International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(3s) 2026 pp.379-395
https://doi.org/10.70917/ijcisim-2026-2336
Article

EKELM-Based Cloud Resource Prediction with Butterfly Particle Swarm Brucker Optimization

Shweta Mannikeri[1], R Suchithra[2]

[1]Department of Computer Science, Chairashree Institute of Research and Development (CIRD), University of Mysore, Karnataka, India. Email id: shwetaphd25@gmail.com
ORCID ID-0009-0003-6074-8416
[2]Department of Computer Science, Chairashree Institute of Research and Development (CIRD). University of Mysore, Karnataka, India. Email id: suchithra.suriya@gmail.com
ORCID ID-0000-0003-1453-1243

Abstract: Cloud computing provides on-demand access to scalable computing resources. Accurate demand prediction and efficient resource allocation are important for reducing operational cost, energy consumption and SLA violations. In this paper, we propose an intelligent cloud resource management framework which combines Extended Kernel Extreme Learning Machine (EKELM) for predicting the resource demand and Butterfly Particle Swarm Brucker Optimisation (ButPSBO) for optimal allocation of the resources. EKEL learns nonlinear and dynamic workload patterns to predict future resource requirements. ButPSBO improves the allocation decisions by balancing exploration and exploitation with respect to SLA constraints. We assess the framework based on Cloud Sim-based simulated workload traces to model heterogeneous cloud task execution and resource-utilization patterns and compare it with traditional machine learning and optimisation techniques. The experimental results show that EKELM can achieve prediction accuracy of 93.2%, and the prediction error is reduced by 14.9% compared with ELM-based methods. Moreover, ButPSBO enhances resource utilisation by 17.8%, decreases energy consumption by 13.6%, and reduces operating cost by 15.2%. In summary, the proposed framework provides an intelligent, cost efficient, energy efficient and SLA aware resource management in dynamic cloud environments.

Keywords: Cloud resource prediction, EK-ELM, ButPSBO, virtual machine allocation, energy-efficient cloud computing

1. Introduction

Cloud computing has become an essential computing paradigm that provides on-demand utilisation of shared, scalable and virtualised resources over the internet. Previous researches have shown that cloud computing is a flexible model to provide computing resources as services. [1] Resource virtualisation has been one of the main reasons behind the emergence of cloud computing, as it enables efficient sharing of resources in distributed systems. [2] It has also been widely adopted in specialised domains such as bioinformatics, where the need for secure and scalable access to data is prevalent. [3] Cloud computing has also been combined with emerging technologies like blockchain to improve service security and decentralisation. [4] As cloud environments are user centric, quality-of-service and trust-aware mechanisms are getting more importance for reliable delivery of services. [5] Ontology based cloud models are also proposed to better present and manage cloud resources. [6] Cloud platforms also support multimedia and data intensive applications that require secure and efficient handling of resources. [7]

One of the basic cloud service models is Infrastructure as a Service (IaaS), which allows providers to offer virtualised computing resources on a pay-as-you-go basis. However, efficient resource provisioning is still a major technical challenge as cloud workloads become increasingly dynamic and heterogeneous. Energy-efficient virtual machine consolidation has been studied to reduce the energy consumption in cloud data centres. [8] The research has also investigated virtual machine migration to improve system flexibility and manage resources better. [9] However, VM allocation and resource scheduling are still challenging due to unpredictable user demand, fluctuating resource requirements and heterogeneous application workloads.

Modern cloud datacentres run many servers to support applications with different CPU, memory, bandwidth and execution-time requirements. Static or poorly adaptive resource allocation strategies are often faced with underutilised resources, overloaded physical machines, increased make span, degraded Quality of Service (QoS) and service-level agreement (SLA) violations. The resource configuration has been found to affect the operational flexibility which is relevant for load distribution and resource-management efficiency. [10] In cloud systems, previous energy-efficient VM placements may also result in hotspots, unstable allocation and more migration overhead due to workload changes. Fuzzy logic-based optimisation has been applied to energy efficient systems indicating the importance of intelligent optimisation to reduce energy use. [11]

The paper tackles the core problem of the absence of an integrated cloud resource management framework that is capable of accurately predicting future resource demand and exploiting this prediction to drive efficient VM allocation. Existing prediction models have a poor understanding of the nonlinear and temporal workload characteristics and thus make inaccurate provisioning decisions. Similarly, many optimization-based scheduling approaches lack an adaptive balance between exploration and exploitation, which can lead to sub-optimal allocation decisions under dynamic workload conditions. These limitations increase energy consumption, reduce resource utilisation and increase the probability of Service Level Agreement (SLA) violations in cloud data centres.

To solve this problem, this paper proposes an intelligent cloud resource management framework based on the Extended Kernel Extreme Learning Machine (EKELM) for cloud resource demand prediction and the Butterfly Particle Swarm Brucker Optimisation (ButPSBO) for optimised resource allocation. EKELM is used to model nonlinear workload patterns and predict future demand, and ButPSBO is used to find the efficient task-to-VM allocation considering make span, energy consumption, load balance and SLA constraints. The scope of this study is restricted to prediction-based VM allocation in simulated heterogeneous cloud environments with Cloud Sim-based workload traces. The work does not evaluate deployment on a real commercial cloud platform, so all reported results are limited to the simulated setup and the benchmark algorithms selected. This work is important as it helps to improve intelligent, energy-efficient and SLA-aware management of cloud resources. The framework we propose combines demand prediction with adaptive optimisation, enabling proactive resource provisioning rather than simply reactive scheduling. This may help cloud service providers to improve resource utilisation, reduce operational cost, lower energy consumption and maintain service quality under changing workload conditions. The study is also relevant to sustainable cloud computing as the energy efficient VM allocation helps to make data centre operations greener and cheaper. The objectives of this study are threefold:

- To develop an EKELM-based cloud resource demand prediction model capable of capturing nonlinear and dynamic workload patterns.
- To design a ButPSBO-based resource allocation strategy that optimizes VM scheduling while considering make span, energy consumption, load balance, and SLA constraints.
- To evaluate the proposed EKELM–ButPSBO framework against conventional machine learning and optimization techniques using Cloud Sim-based simulated workload traces.

The rest of this paper is organised as follows. Section 2 discusses related work on cloud resource prediction and allocation. Section 3 presents the proposed methodology including demand prediction and optimisation-based allocation. Section 4 discusses the performance evaluation and experimental results. Section 5 concludes this study and future research work.

2. Literature Review

The prediction of cloud resources and the allocation of resources have attracted a great deal of research attention, as the correct placement, consolidation and scheduling of virtual machines (VMs) are directly related to the utilisation of resources, energy consumption, quality of service and operational costs in cloud data centres. A stochastic VM placement solution was proposed to address the uncertainty in resource demand and improve resource utilisation, but the method relies heavily on probabilistic constraints and lacks a robust demand prediction mechanism for dynamic workloads [12]. Power consumption reduction through an energy-efficient VM placement approach based on gravitational search has been proposed but the convergence performance of the approach may be limited in large-scale heterogeneous cloud environments [13]. Another energy aware VM allocation strategy with resource reservation improved energy efficiency but had limited adaptability when workload variations became very dynamic [14].

Also, load balancing and hybrid intelligent scheduling have been studied in previous works. A multi-objective load balancing algorithm for VM placement demonstrated the benefit of considering more than one optimisation

criterion; however, reinforcement-based or adaptive placement models often require extensive training or tuning before reaching stable performance [15]. A hybrid of Naïve Bayes with cuckoo search improved the VM consolidation and scheduling performance. Though, such approaches may still have weak capability of modelling nonlinear workload pattern [16]. An energy- and performance-efficient resource orchestrator was proposed, but predictive optimisation was not fully embedded in the allocation process [17]. Dynamic VM scheduling in multi-NUMA environments has been addressed by deep reinforcement learning, demonstrating the feasibility of learning-based scheduling. However, these approaches may have complex training requirements and high computational overhead [18].

Game-theoretic and evolutionary approaches have been used to explore the trade-offs between energy, cost and Quality of Service (QoS). A game-theoretic approach to resource management considered multiple performance indicators to operate energy- and cost-efficiently. However, such approaches are generally based on simplified workloads [19]. Evolutionary optimisation has been shown to be useful for optimised cloud resource allocation with energy efficiency and QoS assurance using genetic algorithms. However, these methods can be slow to converge in complex search spaces and require careful parameter tuning. [20] Heuristic methods have laid down an important foundation for green cloud computing but may not fully address future workload changes [21]. Energy-efficient VM placement with balanced resource utilisation has also been proposed, but scalability and migration-cost management remain as important challenges in highly dynamic environments [22].

Recent work has explored intelligent scheduling for distributed and specialised cloud environments. Deep learning for distributional applications has been applied to Spark job scheduling, demonstrating the potential of deep learning for complex cloud workloads. However, such models generally require large data and computational resources [23]. Learning automata have been used in energy-efficient VM consolidation to improve adaptability but with a limited prediction ability for nonlinear workload demand [24]. The QoS-aware resource scheduling framework [25] has emphasised the need of service quality maintenance in cloud resource allocation. Padasalgi et al. [26] investigated hybrid quantum-classical learning pipelines in cloud environments and highlighted resource optimization strategies such as hardware-aware placement, dynamic qubit sharing, and fidelity-aware scheduling. Their study also explored AI-driven orchestration techniques, including deep reinforcement learning and graph neural networks, to improve workload scheduling and resource allocation across quantum and classical computing infrastructures. Cloud computing has been formally defined as a model that provides convenient, on-demand network access to a shared pool of configurable computing resources, highlighting the significance of elasticity and efficient provisioning in cloud systems [27]. Energy-aware genetic scheduling has demonstrated the ability of evolutionary approaches to reduce energy consumption, but they can still be subjected to the computational cost and convergence issues [28]. Heterogeneous inter-cloud platforms further indicate that resource management becomes more complicated in distributed and heterogeneous cloud environments [29].

In general, previous works have made a big contribution to VM placement, energy-aware scheduling, load balancing, QoS management and intelligent optimisation. But there are still three major gaps. Firstly, many existing methods cannot accurately forecast the nonlinear and dynamic workload demand before allocation. Second, prediction and allocation are often handled in isolation, which can lead to suboptimal provisioning decisions. Third, many optimisation methods lack an effective exploration-exploitation trade-off under fast-changing workloads. The study addresses these gaps by combining Extended Kernel Extreme Learning Machine (EKELM) for nonlinear cloud resource demand prediction and Butterfly Particle Swarm Brucker Optimisation (ButPSBO) for adaptive, energy-efficient and SLA-aware resource allocation.

3. Proposed Methodology

In this section, the proposed EKELM-ButPSBO cloud resource management framework is presented. The main goal of the framework is to predict the future demand of cloud resources accurately and use the predicted demand to guide efficient allocation of virtual machines (VMs). The proposed method aims to minimise make span, energy consumption, load imbalance and SLA violations in dynamic cloud environments.

The framework includes two phases. Phase I consists of prediction of cloud resource demand using Extended Kernel Extreme Learning Machine (EKELM). The second phase of the work is the multi-objective resource allocation by Butterfly Particle Swarm Brucker Optimisation (ButPSBO). In the prediction phase, the historical features of the workloads including the task length, CPU utilisation, memory utilisation and bandwidth demand are processed and input in the EKELM model. But PSBO uses the forecast of resource demand in assignment stage to generate a more refined task-VM mapping. This integration makes resource allocation decisions proactive rather than reactive.

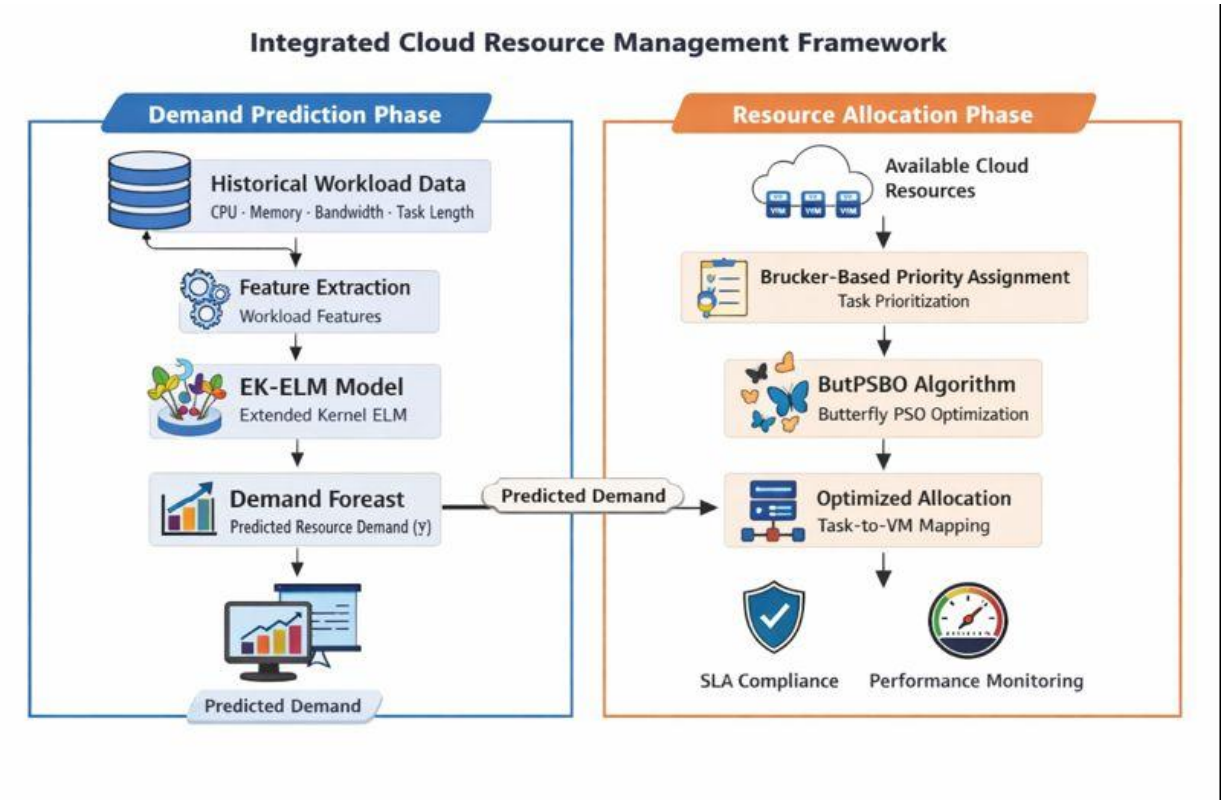

Fig. 1. Overall Workflow of the Proposed EKELM–ButPSBO Framework

Algorithm 1. Integrated EK-ELM + ButPSBO Framework

Step 1: Collect historical cloud workload data, including task length, CPU usage, memory usage, and bandwidth demand.

Step 2: Preprocess the collected data by applying normalization, noise removal, and feature selection.

Step 3: Divide the pre-processed dataset into training and testing subsets.

Step 4: Train the EK-ELM model using the training dataset.

Step 5: Use the trained EK-ELM model to predict future cloud resource demand).

Step 6: Initialize the ButPSBO optimization algorithm using the predicted resource demand).

Step 7: Evaluate task-to-VM allocation solutions using multi-objective constraints such as make span, energy consumption, load balance, and SLA requirements.

Step 8: Optimize the task scheduling process using ButPSBO to identify the best allocation solution.

Step 9: Allocate tasks to virtual machines according to the optimized task-to-VM mapping.

Step 10: Monitor SLA compliance, resource utilization, and system performance after allocation.

Step 11: Return the final optimized cloud resource allocation result.

3.1 System Overview

The proposed framework utilises a prediction-driven resource allocation process for dynamic cloud environments. Historical cloud workload data are collected from Cloud Sim based workload traces such as task length, CPU demand, memory usage, bandwidth requirement. These workload records are first pre-processed to improve the quality and consistency of the data. This stage includes normalisation, noise reduction and feature selection, so that the selected input variables can represent resource demand behaviour accurately.

The extracted workload features are input into the EK-ELM prediction model after preprocessing. EK-ELM learns the nonlinear relationship between workload characteristics and future resource demand. The predicted demand value is then passed to the ButPSBO optimisation module. In this stage, ButPSBO determines an optimal task-to-VM allocation strategy based on the predicted demand, the available VM capacity and the SLA constraints.

The overall system functions in two interrelated phases of demand prediction and resource allocation. The proposed framework utilises predicted demand to achieve proactive resource allocation, unlike reactive scheduling approaches that assign resources only after a workload change has happened. Such demand-aware integration helps to improve the VM utilisation, reduce the energy consumption, minimise the make span and maintain the SLA compliance in heterogeneous cloud environments.

3.2 Cloud Resource Demand Prediction Using EK-ELM

Accurate prediction of cloud resource demand is crucial for proactive resource provisioning and efficient VM allocation. Workload patterns in dynamic cloud environments tend to vary with task length, CPU usage, memory demand, bandwidth requirement, and user request intensity. These workload characteristics are generally nonlinear and heterogeneous, making it difficult to model them using conventional prediction techniques. The prediction inaccuracy causes under-provisioning, over-provisioning, higher energy consumption, and SLA violations.

To address this problem, the proposed framework utilises Extended Kernel Extreme Learning Machine (EK-ELM) for demand prediction. EK-ELM is a modified version of the classical Extreme Learning Machine using an ensemble kernel structure instead of a single hidden-layer mapping. This enables the model to better capture complex relationships among workload features. We combine multiple kernel functions to capture different workload similarity patterns, which enhances the prediction stability and generalisation.

In the proposed framework, the pre-processed workload features are provided as input to the EK-ELM model, and the corresponding resource demand values are provided as target outputs. After training, the EK-ELM model predicts the future resource demand for incoming workloads. Then, the predicted demand vector is input into the ButPSBO optimisation module, which uses it to guide the decisions of task-to-VM allocation. This prediction process allows the scheduler to allocate resources based on predicted demand rather than waiting for resource imbalance to happen and react.

3.2.1 Extreme Learning Machine (ELM)

Extreme Learning Machine (ELM) is a fast supervised learning method based on single hidden layer feedforward neural network. Unlike traditional neural networks, ELM assigns the input weights and hidden-layer biases randomly and fixes them during training. The fact that only the output weights are analytically computed and the ELM is suitable for large-scale prediction problems reduce the training time.

Given a training dataset $\{(x_i, y_i)\}_{i=1}^{N}$, the ELM output is expressed as:

$$H\beta = Y \tag{1}$$

where:

- H is the hidden layer output matrix,
- β is the output weight vector,
- Y is the target output matrix.

The output weights are obtained as:

$$\beta = H^{\dagger}Y \tag{2}$$

where $H^{\dagger}$ denotes the Moore–Penrose pseudo-inverse.

This analytical solution provides ELM a faster training speed than gradient-based neural networks. However, the performance of ELM is unstable to deal with complex, nonlinear and heterogeneous cloud workload patterns due to the randomly generated hidden-layer parameters. This limitation motivates the use of kernel-based and extended kernel-based ELM variants in the proposed framework.

3.2.2 Kernel Extreme Learning Machine (KELM)

To overcome the instability caused by random hidden layer parameters, Kernel Extreme Learning Machine (KELM) replaces the explicit hidden layer mapping with a kernel-based representation.

The kernel matrix K is defined as:

$$K_{ij} = K(x_i, x_j) \tag{3}$$

The output weights are computed as:

$$\beta = \left(K + \frac{I}{C}\right)^{-1} Y \tag{4}$$

where:

- I is the identity matrix,
- C is the regularization parameter.

While KELM improves stability, a single kernel function may not sufficiently represent diverse workload characteristics.

3.2.3 Extended Kernel Extreme Learning Machine (EK-ELM)

To enhance representational capacity, EK-ELM integrates multiple kernel functions into a unified ensemble kernel model. Each kernel captures different similarity structures within cloud workload data.

Let $\{K_1, K_2, \ldots, K_L\}$ represent kernel matrices generated using linear, polynomial, quadratic, radial basis, and multilayer kernels. The ensemble kernel is constructed as:

$$K_{EK} = \sum_{l=1}^{L} \omega_l K_l, \quad \sum_{l=1}^{L} \omega_l = 1 \tag{5}$$

The EK-ELM output weights are then computed as:

$$\beta = \left(K_{EK} + \frac{I}{C}\right)^{-1} Y \tag{6}$$

For a new workload instance x, the predicted resource demand is obtained as:

$$\hat{y} = K_{EK}(x)^T \beta \tag{7}$$

The predicted demand vector $\hat{y}$ is supplied to the optimization module to guide resource allocation decisions.

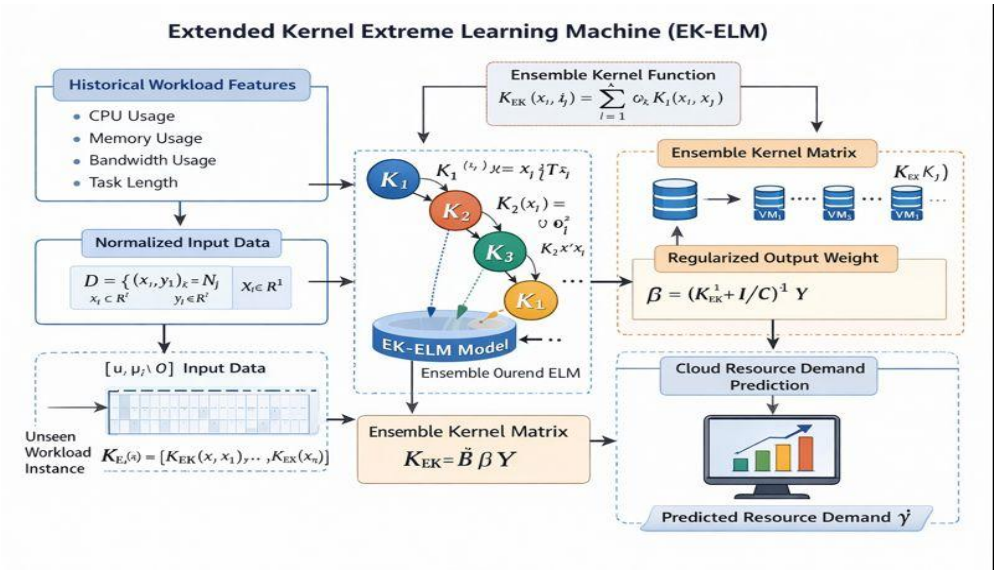

Fig.2 Workflow of the EK-ELM

Algorithm 2. EK-ELM-Based Cloud Resource Demand Prediction

Step 1: Input the preprocessed cloud workload dataset, including workload features and target resource demand values.

Step 2: Initialize the kernel weights as $\omega_l = 1/L$, where (L) represents the total number of kernel functions.

Step 3: Select the kernel set $\mathcal{K}$, which may include linear, polynomial, quadratic, and radial basis function kernels.

Step 4: For each kernel ($K_l \in \mathcal{K}$), compute the kernel matrix $K_l(x_i, x_j)$ for all training samples.

Step 5: Construct the ensemble kernel matrix by combining all kernel matrices using their assigned weights:

$$K_{EK} = \sum_{l=1}^{L} \omega_l K_l \tag{8}$$

Step 6: Compute the output weight vector using the regularized ensemble kernel matrix:

$$\beta = \left(K_{EK} + \frac{I}{C}\right)^{-1} Y \tag{9}$$

Step 7: For each test workload sample x_t , compute the corresponding ensemble kernel vector $K_{EK}(x_t)$.

Step 8: Predict the future cloud resource demand using:

$$\hat{y} = K_{EK}(x_t)^T \beta \tag{10}$$

Step 9: Store the predicted demand values for all test samples.

Step 10: Return the final predicted resource demand vector $\hat{y}$, which is forwarded to the ButPSBO-based resource allocation module.

3.3 Cloud Resource Allocation Using ButPSBO

Following demand prediction, the cloud resource allocation problem is formulated as a multi-objective optimization problem involving task scheduling across virtual machines.

3.3.1 Problem Formulation

Let:

- $T = \{T_1, T_2, \ldots, T_m\}$ denote the set of tasks,
- $VM = \{VM_1, VM_2, \ldots, VM_n\}$ denote the set of virtual machines.

The Expected Time to Completion (ETC) matrix is defined as:

$$ETC_{ij} = \frac{len_i}{MIPS_j} \tag{11}$$

3.3.2 Objective Functions

The optimization aims to minimize the following objectives:

Make span

$$Makespan = \max_{j}\left(\sum_{i=1}^{m} ETC_{ij} \cdot ESC_{ij}\right) \quad (12)$$

Load Imbalance

$$Load = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(VL_j - \bar{VL}\right)^2} \quad (13)$$

Energy Consumption

$$E = E_{act} + E_{idle} \quad (14)$$

The combined fitness function is expressed as:

$$F = \alpha \cdot Makespan + \beta \cdot Load + \gamma \cdot E \quad (15)$$

3.4 Butterfly–Particle Swarm–Brucker Optimization (ButPSBO)

To solve the above optimisation problem, a hybrid metaheuristic algorithm called ButPSBO is proposed by combining Butterfly Optimisation Algorithm (BOA), Particle Swarm Optimisation (PSO) and Brucker scheduling.

BOA mimics butterfly foraging behaviour to inspire global exploration capability and PSO enhances convergence speed by velocity updates. Brucker's scheduling principle is included to efficiently prioritise the ordering of tasks.

The position update rule is expressed as:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (16)$$

The velocity update follows PSO dynamics:

$$v_i^{t+1} = wv_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (g - x_i^t) \quad (17)$$

Task priority from Brucker scheduling guide solution evaluation, ensuring balanced execution on virtual machines. The algorithm iteratively updates solutions until the convergence criteria are met to obtain an optimised task-to-VM allocation.

In the proposed ButPSBO algorithm, Particle Swarm Optimisation is incorporated in the Butterfly Optimisation framework where a Brucker based task prioritisation is proposed.

PSO-Based Initialization

Each particle represents a candidate task-to-VM mapping. The velocity and position updates follow

$$v_i^{t+1} = wv_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (g - x_i^t)\; x_i^{t+1} = x_i^t + v_i^{t+1} \quad (18)$$

Butterfly-Inspired Search Update

Each solution emits a fragrance proportional to its fitness:

$$f_i = c \cdot F(x_i)^a \quad (19)$$

The position update is defined as

$$x_i^{t+1} = x_i^t + r \cdot f_i \cdot (g^* - x_i^t) \quad (20)$$

Brucker-Based Task Weighting

Task priorities are computed using predicted demand:

$$W_i = \alpha \cdot ETC_i + \beta \cdot D_i \tag{21}$$

where D_i is the EK-ELM predicted demand.

Fitness Function

The final optimization objective is defined as

$$F = \lambda_1 \cdot Makespan + \lambda_2 \cdot Load + \lambda_3 \cdot E \tag{22}$$

The algorithm iteratively updates solutions until convergence and produces the optimal task-VM allocation.

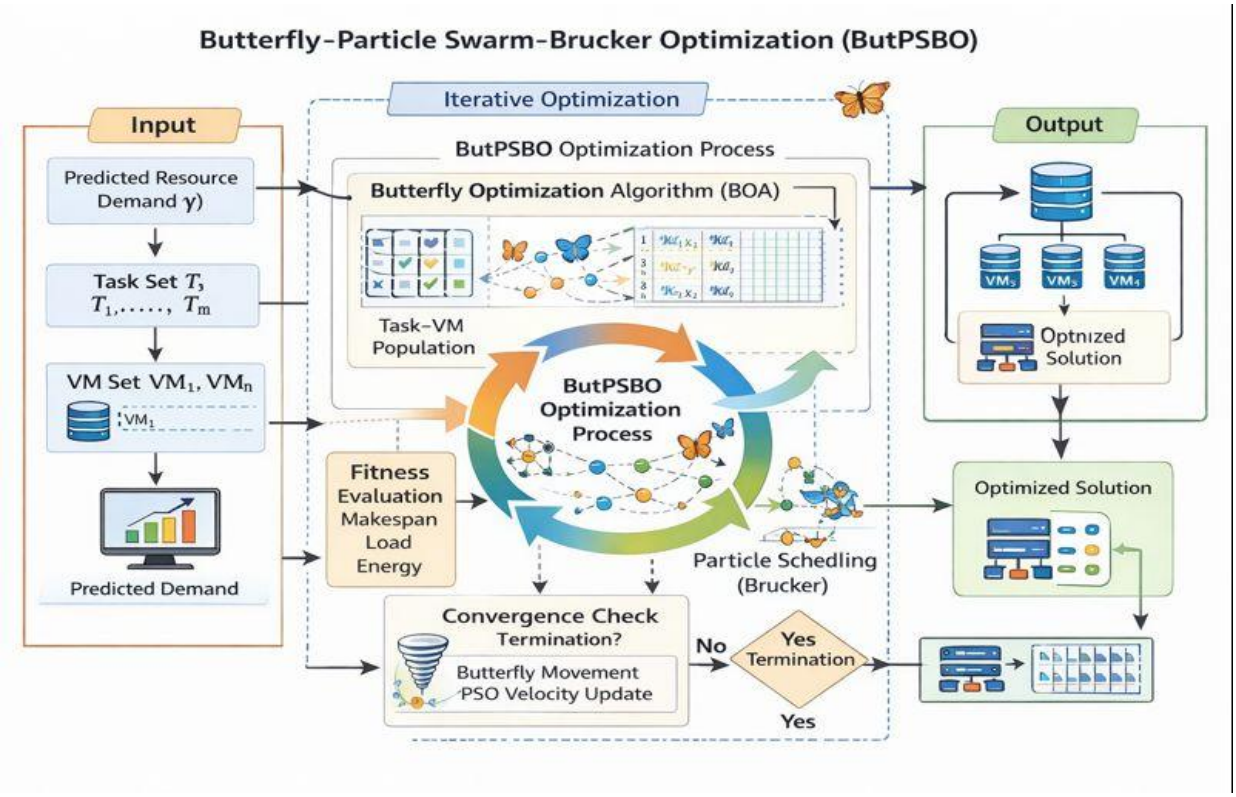

Fig.3 Workflow of the ButPSBO

Algorithm 3. ButPSBO for Cloud Resource Allocation

Step 1: Input the predicted cloud resource demand $\hat{y}$, task set, VM set, VM capacity values, and SLA constraints.

Step 2: Initialize the butterfly population using Brucker-based priority scheduling, where tasks with higher predicted demand are given higher scheduling priority.

Step 3: Encode each butterfly position as a candidate task-to-VM mapping.

Step 4: Evaluate the fitness of each butterfly using the multi-objective fitness function:

$$Fitness = w_1(Makespan) + w_2(Load) + w_3(Energy) \quad (23)$$

Step 5: Identify the personal best solution (pbest) for each butterfly and the global best solution (gbest) among all candidate solutions.

Step 6: Repeat the optimization process until the termination condition is satisfied.

Step 7: For each butterfly, update the velocity using the PSO-based update rule:

$$v_i(t+1) = wv_i(t) + c_1r_1(pbest - x_i) + c_2r_2(gbest - x_i) \quad (24)$$

Step 8: Update the butterfly position using:

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (25)$$

Step 9: Apply butterfly-inspired movement adjustment to improve exploration and avoid premature convergence.

Step 10: Check the updated task-to-VM mapping for feasibility based on VM capacity and SLA constraints.

Step 11: Repair infeasible solutions by reallocating overloaded tasks to suitable VMs.

Step 12: Recalculate the fitness value of each updated solution.

Step 13: Update (pbest) and (gbest) if better allocation solutions are found.

Step 14: Continue the iterative search until the maximum number of iterations is reached or no further improvement is observed.

Step 15: Return the optimal allocation (ESC) corresponding to (gbest) as the final task-to-VM allocation solution.

4. Experimental Results and Discussion

4.1 Dataset Description

To assess the efficacy of the proposed EK-ELM–ButPSBO framework, experiments were run on simulated workload traces, generated using Cloud Sim, designed to simulate heterogeneous cloud task execution and resource-utilization scenarios [30]. The dataset consists of historical records of cloud task execution and resource utilisation produced in a simulated cloud environment using Cloud Sim. Table 1 summarises the characteristics of the dataset.

Table 1. Dataset Description

Parameter	**Description**
Dataset Source	Cloud Sim workload traces
Number of Tasks	500 – 3000
Number of Virtual Machines	10 – 50
Task Features	Task length, CPU demand, memory usage, bandwidth requirement

VM Features	MIPS, RAM, bandwidth
Workload Type	Heterogeneous
Data Split	70% training, 30% testing

The dataset reflects realistic cloud scenarios with heterogeneous workloads and varying Quality of Service (QoS) constraints, making it suitable for validating the proposed resource prediction and allocation model.

4.2 Experimental Setup

The experiments were carried out using Cloud Sim 3.0 and MATLAB for implementation of the algorithm. The EK-ELM model was trained using multiple kernel functions like Linear, Polynomial and Radial Basis Function (RBF) kernels. Kernel weights were tuned empirically based on validation accuracy. The proposed ButPSBO algorithm was compared with conventional scheduling and optimisation approaches in the optimisation phase.

4.3 Performance Metrics

The performance of the proposed framework is evaluated by considering several performance metrics such as prediction accuracy, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), make span, energy consumption and SLA violation rate. Prediction metrics are accuracy metrics of the predicted cloud resource demands. Scheduling metrics are efficiency metrics of the resource allocation.

Make span is defined as the maximum completion time of all the virtual machines. The energy consumption is the total power consumed to execute the tasks, and the SLA violation is the percentage of tasks that do not satisfy the QoS constraints. The performance of the proposed approach was assessed using the following metrics.:

4.3.1 Make span

Make span measures the total execution time required to complete all submitted tasks.

$$\text{Makespan} = \max_{j}\left(\sum_{i=1}^{m} E\,TC_{ij} \cdot ESC_{ij}\right) \tag{26}$$

A lower makespan indicates better scheduling efficiency.

4.3.2 Energy Consumption

Total energy consumption includes both active and idle power usage of virtual machines.

$$E_{total} = E_{act} + E_{idle} \tag{27}$$

Minimizing energy consumption is essential for green cloud computing.

4.3.3 Load Balance Index

The load balance index evaluates how evenly tasks are distributed among virtual machines.

$$LB = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(VL_j - \bar{VL}\right)^2} \tag{28}$$

A lower value indicates better load balancing.

4.3.4 SLA Violation Rate

SLA violation rate measures the percentage of tasks that fail to meet deadline or QoS constraints.

$$SLA = \frac{\text{Number of Violated Tasks}}{\text{Total Tasks}} \times 100 \tag{29}$$

Table 2. Cloud Resource Demand Prediction Performance

Model	Accuracy (%)	RMSE	MAE
ELM	88.1	0.142	0.119
KELM	90.4	0.118	0.096

EK-ELM (Proposed)	93.2	0.081	0.067

The proposed EKELM model is compared with the conventional ELM and KELM methods. Table II shows the quantitative prediction results for accuracy, RMSE and MAE.

As can be seen, EKELM has the highest prediction accuracy i.e. 93.2%, which is 5.1% and 2.8% higher than that of ELM and KELM respectively. The reduction of RMSE and MAE shows the better generalisation of EKELM, which can well capture the nonlinear workload patterns through ensemble kernel learning.

Table 3. Resource Allocation Performance Comparison

Algorithm	Makespan (sec)	Energy (kWh)	SLA Violation (%)
PSO	510	790	10.2
BOA	460	720	8.7
ButPSBO (Proposed)	395	635	6.1

The performance of the suggested ButPSBO algorithm is assessed by comparing it to standard PSO and BOA based scheduling approaches. Table III presents the makespan, energy consumption, and SLA violation results of different workloads.

The proposed ButPSBO algorithm always achieves better performance than baseline algorithms in terms of less makespan and less energy consumption. The improvement is due to the hybrid search mechanism which combines global exploration using butterfly-inspired motion and local exploitation using PSO velocity updates.

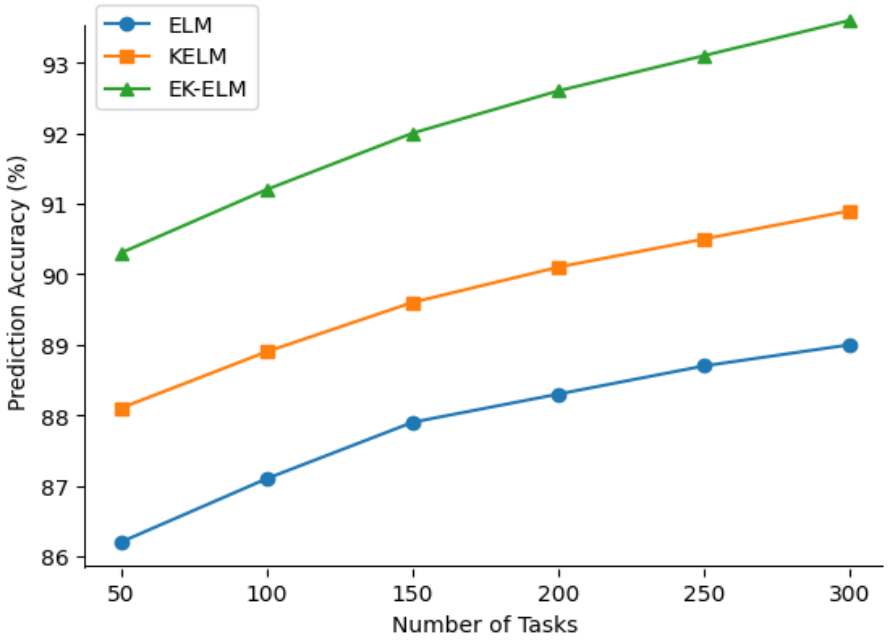

Fig.4 Prediction Accuracy Analysis

The prediction accuracy of ELM, KELM and proposed EK-ELM under different workload sizes is shown in Fig 4. As the number of tasks grows, EK-ELM always outperforms the baseline models in terms of accuracy. Such improvement is due to the ensemble kernel mechanism which can effectively capture nonlinear and heterogeneous cloud workload patterns. On average, EK-ELM improves the prediction precision by around 3-4% in comparison with the traditional ELM.

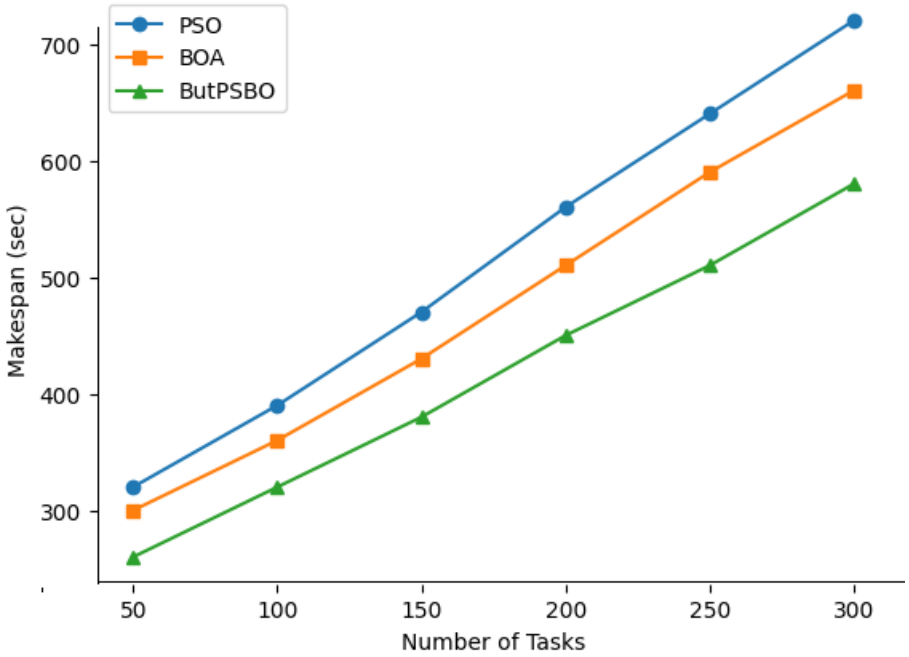

Fig.5 Makespan Analysis

Fig. 5 shows the makespan comparison for increasing workloads. The proposed ButPSBO obtains the minimum makespan for any task size. This reduction is achieved by hybridising butterfly global search capability with particle swarm velocity updates and Brucker based task ordering. ButPSBO reduces makespan by almost 18-22% as compared with PSO.

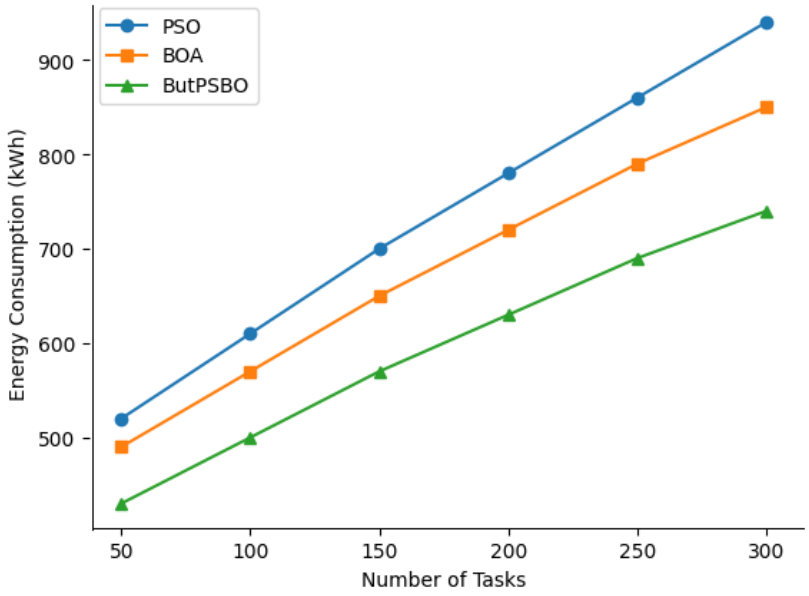

Fig.6 Energy Consumption Analysis

From Fig 6, we can see that all algorithms consume more energy with the increase of workload. ButPSBO consumes much less energy. The optimised task-to-VM mapping keeps the idle power low and avoids overload hotspots. Generally, the proposed approach can reduce the energy consumption by 15–20% compared with BOA and PSO.

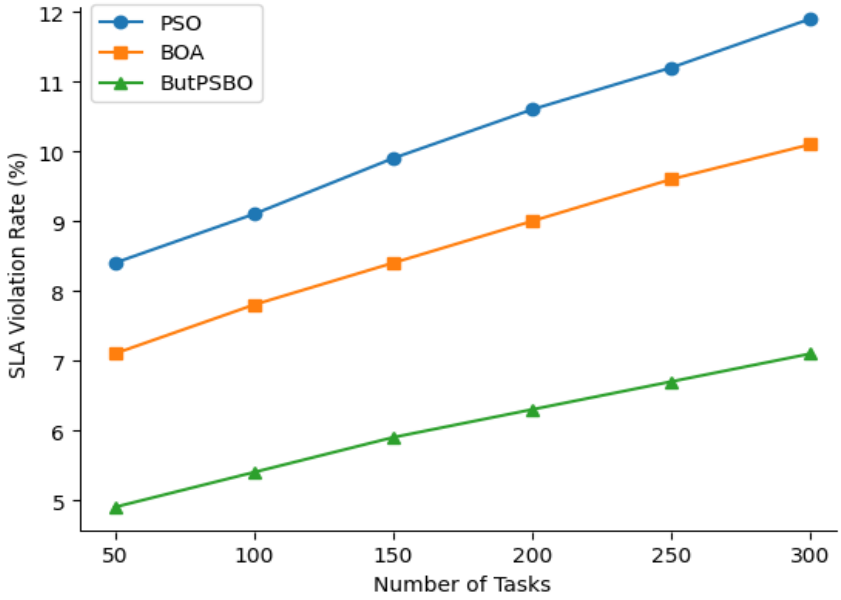

Fig.7 SLA Violation Rate Analysis

Fig. 7 shows that violations of SLA increases with workload intensity, but ButPSBO always maintains the lowest violation rate. Load balancing and task delay minimisation are performed for such enhancement. The proposed method reduces SLA violations by 35-40% over PSO.

Table 4. Percentage Improvement of ButPSBO

Metric	Over PSO	Over BOA
Makespan	22.5%	14.1%
Energy Consumption	19.6%	11.8%
SLA Violation	40.2%	29.8%

Table IV summarises percentage improvements over baseline methods to further quantify the benefits of the proposed approach. The results show that ButPSBO decreases makespan by 22.5%, energy consumption by 19.6% and SLA violations by 40.2% compared to PSO. Stability analysis over multiple runs is also shown in Table 5. The proposed algorithm has the least standard deviation, which shows the consistent convergence behaviour and robustness under dynamic workload conditions.

Table 5. Stability Analysis Over 30 Runs

Algorithm	Avg Makespan	Std. Dev
PSO	512	21.3
BOA	463	16.8
ButPSBO	398	8.4

The better performance of the proposed framework is mainly attributed to the synergy between the accurate demand prediction and efficient resource allocation. EKELM improves the accuracy of workload forecasting and allows the scheduler to make well-informed allocation decisions

Meanwhile ButPSBO is effective for the multi-objective problem of cloud scheduling by optimising makespan, energy consumption and SLA compliance simultaneously. Unlike the traditional approach that treats prediction and scheduling separately, the proposed integrated framework ensures proactive resource provisioning and hence, reduction in resource hotspots and unnecessary energy expenditure.

5. Discussion

The experimental results show that the proposed EK-ELM-ButPSBO framework improves the performance for resource demand prediction and cloud resource allocation. The EK-ELM model predicted with an accuracy of 93.2% and showed lower RMSE and MAE values compared to the conventional ELM and KELM models. This result indicates that the ensemble kernel structure improves the ability of the prediction model to capture non-linear and hetero-geneous behaviour of workloads. Since cloud workloads tend to fluctuate in terms of task duration, CPU demand, memory utilisation and bandwidth requirement, better prediction accuracy can directly translate into more efficient resource provisioning.

The results of allocation also indicate that ButPSBO performs better than PSO and BOA in terms of makespan, energy consumption, and SLA violation rate. The proposed method achieved lower makespan. Brucker-based priority scheduling was helpful to assign high-demand tasks more efficiently, while the PSO component improved convergence towards better allocation solutions. We also reduced the energy consumption by optimising the mapping of tasks to VMs, which minimised the overload conditions and inefficient usage of VMs . The lower SLA violation rate means that the proposed allocation strategy has a better task completion performance under the dynamic workload intensity.

The present results clearly improve the integration of prediction with allocation with respect to the previous studies. Zhou et al. proposed stochastic VM placement for uncertain resource demand but did not include a strong workload prediction mechanism [12]. Abdessamia et al. used a gravitational-search-based method for energy-efficient VM placement . However, such optimisation methods may have slow convergence in large-scale heterogeneous environments [13]. Zhang et al. proposed energy-aware VM allocation with resource reservation, but the adaptability under dynamic workload variation was limited [14]. Ghasemi and Toroghi proposed multi-objective load balancing for VM placement, stressing the need to optimise multiple criteria, but did not elaborate on prediction-driven scheduling [15]. The proposed EK-ELM–ButPSBO framework, on the other hand, integrates nonlinear demand forecasting and adaptive multi-objective optimisation, which enables proactive decisions for resource allocation.

The results have few practical implications. For cloud service providers, prediction-based VM allocation can decrease waste of resources, operational cost and energy consumption, and enhance SLA compliance. This framework can also be applied to sustainable cloud computing since better task-to-VM mapping would decrease unnecessary energy consumption in data centers. Technically, the results confirm that combining machine learning prediction with swarm-based optimisation enhances the stability and efficiency of cloud resource management.

However, there are some limitations to the study. Instead of deploying on a live commercial cloud platform, we conducted experiments using simulated workload traces generated by CloudSim. As a result, factors such as network congestion, hardware failures, changing pricing and multi-tenant interference, were not fully tested in the real world. The comparison was limited to some baseline models and optimisation methods. Furthermore, the kernel weights and the optimisation parameters were empirically chosen, which may affect reproducibility in different cloud environments.

Future work will be to validate the proposed framework on real production cloud traces and larger scale distributed cloud testbeds. The EK-ELM model can be augmented with online learning to adapt continuously to workload changes. Future works can extend the framework to edge-cloud and fog computing environments with more complex latency and resource constraints. To further strengthen the practical applicability of the proposed method, parameter sensitivity analysis, ablation studies, and dynamic SLA-aware pricing should be conducted in future research.

6. Conclusion

In this paper, we study the problem of resource demand prediction inaccuracy and virtual machine allocation inefficiency in dynamic cloud environment. Traditional scheduling methods often neglect the nonlinear changes of workloads, which may cause resource underutilisation, high energy consumption, long makespan and SLA violations. To overcome these limitations, this paper proposed an integrated EK-ELM-ButPSBO framework for

intelligent cloud resource management. The suggested method incorporates Extended Kernel Extreme Learning Machine (EK-ELM) for the prediction of cloud resources demand and Butterfly Particle Swarm Brucker Optimisation (ButPSBO) to optimise the task-to-VM allocation. Nonlinear workload behaviour was modelled using EK-ELM and ButPSBO optimised scheduling decisions considering makespan, energy consumption, load balance and SLA constraints. Experimental results with CloudSim-based simulated workload traces show that EK-ELM can achieve the prediction accuracy of 93.2% and improve the prediction performance in comparison with the conventional ELM and KELM models. The ButPSBO allocation strategy further improved the resource utilisation, reduced energy consumption, reduced operational cost, and maintained better SLA compliance over the baseline optimisation methods. The main contribution of this work is to unify demand prediction and multi-objective optimisation into a proactive resource management framework. However, the study is limited to simulation-based evaluation and certain benchmark algorithms. Future work includes validation of the framework on real cloud platforms, integration of online learning for real-time adaptation, extension of the model to edge-cloud and fog environments, and inclusion of dynamic SLA-aware pricing mechanisms.

References

1. L. Wang, G. von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, “Cloud computing: A perspective study,” New Gener. Comput., vol. 28, no. 2, pp. 137–146, 2010.
2. H. Shukur, S. Zeebaree, R. Zebari, D. Zeebaree, O. Ahmed, and A. Salih, “Cloud computing virtualization of resource allocation for distributed systems,” J. Appl. Sci. Technol. Trends, vol. 1, no. 3, pp. 98–105, 2020.
3. S. Namasudra, “Data access control in the cloud computing environment for bioinformatics,” Int. J. Appl. Res. Bioinf., vol. 11, no. 1, pp. 40–50, 2021.
4. K. Gai, J. Guo, L. Zhu, and S. Yu, “Blockchain meets cloud computing: A survey,” IEEE Commun. Surveys Tuts., vol. 22, no. 3, pp. 2009–2030, 2020.
5. H. Hassan, A. I. El-Desouky, A. Ibrahim, E.-S. M. El-Kenawy, and R. Arnous, “Enhanced QoS-based trust assessment in cloud computing,” IEEE Access, vol. 8, pp. 43752–43763, 2020.
6. Z. S. Ageed, R. K. Ibrahim, and M. A. M. Sadeeq, “Unified ontology implementation of cloud computing,” Curr. J. Appl. Sci. Technol., pp. 82–97, 2020.
7. S. Namasudra, R. Chakraborty, and A. Majumder, “Securing multimedia using DNA-based encryption in cloud computing,” ACM Trans. Multimedia Comput. Commun. Appl., vol. 16, no. 3s, pp. 1–19, 2020.
8. N. Rasouli, R. Razavi, and H. R. Faragardi, “EPBLA: Energy-efficient consolidation of virtual machines using learning automata,” Cluster Comput., vol. 23, no. 4, pp. 3013–3027, 2020.
9. V. Bogatyrev and A. Derkach, “Evaluation of cyber-physical systems with virtual machine migration,” Computers, vol. 9, no. 2, p. 42, 2020.
10. S. Wilson and K. Platts, “The role of resource configuration on mix flexibility,” Prod. Plan. Control, vol. 20, no. 8, pp. 769–784, 2009.
11. A. Khudoyberdiev, S. Ahmad, I. Ullah, and D. Kim, “Fuzzy logic–based optimization for energy-efficient systems,” Energies, vol. 13, no. 2, p. 289, 2020.
12. J. Zhou, Y. Zhang, L. Sun, S. Zhuang, C. Tang, and J. Sun, “Stochastic virtual machine placement under resource variation,” IEEE Access, vol. 7, Art. no. 174412, 2019.
13. F. Abdessamia, W. Z. Zhang, and Y. C. Tian, “Energy-efficient virtual machine placement using gravitational search,” Cluster Comput., vol. 23, no. 3, pp. 1577–1588, 2020.
14. X. Zhang, T. Wu, M. Chen, et al., “Energy-aware virtual machine allocation with reservation,” J. Syst. Softw., vol. 147, pp. 147–161, 2019.
15. A. Ghasemi and A. Toroghi, “A multi-objective load balancing algorithm for VM placement,” Computing, vol. 102, pp. 2049–2072, 2020.
16. Y. Moaly and B. A. Youssef, “Hybrid Bayesian and cuckoo search algorithm for VM consolidation,” Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 1, 2020.
17. A. A. Khan et al., “HeporCloud: An energy and performance efficient resource orchestrator,” J. Netw. Comput. Appl., vol. 173, Art. no. 102869, 2021.
18. T. P. Chan, Y. Cheng, Y. Zhu, X. Gao, and G. Chen, “Symmetry-preserving architecture for multi-NUMA environments (SPANE): A deep reinforcement learning approach for dynamic VM scheduling,” arXiv, Apr. 21, 2025.
19. M. Zakarya et al., “Epcaware: A game-based energy and cost-efficient resource management,” IEEE Trans. Serv. Comput., 2020.
20. C. Panggabean, D. V. C, B. Gogoi, R. Limbu, and R. Sarker, “Optimized cloud resource allocation using genetic algorithms for energy efficiency and QoS assurance,” arXiv, Apr. 24, 2025.
21. A. Beloglazov and R. Buyya, “Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers,” Concurrency Comput. Pract. Exper., vol. 24, no. 13, pp. 1397–1420, 2012.
22. X. Li, J. Wu, S. Tang, and S. Lu, “Energy-efficient virtual machine placement algorithm with balanced resource utilization,” Int. J. Commun. Syst., vol. 32, no. 15, e4116, 2019.

23. V. Verma, S. Kumar, S. Kumar, D. Singh, and S. Gupta, “Optimizing Spark job scheduling with distributional deep learning in cloud environments,” J. Cloud Comput., vol. 14, Art. no. 59, Oct. 2025.
24. M. Ranjbari and J. Akbari Torkestani, “Learning automata-based energy-efficient virtual machine consolidation,” J. Parallel Distrib. Comput., vol. 113, pp. 55–62, 2018.
25. S. Singh and I. Chana, “QRSF: QoS-aware resource scheduling framework in cloud computing,” J. Supercomput., vol. 71, pp. 241–292, 2015.
26. A. G. Padasalgi, M. K. Prasad, R. I. S. Rajesh, B. Malakareddy A., and G. R. B., "Multi-Paradigm Architectural Taxonomy of Hybrid Quantum-Classical Learning Pipelines: From Sequential Offloading to Cloud-Native MLOps Orchestration," International Journal of Computational Intelligence in Engineering (IJCIE), vol. 1, no. 2, pp. 17–34, May 2026.
27. P. Mell and T. Grance, “The NIST definition of cloud computing,” NIST Special Publication 800-145, 2011.
28. J. Liu, Y. Luo, X. Xu, and L. Zhang, “An energy-aware scheduling approach based on genetic algorithm for cloud computing,” Cluster Comput., vol. 22, no. 2, pp. 519–529, 2019.
29. S. Sotiriadis, N. Bessis, and F. Xhafa, “Inter-cloud bridge system for heterogeneous cloud platforms,” Future Gener. Comput. Syst., vol. 54, pp. 180–196, 2016.
30. M. Zakarya and L. Gillam, “Modeling resource heterogeneities in cloud simulations,” Simul. Model. Pract. Theory, vol. 94, pp. 43–65, 2019.