INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

Quantum-PSO and Triplet Attention based Scheduling Mechanism for Resource Management in Cloud Environment

Nikhil Singh[1], Narander Kumar[1]*, Shishir Kumar[1]

1 Department of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, UP, India

OPEN ACCESS

Received: 01/10/2026

Accepted: 28/10/2026

Published: 11/11/2026

Citation: Singh N, Kumar N, Kumar S (2026) Quantum-PSO and Triplet Attention based Scheduling Mechanism for Resource Management in Cloud Environment. Indian Journal of Science and Technology 19(40): 2732-2744. https://doi.org/10.17485/IJST/v19i40.1325

* **Corresponding author**.

nk_iet@yahoo.co.in

Funding: None

Competing Interests: None

Copyright: © 2026 Singh et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Background/Objectives: The rapid growth of the internet and cloud-based applications has led the world to adopt cloud computing. As the number of users grow there rises some problems, virtual machine allocation and cloudlet scheduling are among them. Efficient resource allocation and scheduling are crucial for balancing the load on virtual machines, as they minimize delays and optimize performance. **Method:** This research work proposes a predictive mechanism using Triplet Attention and Quantum-behaved Particle Swarm Optimization (QPSO). By incorporating triplet Attention, VM allocation and load balancing are performed efficiently, and the QPSO optimizes the Scheduling. For predictive models' performance evaluation, R^2 score, MAE, and RMSE metrics are evaluated. **Findings:** The model achieved 96.7% R^2 score and the error metrics are 0.34 MAE and 0.68 RMSE, for experimental outcomes of cloud environment load balancing score, execution time, resource utilization, energy usage, and throughput are evaluated. The experimental results show that average execution time is reduced to 6.34ms from 9.38ms (AFPA), average energy consumption is reduced to 21.18Kwh from 29.95KWh (AFPA), load balancing improved to 98.06 from 98 (AFPA) throughput is increased from 51.63 (FSAOS) to 83.33 cloudlet/ms and load balancing score improved from 98.16 (AFPA) to 98.26. **Novelty:** The proposed model ensembles deep learning (Triplet Attention) with optimization (QPSO), and optimizes multiple key metrics at same time, the proposed model optimizes total six key metrics where the other studies only optimize two or three metrics. This work provides a comprehensive model for VM allocation and cloudlet scheduling.

Keywords: Cloud Computing; Virtual Machine Allocation; Cloudlet Scheduling; Triplet; Quantum-behaved Particle Swarm Optimization

1 Introduction

Cloud Computing has reshaped the way of organizations through which they manage, access, and operate resources. Rather than acquiring and maintaining dedicated hardware, organizations rent the desired capacity of resources from the cloud service provider (CSPs). Cloud platforms enabled on-demand access to Memory, storage and network Bandwidth, all delivered as pay-per-use services. The CSPs pool their resources, meaning the provider's infrastructure is shared among multiple customers. The provider wants to minimize its operational expenses by managing resources efficiently through multiple cloud resource management mechanisms to control the allocation, scheduling, optimization, load balancing, and migration across distributed cloud environments. These mechanisms ensure that the cloud services are delivered efficiently, reliably, and economically, while maintaining the desired quality and without violating the service level agreements (SLA). Thousands of users and application requests for computing resources at the same time. Effective resource management is needed for system stability, scalability, and performance. The Resource management involves virtualization, distributed computing, and optimized scheduling to manage the infrastructure resources cost-efficiently.

The basic idea of resource management is the utilization of physical resources through virtualization technologies in optimal way. Physical machines in the datacenters create multiple virtual machines which can be allocated the resources to users as requested. The resource management system continuously monitors the status of cloud infrastructure and adjusts resource allocation to avoid underutilization or overloading[(1)]. It aims to satisfy several crucial goals simultaneously as: optimize the performance, minimize execution time, reduce latency, and maximize throughput. Another goal is QoS assurance, in which the system ensures that to satisfy the user's requirements, like deadlines, reliability, and availability, are fulfilled during service execution. SLA compliance is also crucial because providers must fulfil contractual obligations regarding availability and performance reported by[(2)]. Resource management is to make such system which maximize resource utilization, and reduce energy consumption as well as operational costs. Efficient resource management ensures load balancing by distributing the workload across all VMs.

Modern cloud resource management uses Machine Learning (ML) and Deep Learning (DL) mechanisms to predict resource utilization and future workload. These ML techniques analyze historical data and forecast future resource demands by learning the workload patterns; techniques such as LightGBM, XGBoost, and Random Forest are used for workload forecasting mentioned by[(3)]. DL techniques such as CNN, LSTM, and Transformer are effective for temporal work- load Attention because they can learn complex sequential pat- terns[(4)]. In recent years, resource management has been procured as integration of intelligent systems. These systems represent the future of autonomous resource management in cloud computing[(5)]. Cloud resource management is a multidisciplinary field that combines distributed systems, virtualization, optimization, machine learning, networking and energy-aware computing.[(6)] It serves as the backbone of the system by optimally utilizing resources while maintaining scalability and reliability[(7)].

The main objectives of this research work are: (1) To develop a predictive model for optimizing allocation and scheduling. (2) To reduce energy consumption, execution time, Makespan. (3) To increase Throughput, load balancing, and resource utilization. Resource allocation and cloudlet scheduling have emerged as fundamental challenges because of the continuous changes in workloads. Traditional scheduling techniques like FCFS (First Come First Serve), RR (Round Robin), and Min-Min/Max-Min algorithms have been widely adopted because they are simple and have low computational cost. To overcome the drawbacks of traditional scheduling mechanisms, many researchers studied different metaheuristic optimization mechanisms[(8)]. In recent works, machine learning techniques were combined with optimization algorithms to improve scheduling decisions[(9)]. Attention-based deep learning architectures have further improved resource Attention by emphasizing workload features, as reported by[(10)].

A hybrid framework that ensembled different mechanisms and combined these with a load balancer[(11)]. Goal of the model is to decrease the Makespan, latency, and energy uses while distributing the load across cloud resources. For resource allocation, an adaptive mechanism is proposed for Optimization in a Cloud environment[(12)]. A study proposed an advanced resource allocation approach utilizing an Adaptive Flower Pollination Algorithm (AFPA). The performance is enhanced via integrating a probability function, a Levy flight balancer inertia (weight), and by improving the initial solutions with a chaotic map. This research is the inspiration for development of Attention based cloud resource optimization[(13)].

A study proposed an enhanced multi-objective VM placement (MVMP) algorithm that incorporated quantum particle swarm optimization (QPSO). It is called QPSO-MOVMP, and its aim is to explore the best solution for the VM allocation with balancing other goals[(14)]. An allocation framework is proposed for services on the cloud with defined time windows. This proposed framework first calculates the resource allocation strategy from the real-time and future load, according to Attentions. Then the PSO-GA is utilized to predict and take real-time actions for exploring the resource allocation strategy[(15)].

For deep learning integration, a study proposed a model that managed the resource allocation problem as a multi-agent Markov Decision Process (MDP), where each VM is treated as an agent. This work optimizes three metrics at the same

time[16]. Another study proposed a machine learning mechanism to predict workload and execution time under varying incoming load in the cloud environment. This mechanism relies on the parameters that reflect the runtime information for high-accuracy Attentions[17]. Another smart system utilizes Clipped Double Deep Q-Learning (CDDQL)[18] and PSO[19] for resource allocation is introduced in the cloud computing environment. The task of PSO is to determine priorities for cloudlets, and the task of CDDQL is to serve as the core of the smart system, allocating incoming cloudlets to the most suitable VM resources, as reported by[20]. A study about the application of DL, particularly CNNs, to the resource allocation and scheduling problem is discussed[21]. In another study, the Resource Allocation using the hierarchical autonomous polynomial convolutional neural network (RA-HAPCNN-CC) framework is introduced to address allocation and scheduling issues. This work gives the idea of CNN based allocation and its limitations[22].

A hybrid mechanism for secure VM allocation with effective cloudlet scheduling in a cloud computing is introduced by[23]. Initially, a lightweight scheduler based on the improved Particle Swarm Optimization algorithm (IPSO)[24] is used to improve performance. Then, for load and Bandwidth, the Graph Attention Neural Network (GANN) is employed for efficient resource management. A Transformer-based Attention model like Temporal Fusion Transformer (TFT) offers enhanced accuracy but introduces latency that limits the real-time deployment. This study proposed LSTM-MARL-Ape-X, a mechanism incorporating BiLSTM for load Attention with MARL models in a distributed architecture[25]. Another study presented the first comprehensive study of five pattern-learning and analysis-driven techniques applied to achieve higher efficiency and performance in multi-constrained cloud resource management.

A study manifested the utility and pattern learning-based resource management as compared with state-of-the-art resource management techniques[26]. The five principles techniques are thoroughly discussed with a coherent depiction of the intended concept, along with numerical illustrations. The most prominent novel models for each technique are further distinguished by their objectives, providing an extensive survey and comparison. Besides conceptual and theoretical analysis, the leading models underlying the major resource management techniques are implemented on a common platform and thoroughly examined using real-world Google Cluster workload traces by[26].

From the extensive review of literature, the findings are that most existing studies focus on either optimization-based scheduling or machine learning-based Attention independently. Limited Attention has been given to integrating deep learning-based workload Attention[27]. Many approaches do not simultaneously optimize metrics like execution time, throughput, load balancing, resource utilization, energy consumption, and Makespan. Though there are studies that optimizes multiple objectives[28] but the proposed model will ensemble deep learning with metaheuristic optimization. These limitations motivate the development of an intelligent hybrid scheduling model capable of providing adaptive, load-aware resource allocation in a dynamic environment. An Attention-guided scheduling mechanism for resource management is proposed, which combines optimization scheme with Attention to optimizes all the important metrics simultaneously and comparisons are also done with state-of-the-art techniques. Table 1 represents the abridgment and comparative analysis of literature used in this paper.

Table 1. Comparative Analysis of Existing Literature

Reference	Abridgment	Method	Objective	Limitations
[1,9,15,17,19,20]	**Machine Learning, Attention**	**Autoscaling, Machine Learning Approaches, PSO-GA**	**Explore the concept of Machine Learning**	**Focus on only one metric**
[2,3,12,14,21,29]	**Scheduling and Allocation**	**VM Placement, HAAPCNN, CNN**	**Basic concepts, Energy and QoS, Bottleneck**	**Higher Complexity, Needs heavy computing power**
[4,8,12,25,26]	**Optimization**	**Resource Utilization, Scheduling**	**Optimization using GA and Scheduling**	**Optimizes only few metrics**
[11,13,29–31]	**Comparison Techniques**	**AFPA, FSAOS, QPSO, Triplet**	**Core concept**	**Needs integration of deep learning,**
[16,18,23,24]	**Deep RL**	**RL, NN, GW optimization**	**Allocation and scheduling**	**Review, Allocation**
[5–7,10,28]	**Attention based forecasting**	**ARNN, Quantum algorithms**	**Ways of utilizing Attention**	**Optimizes two or three metrics**

The obtain results and comparison is discussed in detail in the results and discussion section sec 6.

2 Methodology

The proposed Quantum-PSO and Attention-guided scheduling and allocation approach discusses its primary contribution: an Attention mechanism for load-balanced allocation that decides where to place incoming load based on forecasted values.

QPSO for optimizes cloudlet scheduling, by determining the execution order and timing of cloudlets. Figure 1 shows the system model of proposed Attention-guided Quantum-behaved Particle Swarm Optimization mechanism. The detailed working of the proposed model is given below.

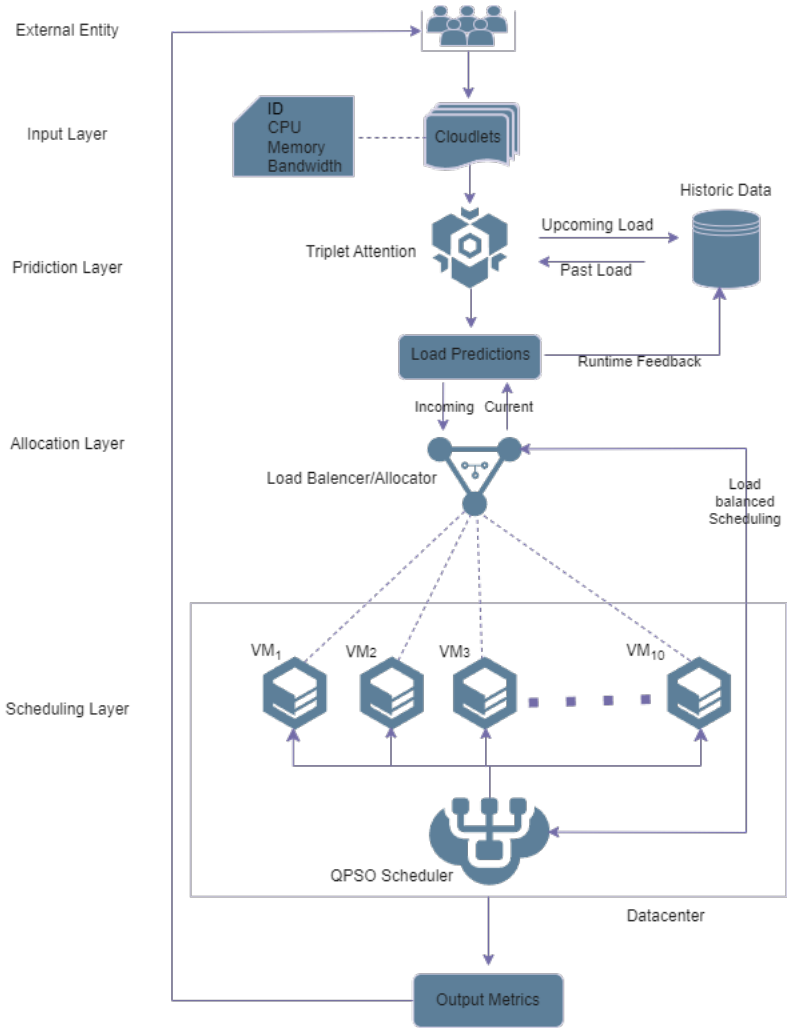


Fig 1. The System Model of the Proposed Quantum-PSO and Triplet Attention-based Scheduling Mechanism

2.1 System Model

A Quantum-PSO and Attention-based model for resource management and scheduling is proposed, which balances VM load, incorporates the Triplet Attention mechanism with QPSO. The Attentions form Triplet Attention which is used in VM allocation, efficient VM allocation is crucial for balancing the load on data centers. The Attention mechanism observes the ongoing execution on VMs and incoming load on every VM, it predicts the incoming load and based on monitoring of every VM it distributes the load in a balanced manner. For executing cloudlets, the QPSO determines the order and timing of cloudlets for each VM. The model uses 70/15/15 train/validation/test split, input features are CPU, Memory and Bandwidth along with timestamp and Id and output variables are CPU Memory and Bandwidth. Sequence window is 25 past time steps and forecast horizon is 3 steps ahead. Attention branches are ChannelGateH, ChannelGateW and SpatialGate, for attention combination arithmetic mean of these branches is used. Sigmoid activation function is used for attention activation the input tensor is B $\times$ 3 $\times$ 25 $\times$ 3. For QPSO the objectives are to minimize the error during allocation and optimize the key metrics, while the parameters are 40 particles, max iterations 70, search dimensions 5, contraction and expansion coefficient 0.6, early stopping

patience 20, weight decay 0.001. The same or equivalent parameter value is used in comparison techniques, and same dataset is utilized.

2.2 Triplet Attention

The self-attention mechanism-based dot-product operations capture long-term dependencies, which led the transformer architecture to achieve remarkable success in real-world applications[29] introduced the Triplet Attention, which introduces triplet connections as the complementary dependency. It allows the self-Attention to learn relations among diverse branches and capture complex dependencies, capturing cross-dimensional interactions among Height (H), Width (W), and Channel (C). The core idea is to map the Attention across three different pairs of dimensions. • C-W interaction • C-H interaction • H-W interaction.

The model rotates the input tensor to learn relationship from different perspectives. Traditional Attention mechanisms often avoid interactions among dimensions, but Triplet Attention have (C-H) (C-W) (H-W) leads to rich feature learning. The input X fetched into three types of vectors, one is query Q = XWQ and another two are keys, $K_{(1)}$ =XWk_1 and $K_{(2)}$ = XWk_2. The key vectors K_j (1) and K_k (2) form a plane, by using the normal vector as equation 1:

$$K_{j\times k} = K_j^{(1)} \times K_k^{(2)} \tag{1}$$

Where x is the cross product of two vectors. The value of $K_{j\times j}$, $K_{j\times j} = |K_{j(1)}|K_{k(2)}|\text{Sin}\Theta$, can be represented as a parallelogram, whose positive area having two vectors as sides, but if these two keys are same then the angle will approach to zero, the magnitude will start to decrease. In an extreme case, if the angle between these vectors is zero, then the parallelogram will disappear. To address this problem, the Scaler Triplet Product (STP) is used to capture complex dependencies. For the query and keys, the (i,j,k)-th STP score is evaluated as equation 2[29]:

$$T_{ijk} = Q_i K_{jk}^T = Q_i (K_j^{(1)} \times K_k^{(2)})^T \tag{2}$$

Triplet Attention operates on three parallel branches that analyze the input feature map from different dimensions. An input tensor represented as $X \in R^{C\times H\times W}$ the model creates different rotated or permuted versions of the tensor to capture cross-dimensional relations. One branch learns channel-width interactions, another captures channel-height dependencies, while the third one focuses on Attention across height and width dimensions. These transformations allow the model to analyze features from multiple orientations rather than relying solely on direct spatial mapping. Each branch implements a Z-pooling operation, which combines average and max pooling. Where global contextual information is managed by average pooling, and the most dominant features within the feature map is managed by max pooling. The Attention mechanism learns important semantic and structural information simultaneously by integrating both pooling strategies.

Quantum-behaved Particle Swarm Optimization:

The PSO is inspired by the physical behavior of particles, where it assumes that particles move in a physical space with properties of position and motion. In PSO, every particle in the swarm population refines itself based on past experiences and the location of the best particle[32]. In a quantum system, particles exhibit quantum behavior by occupying states in the Hilbert space, this space provides the quantum inspiration to create the quantum states and probabilistic behavior of particles instead of depending on Newtonian mechanics. In quantum-behaving particle swarm optimization (QPSO), particles are pulled towards a quantum-field well. The QPSO has been introduced to maintain a balance between local refinement and global exploration and to avoid early convergence[30]. In PSO a particle's next position relies on its current position and velocity where in QPSO, the algorithm maintains the particle's position probabilistically around an attractor determined by its personal best and the swarm's best position. Unlike PSO, which updates velocity, QPSO samples new position using probability distribution whose spread is controlled by contraction-expansion coefficient. The capability of QPSO to successfully reach global minima is improved by including the mean best location (mbest), the global reference point, which influences other particles to move towards the global best. The mbest is the mean of the personal best (pbest) of all particles, which slows the convergence of particles while allowing them to explore globally, and is evaluated as in equation 3.

$$m_{best} = \frac{1}{N}\sum_{i=1}^{N} pbest_i \tag{3}$$

The logarithmic probabilistic term enables particles to search globally while still converging toward promising regions. Higher values encourage global search, whereas lower values promote local refinement near optimal solutions calculated as equation

4.

$$X_i(t+1) = P_i \pm \beta |m_{best} - X_i(t)| \ln\left(\frac{1}{\mu}\right) \tag{4}$$

where:

- X_i(t+1) is new location of the i-th particle at t+1,
- P_i is local attractor of the particle,
- β is the contraction–expansion coefficient value should be < 1.781.
- μ is a random number between 0 and 1.
- ln (1/μ) represents the quantum displacement factor used to update particle positions.

QPSO has been widely applied in engineering optimization, machine learning, neural network training, feature selection, scheduling, and cloud resource allocation. In cloud computing research, QPSO is especially effective for QoS-aware task scheduling, VM allocation, load balancing, and energy optimization. For example, in QoS-aware cloud scheduling, QPSO can optimize resource allocation by simultaneously minimizing execution time, SLA violations, response time, energy consumption, and cost.

The algorithm 1 and algorithm 2 represent the Triplet Attention-based workload Attention and load balancing using QPSO for resource allocation and cloudlet scheduling, respectively.

Algorithm 1: Triplet Attention-Based Workload Attention
Require: Historical workload dataset D
Ensure: Predicted future workload L_{pred}
1: Collect cloudlet attributes: (CPU, Memory, Bandwidth)
2: Preprocess and normalise dataset D
3: Extract workload feature matrix F
4: for each feature map in F do
5: Compute Channel Attention Score
6: Compute Height-wise Attention Score
7: Compute Width-wise Attention Score
8: end for
9: Fuse all Attention maps
10: Generate enhanced feature representation F′
11: Feed F′ into Attention network
12: Predict upcoming workload: L_{pred} = Predict (InverseTransform(F′t+1:t+H))
13: Store Attention results in the workload repository
14: Return L_{pred}

The workload feature matrix F is obtained by preprocessing and normalizing the historical workload dataset, every feature is normalized using Min-Max. Then F is calculated $F \in R^{T \times C}$, T is timesteps and C is number of features. Then the enhanced feature representation F' is calculated by applying Triplet Attention to the input feature maps. For every branch the map is generated using dimensional permutation, Z-pooling and sigmoid function.

Algorithm 2: Load-Balanced QPSO Resource Allocation
Require: Predicted workload L_{pred}, VM set V
Ensure: Optimal Cloudlet-to-VM Allocation
1: Initialize particle population P
2: Generate initial VM allocation solutions
3: Set maximum iterations T
4: for each particle pi do
5: Evaluate Fitness(pi)
6: Store Personal Best P_{best}
7: end for
8: Determine Global Best G_{best}
9: for t = 1 to T do
10: for each particle pi do
11: Calculate Load Balance Score:
12: Calculate Resource Utilization:
13: Compute Fitness:
14: Compute Mean Best Position:
15: Generate random numbers u, r ∈ (0, 1)
16: Update particle position:
17: Evaluate new Fitness
18: if Fitness improves then
19: Update P_{best}
20: end if
21: end for
22: Update G_{best}
23: end for
24: Allocate cloudlets according to G_{best}
25: Deploy workload on selected VMs
26: Collect QoS metrics and runtime feedback
return Final Allocation Plan

2.3 Dataset Description

Algorithms and models are as good as the data they receive; A dataset that is the Alibaba Cluster Trace 2017, is utilized in this work, containing almost 100 thousand cloudlets. It contains metrics such as CPU, Memory, disk utilization and network traffic. Before the model training, several preprocessing operations were performed. Missing values are removed to improve the dataset quality. Then Min-Max normalization is applied to scale the features within the range 0 to 1. To capture temporal dependencies, sequential data was transformed into time-series window. The dataset is selected due to its realistic workload fluctuations, which are commonly observed in cloud computing environments. Therefore, it is appropriate for evaluating the performance of the proposed mechanism.

2.4 Proposed Attention guided mechanism

The model is designed for intelligent Attention and optimization of resource utilization patterns in cloud environments. Its primary purpose is to learn hidden relationships between multiple resource metrics. The Triplet Attention model is trained and tested along with QPSO. The model training is done for 50 epochs with an early stopping mechanism, the patience level is 20 consecutive epochs, meaning if validation loss does not improve for 20 consecutive epochs, the training stops automatically. For QPSO, there are 20 exploratory epochs for hyperparameter optimization, so the final training includes roughly 70 epochs of learning activity. This two-level training is used to improve convergence quality, Attention accuracy, parameter tuning efficiency.

The workflow begins when multiple users send requests to the cloud environment, these requests are divided into smaller executable tasks known as cloudlets C= (c1, c2, c3…. cn). Each cloudlet contains information related to cloudlet id, CPU demand, Memory usage, and Bandwidth requirement D= (ci {CPU}, {mem}, {band}, i=1, 2… n). These cloudlets are the fundamental workload that to be processed by the cloud datacenter. After cloudlet generation, the cloudlets are fetched to the

Attention layer, where workload-related parameters are extracted and pre-processed into a meaningful representation. Instead of using a single Attention map, the Triplet Attention mechanism rotates feature representations and computes Attention from multiple perspectives. To manage the risk of vanishing and exploding gradients, the batch normalization process is used, which is calculated as equation 5, where μ is the batch mean, σ^2 is the variance of the batch, ε is a constant.

$$= \frac{a-\mu}{\sqrt{\sigma^2+\varepsilon}} \tag{5}$$

Batch normalization is especially important in cloud workload Attention because resource data often fluctuates dynamically. The architecture combines deep feature extraction with Attention-based learning and evolutionary optimization to improve Attention accuracy and scheduling efficiency. The extracted features are then combined with historical execution stored in the repository H=({ID}, {CPU}, {mem}, {band}). The historic data plays a critical role in the model, it maintains records of previous cloudlet executions, resource utilization patterns, VM behavior, SLA violations, scheduling outcomes, and energy consumption statistics. This historical information allows the model to learn workload trends and recurring execution behaviors. The channel Attention (C) identifies the most important resource metric at that instance, CPU in this case, and these Attention weights are calculated as in equation 6, where, fc(.) denotes the feature transformation, σ(.) denotes the sigmoid activation function.

$$A_C = \sigma(f_c(X_t)) \tag{6}$$

This equation indicates that higher availability CPU and Memory resource (that is calculated by equation 8) in a VM will receive higher Attention. The second branch studies the relationships among resources; it does not focus on which resource is important, but rather on how resources influence one another. The Attention map is computed as equation 7, where, f_r(.) denotes the resource-relation learning function that captures inter-resource dependencies and interactions.

$$A_R = \sigma(f_r(X_t) \tag{7}$$

The pattern of Memory utilization is calculated by the equation 8:

$$A_M = \sigma(f_t(X_t)) \tag{8}$$

Then these three Attention weights are blended into a single attention weight by the equation 9:

$$A = [A_R.A_C(x)] + [(1-A_R).A_M(y)] \tag{9}$$

The blended Attention contains richer information than any single Attention branch. This blended attention decides how much the model can rely on past patterns. By utilizing past workload patterns, the model becomes capable of adaptive and intelligent Attention rather than reusing the static scheduling mechanisms. The cloudlet scheduler manages the internal execution of assigned cloudlets. The cloudlet scheduler controls cloudlet sequencing, CPU sharing, execution priorities, and time allocation policies. But before scheduling the cloudlets the QPSO is employed for cloudlet allocation to the best probable VM, the QPSO particles explores all the VMs in the environment and calculates the fitness by equation 10 where, C denotes the CPU in use, M denotes Memory in use, and B denotes Bandwidth in use, and w1, w2, w3 are the best particle of explored among all the VMs.

$$Fitness = w_1 C + w_2 M + w_3 B \tag{10}$$

But in this model the attention that is calculated from the equation 9 is combined with the QPSO particle explorations, the fitness is calculated by equation 11.

$$Fitness = \sum_{h=1}^{H} \begin{matrix} (w_1 \bullet prob_C(h) \\ +w_2 \bullet prob_M(h) + w_3 \bullet prob_B(h)) \end{matrix} \tag{11}$$

Without using an Attention mechanism, QPSO schedules cloudlets based solely on the current state and initializes particles randomly. But with the Triplet Attention QPSO scheduled cloudlets using future Attention state, this makes the QPSO predictive scheduler from reactive scheduler. The probability of selecting VM is calculated as equation 13:

$$Prob(VM_j) = \frac{Fitness_j}{\sum_{k=1}^{N} Fitness_k} \tag{12}$$

The attention score is calculated from equation 9 and the probability of selecting VM is calculated from the equation 12 The proposed model optimizes allocation based on prediction score and it will select the best fit VM every time.

The best probable VM is selected based on the calculations from the equation 12. The model provides an intelligent and adaptive solution for cloud computing environments. The integration of Triplet Attention with predictive VM scheduling significantly improves workload Attention accuracy, resource utilization efficiency, execution stability, energy optimization and these metrics are optimized by the QPSO exploration within the VM and the best particle is calculated by the equation 14 where, Ex denotes the execution time, Lb denotes the Load balancing score, Ru stands for Resource utilization, E denotes energy consumption and Tp denotes the Throughput.

$$\text{Best}_{\text{p}} = W_1 Ex + W_2 Lb + W_3 Ru + W_4 E + W_5 Tp \tag{13}$$

The best particle that optimizes these all metrics is selected and based on this particle exploration the cloudlets are executed.

3 Results and Discussion

The Attention model was implemented using Python 3.10 (CUDA), and the cloud environment was created using CloudSim. Specification of the hardware is: AMD Ryzen 5 7535HS CPU, 16 GB RAM, 6GB GPU (NVIDIA RTX 3050), 512 GB SSD. The virtualization platform Xen is utilized and the simulation environment consists 10 heterogeneous VMs each equipped with 4 processing elements (PEs), VM capacity varies from 2500 MIPS to 5000 MIPS, Memory capacity ranges from 16 GB to 256 GB, Network Bandwidth configured between 10000 Mbps to 50000 Mbps. Energy consumption is calculated on model with idle and peak value 100W and 250W. For communication between the Triplet Attention mechanism and the cloud environment, a Py4J connection is established.

The Triplet Attention model achieved very effective confidence score of 96.7% based on error metrics, and the error metrics MAE and RMSE are 0.34, 0.68. After training and testing, the model is integrated into a cloud environment via a Py4J connection, the attention model was integrated in cloud environment due to its high confidence score, where it is evaluated on performance metrics such as load balancing, execution time, Makespan, resource utilization, throughput, and energy consumption. The proposed Attention-guided Quantum-behaved Particle Swarm Optimization (AQPSO) model is compared with novel techniques that already have very effective performance in a cloud environment. The model is compared with the Advanced Flower Pollination Algorithm (AFPA) [13], the Quantum-behaved Particle Swarm Optimization (QPSO) [13], and the Fruit-Fly Simulated Annealing Optimization Scheme (FSAOS) [13,31]. Alibaba cluster trace dataset is used for the performance evaluation and comparison, the dataset contains 100000 cloudlets with ID, CPU, Memory, Bandwidth metrics. The model is compared on 5 thousand, 10 thousand, 20 thousand and 50 thousand cloudlets.

The first evaluation metric is load-balancing score, the Figure 2 shows the comparison of all the techniques and based on load-balancing score the Attention-based allocator decides which VM to be allocated to cloudlets. This score indicates to the allocator which VM is more or less free; a free VM is more stable than a busy VM. The load-balancing score is calculated as equation 13 where, U_i denotes the composite utilization of the i-th VM, computed from CPU, Memory, and Bandwidth usage, $\bar{U}$ is the average utilization of all VMs, m is the total number of VMs.

$$LB_{score} = \left(1 - \frac{\sqrt{\frac{1}{m}\sum_{i=1}^{m}(U_i - \bar{U})^2}}{\bar{U}} \right) \times 100 \tag{14}$$

The next evaluation metric is resource utilization. This metric indicates how much CPU, Memory, and Bandwidth the VMs utilize to process the load in the cloud environment. Figure 3 shows the resource utilization comparison.

The next evaluation metric is energy consumption, Figure 4 shows the comparison of techniques, and it is calculated as equation 15 where, P_{idle} is the power consumed by a VM or host when operating in the idle state, P_{max} is the highest power consumption at maximum resource utilization, U_i represents utilization level of the i-th VM or host, T_i is execution duration of the i-th VM or host, and m denotes the total number of VMs.

$$Energy = \sum_{i=1}^{m} (P_{idle} + (P_{\max} - P_{idle}) \times U_i) \times T_i \tag{15}$$

The next evaluation metric is Makespan, the Figure 5 shows how much time each technique has taken to execute all the cloudlets, it is the total runtime of the batch:

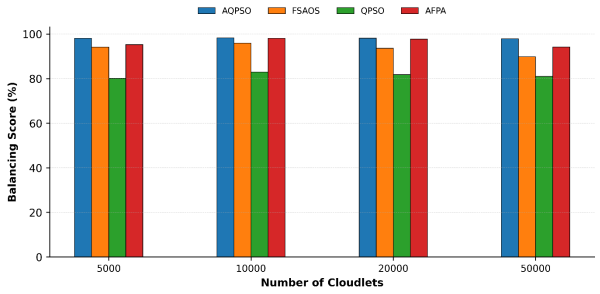


Fig 2. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Load Balancing metric

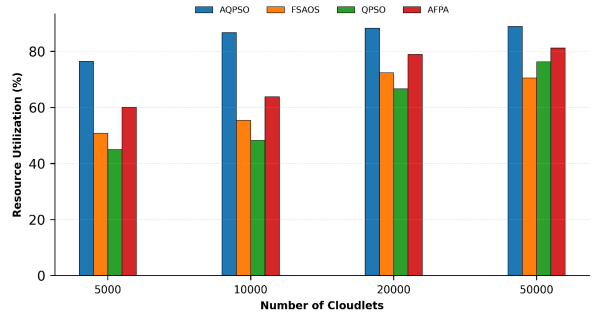


Fig 3. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Resource Utilization metric

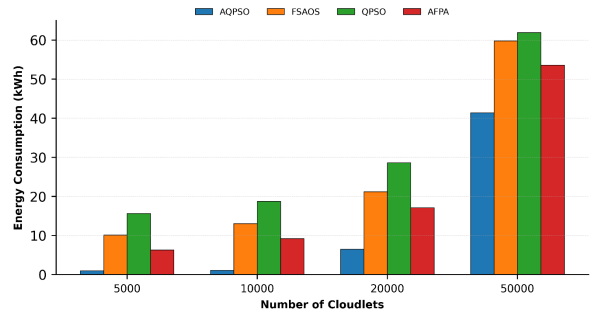


Fig 4. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Energy Consumption metric

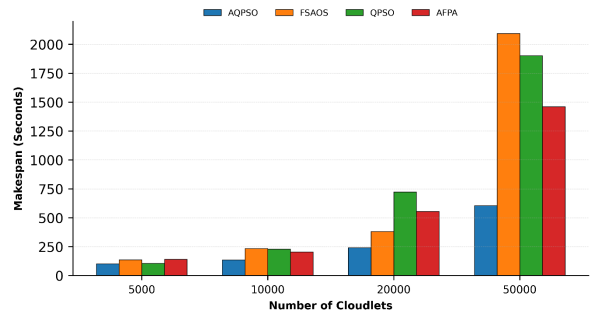


Fig 5. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Makespan metric

The next evaluation metric is Throughput, the Figure 6 shows that how many cloudlets are executed by each technique in one second and it is calculated as equation 16:

$$Throughput = \frac{N_{completed}}{TurnaroundTime} \tag{16}$$

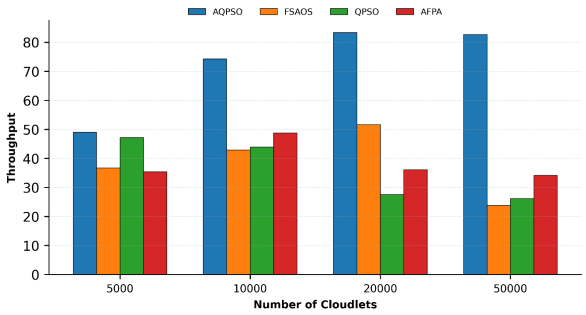


Fig 6. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Throughput metric

The observed reduction in throughput as workload increases is reasonable, as the available computing resources remain constant while the number of cloudlets grows significantly.

The next metric that is evaluated is execution time (ms) for all cloudlets; the Figure 7 shows the execution time for every cloudlet. 1000 samples of the execution time are plotted in the Figure 7 where the plotting batch is 100, the Figure 7 clearly denotes that the proposed AQPSO model outperformed all these novel techniques. The execution time is calculated as equation 17:

$$ExecutionTime = FinishTime - StartTime \tag{17}$$

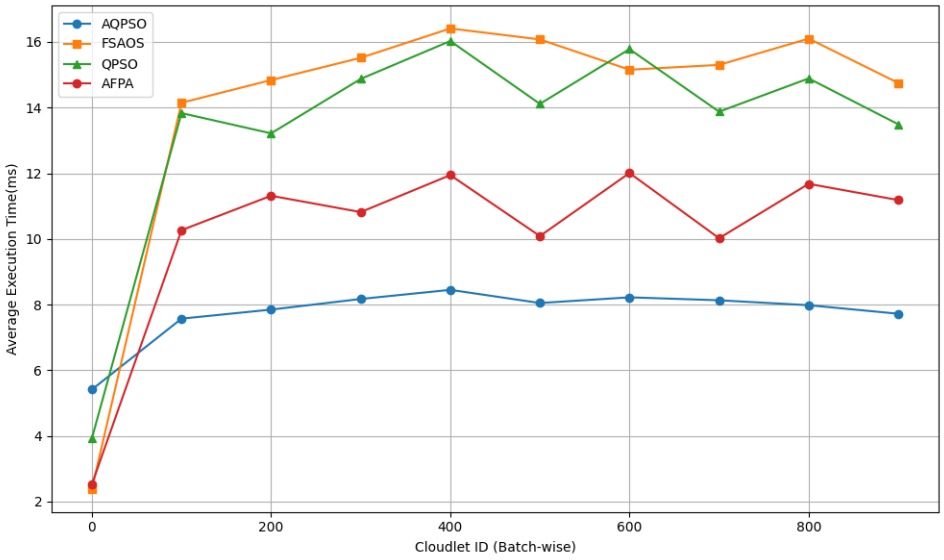


Fig 7. Comparison of AQPSO, QPSO, AFPA, FSAOS techniques on Execution Time metric

The Table 2 shows the comparison of all techniques where it can be observed that load balancing score improved 98.26 from 98.16 of AFPA, resource utilization improved 88.90% from 81.20% of AFPA, energy consumption reduced 41.40KWh (on 10 thousand cloudlets) from 53.60KWh of AFPA, Throughput is increased 83.33 cloudlet/ms from 51.63 cloudlet/ms of FSAOS, Execution Time is compared in Figure 7 and AQPSO has taken less time to execute cloudlets Compared to other techniques.

Table 2. Comparison of techniques (Max load 50000 cloudlets)

Techniques	Load Balancing	Resource Utilization	Energy Consumption	Throughput	Execution Time(max)
AQPSO	98.26	88.90%	41.40KWh	83.33 cloudlet/ms	19.48 ms
FSAOS	95.89	72.40%	59.80KWh	51.63 cloudlet/ms	24.73 ms
QPSO	82.96	76.30%	61.90KWh	47.2 cloudlet/ms	25.62 ms
AFPA	98.16	81.20%	53.60KWh	48.8 cloudlet/ms	15.6 ms

4 Conclusion

The Attention-driven allocation and scheduling approach helps prevent resource congestion, reduces execution delays, improves load balancing, and enhances overall Quality of Service. To address the identified research gap of optimizing multiple metrics and integrating deep learning with optimization scheme. The proposed work presents an intelligent cloud resource allocation and scheduling mechanism that combines Triplet Attention-based workload Attention with predictive virtual machine allocation and QPSO with Attention-based scheduling to improve resource allocation and task execution efficiency in cloud environments. To integrate these two techniques the equation 11, equation 12 and equation 13 plays critical role. The model obtained very effective predictive accuracy 96.7% based on error metrics, then it is employed in the cloud environment, where it outperformed state-of-the-art techniques Quantum-behaved Particle Swarm Optimization (QPSO), Advanced Flower Pollination Algorithm (AFPA), Fruit-Fly Simulated Annealing Optimization Scheme (FSAOS) in all the comparison metrics Execution-time (when load is minimum it is around 0.4 but it increases up to 19.48 milli sec while load increases), Throughput (49 cloudlet/milli sec when load is minimum and it increases upto threshold of 83.33 cloudlet/milli sec while load increases), the metric improved 61.39% against FSAOS , 76.54% against QPSO and 70.75% against AFPA, Resource utilization (76.4% when load is minimum and it increases up to 88.90% when load is maximum), the metric improved 22.79% against FSAOS, 16.51% against QPSO and 9.48% against AFPA, Load-balancing (Load Balancing score remains almost constant between 97.87 to 98.26), the metric improved 2.47% against FSAOS, 18.44% against QPSO and 0.10% against AFPA, Energy-consumption (0.97 KWh when load is minimum and it increases up to 41.4 KWh when load is maximum), the metric improved 30.76% against FSAOS, 33.11% against QPSO and 22.76% against AFPA, and Makespan (The makespan of model is 101.9 seconds when load is minimum and it increases up to 604.7 seconds when load is maximum), the makespan of FSAOS ranges from 173 seconds to 2103 seconds, the makespan of QPSO ranges from 142 seconds to 1801 seconds, the total run time of AFPA ranges from 181 seconds to 1454 seconds. The proposed Attention-guided scheduling and allocation mechanism obtain better results on every metric compared with other techniques.

For future perspective, the model can be made scalable because the proposed model has a fixed number of VMs (10 heterogenous VMs having different capacity configuration as mentioned above). Resource provisioning can be applied to transfer resources from one data center to another. As resources are made scalable then the model will adjust its VMs or datacenters capacity according to the varying load.

5 Acknowledgements

No external fund received for this research.

The first author thanks the academic advisors, [Dr Narander Kumar & Prof. Shishir Kumar], for their invaluable guidance, constructive feedback, and unwavering encouragement.

6 Data Availability

The dataset used in this research work publicly available on: https://www.kaggle.com/datasets/tirkeydivyanshu/alibaba-cluster-traces-2017/data?select=container_usage.csv

References

1) Pintye I, Kovács J, Lovas R. Enhancing Machine Learning-Based Autoscaling for Cloud Resource Orchestration. *Journal of Grid Computing*. 2024;22:68. https://doi.org/10.1007/s10723-024-09783-1.
2) Saidi K, Bardou D. Task scheduling and VM placement to resource allocation in Cloud computing: challenges and opportunities. *Cluster Computing*. 2023;26:3069–3087. https://doi.org/10.1007/s10586-023-04098-4.
3) Rathee A, Dalal S. A systematic literature review of machine learning-based resource allocation techniques in cloud computing. *Computing*. 2025;107:179. https://doi.org/10.1007/s00607-025-01526-8.

4) Dehury JP, Kumar D, Kumar S. Joint optimization of resource utilization and makespan in cloud computing using genetic algorithm. *Discover Computing*. 2026;29:315. https://doi.org/10.1007/s10791-026-10167-0.
5) Shao S, Ding X, Zhao B. Attention-based workload Attention and dynamic resource allocation for heterogeneous computing environments. *Scientific Reports*. 2026;16:8571. https://doi.org/10.1038/s41598-026-38622-4.
6) Predić B, Jovanovic L, Simic V. Cloud-load forecasting via decomposition-aided Attention recurrent neural network tuned by modified particle swarm optimization. *Complex & Intelligent Systems*. 2024;10:2249–2269. https://doi.org/10.1007/s40747-023-01265-3.
7) Deng W, Qiu Z, Liu X. An intelligent scheduling and resource Attention framework using multidimensional data streams and deep reinforcement learning. *Neural Computing and Applications*. 2026;38:457. https://doi.org/10.1007/s00521-026-11917-3.
8) Bouflous Z, Haraka F, Ouzzif M. An optimized bias-resilient double-indexed throttled scheduling algorithm for fair and efficient load balancing in cloud datacenters. *Journal of Cloud Computing*. 2026;15:101. https://doi.org/10.1186/s13677-026-00924-3.
9) Nashaat M, Moussa W, Rizk R. Dynamic machine learning approach for workload Attention in cloud environments. *Scientific Reports*. 2026;16:10983. https://doi.org/10.1038/s41598-026-40777-z.
10) Kashyap S, Singh A. Attention-based scheduling techniques for cloud data center's workload: a systematic review. *Cluster Computing*. 2023;26:3209–3235. https://doi.org/10.1007/s10586-023-04024-8.
11) Shaik MB, Reddy KS, Chokkanathan K, Biabani SA, Shanmugaraja P, Brabin DRD. A Hybrid Particle Swarm Optimization and Simulated Annealing with Load Balancing Mechanism for Resource Allocation in Fog-Cloud Environments. *IEEE Access*. 2024;12:172439–172450. https://doi.org/10.1109/ACCESS.2024.3489960.
12) Moazeni A, Khorsand R, Ramezanpour M. Dynamic Resource Allocation Using an Adaptive Multi-Objective Teaching-Learning Based Optimization Algorithm in Cloud. *IEEE Access*. 2023;11:23407–23419. https://doi.org/10.1109/ACCESS.2023.3247639.
13) Dankolo NM, Radzi NHM, Mustaffa NH, Osman NA, Gabi D, Yusuf MN. Enhanced Task Scheduling and Resource Allocation in Edge-Cloud Continuum Using Modified Flower Pollination Algorithm. *IEEE Access*. 2024;12:162299–162310. https://doi.org/10.1109/ACCESS.2024.3490177.
14) Elsedimy EI, Herajy M, Abohashish SMM. Energy and QoS-aware virtual machine placement approach for IaaS cloud datacenter. *Neural Computing and Applications*. 2025;37:2211–2237. https://doi.org/10.1007/s00521-024-10872-1.
15) Chen Z, Yang L, Huang Y, Chen X, Zheng X, Rong C. PSO-GA-Based Resource Allocation Strategy for Cloud-Based Software Services with Workload-Time Windows. *IEEE Access*. 2020;8:151500–151510. https://doi.org/10.1109/ACCESS.2020.3017643.
16) Prabha PS, Rengarajan A. Adaptive Cloud Resource Allocation Using Attention-Driven Deep Reinforcement Learning. *Engineering, Technology and Applied Science Research*. 2025;15(6). https://doi.org/10.48084/etasr.13443.
17) Pham TP, Durillo JJ, Fahringer T. Predicting Workflow Task Execution Time in the Cloud Using a Two-Stage Machine Learning Approach. *IEEE Transactions on Cloud Computing*. 2020;8(1):256–268. https://doi.org/10.1109/TCC.2017.2732344.
18) AL-Jumaili AHA, Seno ME, Awad WK. A fusion deep Q-learning and particle swarm optimization algorithm for adaptive resource allocation in cloud computing circumstances. *Scientific Reports*. 2026;16:17656. https://doi.org/10.1038/s41598-025-33498-2.
19) Reddy BR, Indiramma M, Nagarathna N. Intelligent particle swarm optimization-based resource provisioning technique in cloud computing. *Indian Journal of Science and Technology*. 2023;16(16):1241–1249. https://doi.org/10.17485/IJST/v16i16.308.
20) Javaheri SDA, Ghaemi R, Naeen HM. An autonomous architecture based on reinforcement deep neural network for resource allocation in cloud computing. *Computing*. 2024;106:371–403. https://doi.org/10.1007/s00607-023-01220-7.
21) Aditi VK, Prasad, Gerogiannis VC. Utilizing convolutional neural networks for resource allocation bottleneck analysis in cloud ecosystems. *Cluster Computing*. 2025;28:22. https://doi.org/10.1007/s10586-024-04720-z.
22) Gurusamy S, Selvaraj R. Resource Allocation with Efficient Task Scheduling in Cloud Computing Using Hierarchical Auto-Associative Polynomial Convolutional Neural Network. *Expert Systems with Applications*. 2024;249(Part B):123554. https://doi.org/10.1016/j.eswa.2024.123554.
23) Zhou H. A Novel Approach to Cloud Resource Management: Hybrid Machine Learning and Task Scheduling. *Journal of Grid Computing*. 2023;21:68. https://doi.org/10.1007/s10723-023-09702-w.
24) Mehta V, Gupta A. A hybrid grey wolf and improved particle swarm optimization technique with Pareto knee selection for energy efficient load balancing in cloud computing. *Discover Applied Sciences*. 2026;8:337. https://doi.org/10.1007/s42452-026-08395-6.
25) Manhary FN, Mohamed MH, Farouk M. A scalable machine learning strategy for resource allocation in database. *Scientific Reports*. 2025;15:30567. https://doi.org/10.1038/s41598-025-14962-5.
26) Saxena D, Singh AK. Workload Pattern Learning-Based Cloud Resource Management Models: Concepts and Meta-Analysis. *IEEE Transactions on Sustainable Computing*. 2025;10(3):418–438. https://doi.org/10.1109/TSUSC.2024.3456429.
27) Chandrashekar C, Krishnadoss P, Sivaraman AK. Efficient task scheduling in cloud computing environment using opposition-based firebug tunicate optimization. *Cluster Computing*. 2026;29:819. https://doi.org/10.1007/s10586-026-06561-4.
28) Werner T, Ullinger F. Quantum algorithms for scheduling problems: a survey. *EPJ Quantum Technology*. 2026;13:31. https://doi.org/10.1140/epjqt/s40507-026-00494-y.
29) Zhau H, Li J, Peng J, Zhang S, Zhang S. Triplet Attention: Rethinking the similarity in Transformers. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2021;p. 2378–2388. https://doi.org/10.1145/3447548.3467241.
30) Omkar SN, Khandelwal R, Ananth TVS, Naik GN, Gopalakrishnan S. Quantum behaved Particle Swarm Optimization (QPSO) for multi-objective design optimization of composite structures. *Expert Systems with Applications*. 2009;36(8):11312–11322. https://doi.org/10.1016/j.eswa.2009.03.006.
31) Gangadevi E, Soufiane BO, Balusamy B. A novel hybrid fruit fly and simulated annealing optimized faster R-CNN for detection and classification of tomato plant leaf diseases. *Scientific Reports*. 2025;15:16571. https://doi.org/10.1038/s41598-025-01466-5.
32) Mehta V, Gupta A. Optimized task scheduling in cloud computing environments using adaptive enhanced particle swarm optimization. *Swarm and Evolutionary Computation*. 2026;108:102510. https://doi.org/10.1016/j.swevo.2026.102510.