON), cho phép cấu trúc linh hoạt và phức tạp.
- **Schema linh hoạt:** Không yêu cầu schema cố định, các tài liệu trong cùng một collection có thể có cấu trúc khác nhau.
- **Khả năng mở rộng:** Hỗ trợ sharding (phân đoạn) ngang để phân phối dữ liệu trên nhiều máy chủ.
- **Hiệu suất cao:** Tối ưu hóa cho việc đọc và ghi dữ liệu với khối lượng lớn.
- **Hỗ trợ các thao tác phức tạp:** Aggregation framework, truy vấn địa lý, tìm kiếm văn bản đầy đủ.



mongo DB

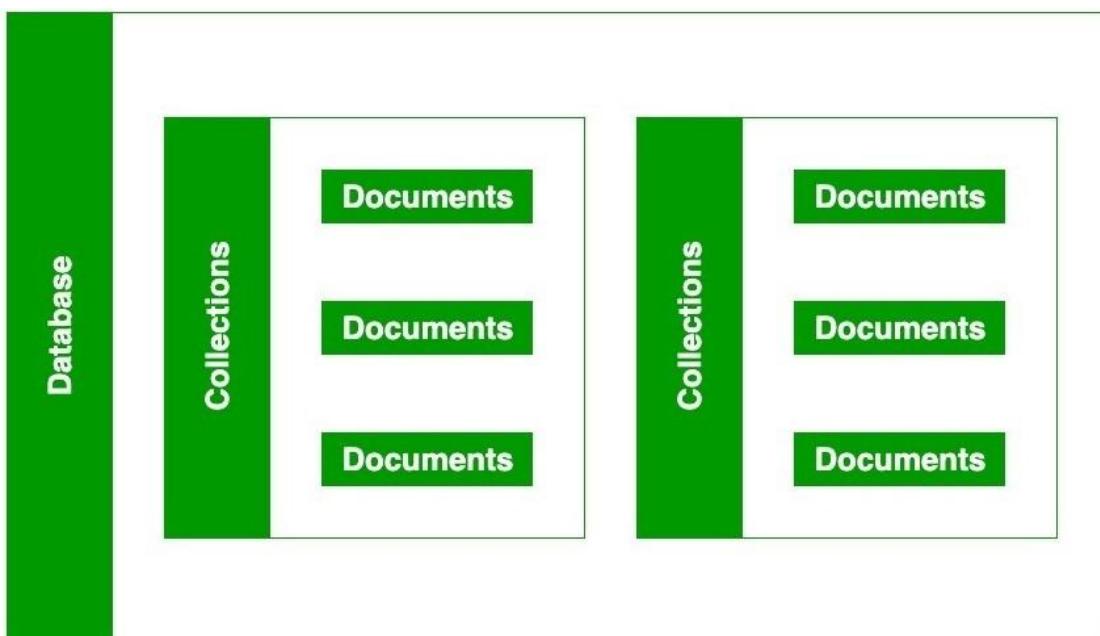
Hình 3. 10 MongoDB

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [23])

Cấu trúc của MongoDB:

- **Database:**
 - Đây là container cấp cao nhất trong cấu trúc MongoDB
 - Mỗi database chứa dữ liệu cho một ứng dụng hoặc dự án cụ thể
 - MongoDB cho phép nhiều database tồn tại độc lập trong cùng một máy chủ
 - Mỗi database có các file riêng biệt trên hệ thống lưu trữ
 - Có chính sách bảo mật và quyền truy cập riêng
- **Collections:**
 - Là nhóm các document liên quan trong database
 - Tương đương với khái niệm "bảng" trong hệ thống CSDL quan hệ
 - Không yêu cầu schema cố định
 - Thường được đặt tên theo loại đối tượng mà nó lưu trữ (users, products, orders...)
 - Được tạo tự động khi document đầu tiên được chèn vào
- **Documents:**
 - Đơn vị lưu trữ dữ liệu cơ bản của MongoDB

- Được lưu dưới dạng BSON (Binary JSON)
- Mỗi document có một định danh duy nhất thông qua trường "_id"
- Có thể chứa các cấu trúc dữ liệu phức tạp như mảng và document lồng nhau
- Kích thước tối đa cho mỗi document là 16MB
- Các document trong cùng collection có thể có cấu trúc khác nhau (schema-less)).



Hình 3. 11 Cấu trúc Mongo DB

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [24])

MongoDB được thiết kế để đáp ứng nhu cầu lưu trữ và truy xuất dữ liệu phi cấu trúc hoặc bán cấu trúc, cung cấp hiệu suất cao và khả năng mở rộng dễ dàng. Kiến trúc này đặc biệt phù hợp cho các ứng dụng web hiện đại, phân tích dữ liệu lớn và các ứng dụng yêu cầu sự linh hoạt cao trong việc quản lý dữ liệu.

3.5.5. Django

Django là một framework web mã nguồn mở được phát triển bằng Python, được tạo ra để giúp các nhà phát triển xây dựng các ứng dụng web một cách nhanh chóng và hiệu quả. Django được thiết kế theo triết lý "batteries-included" (tích hợp sẵn nhiều công cụ), cung cấp nhiều thành phần và tính năng sẵn có mà không

cần phải cài đặt thêm các thư viện bên ngoài. Các đặc điểm chính của Django gồm có:

- **Mô hình MVC (Model-View-Controller):** Django sử dụng mô hình MVT (Model-View-Template) tương tự như MVC, giúp tách biệt các thành phần logic của ứng dụng.
- **ORM (Object-Relational Mapping):** Django có hệ thống ORM mạnh mẽ cho phép tương tác với cơ sở dữ liệu thông qua các đối tượng Python mà không cần viết trực tiếp SQL.
- **Admin interface:** Django cung cấp sẵn giao diện quản trị mạnh mẽ và dễ tùy biến.
- **Authentication:** Hệ thống xác thực và phân quyền đầy đủ.
- **Routing URLs:** Hệ thống định tuyến URL linh hoạt.
- **Template system:** Hệ thống template mạnh mẽ để hiển thị dữ liệu.
- **Form handling:** Xử lý biểu mẫu và xác thực dữ liệu.
- **Security:** Nhiều tính năng bảo mật được tích hợp sẵn như chống tấn công CSRF, XSS, SQL injection.

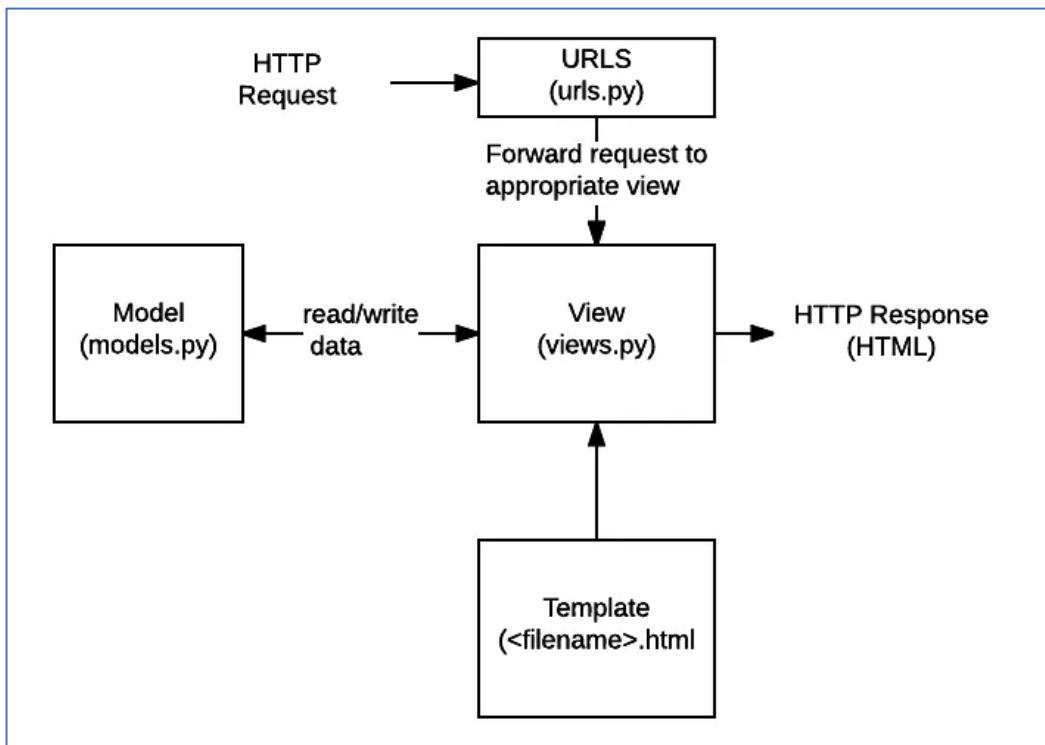


Hình 3. 12 Django

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [25])

Thành phần của Django bao gồm các cài đặt cấu hình cơ sở dữ liệu, tùy chọn cùm thẻ và các cài đặt cùm thẻ trong dự án. Trong đó:

- **manage.py:** Giúp người dùng tương tác với Django theo các cách khác nhau.
- **__init__.py:** Được coi là một dạng python package, chủ yếu là trống.
- **settings.py:** Tập tin cấu hình.
- **urls.py:** Tổng hợp tất cả các khai báo URL của Django và mục lục của website Django.
- **wsgi.py:** Lối vào cho server website tương thích WSGI để thao tác với dự án của người dùng.



Hình 3. 13 Các thành phần cơ bản của Django

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo [26])

Django đã trở thành lựa chọn của nhiều công ty và tổ chức lớn trên thế giới nhờ tính linh hoạt, khả năng mở rộng và bảo mật mạnh mẽ, minh chứng qua việc được sử dụng bởi nhiều trang web nổi tiếng như Instagram với khả năng xử lý hàng tỷ tương tác mỗi ngày, Pinterest với các API và dịch vụ web, Mozilla trong nhiều dự án bao gồm hệ thống hỗ trợ người dùng, Spotify cho backend dịch vụ âm nhạc trực tuyến, Disqus với nền tảng bình luận xử lý hàng tỷ lượt xem, cùng nhiều tổ chức khác như Dropbox và The Washington Post.

CHƯƠNG 4: THỰC NGHIỆM VÀ XÂY DỰNG SẢN PHẨM DEMO

4.1. Dữ liệu thực nghiệm

Quá trình thực nghiệm được tôi tiến hành trên bộ dữ liệu Twitter Sentiment Analysis [27] từ Kaggle. Bộ dữ liệu này tập trung vào phân tích cảm xúc liên quan đến các thực thể (entity) được đề cập trong các tweet trên Twitter.

Bộ dữ liệu có xấp xỉ 74,000 dòng với 4 nhãn là positive, negative, neutral và irrelevant. Bộ dữ liệu được chia làm 2 tập là twitter_training.csv và twitter_validation.csv với các trường như sau:

- Id: id của bài viết
- Entity: thực thể đăng bài viết
- Label: nhãn của bài viết
- Tweet content: nội dung của bài viết

#	id	Entity	Label	Tweet content
6789		Fortnite	Irrelevant	He said told u I'm getting in that box of a brain dead controller player.
4115		CS-GO	Positive	Yo this looks LIT! CS: GO / Overwatch combo
5665		HomeDepot	Negative	@HomeDepot attention executive administrators. If ever your stores are getting big delivery orders sent for pallets of bricks to downtown
9369		Overwatch	Irrelevant	Guy has notified me and says that my name has been forwarded to a litter list lmaoooo get the fuck off overwatch you strange ass duc
2476		Borderlands	Positive	F Loving the new DLC!!!. RhandlerR RhandlerR RhandlerR pic.twitter.com/PKZJxIGLj
11558		TomClancysRainbowSix	Negative	@ Rainbow6Game out on Xbox
1207		Battlefield	Positive	you miss those Battlefield 1 days
10458		RedDeadRedemption(RDR)	Irrelevant	lmao gamer jason could kill me for it
4481		Google	Positive	Most popular Google results in the previous hour.. 08/17/2020, 01:01:50. Europa League. Flash Arrow. Lovecraft Country. Antifa. Alex G
4842		GrandTheftAuto(GTA)	Irrelevant	thinkin' about gta strippers rn.. About this one.

Hình 4. 1 Mẫu dữ liệu huấn luyện và kiểm thử.

4.2. Tiền xử lý dữ liệu

Dữ liệu văn bản từ mạng xã hội Twitter thường chứa nhiều thông tin nhiễu như liên kết, hashtag, ký tự đặc biệt. Quá trình tiền xử lý được thực hiện để làm sạch và chuẩn hóa dữ liệu nhằm cải thiện hiệu suất của mô hình. Các bước tiền xử lý được thực hiện như sau:

- **Loại bỏ dữ liệu trống:** Dùng hàm drop để xóa những dòng trống.

- Chuyển đổi nhãn về số:** Sử dụng StringIndexer để chuyển đổi các nhãn văn bản (Positive, Negative, Irrelevant) thành dạng số để phù hợp với yêu cầu của thuật toán học máy.
- Làm sạch văn bản:** Loại bỏ liên kết URL, hashtag, mention và ký tự đặc biệt; chuyển văn bản về chữ thường để đảm bảo tính nhất quán trong dữ liệu.
- Xây dựng pipeline xử lý và huấn luyện:** Thiết lập quy trình tự động hóa bao gồm tokenization, loại bỏ stopwords và trích xuất đặc trưng, tạo thành một đường ống xử lý dữ liệu hoàn chỉnh cho quá trình huấn luyện mô hình.

# id	Entity	# Label2	Tweet content
6789	Fortnite	3.0	he said told u im getting in that box of a brain dead controller player
4115	CS-GO	1.0	yo this looks lit cs go overwatch combo
5665	HomeDepot	0.0	attention executive administrators if ever your stores are getting big delivery orders sent for pallets of bricks to down
9369	Overwatch	3.0	guy has notified me and says that my name has been forwarded to a litter list lmaoooo get the fuck off overwatch yo
2476	Borderlands	1.0	f loving the new dlc rhandlerr rhandlerr rhandlerr pictwittercompkjxlgxlj
11558	TomClancysRainbowSix	0.0	rainbowgame out on xbox
1207	Battlefield	1.0	you miss those battlefield days
10458	RedDeadRedemption(RDR)	3.0	lmao gamer jason could kill me for it
4481	Google	1.0	most popular google results in the previous hour europa league flash arrow lovecraft country antifa alex g
4842	GrandTheftAuto(GTA)	3.0	thinkin about gta strippers rn about this one

Hình 4. 2 Dữ liệu sau khi được làm sạch.

```

# Define tokenizer
tokenizer = Tokenizer(inputCol="Tweet content", outputCol="tokens")

# Define stopwords remover
stopwords_remover = StopWordsRemover(inputCol="tokens", outputCol="filtered_tokens", stopWords=stop_words)

# Define CountVectorizer
count_vectorizer = CountVectorizer(inputCol="filtered_tokens", outputCol="features", vocabSize=10000, minDF=5)

# Define Logistic Regression
lr = LogisticRegression(maxIter=10, labelCol="Label2", featuresCol="features")

# create the pipeline
pipeline = Pipeline(stages=[tokenizer, stopwords_remover, count_vectorizer, lr])

```

Hình 4. 3 Code Pipeline xử lý và huấn luyện.

4.3. Huấn luyện mô hình

Các mô hình được nghiên cứu trong đồ án này được triển khai trên môi trường máy tính cá nhân với ngôn ngữ lập trình Python, sử dụng framework Apache Spark thông qua thư viện PySpark. Để đảm bảo tính khách quan và tìm ra phương pháp hiệu quả nhất cho bài toán phân loại cảm xúc từ tweet, tôi đã tiến hành thử nghiệm với ba mô hình học máy phổ biến:

- Logistic Regression
- Random Forest
- Naive Bayes

Quá trình huấn luyện và đánh giá mô hình được thực hiện thông qua kỹ thuật Grid Search kết hợp với Cross-validation nhằm tối ưu hóa hiệu suất cho từng mô hình. Phương pháp Cross-validation với 5 folds được áp dụng nhằm đảm bảo đánh giá khách quan, tăng khả năng tổng quát hóa và hạn chế hiện tượng overfitting. Độ chính xác (accuracy) được chọn làm thước đo chính để so sánh hiệu suất giữa các mô hình trong nghiên cứu này.

```

lr = LogisticRegression(maxIter=10, labelCol="Label2", featuresCol="features")

pipeline = Pipeline(stages=[tokenizer, stopwords_remover, count_vectorizer, lr])

paramGrid = (ParamGridBuilder()
    .addGrid(lr.maxIter, [10, 20, 50])
    .addGrid(lr.regParam, [0.0, 0.1, 0.5])
    .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0])
    .build())

evaluator = MulticlassClassificationEvaluator(labelCol="Label2",
    predictionCol="prediction",
    metricName="accuracy")

crossval = CrossValidator(estimator=pipeline,
    estimatorParamMaps=paramGrid,
    evaluator=evaluator,
    numFolds=5)

cvModel = crossval.fit(cleaned_data)
cvModel.bestModel.save("LR_best_model")

```

Hình 4. 4 Code Grid search cho mô hình Logistic Regression

Sau quá trình tìm kiếm siêu tham số tối ưu, mô hình đạt hiệu suất cao nhất từ mỗi thuật toán được lựa chọn và lưu trữ để tiến hành dự đoán trên tập dữ liệu kiểm thử, từ đó thực hiện đánh giá hiệu suất cuối cùng của các mô hình.

4.4. Các kết quả thực nghiệm

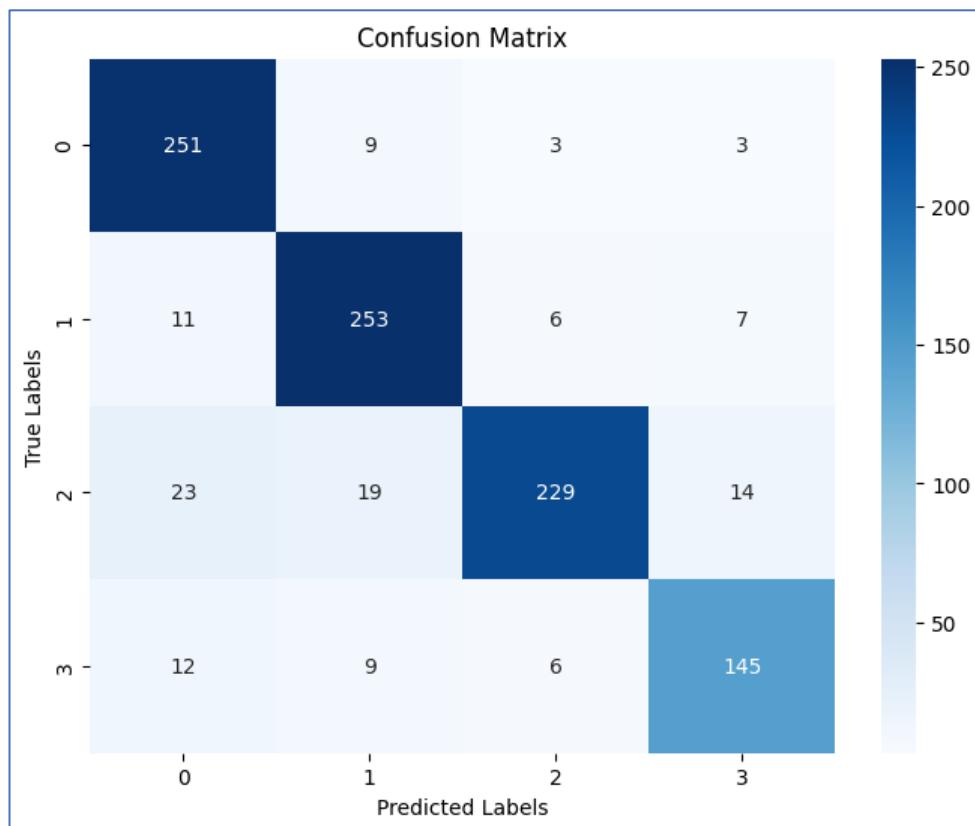
Kết quả thu được từ các thực nghiệm trong nghiên cứu này được thể hiện chi tiết qua Bảng 4.1 và các hình ảnh về confusion matrix của mỗi mô hình dưới

đây. Các số liệu này phản ánh hiệu suất của ba mô hình phân loại cảm xúc khi áp dụng trên tập dữ liệu Twitter Sentiment Analysis.

Bảng 4. 1 Metrics của các mô hình

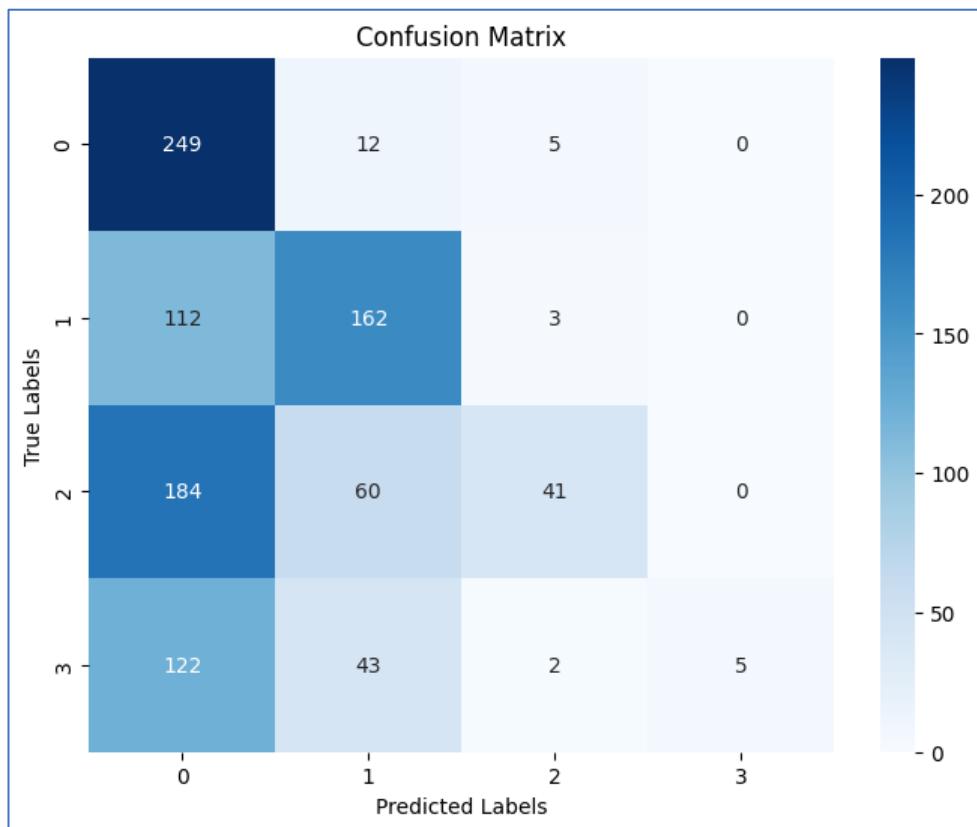
Model \ Metrics	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.878	0.881513	0.878	0.877403
Random Forest	0.457	0.662419	0.457	0.383252
Naive Bayes	0.740	0.747648	0.740	0.736290

Dựa trên các chỉ số đánh giá chính, mô hình Logistic Regression thể hiện hiệu suất vượt trội với độ chính xác đạt 87.8%, cùng với các chỉ số Precision, Recall và F1 Score đều đạt trên 87.7%. Ngược lại, mô hình Random Forest cho kết quả kém nhất với độ chính xác chỉ đạt 45.7% và F1 Score thấp ở mức 38.3%. Mô hình Naive Bayes có hiệu suất ở mức trung bình với độ chính xác 74.0%.



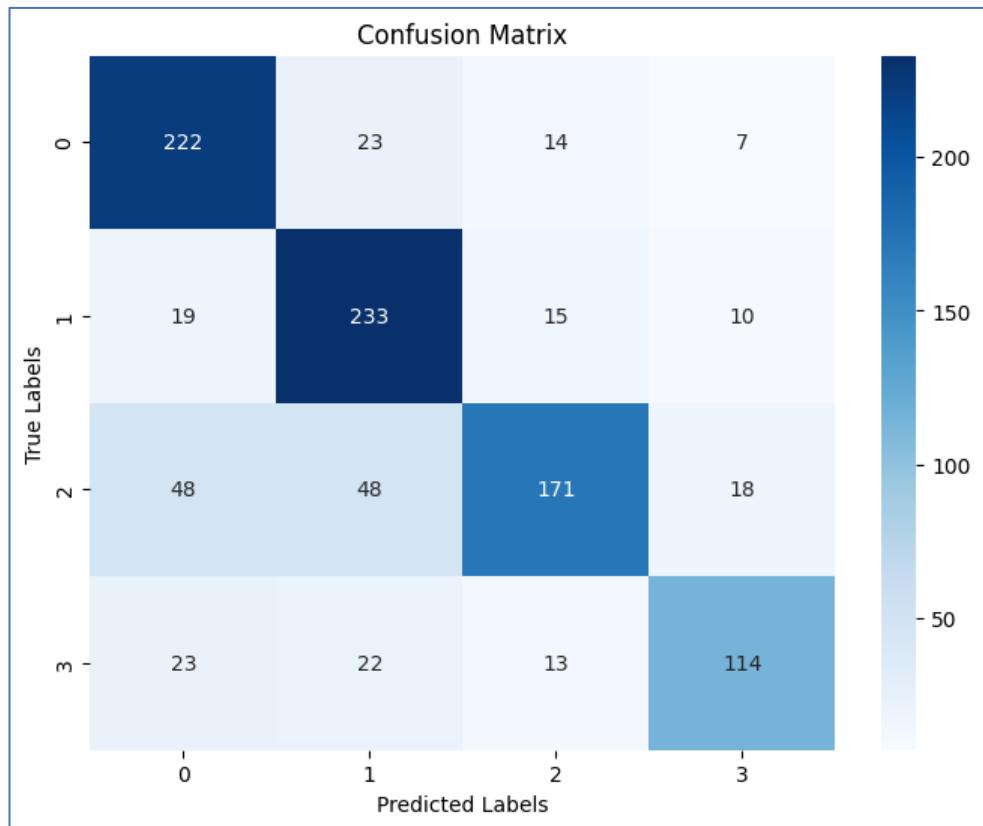
Hình 4. 5 Confusion matrix của Logistic Regression

Phân tích các Confusion matrix ta thấy Logistic Regression có khả năng phân loại chính xác đối với tất cả các nhãn. Cụ thể, mô hình phân loại đúng 251/266 mẫu nhãn negative (nhãn 0), 253/277 mẫu nhãn positive (nhãn 1), 229/285 mẫu nhãn neutral (nhãn 2) và 145/172 mẫu nhãn irrelevant (nhãn 3). Sự cân bằng này trong việc phân loại các nhãn khác nhau đã đóng góp đáng kể vào hiệu suất tổng thể của mô hình.



Hình 4. 6 Confusion matrix của Random Forest

Đối với Random Forest, mặc dù mô hình có thể phân loại tương đối tốt đối với nhãn negative (249 mẫu đúng), nhưng lại gặp khó khăn đáng kể với các nhãn còn lại. Đặc biệt, mô hình này có xu hướng thiên vị nghiêm trọng về nhãn negative (nhãn 0), thể hiện qua việc nhiều mẫu thuộc các nhãn khác bị phân loại nhầm sang nhãn này. Chẳng hạn, có tới 112 mẫu nhãn positive (nhãn 1), 184 mẫu nhãn neutral (nhãn 2) và 122 mẫu nhãn irrelevant (nhãn 3) bị phân loại sai thành nhãn negative, điều này góp phần làm giảm đáng kể hiệu suất tổng thể của mô hình.



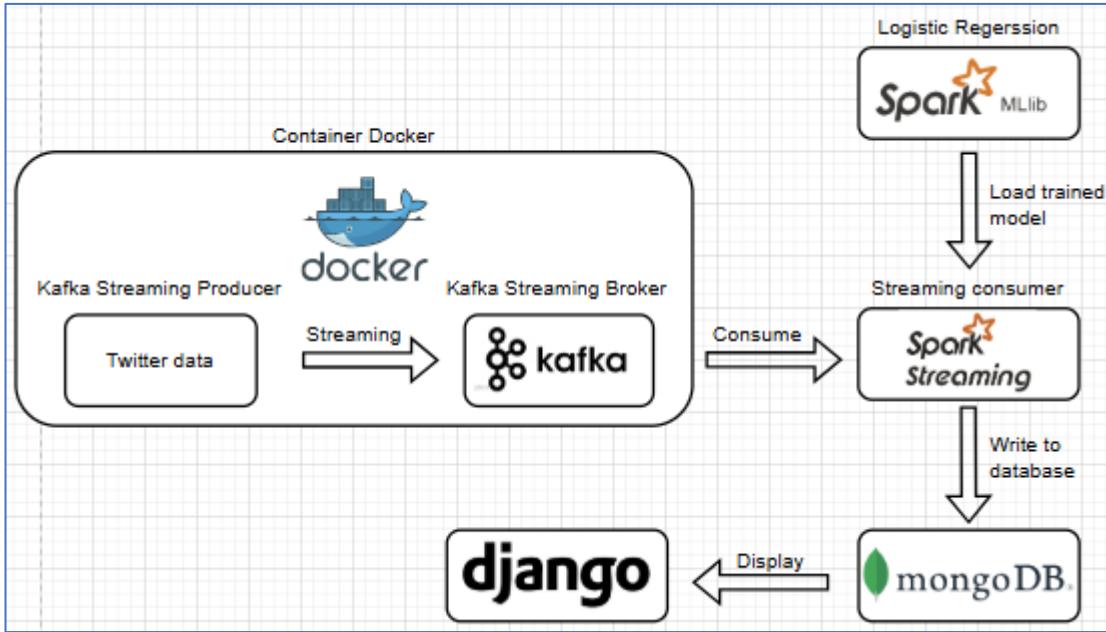
Hình 4. 7 Confusion Matrix của Naive Bayes

Mô hình Naive Bayes thể hiện hiệu suất tốt nhất khi phân loại các tweet có nhãn positive (nhãn 1) với 233/277 mẫu được phân loại chính xác. Tuy nhiên, hiệu suất của mô hình này còn hạn chế đối với các nhãn còn lại, đặc biệt là nhãn neutral (nhãn 2) với chỉ 171/285 mẫu được phân loại đúng. Kết quả này cho thấy mô hình Naive Bayes có khả năng phân biệt tốt các tweet mang tính tích cực nhưng còn hạn chế trong việc nhận diện các sắc thái cảm xúc trung lập.

Từ kết quả thực nghiệm, có thể kết luận rằng mô hình Logistic Regression là lựa chọn phù hợp nhất cho bài toán phân tích cảm xúc trên dữ liệu Twitter trong nghiên cứu này. Mô hình này không chỉ đạt được độ chính xác cao nhất mà còn thể hiện sự cân bằng tốt trong việc phân loại các nhãn khác nhau, điều quan trọng đối với các ứng dụng phân tích cảm xúc trong thực tế.

4.5. Phân tích thiết kế hệ thống

4.5.1. Kiến trúc hệ thống



Hình 4. 8 Kiến trúc hệ thống

Toàn bộ hệ thống phân tích cảm xúc được triển khai trong môi trường Container Docker, đảm bảo tính đồng nhất và dễ dàng trong việc triển khai. Dữ liệu Twitter được đưa vào hệ thống thông qua Kafka Streaming Producer, với nguồn dữ liệu từ file `twitter_validation.csv`. Kafka Streaming Broker đóng vai trò trung tâm điều phối dòng dữ liệu, tạo ra sự tách biệt giữa nguồn dữ liệu và các thành phần xử lý, giúp hệ thống có khả năng mở rộng và xử lý linh hoạt với lượng tweet tương đối lớn.

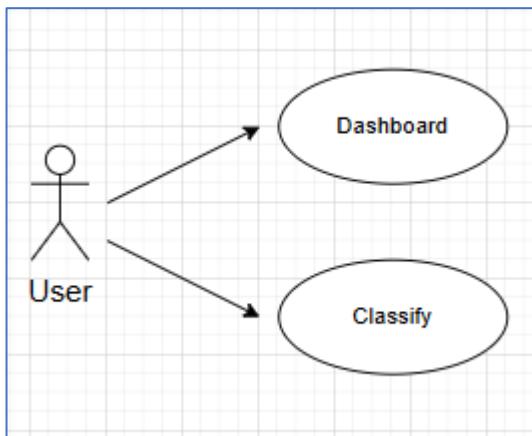
Phản xử lý và phân tích dữ liệu được thực hiện bởi Spark Streaming Consumer, sử dụng mô hình Logistic Regression đã được huấn luyện trước đó bằng thư viện Spark MLlib. Mô hình này được lựa chọn dựa trên kết quả thực nghiệm với độ chính xác cao nhất (87.8%) so với các mô hình khác như Random Forest và Naive Bayes. Spark Streaming tiêu thụ dữ liệu từ Kafka và thực hiện phân tích gần thời gian thực, áp dụng mô hình đã huấn luyện để dự đoán cảm xúc của các tweet.

Kết quả phân tích được lưu trữ trong MongoDB, một cơ sở dữ liệu NoSQL phù hợp cho việc xử lý dữ liệu phi cấu trúc và bán cấu trúc. Việc lựa chọn MongoDB giúp hệ thống linh hoạt trong việc lưu trữ và truy xuất thông tin tweet cùng với kết quả phân tích cảm xúc. Cuối cùng, Django Framework được sử dụng

để xây dựng giao diện người dùng, hiển thị kết quả phân tích dưới dạng trực quan và dễ hiểu.

4.5.2. Biểu đồ use case

Biểu đồ use case tổng quan



Hình 4. 9 Biểu đồ use case tổng quát

4.5.3. Đặc tả use case

- Đặc tả use case Dashboard

Bảng 4. 2 Đặc tả use case Dashboard

Mã use case	UC1
Tên use case	Dashboard
Tóm tắt	Use case cho phép người dùng xem được kết quả phân loại của dữ liệu và các thống kê dữ liệu cơ bản
Actor	User
Tiền điều kiện	Trình duyệt được bật
Đảm bảo tối thiểu	Người dùng mở được trang Dashboard
Đảm bảo thành công	Người dùng xem được kết quả phân loại và các thống kê dữ liệu
Kích hoạt	Người dùng nhấp chọn nút "Dashboard" trên giao diện
Luồng các sự kiện:	
Luồng cơ bản:	

- | |
|--|
| <p>1) Người dùng nhập chọn nút “Dashboard” và màn hình hiển thị ra kết quả phân loại và các thống kê cơ bản</p> <p>2) Hệ thống hiển thị:</p> <ul style="list-style-type: none"> • Bảng dữ liệu văn bản với kết quả phân loại (Neutral, Negative, Positive, Irrelevant) • Tổng số dữ liệu đã phân tích • Biểu đồ tròn thể hiện tỷ lệ phần trăm của từng loại cảm xúc • Biểu đồ cột thể hiện tần suất xuất hiện của từ khóa theo từng phân loại <p>3) Người dùng có thể xem kết quả hiển thị trên màn hình. Use case kết thúc.</p> |
|--|

Ngoại lệ: Không có

Hậu điều kiện: Người dùng xem được toàn bộ kết quả phân loại và thống kê dữ liệu trên Dashboard

- Đặc tả use case Classify

Bảng 4. 3 Đặc tả use case Classify

Mã use case	UC2
Tên use case	Classify
Tóm tắt	Use case cho phép người dùng nhập văn bản và nhận kết quả phân loại cảm xúc từ hệ thống
Actor	User
Tiền điều kiện	Trình duyệt được bật
Đảm bảo tối thiểu	Người dùng mở được trang Classify
Đảm bảo thành công	Người dùng nhận được kết quả phân loại cảm xúc cho văn bản đã nhập
Kích hoạt	Người dùng nhập chọn nút "Classify" trên giao diện

Luồng các sự kiện:

Luồng cơ bản:

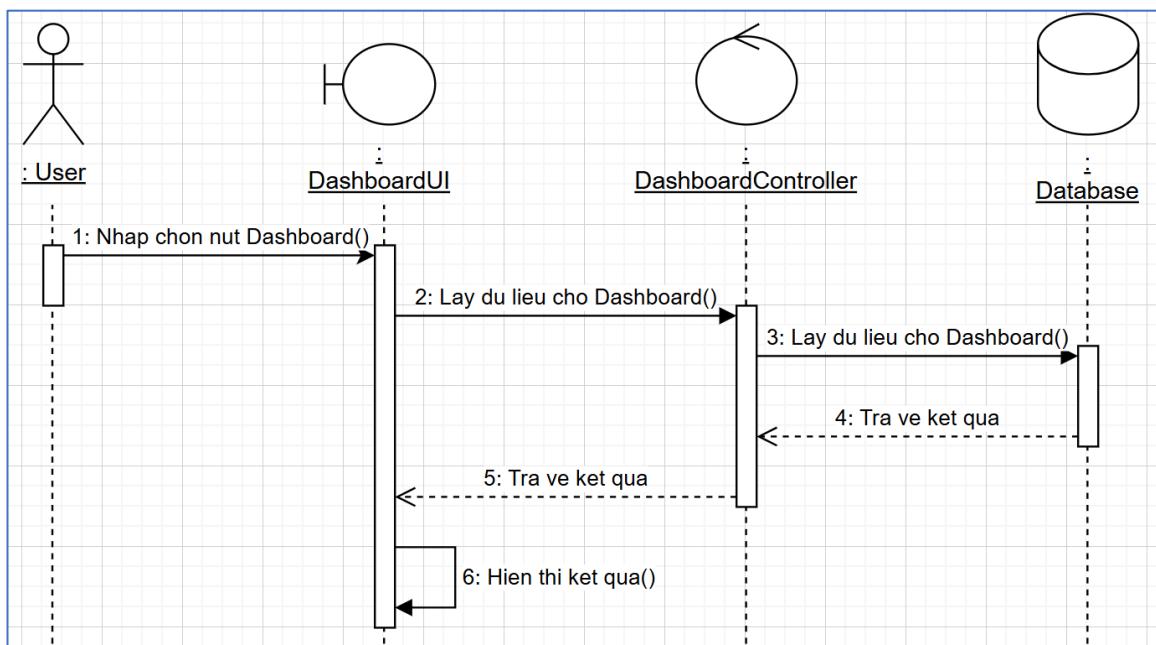
- 1) Người dùng nhập chọn nút "Classify" và hệ thống hiển thị giao diện nhập văn bản.
- 2) Người dùng nhập văn bản cần phân loại vào ô văn bản.
- 3) Người dùng nhấp nút "Classify" để gửi văn bản.
- 4) Hệ thống xử lý và phân tích cảm xúc của văn bản
- 5) Hệ thống hiển thị kết quả phân loại cảm xúc (Negative, Positive, Neutral, hoặc Irrelevant). Use case kết thúc

Ngoại lệ: Người dùng không nhập văn bản: Hệ thống hiển thị thông báo yêu cầu nhập văn bản.

Hậu điều kiện: Không có.

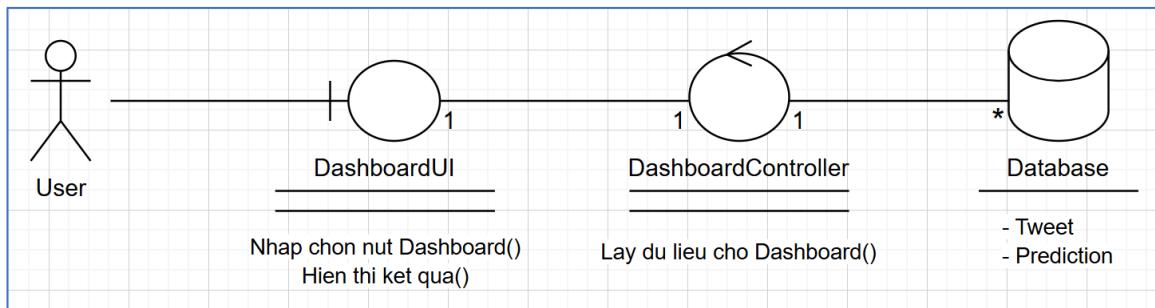
4.5.4. Phân tích các use case

- a) Use case Dashboard
 - Biểu đồ trình tự use case



Hình 4. 10 Biểu đồ trình tự use case Dashboard

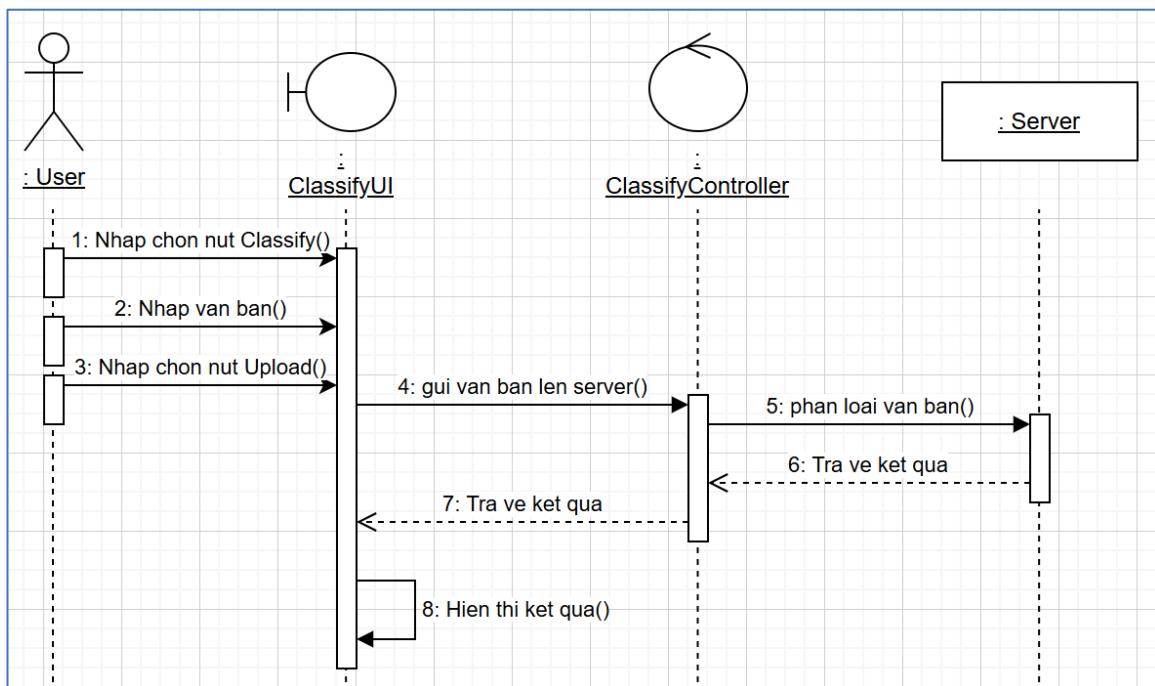
- Biểu đồ phân tích use case



Hình 4. 11 Biểu đồ phân tích use case Dashboard

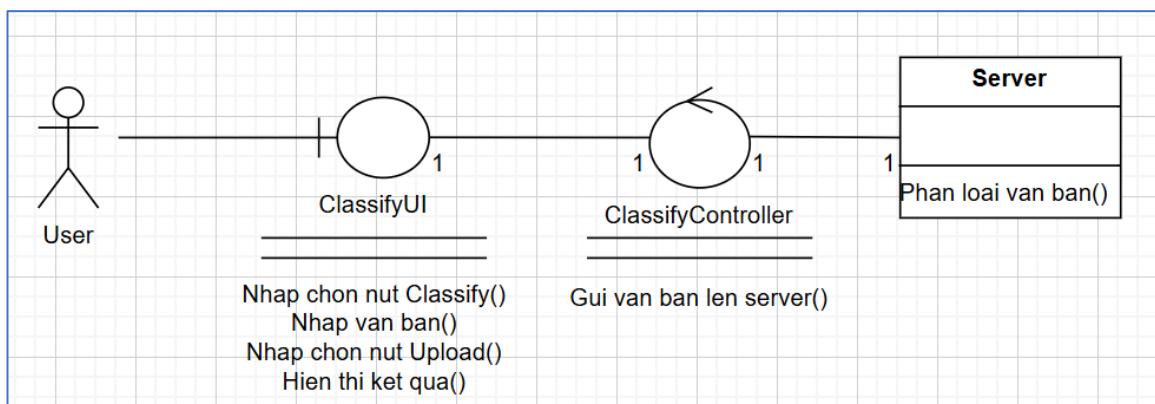
b) Use case Classify

- Biểu đồ trình tự use case



Hình 4. 12 Biểu đồ trình tự use case Classify

- Biểu đồ phân tích use case



Hình 4. 13 Biểu đồ phân tích use case Classify

4.6. Xây dựng phần mềm demo

4.6.1. Cài đặt môi trường

Để triển khai hệ thống phân tích dữ liệu Twitter theo thời gian thực, chúng ta cần cài đặt và cấu hình các thành phần chính bao gồm Apache Kafka, MongoDB và các thư viện Python liên quan.

Apache Kafka được triển khai thông qua Docker để đảm bảo tính đồng nhất và dễ dàng quản lý. File cấu hình “zk-single-kafka-single.yml” được viết để khởi tạo một Zookeeper và một Kafka broker đủ cho môi trường phát triển.

```
version: '2.1'

services:
  zoo1:
    image: confluentinc/cp-zookeeper:7.3.0
    hostname: zoo1
    container_name: zoo1
    ports:
      - "2181:2181"
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_SERVER_ID: 1
      ZOOKEEPER_SERVERS: zoo1:2888:3888

  kafka1:
    image: confluentinc/cp-kafka:7.3.0
    hostname: kafka1
    container_name: kafka1
    ports:
      - "9092:9092"
      - "29092:29092"
      - "9999:9999"
    environment:
      KAFKA_ADVERTISED_LISTENERS: INTERNAL://kafka1:19092,EXTERNAL://${DOCKER_HOST_IP:-127.0.0.1}:9092,DOCKER://host.docker.internal:29092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT,DOCKER:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: INTERNAL
      KAFKA_ZOOKEEPER_CONNECT: "zoo1:2181"
      KAFKA_BROKER_ID: 1
      KAFKA_LOG4J_LOGGERS: "kafka.controller=INFO,kafka.producer.async.DefaultEventHandler=INFO,state.change.logger=INFO"
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_TRANSACTION_STATE_LOG_REPLICATION_FACTOR: 1
      KAFKA_TRANSACTION_STATE_LOG_MIN_ISR: 1
      KAFKA_JMX_PORT: 9999
      KAFKA_JMX_HOSTNAME: ${DOCKER_HOST_IP:-127.0.0.1}
      KAFKA_AUTHORIZER_CLASS_NAME: kafka.security.authorizer.AclAuthorizer
      KAFKA_ALLOW_EVERYONE_IF_NO_ACL_FOUND: "true"
    depends_on:
      - zoo1
```

Hình 4. 14 Cấu hình Zookeeper và Fafka

Chúng ta sử dụng lệnh sau để khởi động: docker-compose -f zk-single-kafka-single.yml up -d

 codedoan		Exited
 zoo1 66016a9ce30c 	confluentinc/cp-zookeeper:7.3.0	Exited (143) 2181:2181
 kafka1 b2c5f83e3f32 	confluentinc/cp-kafka:7.3.0	Exited (143) 29092:29092 Show all ports (3)

Hình 4. 15 Container chứa Zookeeper và Kafka

Sau khi tạo thành công container chứa các dịch vụ trên, chúng ta cần tạo topic "twitter" làm nơi lưu trữ và truyền tải dữ liệu với lệnh:

- kafka-topics --create --topic twitter --bootstrap-server localhost:9092
- kafka-topics --describe --topic twitter --bootstrap-server localhost:9092

```
PS C:\Users\DELL> docker exec -it kafka1 /bin/bash
[appuser@kafka1 ~]$ kafka-topics --create --topic twitter --bootstrap-server localhost:9092
Created topic twitter.
[appuser@kafka1 ~]$ kafka-topics --describe --topic twitter --bootstrap-server localhost:9092
Topic: twitter  TopicId: JfpDMhFKRUqaOdbz8wQRiA PartitionCount: 1      ReplicationFactor: 1
 Configs:
    Topic: twitter  Partition: 0      Leader: 1      Replicas: 1      Isr: 1
[appuser@kafka1 ~]$ |
```

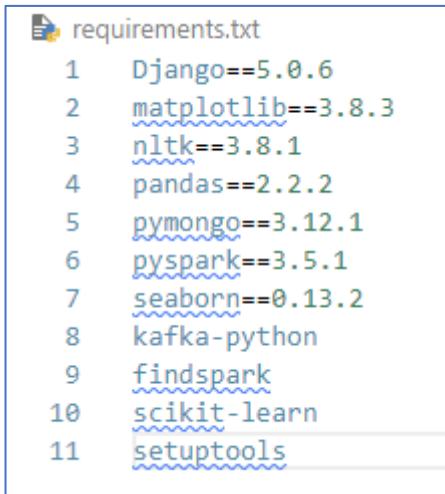
Hình 4. 16 Tạo topic trong Kafka

MongoDB được sử dụng làm cơ sở dữ liệu chính để lưu trữ dữ liệu tweet gốc và kết quả phân loại. Sau khi tải và cài đặt, chúng ta cần tạo database và collection mới để có thể lưu trữ dữ liệu đã được dữ liệu đã được xử lý và phân loại.

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar titled 'Compass' with sections for 'My Queries' and 'CONNECTIONS'. Under 'CONNECTIONS', there's a list with 'localhost:27017' expanded, showing databases 'DoAn', 'admin', and 'bigdata_project'. The 'DoAn' database is selected. In the main area, the 'DoAn' database is selected, and the 'tweets' collection is highlighted. A tooltip for the 'tweets' collection provides the following information: Storage size: 4.10 kB, Documents: 0, and Avg. document size: 0 B.

Hình 4. 17 Database và Collection để lưu dữ liệu

Cuối cùng chúng ta cần cài đặt các thư viện cần thiết như pyspark, pymongo, django, kafka-python và các thư viện xử lý dữ liệu khác nhằm đảm bảo môi trường phát triển đầy đủ để xử lý dữ liệu và vận hành hệ thống.



Hình 4. 18 Các thư viện cần thiết

4.6.2. Triển khai

Sau khi cài đặt xong môi trường, chúng ta bắt đầu triển khai hệ thống phân tích dữ liệu Twitter theo thời gian thực. Việc triển khai được thực hiện theo trình tự các bước để đảm bảo hệ thống hoạt động một cách đồng bộ và hiệu quả.

Đầu tiên, chúng ta cần đọc dữ liệu từ tệp csv, mã hóa dữ liệu và đẩy lên Kafka topic đã tạo. Producer được cài đặt để gửi dữ liệu với tần suất 3 giây một lần, nhằm mô phỏng luồng dữ liệu thời gian thực:

```

from time import sleep
import csv
from kafka import KafkaProducer
import json

producer = KafkaProducer(bootstrap_servers=['localhost:9092'],
                        value_serializer=lambda x:
                        json.dumps(x).encode('utf-8'))

with open('twitter_validation.csv', encoding='utf-8') as file_obj:
    reader_obj = csv.reader(file_obj)
    for data in reader_obj:
        # print(data)
        producer.send('twitter', value=data)
        sleep(3)

```

Hình 4. 19 Code Producer

Sau đây chúng ta cần khởi động Consumer để tiêu thụ dữ liệu từ Producer. Consumer này sẽ sử dụng PySpark để đọc dữ liệu từ Kafka topic "twitter", xử lý dữ liệu và áp dụng mô hình phân loại cảm xúc đã được huấn luyện trước đó, và lưu kết quả phân tích vào MongoDB:

```

consumer = KafkaConsumer(
    'twitter',
    bootstrap_servers=['localhost:9092'],
    auto_offset_reset='earliest',
    enable_auto_commit=True,
    group_id='my-group',
    value_deserializer=lambda x: loads(x.decode('utf-8')))

for message in consumer:
    tweet = message.value[-1] # get the Text from the list
    preprocessed_tweet = clean_text(tweet)
    # Create a DataFrame from the string
    data = [(preprocessed_tweet,)]
    data = spark.createDataFrame(data, ["Text"])
    # Apply the pipeline to the new text
    processed_validation = pipeline.transform(data)
    prediction = processed_validation.collect()[0][6]

    print("-> Tweet:", tweet)
    print("-> preprocessed_tweet : ", preprocessed_tweet)
    print("-> Predicted Sentiment:", prediction)
    print("-> Predicted Sentiment classname:", class_index_mapping[int(prediction)])


    # Prepare document to insert into MongoDB
    tweet_doc = {
        "tweet": tweet,
        "prediction": class_index_mapping[int(prediction)]
    }

    # Insert document into MongoDB collection
    collection.insert_one(tweet_doc)

    print("*"*50)

```

Hình 4. 20 Code Consumer

Cuối cùng, chúng ta khởi động ứng dụng web Django để hiển thị kết quả phân tích cảm xúc. Ứng dụng web này kết nối trực tiếp với MongoDB để truy xuất dữ liệu và hiển thị lên dashboard theo thời gian thực:

```

def dashboard(request):
    data = list(db.tweets.find()) # Fetch all data from MongoDB
    len_data = len(data)
    print("len_data : ", len_data)
    if len_data == 0:
        context = {'len_data': len_data}
        return render(request, 'dashboard/index.html', context)
    sentiment_counts = {label: 0 for label in ['Negative', 'Positive', 'Neutral', 'Irrelevant']}
    for entry in data:
        sentiment_counts[entry['prediction']] += 1

    # Calculate the rate of each class
    sentiment_rates = {label: round((count / len_data) * 100, 2) for label, count in sentiment_counts.items()}

    plot_word_frequencies_per_class(data)

    context = {
        'data': data,
        'len_data': len_data,
        'sentiment_rates': sentiment_rates,
        'sentiment_counts': sentiment_counts,
    }
    return render(request, 'dashboard/index.html', context)

```

Hình 4. 21 Code hàm hiển thị thống kê dữ liệu trong view

```

# Load the model
pipeline = PipelineModel.load("LR_best_model")

def clean_text(text):
    if text is not None:
        # Remove links starting with https://, http://, www., or containing .com
        text = re.sub(r'https?://\S+|www\.\S+\.\com\S+|youtu\.be/\S+', '', text)
        # Remove words starting with # or @
        text = re.sub(r'(@|\#)\w+', '', text)
        # Convert to lowercase
        text = text.lower()
        # Remove non-alphabetic characters
        text = re.sub(r'[^a-zA-Z\s]', '', text)
        # Remove extra whitespaces
        text = re.sub(r'\s+', ' ', text).strip()
    return text
    else:
        return ''

class_index_mapping = { 0: "Negative", 1: "Positive", 2: "Neutral", 3: "Irrelevant" }

def classify_text(text: str) :
    preprocessed_tweet = clean_text(text)
    data = [(preprocessed_tweet,)]
    data = spark.createDataFrame(data, ["Text"])
    # Apply the pipeline to the new text
    processed_validation = pipeline.transform(data)
    prediction = processed_validation.collect()[0][6]

    print("-> Tweet : ", text)
    print("-> preprocessed_tweet : ", preprocessed_tweet)
    print("-> Predicted Sentiment : ", prediction)
    print("-> Predicted Sentiment classname : ", class_index_mapping[int(prediction)])]

    print("/"*50)

    return class_index_mapping[int(prediction)]

```

Hình 4. 22 Code hàm phân loại dữ liệu

```

def classify(request) :
    error = False
    error_text = ""
    prediction = ""
    text = ""

    if request.method == 'POST':
        text = request.POST.get('text')
        print(len(text.strip()))
        if len(text.strip()) > 0:
            error = False
            from .consumer_user import classify_text
            prediction = classify_text(text)

        else:
            error = True
            error_text = "the Text is empty!! PLZ Enter Your Text."

    context = {
        "error" : error,
        "error_text" : error_text,
        "prediction" : prediction,
        "text" : text,
        "text_len" : len(text.strip())
    }

    print(context)
    return render(request, 'dashboard/classify.html', context)

```

Hình 4. 23 Code hàm phân loại dữ liệu trong view

Sau khi hoàn tất các bước triển khai, toàn bộ hệ thống sẽ hoạt động như một pipeline xử lý dữ liệu liên tục: dữ liệu Twitter được đẩy vào hệ thống qua Kafka Producer, xử lý và phân tích bởi Spark Streaming Consumer, kết quả được lưu trữ trong MongoDB và cuối cùng được hiển thị cho người dùng thông qua giao diện web trực quan.

4.6.3. Kết quả

Sau khi triển khai đầy đủ các thành phần của hệ thống, chúng ta đã đạt được kết quả hoạt động như mong đợi với luồng xử lý dữ liệu hoàn chỉnh từ đầu vào đến hiển thị kết quả.

Quá trình bắt đầu với việc Producer đọc dữ liệu từ tệp CSV và truyền tải lên Kafka topic. Dữ liệu được đẩy với tần suất 3 giây một tweet, mô phỏng dòng dữ liệu Twitter theo thời gian thực:

```
[appuser@kafka1 ~]$ kafka-console-consumer --bootstrap-server localhost:9092 --topic twitter --from-beginning
[{"3364", "Facebook", "Irrelevant", "I mentioned on Facebook that I was struggling for motivation to go for a run the other day, which has been translated by Tom's great auntie as 'Hayley can't get out of bed' and told to his grandma, who now thinks I'm a lazy, terrible person \ud83e\udd23"}]
[{"352", "Amazon", "Neutral", "BBC News - Amazon boss Jeff Bezos rejects claims company acted like a 'drug dealer' bbc.co.uk/news/busine\u2026"]
[{"8312", "Microsoft", "Negative", "@Microsoft Why do I pay for WORD when it functions so poorly on my @SamsungUS Chromebook? \ud83d\ude44"]
[{"4371", "CS-GO", "Negative", "CSGO matchmaking is so full of closet hacking, it's a truly awful game."]
[{"4433", "Google", "Neutral", "Now the President is slapping Americans in the face that he really did commit an unlawful act after his acquittal! From Discover on Google vanityfair.com/news/2020/02/t\u2026"]
[{"6273", "FIFA", "Negative", "Hi @EAHelp I've had Madeleine McCann in my cellar for the past 13 years and the little sneaky thing just escaped whilst I was loading up some fifa points, she took my card and I'm having to use my paypal account but it isn't working, can you help me resolve it please?"}]
[{"7925", "MaddenNFL", "Positive", "Thank you @EA Madden NFL!! \n\n New TE Austin Hooper in the ORANGE & BROWN!! \n\n Browns | @AustinHooper18 \n\n pic.twitter.com/GRg4xzFKOn"]
[{"11332", "TomClancysRainbowSix", "Positive", "Rocket League, Sea of Thieves or Rainbow Six: Siege\ud83e\udd14? I love playing all three on stream but which is the best? #stream #twitch #RocketLeague #SeaOfThieves #RainbowSixSiege #follow"]
[{"1107", "AssassinsCreed", "Positive", "my ass still knee-deep in Assassins Creed Odyssey with no way out anytime soon l mao"]
[{"2069", "CallOfDuty", "Negative", "FIX IT JESUS ! Please FIX IT ! What In the world is going on here. @PlayStation @As kPlayStation @Playstationsup @Treyarch @CallofDuty negative 345 silver wolf error code pic.twitter.com/ziRyhrf59Q"]
[{"3185", "Data2", "Positive", "The professional data 2 scene is fucking exploding and I completely welcome it.\n\nGet the garbage out."]
[{"1172", "AssassinsCreed", "Positive", "Itching to assassinate \n\n#TCCGif #AssassinsCreedBlackFlag #AssassinsCreed #The
```

Hình 4. 24 Hình ảnh dữ liệu được thu thập bởi consumer

Consumer nhận dữ liệu từ Kafka và sử dụng Spark để xử lý, làm sạch và áp dụng mô hình Logistic Regression đã được huấn luyện. Quá trình phân loại diễn ra tự động và liên tục:

```
○ (.venv) PS C:\D\DoAn\CodeDoAn\Kafka-PySpark> py ./consumer.py
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
25/05/20 19:05:53 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.JNIBLAS
-> Tweet: I mentioned on Facebook that I was struggling for motivation to go for a run the other day, which has been translated by Tom's great auntie as 'Hayley can't get out of bed' and told to his grandma, who now thinks I'm a lazy, terrible person 🤦
-> preprocessed_tweet : i mentioned on facebook that i was struggling for motivation to go for a run the other day which has been translated by toms great auntie as hayley cant get out of bed and told to his grandma who now thinks im a lazy terrible person
-> Predicted Sentiment: 3.0
-> Predicted Sentiment classname: Irrelevant
///////////
-> Tweet: BBC News - Amazon boss Jeff Bezos rejects claims company acted like a 'drug dealer' bbc.co.uk/news/busine..
-> preprocessed_tweet : bbc news amazon boss jeff bezos rejects claims company acted like a drug dealer bbccouknewsavbusine
-> Predicted Sentiment: 2.0
-> Predicted Sentiment classname: Neutral
///////////
-> Tweet: @Microsoft Why do I pay for WORD when it functions so poorly on my @SamsungUS Chromebook? 😢
-> preprocessed_tweet : why do i pay for word when it functions so poorly on my chromebook
-> Predicted Sentiment: 0.0
-> Predicted Sentiment classname: Negative
///////////
-> Tweet: CSGO matchmaking is so full of closet hacking, it's a truly awful game.
-> preprocessed_tweet : csgo matchmaking is so full of closet hacking its a truly awful game
-> Predicted Sentiment: 0.0
-> Predicted Sentiment classname: Negative
///////////
-> Tweet: Now the President is slapping Americans in the face that he really did commit an unlawful act after his acquittal! From Discover on Google vanityfair.com/news/2020/02/t...
-> preprocessed_tweet : now the president is slapping americans in the face that he really did commit an unlawful act after his acquittal from discover on google vanityfair
-> Predicted Sentiment: 2.0
-> Predicted Sentiment classname: Neutral
///////////
-> Tweet: Hi @EAHelp I've had Madeleine McCann in my cellar for the past 13 years and the little sneaky thing just escaped whilst I was loading up some fifa points, she took my card and I'm having to use my paypal account but it isn't working, can you help me resolve it please?
-> preprocessed_tweet : hi ive had madeleine mccann in my cellar for the past years and the little sneaky thing just escaped whilst i was loading up some fifa points she took my card and im having to use my paypal account but it isnt working can you help me resolve it please
-> Predicted Sentiment: 0.0
-> Predicted Sentiment classname: Negative
/////////
```

Hình 4. 25 Hình ảnh dữ liệu được xử lý và phân loại từ topic

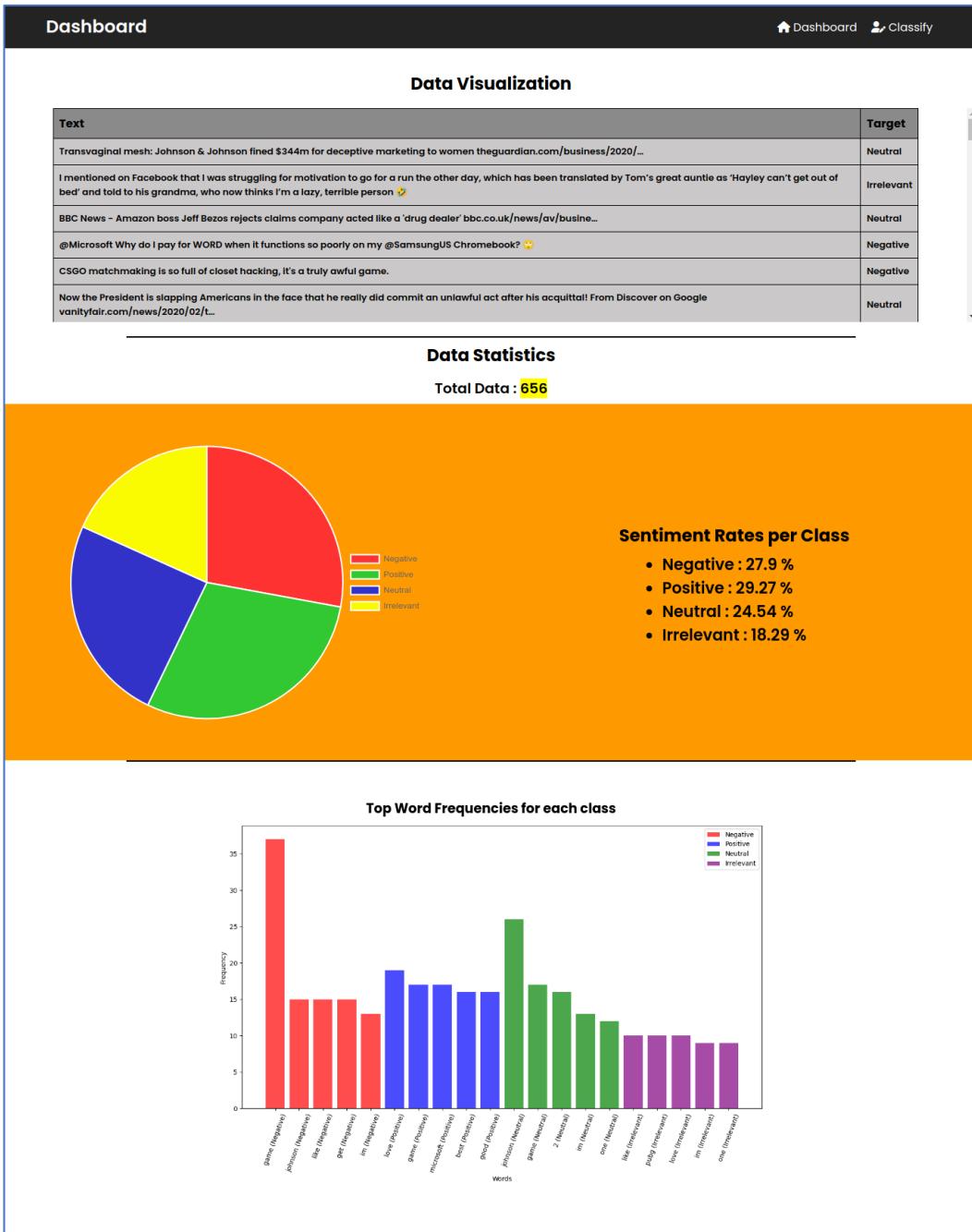
Sau khi xử lý, tweet cùng với kết quả phân loại cảm xúc được lưu trữ trong MongoDB, tạo cơ sở dữ liệu cho việc truy xuất và hiển thị:

<code>_id: ObjectId('682c70298755ea1f7ff7145a')</code>	<code>tweet: "I mentioned on Facebook that I was struggling for motivation to go for..."</code>	<code>prediction: "Irrelevant"</code>
<code>_id: ObjectId('682c70358755ea1f7ff7145b')</code>	<code>tweet: "BBC News - Amazon boss Jeff Bezos rejects claims company acted like a..."</code>	<code>prediction: "Neutral"</code>
<code>_id: ObjectId('682c70418755ea1f7ff7145c')</code>	<code>tweet: "@Microsoft Why do I pay for WORD when it functions so poorly on my @Sa..."</code>	<code>prediction: "Negative"</code>
<code>_id: ObjectId('682c704c8755ea1f7ff7145d')</code>	<code>tweet: "CSGO matchmaking is so full of closet hacking, it's a truly awful game..."</code>	<code>prediction: "Negative"</code>
<code>_id: ObjectId('682c70598755ea1f7ff7145e')</code>	<code>tweet: "Now the President is slapping Americans in the face that he really did..."</code>	<code>prediction: "Neutral"</code>
<code>_id: ObjectId('682c70668755ea1f7ff7145f')</code>	<code>tweet: "Hi @EAHelp I've had Madeleine McCann in my cellar for the past 13 year..."</code>	<code>prediction: "Negative"</code>

Hình 4. 26 Hình ảnh kết quả trong cơ sở dữ liệu

Như có thể thấy trong hình, mỗi bản ghi trong MongoDB bao gồm nội dung tweet, nhãn dự đoán, tạo điều kiện thuận lợi cho việc truy vấn và phân tích sau này.

Cuối cùng là giao diện Dashboard được phát triển bằng Django hiển thị dữ liệu đã được phân tích dưới dạng trực quan, cung cấp cái nhìn tổng quát về phân phối cảm xúc trong tập dữ liệu:

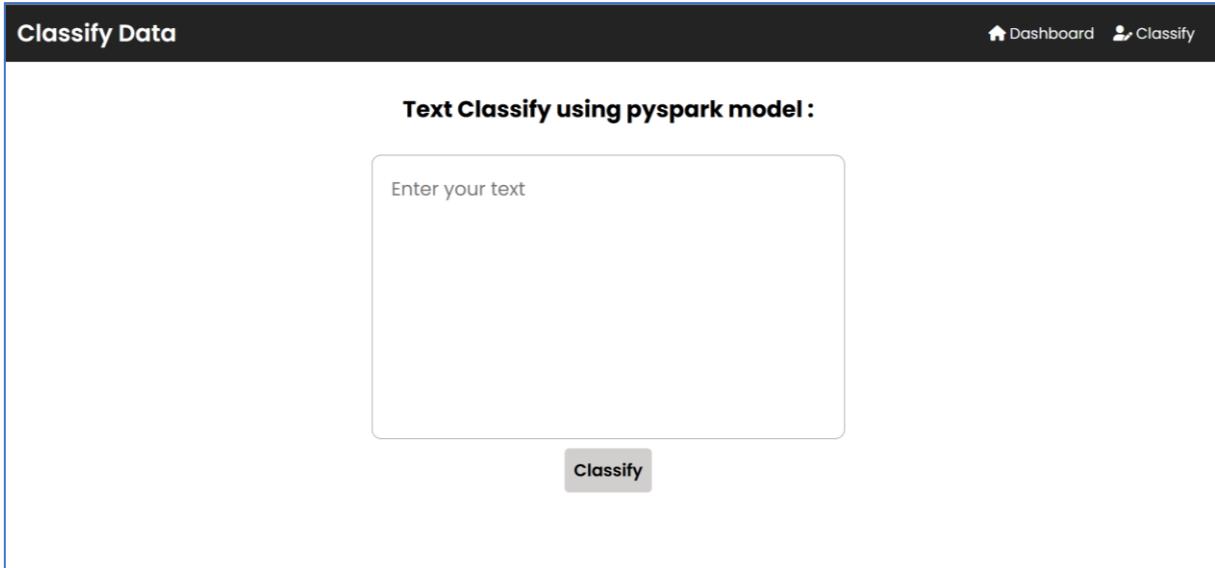


Hình 4. 27 Giao diện Dashboard

Dashboard bao gồm các thành phần chính:

- Bảng hiển thị nội dung tweet và nhãn cảm xúc tương ứng
- Biểu đồ tròn thể hiện tỷ lệ phần trăm của từng loại cảm xúc (Positive, Negative, Neutral, Irrelevant)
- Tổng số lượng tweet đã được phân tích
- Biểu đồ cột hiển thị tần suất xuất hiện của từ khóa phổ biến theo từng loại cảm xúc

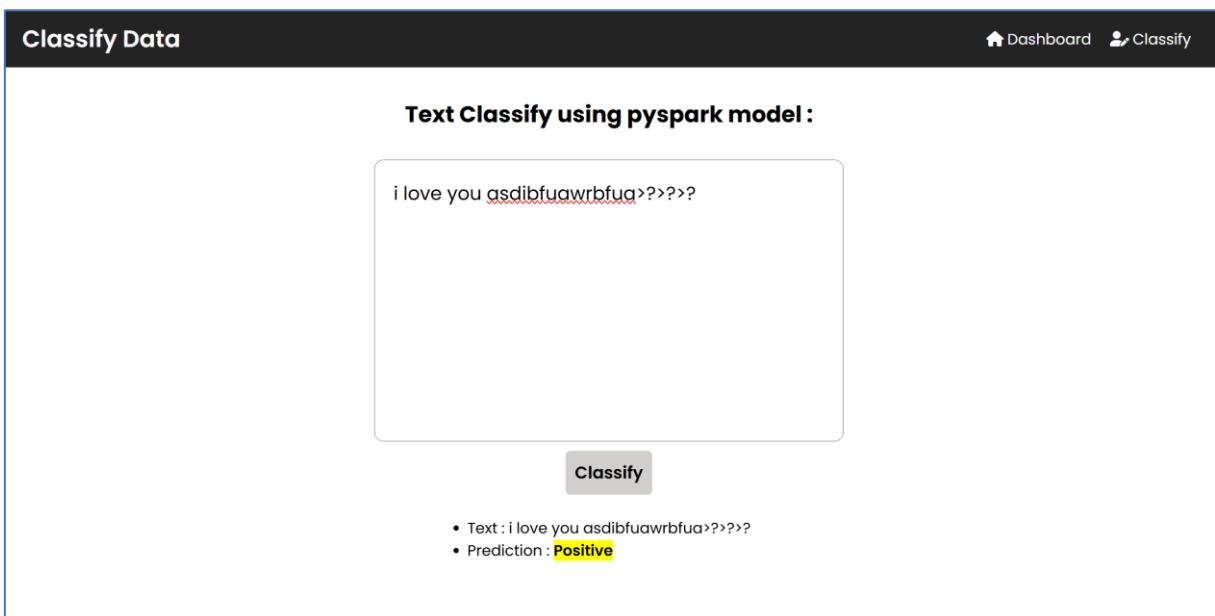
Ngoài Dashboard, hệ thống còn cung cấp chức năng phân loại cảm xúc theo thời gian thực cho văn bản do người dùng nhập vào:



Hình 4. 28 Giao diện phân loại dữ liệu

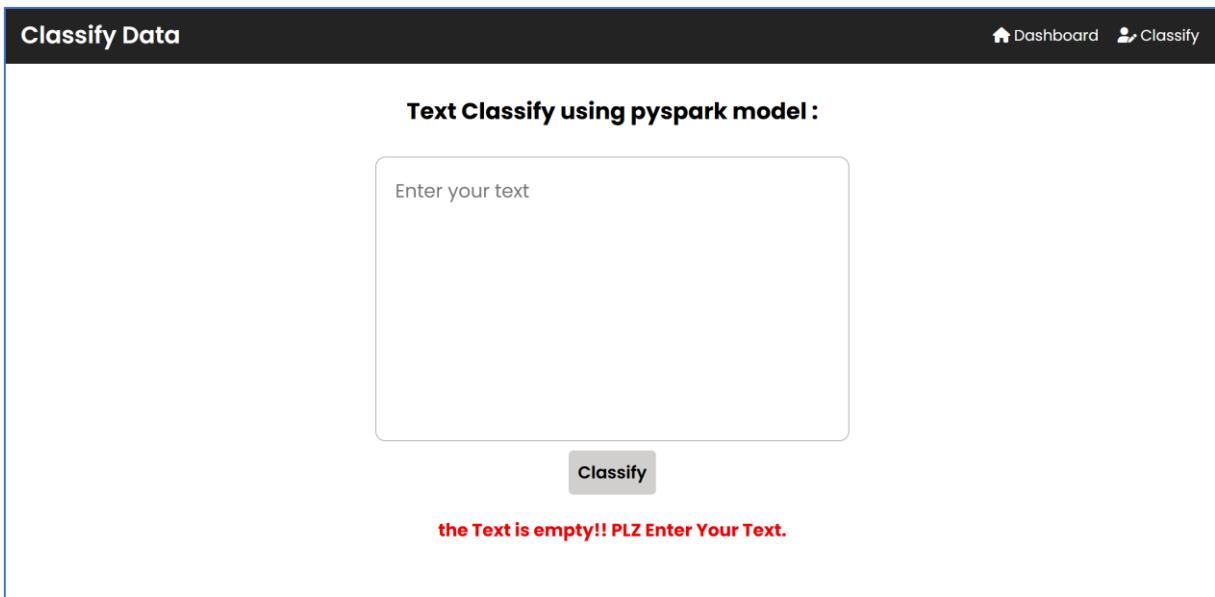
Khi người dùng nhập văn bản và nhấn nút "Classify", hệ thống sẽ xử lý và phân loại cảm xúc của văn bản đó ngay lập tức:

- Khi có dữ liệu nhập vào:



Hình 4. 29 Giao diện phân loại dữ liệu khi nhập văn bản

- Khi không có dữ liệu nhập vào:



Hình 4. 30 Giao diện phân loại dữ liệu khi không nhập văn bản

Hệ thống sẽ hiển thị thông báo yêu cầu người dùng nhập văn bản trước khi tiến hành phân loại.

Qua các kết quả trên, có thể thấy hệ thống đã được triển khai thành công và hoạt động như mong đợi, từ việc thu thập dữ liệu, xử lý phân tích, đến hiển thị kết quả theo thời gian thực trên giao diện người dùng. Hệ thống không chỉ có khả năng phân tích dữ liệu được cung cấp từ tập dữ liệu sẵn có mà còn có thể mở rộng để xử lý văn bản mới do người dùng nhập vào, thể hiện tính linh hoạt và khả năng ứng dụng thực tế cao.

KẾT LUẬN

Trong suốt quá trình thực hiện đề tài "Phân loại cảm xúc người dùng mạng X theo thời gian thực bằng Apache Spark MLlib", tôi đã được tiếp cận và áp dụng các công nghệ hiện đại trong lĩnh vực học máy và xử lý ngôn ngữ tự nhiên để giải quyết bài toán phân tích cảm xúc người dùng mạng xã hội theo thời gian thực.

Đề tài là cơ hội quý báu để tôi vận dụng kiến thức về xử lý ngôn ngữ tự nhiên, học máy và công nghệ dữ liệu lớn vào xây dựng một giải pháp toàn diện - từ thu thập dữ liệu, xử lý văn bản, đến huấn luyện mô hình và triển khai hệ thống. Qua đó, tôi đã phát triển tư duy hệ thống, kỹ năng làm việc độc lập và khả năng quản lý thời gian hiệu quả.

Tôi đã nghiên cứu và triển khai các mô hình học máy với Apache Spark MLlib, từ Logistic Regression, Random Forest đến Naïve Bayes. Việc ứng dụng các mô hình này vào phân loại cảm xúc đặt ra nhiều thách thức về xử lý đặc trưng văn bản, tinh chỉnh tham số và đánh giá hiệu suất mô hình trong môi trường thời gian thực. Mỗi khó khăn đều là cơ hội để tôi đào sâu hiểu biết về học máy và xử lý ngôn ngữ tự nhiên, đặc biệt đối với dữ liệu mạng xã hội.

Hệ thống được phát triển bằng cách kết hợp Apache Spark Streaming để xử lý dữ liệu theo luồng, Kafka để truyền tải thông điệp, MongoDB để lưu trữ và các công nghệ hỗ trợ như Docker, Django cho giao diện người dùng. Thiết kế này giúp hệ thống dễ dàng mở rộng và nâng cấp các thành phần riêng lẻ, đáp ứng nhu cầu phân tích cảm xúc người dùng theo thời gian thực phục vụ cho các chiến lược kinh doanh và truyền thông.

Mặc dù đã đạt được kết quả khả quan trên các dữ liệu cơ bản, hệ thống vẫn còn những hạn chế cần cải thiện như giao diện, xử lý dữ liệu phức tạp và hỗ trợ đa ngôn ngữ. Trong tương lai, tôi định hướng tích hợp các kỹ thuật học sâu tiên tiến như BERT, Transformer, mở rộng hỗ trợ đa ngôn ngữ và phát triển giao diện người dùng trực quan hơn.

TÀI LIỆU THAM KHẢO

- [1] Website tham khảo về ứng dụng của xử lý ngôn ngữ tự nhiên. URL: <https://fpt.ai/vi/bai-viet/ung-dung-cua-xu-ly-ngon-nhu-tu-nhi-trong-tieng-viet/> Lần truy cập gần nhất: 25/3/2025
- [2] Website tham khảo về phân tích cảm xúc và ứng dụng của nó. URL: <https://fpt.ai/vi/bai-viet/sentiment-analysis/> Lần truy cập gần nhất: 25/3/2025
- [3] Website tham khảo về mô hình hồi quy logistic tuyến tính (logistic regression). URL: <https://machinelearningcoban.com/2017/01/27/logisticregression/#mo-hinh-logistic-regression> Lần truy cập gần nhất: 29/3/2025
- [4] Website tham khảo về mô hình SVM (Support Vector Machine). URL: http://en.wikipedia.org/wiki/Support_vector_machine Lần truy cập gần nhất: 29/3/2025
- [5] Website tham khảo về mô hình học sâu CNN. URL: <https://vietnix.vn/cnn-lagi/> Lần truy cập gần nhất: 29/3/2025
- [6] Website tham khảo về mô hình học sâu LSTM (Long short-term memory). URL: <https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/> Lần truy cập gần nhất: 29/3/2025
- [7] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1–135.
- [8] Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 1(12).
- [9] Dos Santos, C., & Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In Proceedings of COLING, pp. 69–78.
- [10] Wang, X., Liu, Y., Sun, C., Wang, B., & Wang, X. (2015). Predicting polarities of tweets by composing word embeddings with long short-term memory. In Proceedings of ACL, pp. 1343–1353.
- [11] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.
- [12] Rosenthal, S., Farra, N., & Nakov, P. (2017). SemEval-2017 task 4: Sentiment analysis in Twitter. In Proceedings of SemEval, pp. 502–518.
- [13] Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). Learning sentiment-specific word embedding for Twitter sentiment classification. In Proceedings of ACL, pp. 1555–1565.
- [14] Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2016). Real-time Twitter sentiment analysis with Spark Streaming. In Proceedings of IEEE Big Data, pp. 1–6.

- [15] Gokulakrishnan, B., Priyanthan, P., Ragavan, T., Prasath, N., & Perera, A. (2018). A distributed framework for real-time Twitter sentiment analysis using Spark. In Proceedings of the International Conference on Advanced Computing, pp. 29–34.
- [16] Rodriguez, P., Ortigosa, A., & Carro, R. (2020). Scalable sentiment analysis for social media data with Spark MLlib. In IEEE Big Data, pp. 1272–1281.
- [17] Singh, J., Singh, G., & Singh, R. (2022). Dynamic sentiment analysis of streaming social media data with Spark MLlib. In Proceedings of the 2022 International Conference on Machine Learning and Data Engineering, pp. 1–6.
- [18] Website tham khảo về Spark Mllib cho việc huấn luyện mô hình. URL: <https://www.qubole.com/developers/spark-getting-started-guide/workflow>
Lần truy cập gần nhất: 19/4/2025
- [19] Website tham khảo về Docker. URL: <https://viblo.asia/p/tim-hieu-co-ban-ve-docker-cong-nghe-dang-de-developer-su-dung-yMnKM0bA57P> Lần truy cập gần nhất: 19/4/2025
- [20] Website tham khảo về Kafka. URL: <https://viblo.asia/p/kafka-la-gi-tai-sao-no-lai-pho-bien-den-vay-PwlVmzME45Z> Lần truy cập gần nhất: 19/4/2025
- [21] Website tham khảo về Kafka. URL: <https://topdev.vn/blog/kafka-la-gi/> Lần truy cập gần nhất: 19/4/2025
- [22] Website tham khảo Spark Streamin. URL: <https://www.educba.com/spark-streaming/> Lần truy cập gần nhất: 19/4/2025
- [23] Website tham khảo về MongoDB. URL: <https://www.opc-router.com/what-is-mongodb/> Lần truy cập gần nhất: 19/4/2025
- [24] Website tham khảo về MongoDB <https://www.geeksforgeeks.org/what-is-mongodb-working-and-features/> Lần truy cập gần nhất: 19/4/2025
- [25] Website tham khảo về Django. URL: <https://lanit.com.vn/django-la-gi.html>
Lần truy cập gần nhất: 19/4/2025
- [26] Website tham khảo về Django. URL: <https://vietnix.vn/django-la-gi/> Lần truy cập gần nhất: 19/4/2025
- [27] Bộ dữ liệu về các tweet trong Twitter được sử dụng để tài. URL: <https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis>
Lần truy cập gần nhất: 19/4/2025