https://doi.org/10.51483/IJDSBDA.2.1.2022.1-9 ISSN: 2710-2599

SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Data Science and Big Data Analytics

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper **Open Access**

Big Data Processing Frameworks for Handling Huge Data Efficiencies and Challenges: A Survey

Kamal Al-Barznji[1*]

[1]Department of Computer Science, University of Raparin, Ranya, Kurdistan Region, Iraq. E-mail: kamal.barznji@uor.edu.krd

Article Info

Volume 2, Issue 1, May 2022
Received : 19 February 2022
Accepted : 22 April 2022
Published : 05 May 2022
doi: 10.51483/IJDSBDA.2.1.2022.1-9

Abstract

The increasing expansion of digital data collected from many sources renders traditional storage, processing, and analysis methods obsolete. For these restrictions, new technologies for processing and storing very massive datasets have been developed. Big data processing is required to extract relevant information from it. Transforming data into information and knowledge is what processing implies. Big data processing is the process of dealing with massive amounts of data and changing it from its raw form into useable information in a more understandable manner. As a result, numerous big data processing execution frameworks have emerged, but determining and selecting the appropriate framework for processing your big data applications is a significant challenge. Therefore, this paper investigates the possible influence of big data challenges and discusses in depth the most well-known approaches to big data processing, which are divided into five classes: batch processing, streaming processing, real-time processing, interactive processing, and hybrid processing, as well as the variety of the most popular frameworks associated with them such as Apache Hadood, Dryad, Samza, IBM Infosphere, Storm, Amazon Kinesis, Drill, Impala, Flink, and Spark. Furthermore, this study presents a comparison among the several features of the frameworks by highlighting their drawbacks and strengths. Thus, it can be used as a guideline for picking the best application framework in IT analytics and will help busi-ness users make faster decisions.

Keywords: *Big data, Challenges in big data, Big data processing, Big data frameworks*

1. Introduction

Data is assisting us in expanding into new areas, better serving current customers, streamlining processes, and generating raw and analyzed data. Organizations may now utilize structured, unstructured, and semi-structured data due to technological improvements. Tabular data is referred to as structured data which is found in relational databases or spreadsheets and accounts for only 5% of all available data (Gandomi and Haider, 2015). Text, images, music, video, social media, and e-commerce are examples of unstructured data. Semi-structured data formats do not comply with rigorous standards, Extensible Markup Language, e-mail, a textual language for transmitting data on the World Wide Web, is often used to describe semi-structured data (Manjula and Prema, 2020). Data has been created at an astounding rate from millions of data sources

* *Corresponding author: Kamal Al-Barznji, Department of Computer Science, University of Raparin, Ranya, Kurdistan Region, Iraq. E-mail: kamal.barznji@uor.edu.krd*

throughout the years. The production of massive amounts of data is possibly the most significant outcome of the digital revolution (Khalid and Yousaf, 2021). As indicated in Figure 1, the International Data Corporation in its white paper predicts that digital data will rise by approximately 80 trillion gigabytes (80 zettabytes) this year (2022) and will reach 175 zettabytes by 2025 (Reinsel *et al.*, 2018).

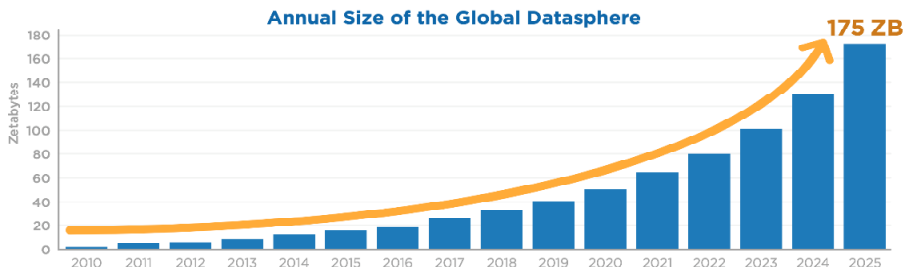

Figure 1: Annual Size of the Global Datasphere (Reinsel *et al.*, 2018)

Because of increasing new services such as the internet of things, cloud computing, and location-based applications, the era of big data has come (Zheng *et al.*, 2015). Big data is described as a vast volume of data that necessitates the development of new technologies and architectures to gain benefit from it by recording and analyzing the process. Big data is vital because the more data we gather, the more accurate results and the ability to optimize the business processes we will have. Big data is essential for both businesses and society. Businesses are mostly interested in unstructured data handling. Big Data has three primary features known as the 3Vs (Volume, Variety, and Velocity). Other companies and big data specialists (engineers, academics, etc.) have expanded these 3Vs to 5Vs by incorporating (Value and Veracity). Volume refers to vast volumes of any type of data from any source. Variety refers to the many sorts of data acquired by sensors, social networks, and cellphones, such as photographs, data logs, text, videos, audio, etc. Furthermore, these data might be structured or unstructured in type. The speed of data transmission is referred to as velocity. The process of collecting useful information from enormous quantities of social data is known as value extraction. The veracity of information relates to its completeness and accuracy. Therefore, we are unable of collecting, organizing, and analyzing a huge volume of data using our present data analysis software technologies (Al-Barznji and Atanassov, 2016).

Since data has become such an important resource, there has been a lot of discussion on how to effectively manage and exploit big data. How to analyze massive volumes of real-time data has become a major research and application problem. It should be highlighted. This study will focus on big data difficulties and different forms of big data processing, emphasizing the variances throughout the process and accessible frameworks. Therefore, the following is the structure of this paper: Section two discusses Big Data Challenges. Most approaches to big data processing and associated frameworks are presented in-depth in the third section. The fourth section (Discussion) presents a collection of common features and compares the frameworks across these features. Finally, the conclusion section summarizes the study and discusses future directions.

2. Challenges in Big Data

The issues of big data analytics are divided into five major groups: heterogeneity and incompleteness, storage and analysis of data, computational complexity, data scalability and visualization, and security and privacy. These challenges are briefly discussed in the subsections that follows.

2.1. Heterogeneity and Incompleteness

The data in the instance of sophisticated heterogeneous mixed data has various rules and patterns substantially. Structured and unstructured data are both possible. Organizations create 80% of their data in an unstructured format. It can take the shape of graphics, email attachments, documents, health data, images, audio, video, etc. They may not be saved as structured data in row/column format. This high degree of heterogeneity is a significant challenge for the next big data research (Al-Barznji and Atanassov, 2016; and Cuzzocrea and

Loria, 2021). Uncertainties arise from incomplete data during data analysis, which must be addressed. For certain samples, missing data field values are referred to as incomplete data. Missing values can be influenced by a lot of factors, including sensor node failure. To address these issues, numerous imputation methods are available (Al-Barznji and Atanassov, 2016).

2.2. Storage and Analysis of Data

Data size has increased tremendously in recent years due to numerous ways, such as, aerial sensory technologies, mobile devices, radio frequency identification readers, remote sensing, etc. These data are kept at great expense, only to be disregarded or erased in the end due to a lack of storage capacity. Phase Change Memory and Solid-State Drive were created to address the drawback of hard disk's poor input-output performance. Yet, the strategies outlined above are incapable of performing these massive data procedures. Certainly, Hadoop and MapReduce aid in gathering a large volume of unstructured data in a relatively short period (Acharjya, 2016).

2.3. Scalability and Visualization of Data

As data volumes rise quicker than CPU performance, there is indeed a natural significant shift in processing techniques, with growing multiple cores being added. The goal of data visualization is to convey data more effectively using graph theory approaches. Every month, online marketplaces such as Amazon, and eBay have millions of customers and billions of products for sale. This produces a large amount of data. To that aim, several businesses employ the Tableau application for massive data visualization. It can convert enormous amounts of complicated data into simple images. However, today's large data visualization solutions typically perform poorly in terms of functionality, scalability, and reaction time. To address this issue, more mathematical models should be linked to computer science (Acharjya, 2016).

2.4. Computational Complexities

Representation and knowledge discovery are critical issues that necessitate sub-fields. To handle problems and / or requests, many combination strategies are employed. As the volume of big data grows, these strategies are inefficient for obtaining significant information. Massive datasets may be managed through data marts and data warehouses. Large datasets necessitate more computational difficulties. Although specialized data related to a certain topic can be used to comprehend complexity. One of the primary goals of the research is to reduce complications and processing costs (Shikha and Jimmy , 2018).

2.5. Privacy and Security

Security is one of the big data difficulties for several reasons. For starters, the big data framework comprises several distinct data formats, each with a particular need to be safe, and security cannot be guaranteed. Second, parallel data processing presents a new difficulty in which we must ensure data security. The third, difficulty stems from real-time analytics: how does the framework protect privacy while doing real-time analysis? Additionally, huge data are kept as distributed files in the cloud, making security increasingly complex. Data backup in big data systems has revealed issues of security, and the production of several replicas puts data in danger, and regulations specify the data that is stored, processed, and analyzed, but there is no guarantee that this data will be saved (Abuqabita *et al.*, 2019).

3. Big Data Processing Frameworks

Big Data processing is the process of dealing with massive amounts of data and converting it from its raw form into a useful approach and a more understandable manner (Benjelloun *et al.*, 2020). This section highlights the most powerful frameworks used to manage massive amounts of quickly generated data. These frameworks are often grouped into five classes and structured as follows based on their data processing approaches: Figure 2 shows batch processing, streaming processing, real-time processing, interactive processing, and hybrid processing.

3.1.Batch Processing

When data is collected or kept in big files, batch processing is utilized (Saadoon *et al.*, 2022). Batch processing is the processing of large data blocks that have been previously recorded in a database (Abuqabita *et al.*, 2019). A batch processing framework needs the collection of data over time and the loading of all data required for the

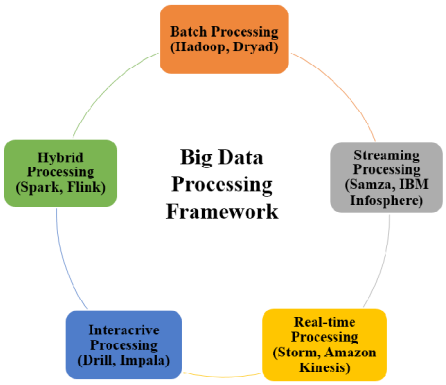

Figure 2: Big Data Processing Frameworks Classification

batch into some kind of storage, such as a file system or database, to be processed. When working with huge amounts of data, batch processing is frequently employed (Cumbane and Gidófalvi, 2019). The most famous frameworks for this type of processing are Apache Hadoop and Apache Dryad Frameworks.

3.1.1. Hadoop Framework

Today, Apache Hadoop is the most popular batch data processing framework. Apache Hadoop is a free and open-source Java framework for processing and querying large volumes of data on commodity hardware clusters. Yahoo! has made enormous technological investments. Apache Hadoop evolved into an enterprise-ready cloud computing platform in 2006. Its influence may be summed up in four key features. Hadoop delivers scalable, cost-effective, adaptable, and fault-tolerant systems. Hadoop is made up of two major components: HDFS which is Hadoop Distributed File System and the MapReduce programming framework. As a result, the storage system is not physically isolated from the processing system (Al-Barznji and Atanassov, 2016; and Otoo-Arthur and Zyl, 2020). "Google's MapReduce and Google File System" were created in 2004 in response to the ever-increasing volume of data on the web. MapReduce is the first and native batch processing programming method (engine) of Hadoop. It is intended for parallel processing of huge amounts of data by separating the work into many distinct jobs. Also, Yet Another Resource Negotiator (YARN), was released in 2012 by Yahoo! and Hortonworks (Khalid and Yousaf, 2021; Benjelloun *et al.*, 2020; and Cumbane and Gidófalvi, 2019). The Hadoop ecosystem also includes the Hadoop kernel, as well as other components: HBase, Apache Hive, Oozie, Zookeeper, Pig, etc. (Al-Barznji and Atanassov, 2016).

3.1.2. Dryad Framework

Apache Dryad is a parallel and distributed processing framework that was initiated by Microsoft in 2004. It is a powerful module that can improve processing capacity and grow from a small cluster to a bigger one. This framework enables users to access a cluster's resources for parallel data processing. Dryad is a highly sophisticated framework that includes entire tasks such as job creation, monitoring, and management, visualization, resource management, and fault tolerance (Abuqabita *et al.*, 2019).

3.2. Streaming Processing

Stream processing is utilized when data has to be analyzed as quickly as it comes, whether it's social data or machine data from IoT systems. The goal of using stream-based processing is to achieve: low latency, and real-time reaction to a new event (Benjelloun *et al.*, 2020). In short, streaming data processing indicates that the data will be examined and activities will be made on the data as soon as possible, usually in near real-time. IBM Infosphere Streams and Apache Samza are the most well-known frameworks for this type of processing.

3.2.1. IBM Infosphere Streams

It is an IBM framework designed to handle unlimited streams of data at high speeds. It can handle both unstructured and structured data streams and can be expanded to a high number of nodes. IBM streams are capable of processing complicated data streams at a fast pace and with extremely low latency. It comes with a

stream process language that allows users to construct stream applications using a high-level programming language (Abuqabita *et al.*, 2019).

3.2.2. Apache Samza

In 2013, Samza was created by LinkedIn and contributed to the Apache Software Community later that year. Samza is designed to accommodate large data stream throughput (millions of messages per second) while also delivering rapid fault recovery and great dependability. Samza is now used by many large corporations, including LinkedIn, Uber, VMware, TripAdvisor, and Netflix (Manjula and Prema, 2020). Samza's vision is to create a lightweight platform for continuous data processing. For task execution systems, platforms such as Apache YARN and Apache Mesos can be used. Apache YARN and Apache Kafka are built-in to Samza (Apache Samza, 2002).

3.3. Real-Time Processing

With the advancement of technology and techniques, real-time data processing ensures that real-time data will be acted on time. That is, the processing is measured in milliseconds, and output is provided as soon as the input comes (Ariyaluran *et al.*, 2019). Real-time systems are incredibly difficult to construct with standard software. Amazon Kinesis and Apache Storm are the most well-known frameworks for this kind of processing.

3.3.1. Amazon Kinesis

Amazon Kinesis is a framework for distributed message queuing. It can handle massive data sets and vast pipelines, and the output created by Kinesis may be used by machine learning techniques (Ariyaluran *et al.*, 2019). Amazon Kinesis allows you to receive, store, and analyze real-time streaming data, allowing you to get insights in seconds or minutes rather than hours or days. With amazingly low latency, Amazon Kinesis can handle any quantity of streaming data and analyze data from hundreds of thousands of sources (Amazon Kinesis, 2022).

3.3.2. Apache Storm

The storm was created by "Nathan Martz of BackType", which Twitter purchased in 2011. In 2012, the storm was made open-source and was then incorporated into Apache projects in 2014 (Khalid and Yousaf, 2021). It is intended to be scalable, robust, extendable, efficient, and simple to manage. Beyond the structure, the primary objective is to eliminate message loss due to node failures and to assure at least one processing (Cumbane and Gidófalvi, 2019). Apache Storm is a real-time distributed large data processing framework developed to process massive volumes of data in the most fault-tolerant and horizontally scalable way possible. It uses Apache Zookeeper to handle the cluster state and the distributed environment. It accepts a raw stream of real-time data at one end and processes it through a succession of small processing units at the other, and produces meaningful data at the other (Basha *et al.*, 2019).

3.4. Interactive Processing

It is an interactive data analysis approach that allows for the interactive querying of Big Data streams to satisfy the requirements in reaction time, a variety of data with a terabyte size. The user is instantly linked to the computer and may interact with it; data can be compared, altered, and assessed in a visual or tabulated format, or both at the same time. The essential issue in interactive processing is dealing with little jobs, which are separated into Map/Reduce and are inefficient to deal with (Abuqabita *et al.*, 2019). The most common frameworks in this category are Apache Impala and Apache Drill.

3.4.1. Apache Impala

Impala, an open-source SQL engine that operates on hundreds of computers as a distributed architecture, is assessed in 2015. Impala has a Massively Parallel Processing (MPP) engine that outperforms Hive and Spark SQL. Impala provides adequate performance by using aggregations, scans, and joins to provide queries; It has a failure tolerance and low latency; impala data is saved in Parquet files; this apache does not utilize Hadoop but instead installs a collection of modules on each Data Node for local processing; this method is designed to prevent bottleneck difficulties. Impala has a relatively low run time and is provided by HiveQL (Abuqabita *et al.*, 2019).

3.4.2. Apache Drill

Apache Drill is a distributed system for interactive big data analysis. It is a distributed query engine for large-scale datasets with minimal latency, including structured and semi-structured. Drill, inspired by Google's Dremel, is intended to expand to thousands of nodes and query petabytes of data at the interactive rates required for BI / Analytics settings. The 'Drillbit' service is at the heart of Apache Drill, responsible for taking client requests, executing queries, and providing results to the client. Drill relies on Zookeeper to keep track of cluster membership and health-check data. The drill is compatible with a wide range of file systems, and NoSQL databases including MongoDB, HBase, MapR-DB, MapR-FS, HDFS, Google Cloud Storage, Amazon S3, Azure Blob Storage, NAS, Swift, and local files. Data from several data stores can be joined in a single query (Apache Drill, 2022).

3.5. Hybrid Processing

The frameworks in hybrid processing can be used for more than one form of data processing, which means that they support both batch data processing and stream data processing. Apache Spark and Apache Flink are the most well-known frameworks for this type of processing.

3.5.1. Apache Flink

Apache Flink is a modern framework for distributed processing and intensive streaming analytics. It is a large-scale data processing framework for the next generation that is meant to operate with low latency and high throughput in all common cluster setups (Toliopoulos *et al.*, 2020). Flink was founded in 2009 as Stratosphere at the Technical University of Berlin. Stratosphere became an open-source project in 2014 as an Apache incubator project called "Flink". Flink is capable of processing data 100 times quicker than MapReduce. Flink is primarily a stream processing engine that does not provide its storage or resource management system. Flink provides two fundamental APIs: the DataSet API for processing bounded data streams (batch processing) and the DataStream API for potentially unbounded data streams (Apache Flink, 2022).

3.5.2. Apache Spark

Apache Spark is an open-source large data processing platform designed for speed and complex analytics. It's simple to use and was created in 2009 at UC Berkeley's AMPLab. It was made available as an Apache project in 2010. Spark allows you to easily create apps in Scala, Java, or Python (Apache Spark, 2022). Spark is an extremely powerful tool that can handle large-size datasets that are structured, semi-structured, or unstructured in a variety of ways. It can handle data in batches or streams. Spark includes MLlib and Spark ML (Pipelines API). Spark MLlib is Spark's implementation of Resilient Distributed Datasets based on machine learning methods. The new Spark ML is built on top of the Spark dataset API. It has tremendous scalability and exceptional usability. The pipeline is a sophisticated capability included with Spark ML. The processing speed of Hadoop MapReduce is slow since it needs disk access for reads & writes. Spark, on the other hand, stores data in memory, decreasing the read or write cycles (Ahmed *et al.*, 2020). In memory, Spark can run applications up to hundreds of times faster than Hadoop MapReduce, while on disk, it can run applications ten times faster (Al-Barznji and Atanassov, 2018). For huge dataset tasks, Spark is preferable more than MapReduce. Spark has been embraced as a processing engine for handling Big Data challenges by numerous research disciplines, like pattern mining and machine learning, due to its diverse capabilities (Hicham and Anis, 2021).

4. Discussion

This section presents a collection of common features discovered throughout this research and compares the frameworks across these features, as summarized in Table 1. As shown here, all the explained frameworks are open-source except for the Dryad framework, which is closed-source, and they are frameworks for massively distributed or / and parallel processing. Moreover, according to the big data processing types, the frameworks are grouped, and most of the frameworks' computation modes are in memory, and their latency is low, but the computation modes of Hadoop and Dryad frameworks are on the disk, and they have high latency. In addition, for processing speed, the Flink framework has the fastest processing, but Hadoop is the slowest one, and all the mentioned frameworks are highly fault-tolerant; only Apache Drill has low fault-tolerant, etc. That will be a very useful guide for selecting the best suitable frameworks based on their characteristics for handling and processing different big datasets or applications efficiently, and so on.

Table 1: Compares the Features of Big Data Processing Frameworks

Frameworks / Features	Hadoop	Dryad	Spark	Flink	Storm	Samza	Drill	Impala
Major Backers	Google, Yahoo!	Microsoft	Berkeley's AMPLab	Apache Software Foundation	BackType, Twitter	LinkedIn	Google's Dremel	Marcel Kornacker
Open Source	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Big Data Processing	Batch	Batch	Hybrid (Batch and Stream)	Hybrid (Batch and Stream)	Real-time	Streaming	Interactive	Interactive
Resource Manager	YARN	YARN	Stand-alone, YARN, Mesos	Stand-alone, YARN, Mesos	YARN, Mesos	Stand-alone, YARN, Mesos	Zookeeper	YARN
Storage	HDFS	Distributed File System (DFS)	HDFS, HBase, Hive, Casan-dra	HDFS, streams databases	HDFS	HDFS	HDFS, HBase, and Hive	HDFS or HBase
Data Sources	HDFS	Computer cluster or a data center	DBMS, HDFS, and Kafka	DBMS, HDFS, and Kafka	Spout	Kafka	HDFS, Hive, RDBMS, HBase, and MongoDB	HDFS, HBase
Computing Mode	Disk-based	Disk-based	In memory	In memory	In memory	In memory	In memory	In memory
Processing Speed	Slow	Fast	Fast	Very Fast	Fast	Fast	Fast	Fast
Execution Model	MapReduc	Directed Acyclic Graph (DAG	Resilient Distributed Dataset (RDD), DAG	Data flow graph	Topology	DAG	Query execution "pipeline model"	HiveQL, Massively Parallel Processing (MPP)
Scalability	High	High	Moderate	High	Moderate	Low	High	High
Fault Tolerance	Yes	Yes	Yes	Yes	Yes	Yes	Yes (Low)	Yes
Latency	High	High	Low	Very Low	Very Low	Very Low	Low	Low
Throughput/ Performance	High	High	High	High	Medium	High	High	High
Implementa-tion Languages	Java	C++	Scala	Java, Scala	Clojure	Scala, Java	Java	SQL
Supported Programming Languages	C, C++, Perl, Ruby, PHP, Python, etc.).	.Net (C#, VB, etc.)	Java, Scala, R, and Python	Java, Scala, R, and Python	Any Programm-ing Language	JVM Languages (Java, Sca-la)	SQL and Alternative Query Languages	All languages supporting JDBC/ODBC

5. Conclusion and Future Work

In the existence of multiple different big data processing frameworks, choosing the best-suited framework based on the application and environment is difficult. Certainly, no explicit guidance is offered to assist developers and users in selecting an appropriate framework for their project. To address this research gap, this study intends to present a comprehensive compilation of the most prominent big data processing frameworks, highlighting the benefits and shortcomings of each framework. Furthermore, this study explored what big data meant from the outset and highlighted the most impacted sources responsible for creating data volume, as well as the big data features and certain big data issues. Then, in detail, the most well-known methodologies of big data processing frameworks were explored and classified into five classes: batch processing, streaming processing, real-time processing, interactive processing, and hybrid processing. As a result, it may be used as a guide for determining the optimal framework for an application, IT analytics, assisting researchers, and readers, as well as business users, in making faster and more informed decisions, enhancement, promoting innovative work, and implementation of such upcoming beneficial frameworks soon. Future work will involve experiments on large data sets via each framework and comparing results to determine their efficiency in handling large volumes of data.

References

Abuqabita., Al-Omoush, R. and Alwidian, J. (2019). A Comparative Study on Big Data Analytics Frameworks, Data Resources and Challenges. *Mod. Appl. Sci.,* 13(7), 1–14.

Acharjya, D.P. (2016). A Survey on Big Data Analytics : Challenges, Open Research Issues and Tools. *Int. J. Adv. Comput. Sci. Appl.,* 7(2), 511–518.

Ahmed, N., Andre L.C. Barczak, Teo Susnjak. and Mohammed A. Rashid. (2020). A Comprehensive Performance Analysis Of Apache Hadoop And Apache Spark For Large Scale Data Sets Using HiBench. *Journal of Big Data*7(110), 1–18, Springer.

Al-Barznji, K. and Atanassov, A. (2016). A Survey of Big Data Mining: Challenges and Techniques. in Proceedings of 24th International Symposium "*Control of Energy, Industrial and Ecological Systems"*, 113–117, Bankia, Bulgaria.

Al-Barznji, K. and Atanassov, A. (2016). A MapReduce Solution for Handling Large Data Efficiently. *Int. Sci. J. "MACHINES. Technol. Mater.,* 23 (12), 20–23.

Al-Barznji, K. and Atanassov, A. (2018). Big Data Sentiment Analysis Using Machine Learning Algorithms in Proceedings of 26th International Symposium "Control of Energy, Industrial and Ecological Systems", Bankia, Bulgaria, 53–58.

Amazon Kinesis (Received on April 2022). https: / / aws.amazon.com / kinesis /

Apache Drill (Received on April 2022). https: / / drill.apache.org / architecture /

Apache Flink (Received on April 2022). https: / / flink.apache.org /

Apache Spark (Received on April 2022). https: / / spark.apache.org / docs / latest / cluster-overview.html

Apache Samza (Received on April 2022). https: / / engineering.linkedin.com / samza / apache-samza-linkedins-stream-processingengine /

Basha, S.A.K., Basha, S.M., Vincent, D.R. and Rajput, D.S. (2019). Challenges in Storing and Processing Big Data Using Hadoop and Spark. *Deep Learning and Parallel Computing Environment for Bioengineering Systems,* Elsevier Inc, 179–187.

Benjelloun, S. *et al.* (2020). Big Data Processing / : Batch-Based Processing and Stream-Based Processing. in Fourth International Conference On Intelligent Computing in Data Sciences (ICDS), *IEEE*, pp. 1–6.

Cumbane, S.P. and Gidófalvi, G. (2019). Review of Big Data And Processing Frameworks for Disaster Response Applications. *ISPRS International Journal of Geo-Information,* 8(387), 1–23.

Cuzzocrea, A. and Loria. (2021). Big Data Lakes: Models, Frameworks, and Techniques. in Proceedings - 2021 IEEE International Conference on Big Data and Smart Computing, pp. 1–4.

Gandomi, A. and Haider, M. (2015). Beyond the Hype: Big Data Concepts, Methods, and Analytics. *Int. J. Inf. Manage,* 35(2), 137–144.

Habeeb, R.A.A., Nasaruddin, F., Gani, A., Hashem, I.A.T., Ahmed, E. and Imran, M. (2019). Real-Time Big Data Processing For Anomaly Detection: A Survey. *Int. J. Inf. Manage,* 45, 289–307.

Hicham, R. and Anis, B.M. (2021). Processes meet Big Data: Scaling process discovery algorithms in Big Data environment. *Journal of King Saud University - Computer and Information Sciences,* 1-12.

Khalid, M. and Yousaf, M.M. (2021). A Comparative Analysis of Big Data Frameworks: An Adoption Perspective. *Appl. Sci.,* 11(22), 1–25.

Manjula, E. and Prema, A. (2020). A Comparative Study on Processing Sequence of Big Data Framework. *Journal of Information and Computational Science,* 10(8), 383–390.

Otoo-Arthur, D. and Zyl, T.L. van (2020). A Scalable Heterogeneous Big Data Framework For E-learning Systems in 2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems- Proceedings, 1–15.

Reinsel, D., Gantz, J. and Rydning, J. (2018). The Digitization of the World - From Edge to Core. *Fram. Int. Data Corp.,* 11, 1-28.

Saadoon, M. *et al.* (2022). Fault Tolerance In Big Data Storage and Processing Systems: A Review On Challenges And Solutions. *Ain Shams Engineering Journal,* Elsevier, 13(2), 1–13, ScienceDirect. Faculty of Engineering, Ain Shams University.

Shikha Soni, M.M.Y. and Jimmy, Singla. (2018). Big Data: Frameworks and Challenges. *J. Emerg. Technol. Innov. Res.,* 5 (10), 535–541.

Toliopoulos, T. *et al.* (2020). Continuous Outlier Mining of Streaming Data in Flink. *Information Systems.* Elsevier Ltd., 93, 1–16.

Zheng, Z., Ping, W., Jing, L. and Shengli, S. (2015). Real-Time Big Data Processing Framework: Challenges and Solutions. *Appl. Math. Inf. Sci. An Int. J.,* 9 (6), 3169–3190.

Cite this article as: Kamal Al-Barznji (2022). Big Data Processing Frameworks for Handling Huge Data Efficiencies and Challenges: A Survey. *International Journal of Data Science and Big Data Analytics,* 2(1), 1-9. doi: 10.51483/IJDSBDA.2.1.2022.1-9.