ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

=====000=====

MÔN: KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN
ĐỀ TÀI: PHÂN TÍCH CẢM XÚC CỦA CÂU SỬ DỤNG HADOOP VÀ SPARK

Lớp : INT3229 37

Giảng viên hướng dẫn : TS. Trần Hồng Việt

CN. Đỗ Thu Uyên

Nhóm : 8

Thành viên : Nguyễn Ngô Việt Trung - 22022598

Nguyễn Tiến Trung - 22022541

Nguyễn Phương Trang - 22022656

Nguyễn Đức Tước - 22022608

MỞ ĐẦU

Cùng với sự phát triển mạnh mẽ của công nghệ Big Data, khả năng khai thác và phân tích lượng dữ liệu khổng lồ đã mở ra nhiều hướng nghiên cứu và ứng dụng quan trọng. Một trong những lĩnh vực thu hút sự chú ý lớn là phân tích cảm xúc (Sentiment Analysis), giúp trích xuất thông tin từ dữ liệu văn bản để nhận diện, phân loại các ý kiến tích cực, tiêu cực hoặc trung lập trong nhiều ngữ cảnh khác nhau, từ truyền thông xã hội đến đánh giá sản phẩm và phân tích tâm lý khách hàng.

Trong bối cảnh đó, việc áp dụng các framework mạnh mẽ như Hadoop và Spark để xử lý và phân tích dữ liệu văn bản đang trở thành xu hướng tất yếu. Hadoop với khả năng lưu trữ phân tán và xử lý MapReduce, cùng với Spark nổi bật bởi tốc độ xử lý nhanh chóng và khả năng làm việc với dữ liệu trực tuyến, là những công cụ cốt lõi giúp hiện thực hóa các giải pháp phân tích cảm xúc trên dữ liệu lớn.

Nhận thấy tiềm năng ứng dụng to lớn của Sentiment Analysis trong nhiều lĩnh vực như kinh doanh, y tế, giáo dục, nhóm chúng em đã lựa chọn đề tài: "Text-Sentiment-Analysis in Hadoop and Spark" để thực hiện báo cáo kết thúc môn học. Nội dung báo cáo gồm 5 chương:

Chương 1: Tổng quan về dữ liệu lớn.

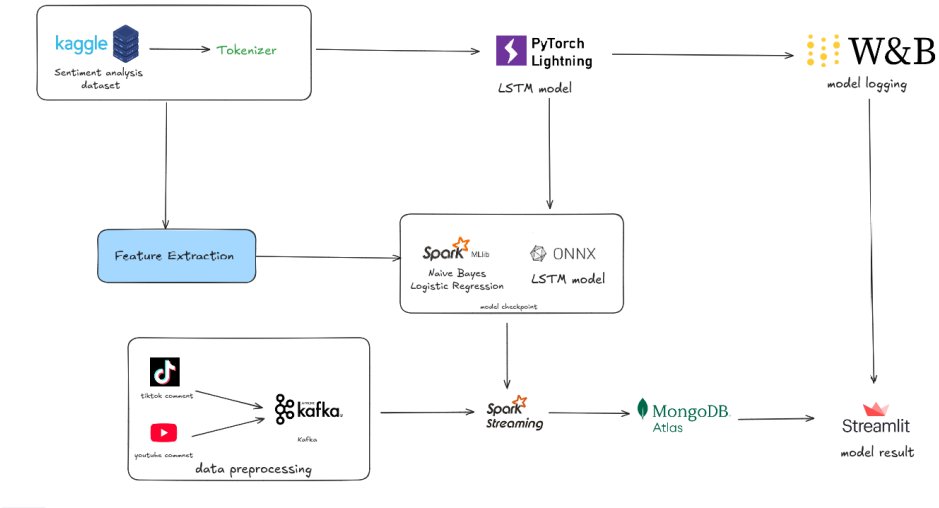
Chương 2: Phân tích cảm xúc của câu sử dụng Hadoop.

Chương 3: Phân tích cảm xúc của câu sử dụng Spark.

Chương 4: Đề xuất mô hình LSTM để phân loại cảm xúc.

Chương 5: Kết luận và hướng phát triển.

Sau đây là tổng quan về chương trình cài đặt của nhóm chúng em:



Hình 1: Pipeline

Với cấu trúc này, chúng em mong muốn mang lại một cái nhìn tổng quan và thực tế về cách áp dụng Big Data vào phân tích cảm xúc, đồng thời đưa ra các giải pháp kỹ thuật hiệu quả cho bài toán xử lý dữ liệu văn bản.

Mục lục

CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN

1.1 Định nghĩa

Theo wikipedia: Dữ liệu lớn là một thuật ngữ chỉ bộ dữ liệu lớn hoặc phức tạp mà các phương pháp truyền thống không đủ các ứng dụng để xử lý dữ liệu này.

Theo Gartner: Dữ liệu lớn là những nguồn thông tin có đặc điểm chung khối lượng lớn, tốc độ nhanh và dữ liệu định dạng dưới nhiều hình thức khác nhau,do đó muốn khai thác được phải đòi hỏi phải có hình thức mới để đưa ra quyết định khám phá và tối ưu hóa quy trình. Dữ liệu đến từ rất nhiều nguồn khác nhau:

Hình 2: Minh họa nguồn gốc của dữ liệu

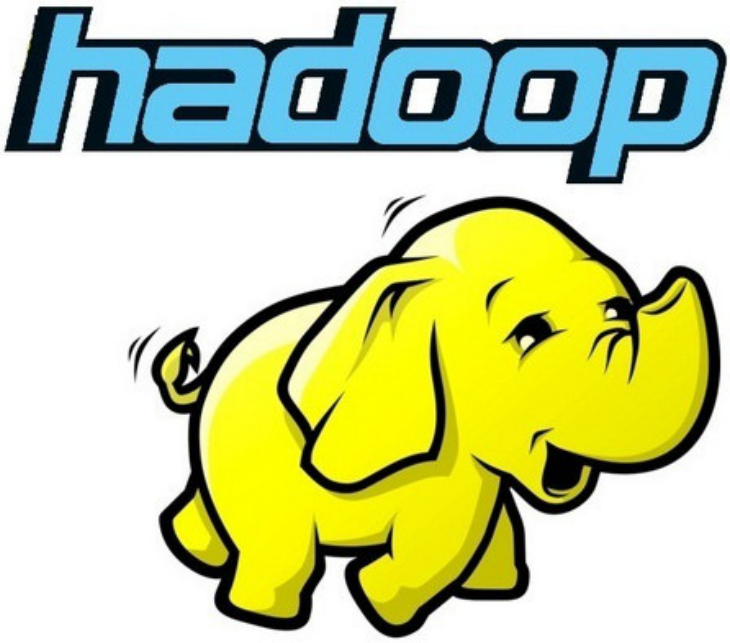
Một số lợi ích có thể mang lại như: Cắt giảm chi phí, tiết kiệm thời gian và giúp tối ưu hóa sản phẩm, hỗ trợ con người đưa ra những quyết định dúng và hợp lý hơn.

1.2 Đặc trưng cơ bản của dữ liệu lớn

1. **Khối lượng lớn (Volume)**: Khối lượng dữ liệu rất lớn và đang ngày càng tăng lên, tính đến 2014 thì có thể trong khoảng vài trăm terabyte.
2. **Tốc độ (Velocity)**: Khối lượng dữ liệu gia tăng rất nhanh.
3. **Đa dạng (Variety)**: Ngày nay hơn 80% dữ liệu được sinh ra là phi cấu trúc(tài liệu, blog, hình ảnh,...)
4. **Độ tin cậy/chính xác(Veracity)**: Bài toán phân tích và loại bỏ dữ liệu thiếu chính xác và nhiễu đang là tính chất quan trọng của bigdata.
5. **Giá trị(Value)**: Giá trị thông tin mang lại.

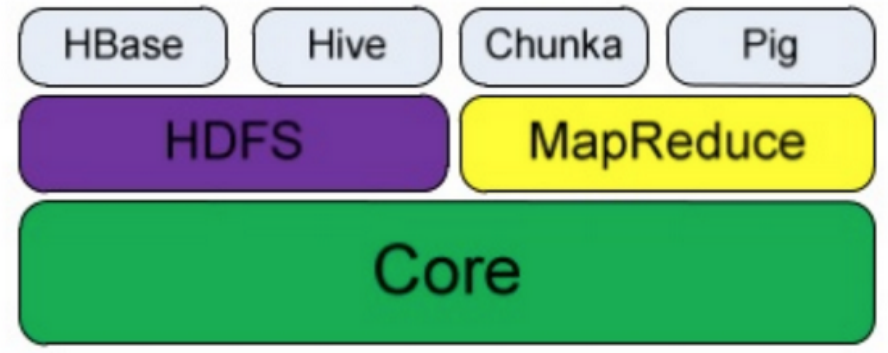
1.3 Tổng quan về Hadoop

Theo apache hadoop: Apache Hadoop là một framework dùng để chạy những ứng dụng trên 1 cluster lớn được xây dựng trên những phần cứng thông thường.



Hình 3: Biểu tượng của Hadoop

Các thành phần của Hadoop: Core, MapReduce engine, HDFS. HBase, Hive, Pig, Chukwa, ... Tuy nhiên tập chung vào 2 thành phần quan trọng nhất: HDFS và MapReduce.



Hình 4: Thành phần của Hadoop

Hadoop thực hiện mô hình Map/Reduce, đây là mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau, và các phần này sẽ được chạy song song trên nhiều node khác nhau.

Hadoop cung cấp một hệ thống file phân tán (HDFS) cho phép lưu trữ dữ liệu lên trên nhiều node. Cả Map/Reduce và HDFS đều được thiết kế sao cho framework sẽ tự động quản lý được

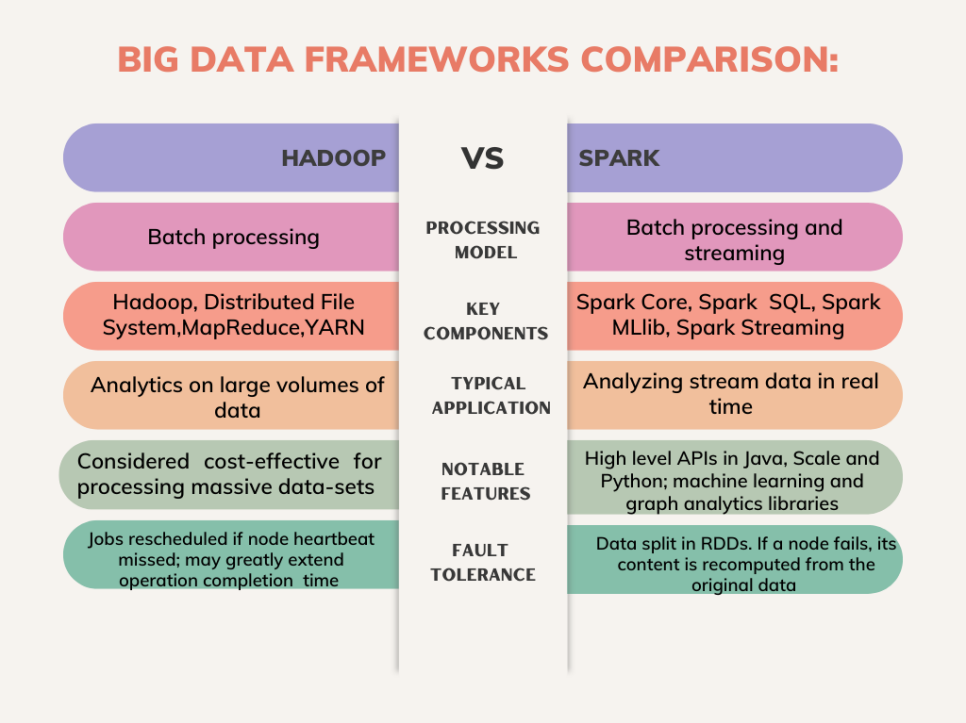
các lỗi, các hư hỏng về phần cứng của các node.

→ **Kết luận:** Hadoop là một framework cho phép phát triển các ứng dụng phân tán được viết bằng java.

1.4 Tổng quan về PySpark

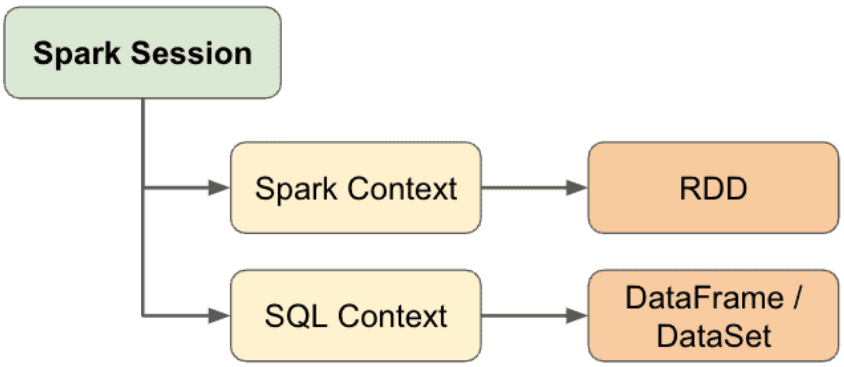
Spark là một framework xử lý phân tán và tính toán cho dữ liệu lớn được phát triển trên nền tảng Hadoop. Spark được thiết kế để hoạt động cùng với Hadoop và sử dụng Hadoop Distributed File System (HDFS) để lưu trữ dữ liệu. Nó cũng tích hợp với Hadoop Yarn để quản lý tài nguyên và phân phối các tác vụ tính toán trên nodes trong cụm Hadoop.

Spark được thiết kế để giải quyết các hạn chế của MapReduce và cung cấp một cách hiệu quả hơn để giải quyết các tác vụ phức tạp.



Hình 5: Spark so với MapReduce

PySpark là một giao diện cho Apache Spark bằng Python. Với PySpark ta có thể viết các lệnh dạng "lai ghép" giữa Python và SQL để truy vấn và phân tích dữ liệu trong môi trường xử lý phân tán. Khi sử dụng PySpark ta có thể làm việc với RDDs (Resilient Distributed Datasets) nhờ vào thư viện Py4j.



Hình 6: Spark Session và Spark Context

PySpark sử dụng điểm lối vào (entry point) thông qua các phương thức Spark Context, Spark Session và chúng có những đặc điểm sau:

Spark Context	**Spark Session**
Là điểm đầu vào chính cho lập trình Spark với RDDs, nó cho phép ta tạo RDDs, accumulators, broadcast variables cũng như là truy cập các dịch vụ Spark và thực hiện các công việc.	Là giao diện hợp nhất kết hợp nhiều chức năng khác nhau của Spark vào một điểm đầu vào duy nhất.
Spark Context cũng cho phép truy cập vào SQL Context và Hive Context để xử lý dữ liệu bán cấu trúc và có cấu trúc.	Spark Session tích hợp cả Spark Context và cung cấp một API cấp cao hơn để làm việc với dữ liệu có cấu trúc thông qua Spark SQL, dữ liệu dòng thông qua Spark Streaming và học máy với MLlib.

1.5 Tổng quan về MapReduce

Định nghĩa: Theo Google, MapReduce là mô hình dùng cho xử lý tính toán song song và phân tán trên hệ thống phân tán

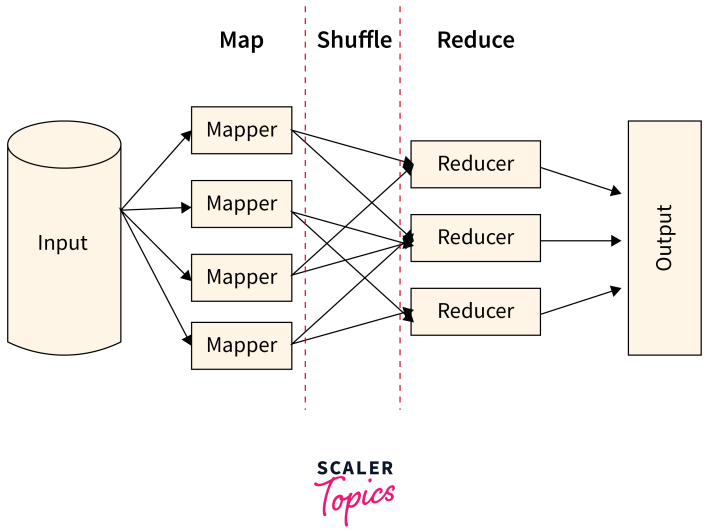
- Bước 1: Phân rã từ nghiệp vụ chính (do người dùng muốn thể hiện) thành các công việc con để chia từng công việc con này về các máy tính trong hệ thống thực hiện xử lý một cách song song
- Bước 2: Thu thập lại các kết quả

Ứng dụng của MapReduce:

- Dữ liệu cần xử lý kích thước lớn
- Các ứng dụng thực hiện xử lý, phân tích dữ liệu, thời gian xử lý đáng kể có thể tính bằng phút, giờ, ...

Thực thi mô hình Mapreduce:

- Hàm Map: Tiếp nhận mảnh dữ liệu input, trích rút thông tin cần thiết từng các phần tử (ví dụ: lọc dữ liệu, trích dữ liệu) tạo kết quả trung gian.
- Hàm Reduce: Tổng hợp kết quả trung gian, tính toán để cho kết quả cuối cùng.



Hình 7: Thực thi mô hình MapReduce

CHƯƠNG 2: PHÂN TÍCH CẢM XÚC CỦA CÂU SỬ DỤNG HADOOP

2.1 Bài toán phân tích cảm xúc

- Phân tích cảm xúc (Sentiment Analysis) là một bài toán trong xử lý ngôn ngữ tự nhiên (NLP) nhằm phân loại các đoạn văn bản thành các cảm xúc tích cực, tiêu cực, hoặc trung tính. Ứng dụng phổ biến của bài toán này bao gồm:
  + Đánh giá sản phẩm trên các trang thương mại điện tử
  + Phân tích phản hồi khách hàng trên mạng xã hội
  + Phân tích xu hướng dư luận cho các sự kiện xã hội

2.2 Ý tưởng chính

2.2.1 Thuật toán phân lớp Naive Bayes trong phân tích cảm xúc

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

$$P(class|data) = \frac{P(data|class)P(class)}{P(data)}$$

Trong đó:

- Dữ liệu là các từ đưa vào mô hình độc lập với nhau. Tức sự thay đổi giá trị của một đặc trưng không làm ảnh hưởng đến đặc trưng còn lại.
- Các đặc trưng đưa vào mô hình có ảnh hưởng ngang nhau đối với đầu ra của mục tiêu.

Khi đó, đẳng thức Bayes trở thành:

$$P(\text{class}|tweet) = P(\text{class}) \prod_{\text{word} \in \text{tweet}} P(\text{word}|\text{class})$$

Naive Bayes thường được sử dụng trong bài toán phân loại văn bản vì:

- Đơn giản, nhanh chóng
- Hiệu quả ngay cả khi ít dữ liệu

2.2.2 Áp dụng MapReduce vào Native Bayes trong phân tích cảm xúc

MapReduce được áp dụng vào Naive Bayes để xử lý dữ liệu song song, gồm hai giai đoạn:

1. Giai đoạn huấn luyện (training phase):

- Đếm số lần xuất hiện của mỗi từ trong các tweet tích cực và tiêu cực

- Tính xác suất $P(word|class)$

2. Giai đoạn kiểm tra (testing phase):

- Tính xác suất $P(class|tweet)$ cho từng tweet kiểm tra

- Dự đoán cảm xúc dựa trên xác suất lớn nhất

2.3 Triển khai thuật toán MapReduce và Naive Bayes cho bài toán phân loại cảm xúc

2.3.1 Huấn luyện

1. Mapper `(Map_Training)`

- Input: Dữ liệu huấn luyện có định dạng

  `[<tweet_id, sentiment_label, timestamp, tweet_text>]`

- Hoạt động:

  (a) Làm sạch văn bản (loại bỏ stopword, chuyển về chữ thường)

  (b) Trả về cặp `<word, sentiment>`

- Output:

  `<great, POSITIVE>`

  `<bad, NEGATIVE>`

2. Reducer `(Reduce_Training)`

- Input: Các cặp `<word, sentiment>` từ Mapper.

- Hoạt động:

  (a) Tổng hợp số lần xuất hiện của mỗi từ theo từng cảm xúc

  (b) Trả về cặp `<word, pos_count@neg_count>`

- Output:

  `<great, 5@1>`

  `<bad, 1@4>`

2.3.2 Kiểm tra

1. Mapper `(Map_Testing)`

- Input: Dữ liệu kiểm tra và mô hình huấn luyện $P(word|class)$

- Hoạt động:

  (a) Làm sạch văn bản tương tự Mapper trong giai đoạn huấn luyện

  (b) Tính xác suất $P(class|tweet)$ dựa trên:

$$P(word|class)P(class|tweet) = P(class) \prod_{word \in tweet} P(word|class)$$

  (c) Dự đoán cảm xúc dựa trên xác suất lớn nhất

- Output:

  `<tweet_id@tweet_text, POSITIVE>`

2.4 Cải tiến thuật toán Naive Bayes với TF-IDF

2.4.1 Giới thiệu về TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) đánh giá tầm quan trọng của một từ trong tài liệu.

- Term Frequency (TF): Tần suất xuất hiện của từ trong tài liệu, tính theo công thức

$$TF(t, d) = \frac{\text{Số lần từ } t \text{ xuất hiện trong tài liệu } d}{\text{Tổng số từ trong tài liệu } d}$$

- Inverse Document Frequency (IDF): Đo lường mức độ phổ biến của từ trên toàn bộ corpus

$$IDF(t, D) = \log\left(\frac{N}{1 + |\{d \in D : t \in d\}|}\right)$$

  Trong đó:

  – N: Tổng số tài liệu trong corpus

  – $|\{d \in D : t \in d\}|$: Số tài liệu chứa từ t

Kết hợp TF và IDF giúp giảm trọng số từ phổ biến và tăng trọng số từ đặc trưng.

2.4.2 Tích hợp TF-IDF trong bài toán phân tích cảm xúc

Bước 1: Tính TF cho mỗi từ trong từng tweet

- Mapper: Tính $TF(t, d)$ bằng cách đếm số lần xuất hiện của từ t và chia cho tổng số từ trong tweet d

- Reducer: Tổng hợp kết quả $TF(t, d)$ cho mỗi từ

Bước 2: Tính IDF cho mỗi từ trên toàn bộ tập tweet

- Mapper: Đếm số lượng tweet mà từ t xuất hiện

- Reducer: Tính toán $IDF(t, D)$ sử dụng tổng số tweet và số lượng tweet chứa từ t

Bước 3: Tính TF-IDF

- Mapper: Nhân giá trị $TF(t, d)$ với $IDF(t, D)$ để có trọng số $TFIDF(t, d)$ cho từng từ trong mỗi tweet

- Reducer: Tạo tập từ có trọng số TF-IDF cao nhất (giảm nhiễu)

2.4.3 Cải tiến và tối ưu hóa

1. **Giảm dữ liệu dư thừa**: Sử dụng các bộ lọc để loại bỏ stopword và từ ít quan trọng trước khi tính TF-IDF

2. **Xử lý song song**: Các Mapper và Reducer được tối ưu hóa để tận dụng tối đa các tài nguyên tính toán trong mô hình MapReduce

3. **Chọn lọc từ ngữ**: Giữ lại 25% từ quan trọng nhất trong mỗi tweet dựa trên giá trị TF-IDF, nhằm tập trung vào từ khóa biểu lộ cảm xúc

2.4.4 Ưu điểm của việc sử dụng TF-IDF

- Tăng độ chính xác của mô hình bằng cách lọc bỏ từ phổ biến không đóng góp ý nghĩa.

- Nổi bật các từ khóa đặc trưng trong từng tweet, cải thiện khả năng phân loại cảm xúc.

- Tăng khả năng mở rộng và hiệu quả trên dữ liệu lớn nhờ tích hợp MapReduce.

2.4.5 Hạn chế

- Tốn thời gian và tài nguyên cho các bước tiền xử lý và tính toán TF-IDF.

- Không xử lý được các vấn đề về ngữ cảnh hoặc ngữ nghĩa của từ ngữ.

2.4.6 Kết quả kỳ vọng

- TF-IDF giúp làm nổi bật các từ quan trọng trong việc xác định cảm xúc của một tweet.

- Hệ thống có độ chính xác cao hơn khi phân loại cảm xúc, đặc biệt trên các tập dữ liệu lớn.

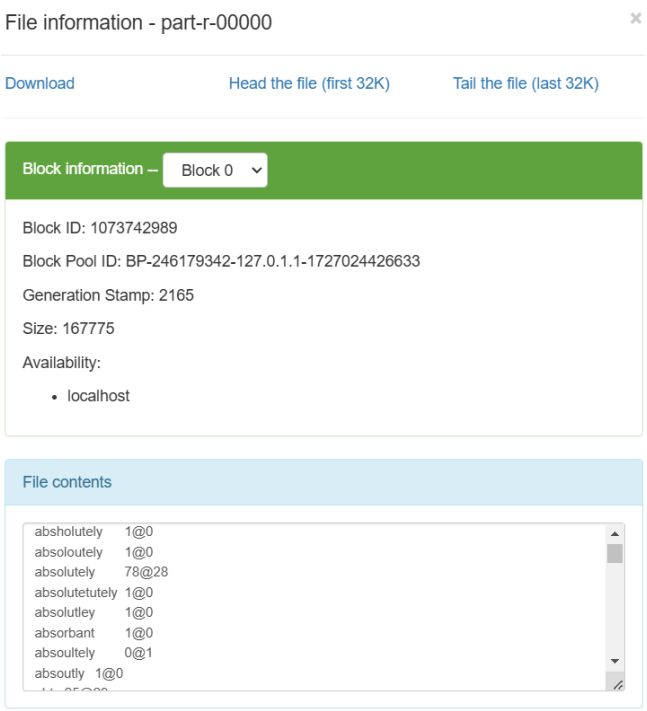
2.5 Demo chương trình cài đặt

2.5.1 Cấu trúc chương trình

- Tệp mã `NativeBayes.java` triển khai các lớp:
  + Mapper (`Map_Training, Map_Testing`): Xử lý dữ liệu đầu vào.
  + Reducer (`Reduce_Training`): Tính toán mô hình xác suất.
- Các **Global Counters** theo dõi thống kê: số lượng từ, số lượng mẫu tích cực/tiêu cực, và hiệu suất dự đoán.

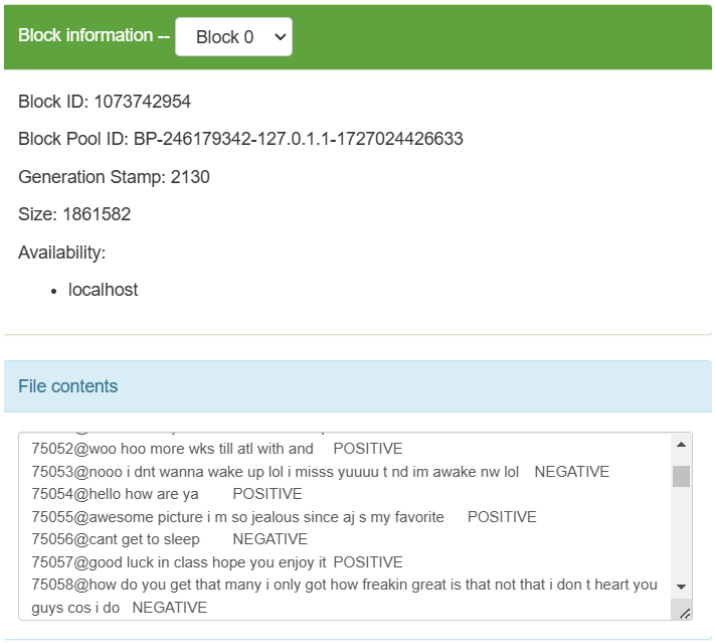
2.5.2 Kết quả

Mô hình huấn luyện:



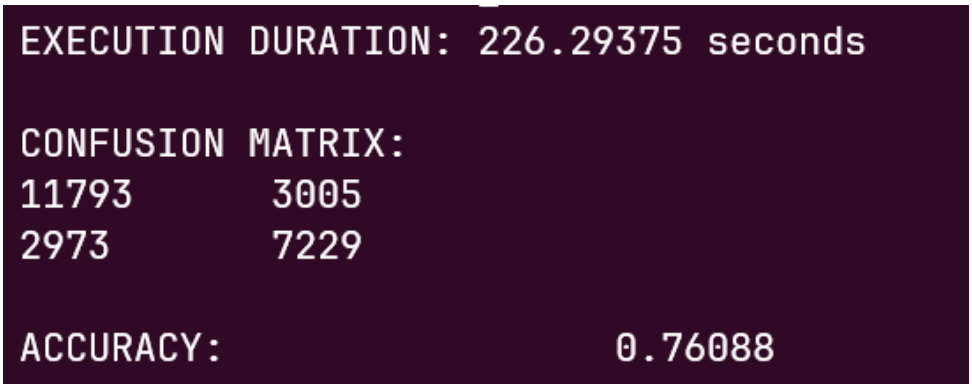
Hình 8: Mô hình huấn luyện

Kết quả:



Hình 9: Kết quả

Confusion Matrix:



Hình 10: confusion matrix

2.5.3 Kết luân

- **Lợi ích**: Mô hình Naive Bayes kết hợp MapReduce xử lý dữ liệu lớn nhanh chóng và hiệu quả.
- **Hạn chế**: Giả thiết về mối quan hệ độc lập giữa các từ có thể không phản ánh đúng ngữ cảnh thực tế.

2.6 Thu thập dữ liệu theo thời gian thực với Kafka, MongoDB

2.6.1 Bài toán phân tích cảm xúc theo thời gian thực

Trong kỷ nguyên mạng xã hội và dữ liệu trực tuyến, phân tích cảm xúc theo thời gian thực trở thành một yêu cầu quan trọng, giúp các doanh nghiệp và tổ chức:

- **Theo dõi phản ứng** người dùng đối với các sản phẩm, dịch vụ hoặc sự kiện.
- **Đưa ra quyết định nhanh chóng** dựa trên các dữ liệu cảm xúc được cập nhật liên tục.
- **Phát hiện xu hướng** và các vấn đề tiềm ẩn trong thời gian thực.

Mục tiêu:

- Xây dựng hệ thống xử lý luồng dữ liệu thời gian thực để phân tích cảm xúc từ các tweet.
- Kết hợp Apache Kafka để truyền tải dữ liệu liên tục và MongoDB để lưu trữ dữ liệu hiệu quả, hỗ trợ việc truy vấn và phân tích nhanh.

2.6.2 Triển khai bài toán

Hệ thống bao gồm các thành phần chính:

1. **Nguồn dữ liệu**: Các tweet được thu thập từ Twitter API hoặc các nguồn tương tự, sau đó đẩy vào Kafka.
2. **Apache Kafka**:
   - Đóng vai trò là hàng đợi tin nhắn (message queue) để truyền tải dữ liệu thời gian thực.
   - Dữ liệu được phân phối qua các chủ đề (topics) trong Kafka, đảm bảo độ tin cậy và khả năng mở rộng.
3. **Xử lý luồng dữ liệu**: Sử dụng Spark Streaming để đọc dữ liệu từ Kafka.
4. **MongoDB**:
   - Lưu trữ kết quả phân tích theo định dạng linh hoạt (document-based).
   - Hỗ trợ truy vấn nhanh để phục vụ hiển thị hoặc báo cáo.

2.6.3 Quy trình hoạt động

1. **Đọc dữ liệu**: Kafka nhận luồng tweet từ nguồn và phân phối tới các consumer.
2. **Xử lý dữ liệu**:
   - Xử lý dữ liệu thô (lọc stopwords, làm sạch văn bản).
   - Dự đoán cảm xúc (tích cực, tiêu cực, trung tính) của từng tweet bằng mô hình đã huấn luyện.

Hình 11: Demo chương trình cài đặt

- Gắn thêm thông tin như timestamp, sentiment score, và từ khóa nổi bật.

3. **Lưu trữ**: Kết quả được ghi vào MongoDB

CHƯƠNG 3: PHÂN TÍCH CẢM XÚC CỦA CÂU SỬ DỤNG SPARK

3.1 Spark trong học máy

Mặc dù thuật toán Naive Bayes trong Hadoop đơn giản và hiệu quả, nhưng nó bị hạn chế bởi giả thiết về mối quan hệ độc lập giữa các đặc trưng và tốc độ xử lý không tối ưu cho dữ liệu lớn. Vì vậy, việc sử dụng Apache Spark với thư viện MLlib là một giải pháp thay thế phù hợp hơn.

Apache Spark là một nền tảng xử lý dữ liệu phân tán mạnh mẽ, được thiết kế để xử lý dữ liệu lớn một cách hiệu quả và nhanh chóng. Trong học máy (machine learning), Spark cung cấp thư viện **MLlib** – một thư viện chuyên dụng cho việc xây dựng và triển khai các mô hình học máy trên dữ liệu lớn.

Ưu điểm:

- Spark được tối ưu hóa để xử lý dữ liệu lớn trong bộ nhớ, giảm thiểu thời gian truy cập đĩa so với Hadoop.
- Tích hợp dễ dàng với HDFS, Cassandra, HBase, và các cơ sở dữ liệu phổ biến khác.
- MLlib cung cấp các thuật toán học máy phổ biến như hồi quy tuyến tính, cây quyết định, cụm K-Means…., hoạt động được hiệu quả đối với dữ liệu lớn.
- Hỗ trợ nhiều ngôn ngữ lập trình, bao gồm Python (PySpark), Scala, Java, và R.

3.2 Bài toán phân loại trong học máy

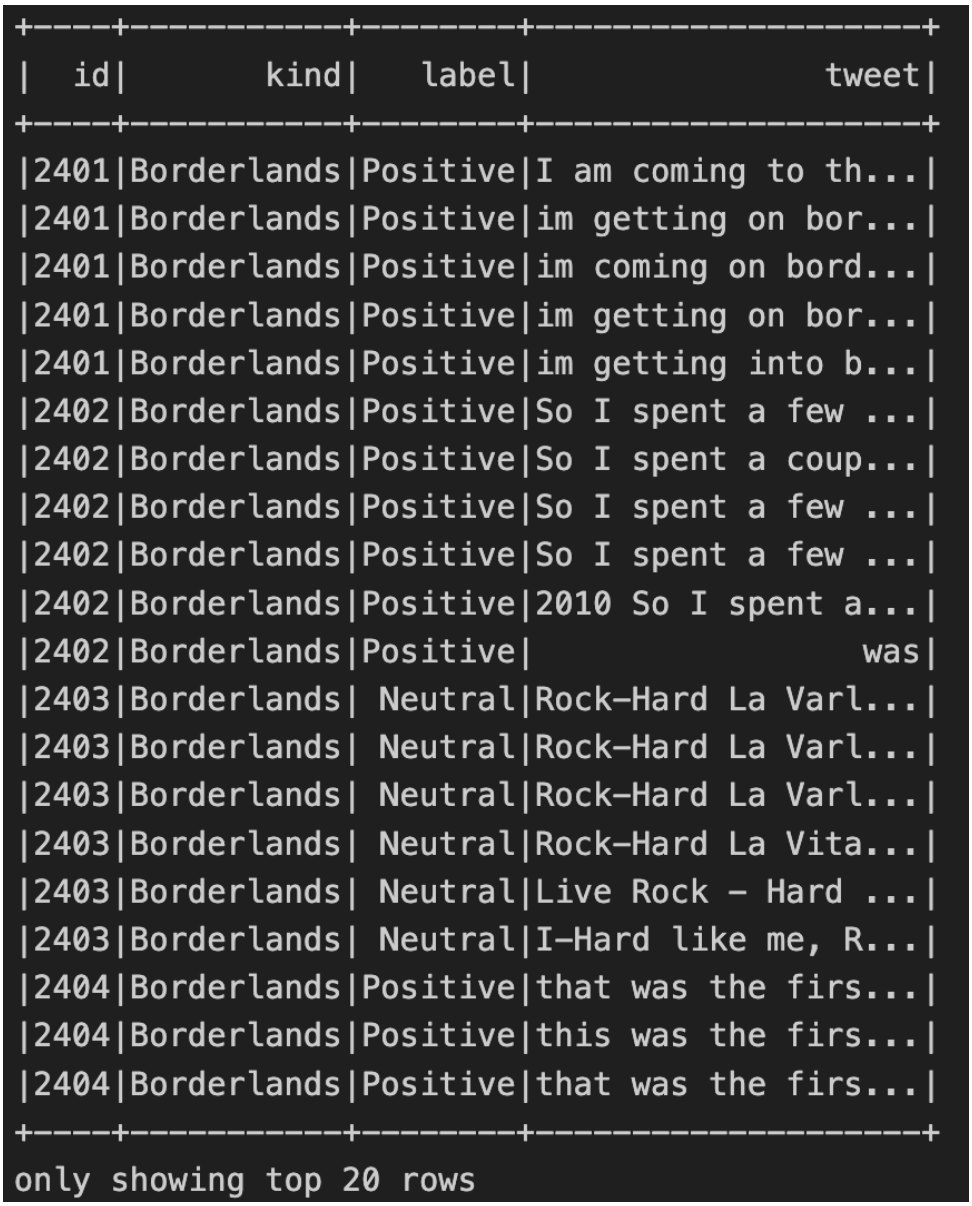
Phân loại (classification) là một trong những bài toán phổ biến trong học máy. Nó bao gồm việc gán một nhãn (label) cho các quan sát đầu vào dựa trên các đặc trưng (features).

Apache Spark hỗ trợ giải quyết bài toán này với thư viện MLlib, cung cấp các thuật toán phân loại như:

- Naive Bayes
- Logistic Regression
- SVM (Support Vector Machines)
- Cây quyết định (Decision Tree)
- Rừng ngẫu nhiên (Random Forest)

3.3 Cài đặt chương trình với PySpark

Input: Bộ dữ liệu bao gồm 2 thành phần chính là bình luận và nhãn của bình luận đó



```
+----+-----------+--------+--------------------+
|  id|       kind|   label|               tweet|
+----+-----------+--------+--------------------+
|2401|Borderlands|Positive|I am coming to th...|
|2401|Borderlands|Positive|im getting on bor...|
|2401|Borderlands|Positive|im coming on bord...|
|2401|Borderlands|Positive|im getting on bor...|
|2401|Borderlands|Positive|im getting into b...|
|2402|Borderlands|Positive|So I spent a few ...|
|2402|Borderlands|Positive|So I spent a coup...|
|2402|Borderlands|Positive|So I spent a few ...|
|2402|Borderlands|Positive|So I spent a few ...|
|2402|Borderlands|Positive|2010 So I spent a...|
|2402|Borderlands|Positive|                 was|
|2403|Borderlands| Neutral|Rock-Hard La Varl...|
|2403|Borderlands| Neutral|Rock-Hard La Varl...|
|2403|Borderlands| Neutral|Rock-Hard La Varl...|
|2403|Borderlands| Neutral|Rock-Hard La Vita...|
|2403|Borderlands| Neutral|Live Rock - Hard ...|
|2403|Borderlands| Neutral|I-Hard like me, R...|
|2404|Borderlands|Positive|that was the firs...|
|2404|Borderlands|Positive|this was the firs...|
|2404|Borderlands|Positive|that was the firs...|
+----+-----------+--------+--------------------+
only showing top 20 rows
```

Hình 12: Demo chương trình cài đặt

Có 4 loại nhãn là: Positive (tích cực), Negative (tiêu cực), Neutral (trung tính), Irrelevant (Không liên quan).

Output: Với input là 1 câu bình luận bất kỳ, đầu ra sẽ là nhãn của nó.

3.4 Các bước triển khai

3.4.1 Bước 1: Tiền xử lý dữ liệu

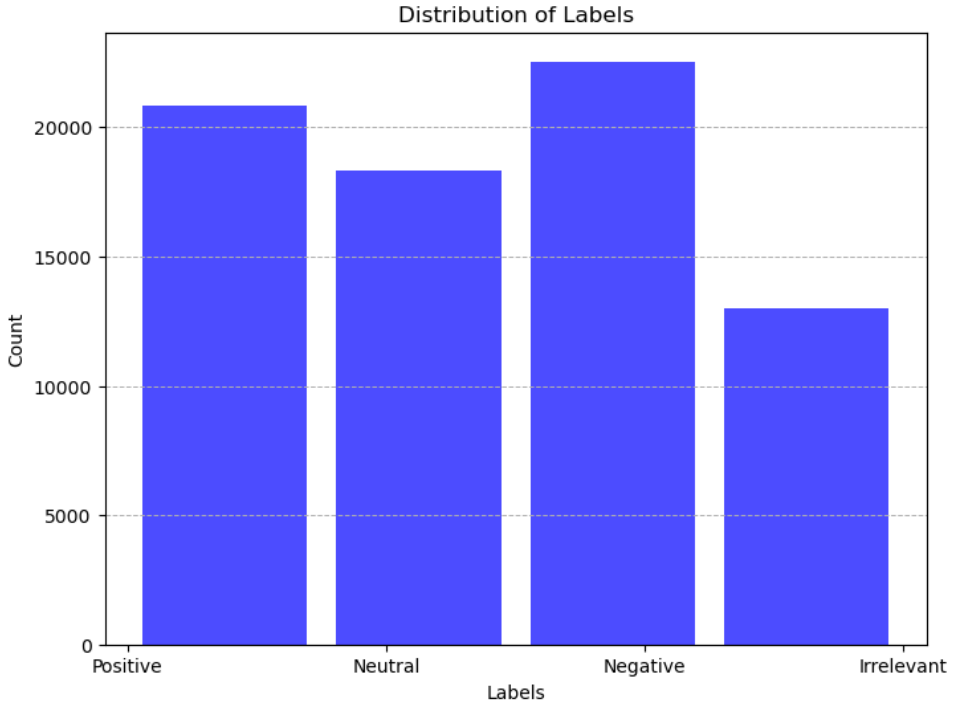
- Loại bỏ tất cả các nhiễu có thể có như hashtag, mention, các đường link...

- Biến đổi chữ hoa thành chữ thường, loại bỏ hết các ký tự đặc biệt ngoại trừ chữ cái thường

và số

- Loại bỏ tất cả các bình luận rỗng

Dữ liệu sau khi được xử lý có phân bố như sau:



Hình 13: Phân bố hiện tại của dữ liệu

3.4.2 Bước 2: Thực hiện các phương pháp biến đổi

- Đổi nhãn dữ liệu sang kiểu số: Negative: 0, Neutral: 1, Positive: 2, Irrelevant: 3

- Tách bình luận thành từng chữ, loại bỏ các stopword

- Làm nổi bật các từ xuất hiện ít mà có giá trị cao, lọc ra những từ xuất hiện phổ biến

```
+-----+--------------------+--------------------+--------------------+--------------------+
|label|               tweet|               words|      filtered_words|            features|
+-----+--------------------+--------------------+--------------------+--------------------+
|    2|i am coming to th...|[i, am, coming, t...|[coming, borders,...|(262144,[12409,14...|
|    2|im getting on bor...|[im, getting, on,...|[im, getting, bor...|(262144,[31015,23...|
|    2|im coming on bord...|[im, coming, on, ...|[im, coming, bord...|(262144,[12409,31...|
|    2|im getting on bor...|[im, getting, on,...|[im, getting, bor...|(262144,[12524,31...|
|    2|im getting into b...|[im, getting, int...|[im, getting, bor...|(262144,[31015,92...|
|    2|so i spent a few ...|[so, i, spent, a,...|[spent, hours, ma...|(262144,[531,1512...|
|    2|so i spent a coup...|[so, i, spent, a,...|[spent, couple, h...|(262144,[531,1512...|
|    2|so i spent a few ...|[so, i, spent, a,...|[spent, hours, so...|(262144,[19698,21...|
|    2|so i spent a few ...|[so, i, spent, a,...|[spent, hours, ma...|(262144,[531,1512...|
|    2|2010 so i spent a...|[2010, so, i, spe...|[2010, spent, hou...|(262144,[531,1512...|
|    2|                was|               [was]|                  []|      (262144,[],[])|
|    1|rock hard la varl...|[rock, hard, la, ...|[rock, hard, la, ...|(262144,[2437,988...|
|    1|rock hard la varl...|[rock, hard, la, ...|[rock, hard, la, ...|(262144,[2437,988...|
|    1|rock hard la varl...|[rock, hard, la, ...|[rock, hard, la, ...|(262144,[2437,988...|
|    1|rock hard la vita...|[rock, hard, la, ...|[rock, hard, la, ...|(262144,[2437,988...|
|    1|live rock   hard ...|[live, rock, , , ...|[live, rock, , , ...|(262144,[2437,988...|
|    1|i hard like me  r...|[i, hard, like, m...|[hard, like, , ra...|(262144,[2437,432...|
|    2|that was the firs...|[that, was, the, ...|[first, borderlan...|(262144,[16793,55...|
|    2|this was the firs...|[this, was, the, ...|[first, borderlan...|(262144,[55875,11...|
|    2|that was the firs...|[that, was, the, ...|[first, borderlan...|(262144,[16793,55...|
+-----+--------------------+--------------------+--------------------+--------------------+
only showing top 20 rows
```

Hình 14: Kết quả sau khi biến đổi

3.4.3 Bước 3: Áp dụng các thuật toán học máy ứng dụng phân loại dữ liệu

Sử dụng 2 thuật toán: Naive Bayes và Logistic Regression để training

- Naive Bayes: bài toán classification với C lớp từ $1, 2, ..., C$. Giả sử có điểm dữ liệu $\mathbf{x}$, Naive Bayes sẽ tính xác suất mà điểm dữ liệu này thuộc vào các lớp tất cả các lớp c: $P(x|c)$. Từ đó, chọn lớp mà có xác suất lớn nhất để kết luận.

- Logistic Regression: thuật toán dựa trên mô hình hồi quy tuyến tính và sử dụng hàm softmax để dự đoán xác suất trong bài toán phân loại nhiều lớp. Với k lớp, xác suất để điểm dữ liệu $\mathbf{x}$ thuộc lớp j là:

$$P(y = j|\mathbf{x}) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

  với $z_j = w_j^\top + b_j$ là điểm số logit của lớp j

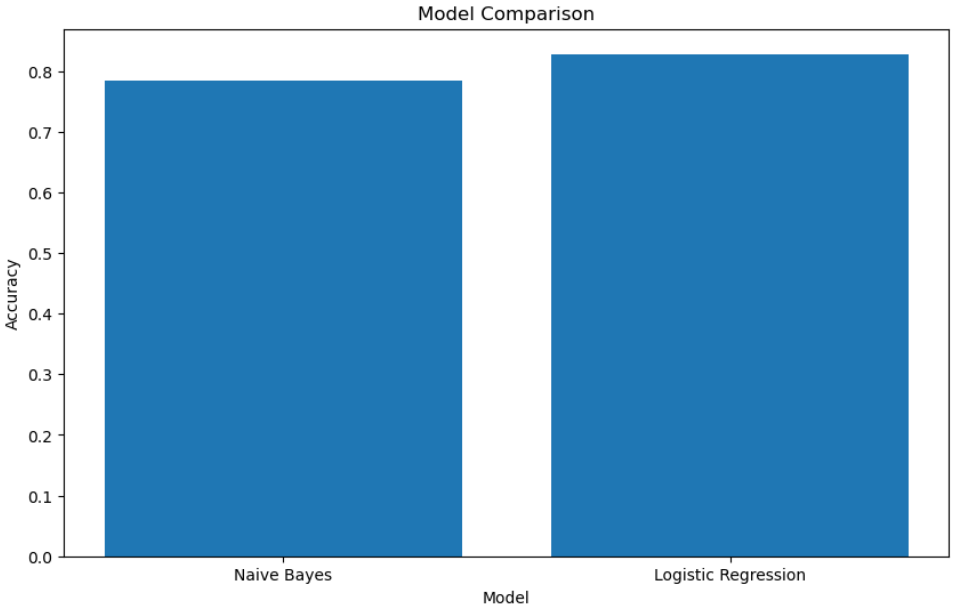
Một số cải tiến:

- Sử dụng grid search

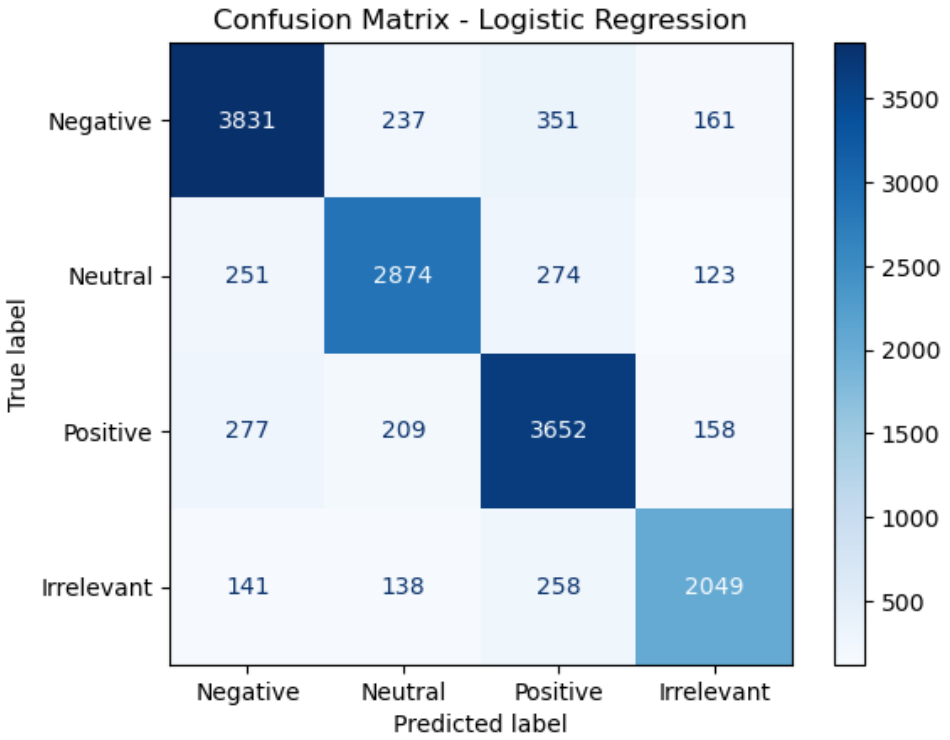
- Sử dụng cross validation

3.4.4 Bước 4: Đánh giá và kết luận

Kết quả khi dự đoán với bộ test:



Hình 15: Model Comparison



Hình 16: Confusion Matrix - Logistic Regression

Hiệu quả của thuật toán:

- Logistic Regression cho thấy hiệu suất vượt trội hơn so với Naive Bayes trong bài toán phân loại đa lớp, đặc biệt là với các lớp có dữ liệu không cân bằng. Điều này nhờ vào khả năng mô hình hóa tốt hơn thông qua hàm softmax.
- Naive Bayes, mặc dù có độ chính xác thấp hơn, nhưng lại thực hiện nhanh và tiêu tốn ít tài nguyên, phù hợp với các bài toán đơn giản hoặc khi cần tốc độ xử lý cao.

Đặc điểm của dữ liệu:

- Một số lớp, như "Irrelevant", có tỷ lệ dữ liệu ít hơn, gây khó khăn cho việc phân loại chính xác. Điều này ảnh hưởng đến hiệu quả tổng thể của các mô hình.
- Các đặc trưng từ TF-IDF có tác động lớn trong việc nâng cao chất lượng dự đoán, nhưng vẫn còn dư địa cải tiến nếu áp dụng thêm các kỹ thuật tiền xử lý hoặc tăng cường dữ liệu.

CHƯƠNG 4: ĐỀ XUẤT MÔ HÌNH LSTM ĐỂ PHÂN LOẠI CẢM XÚC

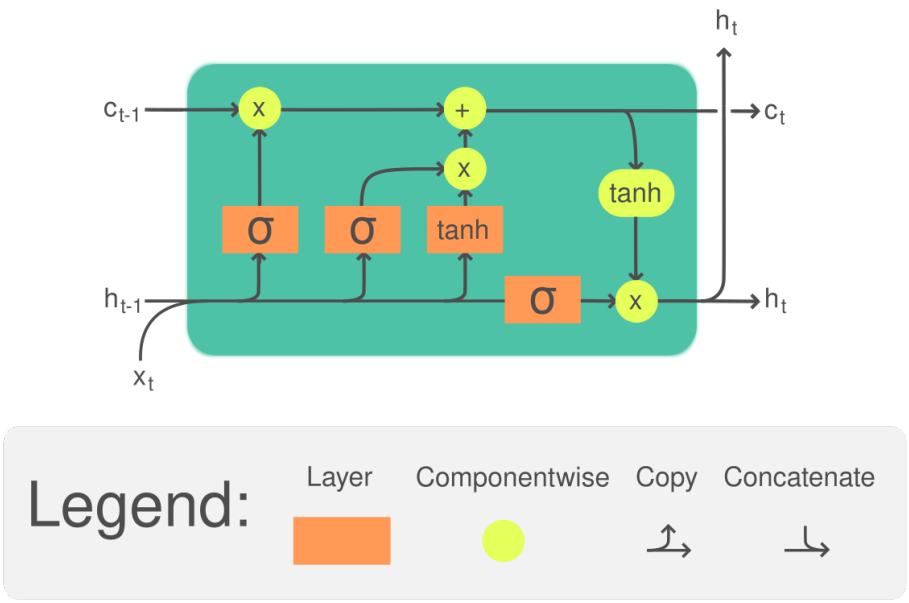
4.1 Bài toán đặt ra

Thông qua các thử nghiệm dựa trên MLlib, các mô hình chỉ được giới hạn trong các thuật toán machine learning cơ bản, tiếp cận theo hướng thống kê. Điều này khiến kết quả của mô hình không đạt được kết quả cao.

Để cải thiện điều đó, nhóm đã quyết định thử nghiệm, xây dựng các mô hình phân loại cảm xúc dựa trên kiến trúc học sâu, thông qua 3 biến thể của mạng LSTM. Việc huấn luyện và tích hợp quá trình infer của mô hình vào trong luồng spark, các metrics như accuracy, f1, .. đạt được kết quả tốt hơn rất nhiều so với phương pháp thống kê học máy.

4.2 Tổng quan về LSTM

Định nghĩa về mạng LSTM: Mạng Long Short-Term Memory (LSTM) là một biến thể của mạng hồi quy (Recurrent Neural Network - RNN), được thiết kế để giải quyết vấn đề về **quên thông tin dài hạn** trong kiến trúc RNN truyền thống. Với cấu trúc đặc biệt bao gồm các gate, LSTM có khả năng chọn lọc thông tin nào cần giữ lại hoặc loại bỏ qua các bước thời gian. Điều này giúp mô hình hoạt động và xử lý rất tốt trên các dạng dữ liệu tuần tự như chuỗi thời gian, văn bản, giọng nói,. . .



Hình 17: Kiến trúc LSTM

Sentiment Analysis là một bài toán nổi bật trong NLP với mục tiêu là phân loại cảm xúc của

một đoạn văn bản (tích cực, tiêu cực, hoặc trung tính). Dựa vào nhưng ưu điểm sau, nhóm quyết định lựa chọn kiến trúc LSTM để xử lý bài toán:

1. Khả năng xử lý ngữ cảnh dài hạn: LSTM có thể hiểu mối liên hệ giữa các từ trong văn bản, ngay cả khi chúng cách xa nhau.

2. Hiểu được cấu trúc câu: Phân tích cảm xúc không chỉ dựa trên từ ngữ riêng lẻ mà còn phụ thuộc vào ngữ cảnh của câu.

3. Xử lý tốt dữ liệu có sự đa nghĩa: LSTM có thể nhận diện cảm xúc dựa trên ngữ cảnh cụ thể, chẳng hạn như câu "Bộ phim này thật sự không tệ lắm"thường mang ý nghĩa tích cực.

4.3 Xử lý dữ liệu

Để huấn luyện các mô hình học sâu, dữ liệu text đầu vào cần được tiền xử lý trước đó, chuyển đổi thành các token tương ứng. Quá trình xử lý này được mô tả như sau:

1. Với từng dòng text đầu vào, text này sẽ được loại bỏ các kí tự đặc biệt, stopword và được chuẩn hóa về một dạng chữ thường. Thông qua việc xử lý này, các dữ liệu none hoặc bị trùng lặp sẽ được loại bỏ.

2. Nhóm sử dụng lớp Tokenizer có sẵn trong thư viện keras và fit lớp này vào từng dòng text đã được xử lý trước đó (với vocab size mặc định là 10.000).

3. Với nhãn, sử dụng LabelEncoder nhằm chuyển chúng về dạng số từ $1 \rightarrow 4$.

4. Để tối ưu chiều dài của từng câu, nhóm lựa chọn max len từng câu là 50, với câu dài hơn, nó sẽ bị cắt bỏ, câu ngăn hơn sẽ được thêm các token <pad> nhằm lấp đầy thông tin. Sau quá trình xử lý , text sẽ được tokenize thành các chuỗi id.

5. Cuối cùng, nhóm xây dựng Dataloader với batch size là 64

Kết thúc quá trình xử lý dữ liệu, bộ train set có 67219 câu và val set có 996 câu.

4.4 Các kiến trúc mạng LSTM được sử dụng

Ban đầu, nhóm sử dụng một kiến trúc mạng LSTM thuần với các lớp như sau:

1. Lớp Embedding: Chuyển đổi word indices trong từ vựng vocab thành các vector với số chiều cố định.

2. Lớp LSTM: có kiến trúc mạng LSTM, xử lý chuỗi đầu vào dưới dạng vector nhúng, lưu trữ thông tin ngữ cảnh thông qua các bước thời gian, và học mối quan hệ giữa các từ trong câu. Đầu ra của lớp này là trạng thái ẩn cuối cùng (hidden state) và trạng thái bộ nhớ cuối cùng (cell state).

3. Lớp Fully Connected: Chuyển đổi trạng thái ẩn cuối cùng của LSTM thành vector đầu ra có kích thước bằng số lớp nhãn (output_dim).

Tiếp theo đó, nhóm thử nghiệm với mô hình LSTM hai chiều kết hợp với cơ chế attention để tập trung vào các phần quan trọng của chuỗi đầu vào. Kiến trúc của lớp có dạng như sau:

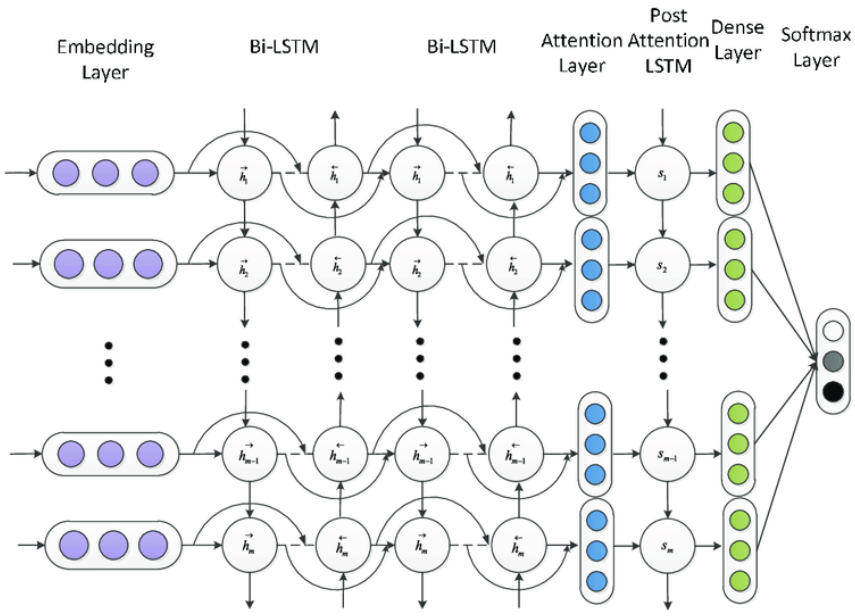
1. Lớp Embedding: Chuyển đổi word indices trong từ vựng vocab thành các vector với số chiều cố định.

2. Lớp Bidirectional LSTM: LSTM học mối quan hệ tuần tự giữa các từ, với cấu trúc hai chiều giúp mô hình học cả ngữ cảnh phía trước và phía sau của mỗi từ.

3. Cơ chế attention: Học trọng số (weights) cho từng trạng thái ẩn của LSTM thông qua cơ chế attention.

4. Lớp Fully Connected: Chuyển đổi vector ngữ cảnh (context vector) được tạo bởi attention thành đầu ra với số lớp tương ứng cần phân loại.

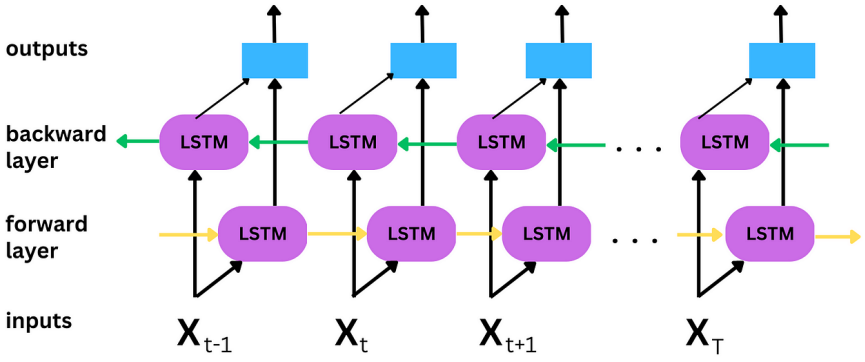
5. Lớp dropout: Dropout ngẫu nhiên một số kết nối để tránh hiện tượng overfitting trong quá trình huấn luyện.



Hình 18: Kiến trúc SA-LSTM

Cuối cùng, nhóm thử nghiệm kiến trúc LSTM hai chiều xếp chồng (stacked Bidirectional LSTM) kết hợp với các lớp chuẩn hóa (batch normalization) và dropout để cải thiện hiệu suất và giảm overfitting với thành phần như sau:

1. Lớp Embedding: Chuyển đổi word indices trong từ vựng vocab thành các vector với số chiều cố định.
2. Lớp Bidirectional LSTM: Mô hình sử dụng **hai lớp LSTM hai chiều, giúp trích xuất các thông tin ngữ cảnh một cách tối đa, tổng quan hơn**.
3. Lớp Batch Normalization: được cài đặt xen kẽ giữa các lớp LSTM nhằm chuẩn hóa các đầu ra của LSTM để tăng tốc độ hội tụ và ổn định việc huấn luyện.
4. Lớp Dropout được áp dụng sau mỗi bước quan trọng (sau embedding, LSTM1, LSTM2, và Fully Connected Layer) để giảm overfitting.
5. Lớp Fully Connected: mô hình sử dụng 2 lớp fc nhằm trích xuất đặc trưng một cách tối đa.



Hình 19: Kiến trúc BiLSTM

4.5 Các cài đặt tham số của mô hình

Ngoài ra, các mô hình được huấn luyện với tối đa 15 epochs, batch_size là 64 và early stopping để thuật toán dừng khi không thể cải thiện thêm hoặc overfitting.

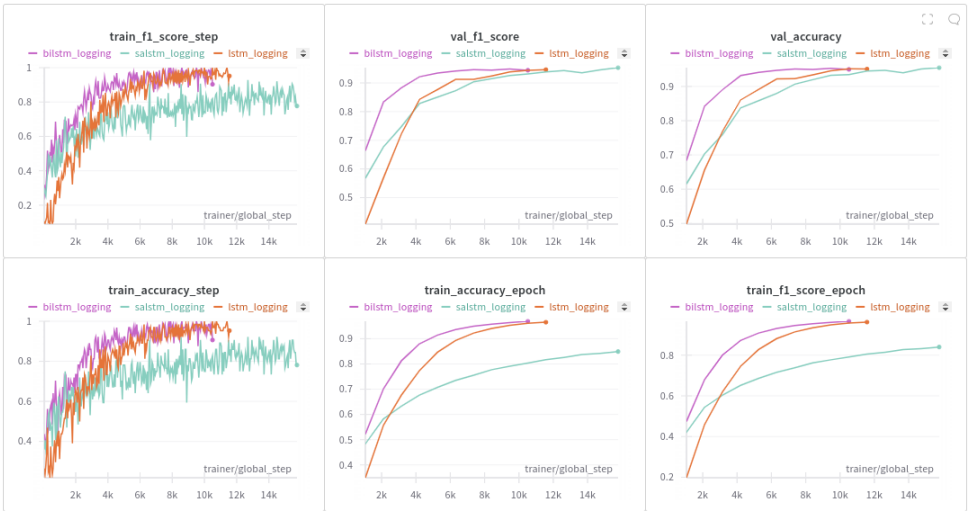
Với learning rate, ban đầu được khởi tạo là 10e-3 và sử dụng ReduceLROnPlateau scheduler để thay đổi learning rate tùy thuộc vào loss hiện tại của mô hình. Tương tự như các bài toán phân loại khác, CrossEntropy được sử dụng tính loss với optimizer là Adam. Mô hình được huấn luyện trên GPU P100.

Thành phần	LSTMModel	SALSTMModel	BiLSTMModel
Kích thước từ vựng	10.000	10.000	10.000
Số chiều embedding	64	100	100
Số chiều ẩn	128	256	128 / 64
Số lớp LSTM	1	2	2
Số chiều đầu ra	4	4	4
Dropout rate	–	0.5	0.5
LSTM 2 chiều	–	x	x
Sử dụng Attention	–	x	–

Bảng 2: So sánh các mô hình LSTM, SALSTM, và BiLSTM

4.6 Kết quả

Kết thúc quá trình huấn luyện, các mô hình được đánh giá như sau:



Hình 20: Kết quả huấn luyện

Các mô hình đều có kết quả tương đối giống nhau trên tập validation, giao động trong khoảng từ 0.94 - 0.95 với các chỉ số f1 và accuracy.

Hình 21: Kết quả huấn luyện

Trên tập huấn luyện, các mô hình BiLSTM và LSTM hội tụ nhanh hơn và đạt độ chính xác cao hơn đáng kể (khoảng 0.88 - 0.9) so với mô hình tích hợp attention. Sự chênh lệch giữa độ chính xác trên tập huấn luyện và tập kiểm tra xuất phát từ sự khác biệt đáng kể về kích thước giữa hai tập dữ liệu. Mô hình sử dụng attention vẫn có tiềm năng cải thiện thêm, do quá trình

huấn luyện chưa dừng lại bởi early stopping và hàm loss vẫn tiếp tục giảm.

Từ những kết quả trên, có thể kết luận rằng mô hình hoàn toàn khả thi và có thể ứng dụng hiệu quả trong thực tế.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Big data đã mang lại cho các tổ chức và doanh nghiệp nhiều cơ hội, thách thức cũng như những tài sản quý giá. Với khả năng xử lý dữ liệu lớn, MapReduce chia nhỏ công việc, phân tán qua các nút tính toán, sau đó tổng hợp kết quả. Đề tài đã ứng dụng mô hình MapReduce và các công cụ Spark để phân tích cảm xúc của từ trong câu và phân cụm dữ liệu, đạt được các kết quả sau:

- Hiểu tổng quan về Big data, Hadoop, Pyspark, MapReduce.
- Hiểu chi tiết về bài toán phân tích cảm xúc (sentiment analysis).
- Hiểu chi tiết về thuật toán Naive Bayes, SVM, Logistic Regression.
- Triển khai ý tưởng và giải pháp MapReduce hóa sentiment analysis.
- Xây dựng thành công các chương trình demo cho đề tài.
- Sử dụng mô hình LSTM (Long Short-Term Memory) để xử lý bài toán phân tích cảm xúc theo thời gian thực, đặc biệt với các dữ liệu có tính liên tục và phụ thuộc ngữ cảnh cao.
- Ứng dụng thành công Kafka vào xử lý dữ liệu theo thời gian thực.
- Đánh giá chương trình.

Tuy nhiên kết quả vẫn còn một số hạn chế:

- Dữ liệu chạy bài toán phân loại cảm xúc vẫn còn ở mức nhỏ.

5.2 Hướng phát triển

Áp dụng kiến thức về Big Data, Apache Hadoop, Pyspark, cải tiến và xây dựng ứng dụng phân tích cảm xúc vào nhiều lĩnh vực khác nhau. Trong quá trình hoàn thành bài tập lớn, nhóm em đã cố gắng tìm hiểu và tham khảo các tài liệu liên quan. Tuy nhiên, thời gian có hạn nên chúng em sẽ không tránh khỏi những thiếu sót, rất mong nhận được sự đóng góp ý kiến của thầy cô và các bạn để báo cáo và kỹ năng của chúng em ngày được hoàn thiện hơn. Dưới đây là một số đề xuất về phương hướng phát triển trong tương lai:

- Ứng dụng Apache Hive để lưu trữ và phân tích lượng lớn dữ liệu văn bản, cải thiện khả năng mở rộng và hiệu quả chi phí.

- Sử dụng Apache Pig và Hive để xử lý, tiền xử lý các tập dữ liệu lớn.
- Mở rộng quy mô dữ liệu và áp dụng vào nhiều lĩnh vực khác nhau.

NHIỆM VỤ CÁC THÀNH VIÊN

Họ và tên	**Công việc**
Nguyễn Phương Trang	+ Tìm hiểu mô hình cải tiến + Cài đặt giao diện + Làm báo cáo và slide
Nguyễn Ngô Việt Trung	+ Tìm hiểu mô hình cải tiến + Cài đặt mô hình LSTM + Làm báo cáo
Nguyễn Tiến Trung	+ Tìm hiểu tổng quan về Spark + Triển khai thuật toán Naive Bayes/SVM trên Spark + Làm báo cáo
Nguyễn Đức Tước	+ Tìm hiểu tổng quan về Hadoop + Triển khai thuật toán Naive Bayes trên Hadoop + Làm báo cáo

TÀI LIỆU THAM KHẢO

1. https://github.com/Coursal/Text-Sentiment-Analysis-In-Hadoop-And-Spark
2. https://github.com/SaeedNajafi/infer-pytorch-pyspark
3. https://github.com/drisskhattabi6/Real-Time-Twitter-Sentiment-Analysis