scientific reports

Check for updates

OPEN

# A hybrid cloud load balancing and host utilization prediction method using deep learning and optimization techniques

Sarita Simaiya[1]✉, Umesh Kumar Lilhore[1], Yogesh Kumar Sharma[2], K. B. V. Brahma Rao[3], V. V. R. Maheswara Rao[4], Anupam Baliyan[1], Anchit Bijalwan[5]✉ & Roobaea Alroobaea[6]

**Virtual machine (VM) integration methods have effectively proven an optimized load balancing in cloud data centers. The main challenge with VM integration methods is the trade-off among cost effectiveness, quality of service, performance, optimal resource utilization and compliance with service level agreement violations. Deep Learning methods are widely used in existing research on cloud load balancing. However, there is still a problem with acquiring noisy multilayered fluctuations in workload due to the limited resource-level provisioning. The long short-term memory (LSTM) model plays a vital role in the prediction of server load and workload provisioning. This research presents a hybrid model using deep learning with Particle Swarm Intelligence and Genetic Algorithm ("DPSO-GA") for dynamic workload provisioning in cloud computing. The proposed model works in two phases. The first phase utilizes a hybrid PSO-GA approach to address the prediction challenge by combining the benefits of these two methods in fine-tuning the Hyperparameters. In the second phase, CNN-LSTM is utilized. Before using the CNN-LSTM approach to forecast the consumption of resources, a hybrid approach, PSO-GA, is used for training it. In the proposed framework, a one-dimensional CNN and LSTM are used to forecast the cloud resource utilization at various subsequent time steps. The LSTM module simulates temporal information that predicts the upcoming VM workload, while a CNN module extracts complicated distinguishing features gathered from VM workload statistics. The proposed model simultaneously integrates the resource utilization in a multi-resource utilization, which helps overcome the load balancing and over-provisioning issues. Comprehensive simulations are carried out utilizing the Google cluster traces benchmarks dataset to verify the efficiency of the proposed DPSO-GA technique in enhancing the distribution of resources and load balancing for the cloud. The proposed model achieves outstanding results in terms of better precision, accuracy and load allocation.**

Cloud computing enables the optimum utilization of computing resources using its dynamic service model. Cloud services require adaptive distribution of computing resources with dynamic resource scalability, which helps in delivering a Quality-of-Service (QoS) with the most minor resource expenses. However, in complicated cloud settings with changing workloads, it might be challenging to implement dynamic resource distribution for heterogeneous applications[1].

Cloud storage solutions have seen a considerable increase in demand since the emergence of the expanding IoT with Industry 4.0 technology regulations for various data-processing activities such as storage spaces, searching for resources, and mapping. The cloud is linked with IoT-enabled applications across multiple industry verticals, which helps them to utilize computing resources from remote locations. Cloud computing represents a pay-per-use mode of offering computing resources accessible on-demand from hosting companies[2].

[1]Department of Computer Science and Engineering, Chandigarh University, Gharuan, Mohali, Punjab 140413, India. [2]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Greenfield, Vaddeswaram, Guntur, AP, India. [3]Department: Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India. [4]Department of Computer Science and Engineering, Shri Vishnu Engineering College for Women(A), Bhimavaram, India. [5]Arba Minch University, Arba Minch, Ethiopia. [6]Department of Computer Science, College of Computers and Information Technology, Taif University, P. O. Box 11099, 21944 Taif, Saudi Arabia. ✉email: saritasimaiya@gmail.com; anchit.bijalwan@amu.edu.et

Cloud computing is becoming incredibly important in the educational and IT sectors and everyday life because of many characteristics, including no initial cost, immediate customer service, reliability, adaptability, and simple accessibility. The phrases Platform as a service (PaaS), Software as a service (SaaS), Data as a Service (DaaS) and Infrastructure as a service (IaaS) are utilized in the solutions offered by organizations such as Google, Microsoft, Amazon, and many other individuals, for devices, infrastructure, software, applications, and any technology. Clients of cloud-based services can utilize them from anywhere, on any device, and at any time. The cloud-computing allows the client to access its resources or programs according to their specific requirements[3].

Many data centers have been established worldwide due to the growing acceptance of cloud technology. These data centers can offer various computing services, such as storage spaces, Networks, servers, and software for Industry, e-commerce, and other online uses. A novel cloud computing architecture mainly includes significant computing resources[4]. Cloud computing services primarily rely significantly on its data center. Because these data centers consist of different kinds of technology, including servers and storage spaces, they represent a significant cause of global power consumption. In addition, it is anticipated that the power consumed by data centers worldwide rise by 70% because of the ongoing proliferation of cloud services. To accomplish long-term growth, it has become necessary to utilize environmentally friendly computing strategies and reduce data center's energy utilization due to their substantial power use, rapid rate of expansion, and cumulative environmental footprint[5].

This issue can be reduced with precise forecasting of the potential workload behaviour for resources via accurate observation. Effective monitoring and keeping a record of how much time and effort are expended on various resources, such as memory, central processing unit, space for storage, and the bandwidth of the network, helps solve this issue. These hints of previous use can then be examined and used in predictive modelling. This sense of anticipation is essential to provision resources in the cloud[6] properly. The primary strategies for increasing resource utilization and reducing excessive provision issues include virtualization, Virtual machines and dynamic workload. Virtualization enables individual physical devices (PMs) to be split into multiple identical VMs, which allow the exploitation of different physical capabilities[7].

The remaining benefits of virtualization include better server management, improved resource utilization, and cost-effective data center architecture. But when the server systems aren't in execution, their resources, such as power and energy, are squandered, and consequently, data center incompetence occurs through idle power consumption. An excellent way to cut down on the electricity and energy used by data centers is to consolidate machines and virtual servers. The VMs are combined over PMs, so no additional PMs need to be installed[8]. In massive operations, integration of VMs and computing resource allocation is essential as groups of machines address complex optimization issues. A cloud computing system must utilize all of its capacities to the most significant potential to satisfy the rising demand for its products and services.

A further considerable symmetry concern must be addressed: higher and lower oscillations within cloud workloads. Any excessive allocations of computing resources might increase energy consumption and raise expenses. In existing research, cloud computing has widely utilized deep learning and machine learning methods for load balancing. Some research optimization-based techniques are also used in VM machine and resource mapping[9]. The critical contribution of the study is as follows:

- This research presents Deep learning with Particle Swarm Intelligence and Genetic Algorithm based "DPSO-GA", a Hybrid model for dynamic workload balancing in cloud computing.
- A PSO method also helps to fine-tune the Hyperparameters. The proposed model integrates the resource utilization in a multi-resource utilization array, which helps to overcome the load balancing and over-provisioning issues.
- The proposed hybrid model is divided into two modules. The first phase utilizes a hybrid PSO-GA approach to address the prediction challenge by combining the benefits of the two methods.
- In the second phase, CNN-LSTM is utilized. Before using the CNN-LSTM Approach to forecast the consumption of resources, a hybrid approach, PSO-GA, is used for training it. In the proposed framework, a one-dimensional CNN and LSTM are used to forecast cloud resource utilization at various subsequent time steps.
- Comprehensive simulations are carried out utilizing the RUBiS and Google cluster datasets to verify the efficiency of the proposed DPSO-GA technique in enhancing the distribution of resources and load balancing for the cloud.
- To the best of our knowledge, this is the first attempt to utilize the fusion of deep learning (CNN-LSTM) and optimization techniques (PSO-GA) for workload prediction and load balancing in the cloud.

The complete article is organized as follows: Section two covers a literature review on cloud computing, load balancing, deep learning and optimization methods in load balancing and their challenges. Section three presents materials and techniques which offer the proposed model's working, design, procedures, and parameters. Section four covers the experimental results and comparison of the proposed model and existing solutions; this section also covers a discussion subsection. Section five covers the conclusion and future direction of the research, limitations and critical aspects.

## Literature review

The difficulties of consolidating cloud data centers need to be studied. Combining VMs is possible through virtualization, which improves resource utilization and reduces energy use. Cloud service providers build many virtual machines on one physical Host. The newest energy-efficient virtual machine integration techniques for data storage processors, including data centers in the cloud, are highlighted in this article section. Research gaps in previous research were also evaluated through various comparative analyses.

## Optimization methods based solutions

VM fusion offered an in-depth evaluation and analysis of the job based on the latest research on load balancing. A researcher mainly aims at a pre-emptive adaptive VM consolidation across cloud data centers, and their findings showed significant discoveries. A load-balancing method is essential in addressing the relationship between cloud resources and performing effective resource utilization.

A resource-efficient and dynamic consolidation of virtual machines-based technique was developed in[10]. The proposed method was based on four algorithms created at different VM fusion phases. The latest solution for VM load balancing in a cloud-based data center that considers SLAs and power consumption was put forward in[11]. A VM distribution method based on a reliable basic PSO was offered after an approach for identifying the overloaded and underloaded VMs. Enhanced distribution by learning automation, based on GA and ACO knowledge, was developed in[12] to reduce energy use. The proposed model utilizes the GA method for finding a suitable machine for a particular workload in the cloud.

An ACO workload distribution approach is presented in[13] to integrate the VMs in a cloud-based system. The research mainly focuses on reducing energy use and improving the distribution of workloads while completing more tasks at high performance; they established lower criteria utilizing the data canter's overall workload utilization and employed ant colony optimization to minimize the frequency of VMs movements. Research[14] explored optimization and ML-based work for load distribution in the cloud environment. The proposed model utilizes a bee colony optimization method. In this work, an energy utilization calculation was also performed for cloud data centers using a Planet Laboratory that included many PlanetLab virtual machines with large-scale modelling configurations.

The discrete-time systems Markov chain model was put forward in[15] to forecast prospective resource utilization. The Host's reliability framework model has the potential to be used to classify hosts according to their state more precisely. Researchers then proposed a multi-objective virtual machine positioning method to find the optimal VMs for host mappings using the dominance-based multiple-purpose Ant Bee Colony methodology.

A multiple objectives technique centered on the PSO technique for the virtual machines' allocation problem, VM-OPSO architecture, was developed in[16]. The proposed VM-OPSO utilizes the population entropy technique to optimize the Virtual Machine Platform and accelerate integration to the most effective solution whilst increasing the number and types of the provided alternatives. In[17], it includes information about existing cost-effective methods and supports researchers in identifying the best practical way. They investigated many state-of-the-art energy-efficient algorithms from many perspectives, including design, modelling, and measurements. The same experimental parameters were used to create and analyze alternate approaches employing the Cloud-Sim software.

A GA-based approach was presented in[18] for adaptive virtual machine migration and host placement. This method has four unique characteristics: first, it chooses locations of hosting where VMs have to be moved that have the slightest access delay, and second, it reduces the total amount of VM migration. A multi-objective combining virtual machines technique was subsequently created using an ant colony system via double thresholds, as mentioned in[19,20]. VMs are migrated to another host when the Host is overburdened or underloaded. The proposed technique used two CPU utilization metrics to determine the Host's load condition. During combining, ACO was utilized to determine which VMs and servers should be transferred simultaneously, using varied techniques depending on the Host's traffic condition. It was built on a tactical competition that included a cloud provider, and all computing devices participated.

## Deep learning-based solution

The forecasting accuracy was enhanced using neural networks with asymmetric evolution using standard adaptation[21]. The adaptive technique improves accuracy by examining the scope of possible responses from many viewpoints and applying various potential answers. In contrast to gradient-based learning methods, this reduces the possibility of becoming spotted in optimal local circumstances. To promote precise scalability operation,[22] presented a better prediction model under a neural network. The proposed technique classified the VMs depending on how they were employed before forecasting how they would be used in future periods. The algorithm used a multilayered perceptron classification approach to achieve the above goal. Containerized cloud-based operations can use the performance-aware automated scaling for the cloud elastic design published in[23] to dynamically allocate the resources available in response to changing demand. A flexible scaling technique that predicts future workload demands has been developed in[24] using Convolutional neural networks and K-means that estimate the utilization of virtual machine (VM) assets, including memory and CPU. The investigation employs a Bayesian technique to forecast VM utilization of resources throughout the typical workday.

A forecasting algorithm based on a recurring LSTM neural network model was presented to predict future HTTP workloads. The suggested method uses an ANN regarding elastic accelerating and automatic deployment-related scaling. In[25], a multi-objective adaptive algorithm was used to estimate both memory and CPU utilization in addition to the consumption of energy for the next time slot. The researchers of[26] offered an approach for forecasting the cloud data center amount of work. The forecasting method discussed in the article was constructed by using an LSTM network. Forecasting data stored in the cloud center demand has been suggested in[27,28]. The forecast made in the article was constructed on a network with LSTM.

The article[29] addresses the issue of load estimation in cloud data centers. The LSTM network has been used to create the computational load forecasting model. Based on the findings from the experiment, the suggested approach has greater accuracy in forecasting than the other approaches that were also considered. In addition, many cloud service providers utilize user-defined resource criteria to offer auto-scaling features, limiting the ability to build models depending on various workload factors.

## Hybrid methods

Adapting a specific workload sequence is typically straightforward for an estimation framework that employs only one predicting model; however, it is difficult with data from the real world, while the workload pattern varies quickly over time[30]. These situations continue despite excessive capacity and under-provisioning. Two internet-based algorithms for collaborative learning[31] were invented to predict their workloads. Models respond quickly to alterations in the cloud load request trend. An innovative cloud load forecasting method[32] was developed for dealing with continually altering workloads. It employs numerous predictors to construct a combination of models capable of correctly anticipating real-world loads. Clouds Intelligence, an algorithm for predicting loads of work, was built with the help of several forecasts[33].

To increase the precision of forecasts, it incorporates seven unique predicting algorithms across the domains of statistical analysis of time series, linear regression, and artificial intelligence. For forecasting a server workload pattern in a cloud-based storage center, a cloud load prediction based on a weighted fractal support vector machine algorithm is presented[34]. In this study, parametric optimization using a method called particle optimization technique was created. A different approach[35] focuses on predicting mega-variant resource consumption in cloud centers' data. These resources comprise bandwidth for the network, processor, and storage. The method indicates resource usage using CNN and LSTM models. In the start phase, the linear connections with the mega-variant data are filtered using the matrix auto-regression approach.

Limiting the number of physical machines (PMs) that are actively processing data was the primary emphasis of previous methods. Problems with energy usage in VM provisioning and load variability are seldom addressed together. A new approach to virtual machine allocation and implementation, AFED-EF (Adaptive energy-aware VM allocation and deployment mechanism), was suggested by[36] for use in IoT applications to address these issues. When it comes to virtual machine allocation and placement, the recommended approach performs well and can effectively manage load fluctuations. Employing a real-world workload consisting of over a thousand PlanetLab VMs, the author conducted a thorough experimental study. When compared with additional energy-aware methods, AFED-EF performed better in terms of performance, SLA violations, and overall energy usage.

To address quality of service and SLA concerns in an SDDC operating in a CAV environment, another article[37] introduces an energy-efficient virtual machine cluster placement technique called EVCT. Using a weighted directed network as a model, the EVCT method makes use of VM similarity to solve the VM deployment issue. Using the "maximum flow and minimum cut theory" to reduce the graph into directed segments while accomplishing high energy-efficient positioning for VMs, EVCT takes traffic across VMs into account. Improved consumption of energy costs, improved scalability, and outstanding level of service for consumers are all benefits of the suggested approach. The authors also conducted a number of tests to test how well the EVCT handled a real-world load.

The issue of lowering the cloud data center's excessive consumption of energy while limiting SLA breaches is discussed in[38]. Existing methods for managing energy resources in cloud data centers primarily aim to reduce power use, even if there are several alternatives. In order to maximize energy efficiency while decreasing power consumption, the author of this research suggests two new adaptive algorithms that take this into account. Cloud data center SLA, and violation rate analysis, are also covered. The proposed energy-aware algorithms include application kinds in addition to CPU and memory resources when deploying virtual machines, which is different from the current methods. According to the testing findings, the suggested methods may successfully reduce energy usage within cloud data centers as they maintain low SLA violations, outperforming the current energy-saving solutions."

Table 1 presents a comparative review of existing strategies in cloud performance enhancement. It was discovered that the previous works could not simulate and anticipate workload requirements for numerous VMs.

| References | Methods | Predication resource | Key points | Dataset | Performance criteria |
|---|---|---|---|---|---|
| [1] | Bidirectional LSTM | VM based | VMs workload distribution on various time series data | GWAT- 12 and 13 | Precision, memory utilisation |
| [6] | CNN with LSTM | VM based | Overcome noise in the data and workload | GWAT dataset | Accuracy, precision, utilization of resources, i.e., network, storage, memory |
| [10] | GRU with CNN | PM based | Overcome migration of load, energy | Telecom dataset | CPU utilisation, precision |
| [17] | PSO with SVM | VM based | Energy uses and workload balancing | Google cluster | Accuracy, precision, utilization of resources, i.e., network, storage, memory |
| [19,20] | PSO-DBN | PM based | Overcome noise in the data, workload | GWAT- 12 and 13 | Accuracy, precision, memory utilization |
| [39] | Learning automata | PM based | Overcome migration of load, energy | CO-Mon dataset | Precision, accuracy, F-measure and memory utilization |
| [23] | Ensemble learning methods with PSO | VM based | Energy uses and workload balancing | Planet Lab dataset | CPU utilisation, precision |
| Proposed Model | DPSO-GA (Deep learning with POS-GA) | VM based | Overcome workload; reduce overloading and resource utilization | Google cluster dataset | Accuracy, precision, and utilization of resources, i.e., network, storage, and memory |

**Table 1.** Review of existing methods for cloud computing.

Because they were developed and educated for only one virtual machine (VM) workload, the idea that VMs are independent and contain associated applications requiring several VMs' capacity is rejected.

## Materials and methods

This section covers the materials and methods related to the present research.

### GA method

The optimization approach known as GA is frequently employed in complicated and massive systems to determine results near the optimal level. Consequently, GA is an excellent technique for training a neural network model for learning. A standard GA is based on a population search method influenced by the process of natural selection that relies on the concept of persistence of the healthiest[40]. GA's primary components are (a) chromosome, (b) selection process, (c) mutation process, (d) crossover, and (e) calculation and evaluation of fitness function.

We start by arbitrarily initializing a population of chromosomes, which we typically consider as potential alternatives to scheduling for any specific task. The allocation of activities to certain machines inside that chromosome allows us to obtain a fitness value (Makespan), which is acquired. After receiving the initial population, we assess each chromosome in the group according to its unique fitness value.

A smaller makespan is always desired to fine-tune the mapping. We use an allocation scheme that statistically replicates a specific chromosome and eliminates others. At the same time, we discover that improved mappings are more likely to be repeated in future generations. At the same time, the number of individuals stays constant over each age. Algorithm 1 presents the working of the GA method[41].

Input: Population set (Ps), Probability of Crossover (Cp), Probability of Mutation (Cm),

Output: Best chromosomes

The initialization of variables, i.e., Population, Population dimension vector ($P_{VD}$)

$$P_s = [P_1 \ldots\ldots\ldots P_n] \quad \text{and} \quad P_{VD} = [P_{i1}, \ldots\ldots\ldots P_{VD}],$$

Initial Gen=1, $Gen_{max}$:$Maximum\ generation$

While(Gen <= $Gen_{max}$) //Repeat steps 3 to

Determine the fitness value by fitness function eq. (1)

3.1 $$Fitness = \frac{1}{Mean\ Absolute\ Error} \quad (1)$$

3.2 A Mean Absolute Error (MAE) can be calculated using equation 2.

$$MAE = \frac{1}{N}\sum_{i=0}^{N}[\ |y_i - z_i|\ ] \quad (2)$$

Where MAE is Mean Absolute Error, $y_i$ is the predicted and $z_i$ is the actual, and N is the number of variables.

The average of MAE can be measured by eq (3).

$$MAE = \frac{1}{N}\sum_{i=0}^{N}[|e_i|] \quad (3)$$

Where $|e_i|$: Input count can be calculated using equation 3.1.

$$|e_i| = |y_i - z_i| \quad (3.1)$$

Determine best chromosome // call_Det_best_chro()

Select best chromosome // call_Sel_best_chro()

Determine Crossover // Crossover()

Determine mutation // mutation()

Return Best chromosomes

END

**Algorithm 1** GA algorithm

### PSO method

The swarm intelligence subcategory of optimization algorithms includes the renowned PSO algorithm. There are numerous scenarios within the literature in which PSO is used to train neural network algorithms effectively. The method comprises several particles that analyze potential solutions across the issue space, eventually arriving at the optimal ones[42]. The detailed algorithm is presented in Algorithm 2.

Input: Initial Population set ($I_{Pi}$), population size(pi) , $P_{best}$ (best chromosome in local), $Gl_{best}$ (best chromosome in global),
Output: Best particles $P_{best}$
Initialisation of variables i.e., Population,
$I_{Pi} = [\, I_{P1} \ldots \ldots \ldots I_{Pn} \,]$
Determine the fitness of $I_{Pi}$ by equation (1)
Calculate the local best chromosome $p_{best} = Local_{Patricle}()$
Calculate the global best chromosome. $gl_{best} = Local_{Patricle}()$
Calculate and update the velocity for each chromosome
Update the position for each chromosome
Return the $p_{best}$ and $gl_{best}$
END

**Algorithm 2** PSO algorithm

## CNN model

Deep learning techniques rely heavily on artificial neural networks (ANNs). Recurrent neural networks (RNNs), which take input as a sequence or periodic information, are a particular kind of ANN. A different type of neural network called a CNN may find crucial details in time series and visualize inputs. It is essential for data analysis, such as object and image classification. The CNN model contains three necessary layers (Convolution: 1, Pooling: 2 and Fully Connected: 3). Figure 1 presents the architecture of the Basic CNN model[43].

Deep learning techniques rely heavily on artificial neural networks (ANNs). Recurrent neural networks (RNNs), which take input as a sequence or periodic information, are a particular kind of ANN. A different type of neural network called a CNN may find crucial details in time series and visualize inputs. It is essential for data analysis, such as object and image classification. The CNN model contains three necessary layers (Convolution: 1, Pooling: 2 and Fully Connected: 3).

## LSTM model

It mostly applies to deep learning. Several RNNs possess the capacity to learn long-term connections, particularly in tasks involving sequence anticipation. Aside from singular observations like images, LSTM includes feedback links, making it suited to interpreting the complete data sequence. It uses automatic translation and the recognition of objects. A unique version of RNN called LSTM exhibits outstanding reliability on various issues. Figure 2 shows the basic architecture of the LSTM model[44,45].

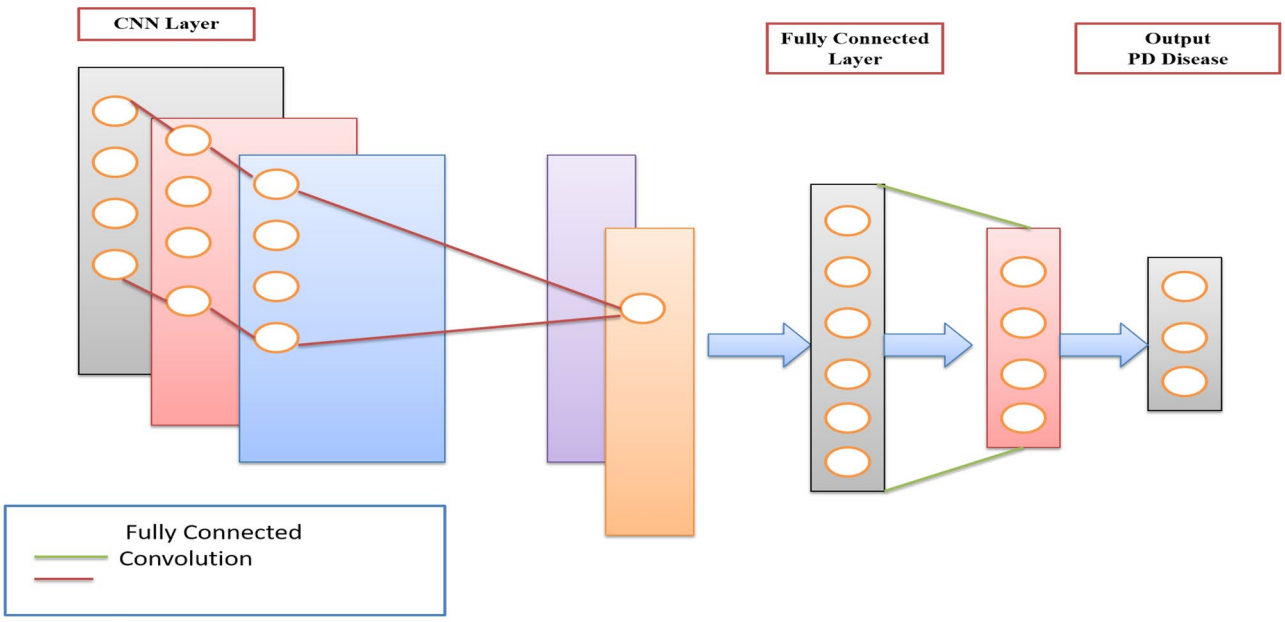


**Figure 1.** The basic architecture of the CNN model.

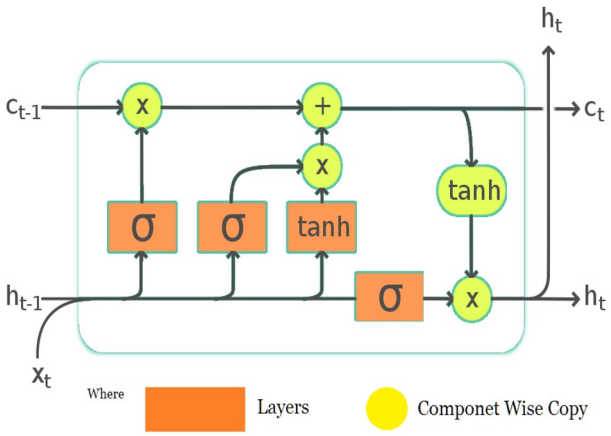


**Figure 2.** The basic architecture of the LSTM model.

A memory cell, described as a "cell status" which preserves its state over time, performs a crucial part in an LSTM model. A horizontal line that travels across the top portion, as presented in Fig. 2, represents the cell's status. It can be visualized as a network of unmodified conveyor belts across which knowledge travels. The following equations, from 4 to 6, are used to determine the computations in LSTM.

$$F_t = \{\sigma[(W_F) \times (H_{t-1}), IX_t] + B_F\} \tag{4}$$

$$I_t = \{\sigma[(W_I) \times (H_{t-1}), IX_t] + B_I\} \tag{5}$$

$$O_t = \{\sigma[(W_O) \times (H_{t-1}), IX_t] + B_I\} \tag{6}$$

where $I_t$: Input Gate, $F_t$: Foregt Gate and $O_t$ : Output Gate, $\sigma$ : Sigmoid Function, $W_G$: Weight for a particular Gate, $H_t$: Output of the current block, $IX_t$ : Input data at present timestamp and $B_G$ : Bias value for a particular Gate.

The cell state and a candidate state with a final output can be calculated using Eqs. (7, 8, and 9).

$$\widehat{C}_t = \{\tanh[(W_C) \times (H_{t-1}), IX_t] + B_C\} \tag{7}$$

$$C_t = \{[(F_t \times (C_{t-1})] + (I_t + \widehat{C}_t)\} \tag{8}$$

$$H_t = [O_t \times \tanh C_t] \tag{9}$$

where $C_t$: Memory (Cell) state at a particular time stamp t, $\widehat{C}_t$: Candidate Cell state at a particular time stamp t.

## Proposed DPSOGA model

To handle the workload balancing issues, this research presents Deep learning with Particle Swarm Intelligence and Genetic Algorithm based "DPSO-GA", a Hybrid model for dynamic workload balancing in cloud computing. The proposed model simultaneously integrates the resource utilization in a multi-resource utilization, which helps overcome the load balancing and over-provisioning issues. The proposed model works in two phases[46]. The details are as follows. Figure 3 shows the architecture of the proposed model DPSOGA model.

- First phase: The first phase utilizes a hybrid PSO-GA approach to address the prediction challenge by combining the benefits of the two methods. A PSO-GA method also helps to fine-tune the Hyperparameters. In this phase, a dynamic decision-making method called the PSOGA is suggested for investigating the goal of the distribution of resources strategy. The PSO method helps to fine-tune the Hyperparameters, adjust their values automatically, and select which parameters to encode as a particle.
- Second phase: In the second phase, CNN-LSTM is utilized with PSO-GA. Before using the CNN-LSTM Approach to forecast the consumption of resources, a hybrid approach, PSO-GA, is used for training it. In the proposed framework, a one-dimensional CNN and LSTM are used to forecast cloud resource utilization at various subsequent time steps. The LSTM module simulates temporal information that predicts the upcoming VM workload, while a CNN module extracts complicated distinguishing features gathered from VM workload statistics.

In the proposed framework, a one-dimensional CNN and LSTM are used to forecast the CPU utilization on cloud-based servers at various subsequent time steps. The LSTM module simulates temporal information that predicts the upcoming VM workload, while a CNN module extracts complicated distinguishing features

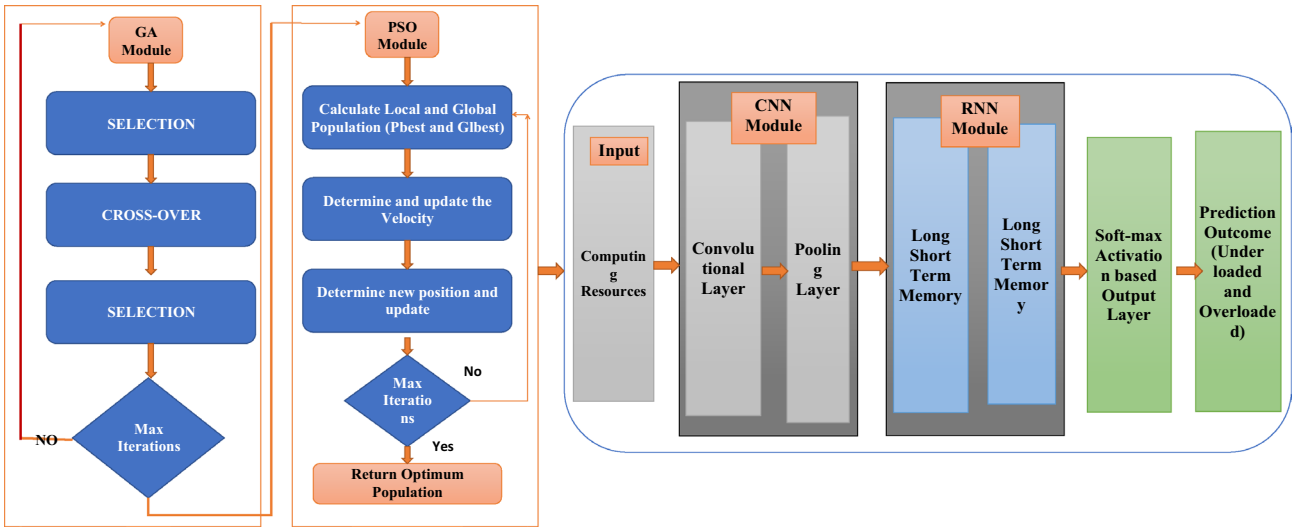


**Figure 3.** The architecture of the proposed model DPSOGA model.

gathered from VM workload statistics. The CNN-LSTM module extracts the relevant components to measure the CPU usage on each cloud server at different time intervals by using connected CNN. The LSTM model keeps the temporal data, which helps reduce information loss and predict the upcoming load. The CNN layer automatically extracts the pattern information. The order of features is learned once again at the LSTM level. The proposed model continuously tunes Hyperparameters according to the results from learning CNN and LSTM[47].

This module is responsible for finding overloaded and underloaded machines. Before introducing the novel paradigm, we evaluate the conventional VM integration architecture design. The VM integration architecture proposal involves a data center containing servers that use hybrid computing, consisting of several hosts operating different programs across multiple VMs within the information center. Each physical and virtual machine has variables, including CPU processing power, memory disc storage, and network bandwidth.

The functions Calculate_CPU_Utlization (), Call Calculate_RAM_Utlization (), Call Calculate_BW_Utlization (), and Call Calculate_Storage_Utlization () help to determine the current status of the ith machine to predict the overloaded machines. The distinctive aspects of such resource calculations are standardized individuals through a zero to one frequency. High utilization is indicated by a value nearest to 1, while low utilization is characterized by a value closest to 0. It removes the uncertainty in calculating different threshold levels in previous approaches[48].

Equations (10–14) present the formulas[49] for a variety of operations over CPU, memory and BW utilization using Calculate_CPU_Utlization (), Call Calculate_RAM_Utlization (); Call Calculate_BW_Utlization (), Call Calculate_Storage_Utlization ().

$$\text{Calculate}_{\text{CPU}_{\text{Utilisation}}} = \frac{\sum Currently\_Running\_Task}{Total\_capacity} * 100 \tag{10}$$

$$\text{Calculate}_{\text{RAM}_{\text{Utilisation}}} = \frac{\sum \text{RAM}\_\text{Current}\_\text{Utlization}}{\text{Total}\_\text{RAM}} * 100 \tag{11}$$

$$\text{Calculate}_{\text{BW}_{\text{Utilisation}}} = \frac{\sum \text{BW}\_\text{Current}\_\text{Utilisation}}{\text{Total}\_\text{RAM}} * 100 \tag{12}$$

$$\text{Calculate}_{\text{Storage}_{\text{Utilization}}} = \frac{\sum \text{Storage}\_\text{Current}\_\text{Utilization}}{\text{Total}\_\text{Storage}\_\text{Capacity}} * 100 \tag{13}$$

$$\text{Resource}_{\text{Utilization}_{\text{score}}} = \sum_{k=0}^{n} \text{Calculate}_{\text{CPU}_{\text{Utilization}}} + \text{Calculate}_{\text{RAM}_{\text{Utilization}}} + \text{Calculate}_{\text{BW}_{\text{Utilization}}} + \text{Calculate}_{\text{Storage}_{\text{Utilization}}} \tag{14}$$

*Overloaded and underloaded machines predication*

Existing overloaded server recognition approaches are unreliable because they concentrate primarily on standard characteristics, including processor and memory usage. We introduced the various resource-conscious congested host identification approaches, which utilize a wide range of computing resources to determine whether a server/VM is overloaded. It also determines Memory usage, RAM and storage utilization, network traffic and bandwidth consumption, and storage capacity.

It is the primary instance when arrays of computing resources are used as parameters in an integrated form to forecast the overloaded machines in the cloud environment. Using different resources improves the efficiency and reliability of the VM integration architecture. All accessible cloud machines and servers are classified into two classes: loaded and overloaded. Any devices that are currently overloaded remain passive or overwhelmed. The approach we suggest is a fusion of the CNN and LSTM methods, where we utilized an appropriate weight control method that normalized the hosts' capabilities towards a usual spectrum of zero to one. It enhances the system's functioning and dependability and offers an adaptive overloading recognition approach.

We proposed an additional resource awareness underloaded server/Machine (VM and PM) recognition approach to boost VM placement effectiveness and decrease migration frequency without jeopardizing SLA compliance. The proposed underloaded recognition mechanism separates each of the three categories of machines. A resource utilization score helps to decide whether the host/machine is overloaded or under load; we are considering four classes: Idle load (IL), Overloaded (OL), Underloaded (UL) and Free Host (FH). It exhibits a relatively simple and efficient strategy that eliminates the tedious and complicated procedure of finding numerous threshold levels. The proposed method achieves flexibility and dynamic processes by normalizing every measurement. Algorithm 3 presents the working of CNN-LSTM fusion with PSO-GA for overloaded and underloaded machine prediction.

```
Input: Array of computing resources, i.e., CPU processing power, Storage, RAM, processor capacity, bandwidth, the array of jobs (in ready queue, waiting for queue and currently executing).
Output:  Virtual machine in different categories.
  i.e., Idle load (IL), Overloaded (OL), Underloaded (UL) and Free Host (FH).
Step 1: Calculate the current utilization of cloud resources.
1.1 Let Hn: number of hosts
1.2 For i=1 to Hn (repeat steps 1.3 to 1.6)
1.3 Call Calculate_CPU_Utlization ();
1.4 Call Calculate_RAM_Utlization ();
1.5 Call Calculate_BW_Utlization ();
1.6 Call Calculate_Storage_Utlization ();
Step 2: Calculate the current task status
2.1 Let Un be the number of users
2.2 For i=1 to Un (repeat steps 2.3 to 2.4)
2.3 Call Calculate_user_request()
2.4 Call Sort_user_request()
Step 3: Assign VMs to the user task
3.1 For i=1 to Un
3.2 For j=1 to Hn
// map task and user request
3.3 call Map_macine()
Step 4: Calculate the integration score for all computing resources
4.1     for I =1 to Hn
4.2 Call   Cal_Integration()
4.3 If Machine_Capacity<= Assign_work
4.4 Mark the machine as "Overloaded."
4.5 Else Mark "Underloaded."
4.5 END
```

**Algorithm 3** To predict the overloaded machine in the cloud environment

*VM selection*

When overburdened servers are noticed, the approach attempts to determine the servers and VMs that must be moved from a particular host to a different one via predefined VM deployment strategies. An additional virtual machine has been chosen if the server is still overburdened. The proposed model utilizes a PSOGA-based method for VM selection, which operates a Lowest_Migration_Time () to select the appropriate machine. Algorithm 4 defines the VM selection process.

Input: VMs with different classes, i.e., Idle load (IL), Overloaded (OL), Underloaded (UL) and Free host (FH).
Output: VM with Underloaded (UL) and Free Host (FH)

Step 1: Calculate the number of UL and OL VMs and Mark them
1.1 For i=1 to n
1.2 If (VMs = = Overloaded)
1.3 Overloaded_count++
1.4 Else
1.5 Underloaded_count++
Step 2: For each overloaded machine, migrate the jobs towards an underloaded machine
2.1 For i=1 to Overloaded_count
2.2 call Lowest_Migration_Time ()
Step 3: Assign an underloaded_VM from Overloaded_VM
3.1 Execute the test underloaded_VM
3.2 End For
3.3 End

**Algorithm 4** To select the correct VM machine

## Performance measuring parameters

The following parameters are used for performance comparison between the proposed and existing methods. Each parameter is calculated separately for both methods (proposed and existing). Performance metrics for the cloud scheduling algorithms are based on the following factors-.

- Average waiting time: It is defined as how long each process has to wait before it gets its time slice.
- Average response time: It is the time taken from when a process is submitted until the first response is produced[15]. Average response times for each algorithm have decreased by increasing the number of CPUs.
- Makespan: It can be defined as the overall task completion time. We denote the completion time of task $T_i$ on $VM_j$ as $CT_{ij}$.
- Energy Consumption: The sum of energy consumed by PMs. A linear cubic energy utilization approach determines PMs' energy utilization.

$$E_c = \{E_c^{Idle} + \left[E_c^{max} - E_c^{idle}\right] * U_c^3 \tag{15}$$

where $E_c$: Energy consumption, $E_c^{Idle}$: Idle state energy, Uc: CPU utilization.

- Precision: It can be calculated by Eq. (16).

$$\text{Precision} = \text{True positive}/\left(\text{True positive} + \text{False positives}\right) \tag{16}$$

- Recall: It can be calculated by Eq. (7).

$$\text{Recall} = \frac{\text{True positives}}{\text{True positive} + \text{False negative}} \tag{17}$$

F-Measure: It can be calculated by Eq. (18).

$$\text{FMeasure} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{18}$$

Accuracy: It can be calculated by Eq. (19).

$$\text{Accuracy} = \frac{\left(\text{True Positive} + \text{True Negative}\right)}{\left(\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}\right)} \tag{19}$$

## Experimental results and discussion

This section covers experimental details and results and discussion. An experimental analysis was performed in two different phases. The details are as follows.

## Phase 1 experimental analysis

The first phase uses the PSO-GA method to perform efficient load balancing. An experimental analysis was performed on a cloud-sim simulator. The existing PSO, GA and PSO-GA (proposed Hybrid) were implemented. With 10–100 VMs and 50–1000 jobs on the cloud-sim simulator, the experiments were carried out through 15 data centers. The task has between 1000 MI (Million Instructions) and 20,000 MI. Table 2: PSO-GA parameters and Table 3 presents the cloud simulator's feature configurations/parameters overview.

Figure 4 presents the waiting time for proposed and existing methods. Waiting time is calculated for various virtual machines from 10 to 100 with different capacities for all three methods. That proposed method shows better waiting time results than existing PSO and GA methods.

Figure 5 presents the makespan time for the proposed and existing methods. Makespan time is calculated for various virtual machines from 10 to 100 with different capacities for all three methods. That proposed method shows better makespan time results than existing PSO and GA methods.

Figure 6 presents the number of task migrations for proposed and existing methods. Fewer task migrations show a better performance. The task migrations in the proposed model are minimal due to extensive dynamic workload prediction and GA fusion with PSO; it helps identify the most appropriate VM for each job.

Table 4 presents the running time of algorithms using existing GA, PSO, and the proposed hybrid PSOGA method. The analysis was performed using the number of tasks from 50 to 1500. The proposed model achieved better running time over existing GA and PSO methods.

Table 5 presents Energy consumption results (KWh) based on the number of tasks for 100 VMs, and Fig. 7 illustrates the graphical comparison of energy consumption results PSO, GA, and Proposed method based on VMs. The proposed model consumes less energy than existing methods.

## Phase two

The second phase utilizes the online Kaggle datasets "Google cluster workload traces 2019"[47]. The trace mainly contains the complete details for each task, i.e., obedience, schedule preference, and resource information consumption for the assignments executed within these clusters. We compare the DPSOGA proposed model (CNN-LSTM with PSO-GA) and the existing CNN LSTM model.

Each operation within the data set is comprised of numerous continuous assignments which are executed on different systems. The dataset includes CPU and memory utilization, disc usage, etc. Prior studies[1,2] demonstrated that less than 3% of operations need an extended period. A lengthy job containing ID "Job-16-17658948" and 62,071 procedures previously utilized in examinations had been selected to assess proposed and existing models.

| Parameter name | Value |
|---|---|
| Population used | 0–100 |
| Cross over type | Single point standard |
| Rate for mutation | 0.06 |
| Iterations | 0–100 |
| Executions count | 0–500 |
| acceleration coefficients one | 1.0 |
| acceleration coefficients two | 1.2 |
| fitness function | 0.1–0.4 |

**Table 2.** PSO-GA parameters.

| S no | Entity | Parameter name | Value |
|---|---|---|---|
| 1 | Cloud lets (Tasks) | No of tasks | 100–1500 |
| | | Length of task | 500–2500 MI |
| 2 | VMs (virtual machines) | No of VMs | 10–100 |
| | | MIPS | 500–2000 |
| | | RAM (VM memory) | 250–3000 |
| | | Bandwidth | 250–1500 |
| | | Cloud let scheduling method | Shared time and shared space method |
| | | No of PEs requirements | 10–50 |
| 3 | Data center | No data center | 25 |
| | | VMS scheduler | Shared time and shared space method |
| | | No of hosts | 20 |

**Table 3.** Cloud sim parameters.

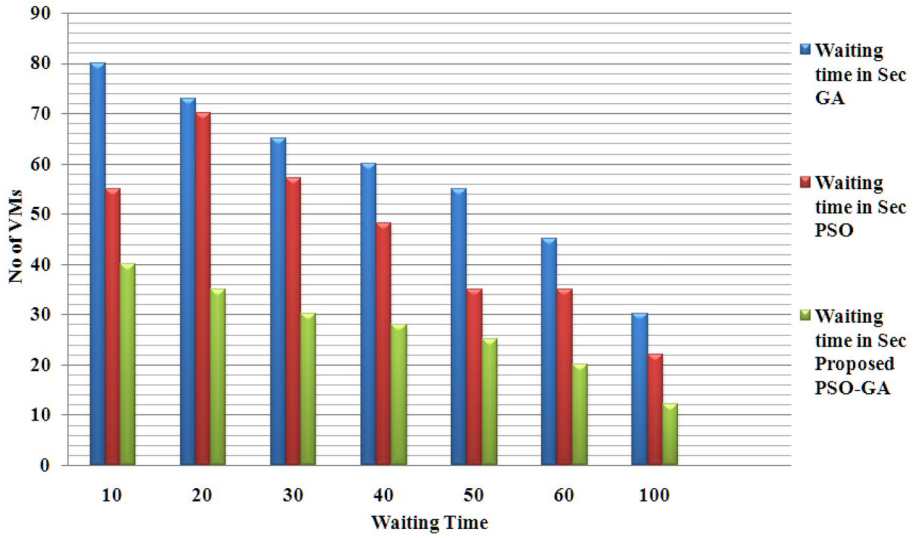


**Figure 4.** Waiting time for proposed and existing methods.

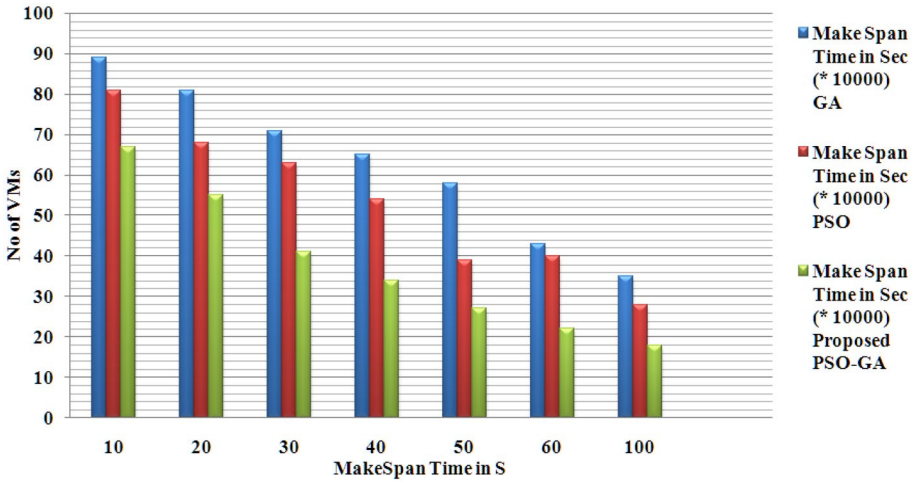


**Figure 5.** Makespan time for proposed and existing methods.

This analysis includes multi-variate analysis that considers storage, processing power and memory and uni-variate analysis that only considers limited parameters. We examined the outcomes of the proposed hybrid model (CNN-LSTM with PSO-GA) to the results of the (CNN-LSTM without PSO-GA). An "exponential linear unit method" (ELUM) is used as an activation parameter in each analyzed model's input and output layers. The outcomes of the proposed hybrid model's assessment compared to alternative approaches concerning "mean absolute error" (MAE) are displayed in Table 6.

The outcomes are calculated using a 'sliding window' with dimension 5. The results prove that the proposed hybrid model (CNN-LSTM with PSO-GA) produces reduced products under uni-variate and multi-variate input scenarios. Figures 8, 9, 10 and 11 present the various outcomes of the analysis of the existing proposed model on multi-variate and uni-variate feature sets.

## Results and discussion

To overcome cloud load balancing and high provision of computing resources issues, this research presented a "DPSO-GA", a Hybrid model for dynamic workload balancing in cloud computing. The complete analysis is divided into two phases to accomplish the research objective. The first phase is based on PSOGA, and the second phase utilizes CNN-LSTM with PSO-GA. The scheduling of tasks in cloud-based systems was addressed in the first phase by the PSOGA algorithm, which was conceived and developed using the Cloud-sim simulation. The proposed method's efficiency was compared with well-known algorithms like GA and PSO.

Figures 4, 5 and 6 present the results of the phase 1 simulation. The PSOGA method's architecture enables task execution across VMs via an equitable workload distribution among fast and slow VMs, avoiding overloading any VM over the others. As opposed to consuming the fast VMs and delaying the job performance overall,

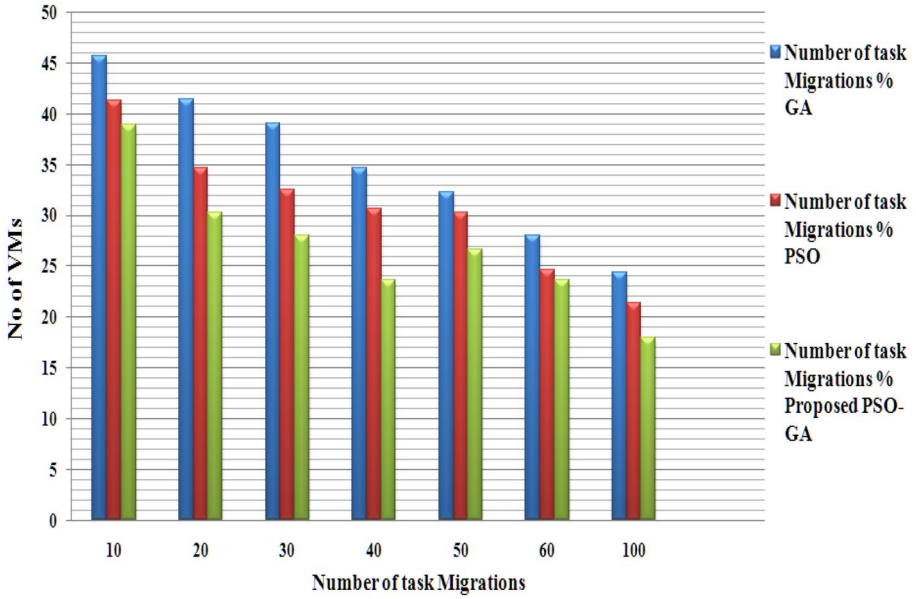


**Figure 6.** No task migrations.

| Technique | Number of tasks | | | | |
|---|---|---|---|---|---|
| | 50 | 100 | 500 | 1000 | 1500 |
| GA | 0.8142 | 1.879 | 17.895 | 20.741 | 22.452 |
| PSO | 0.764 | 1.457 | 16.451 | 18.778 | 19.997 |
| PSOGA | 0.712 | 1.231 | 15.442 | 17.481 | 18.651 |

**Table 4.** Running time (in s) analysis results.

| Technique | Number of tasks | | | | |
|---|---|---|---|---|---|
| | 50 | 100 | 500 | 1000 | 1500 |
| GA | 223.84 | 279.36 | 398.96 | 647.32 | 889.95 |
| PSO | 209.31 | 245.63 | 378.24 | 631.78 | 881.23 |
| PSOGA | 201.77 | 225.91 | 351.40 | 611.78 | 809.91 |

**Table 5.** Energy consumption REsults (KWh) (based on the number of tasks for 100 VMs).

this strategy decreases the Makespan by using the slow VMs relatively. Table 2 presents the PSO-GA parameter setting, and Table 3 shows Cloud Sim parameters. The proposed method (PSO-GA) achieved a better waiting time of 10.2 s for 10 VMs, which is better than the existing PSO and GA. A Makespan time presents the utilization of VMs, and less Makespan time shows a better performance. The proposed PSO-GA achieved the lowest makespan time for 100 VMs and less migration time, delivering better performance than PSO and GA. Similar to Table 4, the number of tasks running time was calculated. The proposed method achieved 0.712 running times (in a s) for 50 studies and 18.651 (s) for 1500 tasks, which is better than the existing method. Infusion of PSO-GA enhances the overall performance of the model.

In experimental two, the proposed model performs better than the existing hybrid model (CNN-LSTM without PSO-GA). PSO-GA model was used to train the CNN-LSTM model. Table 6 presents an MAE analysis for experiment 2 for the proposed and existing method. In the proposed method, a PSO-GA method was used to train the CNN-LSTM model, which helps to enhance the model's performance. For Multi-variate analysis proposed model achieved MAE results Storage: 0.18, Processing power:0.29 and Memory:0.024, and similar to Uni-variate research proposed model acquired MAE results Storage: 0.12, Processing power:0.21 and Memory: 0.017, which is far better than the existing model which achieved For Multi-variate analysis proposed model achieved MAE results Storage: 0.25, Processing power: 0.37 and Memory:0.036, and similar for Uni-variate analysis proposed model earned MAE results Storage: 0.17, Processing power:0.25 and Memory:0.029. Figures 8,

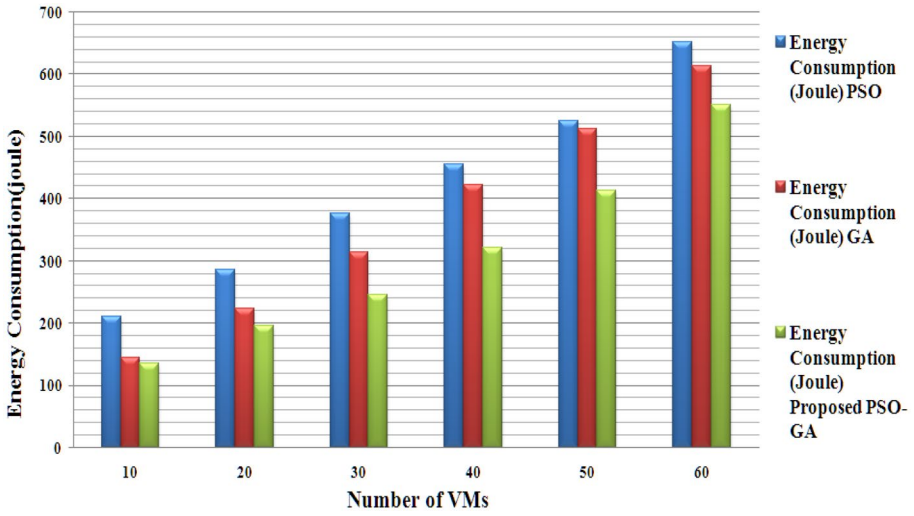


**Figure 7.** Energy consumption based on VMs.

| Model | Input category | Parameters used | | |
|---|---|---|---|---|
| | | Storage | Processing power | Memory |
| Proposed hybrid model | Multi-variate analysis | 0.18 | 0.29 | 0.024 |
| | Uni-variate analysis | 0.12 | 0.21 | 0.017 |
| Existing model | Multi-variate analysis | 0.25 | 0.37 | 0.036 |
| | Uni-variate analysis | 0.17 | 0.25 | 0.029 |

**Table 6.** Simulation results for MAE (existing vs. proposed).

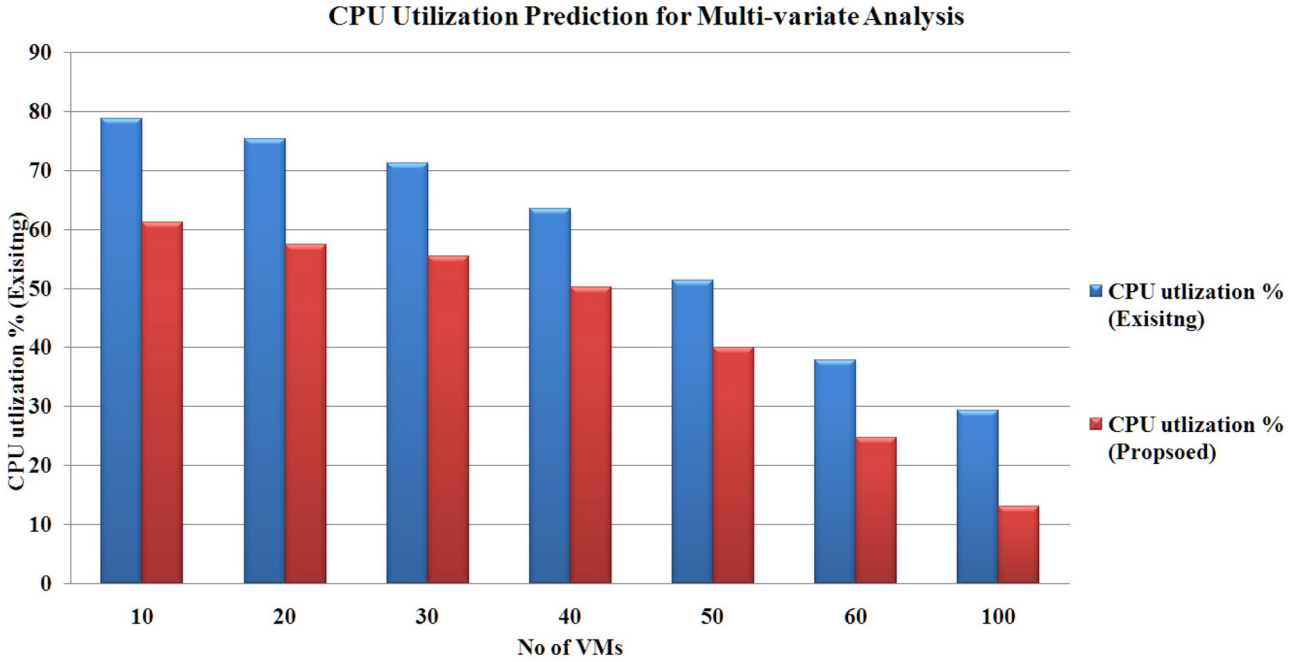


**Figure 8.** CPU utilization % results for multi-variate analysis.

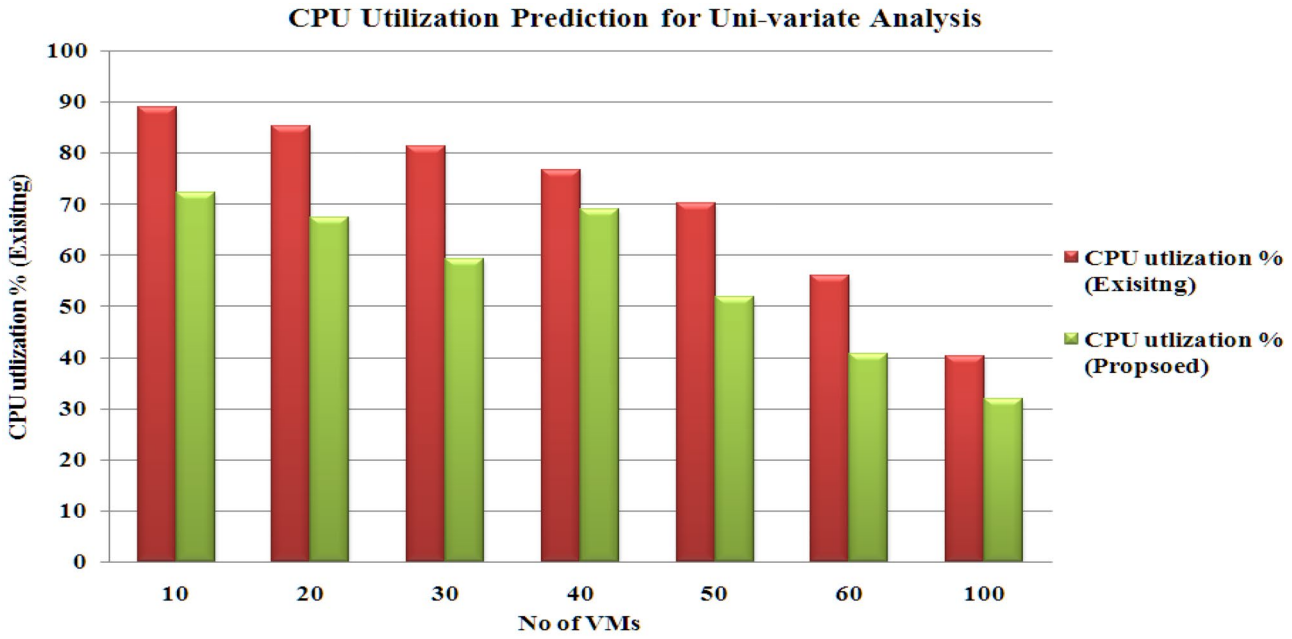


**Figure 9.** CPU utilization % results for uni-variate analysis.

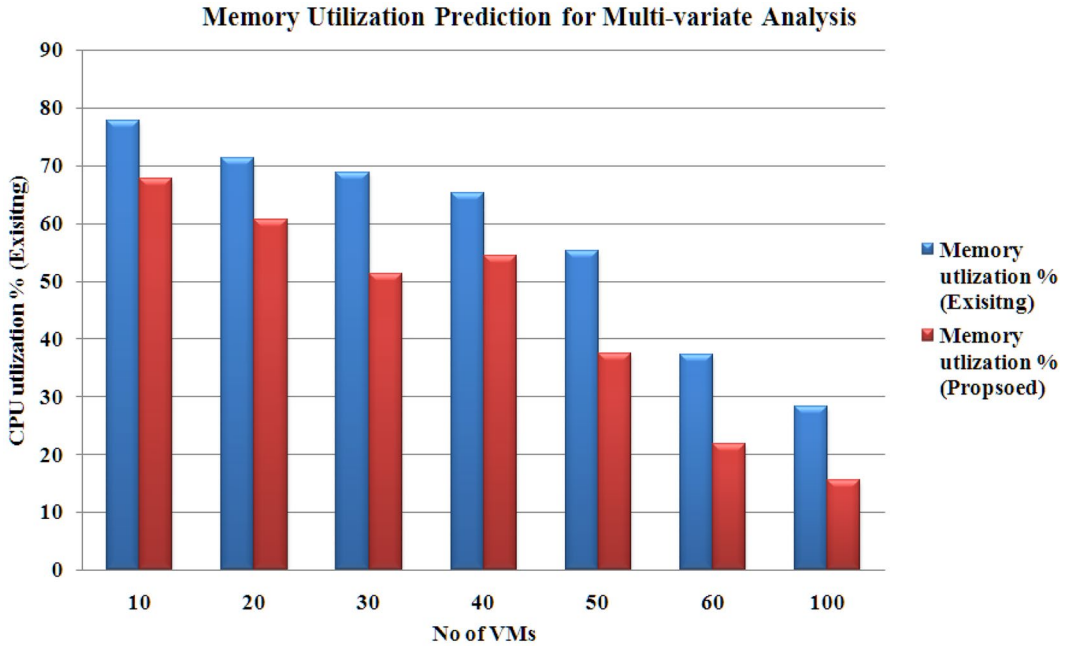


**Figure 10.** Memory utilization % results for multi-variate analysis.

9, 10 and 11 shows the CPU, memory, and storage utilization % results for Multi-variate and Unvaried analysis. The proposed method performs outstandingly in displaying less resource utilization and task migration.

## Time complexity analysis of proposed method

The two main parts of the proposed technique in this paper are the particle and velocity initialization and the updates to the locations and velocities of the particles, as well as the assessment of fitness solutions for PSO and GA. Finding the total time required to execute the method is the first step in calculating its time complexity.

Assume s is the size of the population, v is the size of the virtual machines, and c is the number of tasks associated with the submitted requests. During mass initialization, all of the masses in the population are given random placements and velocities. The fitness of the present mass location is determined at startup. Performing mass initialization has a temporal complexity of $[O(v \times c)]$. Thus, initializing the whole population has a temporal complexity of $O(s \times v \times c)$. The iteration loop begins by updating the global variables. The number of

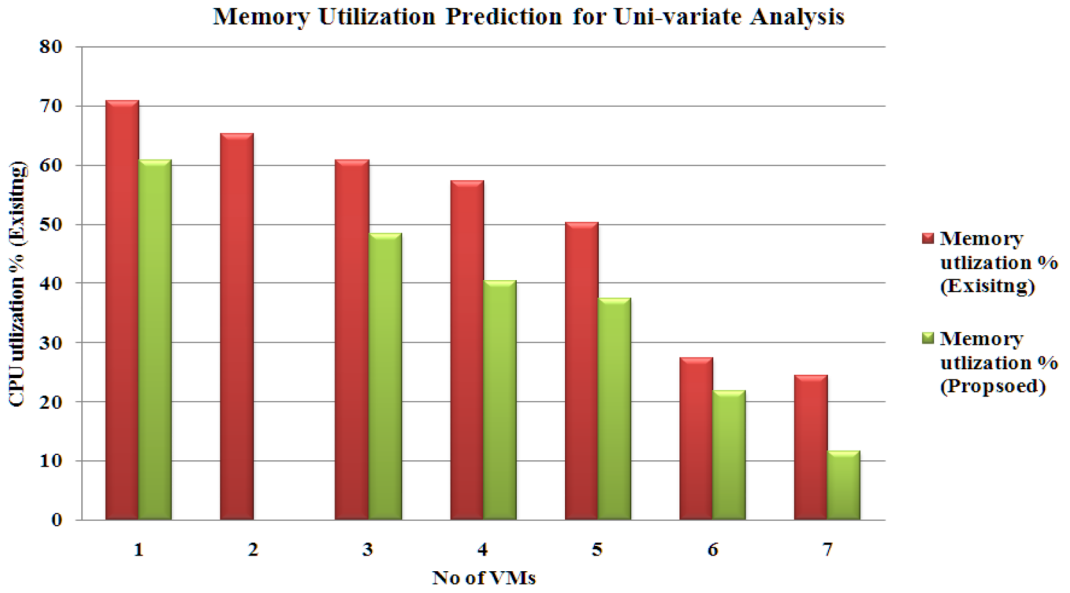


**Figure 11.** Memory utilization % results for uni-variate analysis.

steps required to complete that activity is $[O(s_3 + (s \times c))]$. Iteratively gathering the highest and lowest fitnesses from the whole swarm is necessary within update Global Variables, which is why the process takes so long. Such operations have an $[O(s)]$ time complexity. We also determine the total forces acting on each mass and their acceleration inside this procedure. It takes $[O(s3 + (s \times c))]$ seconds to complete those activities.

Iteratively updating global variables also has this temporal complexity. Secondly, for every mass in the population, a loop is iterated. We are updating the location and velocity of each group. Both of these updates have a temporal complexity of $[O(v \times c)]$. Revising the mass's fitness is the next stage. It takes $[O(v)]$ steps to complete the operation. The particle loop has an overall temporal complexity of $[O(s \times v \times c)]$. Therefore, the temporal complexity for the iterations loop is $[O((s \times MAX\_ITERATION) \times (s2 + (v \times c)))]$. Last but not least, switch back the mapping from cloudlet to virtual machine. This action has a temporal complexity of $O(v \times c)$. The total time complexity, after adding up all the steps' time complexity, is $[O(s \times MAX\_ITERATION \times (s2 + (v \times c)))]$.

## Conclusion and future work

In cloud computing, load balancing plays an essential role in the performance improvement of the entire system. Cloud computing technology offers various opportunities and services for using IT infrastructure as a utility with many possibilities, like scaling down and scaling up, depending upon the organization's needs. However, like most rising technologies, cloud computing also has issues, i.e., high provision of computing resources, load balancing and energy consumption, that must be resolved. To overcome these issues, this research presents a "DPSO-GA", a Hybrid model for dynamic workload balancing in cloud computing. The proposed model works in two phases. The first phase utilizes a hybrid PSO-GA approach to address the prediction challenge by combining the benefits of the two methods. A PSO-GA method also helps to fine-tune the Hyperparameters. In the second phase, CNN-LSTM is utilized. Before using the CNN-LSTM Approach to forecast the consumption of resources, a hybrid approach, PSO-GA, is used for training it.

The simulation results of the first phase include waiting time, task migration time, response time and task running time. The proposed PSOGA fusion performs better than the GA and PSO methods. In the second phase, comprehensive simulations are carried out utilizing the Google cluster traces benchmarks dataset to verify the efficiency of the proposed DPSO-GA technique in enhancing the distribution of resources and load balancing for the cloud. This research can include multiple data centers in a diverse setting. Additionally, the job can be improved by applying dynamic workflow, which gives clients more flexibility to modify the attributes of workflow activities as they are being performed. Numerous parameters make up the scheduling issue, yet some conflict. While enhancing the ideal variables, the developed methods must consider the repercussions of other factors. Furthermore, factors such as privacy and security must be addressed. Applying the modelled algorithm in real-world settings can also present difficulties, including costs associated with administration, energy used for purposes other than computation, hardware problems, and data transfers.

## Data availability

The datasets used in the current research are available from the corresponding author upon individual request.

Received: 23 September 2023; Accepted: 5 January 2024
Published online: 16 January 2024

## References


1. Sumathi, M., Vijayaraj, N., Raja, S. P. & Rajkamal, M. HHO-ACO hybridized load balancing technique in cloud computing. *Int. J. Inf. Technol.* **15**, 1–9 (2023).
2. Assudani, P. J. & Balakrishnan, P. An efficient approach for load balancing of VMs in cloud environment. *Appl. Nanosci.* **13**(2), 1313–1326 (2023).
3. Li, W. *et al.* A hybrid GA-PSO strategy for computing task offloading towards MES scenarios. *PeerJ Comput. Sci.* **9**, e1273 (2023).
4. Ullah, A., Abbasi, I. A., Rehman, M. Z., Alam, T. & Aznaoui, H. *Modified Convolutional Neural Networks and Long Short-Term Memory for Host Utilization Prediction in Cloud Data Center* (2023).
5. Ghosh, T. K., Dhal, K. G. & Das, S. Cloud task scheduling using modified penguins search optimisation algorithm. *Int. J. Next-Gener. Comput.* **14**, 2 (2023).
6. Dalal, S. *et al.* Extremely boosted neural network for more accurate multi-stage Cyber attack prediction in the cloud computing environment. *J. Cloud Comput.* **12**(1), 1–22 (2023).
7. Mishra, K. & Majhi, S. K. A novel improved hybrid optimization algorithm for efficient dynamic medical data scheduling in cloud-based systems for biomedical applications. *Multim. Tools Appl.* **82**, 1–35 (2023).
8. Dang-Quang, N.-M. & Yoo, M. An efficient multi-variate autoscaling framework using Bi-lstm for cloud computing. *Appl. Sci.* **12**(7), 3523 (2022).
9. Patel, E. & Kushwaha, D. S. A hybrid CNN-LSTM model for predicting server load in cloud computing. *J. Supercomput.* **78**(8), 1–30 (2022).
10. Tabrizchi, H., Razmara, J. & Mosavi, A. Thermal prediction for energy management of clouds using a hybrid model based on CNN and stacking multi-layer bi-directional LSTM. *Energy Rep.* **9**, 2253–2268 (2023).
11. Gan, Z., Chen, P., Yu, C., Chen, J. & Feng, K. Workload prediction based on GRU-CNN in cloud environment. In *2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*, 472–476 (IEEE, 2022).
12. Mukherjee, D., Ghosh, S., Pal, S., Aly, A. A. & Le, D.-N. Adaptive scheduling algorithm based task loading in cloud data centers. *IEEE Access* **10**, 49412–49421 (2022).
13. Zeng, J., Ding, D., Kang, K., Xie, H. & Yin, Q. Adaptive DRL-based virtual machine consolidation in energy-efficient cloud data center. *IEEE Trans. Parallel Distrib. Syst.* **33**(11), 2991–3002 (2022).
14. Jamal, M. H. *et al.* Hotspot-aware workload scheduling and server placement for heterogeneous cloud data centers. *Energies* **15**(7), 2541 (2022).
15. Lilhore, U. K., Simaiya, S., Garg, A., Verma, J. & Garg, N. B. An efficient energy-aware load balancing method for cloud computing. In *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, 1–5 (IEEE, 2022).
16. Yan, J. *et al.* Energy-aware systems for real-time job scheduling in cloud data centers: A deep reinforcement learning approach. *Comput. Electr. Eng.* **99**, 107688 (2022).
17. Malik, S., Tahir, M., Sardaraz, M. & Alourani, A. A resource utilization prediction model for cloud data centers using evolutionary algorithms and machine learning techniques. *Appl. Sci.* **12**(4), 2160 (2022).
18. Mohammadzadeh, A., Masdari, M. & Gharehchopogh, F. S. Energy and cost-aware workflow scheduling in cloud computing data centers using a multi-objective optimization algorithm. *J. Netw. Syst. Manag.* **29**, 1–34 (2021).
19. Ouhame, S., Hadi, Y. & Ullah, A. An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model. *Neural Comput. Appl.* **33**, 10043–10055 (2021).
20. Leka, H. L., Fengli, Z., Kenea, A. T., Tegene, A. T., Atandoh, P. & Hundera, N. W. A hybrid cnn-lstm model for virtual machine workload forecasting in cloud data center. In *2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 474–478 (IEEE, 2021).
21. Ajmal, M. S. *et al.* Hybrid ant genetic algorithm for efficient task scheduling in cloud data centers. *Comput. Electr. Eng.* **95**, 107419 (2021).
22. Simaiya, S., Gautam, V., Lilhore, U. K., Garg, A., Ghosh, P., Trivedi, N. K. & Anand, A. EEPSA: Energy efficiency priority scheduling algorithm for cloud computing. In *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, 1064–1069 (IEEE, 2021).
23. Sharma, M. & Garg, R. An artificial neural network based approach for energy efficient task scheduling in cloud data centers. *Sustain. Comput.: Inform. Syst.* **26**, 100373 (2020).
24. Lilhore, U. K., Simaiya, S., Guleria, K. & Prasad, D. An efficient load balancing method by using machine learning-based VM distribution and dynamic resource mapping. *J. Comput. Theor. Nanosci.* **17**(6), 2545–2551 (2020).
25. Sharma, M. & Garg, R. HIGA: Harmony-inspired genetic algorithm for rack-aware energy-efficient task scheduling in cloud data centers. *Eng. Sci. Technol. Int. J.* **23**(1), 211–224 (2020).
26. Lilhore, U. K., Simaiya, S., Maheshwari, S., Manhar, A. & Kumar, S. Cloud performance evaluation: hybrid load balancing model based on modified particle swarm optimization and improved metaheuristic firefly algorithms. *Int. J. Adv. Sci. Technol.* **29**(5), 12315–12331 (2020).
27. Ghasemi, A. & Haghighat, A. T. A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning. *Computing* **102**, 2049–2072 (2020).
28. Boveiri, H. R., Khayami, R., Elhoseny, M. & Gunasekaran, M. An efficient swarm-intelligence approach for task scheduling in cloud-based internet of things applications. *J. Amb. Intell. Hum. Comput.* **10**, 3469–3479 (2019).
29. Pawar, N., Lilhore, U. K. & Agrawal, N. A hybrid ACHBDF load balancing method for optimum resource utilization in cloud computing. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.* **3307**, 367–373 (2017).
30. Chen, Z. *et al.* Pso-ga-based resource allocation strategy for cloud-based software services with workload-time windows. *IEEE Access* **8**, 151500–151510 (2020).
31. Subramoney, D. & Nyirenda, C. N. A comparative evaluation of population-based optimization algorithms for workflow scheduling in cloud-fog environments. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 760–767 (IEEE, 2020).
32. Aggarwal, A., Dimri, P., Agarwal, A. & Bhatt, A. Self adaptive fruit fly algorithm for multiple workflow scheduling in cloud computing environment. *Kybernetes* **50**(6), 1704–1730 (2021).
33. Xie, R., Gu, D., Tang, Q., Huang, T. & Yu, F. R. Workflow scheduling using hybrid PSO-GA algorithm in serverless edge computing for the Internet of Things. In *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, 1–7 (IEEE, 2022).
34. Naik, B. B., Singh, D. & Samaddar, A. B. FHCS: Hybridized optimization for virtual machine migration and task scheduling in cloud data center. *IET Commun.* **14**(12), 1942–1948 (2020).
35. Sardaraz, M. & Tahir, M. A parallel multi-objective genetic algorithm for scheduling scientific workflows in cloud computing. *Int. J. Distrib. Sens. Netw.* **16**(8), 1550147720949142 (2020).
36. Zhou, Z., Shojafar, M., Alazab, M., Abawajy, J. & Li, F. AFED-EF: An energy-efficient VM allocation algorithm for IoT applications in a cloud data center. *IEEE Trans. Green Commun. Netw.* **5**(2), 658–669 (2021).
37. Zhou, Z., Shojafar, M., Li, R. & Tafazolli, R. EVCT: An efficient VM deployment algorithm for a software-defined data center in a connected and autonomous vehicle environment. *IEEE Trans. Green Commun. Netw.* **6**(3), 1532–1542 (2022).
38. Zhou, Z. *et al.* Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms. *Future Gener. Comput. Syst.* **86**, 836–850 (2018).

39. Medara, R. & Singh, R. S. Energy efficient and reliability aware workflow task scheduling in cloud environment. *Wirel. Pers. Commun.* **119**(2), 1301–1320 (2021).
40. Musa, N., Gital, Y. A., Zambuk, F. U., Usman, A. M., Almutairi, M. & Chiroma, H. An enhanced hybrid genetic algorithm and particle swarm optimization based on small position values for tasks scheduling in cloud. In *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*, 1–5 (IEEE, 2020).
41. Zhu, Z., Tan, L., Li, Y. & Ji, C. PHDFS: Optimizing I/O performance of HDFS in deep learning cloud computing platform. *J. Syst. Archit.* **109**, 101810 (2020).
42. Wang, F., Zhang, M., Wang, X., Ma, X. & Liu, J. Deep learning for edge computing applications: A state-of-the-art survey. *IEEE Access* **8**, 58322–58336 (2020).
43. Kumar, P. & Kumar, R. Issues and challenges of load balancing techniques in cloud computing: A survey. *ACM Comput. Surv. (CSUR)* **51**(6), 1–35 (2019).
44. Volkova, V. N., Chemenkaya, L. V., Desyatirikova, E. N., Hajali, M., Khodar, A. & Osama, A. Load balancing in cloud computing. In *2018 IEEE conference of Russian young researchers in electrical and electronic engineering (EIConRus)*, 387–390 (IEEE, 2018).
45. Deepa, T. & Cheelu, D. A comparative study of static and dynamic load balancing algorithms in cloud computing. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, 3375–3378 (IEEE, 2017).
46. Shafiq, D. A., Jhanjhi, N. Z., Abdullah, A. & Alzain, M. A. A load balancing algorithm for the data centers to optimize cloud computing applications. *IEEE Access* **9**, 41731–41744 (2021).
47. Google 2019 Cluster sample. Accessed 17 Jan 2023. https://www.kaggle.com/datasets/derrickmwiti/google-2019-cluster-sample.
48. Singh, A., Juneja, D. & Malhotra, M. Autonomous agent based load balancing algorithm in cloud computing. *Procedia Comput. Sci.* **45**, 832–841 (2015).
49. Malik, N. *et al.* Energy-efficient load balancing algorithm for workflow scheduling in cloud data centers using queuing and thresholds. *Appl. Sci.* **11**(13), 5849 (2021).


## Author contributions

Conceptualization: Ideas and overarching research goals and aims by SS, UKL, YKS and KBV; Methodology: Development and design of methodology by SS and AB; Software: Programming, software development by SS and AB; Investigation and analysis: Conducting a research and investigation process and result analysis by KBV, VVR and UKL, RA; Supervision: Guidance and supervision for the research by UKL.

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to S.S. or A.B.

**Reprints and permissions information** is available at www.nature.com/reprints.

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.