IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 September 2022, accepted 14 October 2022, date of publication 19 October 2022, date of current version 27 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3215531

RESEARCH ARTICLE

LSDStrategy: A Lightweight Software-Driven Strategy for Addressing Big Data Variety of Multimedia Streaming

SAJA DHEYAA KHUDHUR AND HASSAN AWHEED JEIAD
Department of Computer Engineering, University of Technology Iraq, Baghdad 10066, Iraq
Corresponding author: Saja Dheyaa Khudhur (saja.d.khudhur@uotechnology.edu.iq)

ABSTRACT Many devices, users, and applications stream an irregular amount of varied data every second. This rapid generation of data continues at an enormous rate, constructing the big data that increase the need for solutions, despite resource constraints, to analyze and manipulate data. Current methods allocate cloud resources according to the characteristics of the data. Resource allocation requires a comprehensive view of the workload requirements. However, the data characteristics in big data streams are uncertain due to the random nature of data generation. Choosing and allocating the right resources to this stream is challenging. With the variety of big data streams, the stochastic nature of the stream led to unpredictable requirements and specifications. The critical issue is forecasting the workload to avoid the over-provisioning and under-provisioning of resources. Such forecasting needs an adequate dataset to describe the history logs of the incoming workload. A fast release for such a dataset provides a high chance of deploying forecasting at the right time. This paper addresses this issue with a novel strategy named LSDStrategy that analyzes the received multimedia stream based on its binary content using machine learning techniques with artificial and real datasets. LSDStrategy utilizes an evaluating voting technique to select the optimum classifier to trade off accuracy and prediction time as metrics. Multi-classifiers that have been built and tested include Decision Tree (DT), K-Nearest Neighbor (K-NN), and Random Forest (RF) over multi-content-based features. Experiments evaluated the performance of the adopted models and the selected features. According to experimental analysis, the DT approach provides a consistent performance for both artificial and real-world datasets for 85% and 81.3%, respectively. We deploy and evaluate the LSDStrategy efficiency on a regular specification server through a set of experiments using a synthetic stream. The experiments prove the LSDStrategy agility and adaptivity in identifying the multimedia-based workload type utilizing small chunks of load.

INDEX TERMS Big data, content-based analysis, file-type identification, machine learning, multimedia, variety.

I. INTRODUCTION

Incomprehensible amounts of data are generated every minute. This data is streaming everywhere through multiple devices, which are part of the Internet of Things (IoT). It includes sensors' data, devices that produce a massive amount of data [1], vehicles [2], phones and machines [3], social media, cyber-physical systems [4], business transactions, and many others [5]. The big data concept has evolved into the Internet of Big data (IoBd) [6]. There is no universally accepted definition for the big data term. Still, big data represents a vast volume set of varied data structures from multiple sources that traditional data processing tools cannot handle [7]. As introduced firstly by Laney, the big data concept was characterized by three dimensions, Volume, Velocity, and Variety [8]. Researchers later extended these characteristics to include Veracity, Value, and others [9]. The most remarkable feature of big data is the volume which

The associate editor coordinating the review of this manuscript and approving it for publication was Hong-Mei Zhang.

reveals an explosive amount of data. This amount may be multiple terabytes and extended to petabytes or zettabytes. In 2014, when the partners EMC and IDC released the seventh annual Digital Universe study, the main finding was that the digital universe size would double every two years. According to their estimation, the data between 2013 and 2020 will approximately grow from 4.4 zettabytes to 44 zettabytes. While in December 2020, Statista released a survey on the amount of information globally from 2010 to 2024 [10]. This survey used IDC forecasts until 2020 when the average data volume was close to 59 zettabytes, which was greater than their estimation in 2014 by almost 34%. The estimation calculations from 2021 to 2024 used the 2020 figure and the five-year Compound Annual Growth Rate (CAGR) of 26 percent provided by the source. Moreover, the rate of increase in data generation has exceeded expectations. So, it is impractical to seek a specific threshold for the Volume of big data because its volume definition today may not meet the threshold in the future. The current trend toward finding techniques and algorithms for collecting, preparing, processing, storing, and analyzing that data has become very circumstantial among many researchers and scientists.Moreover, an unstructured dataset may qualify as big data, while a structured dataset of the same size may not. The concept of big data goes beyond Volume to include other characteristics, such as Velocity and Variety.

Velocity, the second vital characteristic of big data environments, is the rate at which data changes. There is a clear interaction between big data's Velocity and Volume characteristics. For example, a small amount of data sent every second by thousands of wireless sensors, devices, or tweets is considered big data due to its high generation speed and the need to capture and process it quickly without losing any data [11]. So, to remove the ambiguity of the definition of big data to identify the tools and face the challenge that arises from them, IBM has built a big data taxonomy shown in Table 1 [12]. This taxonomy clarifies the relationship between the characteristics of data and the challenges facing it. It also illustrates when data is classified as big data. This taxonomy helps distinguish the challenges related to big data and enables scientists, users, and researchers to understand the problems they encounter.

The other distinctive characteristic of big data is Variety. Variety implies that big data is collected in varied formats, including blog entries, videos, text, images, audio, social media profiles, and social media updates, typically denoted as unstructured data. This data type imposes additional requirements that traditional solutions cannot provide [13]. Another important characteristic of big data is Veracity, a term coined by IBM that refers to the untrustworthiness inherent in some data sources. That, in turn, led to a considerable number of noisy, inaccurate, incomplete, redundant, and imprecise objects [12].

Furthermore, various resources are necessary for different workloads, areas, or domains. Due to the diversity of cloud resources in terms of memory size, processor speed, etc., resource allocation algorithms must exploit these resources efficiently without violating the service-level agreement (SLA). To ensure the availability of the services, no machine should be overloaded or underloaded. Besides, the load balancing process affects the optimization of the performance parameters of the allocation process. This process has two types of implementations: static and dynamic. Due to the stochastic nature of the big data environment and the need for heterogeneous resources, static resource allocation faces a limitation [14]. The difficulty of choosing the best cloud resources is due to the proliferation of big data on the cloud [15]. Dynamic resource provisioning is a challenging problem in big data application scheduling [16]. However, workload prediction has a crucial impact on the dynamic resource allocation process. Therefore, workload prediction and job estimation have gained the attention of many researchers and data scientists [17], [18], [19], [20], [21]. However, most of the research exploited the existence of historical data used for workload forecasting. In a big data stream, the characteristics of the arrived stream are unpredictable due to the stochastic nature of streaming data sources. In addition, the user cannot determine the requirement in advance. So, it has become challenging to predict the upcoming workload in real time and allocate the appropriate resources. These challenges are the primary motivation for this paper.

Accordingly, the main objectives of the present study are as follows:

1. Propose a strategy to face the heterogeneity issues of big data that automatically filters and classifies the big data workloads using Machine Learning (ML).
2. Implement the strategy as lightweight software to reduce the cost of complexity in time and space.
3. Optimize multi ML models and nominate the lightest, most accurate one.
4. Automatically select the most critical features from the utilized dataset to decrease the online feature extraction cost.

TABLE 1. The taxonomy of big data [12].

Volume	Velocity	Variety	Veracity	Horizontal scalability?	SQL limitation?	Big data?
×	×	×	✓	No	No	No
×	×	✓	×	May	May	May
×	✓	×	×	May	No	May
✓	×	×	×	Yes	May	Type 1
×	×	✓	✓	May	Yes	Type 3
×	✓	✓	✓	Yes	Yes	Type 3 and 4
✓	✓	×	×	Yes	May	Type 1 and 4
✓	✓	✓	×	Yes	Yes	Type 1, 3, and 4
✓	✓	×	✓	Yes	May	Type 1 and 4
✓	✓	✓	✓	Yes	Yes	Type 1, 3, and 4
✓	✓	✓	×	Yes	Yes	Type 2 and 4

Type 1: Structured data, type 2: Hierarchical data (Big graphs), type 3: Semi-structured data, and type 4: Transactional streams.

5. Equip the proposed strategy to analyze the received data types to create a dataset based on time series, ready data scientists to use to forecast and identify the required resources in advance.

The present study achieves the contributions below through the above objectives.

- It proposes a software-driven lightweight strategy named LSDStrategy with a novel utilization of the content-based analysis approach in identifying the big data stream types (Variety).
- It presents the feature engineering and data preprocessing for the pre-constructed unprocessed dataset [22]. Moreover, it selects the high-impact features based on filter and wrapper features selection methods to reduce the dimensions, which is a crucial factor in dealing with the big data stream.
- It tunes the multi-ML classifiers and establishes a comparative study between them to declare the most appropriate concerning accuracy and complexity.
- The study deploys it as a Tri-Modules achieving the analysis and accuracy of 85% despite the reliance on small chunks of 512-byte size compared to what large pieces of richer information provide.

The paper is organized as follows. Section I introduces and outlines the research work, objectives, and contributions of the study. Section II discusses and surveys the background research and presents reviews of the literature. Section III illustrates the problem statements that motivate this research. Section IV provides a theoretical background for the dataset description and the feature engineering techniques. Section V describes the proposed LSDStrategy to address the research problem. Section VI illustrates the empirical experiment results of the proposed LSDStrategy. Finally, we conclude our research in Section VII.

II. BACKGROUND RESEARCH

The most remarkable feature of big data is the explosive volume of data. However, this massive amount of data has become a double-edged sword in the age of big data. As mentioned, the vast data requires nontraditional computational and storage resources. But increasing the volume of the data increases the accuracy of the built models. Moreover, it increases the possibility of extracting valuable features that enable data scientists to predict crucial topics that require sufficient knowledge of all the influences. In an era of great interest in the Coronavirus pandemic, authors in [23] have exploited the feature of big data datasets of six countries to predict the outbreak of COVID-19 in many countries. However, the volume of big data alone degrades the required performance. Big data cannot create value when dealing with the Volume characteristic alone. Addressing the big data Variety will moderate that impact [24]. Due to the remarkable increase in data sources, focusing on Variety has become an important matter [25].

A comprehensive review of the recent literature surveyed the current state of the art for big data variety. The Variety of big data does not have a clear definition. The dominant description of this characteristic is the diversity of data that generates big data, including structured, semi-structured, and unstructured data generated by machines, devices, or humans. That diversity of the generated data, for example, in today's organizations, yields the significant challenges of indexing, searching, sorting, visualizing, and analyzing [26]. Facing these challenges depends on the nature of the process.

A timeline analysis of the reviewed literature indicates that all researchers have identified Variety as a specific domain challenge and offer limited solutions. In ecological science, Variety in ecoinformatics creates a scalability limitation due to its heterogeneous nature [27]. In educational services, big education data composes of variations due to the diversity of the data sources and the used languages and media that, in turn, generate a challenging in the processing, handling, storing, and visualizing that need exploring to unlock the value from the big data [28]. Kumar stated that the cost component of most big data projects falls around the cost incurred in using the services and tools that address the challenges of Variety [29]. As was previously stated, Variety may negatively affect the value taken from big data. Today, one of the most prominent challenges of the Variety problem is identifying the appropriate sources for the workload and exploiting them efficiently [26]. In [7] and [6], the authors identify the big data characteristics as driving parameters for cloud resource allocation.

The main challenges generated by the Variety include the pre-processing, processing, storing, mining, visualizing, and resource provisioning, which are long-term, either human-driven based on a brief analysis of the workload or proactive and reactive models. Navroop et al. [7] faced the workload variety of big data by identifying the workload that significantly impacts resource selections using the probabilistic data structure. In [30], the proposed system categorized the big data stream according to its Variety. Images, audio, video, and text data are each given their own bloom filter, allowing them to pass while blocking data from other varieties. All bloom filters have the same implementation, except for the data type. In another research project, they proposed a solution by utilizing the file identification tool [6]. These three approaches relied on the semantic parsing of the workload in-depth as a limited approach [22]. Moreover, a software-driven model-based non-semantic parsing for the big data workload was proposed in [31]. The proposed method based on metadata identification in big data focuses on the application of self-learning systems to enable automatic data compliance with legal requirements and the possibility of providing essential and easily accessible metadata for data classification. This approach depends on rules that do not apply to all types of workloads [22]. This way, which relies on file signatures or other characteristics designed to identify data type, becomes ineffective when such data is corrupted or missing.

The objective of the proposed LSDStrategy is to predict the type of multimedia data based on its content analysis using an ML algorithm. This algorithm overcomes the limitation of the semantic and non-semantic parsing approaches due to the statistical classification capabilities of ML [32]. This type of analysis is the first, as far as we know, to analyze streaming data. At the same time, it analyzes the batch workload for digital forensics, security, and cyber-physical [33].

In addition, the proposed strategy allows tuning and evaluating several ML models and nominating the optimal model for the desired purpose, including expanding it and exploiting its outputs to build a resource management-based approach. The proposed strategy exploits the principles of pipelining and multi-threading to accelerate and automate work and increase efficiency. Table 2 summarizes the recent studies that relied on content and metadata analysis for file type identification purposes. The most prominent feature of the work with the literature referred to in Table 2 is the deployment of it as a Tri-Modules composing the possibility of analysis and accuracy of 85% despite the reliance on small chunks of 512-byte size compared to what large pieces of richer information provide [34].

III. RESEARCH PROBLEM

To speed up the transfer of large files between servers or nodes, a mechanism of cutting files into smaller segments or chunks increases the productivity and receptivity of requests [42]. In addition, this transmission mode becomes more efficient due to the ease of resending or requesting a small file, enabling the server to receive multiple requests, parsing, and parallel processing. With the advent of the big data era and the Volume and Velocity characteristics, this mechanism confronted these two characteristics by sending several requests simultaneously. Additionally, it allows sending a file with a size that exceeds the capacity of the recipient. This mechanism will address low bandwidth and frequent disconnection constraints that impact transaction management and data handling ability [42].

On the other hand, because big data streams are unpredictable, researchers use the cloud computing environment due to the availability of multiple computing, tool, and storage resources that are ready on demand. This environment is generally stochastic, which may inherit several types of uncertainties, costs, and delays. The primary cost may be due to trying to meet the specifications and requirements. Further, it also formed due to the delays in provisioning and de-provisioning resources. So, to overcome these challenges, dynamic resource scaling approaches are needed by automatically scaling resources down and up depending on the workload, requirements, and customer demand. With the big data stream nature, there are challenges in forecasting the workload variety due to the heterogeneity of the workload. Due to these challenges, the issue of identifying and allocating resources has become the focus of attention of many business owners, data scientists, and researchers [43], [44], [45], [46]. In this paper, LSDStrategy faces the Variety challenge of the big data stream to predict the workload type nature before waiting for the whole load to provide an analysis to help the resource management process. The deployment phase is done automatically in parallel to the receiving process. That means there is no delay in serving the requests while analyzing the workload. LSDStrategy aims to provide data scientists with statistical information about diverse data flows. That information, in turn, aids them in forecasting the future load and managing the resources supply process intuitively to avoid the problems of over-provisioning and under-provisioning resources and the resource's starting time. Overcoming time consumption and providing scalability features will reduce cost and power while staying within the service-level agreement (SL).

IV. THEORETICAL BACKGROUND

A. DATASET ANALYSIS

The dataset utilized here is based on content analysis [22], which comprises twelve features that have a high impact factor in a content-based model. The baseline details of the used dataset are illustrated in Table 3. The dataset uses two levels of labeling, main type, and sub-type. The main type denotes the basic types of files: image, video, audio, and text. However, the sub-type label samples as JPG, PNG, HTML, TXT, MP4, MOV, M4A, and MP3. The main type and sub-type distribution are shown in Figures 1 and 2, respectively.

B. FEATURE ENGINEERING

The feature engineering process uses data domain knowledge to create more informatics features that improve the ML algorithms' performance. As stated by [13], performance has no single definition, but in this work perspective, the *accuracy, complexity*, and *training/testing time* consumption are the primary metrics in applying feature engineering procedures. The outcome of this step is based on the statistical knowledge taken from the feature selection technique, handling imbalanced data and outliers.

1) FEATURE SELECTION

This technique's objective is to decrease the feature set used in learning the ML models. It submits the procedure of finding and selecting the minimum number of the most informative relevant features. It will enable the ML algorithm to be fast trained, making it easier to interpret and reducing the complexity. On the other hand, if it chooses the proper subset of features, the model accuracy will improve and reduce overfitting. Feature Selection falls into three categories: *filter, wrapped, and embedded* [48]. It utilizes three tools, information gain, univariate selection, and Recursive Feature Elimination (RFE). All these tools are feature ranking metrics computed by evaluating each variable's gain (independent feature) in the context of the target variable (dependent feature) in various statistical ways. The first two tools are *filter methods,* where the ranking process does not rely on the ML

TABLE 2. The recent content and metadata analysis studies.

Contributors	year	Method	Type of analysis	Implementation type	file/ fragment	Dataset Name	Accuracy	Deployment
Nicole et al. [35]	2016	K-Means	Content-based	offline	file	Govdocs1 corpora UTSA Filetypes 1 Data Set	57.56 (theoretical model) 74.08 (winning model)	N/A
Kaur and Sood [7]	2017	probabilistic data structure technique	Metadata-based	online	file	• The Stanford mobile visual search data set • The geographical original of music data set • Bag of words data set	N/A	cloud env.
Kaur and Sood [30]	2017	Four blooms filter (probabilistic data structure technique)	Metadata-based	online	file	• The Stanford mobile visual search data set • The geographical original of music data set • Bag of words data set	N/A	cloud env.
Vulinovic et al. [36]	2019	FFNN	Content-based	offline		Govdocs1 corpus	F1-score: 88%	N/A
Kaur et al. [6]	2019	Packet Identifier filter	Metadata-based	online	file	• The Stanford mobile visual search data set • The geographical original of music data set • Bag of words data set	N/A	cloud env.
Bhatt et al. [37]	2020	SVM	Content-based	offline	fragment	Govdocs1 corpora	67.78	N/A
Mittal et al. [38]	2020	CNN	content	offline	Fragment	File Fragment Type (FFT) - 75 Dataset	77.50%	N/A
Alsubhi et al. [39]	2020	generic and specialized technique	content	offline	fragment	Govdocs1 corpus	79.92-85.09%	N/A
Mina and Jalili [34]	2021	SVM	content	offline	fragment	Govdocs1 and Filetypes1 corpora	74.9% in classifying 512-byte fragments and 87.3% in classifying 4096-byte fragments.	N/A
Bhat et at. [40]	2021	FFNN	content	offline	fragment	Govdocs1 corpus	88.7 % - 90.32%	N/A
Haque and Tozal [41]	2022	K-NN	content	offline	fragment	• Govdocs1 corpus • Random dataset	72%	N/A
Vranopoulos et al. [31]	2022	ANN	metadata	offline	file	• Banking Set (ODS) • National Climatic Data Center (NCDC) • Center for Disease Control and Prevention (CDC)	94%	N/A
LSDStrategy	-	DT K-NN RF	content	online	fragment	• A Content-based File Identification Dataset (machine learning-based dataset) • Govdocs1 corpus	85%	virtual env.

TABLE 3. The Baseline details of the constructed dataset.

Feature Index	Feature Name	Features description	Total Samples	Main Type	Sub-type
F1	Mean Byte Value	The arithmetic mean of the values of the bytes in a file	178031	Image	JPG PNG
F2	Unigram Frequencies (STD)	Unigram Frequencies: This is one of the most common *Byte Frequency Distribution (BFD) features. It* calculates the occurrence frequency of the byte value in a file. Due to it dealing with byte level, 256 different values are produced. So, to escape the curse of dimensionality, the normal distribution (standard and mean absolute deviation) and mean value for these values are taken			
F3	Unigram Frequencies (MAD)				
F4	Unigram Frequencies (Mean)			Text	HTML TXT
F5	Longest Streak	The largest number of continuous iterations for each byte			
F6	Longest Byte	The byte value that has the longest streak			
F7	Probability Distribution (STD)	Probability Distribution: Is computed by dividing the unigram frequency of each byte in the file by the file size. As a result, the probability distribution produces a 256-length vector, and due to the same reason of the curse of dimensionality, the normal distribution is taken		Video	MP4 MOV
F8	Probability Distribution (MAD)				
F9	Probability Distribution (Mean)				
F10	Hamming Weight	It is computed by dividing the total number of set bits of each chunk by the total number of bits in it		Audio	MP3 M4A
F11	Shannon Entropy	is a complexity and information content-related feature [47], which computes the amount of information in a file			
F12	Unigram Frequencies Vector	It is the 256-length vector of the Unigram frequencies for each file. This feature is out of this paper's scope			

model, while the RFE tool is a wrapper method that wraps a classifier up in a feature selection process.

2) DATASET BALANCING

The biased or skewed dataset leads to imbalanced classification. The imbalanced classification poses a challenge for predictive modeling since most ML classification algorithms were designed to assume an equal number of samples for each class. This challenge appears when the distribution of the dataset is unbalanced [49]. There are many several methods for handling such distribution problems. The general aspect of these methods lay under the main two procedures, over-sampling the minority class or under-sampling the majority class. Some technique combines these two concepts under custom specifications

3) INTERQUARTILE RANG (IQR) METHOD

The IQR is a commonly accepted method for data outliers' detection. The outlier is a data point or observation noticeably different from the rest. They might arise from an error in data measurement or data collection that skews the dataset and renders inaccurate insights. IQR is a suitable statistic method that summarizes a non-Gaussian distribution sample of data assuming that not all data is normal enough to treat it as being drawn from a Gaussian distribution. It calculates the upper and lower bounds of the samples after arranging them in ascending order. These bounds are calculated using formulas 1 and 2. After that, any samples less than the lower bound or greater than the upper bound will be deleted [51].

$$\text{lower} - \text{bound} = \text{Q1} - (1.5 \times \text{IQR}) \quad (1)$$

$$\text{upper} - \text{bound} = \text{Q3} + (1.5 \times \text{IQR}) \quad (2)$$

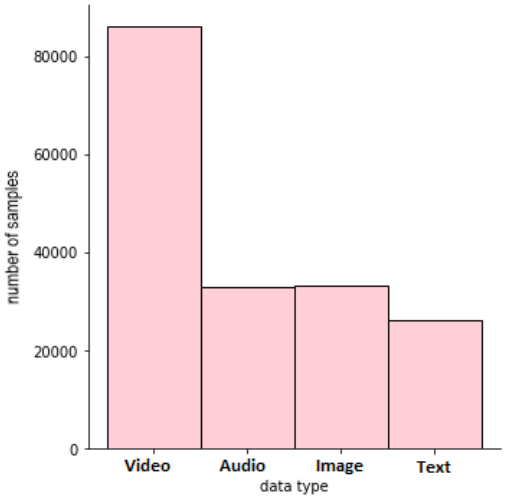

FIGURE 1. The distribution of the main type of the [22] dataset.

Q1 and Q3 are the first quarter (25th percentile) and third quarter (75th percentile), respectively, and the IQR is the difference between Q3 and Q1.

V. METHODOLOGY OF LSDSTRATEGY

The main issue of the proposed LSDStrategy is to identify the type of file data before it arrives completely to provision and allocate, for example, computing resources and storage space in advance. Despite the availability of traditional methods of finding multimedia file types utilizing file metadata or file extension using commercial tools. These tools examine the data blocks' magic numbers (e.g., file signatures and footers), file system metadata, extensions, or packet header

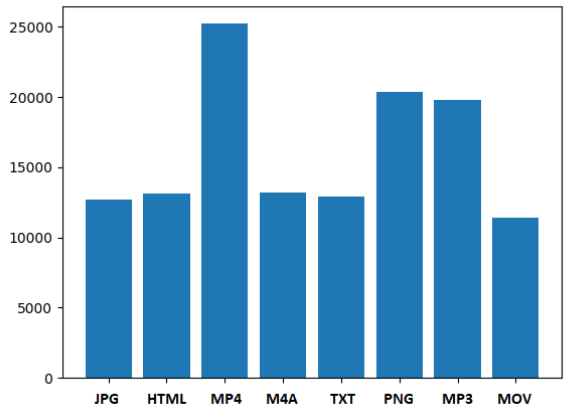

FIGURE 2. The distribution of the sub-type of the [22] dataset.

information. This method that relies on file signatures or other characteristics designed to identify data type becomes ineffective when the data is corrupted or missing. Due to dealing with a stream of chunks, content-based analytics is the best method for data type identification that does not consider magic numbers. To achieve that, the ML supervised classifiers are utilized due to the statistical classification capabilities of ML [52].

The proposed LSDStrateg implements three phases that interact with each other to fulfill the required objective. The first phase, termed the learning phase, accommodates data processing and feature engineering and builds an optimal classification model to predict the big data workload. The second phase, the deployment phase, deploys the selected model that dynamically classifies the workload based on the statistical analysis before it completes arrival. Hence, the output of this phase is a time-series dataset to be used in the last phase. This final phase, termed the forecasting phase, predicts the future load. This forecasting type will serve the provisioning and configuring resources by expecting the workload that arrives in the next time interval. Consequently, it will overcome the resources' starting time consumption and scale resources up and down, an essential issue in big data analysis and processing. Fig. 3 shows the overview of the proposed LSDStrateg.

The detailed work of the first two phases is presented in Sections 5.1 and 5.2, respectively. The third phase is beyond the scope of this paper.

A. LEARNING PHASE

This stage utilizes the previously pre-constructed content-based dataset after pre-processing, which will be the first utilization of this dataset. It uses the processed data to learn three supervised ML models using the machine learning pipeline. These models are Decision Trees (DTs), K-Nearest Neighbor (K-NN), and Random Forest (RF). Besides, the dataset used in this phase had been previously evaluated in terms of completeness, duplication ratio, normality, and missing value [22]. The quality assessment results appear in Table 4.

The raw dataset contained 178,031 samples. Here, the outliers were detected and removed from the data, bringing

TABLE 4. The quality assessment results of the dataset [22].

Assessment	Results	Discerption
duplication rate	5%	Acceptable
Completeness ratio	100%	completed
Missing Values	0	Not- biased
Normality	4.0498	Right-skewed

the number of samples to 128,637 in the prepared dataset. The input parameters at this phase are the models' configuration and the prepared data. The ML models have trained with 80% of the dataset, i.e., 102,909 samples, and tested with 20%, i.e., 25,728 samples. Each ML model has a set of sensible default parameters that cannot be considered optimal for any problem.

Furthermore, it is impossible to determine the best hyper-parameters ahead of time. So, the tuning process for a model is trial-and-error-based engineering. The tuning process uses a specific set of hyperparameters to perform and iteratively evaluate the models through trial and error. Next, it chooses the best collection based on the cumulative observations. When done manually, this process consumes more time than any other activity [53]. So, to decrease the consumed time, the tuning process uses the automatic k-fold cross-validation process provided by the sci-kit learn (i.e., GridSearchCV, the parameter search approach). This search method differs from other parameter search approaches because it exhaustively considers all parameter combinations.

The pipeline's stages of this phase, as shown in Fig. 4, are listed bellow:

1) BALANCING AND OUTLIER DETECTION AND REMOVAL

The dataset distribution avoids building biased ML models. As noted in Fig. 1, the dataset is skewed by class 'Video,' making it the majority class. The idea behind the under-sampling techniques is the removal of samples from the majority class in the training dataset to balance the class distribution in that dataset. The most straightforward implementation of such a technique is randomly removing the majority class samples. Although this method reduces learning time, it poses a limitation where the samples are released without concern for their importance or value in determining the decision boundary between the classes. So, this paper proposes a synthetic under-sampling process to prevent the likelihood of removing useful information. The outliers of the skewed class are detected and removed utilizing the IQR method. Fig. 5 shows the main class distribution after the outlier extraction.

2) FEATURE ASSESSMENT

This step utilizes three feature ranking tools. First, the Mutual Information (MI) method, an information gain ranking metric, provides the statistical correlation between the features

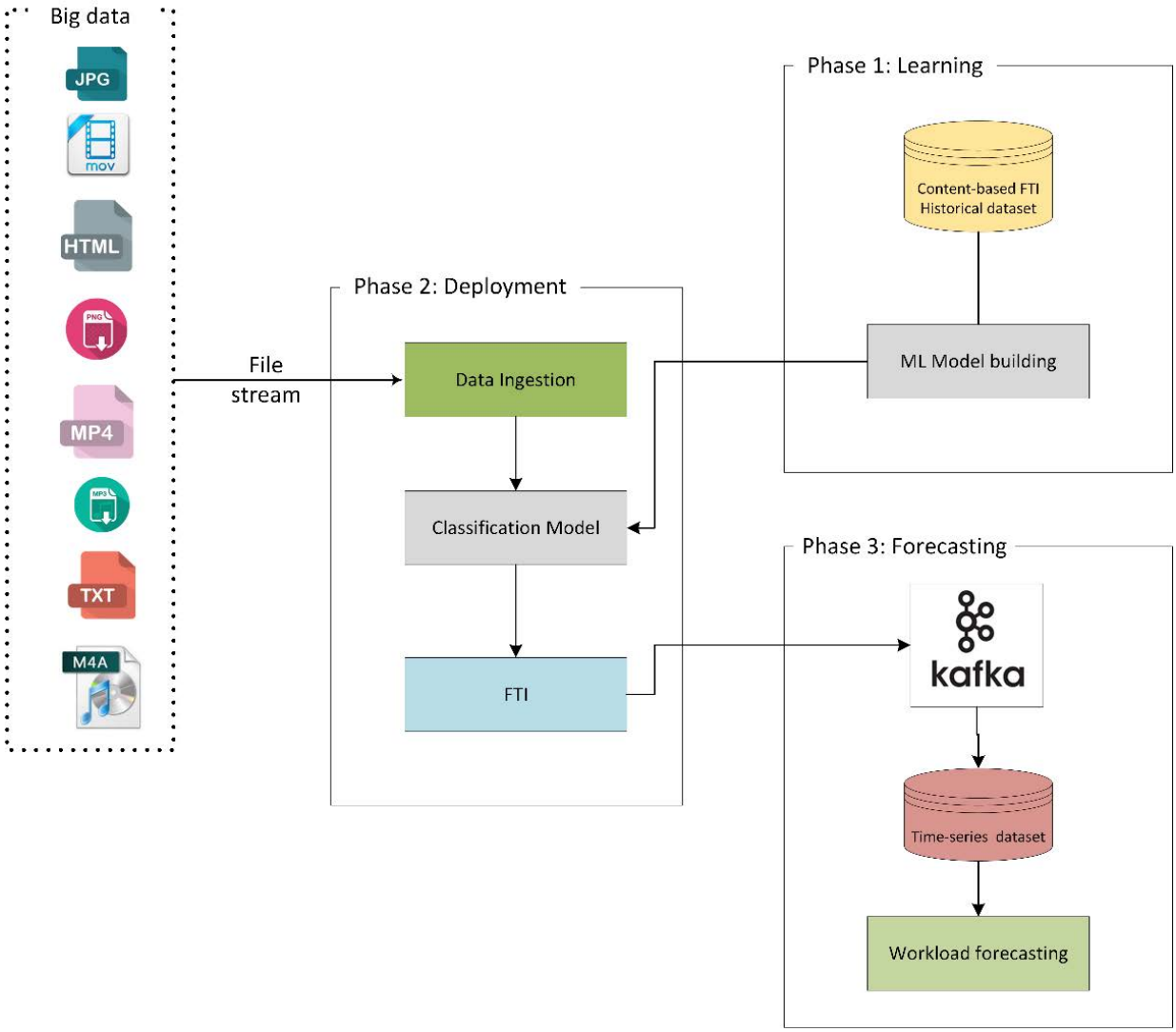

FIGURE 3. The overview of the proposed LSDStrategy.

and the target. This method measures the entropy drops under the condition of the target value in the range from 0 to 1. The higher the value feature, the closer the connection will be to the target. Fig. 6 shows the MI score of all the extracted features. The results from this tool show that all the features have approximately the same connection to the target and have no significant differences between their scores.

Second, another ranking metric, univariate selection, is utilized. This metric is a statistical test that provides the most vital relationships' independent features with the dependent feature. The chi-squared test was used, which measures dependence between stochastic variables that should have non-negative values. In other words, it evaluates their likelihood of association using their frequency distribution. Table 5 shows the results of this test. From the displayed results, the unigram frequency-related features, longest streak, and longest byte features have the highest scores.

Last, it uses the RFE method, a wrapper method, that makes a recursive removal of the features and builds a model on those that remain through an external estimator. This estimator assigns weights to features to identify which ones contribute the most to predicting the target attribute, with the lowest value being the highest weight. The scoring results of this tool will be embedded with the ML model building due to the nature of the wrapper method. Table 6 shows the ranking for the features based on the model. The KNN model does not provide logic to make feature selection, so we did not test it using this method.

3) FEATURE SELECTION

The experiment result shows that the best collection of features is the mutual features from the MI and RFE methods: probability distribution (STD), Longest Streak, Longest Byte, unigram frequencies (MAD), Hamming Weight, and Shannon Entropy.

4) HYPERPARAMETER TUNING

Three supervised ML classifiers by sci-kit learn [54] are adopted: DTs, K-NN, and RF. These models are optimized

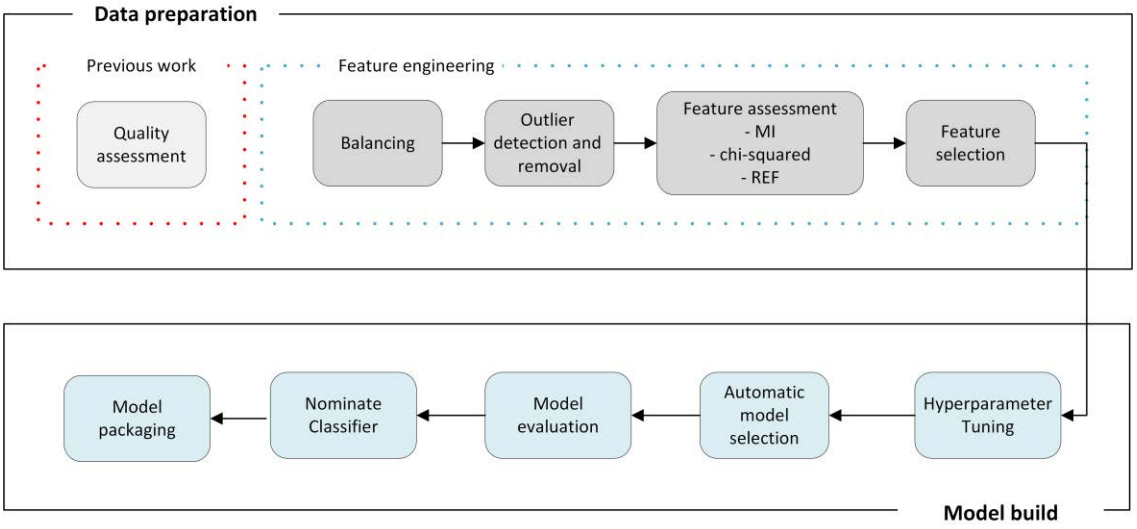

FIGURE 4. The proposed ML pipelines.

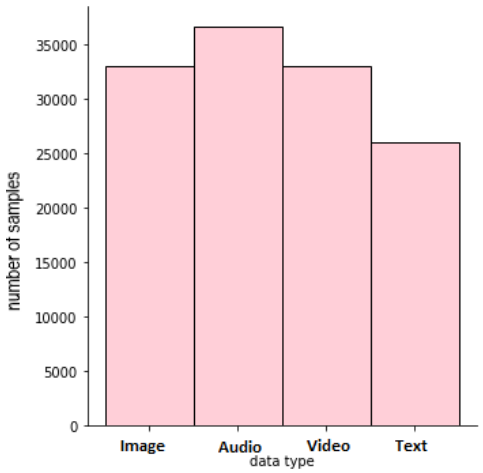

FIGURE 5. The distribution of main classes after outliers drop.

by searching for high-impact parameters to configure each model. That is done through fitting 4-folds for each of 54 candidates of DTs parameters totaling 216 fits, 24 candidates of K-NN parameters totaling 96 fits, and 324 candidates of RF parameters totaling 1296 fits. As an essential point, the consumption time of the 4-fold cross-validation decreases through parallel programming by utilizing all the cores of the used PC to improve the performance of our ML pipeline.

5) AUTOMATIC MODEL SELECTION

The best hyperparameters of each model are automatically selected based on the accuracy as a scoring metric. Tables 7, 8, and 9 show the best set of the searched hyperparameters of the DTs, K-NN, and RF models, respectively.

6) MODEL EVALUATION

A comparative study among three ML classification models based on accuracy, precision, recall, f1-score, and confusion matrixes creates an optimal classification model for the research problem. The primary metrics used in the comparison process are mean accuracy and time consumed. To make a fair evaluation, all the automatically selected models from the previous step are trained with 80% and tested by a non-seen 20% of the dataset. The classification report for the adopted ML models uses three evaluation metrics: precision, recall, and F1-Measure. *Precision* is the ratio of the total number of true positives to the sum of the true positives and false positives, as in the following equation:

$$Precision = \frac{\sum_i \text{TP}}{\sum_i (\text{TP+FP})} \tag{3}$$

While *Recall*, which is known as the true positive rate, is defined as the ratio of the total number of true positives to the sum of the total number of true positives and false negatives and calculated using (4).

$$Recall = \frac{\sum_i \text{TP}}{\sum_i (\text{TP+FN})} \tag{4}$$

Moreover, the F1-score provides an average metric for Precision and Recall. The F1-score is the harmonic mean between Precision and Recall. It is represented mathematically as:

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

$$Accuracy = \frac{\sum (\text{TP} + \text{TN})}{\sum (\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{6}$$

Tables 10, 11, and 12 show the classification report for the DT, K-NN, and RF classifiers, respectively. The macro avg is the arithmetic mean of all the per-class scores for each metric precision, recall, and F1-score. This method applies the same treatment to all classes, regardless of their support values. The weighted average is calculated by averaging all per-class ratings while considering each class's support. The class support represents the number of actual class instances in the dataset.

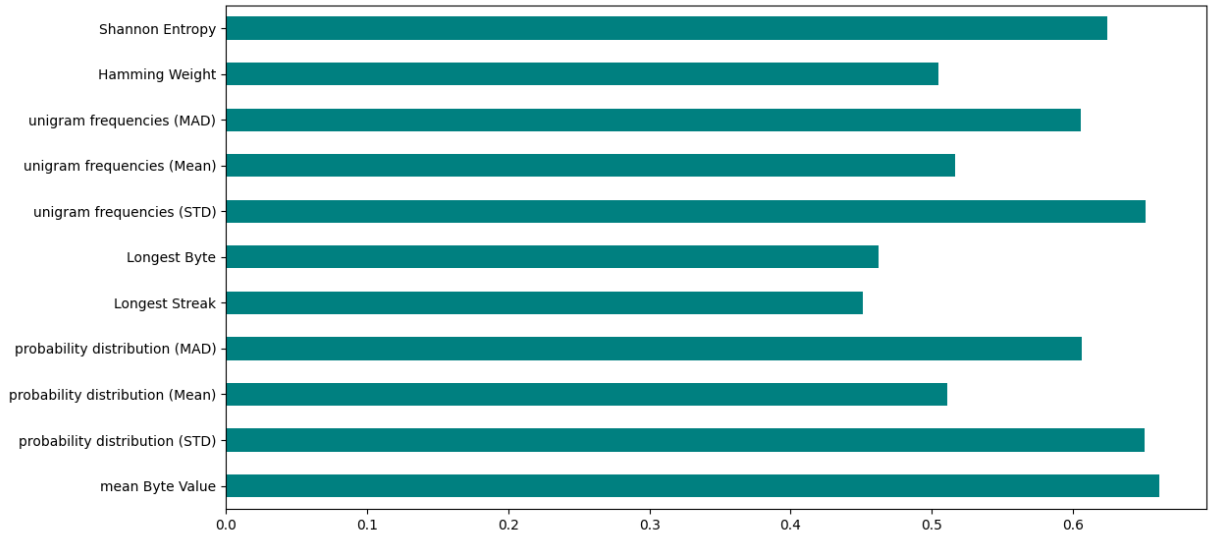

FIGURE 6. MI scores.

TABLE 5. Univariate selection scores (chi-squared).

Feature Name	score
mean Byte Value	952.9147142
probability distribution (STD)	5.524306208
probability distribution (Mean)	14.51625372
probability distribution (MAD)	2.677912567
Longest Streak	**7374.18003**
Longest Byte	**2133.313425**
unigram frequencies (STD)	**2828.523035**
unigram frequencies (Mean)	**7432.382778**
unigram frequencies (MAD)	**1371.097803**
Hamming Weight	1.00009065
Shannon Entropy	53.14388457

The comparative results for the ML classifiers are produced on two levels. In the first level, a comparative study based on the proposed feature sets using the three adopted ML models is performed to find the best features set to the corresponding model. Then, in the second level, a performance comparison between models is discussed according to the best results in the first level. This comparison is based on accuracy and complexity. The complexity term in our study stands for the number of the selected features and the time consumed. Table 13 shows the evaluation report for the three classifiers.

7) NOMINATE CLASSIFIER

The model that advances from this stage is the one that obtains a higher performance than the others. This stage nominates the evaluated model that has the highest nominating rate. The nominating rate for each model is computed using (7). Equation (7) takes the prediction accuracy and time from the

TABLE 6. RFE ranking score based on the models.

Feature Name	DT	RF
mean Byte Value	1	1
probability distribution (STD)	**4**	**2**
probability distribution (Mean)	1	1
probability distribution (MAD)	1	1
Longest Streak	**5**	**5**
Longest Byte	**2**	**3**
unigram frequencies (STD)	1	**6**
unigram frequencies (Mean)	1	1
unigram frequencies (MAD)	**6**	1
Hamming Weight	1	**4**
Shannon Entropy	**3**	1

evaluation metrics report (shown in Table 13) to calculate the nominating rate for each model. The nominating results based on many tests for the models appear in Table 14. As stated, the performance metrics of this paper are a trade-off between accuracy and time consumption. So, in our pipeline, a 50:50 priority for prediction accuracy and time are the parameters.

$$\mathrm{NR}\,(M_i) = (\alpha \mathrm{x}\, M_i\,[\mathrm{Acc}]) + \left(\beta \mathrm{x}\, \left({}^{1}\!/_{M_i\,[T]}\right)\right) \qquad (7)$$

where M_i is the classification model. M_i[Acc] and M_i[T] are the prediction accuracy and time of the corresponding model. At the same time, α and β are the coefficient parameters. The Nominating Classifier procedure steps are summarized in Algorithm 1.

8) MODEL PACKAGING

Also known as model serialization, it is the process of converting a final machine learning model into a specific format (e.g., PMML, PFA, or ONNX) that defines the model that consumed by a commercial application. ML models can be distributed in a variety of formats. The ML model must be present and executable as a separate asset, so the ML models should be usable outside the model-training environment [55]. The model that has advanced to this stage is the one that obtains a high nominating rate among others (see Table 14).

Algorithm 1 Nominating Classifier

procedure Nominating the Optimal Model Based on The Formulated Metrics using Equation ()
Input: Class List of Evaluation Reports (Ev_List) for i Classifier that comprises Model Name (M_i) and its Prediction Accuracy (Acc_i), and Prediction Time (T_i)
Output: Nominated Classifier (NC)
/∗ *Compute the Nominating Rate (NR) for each* M_i ∗/
for each M_i **in** Ev_List **do**
    NR = (M_i [Acc_i] × 0.5) + (1/(M_i [Acc_i]) × 0.5);
    /∗ *generate a class list of classifier Rating (Rate_List)* for i Classifier that comprise Model Name (M_i) and its Nominating rate (R_i) ∗/
    Rate_List.add(NR);
end for
/∗ *Search for the classifier that holds the highest Nominating rate* ∗/
for each R_i **in** Rate_List **do**
    NC= Highest Rate R_i;
end for
return NC;
Exit.

TABLE 7. The best set of the searched DTs hyperparameters.

No.	Hyperparameter Name	Hyperparameter Value
1	attribute selection measure	entropy
2	maximum depth of the tree	20
3	maximum leaf nodes	125
4	minimum number of samples needed to be considered a leaf node	50
	Train Accuracy (%):	84.73
	Test Accuracy (%):	84.74
	Search Time (sec):	53.3

TABLE 8. The best set of the searched K-NN hyperparameters.

No.	Hyperparameter Name	Hyperparameter Value
1	Number of neighbors (K)	55
2	Weight function used in prediction	distance
3	Power parameter for the Minkowski metric.	1 (this is equivalent to using manhattan_distance (l1))
	Train Accuracy (%):	99.9
	Test Accuracy (%):	73.4
	Search Time (sec):	150.8

Thus, the proposed DTs model is packaged and used in the deployment phase. The pickle package is used to save the model as a pickle python object with a.pkl file extension.

B. DEPLOYMENT PHASE

As stated earlier, the main aim of this paper is to estimate the workload of the received big data stream based on Variety. As characteristics of big data, various data sources generate the workload. Combining these data sources will lead to an increase in the workload Variety. That, in turn, can result in difficulty tracking the non-homogeneous data quality landscape and the interaction across datasets. Therefore, the data elements from the different sources will come in various forms, such as images, video, audio, and text.

TABLE 9. The best set of the searched RF hyperparameters.

No.	Hyperparameter Name	Hyperparameter Value
1	The number of trees in the forest	125
2	maximum depth of the tree	190
3	Bootstrap	True
4	minimum number of samples required to split an internal node	10
5	Minimum leaf node's samples needed	2
6	max number of features considered for splitting a node	sqrt
	Train Accuracy (%):	93.9
	Test Accuracy (%):	85.4
	Search Time (sec):	68.5

TABLE 10. DT classifier results.

Class	precision	recall	f1-score	support
image	0.810215	0.629774	0.708689	6650
text	0.999225	0.998645	0.998935	5166
video	0.782176	0.949218	0.857639	7286
audio	0.845438	0.836251	0.840819	6626
accuracy			0.847481	25728
macro avg	0.859263	0.853472	0.851521	25728
weighted avg	0.849298	0.847481	0.843179	25728

TABLE 11. K-NN classifier results.

Class	precision	recall	f1-score	support
image	0.629423	0.526917	0.573627	6650
text	0.999419	0.999613	0.999516	5166
video	0.662525	0.819654	0.732761	7286
audio	0.776589	0.700875	0.736792	6626
accuracy			0.749534	25728
macro avg	0.766989	0.761765	0.760674	25728
weighted avg	0.750991	0.749534	0.74623	25728

TABLE 12. RF classifier results.

Class	precision	recall	f1-score	support
image	0.82801	0.627669	0.714054	6650
text	0.99942	1	0.99971	5166
video	0.789064	0.946747	0.860744	7286
audio	0.839286	0.858286	0.848679	6626
accuracy			0.852184	25728
macro avg	0.863945	0.858175	0.855797	25728
weighted avg	0.854303	0.852184	0.847625	25728

TABLE 13. Evaluation report for the predictive models.

		Raw dataset (177951samples)		Pre-processed dataset (128639 sample)	
		Without features selection (eleven features)	With features selection (Six features)	Without features selection (eleven features)	With features selection (six features)
DT	Accuracy (%)	82	81	82	**85**
	Training/testing time (sec)	2.12	1.16	1.09	**0.93**
K-NN	Accuracy (%)	73	72	73	**75**
	Training/testing time (sec)	7.72	5.15	3.4	**2.87**
RF	Accuracy (%)	82	82	85	**85**
	Training/testing time (sec)	10.4	8.36	7.19	**5.9**

Moreover, due to the basic types of big data analytics, including image, video content, audio, and text analytics, estimating the elementary streams becomes an important goal. To achieve this goal, initially, the incoming stream will be chunked to a predefined size. Then, small chunks of the data stream are taken. These chunks are statistically analyzed to predict the workload variety. The high-level diagram of this phase appears in Fig. 7. Three modules were constructed in this phase. The detailed workflow of each module appears in the following sections. Each module is processed as a cooperative thread to be run in parallel with negligible runtime to provide real-time processing for the received stream.

As shown in Fig. 3, this phase is composed of three stages. The first stage concerns the ingestion process of the big data stream. The first module, Da-Ing, accomplishes this process. Moreover, the base of the second stage, the Classification Model, is the usage of the packaged model nominated by the learning phase. The Wo-IdF module is responsible at this point for classifying the workload with the help of the Fe-Ext module. Finally, the File Type Identification (FTI) stage forms the Wo-IdF module output. That output will be a time series consisting of the type of workload received at the specified time to be used directly in the forecasting phase.

1) DATA INGESTION (DA-ING)

Here, the Bottle framework provides an interface to receive lightweight requests quickly, simply, and efficiently [56]. It provides a distributed quality as a single file module with no dependencies other than the Python Standard Libary. Moreover, it is a Web Server Gateway Interface (WSGI) micro web framework that is well-suited to building a Representational State Transfer (REST) API. REST is a software architecture approach that allows heterogeneous computer systems connected via the internet to communicate with one another. That REST API coordinates all communication between our classification model and incoming requests with an HTTP request. The bottle service can be accessed using the web server's IP address and port number without an extension: http://127.0.0.1:8000/.

TABLE 14. Nominating rate results.

Classifier	Nominating Rate (Accuracy: Prediction Time)				
	30:70	40:60	50:50	60:40	70:30
DTs	**1.00**	**0.9841**	**0.961**	**0.9385**	**0.9158**
K-NN	0.468	0.5088	0.5489	0.5890	0.6292
RF	0.374	0.4425	0.5108	0.5791	0.6473

The Bottle v0.12.19 server applies this stage. Fig. 8 shows the interaction diagram of the DaIng module.

Here, the use case of file upload streaming is demonstrated when the file is chunked into smaller pieces. This use case is the best performance improvement when dealing with a distributed file system, distributed databases, distributed computing, and limited internet capacity [57], [58]. The chunk size that is adopted here is 512 bytes. So, if the streamed file size is larger than the required size, it will be chunked and sent through the internet. Else, it will be sent as a whole. Then, a server recombines the file's chunks. Estimating the Variety of big data workloads is done before it reaches the server completely. So, receiving requests takes place, in the background as a parallel process, in isolation from the analysis process.

2) FEATURES EXTRACTION (FE-EXT)

The best collection of features selected in Section (5.1) is extracted online from the ingested chunks. To achieve that, we construct a module-based multithreading program to reach the maximum performance of processors [59] In this module, each chunk's probability distribution (STD), longest streak, longest byte, unigram frequencies (MAD), Hamming weight, and Shannon entropy are calculated in parallel to decrease the consumption time of calculation. The equations for the extracted features are below, and Table 15 summarizes the notations used in these equations.

- *Unigram Frequencies (MAD):* This feature is the mean absolute deviation of the occurrence frequency of the value of a byte in a file, computed for each chunk using (8).

$$\mathrm{MAD}(U) = \frac{1}{N}\sum_{i=0}^{N-1} |U[i] - \mu(U)| \tag{8}$$

- *Probability Distribution (STD):* This is the standard deviation of dividing the results of the unigram frequency of each byte in the file by the file size. The standard deviation for the probability distribution for each chunk is computed using (9).

$$\sigma(P) = \sqrt{\frac{\sum (P[B] - \mu(P))^2}{N}} \tag{9}$$

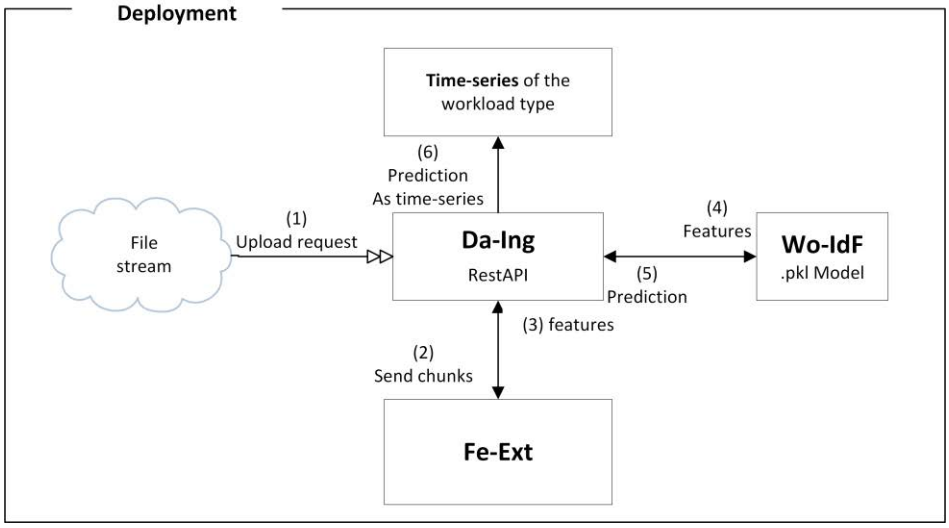

FIGURE 7. The high-level diagram of the deployment phase.

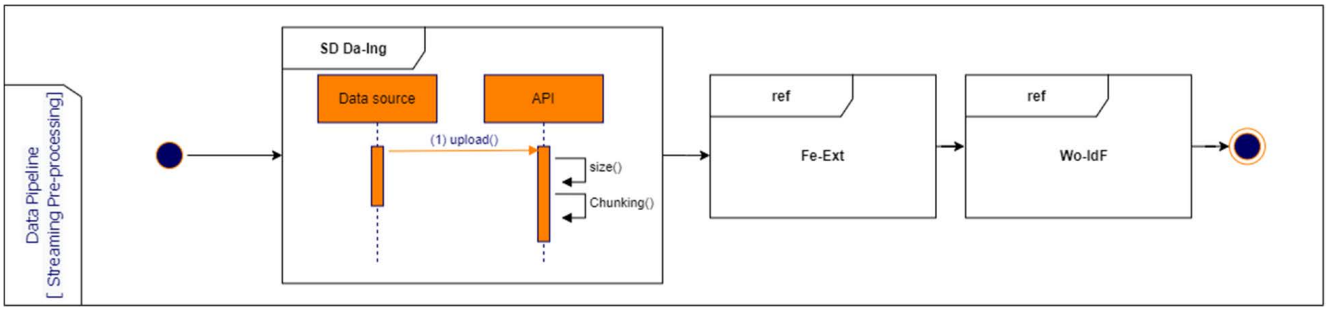

FIGURE 8. The interaction diagram of the Da-Ing module.

$$P[B] = \frac{U[B]}{X} \tag{10}$$

- *Longest Streak:* : It is computed by counting the continuous iterations for each byte in the file and finding the largest one
- *Longest Byte:* This is the longest byte value that constructed the longest streak feature.
- *Hamming Weight:* This feature is computed as in the equation below [22]:

$$\mathrm{H} = \frac{\text{total no. of ones in W}}{\mathrm{x}} \tag{11}$$

- *Shannon Entropy:* The chunk's Shannon entropy is another feature [47]. It is calculated as [60] using (12):

$$\mathrm{E} = -\sum_{\mathrm{i}=0}^{\mathrm{N}-1} P\,[i] \log_2 P\,[i] \tag{12}$$

The extracted features are funded to the third module by the Da-Ing for dynamically classifying the incoming workloads before reaching the server in its entirety. Fig. 9 shows the interaction diagram of the Fe-Ext module.

3) WORKLOAD IDENTIFICATION (WO-IDF)

The primary task of the Wo-IdF falls under the central issue of this paper, which is estimating the Variety in big data workloads. It determines the big data variety characteristic according to its demand. The challenges of Variety include diversity in requests, data to be stored or processed, the job to be scheduled, and so on, which the traditional tools and algorithms cannot efficiently process. The Wo-IdF utilized the packaged model (in Section 5.1) as a predictive model to identify the workload of the incoming stream before it arrives completely. As stated earlier, the estimation of workload type is the primary input to the forecasting algorithm. Fig. 10 shows the interaction diagram of the Wo-IdF module.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

This section discusses the results of testing the performance of the LSDStrategy. The experiments ran on Windows 10 Pro with 12 GB RAM and 2.00 GHz Intel(R) Core (TM) i7. The implementation used the Python programming language with the necessary data manipulation, analysis libraries, and the sci-kit learn package to model ML projects.

Since the main objective is the anticipation of knowing the type of data before it fully arrives on the server and creating time-series data to serve the data scientists for the forecasting process, the HTTP server base Bottle framework is designed. Raw data file corpora collected from the Garfinkel [61] and GaRoFou [62] file corpus and audio files manually are used to produce a stream of variant workload. To create several

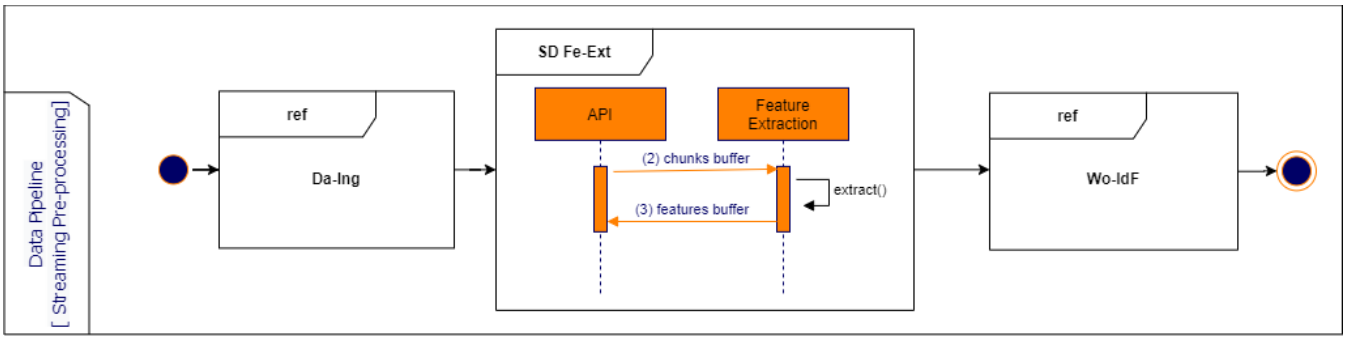

FIGURE 9. The interaction diagram of the Fe-Ext module.

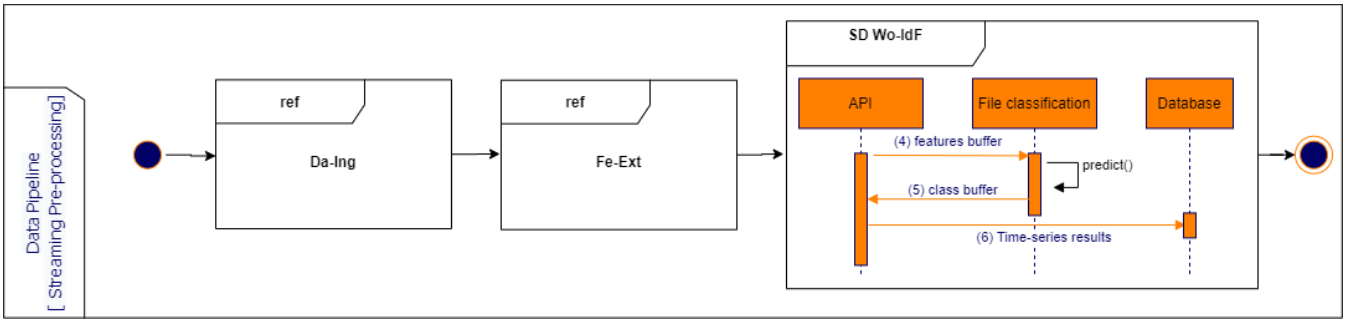

FIGURE 10. The interaction diagram of the Wo-IdF.

TABLE 15. Notations used in feature extraction.

Notation	Brief Meaning
U	The unigram frequency array for a chunk with a length of N
N	Length of U, which equal to 2^8
B	Byte value in (in decimal form), $B = 0, 1, 2 \ldots\ldots 255$
μ	Mean value
P	probability distribution of the bytes in the U array, it is an array of the length of N
σ	Standard deviation
X	Size of the chunk in byte
x	Size of the chunk in a bit (i. e. X * 8)
W	An array composed of the parsed chunk in bit-wise, its length is equal to x

streaming sources, multiple virtual local HTTP clients post requests to the Da-Ing using the http://127.0.0.1:8000/ URL and the file corpora. The mentioned corpora are composed of various data types of multiple sizes.

Since the streaming data is stochastic and due to the differences in the sampling rate of the stream-generating devices, adopting an adaptive window is essential. In addition to choosing the best window that fits the time consumed by the Fe-Ext and Wo-IdF modules, several static sliding window sizes are tested to provide a broad view of the modules. Here, the window size reflects the total number of chunks where the chunk size adopted in this experiment is 512 bytes. The results show that the time consumed in feature extraction exponentially increases with the size of the window, as shown in Fig. 11. This experiment runs on a single server, and the time will decrease when utilizing a distributed processing tool such as Apache Spark or Storm.

The Wo-IdF time consumption versus window size results is shown in Table 16. From the results of the Wo-IdF testing, the accuracy ranges from 76%-83%, with no noticeable dependence on the window size. The accuracy fluctuation depends on the variance in the distribution of online streaming.

The computational complexity of the Wo-IdF module depends on the packaged model. In this paper, an optimized DT model based on the CART (Classification and Regression Trees) algorithm has been nominated as the main classifier of the received load. A balanced binary tree typically takes $O(s * f * (\log s))$ and query time $O(\log s)$ to construct in run time where s is the number of samples and f is the number of attributes in a training dataset. The tree creation algorithm attempts to produce balanced trees, but this is not a guarantee. The cost at each node is determined by searching across $O(f)$ to locate the feature that yields the greatest reduction in the impurity criterion, such as log loss, assuming that the subtrees remain roughly balanced. This has an $O(f * s * (\log s))$ cost for each node, resulting in an $O\left(f * s^2 * (\log s)\right)$ total cost for the entire tree (by adding the cost per node). The space complexity is based on the maximum depth (d) of the constructed tree. Specifically, the storage required is $O(d)$ [54].

The performance definition of this work is based on accuracy and time metrics. So, from the experimental results, the Wo-IdF module is highly recommended for variety estimation in big-data multimedia streaming based on streaming content analysis. Also, the Fe-Ext module test results show the dependency of the module on the sliding window size. So, this paper recommends running this module in a distributed fashion using a distributed framework utilizing

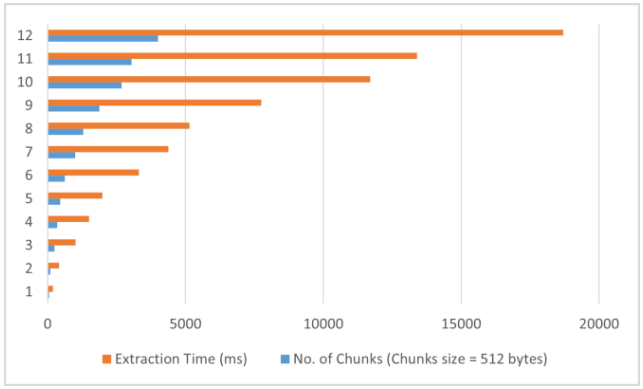

FIGURE 11. Fe-Ext time consumption vs. sliding window size.

TABLE 16. Wo-IdF time consumption versus window size results.

No. of Chunks (Chunks size = 512 bytes)	Prediction Time (ms)	Accuracy
50	3.9	78%
100	3	79%
250	2.9	86%
350	3.9	80%
464	3.9	79%
625	3.9	83%
1000	2.9	83%
1300	1.9	81%
1875	4.9	81%
2684	3.9	82%
3047	4.9	82%
4004	3.9	82%
6014	3.9	81%
Average	**1674.077**	**3.676923**	**81.3%**

cloud computing. The heatmap in Fig. 12 describes the correlation between the image and video data. Such correlation is inconsiderable in resource management due to the close relationship between the required resources for these two data types.

The main objective of the proposed LSDStrategy is to predict the type of multimedia data based on its content analysis. It uses an ML algorithm that will overcome the limitation of the semantic and non-semantic parsing (e.g., non-ML) approaches due to the statistical classification capabilities of ML [32]. This type of analysis is the first, as far as we know, to analyze streaming data. Moreover, the validity of the proposed LSDStrategy is proven by a comparative analysis with the recent non-ML approach. Navroop et al. [7] faced the workload variety of big data by identifying the workload by examining the data blocks' magic numbers (file extensions). This method that relies on file signatures designed to recognize its type becomes useless with corrupted or missing data. Moreover, this method is challenging to implement when dealing with big data streams in the real

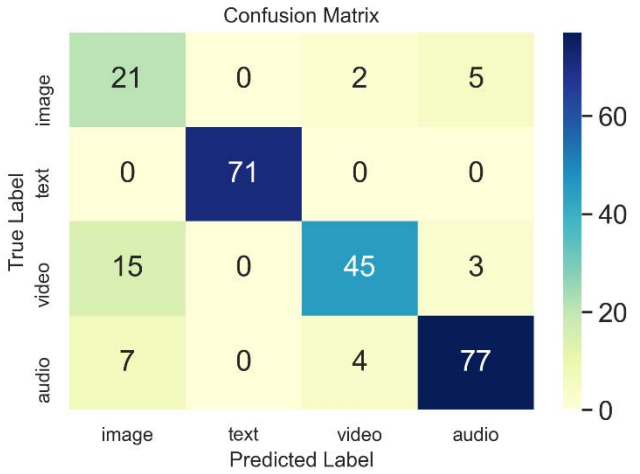

FIGURE 12. Wo-IdF prediction heatmap.

world. The authors of [7] assume that the files are received completely with their metadata and without loss at the time of analysis, which is impossible to guarantee in real time. Our approach analyzes any part of the workload without the whole file and determines its type without metadata. This approach increases the speed of predicting the type of the workload and its use in future workload forecasting before the full load reaches the server.

Furthermore, in other research [6], the proposed solution for Variety prediction utilized the file identification tool. The authors considered the state of receiving data as packets over IP networks. They used the information about the type of data carried by the packets. Each packet consists of a packet header which contains a field called Payload Type (PT). While in-depth, not all transport protocols provide the PT. Besides, the big data stream carried out a variety of elementary streams that enforce dealing with several transport protocol types. Dealing with multiple types of protocol negatively impact the Variety prediction of the workload.

Therefore, the above non-ML approaches, based on the semantic parsing of the workload, have clear limitations compared to our proposal [22].

VII. CONCLUSION AND FUTURE WORK DIRECTIONS

This paper discusses definitions of the characteristics of big data, depending on different domains. Moreover, it proposes a novel strategy named LSDStrategy that analyzes the received multimedia stream based on its binary content using machine learning techniques with artificial and real datasets. LSDStrategy utilizes an evaluating voting technique, selecting the optimum classifier to vote for a decision. It evaluates multi-classifiers, including Decision Tree (DT), K- Nearest Neighbor (K-NN), and Random Forest (RF), over multi-content-based features. Experiments rated the performance of the adopted models and the selected features. According to experimental analysis, the DT approach provides a consistent performance for both artificial and real-world datasets of 85% and 81.3%, respectively. We deploy the LSDStrategy and evaluate its efficiency on a regular specification server

through a set of experiments using a synthetic stream. The experiments prove the LSDStrategy agility and adaptivity in identifying the multimedia-based workload type utilizing small chunks of load.

The following conclusions are based on the problem formulated in this paper and the obtained results.

1. In a world that generates stochastic data in multiple mediums and fields, there is no specific definition of big data characteristics. Thus, it is critical to formulate and define the problem generated by big data before preparing an optimal solution. So, this paper designed a problem that arose from the Variety of big data and proposed the best solution.
2. Given that the growth and diversity of big data are faster than the development of hardware components, it is essential to emphasize the development of software-driven models to reduce the burden on hardware. So, we proposed a lightweight software-driven strategy named LSDStrateg based on statistical computing and machine learning capabilities.
3. Reducing the number of extracted features is essential when analyzing a stream of data in real time. Therefore, evaluating the features and selecting the optimal one is necessary. Three evaluation methods prove which features best correlate to the target variable. Mutual features with the best score from the MI and RFE methods were confirmed as the best-selected features.
4. To extract value from the analyzed data, and because dealing with streaming data requires real-time processing, the adoption of metrics must regard a trade-off between accuracy and time restraints. Therefore, this work conducted a comparative study among three machine learning models using our real pre-constructed dataset, and it nominated the model that guarantees the approved metrics.

In future work, we hope to do the following:

- Deploy the LSDStrateg in a distributed environment to accelerate the computation of the Fe-Ext module.
- Propose a resource management system that uses the results of the classifications extracted from LSDStrateg to forecast the workload in the future and use it as a parameter for the resource management system.

REFERENCES

[1] S. A. Dheyab, M. N. Abdullah, and B. F. Abed, "A novel approach for big data processing using message passing interface based on memory mapping," *J. Big Data*, vol. 6, no. 1, pp. 1–17, Dec. 2019.
[2] N. A. Zghair, A. R. Taresh, and G. M. Abdulsahib, "A modified algorithm to enhance Vanet performance," *J. Southwest Jiaotong Univ.*, vol. 54, no. 4, pp. 1–9, 2019.
[3] D. D. Khudhur and M. S. Croock, "Application of self-managing system in greenhouse with wireless sensor network," *Iraqi J. Comput., Commun., Control Syst. Eng.*, vol. 21, no. 1, pp. 53–61, Feb. 2021.
[4] D. D. Khudhur and M. S. Croock, "Physical cyber-security algorithm for wireless sensor networks," *TELKOMNIKA (Telecommun. Comput. Electron. Control)*, vol. 19, no. 4, p. 1177, Aug. 2021.
[5] M. S. Croock, Z. A. Hassan, and S. D. Khudhur, "Adaptive key generation algorithm based on software engineering methodology," *Int. J. Electr. Comput. Eng. (IJECE)*, vol. 11, no. 1, p. 589, Feb. 2021.
[6] N. Kaur, S. K. Sood, and P. Verma, "Cloud resource management using 3 Vs of internet of big data streams," *Computing*, vol. 102, no. 6, pp. 1463–1485, Jun. 2019.
[7] N. Kaur and S. K. Sood, "Dynamic resource allocation for big data streams based on data characteristics (5 Vs)," *Int. J. Netw. Manage.*, vol. 27, no. 4, p. e1978, Jul. 2017.
[8] D. Laney, "3D data management: Controlling data, volume, velocity and variety," *META Group Res. Note*, vol. 6, no. 70, p. 1, 2001.
[9] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.
[10] *Total Data Volume Worldwide 2010–2025*. Statista. Accessed: May 9, 2022. [Online]. Available: https://www.statista.com/statistics/871513/worldwide-data-created/
[11] R. A. Razooqi, H. Jaleel, and G. Muttasher, "An enhanced hybrid edge-cloud algorithm for reducing iot service delay," *Iraqi J. Comput., Commun., Control Syst. Eng.*, vol. 21, no. 4, pp. 50–59, 2021.
[12] W.-D. Zhu, *Building Big Data and Analytics Solutions in the Cloud*. Armonk, NY, USA: IBM, 2014, p. 101.
[13] T. Erl, W. Khattak, and P. Buhler, *Big Data Fundamentals Concepts, Drivers & Techniques*. Prentice-Hall, 2016.
[14] S. Kaur, Y. Kumar, and S. Kumar, "Soft computing techniques for energy consumption and resource aware allocation on cloud: A progress and systematic review," in *Advanced Soft Computing Techniques in Data Science, IoT and Cloud Computing*, S. Dash, S. K. Pani, and A. A. Y. Liang, Ed. New York, NY, USA: Springer, 2021, pp. 191–213.
[15] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Inf. Syst.*, vol. 47, pp. 98–115, Jan. 2015.
[16] M.-A. Vasile, F. Pop, R.-I. Tutueanu, V. Cristea, and J. Kołodziej, "Resource-aware hybrid scheduling algorithm in heterogeneous distributed computing," *Future Gener. Comput. Syst.*, vol. 51, pp. 61–71, Oct. 2015.
[17] S. KaurSahi and V. S. D. V. S. Dhaka, "A review on workload prediction of cloud services," *Int. J. Comput. Appl.*, vol. 109, no. 9, pp. 1–4, Jan. 2015.
[18] Y. Hu, B. Deng, F. Peng, and D. Wang, "Workload prediction for cloud computing elasticity mechanism," in *Proc. IEEE Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Chengdu, China, Jul. 2016, pp. 244–249.
[19] J. Kumar and A. K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Future Gener. Comput. Syst.*, vol. 81, pp. 41–52, Apr. 2018.
[20] J. Gao, H. Wang, and H. Shen, "Machine learning based workload prediction in cloud computing," in *Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Honolulu, HI, USA, Aug. 2020, pp. 1–9.
[21] M. Kulkarni, P. Deshpande, S. Nalbalwar, and A. Nandgaonkar, "Cloud computing based workload prediction using cluster machine learning approach," in *Applied Computational Technologies*. Singapore: Springer, 2022, pp. 591–601.
[22] S. D. Khudhur and H. A. Jeiad, "A content-based file identification dataset: Collection, construction, and evaluation," *Karbala Int. J. Mod. Sci.*, vol. 8, no. 2, pp. 63–70, May 2022.
[23] S. Dash, C. Chakraborty, S. K. Giri, S. K. Pani, and J. Frnda, "BIFM: Big-data driven intelligent forecasting model for COVID-19," *IEEE Access*, vol. 9, pp. 97505–97517, 2021.
[24] F. Cappa, R. Oriani, E. Peruffo, and I. McCarthy, "Big data for creating and capturing value in the digitalized environment: Unpacking the effects of volume, variety, and veracity on firm performance," *J. Product Innov. Manag.*, vol. 38, no. 1, pp. 49–67, Jan. 2021.
[25] B. Abu-Salih, K. Y. Chan, O. Al-Kadi, M. Al-Tawil, P. Wongthongtham, T. Issa, H. Saadeh, M. Al-Hassan, B. Bremie, and A. Albahlal, "Time-aware domain-based social influence prediction," *J. Big Data*, vol. 7, no. 1, p. 10, Dec. 2020.
[26] M. Naeem, T. Jamal, J. Diaz-Martinez, and S. A. Butt, "Trends and future perspective challenges in big data," in *Advances in Intelligent Data Analysis and Applications*. Singapore: Springer, 2022, pp. 309–325.
[27] S. S. Farley, A. Dawson, S. J. Goring, and J. W. Williams, "Situating ecology as a big-data science: Current advances, challenges, and solutions," *BioScience*, vol. 68, no. 8, pp. 563–576, Aug. 2018.
[28] K. L.-M. Ang, F. L. Ge, and K. P. Seng, "Big educational data & analytics: Survey, architecture and challenges," *IEEE Access*, vol. 8, pp. 116392–116414, 2020.
[29] MaheshKumar. (Oct. 30, 2013). *Why Variety is the Unsolved Problem in Big Data*. SmartData Collective. Accessed: May 9, 2022. [Online]. Available: https://www.smartdatacollective.com/why-variety-unsolved-problem-big-data/

[30] N. Kaur and S. K. Sood, "Efficient resource management system based on 4 Vs of big data streams," *Big Data Res.*, vol. 9, pp. 98–106, Sep. 2017.
[31] G. Vranopoulos, N. Clarke, and S. Atkinson, "Addressing big data variety using an automated approach for data characterization," *J. Big Data*, vol. 9, no. 1, p. 8, Dec. 2022.
[32] N. L. Beebe, L. A. Maddox, L. Liu, and M. Sun, "Sceadan: Using concatenated N-gram vectors for improved file and data type classification," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 9, pp. 1519–1530, Sep. 2013.
[33] I. Kotenko, K. Izrailov, and M. Buinevich, "Analytical modeling for identification of the machine code architecture of cyberphysical devices in smart homes," *Sensors*, vol. 22, no. 3, p. 1017, Jan. 2022.
[34] E. Mina and S. Jalili, "File fragment type classification by bag-of-visual-words," *ISC Int. J. Inf. Secur.*, vol. 13, no. 2, pp. 101–116, 2021.
[35] N. Beebe, L. Liu, and M. Sun, "Data type classification: Hierarchical class-to-type modeling," in *Proc. IFIP Int. Conf. Digit. Forensics*. Cham, Switzerland: Springer, 2016, pp. 325–343.
[36] K. Vulinovic, L. Ivkovic, J. Petrovic, K. Skracic, and P. Pale, "Neural networks for file fragment classification," in *Proc. 42nd Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, Opatija, Croatia, May 2019, pp. 1194–1198.
[37] M. Bhatt, A. Mishra, M. W. U. Kabir, S. E. Blake-Gatto, R. Rajendra, M. T. Hoque, and I. Ahmed, "Hierarchy-based file fragment classification," *Mach. Learn. Knowl. Extraction*, vol. 2, no. 3, pp. 216–232, Aug. 2020.
[38] G. Mittal, P. Korus, and N. Memon, "FiFTy: Large-scale file fragment type identification using convolutional neural networks," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 28–41, 2021.
[39] M. A. Alsubhi, A. Nour Moussa, A. S. Alfaqiri, and F. Layth Khaleel, "GenSpec: A file fragment classification approach," in *Proc. Int. Conf. Comput. Inf. Technol. (ICCIT)*, Tabuk, Saudi Arabia, Sep. 2020, pp. 1–5.
[40] A. Bhat, A. Likhite, S. Chavan, and L. Ragha, "File fragment classification using content based analysis," in *Proc. ITM Web Conf.*, vol. 40, 2021, p. 03025.
[41] M. E. Haque and M. E. Tozal, "Byte embeddings for file fragment classification," *Future Gener. Comput. Syst.*, vol. 127, pp. 448–461, Feb. 2022.
[42] A. B. Nandyal, M. Rafi, M. Siddappa, and B. B. Sathish, "Improving data services of mobile cloud storage with support for large data objects using OpenStack swift," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 6, pp. 866–876, 2021.
[43] S. Ullah, M. D. Awan, and M. Sikander Hayat Khiyal, "Big data in cloud computing: A resource management perspective," *Sci. Program.*, vol. 2018, May 2018, Art. no. 5418679.
[44] O. Runsewe and N. Samaan, "Cloud resource scaling for big data streaming applications using a layered multi-dimensional hidden Markov model," in *Proc. 17th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, Madrid, Spain, May 2017, pp. 848–857.
[45] A. S. Kaseb, A. Mohan, Y. Koh, and Y.-H. Lu, "Cloud resource management for analyzing big real-time visual data from network cameras," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 935–948, Oct. 2019.
[46] K. Braiki and H. Youssef, "Resource management in cloud data centers: A survey," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Tangier, Morocco, Jun. 2019, pp. 1007–1012.
[47] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, Jan. 2014.
[48] J. Brownlee, *Imbalanced Classification With Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning*. Machine Learning Mastery, 2020.
[49] C. E. Shannon, "A note on the concept of entropy," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.
[50] H. P. Vinutha, B. Poornima, and B. M. Sagar, "Detection of outliers using interquartile range technique from intrusion dataset," in *Advances in Intelligent Systems and Computing*. Singapore: Springer, 2018, pp. 511–518.
[51] S. D. Khudhur and D. D. Khudhur, "IgG-IgM antibodies based infection time detection of COVID-19 using machine learning models," *TELKOMNIKA (Telecommun. Comput. Electron. Control)*, vol. 20, no. 2, p. 340, Apr. 2022.
[52] L. Hiester, "File fragment classification using neural networks with lossless representations," Undergraduate Honors Theses, East Tennessee State Univ., Johnson City, TN, USA, 2018. [Online]. Available: https://dc.etsu.edu/honors/454
[53] F. Pedregosa, G. Varoquaux, and A. Gramfort, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011.
[54] S. Zhao, M. Talasila, G. Jacobson, C. Borcea, S. Anwar Aftab, and J. F. Murray, "Packaging and sharing machine learning models via the acumos AI open platform," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Orlando, FL, USA, Dec. 2018, pp. 841–846.
[55] *Bottle: Python Web Framework—Bottle 0.13-DeV Documentation*. Accessed: May 11, 2022. [Online]. Available: https://bottlepy.org/docs/dev/
[56] P. A. Yadav and K. Vartak, "Google file system," *VIVA-Tech Int. J. Res. Innov.*, vol. 1, no. 4, pp. 1–6, 2021.
[57] I. Hababeh, "A novel cloud computing data fragmentation service design for distributed systems," in *Proc. Athens*, 2011, pp. 1–6.
[58] Y. A. Andrade-Ambriz, S. Ledesma, and D.-L. Almanza-Ojeda, "Multithreading programming for feature extraction in digital images," in *Trends and Applications in Software Engineering*. Cham, Switzerland: Springer, 2020, pp. 208–218.
[59] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 4, pp. 623–656, Oct. 1948.
[60] S. Garfinkel, P. Farrell, V. Roussev, and G. Dinolt, "Bringing science to digital forensics with standardized forensic corpora," *Digit. Invest.*, vol. 6, pp. S2–S11, Sep. 2009.
[61] M. Portaz, J. Poignant, B. Mateusz, P. Mulhem, J.-P. Chevallet, and L. Goeuriot, "Construction et évaluation d'un corpus pour la recherche d'instances d'images muséales," in *Proc. 14th French Inf. Retr. Conf.*, Mar. 2017, pp. 17–34.

SAJA DHEYAA KHUDHUR was born in Baghdad, Iraq, in 1989. She received the B.S. and M.S. degrees in computer engineering from the University of Technology Iraq, Baghdad, in 2010 and 2016, respectively. She is currently pursuing the Ph.D. degree in computer engineering. Since 2016, she has been an Assistant Lecturer with the Computer Engineering Department, University of Technology Iraq. She is the author of 11 articles. Her research interests include the fields of database, machine learning, data science, windows application, and web application.

HASSAN AWHEED JEIAD received the B.Sc. degree in electronics and communications engineering, the M.Sc. degree in communication engineering, and the Ph.D. degree in computer engineering from the University of Technology Iraq, Baghdad, Iraq, in 1989, 1999, and 2006, respectively. He is currently a Lecturer with the Department of Computer Engineering, University of Technology Iraq. His research interests include computer architecture, microprocessors, computer networks, multimedia, adaptive systems, and information systems.