'National Conference on Recent Advances of Computational Intelligence
Techniques in Science, Engineering and Technology'
International Journal of Scientific Research in Computer Science,
Engineering and Information Technology | ISSN : 2456-3307 (www.ijsrcseit.com)

Analysis of Parallel Data Processing for The Sentimental Analysis Technique Using Spark and Machine Learning Algorithms

Kesavan M V[1], Dr.Josephine Prem Kumar[2], Manimegalai A[1], Shammi L[1]
[1]Assistant Professor, [2]Professor
[1]Department of Computer Science, East Point College of Engineering and Technology, Bengaluru, Karnataka, India
[2]Department of Computer Science, Cambridge Institute of Technology, Bengaluru, Karnataka, India

ABSTRACT

The market for big data is now expanding quickly. Finding a system that can store and manage a massive amount of data, then analysing that huge amount of data to mine the hidden knowledge, is a major task. This research proposed a comprehensive system for enhancing the performance of large data analysis. Both a big data storage environment and a quick big data processing engine using Apache Spark are included. About 11 Gigabytes of text data, gathered from various sources, are tested by the system for sentiment analysis. The Spark ML package already supports this system, which uses three separate machine learning (ML) techniques. The built-in model comprises of system programs developed in the Java and Scala computer languages.

Keywords: Apache Spark, Classification Algorithms, Amazon Datasets, Sentiment analysis, Hadoop HDFS, Big data, Machine Learning

I. INTRODUCTION

to The term of big data was presented and defined firstly in 2005 by Wigan and Clarke [1] as an immense volume of data that cannot be handled by old data management styles. Traditional business systems, internet/social networks, and internet of things, are the main sources for generating big data. At the end of the year 2020, the number of connected devices will be approximately one hundred billion. This significant increase resulted in a situation that the data is extremely big, and establishments are facing obstacles with handling and managing it. These recently evolving obstacles and challenges with the aspects of increasing data have been referred to as big data problems [2]. Hence, to overcome these problems it's crucial to understand big data analytics techniques and tools. These techniques and tools are preparing good manners and capabilities to manipulate all types of data, instead of using traditional systems that support only a small database [3]. To fulfill the institution's requirements, an efficient tool is important to treat and manage immense data size. In this regard, Apache Spark has appeared as a fast engine for processing large-scale data inside the computer memory. Through its in-memory processing environment, a Spark has been proven to be much faster than Apache Hadoop, specifically in iterative processing like machine learning (ML) programs. However, it can work across a diversity of nodes for parallel data processing. It has presented a new style for data science where a wide range of data problems can be solved by a single processing engine [4]–[6]. Because Spark does not have its own data storage, well big data tool for handling a large amount of data is necessary. The most suitable tool for solving this problem is Apache Hadoop which consists of storagecalled Hadoop distributed file system(HDFS) [7].

The reason for the preference of Spark over the other big data tools is that the Spark ecosystem can offer a rich set of higher-level tools such as Spark SQL for SQL, MLlib for ML, Graph-X for graph processing and Spark Streaming for online processing. Also, Spark can support several programming languages which are Scala, Java, Python, and R [8]. ML package is the most interesting matter in Spark and consists of various types of ML algorithms such as classification, regression, clustering, collaborative filtering as well as model evaluation [9].

Supervised ML algorithms have been used in this study with Pipeline API in the Spark ML package. The Pipeline works to simplify the development and it is tuning multi-stage learning by providing a uniform set of high-level APIs [10]. In another term of meaning by using pipeline it can run a lot of algorithms according to a specific order. Therefore, it is useful for sequencing data pre-processing steps, which deliver cleaner and suitable data to the classifier algorithm. The dataset is passed through a group of preprocessing operations inside the ML pipeline and it is summarized in feature selection, data cleaning, data integration, tokenizing, stop word remover, stemming, and data representation [11]. Moreover, the whole procedure including Spark performance, Hadoop HDFS compatibility, and the effects of various utilized preprocessing steps have been tested through three types of algorithms which are logistic regression, support vector machine, and Naïve Bayes for binary opinion mining polarity (positive or negative) in the amazon product reviews.

II.APACHE HADOOP AND APACHE SPARK

Hadoop is an open-source structure written in Java programming language and designed for distributed storage and treating of very big datasets. It permits to store and analyze big data through multiple clients. Hadoop can scale up an individual's host to thousands of hosts and provide storage calculations for each one. Basically, the Hadoop framework isseparated into four parts HDFS, MapReduce, YARN, and Hadoop Common [12], [13].

HDFS

HDFS stands for Hadoop Distributed File System which divides the file systems into data and metadata. HDFS has two important benefits when compared with the traditional distributed file system. The first one is the great mistake tolerance and the second benefit it allows to use of big data sizes because the Hadoop clusters can remain data sets in petabytes [14]**.** The construction of HDFS is separated into one name-node and many data-nodes. A file in the HDFS is divided into a group of blocks and stored in the data-nodes. On the other hand the name-node is responsible for the block construction, termination, and replication [15].

MapReduce

MapReduce is a method for processing a large dataset stored in Hadoop HDFS. However, it permits the parallel processing of an enormous dataset. The MapReduce algorithm consists of two significant tasks, known as Map and Reduce[16].

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 [17].

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 [15].

Apache Spark

Apache Spark is an open-source platform and in-memory (RAM) data processing. Spark allows fast processing of a huge data size leveraging distributed memory. It caches the data through multiple parallel operations, making it especially fast for parallel processing of distributed data with iterative algorithms. Spark can run multi-threaded lightweight jobs within the Java virtual machine (JVM) processes, supplying fast job start-up and parallel multi-core CPU utilization [18]. The tasks inside Spark accomplish multiple processes consecutively, in memory, only spilling to the local disk when required by memory limitations. It simplifies the management of multiple operations by offering a data pipeline technique. Spark characteristics are very suitable for big data ML and graph algorithms [19].

In addition, Spark was engineered from the bottom-up for performance, it can be multiple times faster than Hadoop for massive data processing by using in-memory calculating and many other optimizations. Besides the reason of time performance, many reasons make Apache spark more suitable for big data analysis over Apache Hadoop such as Hadoop only uses Map and Reduce operation, while Spark use Map, Reduce, Join, and Sample. However, Spark supports four programming environments which are Java, Scala, Python, and R [20], [21]. Using Scala of Spark increases the speed computation of the algorithms and completes them in less time as compared to Java furthermore, the favorites of Scala noticed in supervised ML algorithms such as regression and unsupervised ML algorithms like clustering [22].

Spark distributed environment and cluster managers

Hopefully, Spark is working in one node environment as well as in the multi-node environment. Without any doubt every multi-node environment needs a cluster manager, Spark supports four cluster manager types which are standalone cluster mode, Hadoop Yarn, Apache Mesos, and Kubernetes [18]. Spark architecture based on the distributedenvironment has three primary elements which are [23],[24]:

a. Driver program: it is the heart of the Job execution process in Spark and this element represents the slave node in a Spark cluster. The driver runs the application code that creates RDD’s and then creates an object called Spark Contextthat administers and manages running applications

b. Cluster manager: this element is working for arranging the application workflow that allocated by the driver program to the workers. However, it operates all the resources in the cluster and brings back their situation to the driver program. In another word, the cluster manager is controlling the whole communication between the master node and the slaveswhen they run an application

c. Worker nodes: every worker node denotes a container of one operation throughout the Spark program execution. In another term of meaning, each worker-node has its executors and every executor will run several jobs.

In the Spark distributed environment, the driver program runs in its own Java process. These drivers communicate with a potentially huge number of distributed workers who are called also executors. Every executor is a single java process. A Spark application is a mixture of the driver and its executors. Spark application is running on a group of machines with the assistance of the utilized cluster manager [18].

Spark data access and data structure

One of the greatest advantages of Apache Spark is accessing/reading the data from multiple places such as HDFS, Mesos, Mongo DB, Cassandra, H-Base, and Amazon S3. Similarly, Spark can store/write the data in all the mentioned data storages which means it has a diversity of data reading and writing from various data sources [25]. Also, the data structure of Apache Spark fundamentally consists of three types of data which are [18], [26], [27]:

d. 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

e. 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 with the benefits of Spark SQL's optimized

III. RELATED WORK

A tremendous number of researches are printed on the subject of big data processing recently by utilizing Hadoop and Spark. There have been several approaches to analyzing big data. In this section, the focus only will be on the credible movements and contributions of this field. M. Assefi et al, 2017 [28] explored some views for growing the form of the Apache Spark MLlib 2.0 as an open-source, accessible and achieve ML tests that related to the real world to inspect the attribute characteristics. Also presented a comparison among spark and Weka with proving the advantages of spark over the Weka in many sides like the performance and it is efficient dealing with a huge amount of data. on the other hand, Weka is good for simple users with its GUI and the diversity of algorithms that already exist in it.

Yan *et al*. [29] discovered a micro blog sentiment classification scheme with paralleled support vector machine in the spark multi-node environment. They rose the accuracy by feature space evolution and tuning the parameters. Moreover, the execution speed is risen

with Apache Spark as well as to the speed of support vector machines (SVM) with radial basis function (RBF). However, the capability of Spark is completely utilized since the dataset is extremely large. Al-Saqqa *et al*.[30] discussed Spark's MLlib for hiring it in the classification of sentiment big data scale. They found that SVM is better than the other classifiers in the matter of performance.

Barznji *et al*. [31] talked about sentiment analysis utilizing the algorithms of ML such as Naïve Bayes and SVM for analyzing the text with the benefits of the huge capabilities of Apache Spark. They found that the SVM is more accurate in the condition of total average. Finally, Symeonidis *et al.* [32] tested some important pre-processing techniques and evaluated them in two datasets. Each method tested the accuracy in four ML algorithms. Furthermore, the study was implemented on all, as well as on the high-performance methods, in order to find method interactions. Their investigations present that a few methods supply better outcomes in both datasets utilized for Twitter sentiment analysis, while other methods reduce the accuracy suchas substituting slang and spelling alteration.

Dataset

Sentiment or opinion analysis of product-reviews data denotes the feeling and attitude of the individual on a product and it is available in online sources such as the Amazon product reviews site. In this work, three different datasets about Amazon Reviews have been used because it is the most wildly utilized for opinion analysis in an interactive way. Amazon Reviews is a platform, where customers can post comments on any product and ask/answer questions or share their own opinions. This work will not be limited to concentrating only on the topic of a particular review.

No.	Name of the 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

Three data reviews about different categories have been collected from the Kaggle Repository.

The first dataset is called Amazon reviews-Kindle store category, the second dataset called Amazon reviews for sentiment analysis and the third dataset called web data: Amazon movie reviews. Table 1 shows additional details on the datasets.

IV.THE PROPOSED SYSTEM ARCHITECTURE

The proposed system of this work is constructed on a VMware Workstation software version 15.0.2, which is used to handle Linux- Debian 9, 64-bits as a guest operating system (O.S). The host operating system of the VMware program is Windows 10, 64-bits. Besides, various big data tools have been installed on the guest OS such as Hadoop and Spark. Distributed environments for both Spark and Hadoop are used to gain parallel data processing utilizing three nodes: one master and two slaves (with the capability of expansion for future works). Then Spark needs to be combined with Hadoop to read and write data from and to HDFS. This combination will enrich the data processing capabilities. In other words, Hadoop works to distribute the data across multiple nodes to be seen by all the Spark nodes which work to process the data using in parallel (parallel data processing). In general, the proposed system architecture is explained in Figure 1.

The workflow of the proposed system is consisting of many steps starting from data collection and ending with model testing, all these steps are illustrated by the following four points:

a. Feature selection: a program was written in a java programming language to read the dataset and select the required columns (features) which are text and label from many unimportant columns.

b. Data cleaning-1: a program was written in a java programming language to clean the data by removing unwanted characters and commas within the text column and keeping the sole comma that separates between text and label for each comment.

c. Data integration: this step is divided into three types; each type will be processed separately in the upcoming steps. Type (A), integrated dataset: the three datasets are combined and saved in a single dataset file.

   Type (B), datasets with equal number of comments: A single dataset file (Type A) was divided into three new files with the same number of comments for each one of them.

   Type (C), normal size of datasets: loading and processing the original dataset files with their original size without any integration or division.

d. Data pre-processing and prediction: in this step, the main program for reading and processing data was written in the Scala programming language. Two approaches are used here: The first approach (central data processing) using for reading the integrated dataset (single dataset file-Type A) from the local disk with one node. While, the second one (distributed data processing) using for reading the data from HDFS within three nodes. The distributed approach read all the three types of data (Type A, Type B, and Type C) separately to evaluate the accuracy and performance of each one of them. Then applying data pre-processing stages: datacleaning-2, tokenizing, stop-words remover, and ineffective-words remover which works to remove many words that do not have any effects on the polarity type (positive or negative). After that using the stemming and finally feature extraction (hashing TF and IDF) for changing the texts into vectors.

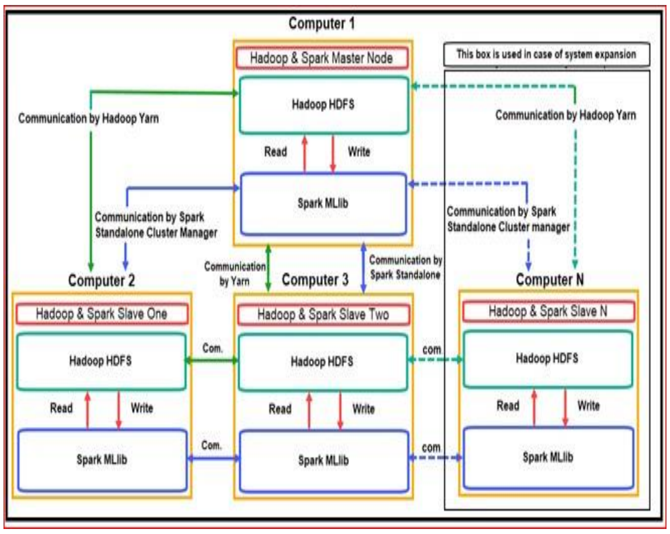

Computer 1
Hadoop & Spark Master Node
Hadoop HDFS
Read
Write
Spark MLlib
This box is used in case of system expansion
Communication by Hadoop Yarn
Communication by Hadoop Yarn
Communication by Spark Standalone Cluster Manager
Communication by Spark Standalone Cluster manager
Communication by Yarn
Communication by Spark Standalone
Computer 2
Computer 3
Computer N
Hadoop & Spark Slave One
Hadoop & Spark Slave Two
Hadoop & Spark Slave N
Com.
com

Figure 1. The proposed system architecture

Table 2 shows the characteristics of the datasets after the first three steps of data preprocessing which are feature selection, , data cleaning-1, and data integration. The whole procedures for the above four practical points are shown in

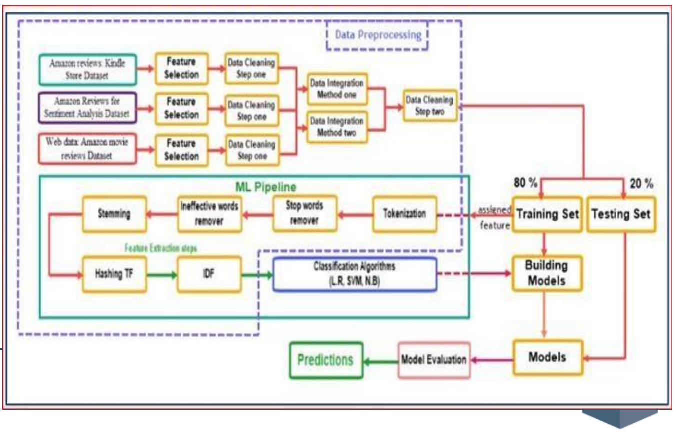
Table 2. Characteristics of the datasets after the first three steps of data preprocessing

Implementation

The Implementation of this work has been done in a very efficient manner which is convenient to use this system for analyzing any data size. The final utilized dataset size was reduced from 10.9 GB to 8.3 GB after the first two pre-processing steps as well the integration process. The implementation is divided into two approaches, approach one is central data processing (single node data processing) and approach two is distributed data processing (multi nodes data processing).

	Name of the dataset	Size	No. of fields	No. of rows
1.	n reviews: Kindle ategory	570 MB	Two	982,899
2.	Amazon reviews for 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 2. Proposed system implementation steps



moreover, utilizing three types of algorithms for classifying the texts which are logistic regression, SVM, and Naïve Bayes. All the implementation steps starting from pre-processing and ending with the classifiers must be executed according to a particular order. So, the pipeline is required for ordering procedures. After the pipeline stage, the data is divided into 80% for the Training set and 20% for the testing set. Finally, testing the constructed model based on the remaining 20% of data and then getting positive and negative results.

Figure 3 shows the proposed system implementation steps. Furthermore, in approach two, multi nodes data processing, the same implementation steps are applied with utilizing Hadoop HDFS for distributing data into blocks across the three nodes. As well As using Spark distributed environment for parallel data processing across the three slaves. Figure 4 shows the screenshot of Spark distributed web console monitoring.

IV. RESULTS AND DISCUSSION

As mentioned before, three classification algorithms were utilized in this work which are logistic regression, SVM, and Naïve Bayes. For each algorithm, the measures of accuracy, precision, recall, f-measure as well as execution time have been computed for both mentioned approaches.

Central data processing (approach one)

The results of all the mentioned measures as well as the comparison between the algorithms have been illustrated in Table 3. The outcomes of binary sentiment analysis using logistic regression and SVM classifiers obtained an excellent rate of accuracy on training data and Naïve Bayes accuracy result is very good. From these results, it can conclude that the utilized pre-processing steps produced a significant positive impact on classification accuracy and execution time.

Table 3. Central data processing results

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

Distributed data processing (approach two)

A few manipulations process on the datasets has been done. This manipulations process has resulted three types of data as mentioned before which are: type (A), (B), and (C). This manipulations process is to gain better accuracy and performance among all the utilized types. However, to check if the system is more interacting with one huge dataset file or with multiple datasets that have the same size as the huge file. The outcomes present that all the utilized data types have approximately the same measures result as shown in Figures 3 and 4.

After performing both approaches central and distributed data processing, now showing some brief comparisons between them are necessary to check the best approach.

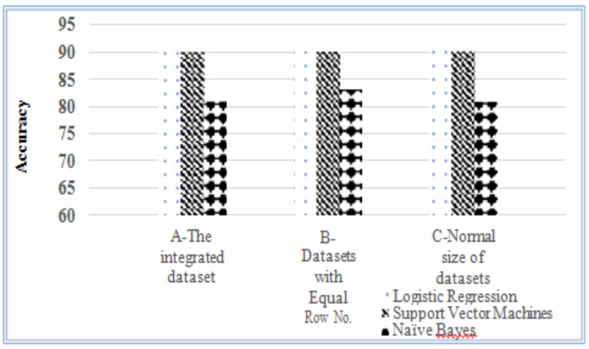


Figure 3. Accuracy comparison between utilized data type in approach two

The comparisons focus on-time performance as well as on the previously obtained measures such as accuracy and F-measure. Because the acquired time performance and the other measures for all the utilized data types (A, B, and C) in the distributed data processing are the same, the comparison will be done between Central data processing and Distributed data processing. The experimental results show that the learning times for all the classifiers are reduced approximately into half by using the distributed system approach as compared with the central approach.

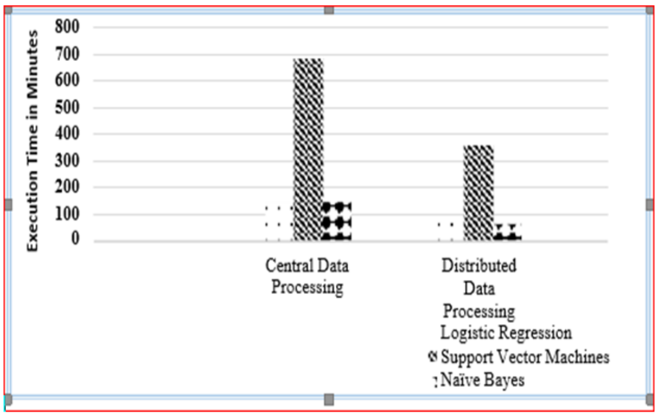


Figure 4. Execution time differences between the approaches

Another measure obtained from all the experiment tests was the amount of time taken for constructing the models in central and all the distributed types. Naïve Bayes required the smallest amount of time to complete the model building and then logistic regression. Whereas, the learning time for the SVM algorithm was very long even in a distributed manner. It is obvious that in both central and distributed manners the highest accuracy and F-measure rates achieved by logistic regression algorithm and then SVM algorithm.

Whereas the Naïve Bayes algorithm scored the lowest accuracy and F- measure rates, in other words, it has the worst results as compared to the other classifiers.

V. CONCLUSION

Undoubtedly big data is one of the major drivers of digitization. In the technical revolution, every single word/number might be useful information and has its benefits for the foundation's progress. The main goal of this work is to build a big data prototype system that consists of two crucial big data requirements which are distributed data storage for handling any size of big data as well as applying parallel data processing across that distributed storage for decreasing the execution time as much as possible. For gaining these two goals Apache Hadoop and Spark have been used in the system construction. The system has been tested with 11 Gigabytes of big text data for sentiment analysis. The datasets are collected from three different Amazon customer reviews and then aggregated in one huge file. Unlike small data, big data needs various data pre- processing steps to obtain good time performance and accuracy results. The proposed preprocessing methods of this work reduced the overall dataset size, only in the first two steps of the utilized pre- processing the size of the aggregated dataset file reduced to 8.3 Gigabytes which consists of about 12,494,271 review comments with binary labels (positive and negative). This reduction leads to performing and applying the algorithms in less time with gaining a better accuracy rate.

Two approaches are utilized in this system, central data processing(one-node) and distributed data processing (multi-node). Distributed processing approach outperformed the central processing approach and it is approximately reduced the execution-time of the model construction into half as compared with the central approach in all the utilized algorithms with acquiring the same accuracy ratio. The system programs have been written in Java and Scala programming languages and the constructed model consists of the classification algorithms as well as the preprocessing steps in a figure of ML Pipeline stages. Three types of algorithms have been used and the experiments indicated that the Logistic Regression out performed both SVM and Naïve Bayes classifiers in both utilized approaches. The final results showed that the system can treat a massive size of data with fast execution time especially in the distributed data processing approach. However, it can gain a very good accuracy depending on the various applied phases of the data pre-processing. In addition, the system tests were presented that the used manners of dataset manipulations such as separated datasets processing (multi-files) or integrated dataset processing (one-file) do not influence on the accuracy and time performance results.

II. 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]. S. Gopalani and R. Arora, "Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means," International Journal of Computer Applications (0975 – 8887), vol. 113, no. 1, pp. 8–11, March 2015, doi: 10.1.1.695.1639.

[5]. L. R. Nair, S. D. Shetty, and S. D. Shetty, "Applying spark based machine learning model on streaming big data for health status prediction," Computers and Electrical Engineering, vol. 65, pp. 393–399, 2017, doi: 110.1016/j.compeleceng.2017.03.009.

[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]. D. Graux, L. Jachiet, P. Genev`es, and N. Laya¨ıda, "SPARQLGX: Efficient Distributed Evaluation of SPARQL with Apache Spark," in The 15th International Semantic Web Conference - the Semantic Web – ISWC 2016. ISWC 2016, pp. 17–21 October 2016, doi: 10.1007/978-3-319-46547-0_9.

[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.

[9]. 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.

[10]. X. Meng et al, "MLlib: Machine Learning in Apache Spark," Journal of Machine Learning Research, vol. 17, no. 1, pp. 1235– 1241, 2016, doi: 10.5555/2946645.2946679.

[11]. 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.

[12]. "Apache Hadoop," Apache software foundation, [Online]. Available: https://hadoop.apache.org/. [Accessed 6-3-2019].

[13]. Y. Mjhool, A. H. Alhilali, and S. Al-Augby, "A proposed architecture of big educational data using Hadoop at the University of Kufa," International Journal of Electrical and Computer Engineering (IJECE), vol. 9, no. 6, pp. 4970– 4978, 2019, doi: 10.11591/ijece.v9i6.pp4970-4978.

[14]. T. Ruzgas, K. Jakubėlienė, and A. Buivytė, "Big Data Mining and Knowledge Discovery," Journal of Communications Technology, Electronics and Computer Science, no. 9, pp. 5–9, 2016, doi: 10.22385/jctecs.v9i0.134.

[15]. C. Kaushal and D. Koundal, "Recent trends in big data using Hadoop," International Journal of Informatics and Communication Technology (IJ-ICT), vol. 8, no. 1, pp. 39–49, April 2019, doi: 10.11591/ijict.v8i1.pp39-49.

[16]. A. Gupta, G. Aggarwal, and A. Kumar, "Hadoop, Map Reduce and HDFS-A Review," International Journal of Scientific & Engineering Research, vol. 8, no. 12, pp. 82–84, December 2017.

[17]. Y. Perwej, B. Kerim, M. Sirelkhtem, and O. E. Sheta, "An Empirical Exploration of the Yarn in Big Data,International Journal of Applied Information Systems (IJAIS), vol. 12, no. 9, pp. 19–29, December 2017, doi: 10.5120/ijais2017451730.

[18]. "Apache Spark," Apache software foundation, 24 Feb 2019. [Online]. Available: http://spark.apache.org/.

[19]. K. Wang and M. M. Hasan Khan, "Performance Prediction for Apache Spark Platform," in 2015 IEEE 17th International Conference on High Performance Computing and Communications (HPCC), pp. 166–173, 2015, doi: 10.1109/HPCC-CSS- ICESS.2015.246

[20]. J. P. Verma and A. Patel, "Comparison of MapReduce and Spark Programming Frameworks for Big Data Analytics on HDFS,"IJCSC, vol. 7, no. 2, pp. 80–84, 2016.Verma, A. H. Mansuri, and N. Jain, "Big data management processing with Hadoop MapReduce and spark technology: A comparison," Symposium on Colossal Data Analysis and Networking (CDAN), pp. 1–4, 2016, doi: 10.1109/CDAN.2016.7570891

[21]. H. K. Omar and A. K. Jumaa, "Big Data Analysis Using Apache Spark MLlib and Hadoop HDFS with Scala and Java," Kurdistan Journal of Applied Research (KJAR), vol. 4, no. 1, pp. 7–14, June 2019, doi: 10.24017/science.2019.1.2.

[22]. W. Inoubli, S. Aridhi, H. Mezni, M. Maddouri, and E. M. Nguifo, "An experimental survey on big data frameworks," Future Generation Computer Systems, vol. 86, pp. 546–564, 2018, doi: 10.1016/j.future.2018.04.032.

[23]. R. K. Chunduri and A. K. Cherukuri, "Scalable formal concept analysis algorithm for large datasets using Spark," Journal of Ambient Intelligence and Humanized Computing-Springer Berlin Heidelberg, vol. 10, pp. 4283–4303, 2018, doi: 10.1007/s12652-018-1105-8.

[24]. U. R. Pol, "Big Data Analysis: Comparision of Hadoop MapReduce and Apache Spark," International Journal of Engineering Science and Computing (IJESC), vol. 6, no. 6, pp. 6389–6391, 2016, doi: 10.4010/2016.1535.

[25]. J. Peloton, C. Arnault, and S. Plaszczynski, "FITS Data Source for Apache Spark," Computing and Software for Big Science- Springer, vol. 2, no. 7, pp. 1–9, October 2018, doi: 10.1007/s41781-018-0014-z.

[26]. Sassi, S. Anter, and A. Bekkhoucha, "A spark-based paraldistributed posterior decoding algorithm for big data hidden Markov models decoding problem," IAES International Journal of Artificial Intelligence (IJ-AI), vol. 10, no. 3, pp. 789–800, 2021, doi: 10.11591/ijai.v10.i3.pp789-800.

[27]. M. Assefi, E. Behravesh, G. Liu, and A. P. Tafti, "Big data machine learning using apache spark MLIib," in 2017 IEEE International Conference on Big Data , pp. 3492–3498, 2017, doi: 10.1109/BigData.2017.8258338.

[28]. B. Yan, Z. Yang, Y. Ren, X. Tan, and E. Liu, "Microblog Sentiment Classification using Parallel SVM in Apache Spark," in 2017 IEEE 6th International Congress on Big Data, pp. 282–288, 2017, doi: 10.1109/BigDataCongress.2017.43.

[29]. S. Al-Saqqaa, G. Al-Naymata, and A. Awajan, "A Large-Scale Sentiment Data Classification for Online Reviews Under Apache Spark," Procedia Computer Science, vol. 141, pp. 183–189, 2018, doi: 10.1016/j.procs.2018.10.166.

[30]. Al-Barznji and A. Atanassov, "BIG DATA SENTIMENT ANALYSIS USING MACHINE LEARNING ALGORITHMS," in Proceedings of 26th International Symposium "Control of Energy, Industrial and Ecological Systems, pp. 53–58, May 2018.

[31]. S. Symeonidis, D. Effrosynidis, and A. Arampatzis, "A comparative evaluation of pre-processing techniques and their interactionsfor Twitter sentiment analysis," Expert Systems with Applications, vol. 110, pp. 298–310, 2018, doi: 10.1016/j.eswa.2018.06.022.

[32]. B. Srigiriraju, "Amazon reviews: Kindle Store Category," [Online]. Available at: https://www.kaggle.com/bharadwaj6/kindle- reviews. [Accessed 22-7-2019].