International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Data Processing Framework Using Apache and Spark Technologies in Big Data

Muneeba Afzal Mukhdoomi, Er. Ankur Gupta

M.Tech Scholar, Department of Computer Sc. & Engineering, RIMT University, Mandi Gobindgarh, Punjab , Mandi Gobindgarh, Punjab
Asst. Professor, Department of Computer Sc. & Engineering, RIMT University, Mandi Gobindgarh, Punjab

Abstract

One of the most difficult problems in big data research is the inability to handle a significant amount of data in a reasonable amount of time. Hadoop and Spark are two distributed data processing frameworks. Hadoop is a widely used and versatile big data processing platform. Spark, being an open-source framework, is suitable for processing iterative data because of its in-memory programming paradigm. The runtime, memory and network utilisation, and central processor efficiency of and Spark frameworks, the big data processing platforms, are examined and compared in this study. As a result, both Hadoop and Spark frameworks implement the K-nearest neighbour (KNN) technique on datasets of various sizes. The results reveal that the KNN algorithm on Spark takes 4 to 5 seconds to run and is faster than Hadoop. Hadoop, according to tests, makes use of additional resources, including a central processor and a network. The conclusion is that Spark's CPU is more efficient than Hadoop's. Hadoop, on the other hand, consumes less memory than Spark.

1 INTRODUCTION

With the growth of technology and the Internet, the number of various searches has expanded substantially over the world. As a result, the amount of data generated throughout the world has increased considerably, giving rise to a new notion known as big data. A huge and complicated dataset that is impossible or difficult to process using typical information processing methods is referred to as big data. [1]. Data processing systems, as well as a variety of cluster computing frameworks, have been developed to accommodate large-scale data on commodity equipment [2]. Because it is scalable, dependable, and fault-tolerant, Google's MapReduce architecture, which was published in 2004, is one of the most popular frameworks for processing massive amounts of data.tolerance characteristics[3]. Apache Hadoop is a free and open-source MapReduce implementation. During execution, the MapReduce architecture does not keep track of data reused or the status of information. As a result, each epoch of MapReduce should read duplicate data and intermediate results from disc, making the operation expensive due to increased disc access, I/O, and wasteful compute operations. Hadoop isn't a good fit for[4].

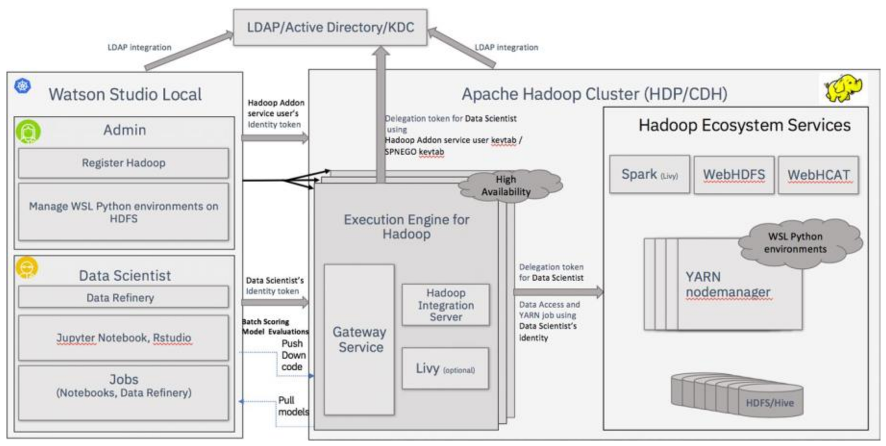
LDAP/Active Directory/KDC
LDAP integration
LDAP integration
Watson Studio Local
Admin
Register Hadoop
Manage WSL Python environments on HDFS
Data Scientist
Data Refinery
Jupyter Notebook, Rstudio
Jobs
(Notebooks, Data Refinery)
Hadoop Addon service user's Identity token
Data Scientist's Identity token
Batch Scoring Model Evaluations
Push Down code
Pull models
Apache Hadoop Cluster (HDP/CDH)
Delegation token for Data Scientist using Hadoop Addon service user keytab / SPNEGO keytab
High Availability
Execution Engine for Hadoop
Gateway Service
Hadoop Integration Server
Livy (optional)
Delegation token for Data Scientist
Data Access and YARN job using Data Scientist's identity
Hadoop Ecosystem Services
Spark (Livy)
WebHDFS
WebHCAT
WSL Python environments
YARN nodemanager
HDFS/Hive

Fig. 1.1 Architecture of Watson Studio Local with a Hadoop cluster by using the Execution Engine

A Spark is a Hadoop-like cluster computing technology that was created to address the Hadoop deficit in iterative operations. During the execution of repetitive processes, Spark introduced a data structure called resilient distributed dataset (RDD), which allows reused data and intermediate results to be stored in the memory of machines in the cluster. It has been established that this feature has effectively improved iterative tasks that require low latency [6]. Figure 1 represents the architecture of Watson Studio Local with a Hadoop cluster by using the Execution Engine for Apache Hadoop add-on from IBM.

Apache Hadoop add-on [6]

The term "big data," which has gained popularity in the recent decade, is typically used to refer to extremely massive or complex datasets that are difficult to examine using traditional technologies [7]. Despite the lack of a precise definition, big data is often used to describe the volume and diversity of datasets that call into question the analytical and processing capabilities of traditional methods [8]. A big data challenge is usually defined by a combination of the four Vs listed below:
• Volume: Businesses are accumulating ever-increasing amounts of data, including petabytes and, astonishingly, exabytes.
• Velocity: It is vital to assure schedules of detections and actions for procedures that are sensitive to time, such as capturing frauds.
• Variety: Big data can include text, sensor data, audio, video, and operator log files, among other things

• Statistical learning theory is a critical paradigm for establishing confidence and trust in big data and inferences or deductions based on large data. Machine learning is a potential data science technology that investigators have developed for presenting exact predictions using data. It is one of the artificial intelligence subfields that focuses on the concentrates on the construction of algorithms, which are able to forecast and learn from data The Apache Spark platform is depicted in Figure 2. ML is concerned with the subject of how to create a computer system that upgrades itself based on its experiences [9].

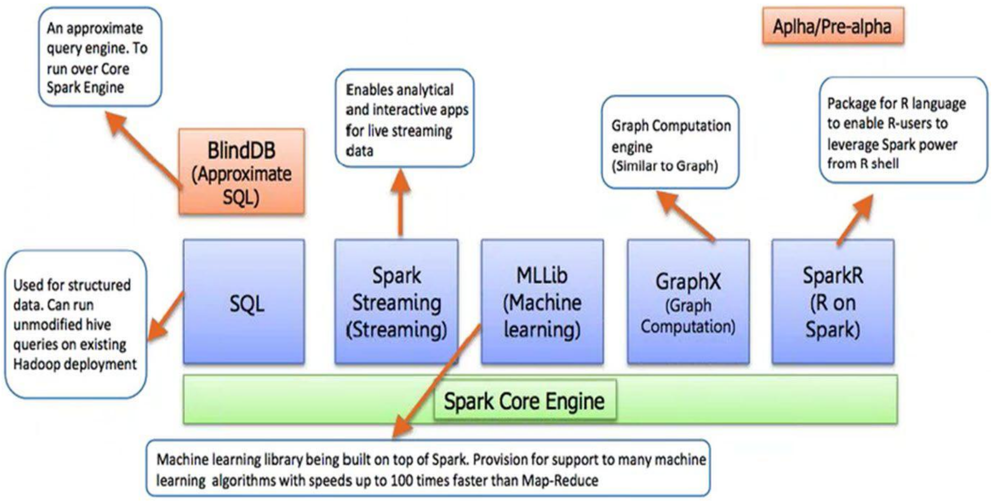
An approximate query engine. To run over Core Spark Engine
Aplha/Pre-alpha
Enables analytical and interactive apps for live streaming data
Graph Computation engine (Similar to Graph)
Package for R language to enable R-users to leverage Spark power from R shell
BlindDB (Approximate SQL)
Used for structured data. Can run unmodified hive queries on existing Hadoop deployment
SQL
Spark Streaming (Streaming)
MLLib (Machine learning)
GraphX (Graph Computation)
SparkR (R on Spark)
Spark Core Engine
Machine learning library being built on top of Spark. Provision for support to many machine learning algorithms with speeds up to 100 times faster than Map-Reduce

Fig. 1.1.1 Apache Spark platform [8]

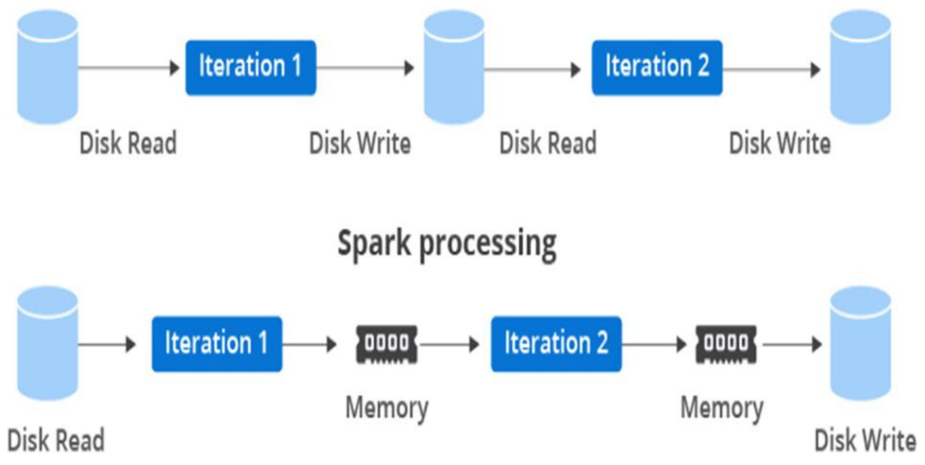

Fig. 1.1.2 MapReduce processing [10]

This research work's novelty can be characterised as follows: A complete experiment was conducted in this study to analyse and compare the performance of the Hadoop and Spark frameworks. As a result, the KNN method was tested on datasets of various sizes, as well as the runtime, memory, and processing power.

The frameworks' network consumption and CPU efficiency were examined. The following is the paper's structure: A review of prior efforts has been done in Sect. 2. Methodology is discussed in Sect. 3. Sect. 4 has reviewed and reviewed the results and debate. Sect. 5 provides a discussion of the conclusion.

2 RELATED WORK :

Hazarika et al. [11] investigated the theoretical differences and functional comparisons of the Spark and Hadoop platforms in 2017. Due to its repeat queries, such as logistic regression, their research shows that Spark is more faster for its cache. Spark's performance is also hampered by its small cache for non-repetitive queries. For modest repetitions from the Hadoop, though, it is much faster.

Gopalani et al. [12] examined the Hadoop and Spark platforms, two big data processing frameworks, in 2015. In other words, they used On a dataset containing sensor data, researchers used the Hadoop and Spark platforms to build the K-means algorithm, a basic machine learning approach, and compared the runtime of both frameworks. Spark outperformed Hadoop in runtime, according to the findings.

In 2013**, Li Guo et al**. [2] compared the memory use and runtime of Hadoop and Spark platforms. The PageRank method was used on numerous network datasets for this goal. Spark took less time to execute and used more memory than Hadoop, according to the findings.

In **2014, Wang et al**. [13] compared the runtimes of MapReduce and Spark. The C4.5 method, which is the core approach for generating decision trees, was used for this purpose on datasets of various sizes. According to the findings of this article, Spark is 950 percent more efficient than Hadoop when dealing with tiny datasets. Furthermore, when datasets are huge, S's efficiency suffers.

In **2010, Zaharia et al.** [5] Using the logistic regression methodology, we compared Hadoop and Spark frameworks. The author of this study focused on a specific application class that uses simultaneous multiple movement to reapply a running complicated of data. This includes interactive data analysis tools as well as a number of iterative machine learning techniques. Spark is a modern framework for securing those applications. This study has been proposed while acquiring the incorrect MapReduce tolerance and scalability. Spark introduces an abstraction called resilient distributed datasets (RDDs) to achieve these goals. An RDD is a read-only collection of objects partitioned among a set of machines that can be re-established if a segment or partition is lost, Spark outperforms Hadoop and can handle a 39 GB query dataset interactively with a sub-second response time.The results of this paper indicated the superiority of Spark.1 3 Investigating the performance of Hadoop and Spark platform

Mavridis and Karatza [14] did research on Hadoop and Spark with log file analysis, with the performance evaluation of log file analysis with Hadoop being a novel aspect of their study and Spark. In this paper, they have investigated log file analysis with the cloud computing frameworks Apache™ Hadoop ® and Apache Spark™. In both frameworks, the authors have improved the analysis of applications to realistic log file and in real Apache Web Server log files, SQL-type queries are carried out. Furthermore, with various parameters, they have led different experiments to compare and study the act and performance of the two structures and frameworks.

Kodali et al. [15] are working on a project that uses MapReduce to implement a k-NN-based technique. In this study, the authors employed the PathSim similarity measure in a Heterogeneous Information Network to categorise the meta-paths in order to locate the k-nearest neighbours using the well-known MapReduce methodology. They deciphered the classification approach that deals with it by utilising MapReduce.

A project on MapReduce performance model is performed by **Glushkova et al.** [16]. Through this research, there is a question and challenge of explaining MapReduce performance model for Hadoop 2.x. The suggested answer is focused on a present performance model for Hadoop 1.x, but keeping attention to obtain the implementation dream of both a MapReduce task and architectural changes via applying network queuing model. In this path, the synchronization intra-job constraints are reflected by the cost model because of the argument at shared sources. By comparison of our model calculations in contrast to assessment and measurements in a real Hadoop 2.x setup, the correctness of our answer is confirmed and validated

Wei et al. [17] conducted research on a massive dataset clustering technique based on MapReduce. The contrasts and similarities between the MapReduce execution of the Canopy algorithm and the K-means algorithm are explained in detail. The prospect of combining the two methods discussed above to create a superior algorithm suitable for big data clustering analysis is investigated in this study.

Jang et al. [18] proposed a study on input initialization for neural network inversion using the k-nearest neighbour method. This study presents a novel initialization strategy for neural network input variables based on the (k-NN) k-nearest neighbour method. The proposed method identifies inputs that resulted in an output next to a target output in a training dataset.

For big scale data that focused on kNN, Chen et al. [19] carried out swift peak of density clustering. The density with kNN-density is replaced by the suggested algorithm that is calculated by swift kNN algorithm like cover tree, giving enormous progress for the computations of density. Nonlocal density peaks and local density peaks are characterized based on kNN-density and a swift algorithm. For maximum cluster resource utilization, a research on optimum parallelism in Spark structure

Table.2.4: Summary of Related Work

Research Work	Approach	Strength	Limitation
Aziz K, Zaidouni D, Bellafkih M (2018)	Apache Spark is used for analysis of Twitter data. To ensure faster processing, it uses several types of processing like batch, iterative operations and streaming.	Apache Spark is much More advanced than MapReduce. It supports several requirements like real-time processing, batch and streaming.	Hadoop is relatively slow Hadoop MapReduce is relatively slower, because it is designed to do batch processing on a large amount of data and varied formats. Spark outperforms hadoop
Hazarika AV, Ram GJSR, Jain E (2017)	briefly discusses Spark and Hadoop architecture, their theoretical differences and the comparison of their performance**.**	Spark computational framework indeed outperforms Hadoop map reduce framework Because of the cache in memory storage, Spark is quite faster for iterative queries like logistic regression. As the cache memory is limited in size, the performance deteriorates with the no of iterations. Nevertheless for small iterations, it is quite faster than Hadoop.	In spite of the remarkable processing power hadoop still, has some shortcomings **.** Hadoop, a computation engine cannot solve some of the problems involving Iterative/Machine learning queries by caching some of the results from previous queries
Gopalani S, Arora R (2015)	compared hadoop and spark along with providing the performance analysis using a standard machine learning algorithm for clustering (K-Means).	Spark is a very strong contender. Observing Spark's ability to perform batch processing, streaming, and machine learning on the same cluster and looking at the current rate of adoption of Spark throughout the industry, Spark will be the de facto framework for a large number of use cases involving Big Data processing.	Hadoop is designed to do batch processing on a large amount of data and varied formats.
Liu, Francis(2013)	Page ranking algorithm	The results indicated that Spark had less execution time and more memory usage in comparison with Hadoop.	Hadoop is comparatively slow as compared to spark
Wang H, Wu B, Yang S, Wang B, Liu Y (2014)	C4.5 algorithm, the basic algorithm for building decision tree, was implemented on datasets with different sizes.	The results of this paper were that when the size of datasets is small, the efficiency of Spark is 95% better than the time when Hadoop is used. Additionally, when the size of datasets is large, the efficiency of Spark is 73% better than the time when Hadoop is used.	Hadoop takes more time to process big data sets

Matei Zaharia Scott Shenker, and Ion Stoica (2010)	compared Hadoop and Spark frameworks using logistic regression algorithm.	the author highlighted a specific applications class which reapplies a running complex of data through parallel multiple movement	Hadoop takes more time to process big data sets
Mavridis I, Karatza H (2017)	Ivestigation of log file analysis with the cloud computational frameworks ApacheTM Hadoop R and Apache SparkTM	The author developed realistic log file analysis applications in both frameworks and we performed SQL-type queries in real Apache Web Server log files	The overall performance shows that spark performs best processing of big data in very less time as compared to Hadoop
Glushkova D, Jovanovic P, Abelló A (2019)	k-NN-based method applying MapReduce	This model can be used for theoretically estimating of the jobs response time at a significantly lower cost than experimental evaluation of real setups. It can also be useful for critical decision making in workload management and resource capacity planning	There are several efficient approaches for modeling the performance of MapReduce workloads in Hadoop 1.x, they could not be applied to Hadoop 2.x due to fundamental architectural
Wei P, He F, Li L, Shang C, Li J (2020)	To analyze the time complexity of the algorithm, MapReduce framework based on canopy partitioning is implemented and filtering K-means	An improved K-means algorithm based on Canopy partitioning and filtering is proposed and this algorithm is implemented in the MapReduce technology framework. This algorithm has obvious performance improvement in terms of clustering accuracy and computational overhead.	The initial clustering center selection and iterative calculation are too large in the traditional K-means algorithm,.
Jang S, Jang YE, Kim YJ, Yu H (2020)	k-nearest neighbour method	The proposed method finds inputs which resulted in an output close to a target output in a training dataset, and combine them to form initial input variables. The proposed approach improved the simulation results with random initialization on a building environment dataset.	Hadoop takes more time to process big data sets

Chen Y, Hu X, Fan W, Shen L, Zhang Z, Liu X et al (2020)	Fast kNN algorithm such as cover tree, yielding huge improvement for density computations.	In order to improve DPeak and make it able to handle large scale data, FastDPeak is proposed to reduce the complexities of two main steps in DPeak, namely computing density ρ and δ. The overall complexity of FastDPeak is about O(nlog(n)) which outperforms naive DPeak and other variants of DPeak.	The initial clustering center selection and iterative calculation are too large in the traditional K-means algorithm

2 Methodology

a. MapReduce

Google launched the MapReduce concept in 2004, which is a programming approach for processing large datasets dispersed across numerous processors. Big data is often broken into smaller parts and processed in parallel in this model. The MapReduce system is divided into two parts: map and reduce. The map portion receives key–value data as input and generates intermediate key–v data as output [5]

b Hadoop

Hadoop is a free and open-source platform written in the Java programming language. Hadoop was inspired by Google's distributed computing publications and the GFS file system, which uses a simple programming approach to offer distributed processing on distributed datasets on linked computers. Doug Cutting built Hadoop to help with the distribution of the search engine project. Hadoop's scalability ranges from a single server to thousands of devices with local memory, with CPU being a key feature [20]. It can also detect and manage failures in the user layer, regardless of hardware, and so provide high-availability services to users. Hadoop is now employed in a variety of commercial applications, including those by Yahoo, IBM, Oracle, and Microsoft [4, 21]. It's critical to divide and conquer.

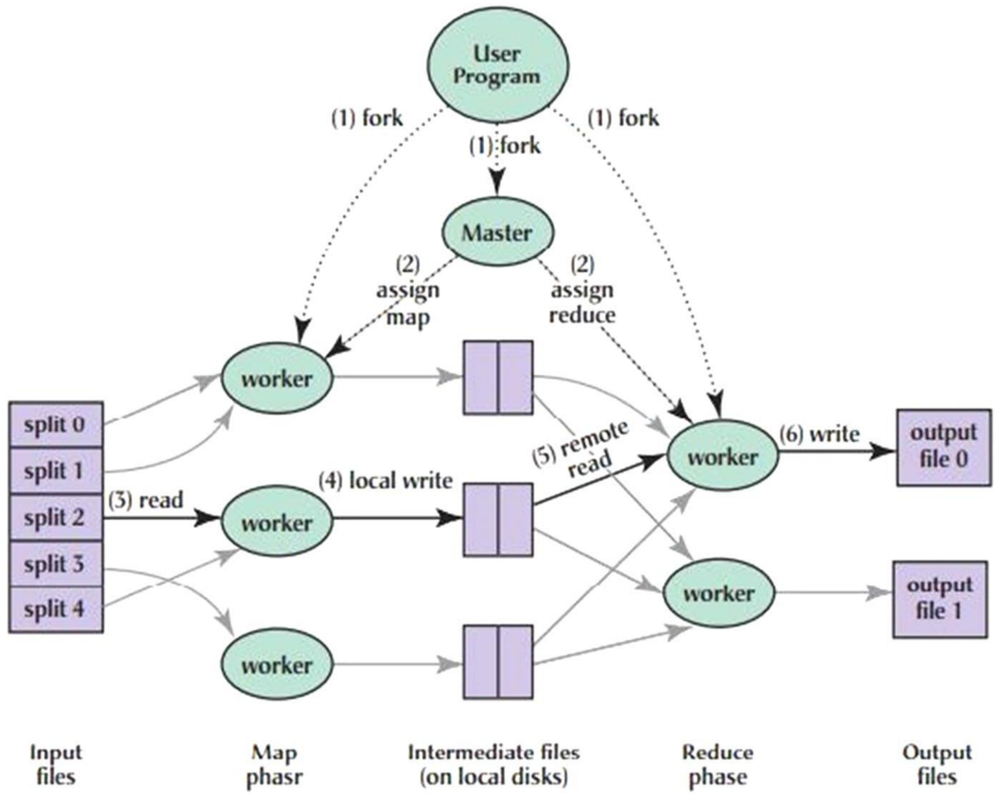

Fig. 5 How MapReduce model works [3]

C SPARK

Spark is an open-source framework for big data processing that aims to improve performance, usability, and complexity. This model was developed at Berkeley University in 2009 and was named one of Apache's projects in 2010. Machine learning iterative algorithms, interactive data analysis tools, and graph algorithms are examples of recursive programmes that should repeat a series of processes.
[23]. As a result, the Spark framework [5] was created to handle these programmes as well as scalability and fault tolerance in the MapReduce architecture. Developers must write a driver programme to use Spark, which initiates several processes in parallel and executes their application's high-level control stream. [24]. Spark introduced distributed datasets that are resilient (RDDs). RDD is a collection of read-only items that can be partitioned among machines and readily restored if the split is removed

4 ALGORITHM DESCRIPTION

In this section, we'll look at the K-nearest neighbour algorithm, which is one of the most significant machine learning techniques (KNN). KNN is a well-known classification method that has a wide range of applications in machine learning and

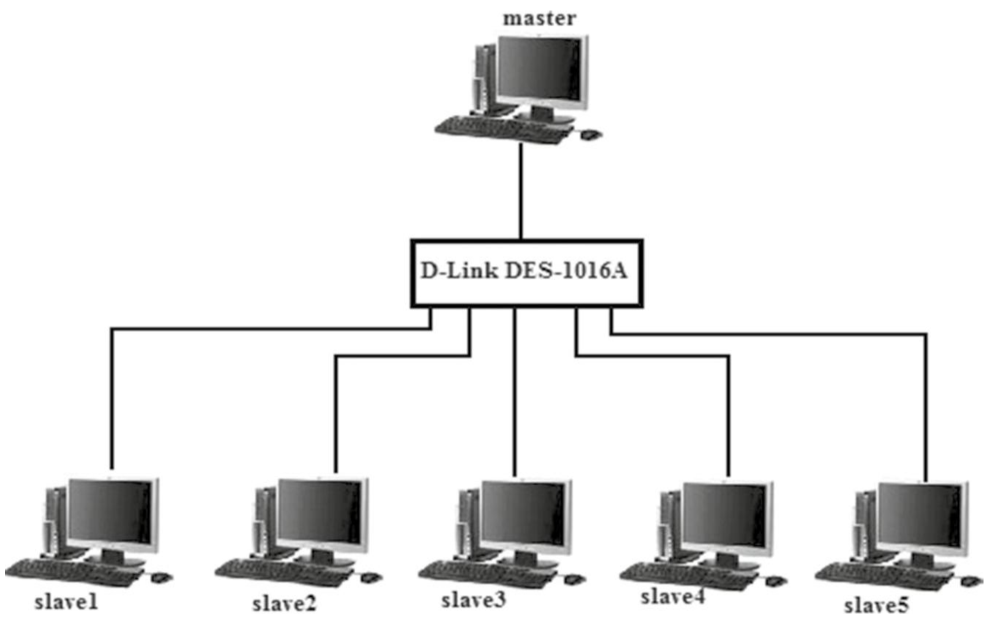

Fig. 6 An overview of laboratory cluster topology

Table 1

Dataset	**Name**	**Size**	**Number of samples**	**Number of features**	**Class label**
	Higgs1	20KB	50	2	2
	Higgs2	40KB	70	2	2
	Higgs3	60KB	90	2	2
	Higgs4	80KB	110	2	2
	Higgs5	100KB	130	2	2
	Higgs6	120KB	150	2	2
Higg	Higgs7	140KB	170	2	2
	Higgs8	160KB	190	2	2
	Higgs9	180KB	210	2	2
	Higgs10	200KB	230	2	2

data mining. KNN is a lazy algorithm because of the storage of input data and the lack of training phase. Also, given that no preconception is considered in input data, this method is a nonparametric method. KNN classifies a new example based on its similarity to training examples. In other words, KNN locates the unlabeled example's k-nearest neighbours and assigns the class label based on a majority vote of the k-nearest neighbours' labels.

4.1 The rule of the nearest neighbors

We'll look at how the KNN algorithm works in this part. We have a training dataset that is completely labelled. A new unlabeled sample is classified using the KNN algorithm. It finds the K-nearest neighbours of a new sample in the training dataset using an arbitrary distance metric, and then classifies the new sample using a majority vote of the k-nearest neighbours' class. The KNN does not create a model at the learning stage and merely uses training data to determine the label of fresh samples, making it simple to apply [25].

• The value of K: K is a user-defined constant. A majority vote of the class of a test sample's k nearest neighbours in the training dataset determines its label.

• Labelled dataset used as training data.

• A measure for measuring the distance between two samples that is appropriate: Several metrics can be employed in the KNN algorithm, however Euclidean distance is the most common.

5 Results and discussion

The results of the implementation of the KNN machine learning algorithm on Hadoop and Spark platforms are described in this section. On the Hadoop and Spark platforms, several studies were conducted to examine various characteristics like as runtime, memory use, CPU use, and network utilisation. On both the Higgs and the LHC datasets, the KNN method was implemented with K = 5 on the Higgs dataset of both the Hadoop and Spark platforms. The cluster arrangement in this test consists of six computers, one of which serves as the master and the others as slaves. Hadoop and Spark platforms have been compared and evaluated based on a number of factors including runtime, memory usage, CPU use, and network utilisation. Ganglia monitoring software [25] was used to record all parameters. In the next sections, the results are examined separately.

5.1 Comparison of the runtime

On the Higgs dataset, Figure 7 demonstrates the runtime of Hadoop and Spark platforms using the KNN machine learning algorithm. The graph shows that Spark has a faster runtime than Hadoop. Spark has a 40% advantage over Hadoop in terms of performance.

5.2 Comparison of the memory usage

Figure 8 shows memory usage of the Hadoop and Spark platforms using the KNN algorithm on the Higgs dataset. According to the findings, Spark uses several times more memory than Hadoop, indicating that Hadoop is superior to Spark. The in-memory programming style, which maintains all data and intermediate outcomes in memory, is to blame for Spark's increased memory utilisation. Figure 8 shows that the most Spark is used when the running time is between 120 and 660 KB. However, Hadoop is most useful when running times are between 100 and 570 KB.

5.3 Comparison of the CPU utilization

On the Higgs dataset, Figure 9 demonstrates CPU use on both the Hadoop and Spark platforms using the KNN algorithm. The results show that Hadoop's CPU utilisation is several times higher than Spark's, indicating that Spark is superior to Hadoop.

The quantity of CPU usage on two platforms is related to their runtime, implying that the amount of CPU usage and runtime have a direct relationship. On the other hand,

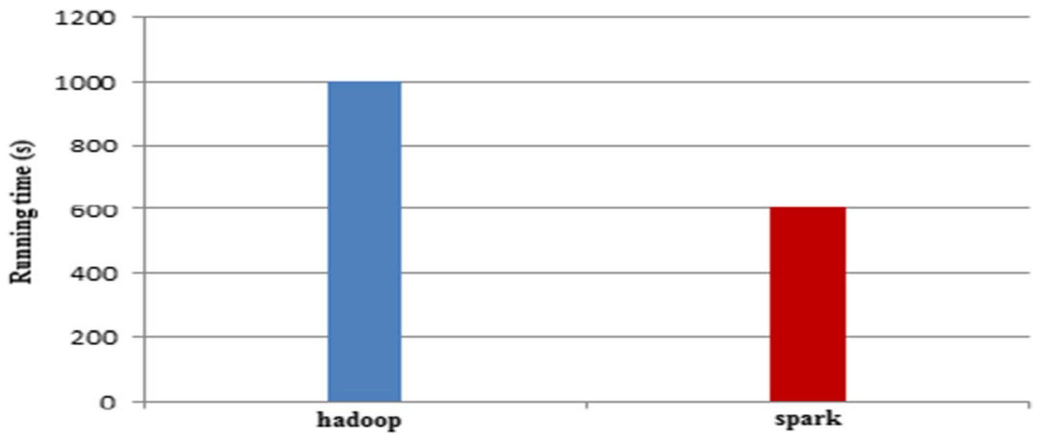

Fig. 7 Runtime comparison of the cluster on KNN algorithm

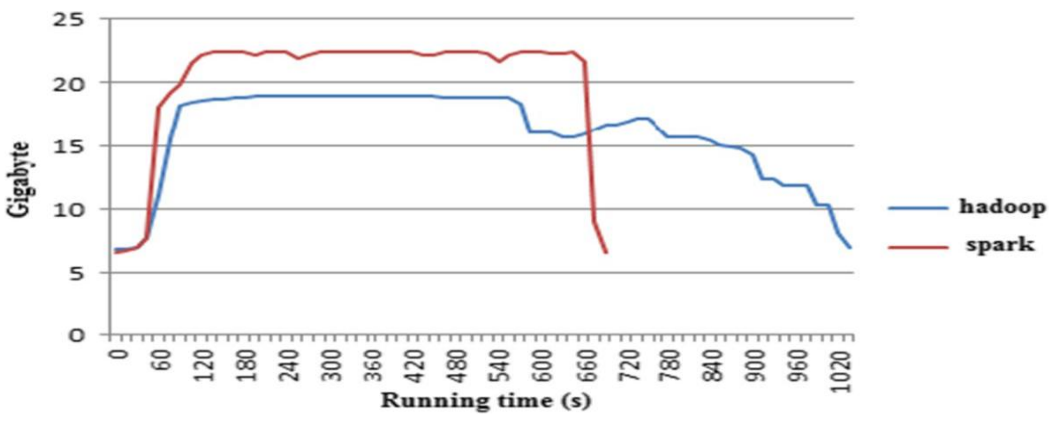

Fig. 8 Memory usage comparison of the cluster on KNN algorithm

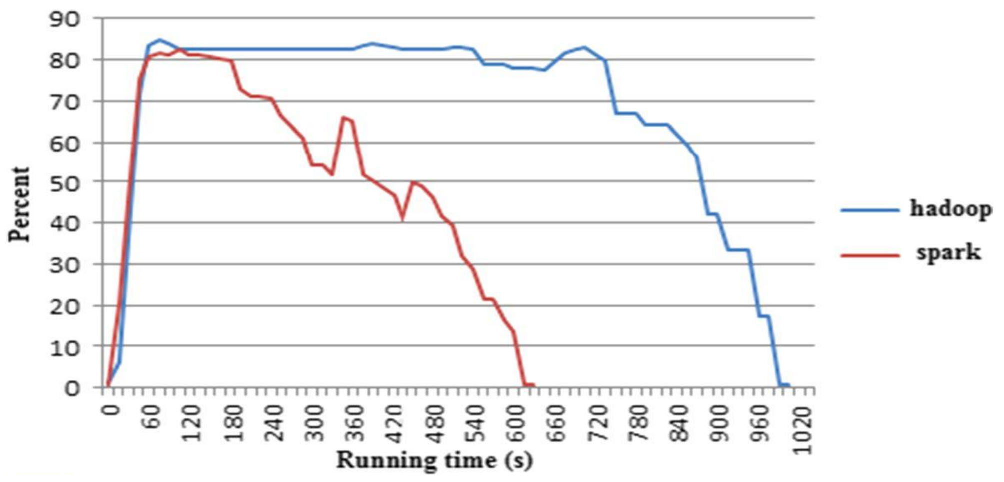

Fig. 9 CPU utilization comparison of the cluster on KNN algorithm

Because of its programming paradigm, Spark takes less time to run than Hadoop, resulting in reduced CPU use. On KNN, Figure 9 depicts a CPU utilisation comparison of clusters.For running times of between 60 and 120 Gigabytes, the maximum utilisation for Spark occurs. However, Hadoop's maximum capacity is between 60 and 720 Gigabytes.

5 Conclusions

The exponential growth and spread of information has become a singular phenomena. Analyzing and storing such a vast amount of data necessitates the development of new ideas capable of processing and managing such a big amount of data. Because of its enormous potential, big data has piqued the curiosity of professionals from all sectors. Many government entities have stated efforts to speed up big data and application research

The Spark and Hadoop frameworks were investigated in this article. Several tests employing the KNN algorithm were undertaken to evaluate the performance of various systems. The following results can be noticed given the structural aspects of Hadoop and Spark, as well as the KNN algorithm implementation:

• When the dataset is small, such as the Higgs10, Spark outperforms Hadoop, with Spark outperforming Hadoop by 4.5 to 5 times. The larger dataset, such as the Higgs, however, lowers Spark's advantage over Hadoop, which is 1.5 to 2 times more expensive.

• Hadoop isn't designed to handle tiny datasets. Spark, on the other hand, is well-suited to dealing with such data.

• Because Spark can retain data in memory, it's ideal for iterative computations. This reduces I/O of intermediate results, which accounts for a significant portion of Hadoop's lost time.

• Hadoop uses less memory than Spark in terms of memory use. Spark is generally quicker than Hadoop, although it consumes a lot of memory. If speed is required and memory is available, Spark is a good alternative; however, if sufficient memory is not available to store and retain data and interim results, Hadoop may be a better option.In the evaluation of the CPU and the network usage, Hadoop uses more these resources than Spark.

• Based on these findings, it can be stated that the Hadoop cluster's total network utilisation is higher than the Spark cluster's.

References

1. Chen M, Mao S, Liu Y (2014) Big data: a survey. Mob Netw Appl 19(2):171–209

2. Wu C, Zapevalova E, Chen Y, Zeng D, Liu F (2018) Optimal model of continuous knowledge transfer in the big data environment. Computr Model Eng Sci 116(1):89–107

3. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113

4. Tang Z, Jiang L, Yang L, Li K, Li K (2015) CRFs based parallel biomedical named entity recognition algorithm employing MapReduce framework. Clust Comput 18(2):493–505

5. Tang Z, Liu K, Xiao J, Yang L, Xiao Z (2017) A parallel k-means clustering algorithm based on redundance elimination and extreme points optimization employing MapReduce. Concurr Comput Pract Exp 29(20):e4109

6. Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauly M, Michael J, Franklin SS, Stoica I (2012) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: Presented as Part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12), pp 15–28

7. Cobb AN, Benjamin AJ, Huang ES, Kuo PC (2018) Big data: more than big data sets. Surgery 164(4):640–642

8. Qin SJ, Chiang LH (2019) Advances and opportunities in machine learning for process data analytics. Comput Chem Eng 126:465–473

9. Jordan MI, Mitchell TM (2015) Machine learning: trends, perspectives, and prospects. Science 349(6245):255–260

10. Russell SJ, Norvig P (2016) Artificial intelligence: a modern approach. Pearson Education Limited, Kuala Lumpur

11. Hazarika AV, Ram GJSR, Jain E (2017) Performance comparison of Hadoop and spark engine. In: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). IEEE, pp 671–674
12. Gopalani S, Arora R (2015) Comparing apache spark and map reduce with performance analysis using k-means. Int J Comput Appl 113(1):8–11
13. Wang H, Wu B, Yang S, Wang B, Liu Y (2014) Research of decision tree on yarn using mapreduce and Spark. In: Proceedings of the 2014 World Congress in Computer Science, Computer Engineering, and Applied Computing, pp 21–24
14. Mavridis I, Karatza H (2017) Performance evaluation of cloud-based log file analysis with Apache Hadoop and Apache Spark. J Syst Softw 125:133–151
15. Kodali S, Dabbiru M, Rao BT, Patnaik UKC (2019) A k-NN-based approach using MapReduce for meta-path classification in heterogeneous information networks. In: Soft Computing in Data Analytics. Springer, Singapore, pp 277–284
16. Glushkova D, Jovanovic P, Abelló A (2019) Mapreduce performance model for Hadoop 2.x. Inf Syst 79:32–43
17. Wei P, He F, Li L, Shang C, Li J (2020) Research on large data set clustering method based on MapReduce. Neural Comput Appl 32(1):93–99
18. Jang S, Jang YE, Kim YJ, Yu H (2020) Input initialization for inversion of neural networks using k-nearest neighbor approach. Inf Sci 519:229–242
19. Nguyen MC, Won H, Son S, Gil MS, Moon YS (2019) Prefetching-based metadata management in advanced multitenant Hadoop. J Supercomput 75(2):533–553
20. Javanmardi AK, Yaghoubyan SH, Bagherifard K et al (2020) A unit-based, cost-efficient scheduler for heterogeneous Hadoop systems. J Supercomput. https ://doi.org/10.1007/s1122 7-020-
03256 -4
21. Guo A, Jiang A, Lin J, Li X (2020) Data mining algorithms for bridge health monitoring: Kohonen clustering and LSTM prediction approaches. J Supercomput 76(2):932–947
22. Kang M, Lee J (2020) Effect of garbage collection in iterative algorithms on Spark: an experimental analysis. J Supercomput. https ://doi.org/10.1007/s1122 7-020-03150 -z
23. Xiao W, Hu J (2020) SWEclat: a frequent itemset mining algorithm over streaming data using Spark Streaming. J Supercomput. https ://doi.org/10.1007/s1122 7-020-03190 -5
24. Harrington P (2012) Machine learning in action. Manning Publications Co, New York
25. Lai WK, Chen YU, Wu TY, Obaidat MS (2014) Towards a framework for large-scale multimedia data storage and processing on Hadoop platform. J Supercomput 68(1):488–507