A Comprehensive Survey of MapReduce and its Applications: Advantages,Challenges, and Research Opportunities

1stFarha Naaz
Department of CSE
KBN Univesity
Kalaburagi, India
farhanaaz_12@rediffmail.com

2ndSameena Banu
Department of CSE
KBN Univesity
Kalaburagi, India
sameenabanu271@gmail.com

Abstract –
In the time of exponentially growing data, the effective management and processing of large datasets have become prevalent. The MapReduce framework, originally popularized by Google, has emerged as an essential technology for handling extensive data, offering parallel processing capabilities, load balancing, and fault tolerance. This survey paper presents a comprehensive examination of MapReduce, its diverse implementations, and its practical implications. While the significance of MapReduce is widely recognized, the existing literature lacks a thorough evaluation and comparison of various MapReduce implementations, particularly in real-world scenarios. Practical aspects, such as scalability, fault tolerance, and efficiency, remain relatively unexplored. Additionally, security and privacy concerns within MapReduce, including authorization frameworks and trust domains, require more extensive analysis. Furthermore, this paper explores the evolving field of MapReduce, encompassing open-source implementations like Hadoop, and highlights its application in diverse domains, ranging from telecommunication to pharmaceuticals. By addressing these research gaps, this survey paper aims to provide a comprehensive reference for researchers and practitioners navigating the complex terrain of big data processing and management.

Keywords - MapReduce Framework, Data Management, Hadoop, Big Data Processing.

I. INTRODUCTION

The idea that the rate of data increase follows an exponential trajectory is well supported by recent developments in the big data area [1]. Numerous academics have investigated fresh research opportunities in the field of big data across many disciplines as a result of this growth. The need to increase the effectiveness of MapReduce-based large-scale data processingsystems is now on the rise. It is impossible to overestimate the significance of improving resources and task scheduling since they are essential to helping applications meet their performance goals across a variety of usage situations [2]. The use of scheduling strategies is crucial for the optimization of large data processing. These methods are essential for reducingprocessing costs and execution times.

Furthermore, scheduling is not a new issue; the literature on distributed computing has extensively covered it. The MapReduce platform heavily relies on task scheduling, a crucial feature. The scheduling mechanism takes on the critical responsibility of distributing and controlling access to various resources within the context of MapReduce tasks. Process time, CPU use, communication channels, and bandwidth are some of the resources. The access made available makes it possible to complete tasks quickly and deliver the best possible service [3]. The scheduling issue must be effectively addressed, thus it is essential to carefully evaluate all available options and choose one that will work. The issue of task scheduling is intended to be addressed by the unique methods described in their paper. With an emphasis on increasing time and energy economy, the algorithm was created employing multi-threading concepts. Jayasena et al.'s work [4] proposed the idea of leveraging multimedia big data analytics for data distribution. The service layer, platform layer, and infrastructure layer make up the three tiers of the multimedia implementation system. The MapReduce architecture of the Hadoop distributed file system and the Xuggler media processing libraries were used in the design and implementation of the platform layers of the system. The method described above effectively shortens the time needed to transcode huge amounts of data into other formats. Furthermore, the ant colony optimization algorithm (ACO) is used to allocate resources inside the infrastructure layer efficiently. Based on the results, it can be concluded that the Ant Colony Optimization (ACO) method performs better than Particle Swarm Optimization (PSO) and Artificial Bee Colonies (ABC) at assigning virtual machines efficiently and quickly. The technology of MapReduce scheduling has been the subject of extensive investigation and examination in recent years. This study has examined the subject from a number of angles. The creation of novel approaches, optimization, and the installation of frameworks in varied situations have been the main areas of concentration in bigdata research [5]. Hadoop clusters' growing popularity is also due to its ability to effectively store and handle massive amounts of unstructured data in a distributed cluster setting. The proper management of the sizable volume of data processing in a distant cluster depends on the deployment of an effective scheduling approach. This study's main goal is to conduct a thorough evaluation of MapReduce scheduling in a sizable distributed system. In order to distribute workloads among various resources and ensure adherence to resource management limitations, the system uses a job and task distribution mechanism. The broad use of various MapReduce scheduling algorithms and the availability of resource scheduling frameworks are the two key variables that have a big influence on our attention. The primary goal [6] of this research is to provide a thorough resource for further efforts to create MapReduce scheduling algorithms. This project's main objective is to streamline the process of implementing cutting-edge scheduling frameworks and algorithms that are designed exclusively for big data analytics. A first presentation is made on the comparative examination of the Hadoop, Mesos, and Corona frameworks in order to accomplish the aforementioned objectives. A thorough examination of the current scheduling systems will be provided in this section. According to the degree of complexity required for their execution, the systems will be categorized. The defined categories, which include workload, speculative execution, scheduling strategy, resource needs, and optimization methods, are taken from the MapReduce scheduling framework [7]. In-depth study of scheduling in a distributed context will be provided in this part, with an emphasis on each unique category.

We'll further categorize the aforementioned categories into groups based on their core qualities. A thorough overview and a comparison table are also included for each area. An examination of the effectiveness tests carried out on various scheduling algorithms used inside the framework is provided in the paper. The future course of MapReduce scheduling will now be the subject of this debate. This document's goalis to provide an overview of the limitations of earlier scheduling system solutions in order to help researchers quickly identify these drawbacks. A number of investigations of the MapReduce architecture and its scheduling algorithms have recently been carried out. The survey [8-10] performed a thorough investigation of several MapReduce scheduling techniques, including an assessment of their inherent drawbacks. Furthermore, in [11] categorization system for MapReduce scheduling methods was suggested. The categorization is based on the necessary standards for quality, the parties involved, and the environment. Theextant literature offers insightful ideas, but it is deficient in complete information on the precisedevelopment procedures for each method. This paper provides a comprehensive analysis by looking at recent developments in different algorithms and talking about the difficulties and restrictions of existing approaches. Additionally, the papers named "Malleable scheduling for flows of jobs and applications to MapReduce" and "MapReduce scheduling for deadline- constrained jobs in heterogeneous cloud computing systems" [12] are seen to be pertinent in the current situation. [13] OEHadoop as an improvement to the current Hadoop framework with the aim of speeding up the traditional Hadoop framework's convergence time. The HadoopMapReduce pipeline has modified the traditional replication approach used in it. To improve the multicasting traffic architecture of the data center network, the optical multicast is being modified. The data center's inclusion of a software-defined network controller makes itpossible to modify the network architecture and streamline data transfer between the network and application layers. The experimental findings support the assertion that the suggested multicast request scheduling strategy performs better than more conventional methods. Additionally, OEHadoop has been found to display a greater rate of convergence than traditional Hadoop.

A. PROBLEM STATEMENT

The proliferation of emerging technologies, coupled with the exponential growth of data across various domains, has resulted in a diverse background of data management and processing approaches. In this context, there is a persistent need to comprehensively review and synthesize the current state of the art, trends, challenges, and best practices in data management and processing, spanning areas such as big data, machine learning, cloud computing, and database systems. This survey paper aims to address this critical gap by providing an in-depth analysis of the existing literature, identifying key research themes and gaps, and offering valuable insights for researchers, practitioners, and decision-makers in navigating the evolving data landscape and making informed decisions regarding data management and processing strategies.

II. RELATED WORK

The term "Big Data" is used to describe extensive collections of information that require specialized computer systems for analysis. The analysis of large volumes of data necessitates a significant amount of human effort. The task of analyzing large amounts of data poses a considerable challenge in the present era.

The utilization of the MapReduce framework has experienced a substantial rise in recent years. The aforementioned outcome is a direct consequence of the increasing demand for software applications capable of effectively handling large quantities of data. The MapReduce programming paradigm and its implementation aim to tackle the practical challenges associated with large-scale data production and manipulation[14]. A user-friendly parallel programming option has been introduced in the MapReduce paradigm. The system possesses the aforementioned characteristics, along with fault toleranceand load balancing capabilities [15]. Applications that frequently handle large datasets often utilize Google File System (GFS), a reliable and high-performance distributed data storage technology. The utilization of a MapReduce system is commonly observed in conjunction withthe system [16]. The existing primitives for map and reduce in functional languages were improved in order to develop the MapReduce framework. The current market offers a variety of options, including shared-memory multi-core systems, asymmetric multi-core processors, graphic processors, and clusters of networked workstations [17]. The facilitation and enhancement of large-scale distributed application development is achieved through the utilization of Google's MapReduce engine. The key attribute of the MapReduce paradigm is itsability to efficiently process large datasets that are distributed across multiple nodes in parallel[18]. The usage of Google's Map-Reduce software is restricted to the general public due to itsproprietary nature. In order to optimize distributed computing, it is essential to utilize the primitives Map and Reduce. In order to optimize performance, it is crucial for the underlying infrastructure to possess inherent complexity [19]. The distributed file system is a crucial element in Google's MapReduce design, as it guarantees the availability and proximity of data.The utilization of data parallelism across multiple computer nodes enables the advancement ofefficient distributed computing. Consequently, the performance of systems improves, achieving higher levels of optimization and reliability, enabling efficient processing of data at terabyte and petabyte scales. One approach to achieve this is by integrating a robust distributedfile system with the MapReduce programming paradigm. Based on the study's findings, it has been demonstrated that the MapReduce tool is highly reliable and efficient in optimizing data.This objective is achieved by reducing data access and loading times by over 50% [20]. The MapReduce method experienced a substantial surge in usage following its introduction by Google. The MapReduce technique [21] has garnered significant attention from the scientific community due to its capability to perform highly parallel data processing. The Hadoop framework, which is open-source, utilizes the MapReduce programming model. Additionally, it incorporates its proprietary Hadoop Distributed File System (HDFS). The necessity of the Google File System (GFS) has been found to be unnecessary. The continuous replication of HDFS data blocks results in the duplication of data. The system's multiple nodes are then provided with the copied blocks sequentially. The Hadoop Distributed File System (HDFS) operates in a manner that is comparable to other file systems. The system places a significant emphasis on achieving a high level of failure tolerance. It accomplishes this by effectively managing dispersed data blocks through processing them on the relevant nodes. The distributedfile system (DFS) can operate on standard personal computers and software, without the needfor any specialized hardware. When designing the solution, it is important to consider scalability issues. The platform neutrality of HDFS enables seamless transfer of data between computers with varying hardware and software configurations [22].

The development of Hadoop was driven by the widespread adoption of MapReduce, a highly successful open- source implementation. The implementation of MapReduce can be simplified by utilizing the open source Hadoop framework [23]. The design of the parallel programming system consistsof two primary components: A MapReduce engine and a user-level filesystem that manages the storage resources of the cluster. The two components are compatible with a wide range of operating systems, such as Linux, FreeBSD, Mac OS X, Solaris, and Windows. The software applications were developed utilizing the Java programming language. Optimal performance of the program can be achieved by leveraging optimization techniques when working in conjunction with commonly utilized hardware configurations.

The MapReduce programming paradigm is utilized by several frameworks. By utilizing the Map and Reduce features, the paradigm empowers end users to architect multiple concurrent processes. The foundational approach of this methodology is built upon the functional programming language Lisp. The management of Big Data challenges holds great importance. This approach alleviates concerns related to complex parallelism challenges such as fault tolerance, data propagation, and load balancing. The MapReduce system consists of two fundamental components: Map and Reduce. The Map function utilizes the argument of the Reduce function as an input for performing computations. The significance of this pattern in the sequential batch processing of big data is readily apparent [24]. The architectural arrangement enables the initiation of parallel processing by distributing map jobs across multiple nodes and simultaneously processing them. The processing is completed by merging the map outputs using the reduction function, resulting in the final results. The recent progress and growth of MapReduce, particularly its open-source version called "Hadoop," have led to the publication of numerous articles in reputable journals and conferences. The system provides a range of features including task and job scheduling, fault tolerance, cluster load balancing, job energy efficiency, security, performance, and flexibility. There exist several competing programming paradigms in the field of MapReduce. Both Spark and DataMPI are significant alternatives that should be considered. The primary focus of the study was on the MapReduce programming paradigm, commonly utilized by large enterprises for handling batch workloads. The selection of this option was based on its open source nature and robust performance capabilities [25]. The adoption of the MapReduce architecture as a viable approach for effectively handling large volumes of data has garnered significant attention. The MapReduce computing paradigm has gained popularity as the preferred technique for processing large volumes of data in parallel. The widespread adoption of MapReduce can be attributed to its successful implementation by Google. The subject of this discussion is a data processing system that is both highly scalable and fault-tolerant. This technology enables the concurrent processing of significant volumes of data by leveraging multiple low-end computer nodes. MapReduce has gained rapid popularity due to its simplicity, capacity to handle large-scale data processing, and exceptional fault tolerance properties. In addition, prominent academic and professional institutions are providing robust support for the endeavor. The process of processing can be partitioned into separate jobs, allowing them to be executed simultaneously on multiple cluster nodes in order to achieve optimal performance. In the initial phase of the MapReduce architecture, data is distributed across multiple computers using a distributed file system (DFS). The data is subsequently organized and presented in the form of (key, value) pairs. The MapReduce architecture is typically executed on a single master machine.

The primary responsibilities of this machine involve preprocessing incoming data prior to executingmap functions and postprocessing the output of reduction functions. Hadoop is a widely adopted implementation of the open-source MapReduce framework. The software is capable of executing multiple iterations of map and reduce operations, as determined by the provided parameters. It has been specifically designed to efficiently analyze large datasets. The utilization of a distributed user-level filesystem can enhance the management of storage resources within a cluster [26]. The system's performance is enhanced by increasing the number of computer nodes and expanding the dataset. When comparing the suggested method to traditional data mining and other processing methodologies, it demonstrates a reduction of 30% in execution time. The system exhibits a decrease in processing performance when handling smaller datasets.

The study conducted by [27] investigated the security and privacy features offered by the MapReduce architecture within a cloud environment. The partnership between MapReduce and the cloud is established through a direct connection. The utilization of the MapReduce architecture in public cloud environments enables customers to effectively and efficiently process large volumes of data. It is imperative to acknowledge that this deployment is deficient in robust security controls required to guarantee the confidentiality and integrity of computations and stored information. This method presents significant concerns regarding privacy and security. The inclusion of security-related components in MapReduce programs was also observed by the authors. This list comprises projects that focus on user consent, output verification, and auditing the confidentiality, integrity, and availability (ACIA) of the data computation pair. The evaluation of privacy issues involves assessing the ability of each participant to prevent unauthorized access by adversarial groups to data, codes, computations, and outputs, while also ensuring the security of the system. The authors failed to address two important security concerns: the requirement for specific MapReduce algorithms for data encryption, and the need for privacy restrictions within authorization frameworks and MapReduce trust domains.

In their study, [28] examined the application of MapReduce, a promising technology, across multiple sectors such as government agencies, manufacturing, the pharmaceutical industry, and communications. To effectively address and alleviate the current problems, this research integrates the strategies and solutions developed from 2006 to 2015. The present study employed multiple data sources, such as keywords, abstracts, titles, affiliations, citations, nations, and authorship, to perform a fundamental bibliometric analysis. This study also examined the most notable papers on the Scopus platform related to MapReduce. The compilation of papers presented in this collection delves extensively into various subjects related to MapReduce. This topic list encompasses declarative interfaces, data access, processing and transfer, iteration, resource allocation, and communication. The research assesses the benefits and drawbacks of these enhancements.

The study [29] encompassed an investigation into the foundational aspects of the MapReduce architecture, along with an analysis of its limitations and proposed enhancement approaches. Optimization techniques encompass various categories, including job scheduling optimization, the advancement of the MapReduce programming model, facilitation of real-time computation for stream data, enhancement of system hardware acceleration, performance tuning through configuration parameters, prioritization of energy savings as a significant cost consideration,

and augmentation of security via improved authentication and authorization mechanisms.
The analysis conducted by [30] focused on data-intensive processing. It explored multiple methods, discussing their respective benefits and drawbacks. The study also researched into the MapReduce programming paradigm and its implementation across different domains. There exist several alternatives to Hadoop for effectively managing large volumes of data, which include: In the TeraSort test, it has been observed that the Sector and Sphere systems exhibit superior performance compared to the Sphere system, particularly in terms of processing speed. DryadLINQ is a programming framework that enhances software development by integrating LINQ expressions with sequential programming, resulting in a streamlined process. The combination of Hadoop and Kepler in process
applications presents a user-centric architecture with exceptional performance. Table 1 shows the survey table.

Table 1 Survey Table

Aspect	**Contribution**	**Advantages**	**Research gap**
Big data	Big Data refers to extensive data collections requiring specialized computer systems for analysis. Analyzing large data poses significant challenges. MapReduce framework usage has surged due to	Provides scalable data processing. Simplicity and efficiency.	Lack of information on specific challenges in handling different types of big data.
MapReduce Framework	MapReduce offers parallel programming, simplicity, fault tolerance, and load balancing. - Google File System (GFS) complements MapReduce for data storage. Derived from functional languages, it has various implementations.	Enables distributed processing and fault tolerance. Compatible with different platforms.	Need for detailed performance comparisons between different MapReduce implementations.
Hadoop and HDFS	Hadoop is an open- source MapReduce implementation with its own Hadoop Distributed File System (HDFS). - HDFS replicates data for fault tolerance. - It runs on commodity hardware and is scalable and portable.	Open-source nature, scalability,and portability. High fault tolerance.	Lack of exploration of advanced Hadoop configurations and best practices.
Parallel Processing with MapReduce	MapReduce allows parallel processing of distributed data. - Google's MapReduce is proprietary but efficient. - It simplifies distributed computing with data parallelism. Achieves high performance via parallelism.	Efficient parallel processing and data parallelism. Improved system performance.	Limited discussion on the challenges of scaling MapReduce to extremely large datasets.

Map and Reduce Functions	MapReduce is inspired by Lisp and provides simplicity in expressing parallel processes. - The architecture consists of Map and Reduce functions. - Parallel processing is initiated by distributing map tasks across nodes.	Expressiveness in parallel processing. Dividing tasks across nodes for efficiency.	Insufficient exploration of advanced Map and Reduce functions beyond basic examples.
Popularity and Competing Frameworks	MapReduce's popularity is attributed to Google's success. - It is used for highly parallel data processing. Spark and DataMPI are competitors. MapReduce is chosen for its open-source nature and performance.	Open-source nature and high performance.	In-depth comparison between MapReduce and competing frameworks is needed.
Security and Privacy in MapReduce	The use of MapReduce in cloud environments raises security and privacy concerns. - Security-related projects address issues like authentication, authorization, and auditing. Privacy aspects include data protection and access control.	Addresses security and privacy challenges in cloud environments	Lack of focus on solutions for advanced security issues and privacy challenges in MapReduce.
Applications and Bibliometrics	MapReduce is applied across various domains, including telecommunications, manufacturing, pharmaceuticals, and government. - Bibliometric studies analyze keywords, abstracts, titles, affiliations, citations, and authorship.	Widely applicable in diverse industries. Bibliometrics offer insights and trends analysis.	Need for more specific case studies and real-world applications in various domains.
Optimization Methods for MapReduce	Optimization methods include job scheduling, programming model improvement, real- time support, hardware acceleration, configuration tuning, energy efficiency, and security enhancement.	Enhances performance, energy efficiency, and security.	Further research on the practical implementation of optimization methods and their impact.
Comparison with Alternatives	Alternatives to Hadoop include Sector, Sphere, DryadLINQ, and Kepler integration with Hadoop	Provides options for tailored solutions. Improved	Need for more extensive benchmarks and evaluations of

	for workflow applications. - Sector and Sphere excel in TeraSort performance. DryadLINQ simplifies programming. Kepler and Hadoop integrate well.	performance and simplified programming.	alternatives in various scenarios.

A. Research gap

1. **Limited Comprehensive Evaluation**: Existing research emphasizes the importance of MapReduce and its practicality for large-scale data processing. However, there is a notable gap in conducting a comprehensive evaluation and comparison of various MapReduce implementations. Researchers have yet to thoroughly examine and assess the practical implications, strengths, and weaknesses of these implementations.
2. **Practical Application and Efficiency**: While the significance of MapReduce is acknowledged, there is a need for in-depth investigations into how different MapReduce implementations perform in real-world scenarios. Practical aspects, such as their applicability to specific use cases, scalability, fault tolerance, and overall efficiency, remain underexplored.
3. **Security and Privacy Concerns**: While some research has addressed security and privacy challenges related to MapReduce in cloud environments, a comprehensive analysis is lacking. Specific issues, like authorization frameworks and trust domains, have not received sufficient attention in the literature. Bridging this gap requires more extensive exploration of security aspects within MapReduce.
4. **Evolution of MapReduce**: The research community can benefit from a deeper understanding of the evolving landscape of MapReduce, particularly in the context of its open-source implementations like Hadoop. Investigating new features, improvements, and challenges related to energy efficiency, fault tolerance, load balancing, security, performance, and elasticity would be valuable.
6. **Applications Beyond Google**: While Google popularized MapReduce, further research should explore its applicability in various domains, such as telecommunication, manufacturing, pharmaceuticals, and government organizations. Investigating how MapReduce is being used and improved in these diverse fields could uncover additional research opportunities.

III. CONCLUSION

In summary, this survey paper has researched into MapReduce, highlighting its essential role in managing and processing large datasets within the data-intensive landscape. While acknowledging its significance, we have identified a critical research gap that calls for further exploration. The comprehensive evaluation and comparison of diverse MapReduce implementations, particularly in real-world scenarios, remain relatively underexplored. As practical considerations like scalability, fault tolerance, and efficiency demand deeper examination, the security and privacy aspects of MapReduce, encompassing authorization frameworks and trust domains, require heightened attention. With MapReduce's application spanning various domains, this survey paper serves as a starting point for future research prospect, aiming to bridge these gaps and provide practical insights to harness the full potential of MapReduce in the ever-evolving realm of big data processing and management.

ACKNOWLEDGEMENT

I would like to express our sincere gratitude to all those who have supported and contributed to this research project. Primarily, I extend our heartfelt thanks to our guide for his unwavering guidance, invaluable insights, and encouragement throughout the research process. No funding is raised for this research.

REFERENCES

[1] B. Aragona and R. De Rosa, Big data in policy making, Math. Popul. Stud., vol. 26, no. 2, pp. 107– 113, 2019.

[2] G. Kaur, P. Tomar, and P. Singh, Design of cloud-based green IoT architecture for smart cities, in Internet of Things and Big Data Analytics Toward Next-Generation Intelligence, N. Dey, A. E. Hassanien, C. Bhatt, A. S. Ashour, and S. C. Satapathy, eds. Cham, Germany: Springer, 2018, pp. 315–333.

[3] A. Holst, Amount of information globally 2010–2024, https://www.statista.com/statistics/871513/worldwidedata-created/, 2020.

[4] Z. H. Sun, L. Z. Sun, and K. Strang, Big data analytics services for enhancing business intelligence, J. Comput. Inf. Syst., vol. 58, no. 2, pp. 162–169, 2018.

[5] A. Wibisono, P. Mursanto, J. Adibah, W. D. W. T. Bayu, M. I. Rizki, L. M. Hasani, and V. F. Ahli, Distance variable improvement of time-series big data stream evaluation, J. Big Data, vol. 7, no. 1, p. 85, 2020.

[6] B. N. Silva, M. Khan, and K. Han, Big data analytics embedded smart city architecture for performance enhancement through real-time data processing and decision-making, Wirel. Commun. Mob. Comput., vol. 2017, p. 9429676, 2017.

[7] T. Daghistani, H. AlGhamdi, R. Alshammari, and R. H. AlHazme, Predictors of outpatients' no-show: Big data analytics using apache spark, J . Big Data, vol. 7, p. 108, 2020.

[8] T. Nibareke and J. Laassiri, Using Big Data-machine learning models for diabetes prediction and flight delays analytics, J. Big Data, vol. 7, p. 78, 2020.

[9] N. Ahmed, A. L. C. Barczak, T. Susnjak, and M. A. Rashid, A comprehensive performance analysis of Apache Hadoop and Apache Spark for large scale data sets using HiBench, J . Big Data, vol. 7, no. 1, p. 110, 2020.

[10] Fonseca and B. Cabral, prototyping a GPGPU neural network for deep-learning big data analysis, Big Data Res., vol. 8, pp. 50–56, 2017.

[11] N. M. B. Rashid, M. Ahmed, L. F. Sikos, and P. HaskellDowland, Cooperative co-evolution for feature selection in Big Data with random feature grouping, J . Big Data, vol. 7, no. 1, p. 107, 2020.

[12] S. Srivastava, Top 10 countries & regions leading the big data adoption in 2019, https://www.analyticsinsight. net/top-10-countries-regions-leading-the-big-data-adoption-in-2019/, 2020.

[13] K. Nti, J. A. Quarcoo, J. Aning and G. K. Fosu, "A mini-review of machine learning in big data analytics: Applications, challenges, and prospects," in Big Data Mining and Analytics, vol. 5, no. 2, pp. 81-97, June 2022, doi: 10.26599/BDMA.2021.9020028.

[14] M. S. Mahmud, J. Z. Huang, R. Ruby, A. Ngueilbaye and K. Wu, "Approximate Clustering Ensemble Method for Big Data," in IEEE Transactions on Big Data, vol. 9, no. 4, pp. 1142-1155, 1 Aug. 2023, doi: 10.1109/TBDATA.2023.3255003.

[15] A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," in Big Data Mining and Analytics, vol. 5, no. 1, pp. 32-40, March 2022, doi: 10.26599/BDMA.2021.9020016.

[16] W. Zhong, N. Yu and C. Ai, "Applying big data based deep learning system to intrusion detection," in Big Data Mining and Analytics, vol. 3, no. 3, pp. 181-195, Sept. 2020, doi: 10.26599/BDMA.2020.9020003.

[17] T. Shang, F. Zhang, X. Chen, J. Liu and X. Lu, "Identity-Based Dynamic Data Auditing for Big Data Storage," in IEEE Transactions on Big Data, vol. 7, no. 6, pp. 913-921, 1 Dec. 2021, doi: 10.1109/TBDATA.2019.2941882.

[18] T. Shang, F. Zhang, X. Chen, J. Liu and X. Lu, "Identity-Based Dynamic Data Auditing for Big Data Storage," in IEEE Transactions on Big Data, vol. 7, no. 6, pp. 913-921, 1 Dec. 2021, doi: 10.1109/TBDATA.2019.2941882.

[19] L. Yan, W. Huang, L. Wang, S. Feng, Y. Peng and J. Peng, "Data-Enabled Digestive Medicine: A New Big Data Analytics Platform," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 18, no. 3, pp. 922-931, 1 May-June 2021, doi: 10.1109/TCBB.2019.2951555.

[20] W. Yu, Y. Liu, T. Dillon, W. Rahayu and F. Mostafa, "An Integrated Framework for Health State Monitoring in a Smart Factory Employing IoT and Big Data Techniques," in IEEE Internet of Things Journal, vol. 9, no. 3, pp. 2443-2454, 1 Feb.1, 2022, doi: 10.1109/JIOT.2021.3096637.

[21] A. R. Kulkarni, N. Kumar and K. R. Rao, "Efficacy of Bluetooth-Based Data Collection for Road Traffic Analysis and Visualization Using Big Data Analytics," in Big Data Mining and Analytics, vol. 6, no. 2, pp. 139-153, June 2023, doi: 10.26599/BDMA.2022.9020039.

[22] Z. A. Al-Sai, R. Abdullah and M. H. Husin, "Critical Success Factors for Big Data: A Systematic Literature Review," in IEEE Access, vol. 8, pp. 118940-118956, 2020, doi: 10.1109/ACCESS.2020.3005461.

[23] Gai, M. Qiu and H. Zhao, "Privacy-Preserving Data Encryption Strategy for Big Data in Mobile Cloud Computing," in IEEE Transactions on Big Data, vol. 7, no. 4, pp. 678-688, 1 Oct. 2021, doi: 10.1109/TBDATA.2017.2705807.

[24] X. Yang, R. Lu, K. K. R. Choo, F. Yin and X. Tang, "Achieving Efficient and Privacy-Preserving Cross- Domain Big Data Deduplication in Cloud," in IEEE Transactions on Big Data, vol. 8, no. 1, pp. 73-84, 1 Feb. 2022, doi: 10.1109/TBDATA.2017.2721444.

[25] S. Sarker, M. S. Arefin, M. Kowsher, T. Bhuiyan, P. K. Dhar and O. -J. Kwon, "A Comprehensive Review on Big Data for Industries: Challenges and Opportunities," in IEEE Access, vol. 11, pp. 744-769, 2023, doi: 10.1109/ACCESS.2022.3232526.

[26] M. S. Rahman and H. Reza, "A Systematic Review Towards Big Data Analytics in Social Media," in Big Data Mining and Analytics, vol. 5, no. 3, pp. 228-244, September 2022, doi: 10.26599/BDMA.2022.9020009.

[27] A. Pal, J. Wang, Y. Wu, K. Kant, Z. Liu and K. Sato, "Social Media Driven Big Data Analysis for Disaster Situation Awareness: A Tutorial," in IEEE Transactions on Big Data, vol. 9, no. 1, pp. 1-21, 1 Feb. 2023, doi: 10.1109/TBDATA.2022.3158431.

[28] B. Xu, K. Wang, Y. Sun, S. Guo and A. Y. Zomaya, "Redundancy Avoidance for Big Data in Data Centers: A Conventional Neural Network Approach," in IEEE Transactions on Network Science and Engineering, vol. 7, no. 1, pp. 104-114, 1 Jan.-March 2020, doi: 10.1109/TNSE.2018.2843326.

[29] A. B. Rawat, R. Doku and M. Garuba, "Cybersecurity in Big Data Era: From Securing Big Data to Data-Driven Security," in IEEE Transactions on Services Computing, vol. 14, no. 6, pp. 2055-2072, 1 Nov.-Dec. 2021, doi: 10.1109/TSC.2019.2907247.

[30] O. Sagi and L. Rokach, Ensemble learning: A survey, Data Mining and Knowledge Discovery, vol. 8, no. 4, pp. 1–18, 2018.