International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 9 Issue 7 ǁ 2021 ǁ PP. 55-61

Effective Analysis of Sentiment Twitter Data Set Using a Unified Analytic Engine - Apache Spark

Manjunath R[1], Abhilash L Bhat[2], Sumanth V[3]
[1]Professor, Dept of CSE, R.R. Institute of Technology, Bengaluru
[2]Assistant Professor, Dept of ISE, R.R. Institute of Technology, Bengaluru
[3]Assistant Professor, Dept of CSE, R.R. Institute of Technology, Bengaluru

Abstract
Sentiment analysis has become an interesting field for both research and industrial domains. The expression sentiment refers to the feelings or thought of the person across some certain issues. Furthermore, it is also considered a direct application for opinion mining. Actually, the amount of data, which is massive, grows rapidly per second and this is called big data which requires special processing techniques and high computational power in order to perform the required mining tasks. In this project, we perform a sentiment analysis with the help of Apache Spark framework ,The goal of using Apache Spark's Machine learning library (MLIB) is to handle an extraordinary amount of data effectively We recommend some Pre-processing and Machine learning text feature extraction steps for getting greater results in Sentiment Analysis classification.
***Keywords**: Sentiment Analysis, Machine learning, Apache Spark, Big data, Opinion Mining*

Date of Submission: 06-07-2021 Date of acceptance: 19-07-2021

I. INTRODUCTION

Sentiment analysis has become an interesting field for both research and industrial domains. The expression sentiment refers to the feelings or thought of the person across some certain issues. Furthermore, it is also considered a direct application for opinion mining. The huge amount of unstructured data has been the source of textual data and one of the most essential data volumes; therefore, this data has different aims, such as business, industrial or social aims according to the data requirement and needed processing. Actually, the amount of data, which is massive, grows rapidly per second and this is called big data which requires special processing techniques and high computational power in order to perform the required mining tasks. In this project, we perform a sentiment analysis with the help of Apache Spark framework, which is considered an open source distributed data processing platform which utilizes distributed memory abstraction. The goal of using Apache Spark's Machine learning library (MLIB) is to handle an extraordinary amount of data effectively. We recommend some Pre-processing and Machine learning text feature extraction steps for getting greater results in Sentiment Analysis classification. The effectiveness of our proposed approach is proved against other approaches achieving better classification results when using Naïve Bayes, and Decision trees classification algorithms. Finally, our solution estimates the performance of Apache Spark concerning its scalability.

1.1 Introduction to Big data

Big data is a blanket term for the non-traditional strategies and technologies needed to gather, organize, process, and gather insights from large datasets. While the problem of working with data that exceeds the computing power or storage of a single computer is not new, the pervasiveness, scale, and value of this type of computing has greatly expanded in recent years. In this we will talk about big data on a fundamental level and define common concepts you might come across while researching the subject. We will also take a high-level look at some of the processes and technologies currently being used in this space.

What Is Big Data?
An exact definition of "big data" is difficult to nail down because projects, vendors, practitioners, and business professionals use it quite differently. With that in mind, generally speaking, **big data** is:

- large datasets
- the category of computing strategies and technologies that are used to handle large datasets

In this context, "large dataset" means a dataset too large to reasonably process or store with traditional tooling or on a single computer. This means that the common scale of big datasets is constantly shifting and may vary significantly from organization to organization.

Why Are Big Data Systems Different?

The basic requirements for working with big data are the same as the requirements for working with datasets of any size. However, the massive scale, the speed of ingesting and processing, and the characteristics of the data that must be dealt with at each stage of the process present significant new challenges when designing solutions. The goal of most big data systems is to surface insights and connections from large volumes of heterogeneous data that would not be possible using conventional methods. In 2001, Gartner's Doug Laney first presented what became known as the "three Vs of big data" to describe some of the characteristics that make big data different from other data processing:

Volume

The sheer scale of the information processed helps define big data systems. These datasets can be orders of magnitude larger than traditional datasets, which demands more thought at each stage of the processing and storage life cycle. Often, because the work requirements exceed the capabilities of a single computer, this becomes a challenge of pooling, allocating, and coordinating resources from groups of computers. Cluster management and algorithms capable of breaking tasks into smaller pieces become increasingly important.

Velocity

Another way in which big data differs significantly from other data systems is the speed that information moves through the system. Data is frequently flowing into the system from multiple sources and is often expected to be processed in real time to gain insights and update the current understanding of the system. This focus on near instant feedback has driven many big data practitioners away from a batch-oriented approach and closer to a real-time streaming system. Data is constantly being added, massaged, processed, and analyzed in order to keep up with the influx of new information and to surface valuable information early when it is most relevant. These ideas require robust systems with highly available components to guard against failures along the data pipeline.

Variety

Big data problems are often unique because of the wide range of both the sources being processed and their relative quality. Data can be ingested from internal systems like application and server logs, from social media feeds and other external APIs, from physical device sensors, and from other providers. Big data seeks to handle potentially useful data regardless of where it's coming from by consolidating all information into a single system. The formats and types of media can vary significantly as well. Rich media like images, video files, and audio recordings are ingested alongside text files, structured logs, etc. While more traditional data processing systems might expect data to enter the pipeline already labeled, formatted, and organized, big data systems usually accept and store data closer to its raw state. Ideally, any transformations or changes to the raw data will happen in memory at the time of processing.

Characteristics of Big data

Various individuals and organizations have suggested expanding the original three Vs, though these proposals have tended to describe challenges rather than qualities of big data. Some common additions are:

- **Veracity**: The variety of sources and the complexity of the processing can lead to challenges in evaluating the quality of the data (and consequently, the quality of the resulting analysis)
- **Variability**: Variation in the data leads to wide variation in quality. Additional resources may be needed to identify, process, or filter low quality data to make it more useful.
- **Value**: The ultimate challenge of big data is delivering value. Sometimes, the systems and processes in place are complex enough that using the data and extracting actual value can become difficult.

What Does a Big Data Life Cycle Look Like?

So how is data actually processed when dealing with a big data system? While approaches to implementation differ, there are some commonalities in the strategies and software that we can talk about generally. While the steps presented below might not be true in all cases, they are widely used.
The general categories of activities involved with big data processing are:

- Ingesting data into the system
- Persisting the data in storage
- Computing and Analyzing data
- Visualizing the results

Before we look at these four workflow categories in detail, we will take a moment to talk about **clustered computing**, an important strategy employed by most big data solutions. Setting up a computing cluster is often the foundation for technology used in each of the life cycle stages. Clustered Computing: Because of the qualities of big data, individual computers are often inadequate for handling the data at most stages. To better address the high storage and computational needs of big data, computer clusters are a better fit. Big data clustering software combines the resources of many smaller machines, seeking to provide a number of benefits:

- **Resource Pooling**: Combining the available storage space to hold data is a clear benefit, but CPU and memory pooling is also extremely important. Processing large datasets requires large amounts of all three of these resources.
- **High Availability**: Clusters can provide varying levels of fault tolerance and availability guarantees to prevent hardware or software failures from affecting access to data and processing. This becomes increasingly important as we continue to emphasize the importance of real-time analytics.
- **Easy Scalability**: Clusters make it easy to scale horizontally by adding additional machines to the group. This means the system can react to changes in resource requirements without expanding the physical resources on a machine.

Using clusters requires a solution for managing cluster membership, coordinating resource sharing, and scheduling actual work on individual nodes. Cluster membership and resource allocation can be handled by software like **Hadoop's YARN** (which stands for Yet Another Resource Negotiator) or **Apache Mesos**. The assembled computing cluster often acts as a foundation which other software interfaces with to process the data. The machines involved in the computing cluster are also typically involved with the management of a distributed storage system, which we will talk about when we discuss data persistence.

1.2 Apache Spark

Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching and optimized query execution for fast queries against data of any size. Simply put, **Spark is** a fast and general engine for large-scale data processing.

Apache Spark is a unified analytics engine for large-scale data processing. Fig 1 shows Apache Spark Eco system where it provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for graph processing, and Structured Streaming for incremental computation and stream processing.

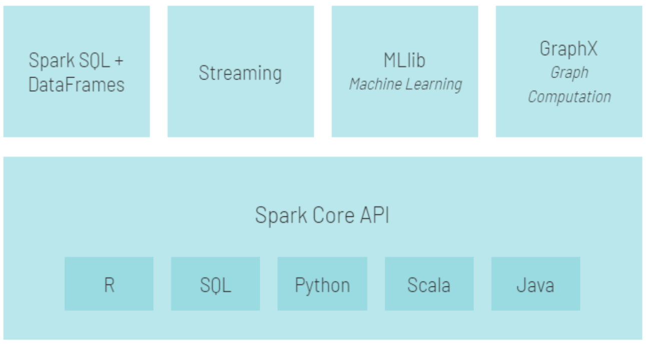

Figure1: Apache Spark Ecosystem

Benefits of Apache Spark

Speed: Engineered from the bottom-up for performance, Spark can be 100x faster than Hadoop for large scale data processing by exploiting in memory computing and other optimizations. Spark is also fast when data is stored on disk, and currently holds the world record for large-scale on-disk sorting.

Ease of Use: Spark has easy-to-use APIs for operating on large datasets. This includes a collection of over 100 operators for transforming data and familiar data frame APIs for manipulating semi-structured data.

A Unified Engine: Spark comes packaged with higher-level libraries, including support for SQL queries, streaming data, machine learning and graph processing. These standard libraries increase developer productivity and can be seamlessly combined to create complex workflows.

II. LITERATURE SURVEY

[1] Text classification is one of the most widely used natural language processing technologies. Common text classification applications include spam identification, news text classification, information retrieval, emotion analysis, and intention judgment, etc. Traditional text classifiers based on machine learning methods have defects such as data sparsity, dimension explosion and poor generalization ability, while classifiers based on deep learning network greatly improve these defects, avoid cumbersome feature extraction

process, and have strong learning ability and higher prediction accuracy. For example,convolutional neural network (CNN).This paper introduces the process of text classification and focuses on the deep learning model used in text classification.

[2] We propose a language model of mix CNN (Convolution Neural Network) with bi-RNN (Bidirectional Recurrent Neural Network) to classify the text at the character-level. Unlike word-level model is that avoiding the problem of unregistered words and improves the robustness of the text representation in character-level model. The language model mainly uses the data augment by different convolution filters of CNN and then the bi-RNN obtain the contextual information in both directions to classify the text. The results show that this model have a better performance than the common CNN and LSTM(long short-term memory) classification methods.

[3] In recent years, with the rapid development of Internet Technology, text data is growing rapidly every day. Users need to filter out the information they need from a large amount of text. Therefore, automatic text classification technology can help users find information. In order to address problems, such as ignoring contextual semantic links and different vocabulary importance in traditional text classification techniques, this paper presents a vector representation of feature words based on the deep learning tool Word2vec, and the weight of the feature words is calculated by the improved TF-IDF algorithm. By multiplying the weight of the word and the word vector, the vector representation of the word is realized. Finally, each text is represented by accumulating all the word vectors. Thus, text classification is carried out.

[4] As an important algorithm used in text classifications, the K- Nearest Neighbors (KNN) has the advantage of simple and effective. However, its computational overhead is very high. Addressing this problem, an improved KNN algorithm named KM-RS-KNN which introduces the K-medoids and rough set to the KNN is proposed in this paper. The K-medoids is used to reduce the training data by similarity between the new incoming text and the cluster center, while the upper and lower approximations in the rough set theory are used to reduce the sample search space. The simulation shows that KM-RS-KNN has better classification efficiency and classification effect than the traditional KNN algorithm.

[5] With the rapid development of mobile Internet, the network has become an important medium for people to exchange information. The research on text classification has practical significance. Using the Hadoop platform to parallelize the KNN classification algorithm can quickly and accurately classify the text, but when calculating the similarity or distance of the sample points, the KNN algorithm will increase with the increase of the sample data, which will lead to the algorithm time. Increased complexity and reduced classification accuracy. Therefore, Parallel Processing of Improved KNN text classification algorithm based on Hadoop platform is proposed. The CLARA clustering algorithm is used to cut out the samples with low similarity in the dataset, and the calculation of sample distance in the dataset is reduced. Then design the parallel KNN MapReduce program to classify the network public opinion data. The experimental results show that the improved parallel KNN algorithm improves the accuracy and time of text classification.

[6] The traditional text classification methods are based on machine learning. It requires a large amount of artificially labeled training data as well as human participation. However, it is common that ignoring the contextual information and the word order information in such a way, and often exist some problems such as data sparseness and latitudinal explosion. With the development of deep learning, many researchers have also been using deep learning in text classification. This paper investigates the application issue of NLP in text classification by using the Bi-LSTM-CNN method. For the purpose of improving the accuracy of text classification, a kind of comprehensive expression is employed to accurately express semantics. The experiment shows that the model in this paper has great advantages in the classification of news texts.

[7] Sentiment analysis has become an interesting field for both research and industrial domains. The expression sentiment refers to the feelings or thought of the person across some certain issues. Furthermore, it is also considered a direct application for opinion mining. The huge amount of tweets jotted down daily makes Twitter a rich source of textual data and one of the most essential data volumes; therefore, this data has different aims, such as business, industrial or social aims according to the data requirement and needed processing. Actually, the amount of data, which is massive, grows rapidly per second and this is called big data which requires special processing techniques and high computational power in order to perform the required mining tasks. In this work, we perform a sentiment analysis with the help of Apache Spark framework, which is considered an open source distributed data processing platform which utilizes distributed memory abstraction. The goal of using Apache Spark's Machine learning library (MLIB) is to handle an extraordinary amount of data effectively. We recommend some Preprocessing and Machine learning text feature extraction steps for getting

greater results in Sentiment Analysis classification. The effectiveness of our proposed approach is proved against other approaches achieving better classification results when using Naïve Bayes, Logistic Regression and Decision trees classification algorithms.

[8] Action Rules are vital data mining method for gaining actionable knowledge from the datasets. Meta actions are the sub-actions to the Action Rules, which intends to change the attribute value of an object, under consideration, to attain the desirable value. The essence of this paper to propose a new optimized and more promising system, in terms of speed and efficiency, for generating meta-actions by implementing Specific Action Rule discovery based on Grabbing strategy (SARGS) algorithm. For this, we perform a comparative analysis of meta-actions generating algorithmic implementation in Apache Spark driven system and conventional Hadoop driven system using the Twitter social networking data and evaluate the results.

III. METHODOLOGY

3.1 System Architecture

The System Architecture is as shown in the below figure.

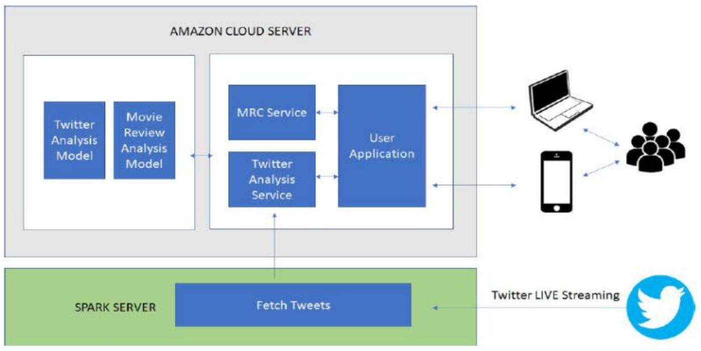

Figure2: System Architecture

The major components are

Tweets Analysis Model and Movie Review Classification Model

In this module, we will be implementing Multinomial Naïve Bayes Algorithm for analysing the sentiment of the inputted tweet. We will be using a dedicated set of training and testing data sets to do that. The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work. In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a nonprobabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

Service Applications

In this module, we will be implementing a couple of REST based applications to expose the previously implemented models to the outside world. This REST APIs will be the POST methods. The first Service application will expose the API which consumes the Tweet along with the user's email ID who requested for analysing that tweet. The service will then run the twitter sentiment analysis model against this tweet to find out the sentiment and updates it in the database so that this can be visualized in the dashboard of the user application. The second Service application will expose the API which consumes the Movie review from the user and analyzes the sentiment of that review (either positive or negative) based on the IMDB datasets we have earlier used for training purpose. The output will be sent back to the user who invoked this web service.

Twitter Live Streaming Using Spark and Scala

Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams. In Spark Streaming, the data can be ingested from many sources like Kafka, Flume, Twitter, ZeroMQ, Kinesis, or TCP sockets, and can be processed using complex algorithms expressed with high-level functions like map, reduce, join and window. Finally, the processed data can be pushed out to the filesystems, databases, and live dashboards. In fact, you can apply Spark's machine learning and graph processing algorithms on data streams. Spark Streaming provides a high-level abstraction called discretized stream or DStream, which represents a continuous stream of data. DStreams can be created either from input data streams from various sources such as Kafka, Flume, and Kinesis, or by applying high-level operations on other DStreams. Internally, a DStream is represented as a sequence of RDDs.

User Application

This application provides the following five operations to the end users.

User Profile Operations: This module implements the basic user profile operations on the prototype application. The user profile operations include creating a new account, logging in to the existing account, logging out, editing the profile, changing the password, and deleting the profile if not needed anymore. This application is also deployed on the cloud server so that this can be accessed by anyone across the globe using the IP address of this cloud server. The implementation is done using the J2EE architecture and for the database needs we have used SQLITE3.

Dashboard: In this component, the users will see a summary of the sentiment of all the tweets they have streamed from scala to this application. It will show the sum of positive tweets and the sum of negative tweets.

Tweets Analysis: In this component, the users will see a detailed analysis of the tweets including the actual tweets text pulled from the scala Application.

Adhoc Run: This component can be used by the users if they have any text (temporary or adhoc) which has to be analysed for sentiment before posting that text in the digital media.

Movie Review Analysis: This component will allow the users to add a new movie to the collections, view all the movies added by any users, view the rating of the movies, provide the rating to the added movie. The component analyses the reviews of all the users and computes the rating for each movie.

3.2 Sequence diagram

A sequence diagram in a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It shows the participants in an interaction and the sequence of messages among them; each participant is assigned a column in a table.

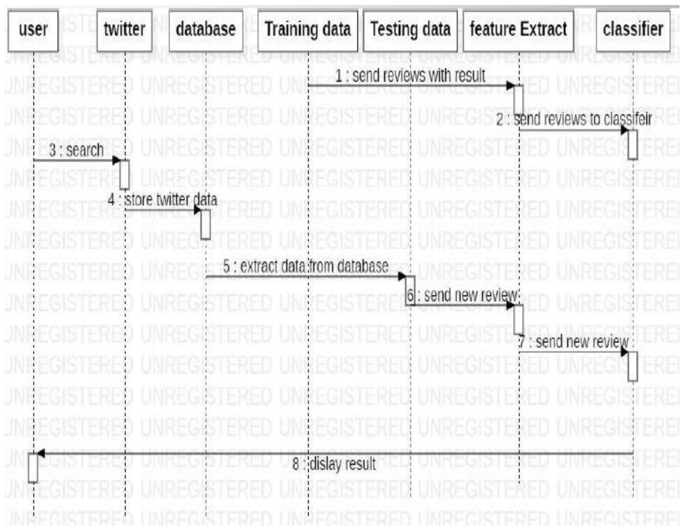

Figure3: Sequence diagram

IV. CONCLUSION AND FUTURE SCOPE

An efficient Sentiment Analysis system have been proposed in this project using Multinomial Naïve Bayes and the Linear SVC algorithms to perform the sentiment analysis of the tweets and the movie reviews. The accuracy found seems to be promising and the web service component gives the advantages to the third party applications to readily integrate the solution into their apps without having to do a lot of coding.

In Future, we extend our solution to perform the sentiment analysis of the images using facial expressions.

REFERENCES

[1]. Medhat, Walaa, Ahmed Hassan, and Hoda Korashy , "Sentiment analysis algorithms and applications"**,** 2014
[2]. Qu Hua, Shi Qundong, Jian Ding , "A Character-level Method for Text Classification", 2018
[3]. Yong-Quan Yang , "Research of Text Classification Based on Improved TF-IDF Algorithm", 2018
[4]. Yuxuan Tan , "An Improved KNN Text Classification Algorithm Based on K-medoids and Rough Set", 2019
[5]. Shaobo Du,Jing Li , "Parallel Processing of Improved KNN Text Classification Algorithm Based on Hadoop", 2019
[6]. Chenbin Li Guohua Zhan Zhihua L , "News Text Classification Based on Improved Bi-LSTM-CNN", 2019
[7]. Hossam Elzayady; Khaled M. Badran;Gouda I. Salama",Sentiment Analysis on Twitter Data using Apache Spark Framework", 2018
[8]. Jaishree Ranganathan, Allen S. Irudayaraj, Angelina A.Tzacheva, "Action Rules for Sentiment Analysis on Twitter Data Using Spark", 2017