Ingénierie des Systèmes d'Information
Vol. 29, No. 1, February, 2024, pp. 147-152
Journal homepage: http://iieta.org/journals/isi

Comparative Efficiency Evaluation of Hadoop and Spark Frameworks Using Random Forest Algorithm for Intrusion Detection

Wasnaa Jawad[1*], Abbas Al-Bakry[2]

[1] Informatics Institute for Postgraduate Studies, Iraqi Commission for Computers and Informatics, Baghdad 10011, Iraq
[2] University of Information Technology and Communications, Baghdad 10011, Iraq

Corresponding Author Email: phd202120678@iips.edu.iq

Copyright: ©2024 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/isi.290116

Received: 18 June 2023
Revised: 6 November 2023
Accepted: 7 December 2023
Available online: 27 February 2024

Keywords:
big data, distributed frameworks, Hadoop, intrusion detection system, machine learning, performance, Random Forest, spark

ABSTRACT

This study uses the Random Forest algorithm to evaluate the efficiency of Hadoop, and Spark distributed computing systems for intrusion detection, highlighting the growing importance of efficient distributed systems in handling big data. This research aims to assess and compare the performance of Hadoop and Spark in the context of an intelligent intrusion detection system. We use the Random Forest machine learning algorithm to train and test the system. The methods developed an intrusion detection system using Hadoop and Spark frameworks, followed by a thorough performance assessment using a real-world dataset. The problem this study tackles is the ever-increasing demand for processing data swiftly and accurately in a distributed fashion. We aim to identify the strengths and weaknesses of Hadoop and Spark in the context of machine learning-based intrusion detection. The "intelligent network detection system for intrusions" in this study uses a sophisticated security system using machine learning algorithms to detect potential intrusions, assessing Hadoop and Spark's performance in real-world scenarios and handling large-scale data processing. The findings provide insightful information about the efficacy and efficiency of distributed systems in machine learning activities, which can help select big data application frameworks.

1. INTRODUCTION

Intrusion detection systems are essential to network security because they use various methods to differentiate between legitimate and suspect network activity, protecting against online attacks. These systems rely on variables such as service request patterns and network communication frequency to identify unusual network user behaviour. As the term "big data" implies, vast and diverse datasets covering various network activities are constantly created. Cybersecurity experts find it difficult to manage this data, which includes both potentially dangerous breaches and legitimate network exchanges, because of the number and severity of assaults.

Context: Intrusion detection systems were created in response to the growing worries about cybersecurity in modern network environments.

Problem Statement: This study aims to tackle the challenges caused by the increasing amount of data in the intrusion detection system.

Spark and Hadoop: These distributed frameworks have an exceptional track record for handling huge amounts of data in an effective manner. We assess its application in improving intrusion detection skills in this work.

Methods: To gain a deeper understanding of intrusion detection, we will contrast Spark with Hadoop. We assess several frameworks in machine learning using the Random Forest method.This research purpose is to promote the advancement of even more robust and effective intrusion detection methods during the rise of big data and widespread computing.

2. RELATED WORK

Many strategies have been researched to improve intrusion detection system (IDS) performance in the context of network security. The original Random Forest method's weaknesses were analysed by eminent cybersecurity specialists in the study of Masarat et al. [1]. The authors, who are considered to be experts in their domain, presented a brand-new parallel Random Forest method for IDS. Enhancing feature selection, classifier selection efficiency, training feature quantity, and combination phases were the goals of this approach. Their study fared better overall, in terms of misclassification costs and scalability, than both the regular Random Forest approach and a Hadoop-based versions.Because log files include enormous amounts of data in various forms, log file analysis is useful for processing and is essential to understanding system activity. A university research team carried out investigations in the study of Mavridis and Karatza [2] to meet

this requirement. The aim of their study was to evaluate log file analysis with Hadoop and Apache Spark, two cloud computing frameworks. This project demonstrated us the general ability to manage several log files.Additionally, cloud-based log file analysis methods have been looked at in the study of Kotiyal et al. [3]. The well-known big data specialists who wrote the paper stressed how traditional relational databases are unable to manage the volume of log files generated. Their research aimed to set the scene by emphasising the advantages of using Hadoop clusters for log file analysis and using its wide data processing capabilities.

A group of data processing specialists developed a weblog analysis platform using Pig Latin, Hadoop HDFS, and Hadoop MapReduce [4]. This method sought to get over the limits on data processing that come with using relational databases in the conventional sense.

There is a growing trend among healthcare establishments to use social media information to enhance their services while reinforcing network protection. A practical method for monitoring and examining Twitter posts was suggested by esteemed industry expert Li Wang in the study of Masarat et al. [1], which aims to determine user feelings and formulate messages that will strike a chord with a large number of users. This research was undertaken with the goal of boosting the efficiency of healthcare systems by incorporating the opinions of users.Finally, within the related work topic, several studies on improving network security, analysing log files, and using social media data for service improvement are given. Collectively, these studies broaden the body of knowledge that informs our research on the efficacy of Hadoop and Spark frameworks for intrusion detection.

3. DISTRIBUTED FRAMEWORKS

The open-source Hadoop technology enables distributed storage and large-scale data processing. Big data is the phrase for datasets that are too large or complex for traditional data processing technologies, and Hadoop is designed to handle these types of information. It offers fault tolerance and flexibility through distributed data processing among computer clusters. The MapReduce programming language and the Hadoop Distributed File System (HDFS) are the two primary components of Hadoop [5, 6].

Hadoop's Benefits The main advantage of Hadoop is its ability to manage enormous amounts of data, which makes it appropriate in situations requiring large amounts of data storage and batch processing. For instance, corporations use Hadoop in the real world for applications like clickstream analysis, log file analysis, and recommendation engines. Huge files can be stored over numerous workstations and multiple Hadoop cluster nodes can access high-throughput data thanks to the Hadoop Distributed File System, or HDFS. For fault tolerance, data is also replicated across several nodes. When data dependability and processing continuity are crucial, this fault-tolerant architecture is handy.

Benefits of Spark: Spark is well known for its ability to process data in memory, significantly speeding up data processing when compared to traditional disk-based storage solutions. This speed is very useful for real-time data processing. In real-world applications, Spark is used in fraud detection, streaming data analysis, and sensor data processing. Spark leverages Resilient Distributed Datasets (RDDs) to store data in memory across a cluster of computers. This results in processing that is faster. It is often applied in scenarios including streaming data analysis, graph processing, and machine learning.

Purpose of the Mahout: The open-source Mahout machine learning library is constructed on top of Apache Hadoop. Mahout offers scalable machine-learning algorithms and building blocks for applications that process and analyze large volumes of data. Businesses utilize Mahout, for instance, for recommendation engines, sentiment analysis, and anticipating customer behavior. Because Mahout can operate in a distributed computing environment on top of Hadoop, it is helpful for managing extensive data collections [7].

The MapReduce programming paradigm enables distributed processing of large data sets across computer clusters. HDFS Architecture can handle enormous amounts of data in parallel by breaking the data into smaller chunks, processing each chunk separately, and then aggregating the outcomes [8, 9].

The two primary parts of the MapReduce model are the mapping function and the reduction function. The map function converts input data into key-value pairs, which are then handled concurrently by several nodes in a cluster. The reduce function condenses the result of the map function into a more manageable set of key-value pairs [10].

Because the map function is independent and scalable by design, it can operate concurrently across several cluster nodes. Every node creates intermediate key-value pairs and performs some data processing. After these intermediate pairings have been combined and sorted, the reduction function is applied, as illustrated in Figure 1.

Processing and analyzing large amounts of data is a unique application for MapReduce because of its ease of expansion to handle data sets too big to fit on a single machine. Additionally, it offers fault tolerance because processing can keep going even if a node fails. After all, data is duplicated across several cluster nodes [11].

Apache Hadoop is the foundation for the open-source machine learning library known as Mahout. It offers a collection of scalable machine-learning algorithms and building blocks for creating applications that can process and analyze massive amounts of data.

Among other machine-learning tasks, Mahout provides several algorithms for data mining, classification, clustering, and cooperative filtering. It can be utilized for applications like engines for recommendation, fraud detection, and analysis of sentiment because it is made to work with large-scale data sets [12].

Mahout's ability to operate on top of Hadoop allows it to process big data sets in a distributed computing environment, which is one of its main advantages. As a result, machine learning processes can be carried out on data sets that are too big to accommodate a single machine [13, 14].

Mahout also has tools for analyzing and visualizing the output of the machine learning algorithms, which makes it simpler to comprehend and analyze the output of the algorithms as shown in Figure 2.

Spark is an open-source distributed computing platform that is made for processing and analyzing massive amounts of data. It was created at the AMPLab at UC Berkeley, and the Apache Software Foundation now looks after its upkeep.

Programming distributed data processing processes across computer clusters is feasible using an interface called Spark. It includes several APIs for handling graph processing, machine learning, and unstructured and structured data. Processing data

in memory allows Spark to operate significantly faster than traditional data processing platforms that rely on disk-based storage, making it one of its essential characteristics. Spark uses a data processing model called Resilient Distributed Datasets (RDDs), which enables data to be stored in memory throughout a cluster of workstations. As a result, Spark performs intricate computations far faster than traditional batch-processing systems [15].

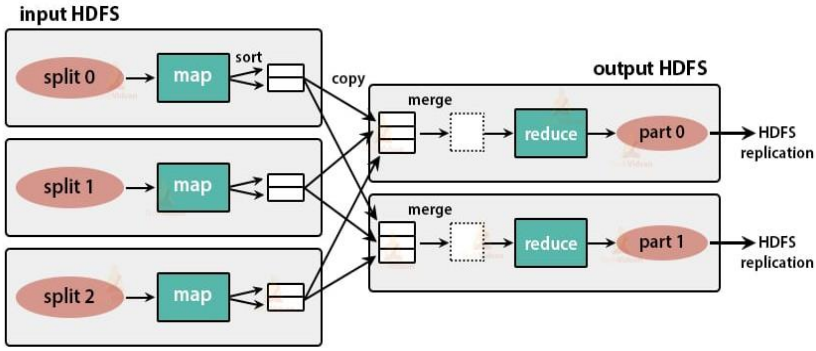

Figure 1. Hadoop MapReduce architecture [10]

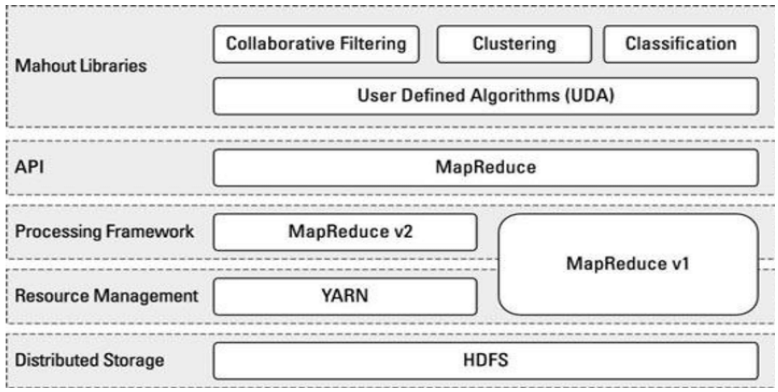

Figure 2. Mahout architecture [13]

Moreover, Spark includes several libraries for handling graphs, machine learning, and streaming data. These libraries simplify creating sophisticated data processing software that can deal with massive data sets in real time [16].

Spark is an effective tool for handling and evaluating large-scale data collections [17, 18].

In this paper, we have adopted the adult dataset. Barry Becker extracted data. The following conditions were used to obtain a group of substantially clean records: (AAGE>16, AGI>100, 5AFNLWGT>1, HRSWK>0, and AGI>100)), The prediction task is to ascertain whether a person earns more than $50,000 annually.

Variable Explanation: The dataset used in this paper is the adult dataset, which was extracted by Barry Becker from the 1994 Census database. To enhance clarity, here are explanations for the variables:

(1) fnlwgt: The "fnlwgt" variable represents the final weight. It is a numerical value used in survey sampling to account for the unequal probability of being sampled. This variable is often used to ensure that the dataset is representative of the population.

(2) Education Number: The "Education Number" is a numerical representation of a person's educational level. It is typically mapped from the "education" variable and can be used more straightforwardly to represent academic qualifications.

(3) Native-Country: "Native-country" denotes an individual's country of origin or citizenship. It specifies the nation where the person is from.

(4) AGI: The "AGI" variable refers to Adjusted Gross Income. It typically represents an individual's or household's income after certain deductions and adjustments have been made, which are allowed by tax laws. In the context of this dataset, it could be a measure of a person's income and may play a crucial role in predicting whether they earn more than $50,000 annually.

(5) HRSWK: The "HRSWK" variable denotes the hours worked per week. It quantifies the weekly working hours of an individual. Working hours can be an important factor in determining income, and it's often used in predictive models to understand the relationship between hours worked and earnings.

3.1 List of characteristics

>50K, <=50K.

continuous age.

workclass: Individual, Federal, local, and state governments, as well as self-employment corporations, Unpaid, never had a job.

fnlwgt: perpetual.

education: bachelor's, some college, 11th, high school graduate, prof school, associate acdm, associate voc, 9th, 7th-8th, master's, first through tenth, doctorate, fifth through sixth, preschool.

Education Number: ongoing.

marital-status: Married, Civil Partner, Divorced, Single split up, widowed, Married-spouse-absent, Married-AF-spouse.

occupations include tech support, craft repair, other services, sales, executive management, professional speciality, handlers and cleaners, machine operators and installers, administrative support, farming, fishing, transport and moving, priv home service, protective service and armed forces.

spouse, parent, husband, not in family, other relative, and single.

white, pacific islander, American Indian, Eskimos, other, and black.

sex: Male and female.

Gains from capital are ongoing.

Continuous capital loss.

weekly hours: constant.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

Predictive Models: For this type of dataset, various predictive models can be employed like:

(1) Logistic Regression
(2) Decision Trees
(3) Random Forest
(4) Support Vector Machines
(5) Naive Bayes
(6) Neural Networks

4. METHODOLOGY

The Random Forest approach was used in this study to build an intelligent network system for intrusion detection using machine learning. The system's model was trained using the adult dataset and implemented using Hadoop-Mahout and Spark, two distributed frameworks. A Random Forest machine learning approach is frequently employed for classification and regression problems. It is a member of the ensemble method family, which combines the results of various models to enhance performance. Within the forest of decision trees that the algorithm produces, each decision tree is trained using a different subset of the characteristics and a random subset of the data. When utilizing a Random Forest to produce a prediction, the algorithm first assesses the input information on each of the choice trees in the forest before averaging (for classification) or using a majority vote (for regression) to combine the findings. This method enhances the overall accuracy and generalizability of the model while reducing the influence of specific decision trees that may overfit the data.

Its key benefits are its scalability, resilience, and capacity to handle high-dimensional information with intricate feature interactions [19, 20].

Several factors make the machine-learning algorithm Random Forest popular [21]:

(1) Robustness.
(2) Flexibility.
(3) Accuracy.
(4) Interpretability.
(5) Scalability.

The values of the Random Forests algorithm's parameters in the Hadoop and Spark frameworks, respectively, are shown in Tables 1 and 2.

This decision was supported by several important factors that make the haphazard forest approach especially appropriate for our study goals:

(1) Ensemble Learning: One of the products of the ensemble method family, Random Forest, is renowned for its ability to aggregate the results of several models. This feature is beneficial for our work since it lets us use the combined predictive strength of many decision trees in the forest, which improves model performance.

(2) Reduction of Overfitting: Overfitting is a standard machine learning problem that occurs when a model is overly complex and performs well on training data but badly on new, unseen data. Random Forest lowers this risk by training each decision tree on a distinct subset of data using a random feature selection. This improves the model's generalizability by preventing the overfitting specific trees.

(3) Accuracy and Robustness: Random Forests are known for their robustness, accuracy, and flexibility. These characteristics are critical for intrusion detection in a dynamic, diversified network environment.

Table 1. Random Forest parameters in Spark

Parameter	Value
Number of trees	20
Maximum depth of trees	5
Minimum number of instances per leaf	1
Number of features to consider for each split	Sqrt (number of features)
Bootstrap sampling with replacement	Enabled

Table 2. Random Forest parameters in Hadoop-Mahout

Parameter	Value
Number of trees	100
Maximum depth of trees	Unlimted
Minimum number of instances per leaf	1
Number of features to consider for each split	Sqrt (number of features)
Bootstrap sampling with replacement	Enabled

Because Random Forest can generate results that are accurate, dependable, and easily comprehensible it was chosen above other machine learning methods. It continues to be a reliable method for classifying network activity into suspicious and regular patterns, which is essential to our investigation's primary objectives.In this study, we used the Random Forest machine learning technique for intrusion detection to Hadoop-Mahout and Spark, two popular distributed frameworks. There were benefits and disadvantages to each of the different variables used in choosing one of these frameworks.

Comparing Frameworks: Scale and speed are balanced in the decision between Spark and Hadoop-Mahout. Large-scale dataset handling and machine learning algorithm integration are strengths of Hadoop-Mahout, and Spark's in-memory processing speed allows for quick replies for real-time intrusion detection. Our study's particular needs, which emphasised the necessity for both speed and scalability, further led our conclusion.

Feature Importance Analysis: Random Forest models inherently provide a measure of feature importance during their operation. As each decision tree in the forest is

constructed, it computes a metric known as the Gini impurity, which quantifies the extent of feature importance. The feature that results in the most significant reduction of Gini impurity during the splitting of decision tree nodes is deemed the most important feature for classification.

Most Influential Features: In the context of our intrusion detection model, feature importance analysis indicated that several features played a pivotal role in classifying network behavior into normal and suspicious patterns. While the specific ranking of feature importance can vary between individual models and datasets, some of the features that emerged as influential include "age," "hours worked per week," "education level," and "marital status."

For example, the "age" feature is often highly influential as older individuals may have more stable employment and financial patterns. "Hours worked per week" can provide insights into employment status, while "education level" may correlate with higher income. "Marital status" can also indicate financial stability and earning potential. By examining the feature importance rankings, we gained valuable insights into the factors that significantly impact the classification of income levels in the context of intrusion detection.

Class imbalance occurs when one class (e.g., 'earnings >$50,000') dramatically surpasses the other (e.g., '<=$50,000') in intrusion detection datasets, including the adult dataset employed in our study. To stop biased results from resulting from the prediction model's preference for the majority class while ignoring the minority class., it is imperative to address the class imbalance. We handled this problem as follows during model training:

Techniques for Resampling: We used resampling methods to address the issue of class imbalance. To produce a more balanced train dataset, these strategies try to either undersample the majority class or oversample the minority class. In particular, we oversampled the minority class employing the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE generates synthetic examples from the minority class by interpolating between existing instances. This balanced the class distribution and prevented the model from being biased towards the majority class.

Evaluation Metrics: in the evaluation of our model, we considered a range of appropriate metrics beyond accuracy. For imbalanced datasets, accuracy alone can be misleading. We paid close attention to metrics such as precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC). These metrics provide a more comprehensive assessment of the model's performance, accounting for the true positives, false positives, and false negatives. This allowed us to better understand the model's ability to detect intrusions while minimizing false alarms.

5. RESULTS AND DISCUSSION

By constructing the system and evaluating the suggested solution interference detection system's precision using a number of performance assessment parameters like the precision, recall, and F1-score, a comparison between the two distribution frameworks was made.

The ratio of correctly categorized samples to all samples in the dataset is known as accuracy. It is determined by:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (1)$$

The model's accuracy indicates how well it predictions positive as well as negative samples.

Precision is defined as the ratio of actual positive samples to all anticipated positive samples. It is determined by:

$$Precision = \frac{TP}{(TP + FP)} \quad (2)$$

Precision assesses the degree to which the model distinguishes between positive and negative samples while not being overly forgiving when categorizing the latter as positive [20].

Recall is defined as the proportion of real positive samples to all positive values in the dataset. It is determined by:

$$Recall = \frac{TP}{(TP + FN)} \quad (3)$$

The average harmonic of recall and precision is known as the F1-score. It is a fair measurement that accounts for both recall and precision. It is determined by:

$$F1-score = \frac{2 * Precision * Recall}{(Precision + Recall)} \quad (4)$$

A high F1-score shows that the model has both high precision and high recall. The F1-score assesses the ratio between precision and recall.

After performing a simulation of the model that we built and using the adult dataset, the results appeared as follows: For the Spark framework as shown in Table 3:

Table 3. Results for Spark framework.

Accuracy	Test Error
98.4%	1.56%

For Hadoop framework as shown in Table 4:

Table 4. Results for Hadoop framework

Accuracy	Reliability	Precision	Recall	F1-Score
85.84%	52.18%	0.8532	0.8584	0.8542

After comparing these results, we conclude that the Spark framework has obtained higher accuracy rates than the Hadoop framework.

6. CONCLUSIONS

The study's main findings entail a comparative analysis of Hadoop and Spark for intrusion detection. Spark outperformed Hadoop with higher accuracy, precision, and recall due to its smaller tree count and finite tree depth. Spark's real-time processing excelled, while Hadoop suited offline tasks. The class imbalance was handled using SMOTE. Critical features like "age," "hours worked per week," "education level," and "marital status" proved influential. Practical implications lie in real-time intrusion detection system enhancement. The study advances intrusion detection research in processing large-scale data for security applications.

Benefit analysis:

(1) Improved Security: Enhance intrusion detection systems with the proper framework, improving cybersecurity.

(2) Real-Time Protection: Utilize Spark for immediate intrusion detection in critical scenarios.

(3) Efficient Resource Use: Optimize resource allocation, saving time and costs.

(4) Advancing Research: Contribute to intrusion detection research, promoting industry best practices.

The results of the essay hold significance as they offer practical guidance for choosing the appropriate framework for intrusion detection, addressing real-time security needs, and advancing research in large-scale data processing for cybersecurity.

Spark's real-time capabilities empower intrusion detection systems to swiftly analyze network activities, making it indispensable in scenarios where immediate threat detection and response are paramount, such as in financial transactions, critical infrastructure monitoring, and online threat prevention. Its capacity to process and act on data in real-time enables rapid identification of suspicious behaviors and immediate countermeasures, bolstering cybersecurity in an increasingly dynamic and interconnected digital landscape.

REFERENCES

[1] Masarat, S., Sharifian, S., Taheri, H. (2016). Modified parallel Random Forest for intrusion detection systems. The Journal of Supercomputing, 72(6): 2235-2258. https://doi.org/10.1007/s11227-016-1727-6

[2] Mavridis, I., Karatza, H. (2017). Performance evaluation of cloud-based log file analysis with Apache Hadoop and Apache Spark. Journal of Systems and Software, 125: 133-151. https://doi.org/10.1016/j.jss.2016.11.037

[3] Kotiyal, B., Kumar, A., Pant, B., Goudar, R.H. (2013). Big data: mining of log file through Hadoop. In 2013 International Conference on Human Computer Interactions (ICHCI), Chennai, India, pp. 1-7. https://doi.org/10.1109/ICHCI-IEEE.2013.6887797

[4] Wang, C.H., Tsai, C.T., Fan, C.C., Yuan, S.M. (2014). A hadoop based weblog analysis system. In 2014 7th International Conference on Ubi-Media Computing and Workshops, Ulaanbaatar, Mongolia, pp. 72-77. https://doi.org/10.1109/U-MEDIA.2014.9

[5] Imran, Ghaffar, Z., Alshahrani, A., Fayaz, M., Alghamdi, A. M., Gwak, J. (2021). A topical review on machine learning, software defined networking, internet of things applications: Research limitations and challenges. Electronics, 10(8): 880. https://doi.org/10.3390/electronics10080880

[6] Duque Barrachina, A., O'Driscoll, A. (2014). A big data methodology for categorising technical support requests using Hadoop and Mahout. Journal of Big Data, 1(1): 1-11. https://doi.org/10.1186/2196-1115-1-1

[7] Kowalski, C.W., Lindberg, J.E.M., Fowler, D.K., Simasko, S.M., Peters, J.H. (2020). Contributing mechanisms underlying desensitization of cholecystokinin-induced activation of primary nodose ganglia neurons. American Journal of Physiology-Cell Physiology, 318(4): C787-C796. https://doi.org/10.1152/ajpcell.00192.2019

[8] Qin, Y., Tang, Y., Zhu, X., Yan, C., Wu, C., Lin, D. (2020). Zone-based resource allocation strategy for heterogeneous spark clusters. In Artificial Intelligence in China: Proceedings of the International Conference on Artificial Intelligence in China, Singapore, pp. 113-121. https://doi.org/10.1007/978-981-15-0187-6_13

[9] Cobb, A.N., Benjamin, A.J., Huang, E.S., Kuo, P.C. (2018). Big data: More than big data sets. Surgery, 164(4): 640-642. https://doi.org/10.1016/j.surg.2018.06.022

[10] Yang, L., Xu, K., Liu, S. (2017). PADP: A parallel data possession audit model for cloud storage. Concurrency and Computation: Practice and Experience, 29(20): e4154. https://doi.org/10.1002/cpe.4154

[11] Kodali, S., Dabbiru, M., Thirumala Rao, B., Kartheek Chandra Patnaik, U. (2019). A k-NN-based approach using MapReduce for meta-path classification in heterogeneous information networks. In Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018, Singapore, pp. 277-284. https://doi.org/10.1007/978-981-13-0514-6_28

[12] Wei, P., He, F., Li, L., Shang, C., Li, J. (2020). Research on large data set clustering method based on MapReduce. Neural Computing and Applications, 32: 93-99. https://doi.org/10.1007/s00521-018-3780-y

[13] Mostafaeipour, A., Jahangard Rafsanjani, A., Ahmadi, M., Arockia Dhanraj, J. (2021). Investigating the performance of Hadoop and Spark platforms on machine learning algorithms. The Journal of Supercomputing, 77: 1273-1300. https://doi.org/10.1007/s11227-020-03328-5

[14] Ahmed, A.A., Agunsoye, G. (2021). A real-time network traffic classifier for online applications using machine learning. Algorithms, 14(8): 250. https://doi.org/10.3390/a14080250

[15] Gopalani, S., Arora, R. (2015). Comparing apache spark and map reduce with performance analysis using k-means. International Journal of Computer Applications, 113(1): 8-11. https://doi.org/10.5120%2F19788-0531

[16] Glushkova, D., Jovanovic, P., Abelló, A. (2019). Mapreduce performance model for Hadoop 2. x. Information systems, 79: 32-43. https://doi.org/10.1016/j.is.2017.11.006

[17] Guo, A., Jiang, A., Lin, J., Li, X. (2020). Data mining algorithms for bridge health monitoring: Kohonen clustering and LSTM prediction approaches. The Journal of Supercomputing, 76: 932-947. https://doi.org/10.1007/s11227-019-03045-8

[18] 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, Las Vegas, Nev, USA, pp. 21-24.

[19] Nguyen, M.C., Won, H., Son, S., Gil, M.S., Moon, Y.S. (2019). Prefetching-based metadata management in advanced multitenant hadoop. The Journal of Supercomputing, 75: 533-553. https://doi.org/10.1007/s11227-017-2019-5

[20] Sassi, I., Anter, S., Bekkhoucha, A. (2021). A spark-based parallel distributed posterior decoding algorithm for big data hidden markov models decoding problem. IAES International Journal of Artificial Intelligence, 10(3): 789. https://doi.org/10.11591/ijai.v10.i3

[21] Harrington, P. (2012). Machine learning in action. Simon and Schuster, New York, USA.