Computers, Materials & Continua
DOI:10.32604/cmc.2022.019962
Article

Tech Science Press

Enhancing Cloud Performance Using File Format Classifications

Muhammad Junaid[1,*], Adnan Sohail[1], Monagi H. Alkinani[2], Adeel Ahmed[3], Mehmood Ahmed[3] and Faisal Rehman[4]

[1]Department of Computing, Iqra University, Islamabad, Pakistan
[2]Department of Computer Science and Artificial Intelligence, College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia
[3]Department of Information Technology, The University of Haripur
[4]Department of Computing, COMSATS University, Abbottabad Campus, Pakistan
[*]Corresponding Author: Muhammad Junaid. Email: mjunaid@uoh.edu.pk
Received: 03 May 2021; Accepted: 15 June 2021

Abstract: Metaheuristic approaches in cloud computing have shown significant results due to their multi-objective advantages. These approaches are now considering hybrid metaheuristics combining the relative optimized benefits of two or more algorithms resulting in the least tradeoffs among several factors. The critical factors such as execution time, throughput time, response time, energy consumption, SLA violations, communication overhead, makespan, and migration time need careful attention while designing such dynamic algorithms. To improve such factors, an optimized multi-objective hybrid algorithm is being proposed that combines the relative advantages of Cat Swarm Optimization (CSO) with machine learning classifiers such as Support Vector Machine (SVM). The adopted approach is based on SVM one to many classification models of machine learning that performs the classifications of various data format types in the cloud with best accuracy. In CSO, grouping phase is used to divide the data files as audio, video, image, and text which is further extended by polynomial Kernel function based on various input features and used for optimized load balancing. Overall, proposed approach works well and achieved performance efficiency in evaluated QoS metrics such as average energy consumption by 12%, migration time by 9%, and optimization time by 10%, in the presence of competitor baselines.

Keywords: Classification; load balancing; optimization; cloud computing

1 Introduction

A massive increase in the volume of data over the years has made the extraction of appropriate features quite difficult. Further, the variety of data in this huge volume is adding computation complexities and requires powerful resources for processing. Cloud computing has gained popularity in handling such massive data and providing an opportunity to the end-users to access data as per their requirements. Both end-users and cloud service providers (CSP) are using the capabilities of the cloud on an agreed pricing model. A large number of custom driven applications are

being developed by the companies in the cloud that emphasis on the integration of their products through the concept of cloud interoperability [1]. The effect of the same has shown huge growth in public cloud computing markets with more than 20% rise during the last five years and is expected to grow further [2]. This growth not only requires a high level of infrastructure resources but also requires approaches facilitating the use of resources from the customers' point of view. From a vendor's point of view, various services such as fault tolerance and virtualization need to be optimized, whereas, from the customer's point of view, the efficient management practices such as proper resource selection options, legal obligations, and uninterrupted service delivery are of much concern [3]. In addition, the relation between environmental impact and numerous factors like energy consumption, cost, throughput, execution, and overhead, demands optimal resource utilization for load balancing in clouds [4]. Similarly, different cloud service models such as SaaS, PaaS and IaaS provides greater flexibility in the current as well in the future setups of the cloud in implementing the afore mentioned QoS metrics [5].

Recently, cloud computing has been expanded to new research areas such as fog computing, trust in cloud computing and serverless computing in which a number of challenges are faced. Few of the challenges concern are auto-scaling of web applications for determining the incoming workload and security of the cloud computing applications well addressed by the PVI (Private Virtual Infrastructure), [6,7]. Other challenges are load balancing, overhead, resource management, reliability, scalability, and optimization. Most metaheuristic approaches cover only one or more parameters of optimization but neglect several crucial variables that play a key role in achieving multi-objective optimization [8]. Moreover, the above-mentioned challenges can be further minimized by the introduction of hybrid metaheuristic techniques solving complex optimization problems in more efficient manner [9–12]. These hybrid metaheuristic techniques combine the relative advantages of two or more approaches without compromising the efficiency of the system [13–16]. Load balancing in the cloud is one of the critical areas that face certain optimization challenges and recent studies have shown that the hybrid metaheuristics approaches are providing optimal solutions [17]. There are a limited number of studies in the literature for multi-objective metaheuristic load balancing approaches. Therefore, there is still a need for multi-objective based optimal solutions for load balancing considering different conflicting objectives.

Recently, machine learning models have made a huge impact in achieving performance efficiency in dynamic load balancing in the cloud [18]. These machine learning models are being integrated with load balancing algorithms resulting in hybrid techniques [19,20]. Classification is one of the important concepts of supervised machine learning which further divides into SVM [21], Discriminant Analysis [22], Naïve Bayes (NB) [23], K-Nearest Neighbor (K-NN) [24], and Neural Networks (NN) [25]. The SVM-based algorithms can manage multiple instances, both categorical and continuous forms. SVM signifies the data set components or data, each having 'n' number of characteristics in an n-dimensional space separated by a margin known as a hyper-plane. Data files are then separated into an n-dimensional space to get the observation of the classification to which they pertain depending on the edge of the hyperplane. The principal components of such algorithms are the assignment, system structure, fitness function, search process, and data analysis. Based on the same components, this research is focusing on the development of new hybrid metaheuristic algorithm named as CFLB (Cat Flow Load Balancing) that employs Support Vector Machine-a classifier, using Cat Swarm Optimization-a scheduling algorithm. Multi-objective QoS metrics such as energy consumption, migration time, and optimization time are considered as performance measurement indicators by the proposed CFLB algorithm.

Classification of multiple file formats in the cloud can achieve maximum accuracy because of accurate data class assignments. This accuracy needs to be verified through validation functions and confusion matrix so that the authenticity of the developed approach is established. One of the types of file format classification is an audio classification that retrieves audio files immediately and makes content classification. It is the method of extracting the best features from audio datasets [26]. The audio classification exists in various forms such as noise, speech, silence, and music, etc. Similarly, deep learning algorithms such as Convolutional Neural Network (CNN) are being used for audio classification [27]. In addition, for extraction and archiving, video datasets require proper categorization and automated classification. This helps to explain the semantic void and the difficulty of computation. Integrated metaheuristic algorithms are used to achieve greater precision in video detection using multiple classifiers, such as ACO, ABC, PSO, etc. [28–30]. A big rise in text documentation has been found to make the extraction process very complicated. Text clustering is used to categorize text documents for text mining, but it is not possible to choose a text element. [31]. Similarly, text classification and text retrieval are used for feature selection in the text files. The metaheuristic algorithms and classifiers such as GA, HS, PSO, NN, and SVM are widely used for text classifications in high dimension space providing high accuracies [32–34]. The same algorithms can also be used in the cloud computing environment for load balancing in high dimension text classification datasets [35]. The classification of images requires the filtering of subsets of image features from broad feature spaces. The identification of optimal features in image classification is a dynamic method that is solved by multiple hybrid metaheuristic techniques such as CSO, GA, ACO, PSO, with SVM, NN, K-NN [36–39]. There are huge image datasets with extensive features and dimensions that are computation intensive. In image classification, feature selection is a common process in which relevant features are extracted from the given datasets and rest of the features are eliminated. This helps in achieving more accuracy in getting requisite features in less computation time. The same classification approach can easily be used in cloud computing for image related datasets.

SVM provides better accuracy in many applications [40–42]. It is a simple classifier that involves only two classes as compared to other classifiers with a large number of classes. The complexity of SVM is measured using a number of dimensions providing unique solutions. Even with high dimensions, SVM performed better due to its good generalization property. In this analysis, SVM is implemented to create the data class over a range of file formats such as audio, video, text, and images in the cloud environment using one-to-many classifications. Previous approaches to data formats such as AWS and PostgreSQL rely only on grouping of data types but not on types of file formats [43]. However, the proposed approach is new using only file format types for classification in the cloud environment and then using the resultant data class into load balancing algorithm CSO for scheduling.

When applied to load balancing, this processed type of data can vastly boost scheduling using QoS parameters such as execution time, throughput time, and optimization time, etc. [44,45]. Similarly, various file formats are mapped into an acceptable class in the suggested solution that has achieved minimal computational complexity during processing. CSO is a metaheuristic scheduling algorithm based on the behavior of Cats proposed by [46]. Over the years, the number of CSO variants has been proposed in different applications [47]. This research is also using improved CSO keeping in view its simple implementation and faster convergence speed that attains better accuracy and global optimum in a smaller number of iterations. This helps in reducing the complexity involved and producing competitive results as compared to other metaheuristics. In this study, the grouping phase is introduced that divides the various file formats into four groups

such as audio, video, image, and text. This helps in achieving an overall reduced time complexity and better performance in the presence of other baselines. The baselines used in this research consist of Min–Min [48], CGSA [49], FSALB [50], PLBA [51], AFA [52] and HCA [53]. Min–Min is meant to decrease the makespan time, but it fails to produce load balancing schedules in tasks. It is usually used in small scale datasets but failed to produce scalability with huge and complex tasks. CSGA provides multi objective solution such as energy, profit and cost for various tasks in the cloud on small number of iterations and tasks but experimentation lacks huge number of tasks execution. FSALB primarily concentrated on reducing contact delays faced by users of machine learning and hence increased response time but lacking a multi-objective perspective. PLBA uses task migration approach to minimize the makespan time but did not consider number of important QoS parameters. AFA considers only single objective execution time with the help of firefly algorithm with minimum number of tasks leaving number of considerations. HCA focuses on degree of imbalance and completion time with lower makespan time in limited environment but leaves migration time, execution time and other important parameters. The suggested CFLB algorithm, however, not only addresses their shortcomings, but also strengthens them and adopts a multi-objective strategy.

The key aim of this analysis is to introduce a new hybrid metaheuristic algorithm in a cloud that efficiently performs classification and load balancing. In comparison, this paper's contributions include:

- File format classification is proposed using SVM classification in the cloud environment to group the datafiles in respective classes.
- New multi-objective hybrid algorithm CFLB achieves highest classification accuracy in the cloud and further integrates with improved CSO for load balancing.
- CFLB has provided strong convergence and better results in the presence of a certain state of the art baselines and in terms of developed multi-objective QoS metrics.

The rest of this research is organized as: a literature review is presented in Section 2, the proposed methodology is discussed in Section 3, the experimental setup is described in Section 4, results and discussion are discussed in Section 5 and conclusions are presented in Section 6.

2 Literature Review

The algorithms for load balancing are categorized as dynamic, static, or hybrid, and rely on the state of the system. Based on the characteristics used in load balancing, they are often recognized as allocation and scheduling algorithms. In addition, based on their mix, they are listed as VM-load balancing, CPU-based load balancing, task-based load balancing, server-based load balancing, network-based load balancing and normal cloud load balancing. Much of the analysisdid not consider substantial and important QoS metrics. In addition, some important QoS metrics, such as migration length, migration costs, breach of service efficiency, mission failure rate, and balance level, are not addressed. It has been found that algorithm complexity is not granted much attention when maximizing the performance of load balancing. The research also concluded that many problems remain a major problem in load balancing that can be solved in the long term by introducing a suitable, reliable, and robust algorithm for load balancing.

The studies [54–56] are suggesting several taxonomic classifications of load balancing algorithms that invite future researchers to adequately address load unbalancing issues. The load balancing algorithm must enhance responsiveness, cost of deployment, time of implementation, throughput, vulnerability to faults, length of migration, makepan, resource throughput, and

usability. The drop in these dimensions leads to low Quality of Services (QoS) at Cloud Service Centers (CSC) and a diminished CSP economy. Few of these problems are addressed in the following sections that mostly revolve around our established QoS metrics.

2.1 Approaches Used in Load Balancing

Strumberger et al. [57] suggested metaheuristic hybrid algorithm called WOA-AEFS in cloud storage in which they addressed the resource scheduling problem. This report has two methods to scheduling in which the first is a single objective to reduce makespan and second is multi-objective in which both makespan and cost are considered. The algorithm outperformed other metaheuristic algorithms such as original BAT and PSO. However, the algorithm did not consider other factors such as execution time which is an important one for showing the performance efficiency of the proposed algorithm. The Extended BAT Algorithm (EBA) implies that three benchmark functions, such as Ackley, Hyperellipsoid, and Rosenbrock, should be updated [58]. This move aims to achieve greater efficiency by looking for optimal options, fitness features and convergence speeds. The two versions namely modified and hybridized showed better performance in terms of faster convergence. This helps in enhancing exploitation in the final iterations of the algorithm, improving exploration in the initial execution cycles when compared with other metaheuristics, such as MBO, MMBO, and MBO-FS in the same cloud. An analysis suggested the Gravitational Search Algorithm (GSA) for cloud load balancing where high convergence over a number of iterations was demonstrated by outperforming other algorithms such as MinMin, SA, Tabu Search, Min–Min and GA [59]. The downside of the algorithm is computing intensive that conditionally works well under certain iterations that need to be extended for evaluating the scalability of the algorithm. One of the research suggested hybrid IRRO-CSO which is inspired by the perception of information flow in raven social activity among food search participants while CSO is focused on chicken behavior during a food hunt [60]. The algorithm was tested against the benchmark features of CEC 2017, resulting in enhanced performance over BAT, PSO, individual RRO, and CSO. The output, however, needs to be tested on large real-time datasets while optimizing the execution time. In [61], Modified Heterogeneous Earliest Finish Time (MHEFT) is recommended for dynamic load balancing in cloud so that workload is uniformly distributed among processors according to task rankings that result in reduced makespan. The algorithm works well under a smaller number of tasks and did not consider other QoS metrics for performance evaluations. CEGA is a balancing algorithm that is genetically inspired and built to fulfil deadline restrictions by decreasing the execution time of tasks [62]. CEGA findings have shown improved efficiency on the same workflows as LIGO, Epigenomics over IC-PCP, PSO, and RTC. However, this algorithm suffers exponential time complexity. An auction-based method OVMAP for virtual machine allocations, pricing models, and provisioning using the number of resources in the cloud computing environment is proposed in the work [63]. Extensive experiments have done to demonstrate the performance of OVMAP and it is shown that an online mechanism generates good results with faster processing. However, this method focuses only on fulfilling online requests. Rekha et al. proposed an ETA-GA model for job scheduling in cloud computing to reduce task completion time, increased throughput, and reduced makespan [64]. However, this algorithm did not consider real-time complex tasks and hybrid approaches used for optimization. Another study proposed an online error prediction model for online algorithms [65]. This model chooses a specific algorithm based on experience, performance, and cost. The algorithm performs well against the weighted majority algorithm when 55 VMs are averaged together under the same configuration and parameters. However, the algorithm only works well under predefined and limited conditions. Load balancing mechanism for cloud computing utilizes the concept

of hierarchical classification to minimize the vulnerabilities caused by multi-variant and multi-constraints [66]. The authors claimed a strong classification system is lacking in recent survey articles. The drawback of current surveys can be illustrated as there are no specific and notable taxonomic hierarchical structure algorithms. A variant of CSO called Enhanced Cat Swarm Optimization (ICSO) is proposed by [67]. With adjusting location and velocity equations, the first change is enhancing the tracing mode. Similarly, other option is to make changes in such a manner that the problem of local optima is avoided. The algorithm, however, is only evaluated on 12 benchmark functions with fewer operations, although QoS metrics are not included, such as execution time, migration time, and energy consumption.A hybrid metaheuristic algorithm, HBMMO, was developed by Nazia et al. [68] for workflow preparation in cloud computing to boost cloud performance. The study utilizes a multi-objective approach that is measured over workflows, but energy consumption is not taken into account.

For predicting load sequencing in a datacenter using a cloud computing platform, Weighted Wavelet SVM (WW-SVM) is suggested in [69]. Experiments shown that the proposed algorithm outperformed WSVM, TSVR, ANFIS, and ARIMA in terms of execution time, throughput, and error prediction. The algorithm utilizes only estimation and consistency, although there is no talk of simple multi-goals. For load balancing in the cloud with the target of minimal resource waste, an enhanced SLA violation is addressed in [70]. The algorithm used optimal tools in which up to 17% compared to RR, there is less failure rate of completing tasks and retaining low energy consumption, less migrations, and less SLA breaches. Execution time and various empirical analyses were not considered by the algorithm. SLA agile-based VMs is suggested by Sharma et al. to minimize reaction time [71]. This study used ghost VM to cut the amount of VM output by roughly 12 percent, resulting in two algorithms improving performance. SLA violations are minimized by taking benefit of admission monitoring and rescheduling. Static workloads have been used and this literature does not address efficiency metrics. One of the studies considered five metaheuristic algorithms wherein QoS restrictions and penalty costs are taken into account, detailed comparisons of SLA breaches in a cloud context are carried out [72]. The algorithm is evaluated with statistical evaluations on CEC benchmarks, though QoS metrics (except SLA) are not addressed, except scalability.

The MMA dynamic VM migration algorithm is intended for high computing (HPC) [73]. The MMA algorithm reduces the load and decreases connectivity costs on overwhelmed computers Performance deterioration can be caused by saturation and computation sophistication is a concern. Another research addressed a technique for VM migration that offers increased scalability over all other strategies [74]. A hybrid scoring method that measures the static and dynamic score of every other host is used in this method. There is high computation cost and overhead. Further, the only number of migrations is considered leaving other QoS metrics. The study provides an increase in decreased overhead using a metaheuristic automated electricity algorithm in VANETs [75]. This study accomplishes energy-efficient transmission by up to 50% with the help of routing protocol. Overall, PSO outperformed other competitors in the conducted experiments and obtained energy efficiency as well as reduced network overhead. For the latest search of a neighbor using an optimal decision by ACO, the Biological Motivated Self Ordered Autonomous Routing Protocol (BIOSARP) is proposed [76]. This algorithm has a relatively higher overhead cost. Harmony-inspired genetic algorithm (HIGA) for reducing overhead in a cloud environment combining GA (for exploration) and HS (for exploitation) metaheuristic algorithms is discussed by [77]. While using intelligent capability, the global optimum is achieved in a reduced number of iterations resulting in reduced overhead and less energy consumption. Simulations conducted

over CyberShake and Montage results in better performance of HIGA such as 39% low overhead, energy savings up to 33%, and overall, 47% performance. From the review of papers discussed here, it is noted that there is no systematic multi-objective approach that optimizes the QoS metrics without compromising the best solutions.

2.2 Overview of Classification-Based Approaches

Over the years, classification is playing a significant role in minimizing the load on the system in a number of applications such as load balancing, image recognition, textual identification, multimedia data classification and many more. Classification is a supervised learning mechanism in which the application learns to train the data and assigns to appropriate categories [78]. In the subsequent text, a few of the classifiers are being presented in Tab. 1 to get an insight into the frequently used classifiers.

Table 1: Survey of various classification-based approaches

Reference	Classification approach	Algorithm	Advantages	Limitations
[79]	SVM	Survey study	High accuracy using metaheuristics and classifier	Limited to few parameters
[80]	SVM	MCSO + SVM	Better optimization solution	Time complexity
[81]	SVM	NMCSO	Fast convergence, accuracy	Less scalable
[82]	SVM	FFA + SVM	Fast convergence	Less accuracy
[83]	SVM	PSO + SVM	Better accuracy	Limited datasets
[84]	Decision Tree	Survey study	Cyber-attacks are classified and discussed	Limited to only cyber attacks
[85]	K-NN	K-NN + NB	Better accuracy	Datasets are limited
[86]	K-NN	MKNN	Highest accuracy	Single objective considered
[87]	NB	NB + SVM	Accurate anomaly detection	More execution time
[88]	NB	NB + SVM	Improved preprocessing	Less reliability
[89]	NB	Modified NB	Better personality classification	Few features
[90]	RF	RF + SVM	Limited spectral bands	Datasets are few
[91]	RF	RF	Highest accuracy	Conditional constraints
[92]	RF	RF + SVM	Better accuracy	Limited datasets
[93]	RF	RMDL	Better robustness	Scalability issue

To summarize, most of the aforementioned classifiers considered either one or at most two features in different forms but have not considered multiple features simultaneously. The proposed approach addresses the limitations of previous work by taking into account multiple objectives to reach optimal solutions.

In the cloud, there exists a huge volume and variety of data in multiple formats that require extra resources to get processed. The studies mentioned in the literature did not consider the data

formatting approaches to solve such problems with respect to load balancing and classification such as audio, video, image and text in a combined fashion. Similarly, those studies focused mostly on single parameter such as either audio, video, image or text with one or more QoS parameters. However, their combination with metaheuristics in cloud still has a lot of research to perform. As a result, much of the time is consumed in obtaining the actual data at run time. This time and other resources are greatly affected by the offline preprocessing approach using classification of data introduced in the study. Similarly, combination of optimization approaches with machine learning models in the cloud is relatively a new approach especially with respect to classification of data formats on number of QoS metrics. When doing offline preprocessing, the classification procedure will quickly minimize those complications and make accessible the data in the processed form. There are several constraints simultaneously encountered by the cloud service providers and clients about rapid accessibility to the cloud services. Because of the large volume and variety of data stored in the cloud, extracting actual information is a complex task that necessitates more resources. When processing large-scaled, computationally intensive, and resource-intensive applications, the problem becomes even more challenging. Data preprocessing can be useful in these situations, as an offline categorization of data using machine learning models may greatly reduce execution time and memory needs during the online processing phase. Furthermore, job assignment to VMs must be done with care to achieve effective load balancing. As a consequence, we suggested a new hybrid model based on SVM and CSO that provides optimal load balancing performance when compared to current models in order to produce finer classification results and efficient work assignment. By combining these two models into a hybrid with a multi-objective approach, their individual flaws are addressed while their combined merits are reinforced.

3 Research Methodology

The methodology involved in this research has been implemented based on the integration of SVM classifier and load balancing algorithms using the data format classification approach. Fig. 1 shows an architectural working flow of the set of activities in which the input data provided to the cloud environment is classified into different segments/formats such as audio, video/HDs, text formats, images, and digital maps etc. The classified data is then fed into an optimization algorithm where evaluations are performed according to the QoS matrix. Data classification is performed using SVM as one to many classifier types. The process is initialized with the size of input data files and proceeds with the classification process followed by the QoS evaluations. The input data is collected from various sources and fed into the system. SVM classifier accurately classifies various datafiles into appropriate data class. This is the stage where preprocessing is performed, and classified data is ready to be given to a scheduling algorithm. The next stage of data processing is the scheduling of data. The data collected are then placed to the proposed scheduling algorithmCFLB. Working of the proposed architecture is divided into following four steps:

Step 1: In the first step, the user enters a number of tasks in the cloud environment that needs to be scheduled up to 100,000. These tasks are various types of datasets, such as audio, video, text, and image. Selection of the tasks is made at random from the developed datasets.

Step 2: In the second step, these tasks are classified to respective classes such as class A for audio, class B for video, class C for image and class D for text datasets. Here training and testing of the data is passed through SVM one to many classifiers such as 70% for training and 30% for testing data. The SVM uses Polynomial Kernel function to make classifications at higher dimensional space which further helps to accurately classify the data.

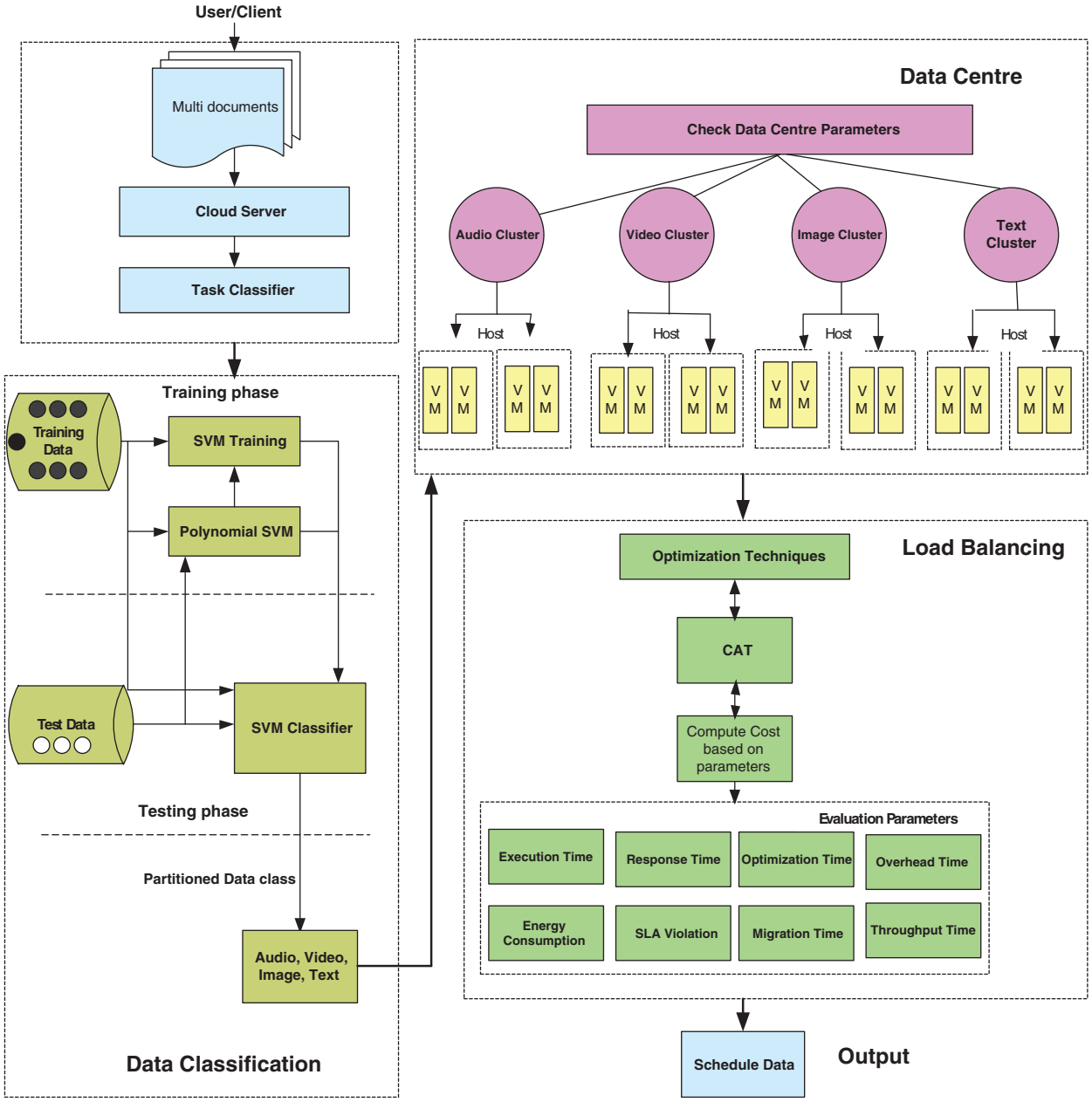

Figure 1: Architecture of CFLB

Step 3: In this step, the data classification of the appropriate class is fed into datacenter which is configured to various parameters such as type of number of datacenters, number of hosts, operating system, ram size, storage size, data size, bandwidth etc. Here, four sets of VMs are used in clustering forms such as a set of VMs each for audio, video, image, and text data with respect to the following:

- A task dependent on task specifications is assigned to each computer (VM). Video tasks, for example, demand 1000 floating-point and 16 GB memory operations, audio data requires

800 floating-point and 12 GB memory operations, image tasks involve 800 floating-point and 8 GB memory operations, and textual tasks require 400 floating-point and 4 GB memory operations. The SVM classifier then defines the collection of VM types, such as VideoVM, AudioVM, TextVM, ImageVM, depending on the specifications, format, and functionality. The corresponding VM relating to each assignment is allocated here. Therefore, SVM aims to define data and fit it to the most acceptable type of VM.

- Extracted features of sequences of 40 frames are derived from four separate video classes for video data classification, where a 40-4096 matrix is used, where each row corresponds to attributes of one frame (one frame per row) that assist in classifying videos within these four distinct classes. A new video is preprocessed to restrict the number of frames and then to identify it by removing features from this video. Assume there will be four (ci, i = 1,…, 4) video groups. Each video has 40 (n = 1,…, 40) frames and we extracted 4096 features from each frame ([1 × 4096]). Since each frame has enough data to estimate the video class (ci), we used 40 frames as training/test samples from each video, which generates an input matrix of [160 × 4096] dimensions, with 160 samples and 4096 features for each sample. In addition, an output vector [160 × 1] containing the mark of each ci = I class is generated, where i = 1. Four audio feature sets are assessed to classify five types of audio groups for audio data classification: classical music, popular music, crowd noise, voice, and basic noise. Centered on perceptual models of sound, the feature sets include low-level signal properties, mel-frequency spectral coefficients, and two new sets. 256 × 256 pixels (65,536 pixels in total) are considered for image classification, in which each pixel is used as an attribute in the SVM classifier.
- There are text records of around 6 GB for text classification that are extracted in the form of unstructured text. Stemming and stop word elimination is done and the words in the form of functions are then removed.

Step 4: In the last step, the output of the above phase is fed into the load balancing algorithm for optimization. In this architecture, CSO is proposed which is used to optimize the tasks and schedule the data properly. CSO is used in this research and comparison with baselines is performed in order to determine its superiority. In order to achieve accurate load balancing by contemplating the conditions, the efficiency analysis of the proposed model is then carried out on various parameters such as energy consumption, optimization time, and migration time.

The scheduling process will be started until the number of tasks and VMs are chosen. N instances are initially generated and separated into G classes. The action of the cats is taken into account by CSO in two modes, searching mode and tracing mode. Swarm algorithms are generally accepted as they have developed the best solutions obtained to look for the closest neighbors (nodes). Thus, the cat activity is considered in this technique to look for a solution space. Each cat has d-dimensions with different velocities used for each dimension of its place. If the fitness is not equal, then measure the likelihood using Eq. (1) and, by default, the probability value is set to 1. Each cat is evaluated using the fitness function. In order to determine whether the cat is in search mode or tracing mode, we used the Boolean flag variable. In terms of its fitness function, the tracing mode is called where the position of the cat is modified according to the fitness function. The fitness function of CSO can be obtained as:

$$P_i = \frac{FS_i - FS_{max}}{FS_{max} - FS_{min}}, \quad where\ 0 < i < j, \tag{1}$$

where P_i shows the probability value associated with the position of i^{th} Cat. FS_i is the fitness of i^{th} cat, FS_{max} represents the maximum fitness value and FS_{min} represents the minimum fitness value achieved.

Algorithm I: SVM

Input: Video, Text, Audio VM, N = 100
Output: Data class
1: **for each** data classification **do**
2: SVM.data Evaluate (F_k)
3: **for each***P* (*u*, *v*) **do**
4: Evaluate ← Kernel SVM
5: **for each** Classification accuracy ≠ 100 **do**
6: Compute the data accuracy
7: **if** Number of iterations ≠ N **then**
8: Classify ← data categorization
9: **return** Data class (CSO)

Line1: Input (dataset), Output: Data class (Classified data)

Line 2: In order to classify the data, first the data will be evaluated (to find out the class and is the input is similar to the same class)

Line 3: Condition check where u = class type and v = data (1,…, n)

Line 4: Evaluation of data in kernel SVM

Lines 5 & 6: Condition check for accuracy. If accuracy not equal to 100, evaluate the data accuracy depending on the input.

Lines 7 & 8: Iteration check. If iteration is not equal to 100, the data will be classified as specific data class.

Line 9: Output data which will be the input to CSO.

CFLB is based on load balancing method and increasing the energy efficiency of modelled technique. Load balancing aims at obtaining the best resource utilization among number of entities. This helps in achieving the maximum throughput which is the amount of data transmitted successfully for optimization.

Algorithm II: CFLB

Input: video, text, audio, image, N, number of virtual machines (VM), number of cats, max_iterations, SMP (seeking memory pool), MR is the Mixture ratio, Sz: size of population, maximum_iter
Output: Data class, Scheduled data
1: **for** data classification **do**
2: SVM.data Evaluate (F_k)
3: **for each** *P* (*u*, *v*) **do**
4: Evaluate ← Kernel SVM
5: **for each** Classification accuracy ≠ 100 **do**

(Continued)

```
6:          Evaluate. Data accuracy
7:          if Number of iterations ≠10 then
8:           Classify ← data categorization
9: return Data class (RCSO)
10: for load balancing do
11:     Cat.Population Initialization (X_i)
12:     for each X_i population I < Imax do
13: Calculate ← Fitness
14:       for each X_g ← Cat with best solution (i = 1) do
15:         if mode has solution and SPC = 1 then
16:          Start ← seeking mode
17:          if mode has solution and SPC ≠ 1 then
18:            Start ← tracing mode (Iteration Process)
19. return Scheduled Data
20. Exit.
```

The mechanism of the proposed approach called CFLB is represented by Algorithm II. Lines 1 to 9 detailed data categorization in this algorithm, which first classified the category of data and then classified the type of VMs using SVM and allocated it to the specific class. Lines 10 to 20 used CSO to do load balancing and then output the data for the scheduling. In terms of the movement of cats, which is dependent on the excellent hunting ability of cats, the tracing mode of the CSO system is defined. In tracing mode, the movement of cats for each dimension is as per their velocity and then their positions are modified accordingly. Modified cats' and velocity positions are determined using Eqs. (2) and (3). These equations can be

$$V_{i,d} = V_{i,d} + r_i c_i \left(X_{best,d} - X_{i,d}\right), \quad d = 1, 2, \ldots M, \tag{2}$$

$$X_{i,d} = X_{i,d} + V_{i,d}, \tag{3}$$

For the position of the Cats, different terminologies are used here such as $X_{best,d}$ is the best position of a cat in d-dimensional space, $X_{i,d}$ position of Cat_i, $V_{i,d}$ is the velocity, C_i, is a constant value, r_i is a random value between [0, 1]. In the algorithm for searching mode and tracing mode, the Mixture Ratio (MR) is used to merge the two modes and to determine the Cats' ratio in the modes. The seeking mode of the CSO technique composed of the parameters such as: Seeking Memory Pool (SMP), Seeking Range of selected Dimension (SRD), Counts of Dimension Change (CDC), and Self-Position Consideration (SPC).Centered on computation time, we have measured the CFLB model assessment. We also used various parameters that are stated in optimization computation as well as classification algorithms, as our model follows a hybrid methodology that is the synthesis of SVM and CSO. These parameters are shown in Tab. 2 for CSO. We have used parameter values for CloudSim4.0 as shown in Tab. 3. These test sets are selected based on the convergence of the CFLB algorithm after many experiments have been conducted. From experiments, it is observed that the seeking mode takes less time as compared to tracing mode. Thus, the overall time complexity of the proposed CFLB is $O(N^3 + N^3(n + ks + l))$ which gives to $O\left(N^3 + N^3.n + N^3.ks + N^3.l\right)$ and ultimately the time complexity is solved to $O(N^3)$.

Table 2: Parameters used for CFLBalgorithm

Parameter	Value	Parameter	Value
Size of the cat	100	Coefficient (c)	1.5–2.5
Size of the population	100–1000	Random number (r)	0 to 1
Counts of dimension to change (CDC)	70%	Maximumiterations	1000
Seeking range of selected dimension (SRD)	30%	Mixture ratio (MR)	(1–3)%
Self-position consideration (SPC)	1 or 0	Seeking memory pool (SMP)	4–10

Table 3: Parameter used for CFLB in the CloudSim4.0

Datacenter	Parameters	Values	Parameters	Values
Cloudlets or Tasks	Datacenters size	2–4	RAM size	16–32 GB
	Hosts in the datacenters	4–16	Storage capacity	0.5–1 TB
	RAM size for Host (each)	2–4 GB	Data BW	1–10 GB/s
	Total tasks in number	40000	PE Power	10^6 MIPS
	Size of each task	0.5 MB–1 GB	Tasks size	22.2 GB
VM	VM ID	[1–40]	VMM	Xen

4 Experimental Setup

In this portion, different files, namely audio, video, text, and images taken from the UCI repository [94] and other databases, are provided in the form of datasets. As given in Tab. 4, there are a total of 40,000 datasets that are further divided evenly. In addition, different dataset files with a ratio of 60:40 are put in training and testing mode, where 60% of data files are training datasets and 40 percent are testing datasets as defined in Tab. 4. In contrast to other simulation models, CloudSim4.0 [95] is commonly used in the conduct and deployment of cloud-related research. In virtualization mode, the simulator offers on-demand services and has a variety of benefits, such as accessibility, efficiency, and ease of use. The area, configuration, operating system, VM, memory, data transfer cost of storage, and the amount of physical hardware units are configured in a datacenter. In this scenario, along with 4096, 8192 MB of RAM and 2 TB of memory, we have set 500 and 1000 VMs, each, during experiments. Both simulations are conducted on a Desktop PC consisting of an MS Windows 10 operating system, a 2.6 GHz Intel Quad-Core i7 machine, 12 GB RAM and 1 TB HDD.

Table 4: Training and Test sets details

Samples	Size in quantity	Samples	Size in quantity
Audio datafiles (Training)	6000	Audio datafiles (Testing)	4000
Video datafiles (Training)	6000	Video datafiles (Testing)	4000
Images datafiles (Training)	6000	Images datafiles (Testing)	4000
Text datafiles (Training)	6000	Text datafiles (Testing)	4000
Total datafiles	**40000**		

To make the findings accurate, all algorithms are applied in CloudSim4.0 obtained from their corresponding research papers with the same setup and environment setting. In addition, the results are objectively checked to verify their significance by student t-test review, which avoids the fact that the values also are not by chance.

5 Results and Discussion

The proposed algorithm has been split into two main sections in this section. In the first section, the dataset classification of the file is performed using the cloud SVM classifier. In the second component, SVM output is fed into CSO for cloud environment load balancing. We have used One-vs-All classification method to achieve easier, fast, and reliable performance, which initially classifies file datasets by contrasting them with all classes [96]. Consequently, One-vs-AllSVMs, has exceeded other classifiers in terms of accuracy in the first level.

5.1 Accuracy of CFLB

Cross validation such as accuracy, precision, recall, and F-measure are used to check the accuracy of the CFLB. The classifiers such as ACO-SVM [97], SVM-GA [98], GASOM [99], and KNSC [100] are used to perform comparative analysis given in Tab. 5. The results of CFLB are presented as comparative analysis over other algorithms in which CFLB has shown better performance in all validation methods. The results of classification algorithms are validated using classification validation accuracy measures with respect to Accuracy, Precision, Recall and F-Measure [101].

Table 5: Comparative analysis of CFLB with classification techniques

Metric	CFLB	SVM-GA	GASOM	KNSC	ACOSVM
Accuracy	0.984	0.971	0.881	0.925	0.971
Precision	0.983	0.969	0.899	0.921	0.981
Recall	0.981	0.977	0.921	0.964	0.979
F-Measure	0.997	0.979	0.976	0.984	0.981
G-Mean	0.970	0.952	0.958	0.951	0.969
AUC	0.981	0.954	0.961	0.959	0.979

5.2 Evaluation of CFLB on Energy Consumption

Following Eq. (4) is used for energy consumption calculation:

$$E_C = \sum_{k=1}^{N} \sum_{N=1}^{n} \left(\frac{T_E(\mathrm{VM}_N)}{E(T_k)} \right), \tag{4}$$

Fig. 2 shows energy consumption of CFLB against all baselines. A good difference in energy consumption can be seen from the very start which continues till 10000 VMs. It is important to mention that all parameters of the simulation are set with same configurations resulting in reliable results in all cases.

With the increase in number of VMs, an increase in energy consumption is seen with all baselines. Few of the baselines are consuming more energy showing unstable and non-scaling

behavior as compared to other baselines. In computational terms with respect to percentage, Min–Min is consuming a total of 26.16% energy followed by CGSA with 20.1%, AFA with 17.31%, PLBA with 11.96%, HCA with 10.48% and CFLB with 5.63% only. Cloud classification preprocessing decreased computational complexity resulting in lower energy consumption and further faster convergence of CSO achieved the best solution in the minimal number of iterations that also retained the energy.

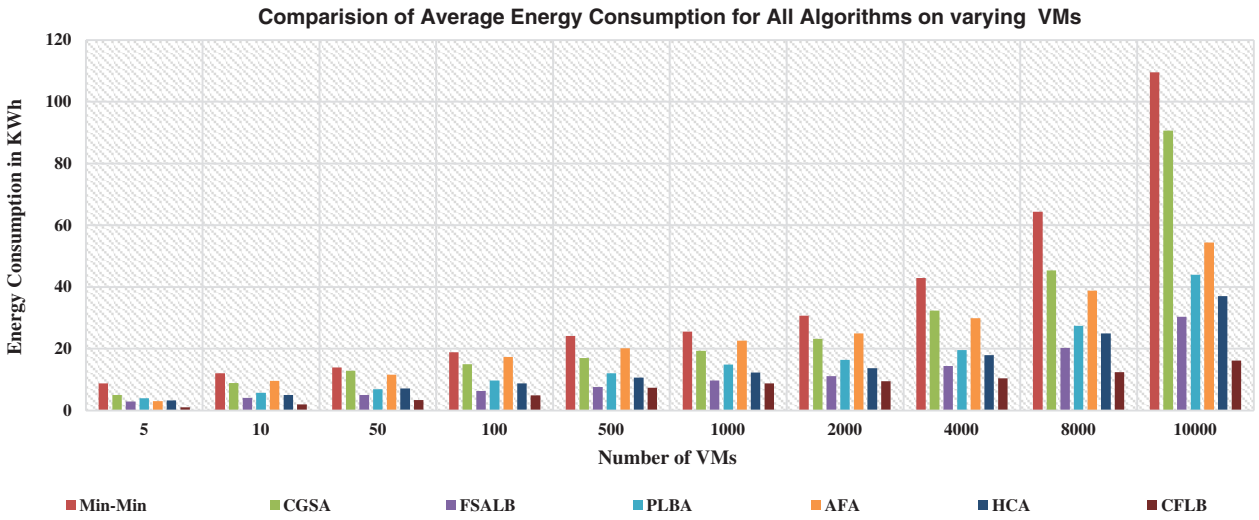

Figure 2: Performance of CFLB and baselines on energy consumption on VMs (5–10000)

5.3 Evaluation of CFLB on Migration Time

Following function is used to calculate the migration time in implementation.

$$M_t = \sum_{k=1}^{n} M_{S_k}(VM_i, VM_{i+1}) \tag{5}$$

Here, $M_{S_k}(VM_i, VM_{i+1}) = M_{S_k}(VM_i \rightarrow VM_{i+1})$, is the scheduling used to allocate k^{th} data from i^{th} VM to $(i+1)^{th}$ VM depending upon the VM availability.

Fig. 3 displays the migration time that all baselines for various tasks and VMs have taken. As compared to less migrations, more migrations need more time. The lowest number of migrations are made by optimization algorithms since a single migration requires several processes, their correspondence, interrupts, and other variables. The VM migration policy is controlled by the administrator, where a law determines when the VM migration from one host to another host should be activated.Generally, a threshold is established that allows VM migrations to minimize the amount of SLA breaches and migrations when considering the host machines' computing capabilities. As a hybrid approach, both heuristic and metaheuristic algorithms perform well in seeking the solution to VM migration. Metaheuristic algorithm works very well for optimization during migration. Fig. 3 depicts that in the start, least VM migrations in case of CFLBare seen which remain there till the end of simulation with every run. Similarly, in terms of percentages,

Min-min algorithm has taken 19.1% migration time as compared to PLBA which takes 18.3% migrations followed by AFA with 16.6%, CSGA with 14.9%, HCA with 13.6%, FSALB with 11.4% and CFLB with 6% only.

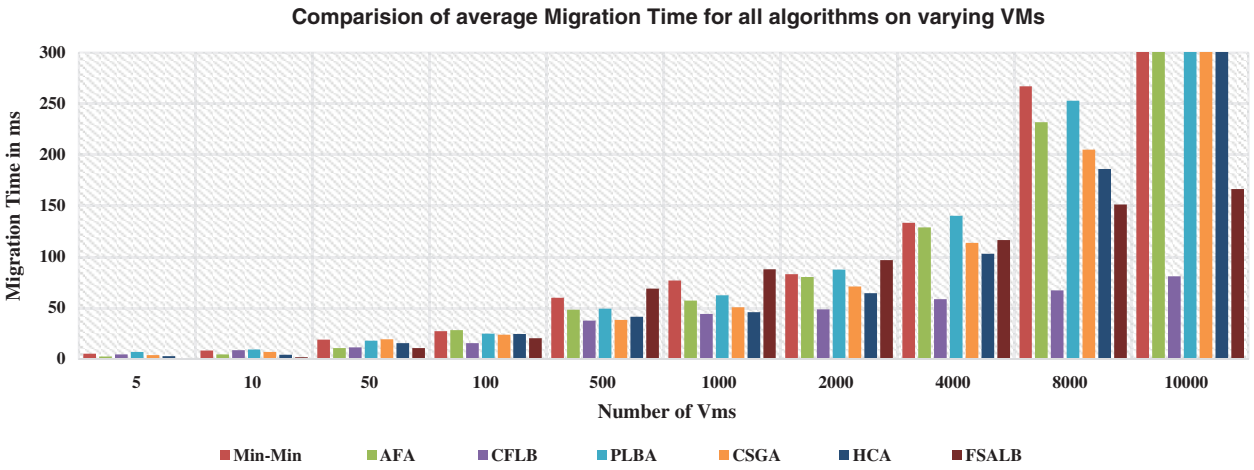

Figure 3: Performance of CFLB and baselines on migration time on VMs (5–2000)

5.4 Evaluation of CFLB on Optimization Time

Following Eq. (6) is used to calculate the optimization time:

$$O_T = \sum_{i=0}^{Imax} T_j. \tag{6}$$

where, Tj: Tj is the time taken for j^{th} iteration, $\sum_{j=0}^{Imax} T_j$: Complete iteration time

Fig. 4 shows that two algorithms Min–Min and CGSA have taken time exponentially than other algorithms. This shows the unstable behavior of both algorithms. In percentage terms, Min–Min algorithm has taken 25.4% optimization time more than all other algorithms. Similarly, CGSA has taken 22.4% optimization time followed by PLBA with 12.8%, HCA with 11.5%, AFA having 11% and finally CFLB with 7% optimization time only showing fastest convergence behavior.

5.5 Statistical Analysis

We also tested all metrics' resulting values and found that their distribution is normal. Within this case, a parametric test involving 2 variables is needed since we have taken one baseline at a time and compared it to CFLB. The optimal measure for 2 variables of normal distribution in statistics is the ANOVA test given in Tab. 6. Similarly, we can see values like mean, standard deviation (SD), p-value, and t-value in Tab. 6. The level of significance, meanwhile, is set to $p < 0.05$ [102]. At this stage, we need to define the hypothesis in the following manner:

H0:CFLB and other baselines have no difference.

H1: A significant difference exists between CFLB and other baselines.

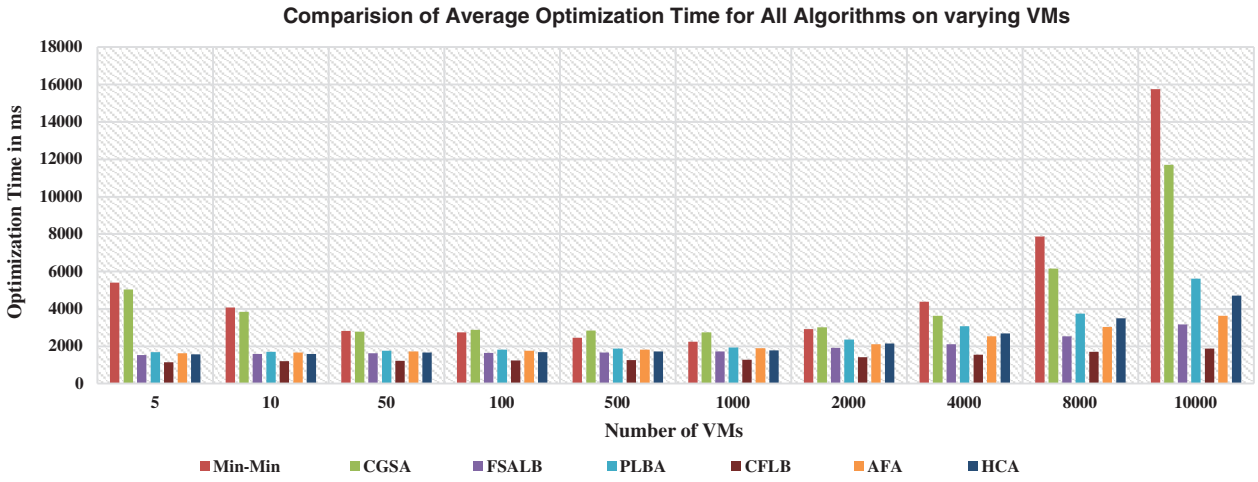

Figure 4: Performance of CFLB and baselines on optimization time on VMs (5–10000)

Table 6: Statistical comparison of CFLB with baselines

Source of variation	SS	Df	MS	F	P-value	F crit
ANOVA test (energy consumption)						
Between groups	5653.84226	6	942.307	3.033953	0.011384	2.246408
Within groups	19566.997	63	310.5873			
Total	25220.8392	69				
ANOVA test (migration time)						
Between groups	21710.07	6	3618.344	2.202807	0.037471	0.246408
Within groups	1124004	63	17841.34			
Total	1145714	69				
ANOVA test (optimization time)						
Between groups	1.13E + 08	6	18830338	4.658749	0.000559	2.246408
Within groups	2.55E + 08	63	4041930			
Total	3.68E + 08	69				

In all cases, we could see that p-values are lower than the significance level <0.05, which indicates that there is a significant difference between the values of CFLB and other baselines. Therefore, we are right in denying the null hypothesis and supporting the alternative hypothesis. Similarly, in terms of energy consumption, migration time, and optimization time, we can say that a substantial difference occurs in the resulting values.

6 Conclusion

There is huge amount of data in cloud, located at distributed sites geographically where computation is performed over the network and tasks are transacted by read and write operations, then dynamic energy consumption and load balancing are not unanticipated but a common aspect.

To make the cloud load balanced and energy efficient is challenging task. This research proposed a novel energy efficient load balancing strategy for cloud computing that provides an efficient utilization of resources in cloud environment. The developed model CFLB improves the accuracy using hybrid approach that first classifies the data into specific data class using SVM (one to many classification) and then scheduled the data class using RCSO. Initially, accuracy of the CFLB is determined using SVM and compared with other classification techniques. The results have shown that our designed approach is much better and achieved more than 98 percent accuracy on average. Similarly, the experimental setup is developed using CloudSim4.0 simulator. The results are generated gradually with varying tasks number, tasks size, tasks format and VMs size. The developed CFLB algorithm outperformed baselines like HCA, Min–Min, CSGA, FSALB, AFA, and PLBA in QoS metrics such as energy consumption, migration time, and optimization time. This shows that results have achieved scalability, performance, and intelligence in our proposed model.

Our future work will take big data, OpenStack and deep learning techniques into consideration as classification problems are being well addressed by them. Other important parameters such as deadline constraints, task immigrations, resource constraints and self-adoption need to be addressed. Furthermore, the concept of auto-scaling for predicting the incoming workload, multi cloud auto-scaling, energy aware auto-scaling, and bin packing auto-scaling requires deliberation for future research. Other issues of concern are security, cost management, pricing model and exploring monitoring tools in the cloud form strong areas of future research in the cloud domain.

Acknowledgement: Thanks to our families & colleagues who supported us morally.

Funding Statement: This work was funded by the University of Jeddah, Saudi Arabia. The authors, therefore, acknowledge with thanks to the University technical support. The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the Project Number MoE-IF-20-01.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Nayyar, "Interoperability of cloud computing with web services," *International Journal of Electro Computational World & Knowledge Interface*, vol. 1, no. 1, pp. 36–43, 2011.

[2] S. Lange, J. Pohl and T. Santarius, "Digitalization and energy consumption. Does ICT reduce energy demand?" *Ecological Economics*, vol. 176, no. 106760, pp. 1–14, 2020.

[3] R. Buyya, CS. Y., S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[4] L. Heilig, E. L. Ruiz and S. Voß, "A cloud brokerage approach for solving the resource management problem in multi-cloud environments," *Computers & Industrial Engineering*, vol. 95, no. 4, pp. 16–26, 2016.

[5] A. Nayyar, "Handbook of cloud computing: Basic to advance research on the concepts and design of cloud computing," in *Introduction to Cloud Computing*, 1st ed., vol. 1. New Delhi, India: BPB Publications, pp. 1–52, 2019.

[6] A. Nayyar, "Private virtual infrastructure (PVI) model for cloud computing," *International Journal of Software Engineering Research and Practices*, vol. 1, no. 1, pp. 10–14, 2011.

[7] P. Singh, P. Gupta, K. Jyoti and A. Nayyar, "Research on auto-scaling of web applications in cloud: Survey trends and future directions," *Scalable Computing Practice and Experience*, vol. 20, no. 2, pp. 399–432, 2019.
[8] L. Heilig, R. Buyya and S. Voß, "Location-aware brokering for consumers in multi-cloud computing environments," *Journal of Network and Computer Applications*, vol. 95, no. 6, pp. 79–93, 2017.
[9] A. Kaur, B. Kaur and D. Singh, "Comparative analysis of metaheuristics based load balancing optimization in cloud environment," in *Smart and Innovative Trends in Next Generation Computing Technologies-Communications in Computer and Information Science*, vol. 827. Singapore: Springer, pp. 30–46, 2018.
[10] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: A survey," *ACM Computing Surveys*, vol. 51, no. 6, pp. 35, 2019.
[11] G. Wendong, Q. Chengzhi, J. Liu and J. Zhang, "A novel hybrid meta-heuristic algorithm for optimization problems," *Systems Science & Control Engineering*, vol. 6, no. 3, pp. 64–73, 2018.
[12] P. Kaur and P. D. Kaur, "Efficient and enhanced load balancing algorithms in cloud computing," *International Journal of Grid and Distributed Computing*, vol. 8, no. 2, pp. 9–14, 2015.
[13] M. Yadav and G. Sachin, "Hybrid meta-heuristic VM load balancing optimization approach," *Journal of Information and Optimization Sciences*, vol. 41, no. 2, pp. 577–586, 2020.
[14] C. M. kumar, S. Karthik and V. P. Arunachalam, "Hybrid metaheuristic algorithm for improving the efficiency of data clustering," *Cluster Computing*, vol. 22, no. S1, pp. 435–442, 2019.
[15] H. Wang, K. Li and W. Pedrycz, "An elite hybrid metaheuristic optimization algorithm for maximizing wireless sensor networks lifetime with a sink node," *IEEE Sensors Journal*, vol. 20, no. 10, pp. 5634–5649, 2020.
[16] S. S. kumar and S. S. Manivannan, "A hybrid meta-heuristic approach for optimization of routing and spectrum assignment in elastic optical network (EON)," *Enterprise Information Systems*, vol. 3, pp. 1–24, 2020.
[17] N. A. Alsabour, "Hybrid metaheuristics for classification problems," in *Pattern Recognition-Analysis and Applications*, 1st ed., vol. 1. London, UK: InTech, 2016.
[18] M. Li, J. Zhang, J. Wan, Y. Ren, L. Zhou *et al.,* "Distributed machine learning load balancing strategy in cloud computing services," *Wireless Networks*, vol. 26, no. 8, pp. 5517–5533, 2020.
[19] C. Gomez, S. Abdallah and W. Xianbin, "Machine learning aided scheme for load balancing in dense IoT networks," *Sensor*, vol. 18, pp. 3779, 2018.
[20] X. Sui, D. Liu, L. Li, H. Wang and H. Yang, "Virtual machine scheduling strategy based on machine learning algorithms for load balancing," *EURASIP Journal on Wireless Communication and Networking*, vol. 160, no. 1, pp. 683, 2019.
[21] Y. Gao, W. Chen, H. Zhang, L. Liu and J. Liu, "SVM-based connection classification and load balancing mechanism," in *Proc. IEEE 2nd Int. Conf. on Knowledge Innovation and Invention*, Seoul, Korea (South), pp. 202–205, 2019.
[22] M. K. Rao, K. V. Swamy and K. A. Sheela, "Advanced machine learning discriminant analysis models for face retrieval system," in *Proc. 2nd Int. Conf. on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, Palladam, India, pp. 609–613, 2018.
[23] J. Chen, Z. Dai, J. Duan, H. Matzinger and I. Popescu, "Naive bayes with correlation factor for text classification problem," in *Proc. 18th IEEE Int. Conf. on Machine Learning And Applications*, Boca Raton, FL, USA, pp. 1051–1056, 2019.
[24] A. Giri, M. V. V. Bhagavath, B. Pruthvi and N. Dubey, "A placement prediction system using k-nearest neighbors classifier," in *Proc. Second Int. Conf. on Cognitive Computing and Information Processing*, Mysore, India, pp. 1–4, 2016.
[25] M. Ozyildirim and A. Mutlu, "Generalized classifier neural network," *Journal of the International Neural Network Society*, vol. 39C, pp. 18–26, 2012.
[26] A. Khamparia, D. Gupta, N. G. Nguyen, A. Khanna, B. Pandey *et al.,* "Sound classification using convolutional nneural network and tensor deep attacking network," *IEEE Access*, vol. 7, pp. 7717–7727, 2019.

[27] B. Tang, Y. Li, X. Li, L. Xu, Y. Yan *et al.,* "Deep CNN framework for environmental sound classification using weighting filters," in *Proc. IEEE Int. Conf. on Mechatronics and Automation*, Tianjin, China, pp. 2297–2302, 2019.

[28] A. Zakaria, R. Ranti and D. Oky, "Particle swarm optimization and support vector machine for vehicle type classification in video stream," *International Journal of Computer Applications*, vol. 182, no. 18, pp. 9–13, 2018.

[29] Y. F. Huang and S. H. Wang, "Movie genre classification using svm with audio and video features," in *Active Media Technology, AMT, Lecture Notes in Computer Science*, R. Huang, A. A. Ghorbani, G. Pasi, T. Yamaguchi, N. Y. Yen, B. Jin (eds.), vol. 7669. Berlin, Heidelberg: Springer, 2012.

[30] K. M. Salama and A. M. Abdelbar, "Learning neural network structures with ant colony algorithms," *Swarm Intelligence*, vol. 9, no. 4, pp. 229–265, 2015.

[31] L. Jiao and F. Liping, "Text classification based on ant colony optimization," in *Fourth Int. Conf. on Information and Computing*, Wuxi, China, vol. 3, pp. 229–232, 2011.

[32] Q. Wang, R. Peng, J. Wang, Y. Xie and Y. Zhou, "Research on text classification method of LDA-SVM based on pso optimization," in *Chinese Automation Congress*, Hangzhou, China, pp. 1974–1978, 2019.

[33] A. Pietramala, V. L. Policicchio, P. Rullo and I. Sidhu, "A genetic algorithm for text classification rule induction," in *Joint European Conf. on Machine Learning and Knowledge Discovery in Databases,*, Springer, Berlin, Heidelberg, pp. 188–203, 2008.

[34] H. Hasanpour., G. R. Meibodi and K. Navi, "Improving rule-based classification using harmony search," *PeerJ Computer Science*, vol. 5, no. 12, pp. e188, 2019.

[35] F. Yiğit and Ö.K. Baykan, "A new feature selection method for text categorization based on information gain and particle swarm optimization," in *Proc. IEEE 3rd Int. Conf. on Cloud Computing and Intelligence Systems*, Shenzhen, China, pp. 523–529, 2014.

[36] H. Peng, C. Ying, S. Tan, B. Hu and Z. Sun, "An improved feature selection algorithm based on ant colony optimization," *IEEE Access*, vol. 6, pp. 69203–69209, 2018.

[37] C. L.Franco, L. Villavicencio, N. A. Daniel and A. Alma, "Image classification using PSO-SVM and an RGB-D sensor," *Mathematical Problems in Engineering*, vol. 2014, no. 3, pp. 1–17, 2014.

[38] C. Sukawattanavijit, J. Chen and H. Zhang, "GA-SVM algorithm for improving land-cover classification using SAR and optical remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 3, pp. 284–288, 2017.

[39] V. Pallavi and V. Vaithiyanathan, "Combined artificial neural network and genetic algorithm for cloud classification," *International Journal of Engineering and Technology*, vol. 5, pp. 787–794, 2013.

[40] Y. Shao and S. L. Ross, "Comparison of support vector machine neural network, and CART algorithms for the land-cover classification using limited training data points," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 70, no. 22, pp. 78–87, 2013.

[41] E. A. Zanaty, "Support vector machines versus multilayer perception in data classification," *Egyptian Informatics Journal*, vol. 13, no. 3, pp. 177–183, 2012.

[42] H. Z. M. Shafri and F. S. H. Ramle, "A comparison of support vector machine and decision tree classifications using satellite data of Langkawi island," *Information Technology Journal*, vol. 8, no. 1, pp. 64–70, 2009.

[43] AWS Amazon, "Using a PostgreSQL database as a source for AWS-DMS, AWS Amazon," 2016. [Online]. Available: https://docs.aws.amazon.com/dms/latest/userguide/.

[44] M. Junaid, A. Sohail, A. Ahmed, A. Baz, I. A. Khan *et al.,* "A hybrid model for load balancing in cloud using file type formatting," *IEEE Access*, vol. 8, pp. 118135–118155, 2020.

[45] Google Cloud, "Data preprocessing for machine learning: Options and recommendations, Google Cloud," 2017. [Online]. Available: https://cloud.google.com/solu.

[46] SC. C., P. Tsai. and JS. P., "Cat swarm optimization," in *PRICAI: Trends in Artificial Intelligence, Lecture Notes in Computer Science*, Q. Yang, G. Webb (eds.), vol. 4099. Berlin, Heidelberg: Springer, pp. 40992006.

[47] M. A. Ahmed, A. T. Rashid, A. Soran, M. Saeed and G. A.J., "Pulido, cat swarm optimization algorithm: A survey and performance evaluation," *Computational Intelligence and Neuroscience*, vol. 2020, no. 1, pp. 1–20, 2020.

[48] T. Kokilavani and G. Amalarethinam, "Load balanced min-min algorithm for static meta-task scheduling in grid computing," *International Journal of Computer Applications*, vol. 20, no. 2, pp. 43–49, 2011.

[49] K. Pradeep and T. P. Jacob, "CGSA scheduler: A multi-objective-based hybrid approach for task scheduling in cloud environment," *Information Security Journal: A Global Perspective*, vol. 27, no. 2, pp. 77–91, 2020.

[50] K. Sekaran, M. S. Khan, R. Patan, A. H. Gandomi, P. V. Krishna *et al.,* "Improving the response time of m-learning and cloud computing environments using a dominant firefly approach," *IEEE Access*, vol. 7, pp. 30203–30212, 2019.

[51] K. Mohit and S. Subhash, "Dynamic load balancing algorithm to minimize the makespan time and utilize the resources effectively in cloud environment," *International Journal of Computers and Applications*, vol. 39, pp. 1–10, 2017.

[52] N. Singh and N. Kaur, "Efficient task scheduling over cloud computing with an improved firefly algorithm," *International Journal of Engineering Development and Research*, vol. 4, no. 2, pp. 1514–1518, 2016.

[53] J. P. B. Mapetu, Z. Chen and L. Kong, "Heuristic cloudlet allocation approach based on optimal completion time and earliest finish time," *IEEE Access*, vol. 6, pp. 61714–61727, 2018.

[54] A. Thakur and M. S. Goraya, "A taxonomic survey on load balancing in cloud," *Journal of Network and Computer Applications*, vol. 98, no. C, pp. 43–57, 2017.

[55] S. K. Mishra, S. Bibhudatta and P. P. Parida, "Load balancing in cloud computing: A big picture," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, 2020.

[56] R. B. Shaikh and M. Sasikumar, "Data classification for achieving security in cloud computing," *Procedia Computer Science*, vol. 45, no. 12, pp. 493–498, 2015.

[57] I. Strumberger, N. Bacanin, T. Milan and T. Eva, "Resource scheduling in cloud computing based on a hybridized whale optimization algorithm,Applied Sciences". Vol. 9, no. 22, pp. 4893, 2019.

[58] D. Pebrianti, A. Nurnajmin, B. Luhur, A. Hasma, Z. Zainah *et al.,* "Extended bat algorithm as an improved searching optimization algorithm: Methods and protocols," in *Proc. National Technical Seminar on Underwater System Technology*, Singapore, Springer, vol. 538, pp. 239–237, 2018.

[59] D. Chaudhary and B. Kumar, "Cloudy GSA for load scheduling in cloud computing," *Applied Soft Computing*, vol. 71, pp. 861–871, 2018.

[60] S. Torabi and F. S. Esfahani, "A dynamic task scheduling framework based on chicken swarm and improved raven roosting optimization methods in cloud computing," *Journal of Supercomputing*, vol. 74, pp. 2581–2626, 2018.

[61] K. Dubey, K. Mohit and S. C. Sharma, "Modified HEFT algorithm for task scheduling in cloud environment," *Procedia Computer Science*, vol. 125, pp. 725–732, 2018.

[62] J. Meena, M. Kumar and M. Vardhan, "Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint,IEEE," *Access*, vol. 4, pp. 5065–5082, 2016.

[63] L. Mashayekhy, "An online mechanism for resource allocation and pricing in cloud," *IEEE Transactions on Computers*, vol. 65, no. 4, pp. 1172–1184, 2016.

[64] P. Rekha and M. Dakshayini, "Efficient task allocation approach using genetic algorithm for cloud environment," *Cluster Computing*, vol. 22, no. 21, pp. 1241–1251, 2019.

[65] C. Joshua, S. Yao, N. Chen, H. Xing and L. Zhenhua, "Online optimization in cloud resource provisioning: Predictions, regrets, and algorithms," in *Joint Int. Conf. on Measurement and Modeling of Computer Systems*, NY, USA, ACM, pp. 47–48, 2019.

[66] S. Afzal and G. Kavitha, "Load balancing in cloud computing—A hierarchical taxonomical classification," *Journal of Cloud Computing*, vol. 8, no. 1, pp. 1–24, 2019.

[67] Y. Kumar and P. K. Singh, "Improved cat swarm optimization algorithm for solving global optimization problems and its application to clustering," *Applied Intelligence*, vol. 48, no. 9, pp. 2681–2697, 2017.

[68] A. Nazia and D. Huifang, "A hybrid metaheuristic for multi-objective scientific workflow scheduling in a cloud environment," *Applied Sciences*, vol. 8, no. 4, pp. 538, 2018.

[69] W. Zhong, Y. Zhuang and J. Sun, "A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine," *Applied Intelligence*, vol. 48, no. 11, pp. 4072–4083, 2018.

[70] M. Ashouraei, S. N. Khezr, R. Benlamri and N. J. Navimipour, "A new SLA-aware load balancing method in the cloud using an improved parallel task scheduling algorithm," in *Proc. IEEE 6th Int. Conf. on Future Internet of Things and Cloud*, Barcelona, Spain, pp. 71–76, 2018.

[71] N. Sharma and S. Maurya, "SLA-based agile VM management in cloud and datacenter," in *Proc. Int. Conf. on Machine Learning, Big Data, Cloud and Parallel Computing*, Faridabad, India, pp. 252–257, 2019.

[72] K. Ajay and S. Bawa, "A comparative review of meta-heuristic approaches to optimize the SLA violation costs for dynamic execution of cloud services," *Soft Computing*, vol. 24, no. 6, pp. 3909–3922, 2020.

[73] X. Song, Y. Ma and D. Teng, "A load balancing scheme using federate migration based on virtual machines for cloud simulations," *Mathematics Problems in Engineering*, vol. 2015, pp. 1–11, 2015.

[74] J. R.Cornabas, "A distributed and collaborative dynamic load balancer for virtual machine," in *Proc. European Conf. on Parallel Processing*, Ischia, Italy, pp. 641–648, 2010.

[75] T. Jamal and A. Enrique, "Metaheuristics for energy-efficient data routing in vehicular networks," *International Journal of Metaheuristics*, vol. 4, no. 1, pp. 27–56, 2015.

[76] K. Saleem and K. N. Fisal, "Enhanced ant colony algorithm for self-optimized data assured routing in wireless sensor networks," in *Proc. IEEE Int. Conf. on Networks*, Singapore, pp. 422–427, 2012.

[77] S. Mohan and R. Garg, "Harmony-inspired genetic algorithm for rack-aware energy-efficient task scheduling in cloud data centers," *Engineering Science and Technology Journal*, vol. 23, no. 1, pp. 211–224, 2020.

[78] D. Amancio, C. Comin, C. D. Travieso, G. Bruno, O. R. Francisco *et al.,* "A systematic comparison of supervised classifiers," *PLOS ONE*, vol. 9, pp. e94137, 2014.

[79] C. Tsai and J. J. P. C. Rodrigues, "Metaheuristic scheduling for cloud: A Survey," *IEEE Systems Journal*, vol. 8, no. 1, pp. 279–291, 2014.

[80] P. Mohapatra, S. Chakravarty and P. K. Dash, "Microarray medical data classification using kernel ridge regression and modified cat swarm optimization-based gene selection system," *Swarm and Evolutionary Computation*, vol. 28, no. Suppl. 8, pp. 144–160, 2016.

[81] L. Pappula and D. Ghosh, "Cat swarm optimization with normal mutation for fast convergence of multimodal functions," *Applied Soft Computing*, vol. 66, no. 4, pp. 473–491, 2018.

[82] A. Sharma, A. Zaidi, R. Singh and S. Jain, "Optimization of SVM classifier using firefly algorithm," in *Proc. IEEE Second Int. Conf. on Image Information Processing*, India, pp. 198–202, 2013.

[83] D. Jinglin, Y. Liu, Y. Yu and W. Yan, "A Prediction of precipitation data based on support vector machine and particle swarm optimization algorithms," *Algorithms*, vol. 10, no. 2, pp. 1–57, 2017.

[84] T. Thomas, A. P. Vijayaraghavan and S. Emmanuel, "Applications of decision trees," in *Machine Learning Approaches in Cyber Security Analytics*. Springer, Singapore pp. 157–184, 2020.

[85] S. R. Basha, J. K. Rani, J. P. Yadav and G. R. Kumar, "Impact of feature selection techniques in text classification: an experimental study," *Journal of Mechanics of Continua and Mathematical Sciences, Special Issue*, vol. 3, no. 3, pp. 39–51, 2019.

[86] I. Okfalisa, G. Mustakim and N. G. I. Reza, "Comparative analysis of k-nearest neighbor and modified k-nearest neighbor algorithm for data classification," in *Proc. 2nd Int. Conf. on Information Technology, Information Systems and Electrical Engineering*, Yogyakarta, Indonesia, pp. 294–298, 2017.

[87] S. Shakya and S. Sigdel, "An approach to develop a hybrid algorithm based on support vector machine and naive bayes for anomaly detection," in *Proc. Int. Conf. on Computing, Communication and Automation*, Noida, India, pp. 323–327, 2017.

[88] N. A. S. Selvakumari, "A voice activity detector using svm and naïve bayes classification algorithm," in *Proc. Int. Conf. on Signal Processing and Communication*, Coimbatore, India, pp. 1–6, 2017.

[89] B. Y. Pratama and R. Sarno, "Personality classification based on twitter text using NB, KNN and SVM," in *Proc. Int. Conf. on Data and Software Engineering*, Yogyakarta, India, pp. 170–174, 2015.

[90] E. Raczko and B. Zagajewski, "Comparison of support vector machine, random forest and neural network classifiers for tree species classification on airborne hyperspectral APEX images," *European Journal of Remote Sensing*, vol. 50, no. 1, pp. 144–154, 2017.

[91] A. S. Tomala, E. Raczko and B. Zagajewski, "Comparison of support vector machine and random forest algorithms for invasive and expansive species classification using airborne hyperspectral data," *Remote Sense*, vol. 12, no. 3, pp. 516, 2020.

[92] Y. Al-Amrani, L. Mohamed and K. Eddine, "Random forest and support vector machine based hybrid approach to sentiment analysis," *Procedia Computer Science*, vol. 127, pp. 511–520, 2018.

[93] M. Heidarysafa, K. Kamran, B. Donald, J. Meimandi and K. B. Laura, "An improvement of data classification using random multimodel deep learning," *International Journal of Machine Learning and Cybernetics*, vol. 8, pp. 298–310, 2018.

[94] Synchronous Machine Datasets, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml [Accessed: 20-May-2020].

[95] R. N. Calheiros, R. Ranjan, C. A. F. D. Rose and R. Buyya, "CloudSim: A novel framework for modeling and simulation of cloud computing infrastructures and services," *Computing Research Repository*, vol. 1, pp. 1–9, 2009.

[96] K. B. Duan, J. C. Rajapakse and M. N. Nguyen, "One-vs.-one and one-vs.-all multiclass SVM-RFE for gene selection in cancer classification," in *EvoBIO 2007: European Conf. on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, Valencia, Spain, Lecture Notes in Computer Science book series (LNCS), vol. 4447, pp. 47–56, 2007.

[97] M. Junaid, A. Sohail, A. Ahmed, A. Baz, I. A. Khan *et al.,* "A hybrid model for load balancing in cloud using file type formatting," *IEEE Access*, vol. 8, pp. 118135–118155, 2020.

[98] A. S. Girsang, A. S. Manalu and K. Huang, "Feature selection for musical genre classification using a genetic algorithm," *Advances in Science Technology and Engineering Systems Journal*, vol. 4, no. 2, pp. 162–169, 2019.

[99] D. Karim, A. Cherif and S. Hadji, "An optimization of audio classification and segmentation using GASOM algorithm," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 4, pp. 143–157, 2019.

[100] B. Desgraupes, *Clustering Indices*. Lab Modal'X: University of Paris Ouest, pp. 1–34, 2013.

[101] K. Tao, S. Lin and Y. Zhang, "A novel local classification method for multimedia semantic analysis," in *Proc. IEEE Int. Conf. on Multimedia and Expo*, New York, NY, USA, pp. 402–405, 2009.

[102] M. G. Rani, "A moth-flame optimization algorithm for web service composition in cloud computing: simulation and verification," *Software*, vol. 48, no. 10, pp. 1865–1892, 2018.