RESEARCH ARTICLE OPEN ACCESS

# Analysis of cloud computing with Apache Spark and Machine Learning Algorithms for the Sentimental Analysis Technique

Gattu Prasad[1],Sandhya Rani Pabbathi[2] , K.Varalaxmi[3],Rapolu Manjula[4]

[1](Assistant Professor,,Sphoorthy Engineering CollegeNadargul Village,Saroornagar Mandal, Hyderabad, Telangana- 501510, India)
[2](Assistant Professor, , Sphoorthy Engineering CollegeNadargul Village,Saroornagar Mandal, Hyderabad, Telangana- 501510, India)
[3] (Assistant Professor, , NNRG College of Engineering, Korremula 'X' Road, Via Narapally, Chowdariguda (Vill), Ghatkesar (Mandal), Hyderabad. - 500088)
[4](Assistant Professor, ,Vignan's Institute of management and technology for women, Kondapurhatkesar, Medchal, Telangana 501301)

## Abstract:

The big data sector is rapidly expanding, making it challenging to manage and store massive amounts of data. This study offers a comprehensive approach to enhance big data analysis performance, including a large data storage environment and Apache Spark processing engine. The system tests sentiment analysis on 11 gigabytes of text data, using three machine learning techniques covered by the Spark ML package. The embedded model is composed of Scala and Java programs.
.

**Index Terms**: Big data, machine learning, Hadoop HDFS, Amazon datasets, Apache Spark, classification algorithms, and sentiment analysis.
.

### I.I NTRODUCTION

Big data, defined in 2005, refers to the vast amount of data generated by traditional business systems, internet/social networks, and the internet of things. As connected devices reach 100 billion by 2020, organizations face challenges in handling and managing this data. To overcome these issues, understanding big data analytics techniques and tools is crucial. Apache Spark, a fast engine for processing large-scale data, is faster than Apache Hadoop in iterative processing and can work across diverse nodes for parallel data processing.

Spark is a popular big data tool for handling large amounts of data, as it lacks its own data storage. Apache Hadoop, with its Hadoop distributed file system (HDFS), is the most suitable for handling large amounts of data. Spark offers a rich set of higher-level tools, including SQL, MLlib, Graph-X, and Spark Streaming, and supports programming languages like Scala, Java, Python, and R. The ML package in Spark includes various algorithms for classification [1], regression, clustering, collaborative filtering, and model evaluation. The Pipeline API simplifies development and runs multiple algorithms in a specific order.

### APACHE HADOOP AND APACHE SPARK

Large-scale distributed data processing and storage are the goals of Hadoop, an open-source Java programming framework. It allows large data to be stored and analyzed across several clients. A single host can be scaled up to thousands of hosts using Hadoop, which can also calculate storage requirements for each host. The four main components of the Hadoop framework are Hadoop Common, YARN, Map Reduce, and HDFS [2,3].
Hadoop is an open-source Java framework designed for distributed storage and analysis of large datasets. It can scale up an individual host to thousands and provides storage calculations for each. The framework consists of Metadata, offering great mistake tolerance and allowing for big data sizes. It is divided into one name-node and many data-nodes, with a file divided into blocks and stored in data-nodes.

Metadata [4], offering great mistake tolerance and allowing for big data sizes. It is divided into one name-node and many data-nodes, with a file divided into blocks and stored in data-nodes.

**Map Reduce**

Map Reduce is a method for processing a large dataset stored in Hadoop HDFS. However, it permits the parallel processing of an enormous dataset. The Map Reduce [5] algorithm consists of two significant tasks, known as Map and Reduce.

**YARN**

Yet another resource negotiator (YARN) is the Hadoop cluster resource manager which means it handles the Hadoop cluster resources like memory and CPU. Fortunately, versions 2 and 3 of Hadoop with Yarn opens a new door for data treating environment .

**Hadoop common**

This part consists of Java libraries and some facilities that are required by other Hadoop parts. These libraries provide level abstractions for the OS, files system and necessary Java libraries and some scripts are compulsory to initialize Hadoop [8].

**Apache Spark**

Apache Spark is an open-source platform for in-memory data processing, enabling fast processing of large data sizes using distributed memory. It caches data through multiple parallel operations, making it ideal for parallel processing of distributed data with iterative algorithms. Spark can run multi-threaded lightweight jobs within Java virtual machine processes, providing fast job start-up and parallel multi-core CPU utilization. It simplifies data pipeline management and is suitable for big data ML and graph algorithms. Spark is engineered for performance, being multiple times faster than Hadoop for massive data processing. It supports four programming environments: Java, Scala, Python, and R. Scala is particularly useful for supervised ML algorithms like regression and unsupervised ML algorithms like clustering.

**Spark distributed environment and cluster managers**

Spark is a distributed environment management system that supports four cluster manager types: standalone cluster mode, Hadoop Yarn, Apache Mesos, and Kubernetes. It operates in one node and multi-node environments, with the driver program being the heart of the job execution process. The cluster manager manages the application workflow allocated by the driver program to workers, controlling communication between the master node and slaves. Each worker node represents a container of one operation, with executors running multiple jobs. In the Spark distributed[7] environment, the driver program runs in its own Java process, communicating with distributed workers called executors. A Spark application is a combination of the driver andexecutors, running on a group of machines with the help of the used cluster manager.

**Spark data access and data structure**

Apache Spark offers a wide range of data access and storage options, including HDFS, Mesos, Mongo DB, Cassandra, H-Base, and Amazon S3, ensuring diversity in data reading and writing from various sources. Its data structure consists of three types: [18], [26], and [27].

a. Resilient distributed datasets (RDD): spark uses a particular data structure known as RDD which is a logical collection of data and separated over machines. RDD is Spark's primary abstraction, which is a fault- tolerant collection of elements that can be worked in parallel

b. Data frame (DF): it is a dataset organized into named columns or a collection of distributed records. DF is exactly such as RDD but, it is shaped into named columns with covering the characteristics of Spark SQL's execution. It is conceptually like a table in a relational database with better optimizations.

- Dataset: it is a distributed collection of data. Dataset is a new interface inserted in Spark 1.6 that offers the benefits ofRDDs[9] with the benefits of Spark SQL's optimized

## II. RELATED WORK

Recent research on big data processing using Hadoop and Spark has led to numerous advancements in machine learning techniques. Apache Spark MLlib 2.0 is an open-source, accessible, and efficient tool for analyzing attribute characteristics. It outperforms Weka in terms of performance and data handling efficiency, while Weka is better for simple users due to its GUI and diverse algorithms. Yan et al. discovered a micro blog sentiment classification scheme using paralleled support vector machines in Spark multi-node environments, improving accuracy through feature space evolution and parameter tuning. Apache[10] Spark's capabilities are fully utilized due to its large dataset. Al-Saqqa et al. found SVM to be better than other classifiers in performance.

.

Barznji et al. and Symeonidis et al. conducted sentiment analysis using ML algorithms like Naïve Bayes and SVM, using Apache Spark's large capabilities. They found SVM to be more accurate for total average. Symeonidis et al. tested pre-processing techniques in two datasets, comparing accuracy across four ML algorithms. They found some methods improved results, while others reduced accuracy, such as slang and spelling alteration.

.

### Dataset

This study analyzes sentiment or opinion on product-reviews data, focusing on three datasets from Amazon Reviews, a popular platform for interactive opinion analysis. The analysis considers the individual's feelings and attitudes towards a product, allowing customers to post comments, ask questions, and share their opinions. The focus is not limited to a specific review topic.

| No. | Name ofthe dataset | Size | No. of fields (attributes) | No. of rows |
|---|---|---|---|---|
| 1. | Amazon reviews: kindle store category | 685 MB | Nine | 982,899 |
| 2. | Amazon reviews for sentiment analysis | 1.6 GB | Two | 3,607,482 |
| 3. | Web data: Amazon movie reviews | 8.69 GB | Eight | 7,903,890 |
| 4. | Total size | 10.9 GB | Two | 12,494,271 |

Table 1.Datasets with their original size

## III. THE PROPOSED SYSTEM ARCHITECTURE

This work proposes a system using VMware Workstation software version 15.0.2, which handles Linux-Debian 9, 64-bits as a guest operating system. The host operating system is Windows 10, 64-bits. Big data tools like Hadoop and Spark are installed on the guest OS, allowing parallel data processing using three nodes: one master and two slaves. Spark is combined with Hadoop to read and write data from and to HDFS, enriching data processing capabilities. The workflow of the system includes feature selection, data cleaning, and data integration.

Feature selection involves a Java programming language program that reads the dataset and selects required columns. Data cleaning removes unwanted characters and commas, while data integration divides the dataset into three types: integrated datasets, datasets with equal number of comments, and normal size of datasets. The system architecture is explained in Figure 1.

The main program for data pre-processing and prediction was written in Scala using two approaches: central data processing (reading the integrated dataset from a local disk with one node) and distributed data processing (reading data from HDFS[12] within three nodes). The distributed approach reads all three types of data separately to evaluate their accuracy and performance. Data pre-processing.

| S.No | Name of the dataset | Size | No. of fields | No. of rows |
|---|---|---|---|---|
| 1. | n reviews: Kindle ategory | 570 MB | Two | 982,899 |
| 2. | Amazon reviewsfor sentiment analysis | 1.37 GB | Two | 3,607,482 |
| 3. | Web data: Amazon movie reviews | 6.4 GB | Two | 7,903,890 |
| 4. | Aggregation files (integration of the three datasets) | 8.35 | Two | 12,494,271 |

**Figure 1. The proposed system architecture**

Table 2 shows the characteristics of the datasets after the first three steps of data preprocessing [13] which are feature selection, data cleaning-1, and data integration.

## IV. RESULTS AND DISCUSSION

This study used three classification algorithms: logistic regression, SVM, and Naïve Bayes[14], with measures of accuracy, precision, recall, f-measure, and execution time

| Classi. algo. | Accuracy | Precision | Recall | F-meas. | Exec. time in min. |
|---|---|---|---|---|---|
| LR | 90.7% | 90.5 % | 90. 7% | 90.4% | 146.7 |
| SVM | 90.0% | 89.8 % | 90. 0% | 89.6% | 684.1 |
| NB | 80.8% | 84.9 % | 80. 8% | 81.9% | 145.8 |

approach

**Table 3. Central data processing results**

The datasets underwent manipulation to create three types of data: type (A), type (B), and type (C) for improved accuracy and performance. The system was tested with one large dataset file and multiple datasets of similar size. Results showed that all data types had similar results, as shown in

Stages included datacleaning-2, tokenizing, stop-words remover, and ineffective-words remover, followed by stemming and feature extraction to convert texts into vectors.

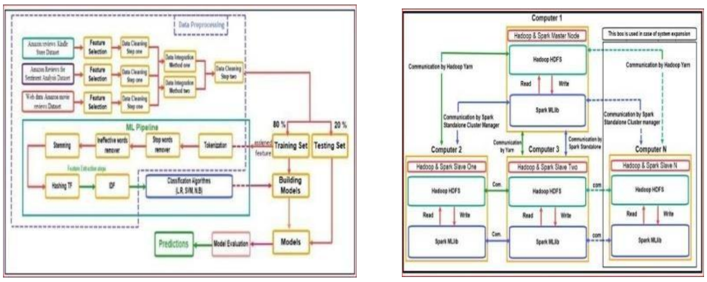

**Figure 2. Proposed system implementation steps**

The text describes a system for text classification using logistic regression, SVM, and Naïve Bayes algorithms. The system involves a pipeline for ordering procedures, data division into training and testing sets, and testing the model. The system also applies Hadoop HDFS and Spark distributed environment for parallel data processing across three nodes. The system is tested using positive and negative results. The system includes a screenshot of the Spark distributed web console monitoring.

computed. Results showed that logistic regression and SVM classifiers achieved excellent accuracy rates on training data, while Naïve Bayes had a good accuracy result. The study concluded that pre-processing steps significantly improved classification accuracy and execution time.

The datasets underwent manipulation to create three
Types of data: type (A), type (B), and type (C) for
Improved accuracy and performance. The system was
Tested with one large dataset file and multiple datasets
of similar size. Results showed that all data types had
Similar results, as shown in Figures 3 and 4.
Comparisons between them are necessary to check the best

Figures 3 and 4.After performing both approaches central and distributed data processing, now showing some brief

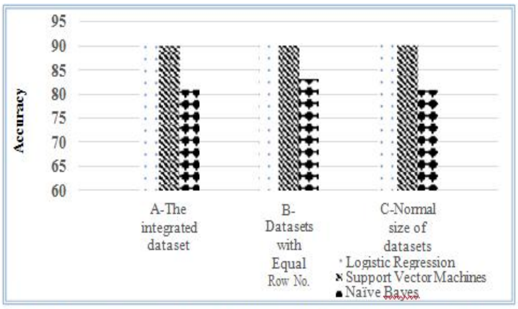

Figure 3. Accuracy comparison between utilized data type in approach two

Comparisons are made based on performance on time as well as previously acquired metrics like accuracy and F-measure. A comparison between central and distributed data processing will be made since the obtained time performance and other metrics for all the used data types (A, B, and C) in the distributed data processing are the same. The experimental findings demonstrate that, when utilising the distributed system technique instead of the central strategy , the learning times for all classifiers are roughly cut in half. The time required to generate the models in both the central and distributed types was another metric derived from all experiment testing. The least amount of time was needed for Naïve Bayes to finish model construction and logistic regression. On the other hand, even when the SVM method was used in a distributed fashion, its learning time was very long. It is clear that the logistic regression algorithm, followed by the SVM algorithm, attained the highest accuracy and F-measure rates in both central and distributed fashions. In contrast to the other classifiers, the Naïve Bayes algorithm yielded the worst results, as evidenced by its lowest accuracy and F-measure rates.

## V. CONCLUSION

This work aims to create a big data prototype system that combines distributed data storage and parallel data processing to handle any size of big data. Apache Hadoop and Spark are used to achieve these goals. The system was tested with 11 Gigabytes of big text data for sentiment analysis, collected from three Amazon customer reviews. The proposed pre-processing methods reduced the dataset size to 8.3 Gigabytes, resulting in faster algorithm execution and better accuracy. Two approaches, central data processing and distributed data processing, are used in the system. The distributed processing approach outperforms the central one-node approach, reducing execution time by half in all algorithms, while maintaining the same accuracy ratio.

## VI. REFERENCES

[1]. M. R. Wigan and R. Clarke, "Big Data's Big Unintended Consequences," IEEE Computer Society, vol. 46, no. 6, p. 46–53, 2013, doi: 10.1109/MC.2013.195.

[2]. P. Géczy, "Big Data Management: Relational Framework," Review of Business & Finance Studies, vol. 6, no. 3, pp. 21–30, 2015.

[3]. S. Alkatheri, S. Anwar Abbas, and M. A. Siddiqui, "A Comparative Study of Big Data Frameworks," International Journal of Computer Science and Information Security (IJCSIS), vol. 17, no. 1, pp. 66–73, January 2019.

[4]. CHARY, DR CH NARASIMHA, MOCHARLA RAMESH BABU, and S. KRISHNA MORE SADANANDAM. "Leveraging Deep Learning Techniques for the Stability Principles of Current Artificial Neural Networks Are Emerging Into Their Activation Functions." (2023)

.

[5]. VIJAYAJYOTHI, C., and D. SRINIVAS. "Abnormal Activity Recognition in Private Places Using Deep Learning..".." *International Journal of Computer Techniques* 10.2 (2023): 1-11..

[6]. S. Ra, B. Ganesh H.B., S. Kumar S, P. Poornachandran, Soman K.P., "Apache Spark a Big Data Analytics Platform for Smart Grid,"Procedia Technology, vol. 21, pp. 171–178, 2015, doi: 10.1016/j.protcy.2015.10.085.

[7]. Cholleti, Narasimhachary, And Tryambak Hirwarkar. "Biomedical Data Analysis In Predicting And Identification Cancer Disease Using Duo-Mining." *Advances In Mathematics: Scientific Journal* 9 (2020): 3487-3495

.

[8]. M. A. Khan, Md. R. Karim, and Y. Kim, "A Two-Stage Big Data Analytics Framework with Real World Applications Using Spark Machine Learning and Long Short-Term Memory Network," Symmetry, vol. 10, no. 10, pp. 1–19, 2018, doi: 10.3390/sym10100485.1235– 1241, 2016, doi: 10.5555/2946645.2946679.

[9]. Gupta, K. Gurnadha, Ch Narasimha Chary, And A. Krishna. "Study On Health Care Life Log By The Level Of Care Required Using Keygraph Technology In Text Data Mining

[10]. S. Harifi, E. Byagowi, and M. Khalilian, "Comparative Study of Apache Spark MLlib Clustering Algorithms," in International Conference on Data Mining and Big Data-Second International Conference, DMBD, pp. 61–73, 2017, doi: 10.1007/978-3-319- 61845-6_7

[11]. Ravi, Chinapaga, et al. "Analysis of Concept Drift Detection–A Framework for Categorical Time Evolving Data."

[12]. BHUSHAN, P. V., NITESH, V., CHARY, C. N., & GUPTA, K. G. Novel Approach for Multi Cancers Prediction system using Various Data Mining Techniques

[13]. X. Meng et al, "MLlib: Machine Learning in Apache Spark," Journal of Machine Learning.

[14]. O. Faker and E. Dogdu, "Intrusion Detection Using Big Data and Deep Learning Techniques," in ACM SE '19 Proceedings of the 2019 ACM Southeast Conference, pp. 86–93, 2019, doi: 10.1145/3299815.3314439.

.