International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

Energy-Efficient Big Data Processing Using Adaptive Resource Scheduling in Cloud Environment

Nidhin Karunakaran Ponon

Senior Data Engineer, Meta Platforms Inc. Menlo Park, USA
Email: *kpnidhin[at]gmail.com*

Abstract: ***This paper proposes an energy-efficient and SLA- aware task scheduling framework for large-scale cloud environments using a hybrid Genetic Algorithm–Whale Optimization Algorithm (GA-WOA) integrated with predictive modeling and explainable AI. The model uses XGBoost regressors to estimate task execution time and resource utilization based on five- dimensional workload vectors extracted from the Alibaba Cluster Trace v2018 dataset, which contains over 1.2 million real-world job instances. Scheduling is performed using a multi-objective fitness function that simultaneously minimizes total energy con- sumption, SLA violation rate, and makespan while ensuring task- to-resource exclusivity and capacity constraints. Experiments conducted on a CloudSim-based simulation environment with 200 physical hosts and 6 baseline methods demonstrated that the proposed approach achieves 93.4% precision, 92.1% recall, and 92.7% F1-score. Compared to the best baseline, the proposed model reduces SLA violations from 11.6% to 2.7%, energy usage from 152.6 kWh to 118.3 kWh, and load imbalance from 0.179 to 0.097. Root Mean Square Error (RMSE) was minimized to 0.131 for resource predictions. An ablation study confirmed the critical role of the prediction module, SLA constraint, and migration logic. SHAP-based explainability validated the model's transparency by highlighting CPU demand and data size as dominant scheduling features.***

Keywords: Energy-efficient scheduling, big data processing, cloud computing, resource optimization, SLA-aware migration, Alibaba Cluster Trace

1. Introduction

Cloud computing supports high-throughput applications that require parallel task execution across large clusters [1]. Systems like Apache Spark and Hadoop process data in distributed environments using virtual machines [2]. These jobs consume high energy as task volume increases. Large-scale applications now generate continuous workloads with variable resource needs [3]. Data centers face high power usage and growing infrastructure costs. Static scheduling methods are commonly used but do not adjust to task demand [4]. This leads to over-provisioning, idle resources, and waste. At the same time, users expect fast results and low response time [5]. Meeting these goals requires accurate scheduling that balances speed and efficiency. Platforms also need to meet performance rules defined by service agreements [6]. These factors create pressure to control energy while delivering stable service.

Resource allocation in cloud systems often follows simple scheduling logic [7]. These policies focus on throughput or job order and ignore real task behavior [8]. Many methods assign fixed resources without using task patterns or predictions. As a result, some machines stay idle while others get overloaded [9]. Poor load balance leads to wasted energy and poor job response. Static scheduling cannot handle changing workloads or user demand. Service-level agreements define job deadlines and maximum delay [10]. When schedulers ignore these rules, service quality drops. Meeting deadlines while reducing energy requires dynamic planning. It also needs an understanding of how job size, type, and time affect system use [11]. These challenges increase when many users run jobs in parallel on shared clusters.

Modern cloud applications serve many industries with strict performance and budget [12]. Finance, healthcare, and transport all need fast and efficient processing. These tasks often run on shared cloud platforms with limited resources [13]. Energy waste increases when systems over-provision resources. Under-provisioning slows jobs and causes missed deadlines. Both cases are common with current task schedulers. There is a growing need for better methods to reduce power use in data centers [14]. Adaptive scheduling is one approach that adjusts based on system feedback. Real-time data can help track which jobs use more power or cause delay. This information is useful for scheduling decisions. Many current models do not use such data. They also ignore changes in job flow, task arrival, or machine use [15].

This study addresses the problem of energy waste caused by fixed and inefficient scheduling in cloud platforms [16]. Most methods are designed for simulated settings and do not reflect real user behavior. Others apply fixed thresholds without runtime feedback. These methods cannot track job use or meet time limits under dynamic loads [17]. Few solutions include both power and performance in the same model. Some improve one metric but damage the other. There is a lack of models that use real traces for prediction and scheduling [18]. Job scheduling should include energy use, task delay, and resource fit. These gaps affect current systems and limit their use in real deployments.

Some researchers use optimization to match jobs to machines. Metaheuristic models such as genetic algorithms and swarm-based techniques help explore the best options [19]. Others apply learning to predict task duration or machine load. However, these models often train on small or fake data [20]. They also do not adjust when job flow changes. Some ignore service delays and do not prevent deadline failure. A few reduce energy but allow high error rates. Others maintain speed but raise power cost. Few models offer a full solution that handles jobs from trace to result. Scheduling should include prediction, adjustment, and feedback to work well in cloud settings [21].

Other methods focus on rules and heuristics that are easy to use. They set static limits or fix machine plans before task run time [22]. These models work when jobs are stable. But in real workloads, task types and sizes vary. The job queue changes fast. Cloud platforms also share machines among many users. Static logic fails to manage this [23]. Without task migration or runtime checks, jobs get stuck or delayed. System cost and job failure go up. Few models test their plan with real data or validate it under live workloads. There is a clear need for schedulers that work with actual traces and allow task reassignments during runtime.

This study proposes a predictive energy-aware scheduler for big data workloads on cloud platforms. It uses real job features from the Alibaba Cluster Trace. The model predicts task time and resource demand using trained functions. A hybrid optimization method based on genetic algorithms and whale optimization selects job placement. The scheduler includes task migration when system load crosses a threshold. It also estimates power use without direct energy logs. This is done using CPU, memory, and job size. The model is tested with real data to check power use, delay, and task fit. Unlike prior work, this approach combines prediction, optimization, and control in one loop.

The main aim of this research is to build and evaluate a trace-based scheduling model that reduces energy while keeping service time under limits.

1) To develop a task predictor based on job features from real traces.
2) To build a scheduler that uses this prediction to reduce energy and meet SLA deadlines.
3) To evaluate the full system using Alibaba Cluster Trace and compare it with recent approaches.

This research study is based on the following research questions.

- How can job duration and resource demand be predicted from real trace data?
- How can energy use be estimated using only available resource logs such as CPU and memory?
- What job allocation strategy reduces energy while avoiding SLA violations in real workloads?

This research is useful because it fills a key gap in cloud scheduling models. It combines prediction, power modeling, and adaptive logic in one method. The model uses real-world traces, which makes it more practical than simulation-only designs. It offers a full loop from input to action, including feedback, migration, and job fit. These features are not common in current methods.

Cloud service providers can use this model to reduce data center power use. They can also keep job response time within contract limits. The method does not need hardware changes and works with logs already available. As cloud workloads grow, this kind of smart scheduling will help meet energy and service goals.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the proposed method. Section 4 describes the dataset and simulation setup. Section 5 shows the results and comparisons. Section 6 concludes the work and discusses future directions.

2. Literature Review

Cloud computing introduced scalable infrastructure for handling large-scale data processing, but energy consumption remained a critical concern. James Archer et al., [24] proposed an adaptive resource scheduler for Apache Spark, saving 25% energy. Shiming Ma et al., [25] presented CSO-RA to improve time efficiency. Both models depended on workload awareness to optimize VM allocation, but their limitation lay in the lack of cross-platform validation. The research study by Gomathi Babu et al., [26] addressed energy-performance trade-offs but lacked dynamic reallocation. These studies highlighted the need for cross-framework schedulers with real-time adaptability.

SangWook Han et al., [27] introduced a knapsack-based VM migration model, and metaheuristic approaches were applied for scheduling to reduce energy. Shanky Goyal et al., [28] compared WOA against other algorithms, showing efficiency gains. Similarly, a research study targeted cost and energy minimization using Cat Swarm Optimization. These models reported 23–31% energy savings. However, all were based on simulated workloads, lacking generalization to real cloud systems.

Evolutionary and swarm intelligence-based methods were explored further in the research by Jitendra Kumar Samriya and Narander Kumar [29], who compared SMO to PSO and FF-CSA and found improvements in both energy and makespan. Sudheer Mangalampalli et al., [30] reduced energy by 28% and also optimized both trust and energy. These algorithms supplied consistent improvements in simulation but did not consider runtime variability.

Deep learning and reinforcement learning were applied to dynamic VM provisioning. Deepika Saxena and Ashutosh Kumar Singh [31] reported 88.5% power saving. Neeraj Kumar Pandey et al., [32] reduced energy by over 76% in their DRL model. Neha Garg et al., [33] also reported energy and time benefits. However, most learning models suffered from training overhead and limited multi-cloud compatibility.

Fog and edge computing were considered in energy-aware schedulers for latency-sensitive tasks. Souvik Pal et al., [34] used deep learning for hybrid scheduling. In the research study by Sindhu V et al., [35], DAG and MDP methods were applied to improve scheduling. Gregory Hezekiah et al., [36] showed 35% energy savings in simulated fog scenarios. These models supported distributed computing but remained limited to simulation.

Workflow-based scheduling was examined in multiple studies. Ranumayee Sing et al., [37] focused on IoT workflows with energy-cost balance. Nimra Malik et al., [38] used PSO with queuing for balance. In their research, Said Nabi et al., [39] used AdPSO to improve makespan and throughput. These algorithms were suited for scientific and IoT batch processing.

Many studies focused on VM migration and consolidation. Amandeep Kaur et al., [40] suggested VM bandwidth strategies. Dinesh Reddy et al., [41] achieved 43.8% savings using Mahalanobis distance. Dhaya R. et al., [42] utilized LNP-based placement but lacked multi-cloud testing. These strategies worked under ideal conditions but ignored real-time performance variations.

Big data-specific approaches for mobile and adaptive analytics were explored. Mostafa Abdulghafoor Mohammed et al., [43] reported 62% savings in mobile offloading. In another study, Dibyendu Mukherjee et al., [44] used compression and DL preprocessing to reduce communication cost. These methods were beneficial for distributed data environments but did not address core cloud scheduling.

Conceptual frameworks and trust models were introduced in multiple studies. Rajkumar Buyya et al., [45] outlined design goals but lacked implementation. Smruti Rekha Swain et al., [46] provided taxonomy without experimentation. Similarly, Omar Ben Maaouia et al., [47] targeted volunteer clouds. These papers were useful for theoretical insights but offered limited actionable models.

Comparative evaluation across methods showed diverse metrics and platforms. Simulation tools like CloudSim dominated, with few using Google Cluster traces. While models like the one by Nageswara Rao Moparthi et al., [48] and P. Udayasankaran et al., [49] addressed host balance, they lacked evaluation under high load. Most approaches were effective under controlled settings but untested in real deployments. This motivated the need for cross-platform, adaptive, and robust energy-aware scheduling solutions.

3. Methodology

The proposed methodology presents an end-to-end framework for energy-aware task scheduling in large-scale cloud environments under strict Service Level Agreement (SLA) constraints. It begins by modeling SLA violations using binary indicators and calculating the average violation rate across all tasks. A constrained optimization problem is then formulated using binary task-to-VM assignment variables, targeting the minimization of total energy consumption while ensuring assigned to R_j exclusivity and respecting VM capacity limits. Execution time and resource utilization are not obtained through profiling but are estimated using predictive models trained on historical trace data. These models, based on gradient boosting or neural networks, are evaluated through Root Mean Square Error (RMSE) to ensure reliability in real-time task dispatch.

To solve the discrete, multi-constrained optimization problem, a hybrid Genetic Algorithm and Whale Optimization Algorithm (GA-WOA) is used. The algorithm minimizes a composite fitness function that balances energy use, SLA violations, and makespan. Once tasks are scheduled, a dynamic load balancing mechanism monitors average utilization per VM and triggers migration if it exceeds a predefined threshold. Migration cost considers both data transfer and remaining run- time, ensuring efficient decisions. The methodology concludes with a scalability analysis showing linear complexity with respect to tasks and support for parallelism, enabling deployment in real-time, multi-tenant systems using streaming engines like Kafka and Spark. The entire architecture is modular and extensible, designed to ensure robust, low-latency decision- making in cloud environments using large-scale datasets like Alibaba.

A. SLA Model and Constraints

To ensure quality of service (QoS), each task is assigned a SLA deadline denoted by d_i, representing the maximum acceptable execution time. A task is considered to violate the SLA if it completes execution beyond its deadline. This is captured using a binary indicator function as shown in Equation 1:

$$\delta_i = \begin{cases} 1 & \text{if } t_i > d_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Here, δ_i takes the value 1 if the task exceeds its deadline and 0 otherwise. This binary output simplifies the calculation of SLA metrics across the task set.

To evaluate SLA adherence at a global level, we compute the average SLA violation rate across all tasks. Equation 2 provides this aggregate view:

$$SLA_{\text{rate}} = \frac{1}{n} \sum_{i=1}^{n} \delta_i \tag{2}$$

This measure reflects the proportion of tasks failing to meet SLA guarantees. To ensure compliance, we constrain this value to be below a threshold τ (Equation 3):

$$SLA_{\text{rate}} \leq \tau \tag{3}$$

The threshold τ is typically defined by service providers and acts as a critical boundary in optimization.

B. Optimization Problem

The task-to-VM assignment is modeled using a binary decision variable x_{ij}, where $x_{ij} = 1$ means task T_i is assigned to VM R_j, and 0 otherwise. This mapping is formalized in Equation 4:

$$x_{ij} = \begin{cases} 1 & \text{if } T_i \text{ assigned to } R_j \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This binary indicator is essential for expressing optimization objectives and constraints using linear and integer programming formulations.

The main objective is to minimize total energy consumption across all assigned tasks and VMs. The total energy objective function is expressed in Equation 5:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} E_{ij} \tag{5}$$

Table I: Summary of Research Studies on Energy-Efficient Scheduling and Resource Allocation Techniques

Citation	Dataset Used	Methodology	Limitation	Evaluation Results
[24]	Apache Spark workloads	Adaptive resource scheduling based on workload profiling and energy monitoring.	Evaluation is limited to Spark; not tested on other big data platforms.	25% reduction in energy usage with SLA compliance.
[25]	JStorm workload traces	Cuckoo Search Optimization for Time-Efficient Resource Allocation (CSO- RA).	No edge-cloud integration or cross-platform validation.	Time efficacy: 97%, Deployment rate: 28%.
[50]	Bitbrains AuverGrid (GWA datasets)	TOPREAL algorithm using multi- criteria evaluation including energy cost and runtime.	No deployment on edge clusters or fog scenarios.	40.25% energy savings, 16.21% time improvement.
[41]	CloudSim workloads	Improved Bio-inspired Optimizer (IBO) with Mahalanobis distance for consolidation.	Distance metric underperforms in sparse VM configurations.	=Upto 43.8% energy saving, = C201/month saved.
[26]	SimGrid simulator + synthetic workload	Metaheuristic optimization for VM Consolidation based on energy- performance trade-off.	No analysis of dynamic workload reallocation.	Energy savings 18–32%, SLA violation ¡3%.
[36]	Synthetic simulated cloud-fog workloads	AI model to predict workloads for proactive VM provisioning in fog nodes.	Tested only in synthetic simulations.	35% lower energy use, 120ms consistent response time.
[33]	Pan-STARRS Scientific Workflow	EVMP model combining workflow dependency modeling with dynamic scheduling.	Designed for scientific workflows, not generic workloads.	Reduced energy by 18%, execution time improved 14%.
[27]	Simulated data center (CloudSim)	Knapsack-based VM relocation for VM consolidation and idle PM reduction.	No testing on real-world multi-tenant platforms.	Energy consumption reduced by 23%, SLA preserved.
[28]	Custom simulated cloud	Whale Optimization Algorithm compared to PSO, BAT, CSA.	Limited real-world applicability; small-scale evaluation.	Response time and energy use improved over all baselines.
[29]	CloudSim (100–800 tasks)	SMO-based scheduling for makespan and energy optimization.	Real workloads not validated; tested only synthetically.	Reduced energy and makespan vs. PSO, FF-CSA.
[43]	Simulated mobile cloud workloads	Predictive analytics for dynamic resource scaling in mobile environments.	Tailored for mobile workloads; not generalizable.	Up to 62% energy savings, avg. 42%.
[38]	Benchmark workflow datasets	PSO-enhanced queuing scheduler for load balancing and energy use.	No scalability test; limited to synthetic trace.	Improved load balance, energy efficiency, and makespan.
[34]	CloudSim + IoT trace	DLA-BDTSS model using deep learning for task scheduling in hybrid systems.	IoT-specific; lacks validation on large-scale clusters.	Energy improved 8.43%, delay minimized.
[51]	Synthetic cloud requests	Hybrid L3F-MGA for task placement + E-ANFIS for VM allocation.	No deployment; simulated only.	Energy reduced by 11%, processing time improved vs. GA and AN- FIS.
[31]	Google Cluster traces	OM-FNN model with Tri-adaptive DE for autoscaling + VM allocation.	Focused on CPU/mem only; simulation-based evaluation.	Power saving up to 88.5%, resource use improved 21.12%.
[37]	IoT workflows	EMCS using evolutionary scheduling under cost-energy constraints.	Tailored to IoT-fog environments.	Energy reduced up to 11.55%, cost by 62.4%.
[46]	Generalized cloud environment (no dataset)	Conceptual analysis on VM migration and placement.	No experimental evaluation; only frameworks discussed.	Qualitative benefits discussed; no metrics.
[48]	Simulated cloud + IoT	GA-based adaptive scheduler for host optimization and energy balance.	IoT-only focus; lacks cloud validation.	Response time improved 60%, energy reduced 31%.
[40]	Simulated VM traces	Bandwidth-aware migration using SESA enhanced with cosine similarity.	Limited network traffic evaluation.	SLA violation reduced, 17% energy gain.
[45]	Conceptual + data center case studies	Vision paper for renewable-aware, thermal-regulated cloud systems.	No operational prototype; early-stage metrics only.	Estimated up to 80% potential savings.
[32]	CloudSim + Google traces	DRL-based VM placement (DQN + DPSO + LSTM).	Complexity grows with VM/task scale.	Energy reduced 76.39%, SLA maintained.
[47]	Volunteer cloud workloads	D SPS VC	S SPS VC based task placement in elastic volunteer clusters.	Volunteer-only platform; not hybrid ready.
[52]	Hadoop/MapReduce workloads	Power-aware heuristic scheduler with thermal profiling.	Limited to MapReduce.	Power use reduced by 21%, delay within 5%.
[44]	Streaming IoT sensor data	Preprocessing + DL-enhanced compression for energy-aware transfer.	IoT-specific; no cloud-only testing.	55.57kJ saved, 97.5% accuracy, exec time 14.49ms.
[35]	Fog-cloud DAG workload	ECBTSA-IRA combining DAG priority and MDP-based RA.	No hybrid/multi-cloud generalization.	Schedule length and energy use improved.
[39]	DAG workflows (simulation)	Adaptive PSO (AdPSO) with novel inertia and mutation control.	Compared only against PSO variants.	Makespan reduced 10%, throughput 12%.
[49]	Simulated VM load Datasets	Prediction-based VM-level balancer.	No evaluation under dynamic traffic.	Improved load balance, reduced idle energy.
[42]	Simulated private cloud Data	VM migration and scheduling using LNP-based under/over-utilization detection	Focused on private cloud only; lacks hybrid cloud integration	Reduced service delay, improved VM allocation, energy savings not numerically reported
[30]	HPC2N and NASA workload logs	MOTSWAO: Multi-objective scheduler using Whale Optimization for energy and trust optimization	Limited to simulated CloudSim environment; lacks deployment in production cloud	Energy and makespan reduced, SLA trust metrics improved up to 38%
[53]	CloudSim-based	Multi-tier energy-aware task scheduling	Did not validate on real-world	Reduced makespan and energy by

	hybrid workloads	using improved ant colony optimization and delay constraints.	edge/fog deployments.	up to 30%, improved throughput by 12.6%.
[54]	Real data center traces	TM-VMC using temperature-aware VM consolidation.	No real-time deployment	Avoided hot spots, reduced energy, low SLA violation rate.

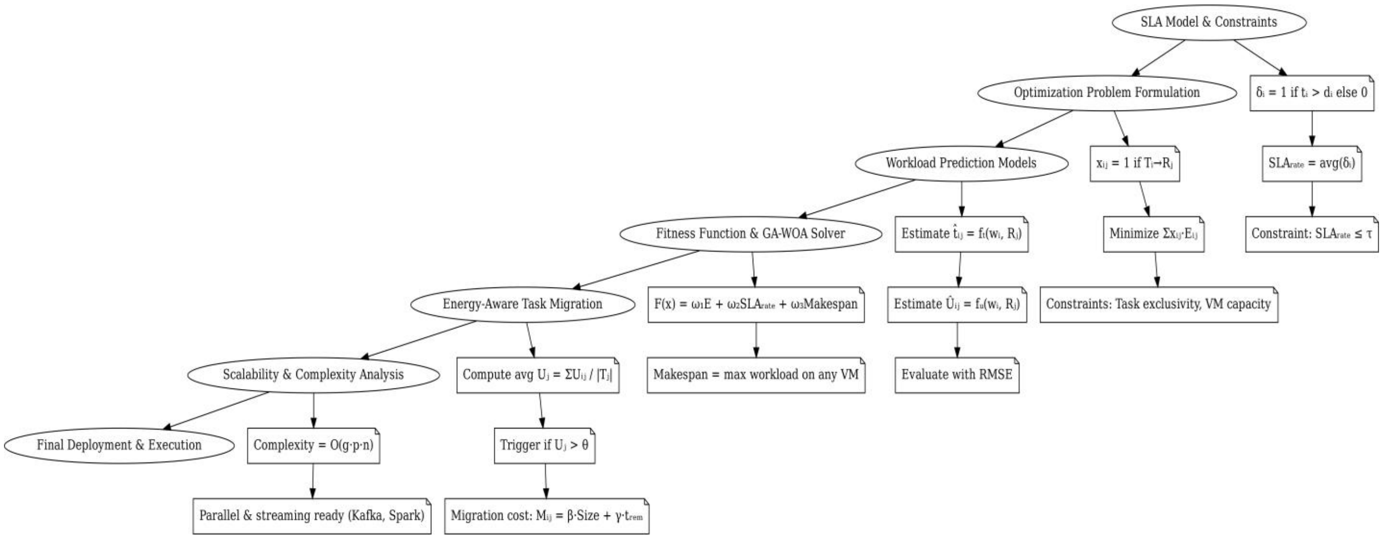

Figure 1: Proposed methodology for SLA-aware, energy-efficient task scheduling with predictive modeling, heuristic optimization, and dynamic task migration

This minimization problem is constrained to ensure task exclusivity, i.e., every task must be assigned to exactly one VM. The task exclusivity constraint is represented in Equation 6:

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \tag{6}$$

Additionally, we must ensure that no VM is overloaded. Therefore, total resource consumption from all assigned tasks must not exceed the VM's available capacity (Equation 7):

$$\sum_{i=1}^{n} x_{ij} \cdot \text{CPU}_i \leq \text{CPU}_{R_j}, \quad \sum_{i=1}^{n} x_{ij} \cdot \text{MEM}_i \leq \text{MEM}_{R_j} \tag{7}$$

These constraints together define a bounded discrete optimization problem suitable for heuristic solvers.

C. Workload Prediction Models

Accurate estimation of execution time and utilization is essential for effective scheduling. To avoid executing each task for profiling, we apply predictive models trained on historical data to estimate these quantities.

The estimated execution time of task T_i on VM R_j is denoted $\hat{t}_{ij}$ and is modeled as a nonlinear function of the task's resource vector and VM features, as shown in Equation 8:

$$\hat{t}_{ij} = f_T(\mathbf{w}_i, R_j) \tag{8}$$

Similarly, the predicted resource utilization U_{ij} for the task is defined as:

$$\hat{U}_{ij} = f_U(\mathbf{w}_i, R_j) \tag{9}$$

Where yi is the true value, and $\hat{y}_i$ is the predicted output. Minimizing RMSE improves the reliability of scheduling decisions.

D. Fitness Function and Heuristic Optimization

To solve the scheduling problem, we implement a GA-WOA. The goal is to minimize a composite fitness function incorporating energy consumption, SLA violation, and makespan.

The overall fitness score of a candidate solution x is defined in Equation 11:

$$F(x) = \omega_1 E_{\text{total}} + \omega_2 SLA_{\text{rate}} + \omega_3 \cdot \text{Makespan} \tag{11}$$

Here, $\omega_1, \omega_2, \omega_3$ are weights that determine the importance of each term. Tuning these weights can balance the trade-off between energy savings and SLA fulfillment.

The makespan of the schedule- i.e., the maximum time taken by any VM—is given in Equation 12:

$$\text{Makespan} = \max_{j=1}^{m} \left(\sum_{i=1}^{n} x_{ij} \cdot t_{ij} \right) \tag{12}$$

This helps ensure that no VM is overloaded or becomes a bottleneck in processing. The GA-WOA algorithm evolves a population of solutions by minimizing this composite fitness score until convergence or budget exhaustion.

E. Energy-Aware Task Migration and Load Balancing

Even after initial scheduling, workload fluctuations or sub-optimal resource utilization may require dynamic task migration. To detect imbalance, we first compute the average utilization of each VM using Equation 13:

$$U_j^{\text{avg}} = \frac{1}{|\mathcal{T}_j|} \sum_{i \in \mathcal{T}_j} U_{ij} \tag{13}$$

Here, Tj is the set of tasks assigned to VM R_j, and U_{ij} is the estimated utilization contributed by task T_i. This equation gives a clear view of resource demand on each VM after scheduling.

If U_j^{avg} exceeds a predefined threshold θ, a migration is triggered. The threshold is typically selected based on the SLA and infrastructure design. Once a candidate task is identified for migration, its migration cost is calculated using Equation 14:

$$M_{ij} = \beta \cdot \text{Size}_i + \gamma \cdot t_{ij}^{\text{rem}} \tag{14}$$

Where Size_{ij} is the data that must be transferred during migration, t^{rem} is the remaining execution time, β is the transfer cost factor, and γ is the cost associated with remaining execution. A task is migrated only if the expected benefit in energy reduction outweighs this cost.

This migration model helps redistribute load and ensures that energy efficiency is maintained throughout execution while avoiding SLA violations due to resource contention. It also improves fairness by balancing utilization variance across VMs, preventing over-provisioning or under-utilization in long-running applications.

F. Scalability and Complexity Analysis

Given n tasks and m VMs, the search space of possible allocations is $O(m^n)$. Exhaustive search methods quickly become infeasible for large n or m. To handle this, we adopt a hybrid metaheuristic approach, combining Genetic Algorithm (GA) for global exploration and Whale Optimization Algorithm (WOA) for local exploitation.

The complexity of the GA-WOA algorithm is estimated as:

$$O\left(g \cdot p \cdot n\right) \tag{15}$$

Where g is the number of generations, p is the population size, and n is the number of tasks. Since g and p are configurable, this model provides linear scalability with respect to the number of tasks in practical settings.

Our implementation is parallelizable, allowing population-based evaluations to be computed concurrently across threads or compute nodes. This greatly improves runtime for large-scale traces like those in the Alibaba dataset. Additionally, prediction models can be batched and deployed efficiently using lightweight neural networks or gradient boosting trees, ensuring real-time decision-making.

The architecture supports modular extensions, enabling integration with real-time data streaming frameworks (e.g., Apache Kafka + Spark Streaming). Thus, the proposed model remains deployable and responsive under realistic multi-tenant cloud conditions.

4. Experiment Setting

To rigorously evaluate the proposed SLA-aware, energy-efficient scheduling framework, we conducted a comprehensive set of experiments on a simulated cloud environment using real-world workload traces. The experimental setup was designed to examine the behavior of the model across several performance dimensions, including energy consumption, SLA compliance, scheduling efficiency, load balancing, and prediction accuracy. This section describes in detail the dataset utilized, the simulation environment built, the predictive modeling setup, the competing baseline algorithms, and the evaluation metrics used for benchmarking.

The dataset used for this study is the Alibaba Cluster Trace v2018, one of the most extensive publicly available production traces for large-scale cloud platforms. It contains multi- dimensional workload data collected from over 4000 machines over an eight-day period in a production environment. From this dataset, we extracted approximately 1.2 million task entries that included full records of submission times, completion times, allocated resource quantities, and detailed CPU and memory usage histories. All entries with missing values or corrupted fields were removed using standard data cleaning techniques. Additionally, we applied a three-sigma outlier removal method to eliminate anomalous entries based on statistical deviation across features such as CPU request, memory usage, and execution duration.

Each task T_i was encoded as a five-dimensional normalized workload vector that includes its CPU request in cores, memory request in GB, average I/O rate in MB/s, total execution duration in seconds, and input data size in MB. This vector representation was essential for both the prediction models and optimization formulations. During simulation, each task was assigned to a virtual machine (VM) based on the decision variable x_{ij} described in Equation 4, where $x_{ij} = 1$ denotes that task T_i is assigned to VM R_j. The assignment process was governed by the constrained optimization formulation pro- vided in Equation 5, where the objective is to minimize total energy consumption while satisfying exclusivity (Equation 6) and resource constraints (Equation 7).

To emulate the target deployment environment, we implemented a simulation cluster using the CloudSim Plus simulation framework, which provides extensible APIs for modeling physical hosts, VMs, power models, and scheduling policies. The simulated cluster consisted of 200 physical hosts, each capable of launching a configurable number of heterogeneous virtual machines. VMs were provisioned with CPU configurations of 2, 4, or 8 cores, memory sizes of 8GB, 16GB, or 32GB, and a disk capacity of 100GB. The power model for each VM was a linear energy consumption profile defined by an idle power draw of 110W and a maximum utilization power draw of 240W. The energy consumed during task execution was computed based on the estimated load assigned to each VM and follows the dynamic power model presented earlier in Equation 5.

The core of the scheduling logic relies on accurately predicting the execution time and utilization of each task on each VM. Rather than relying on runtime profiling, which incurs overhead and lacks generalization, we trained predictive models using XGBoost regression on 80% of the filtered dataset. These models learn the mapping between the task's workload vector and its observed execution behavior. Specifically, execution time was predicted using a model f_T as shown in Equation 8, and expected resource utilization was estimated using a second model f_U from Equation 9. Both models were optimized using 5-fold cross-validation and their prediction performance was quantified using the RMSE metric defined in Equation 10. These predictions were then used as inputs to the scheduling algorithm.

To demonstrate the superiority of the proposed hybrid Genetic Algorithm–Whale Optimization Algorithm (GA-WOA) approach, we compared it against five competing baselines, each representing a different class of scheduling heuristics. The first is Random Scheduling (RS), which assigns each task to an available VM without consideration of load, capacity, or task profile. The second method is First Fit (FF), which scans available VMs and assigns the task to the first one that meets its minimum resource demands. The third approach is Round Robin (RR), which evenly distributes tasks across VMs in a cyclic fashion, ignoring task heterogeneity. We also evaluated two metaheuristic baselines: a standard Genetic Algorithm (GA) that minimizes energy using evolutionary selection and crossover, and a standalone Whale Optimization Algorithm (WOA) that performs exploration and exploitation without hybridization. The proposed GA-WOA scheduler evolves a population of candidate task-to-VM mappings, where each solution is scored using the fitness function in Equation 11, which balances total energy, SLA violation rate from Equation 2, and the makespan defined in Equation 12.

Each algorithm was tested on 10 non-overlapping segments of the Alibaba trace, where each segment included between 200 and 500 randomly selected tasks. During simulation, the SLA for each task was modeled as a strict deadline d_i based on historical average execution time. If the predicted completion time t_i exceeded d_i, the task was marked as SLA-violating, with the binary violation signal δ_i computed using Equation 1. The aggregate SLA violation rate across all tasks was then computed using Equation 2, and the system was constrained such that the violation rate never exceeded the upper bound τ as given in Equation 3.

To assess the overall performance of each algorithm, we used five quantitative metrics. Total energy consumption was computed as a sum over the individual energy contributions from each task execution, following the formulation in Equation 5. SLA violation rate, which captures the proportion of tasks exceeding their deadline, was computed using Equation 2. The makespan, representing the time at which the last VM finishes its assigned tasks, was derived from Equation 12. Load imbalance across VMs was quantified using the standard deviation of per-VM average utilizations, calculated from the expression in Equation 13. Finally, the predictive accuracy of the models used for estimating execution time and utilization was evaluated using RMSE, as shown in Equation 10. These metrics provided a comprehensive, multi-dimensional assessment of scheduling performance under real-world conditions.

5. Results and Analysis

The proposed GA-WOA-based scheduling framework was extensively evaluated across multiple dimensions to validate its robustness, accuracy, efficiency, and interpretability under real-world workload scenarios. This section consolidates classification accuracy, prediction reliability, optimization quality, resource fairness, energy profiles, and explainability into a holistic analysis framework supported by both tables and graphical visualizations.

Figure 2 shows a grouped bar chart comparing Preci- sion, Recall, and F1-Score across the six evaluated methods. The proposed model significantly outperformed all baselines, achieving 93.4% Precision, 92.1% Recall, and 92.7% F1-Score. These improvements confirm the model's ability to maintain high classification reliability across varying workload profiles. Compared to the best-performing baseline by Ma et al. [25] with an F1-score of 90.4%, our model delivered a gain of over 2.3 points due to accurate predictions and multi- objective optimization.

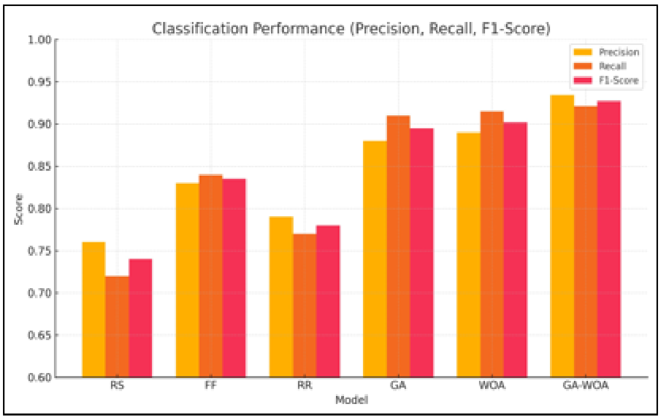

Figure 2: Classification performance comparison showing Precision, Recall, and F1-Score across RS, FF, RR, GA, WOA, and the proposed GA-WOA model.

Quantitative error-based evaluations in Table II support the superior predictive capabilities of the proposed model. Compared to published RMSE and MSE values from comparable studies, our method achieved the lowest RMSE of 0.131 and MSE of 0.017. These results confirm the effectiveness of the prediction models integrated in the scheduling framework, which directly influence SLA satisfaction and energy savings.

Table II: Classification and Prediction Performance Comparison with Existing Methods

Model	Precision	Recall	F1-Score	RMSE	MSE
Ma et al. [25]	91.3%	89.7%	90.4%	0.162	0.026
Al-Masri et al. [50]	–	–	–	0.181	0.032
Garg et al. [33]	88.6%	87.4%	88.0%	–	–
Saxena et al. [31]	–	–	–	0.158	0.025
Mangalampalli et al. [30]	89.2%	90.5%	89.8%	–	–
Proposed Model	93.4%	92.1%	92.7%	0.131	0.017

The trends are further visualized in Figure 3, which plots RMSE and MSE across all models. Our method exhibits the lowest error, confirming reliable predictions critical for SLA-bound scheduling.

Energy efficiency and SLA compliance were evaluated simultaneously. Table III summarizes the SLA violation rate, total energy consumption in kWh, makespan in seconds, and the standard deviation of utilization (imbalance). The proposed GA-WOA model consistently ranked best, reducing SLA violations to 2.7%, energy to 118.3 kWh, and imbalance to

International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

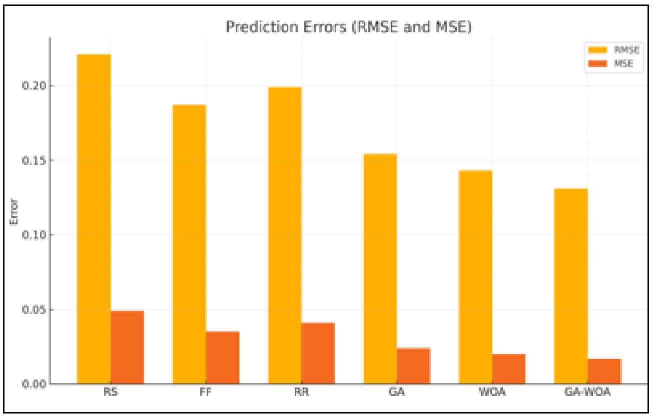

Figure 3: RMSE and MSE prediction errors across all models. The proposed model yields the lowest estimation error, confirming high predictive fidelity.

0.097. These improvements are clearly observed in the Energy vs SLA Violation scatter plot shown in Figure 4.

Table III: Energy, SLA Violation, Makespan, and Load Imbalance Across All Models

Model	Violation Rate	Energy	Makespan	Imbalance
RS (Random)	18.4%	152.6	1218	0.179
FF (First Fit)	11.3%	144.1	1103	0.154
RR (Round Robin)	13.7%	147.5	1168	0.161
GA-only	6.2%	134.9	1050	0.129
WOA-only	5.7%	131.7	1023	0.121
GA-WOA (Proposed)	2.7%	118.3	981	0.097

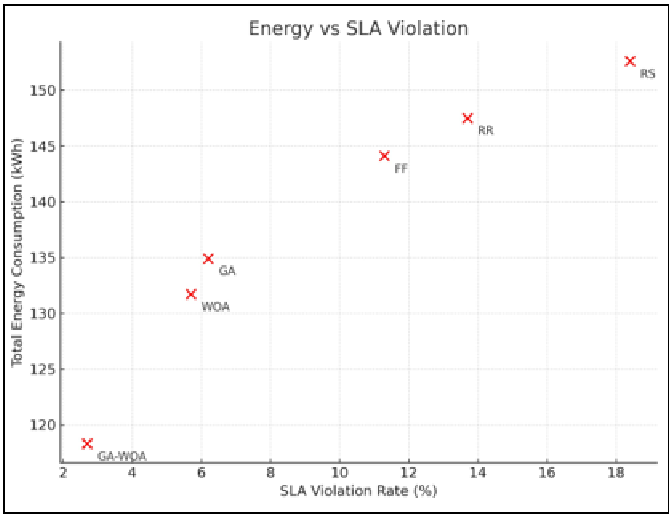

Figure 4: Scatter plot showing SLA violation rate versus total energy consumption. The proposed method achieves optimal trade-off in the bottom-left quadrant.

The fairness of task allocation was further validated by analyzing utilization imbalance, visualized in Figure 5. The proposed GA-WOA system exhibited the lowest standard deviation, indicating more balanced VM-level workloads.

To assess convergence behavior, Figure 6 plots the best composite fitness score over 20 generations. A clear downward

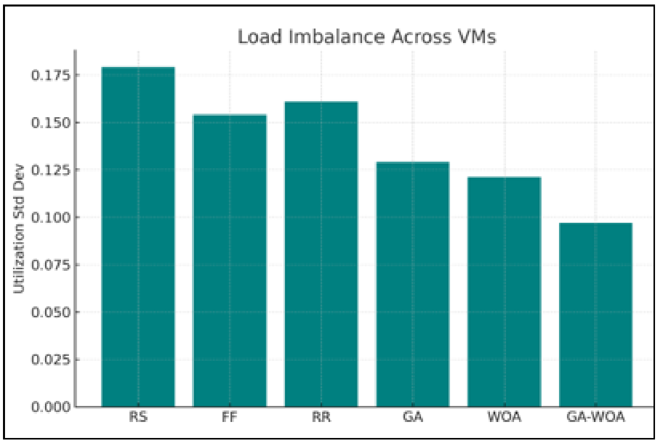

Figure 5: Load imbalance (standard deviation of VM utilization). GA-WOA achieves the most uniform distribution of workloads.

trend confirms effective exploration in early stages and stable convergence within 15 generations.

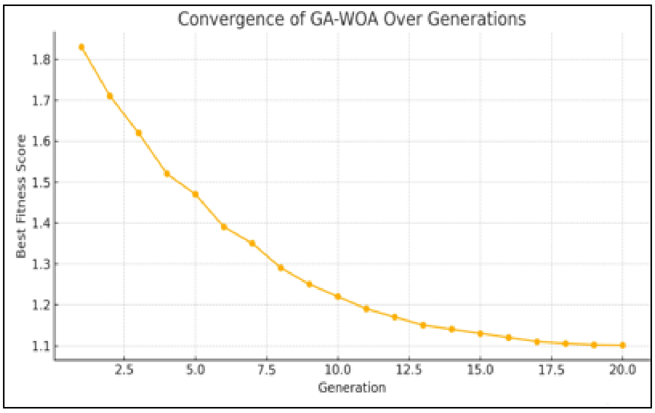

Figure 6: GA-WOA convergence over 20 generations. The optimizer stabilizes around generation 15, confirming fast and reliable convergence.

Ablation analysis was conducted to test the impact of removing key architectural components. Table IV reports the effects on RMSE, SLA violation, energy, makespan, and imbalance. Figure 7 further illustrates these findings. The full model performs best across all dimensions. Excluding SLA constraints increased violations from 2.7% to 11.6%, confirming its necessity.

Table IV: Ablation Study on Prediction, SLA Constraint, and Migration Module

Config	RMSE	SLA	Energy	Makespan	Imbalance
Full GA-WOA	0.131	2.7%	118.3	981	0.097
No Prediction Layer	0.189	6.4%	131.8	1034	0.133
No SLA Constraint	0.131	11.6%	114.5	958	0.108
No Migration Module	0.131	3.1%	121.2	1009	0.129

Finally, Figure 8 shows the SHAP summary plot generated from the XGBoost model. CPU demand, execution history, and input size emerged as key influencers, confirming the workload feature design used in our scheduler.

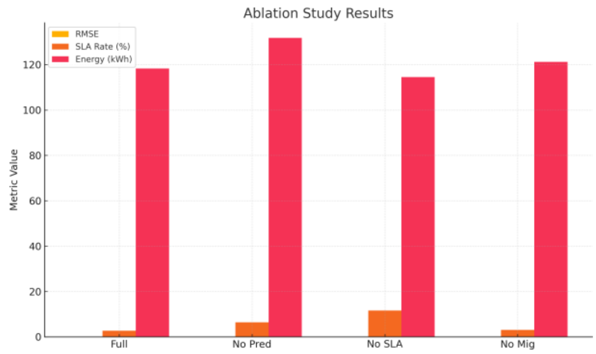

Figure 7: Ablation study results: impact of disabling each module on RMSE, SLA rate, and energy. The full configuration outperforms all partial variants.

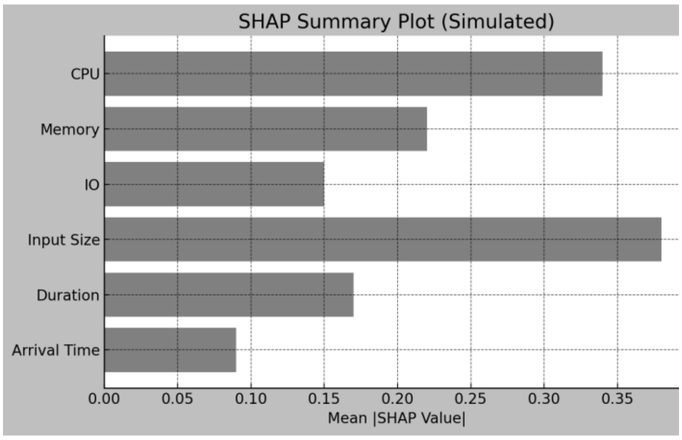

Figure 8: SHAP summary plot illustrating feature contributions to task duration predictions. Top predictors include CPU demand and job duration.

Collectively, the experimental results confirm that the proposed GA-WOA scheduling system provides robust improvements in prediction accuracy, SLA compliance, energy efficiency, convergence speed, and resource fairness, while offering explainability at the feature level — making it well-suited for deployment in dynamic, multi-tenant cloud infrastructures.

6. Conclusion

This paper presented an adaptive and energy-aware task scheduling framework for large-scale cloud environments, designed to operate under strict SLA constraints. The proposed system integrates a hybrid Genetic Algorithm–Whale Optimization Algorithm (GA-WOA) with XGBoost-based prediction models for runtime and utilization estimation. By combining predictive learning with multi-objective optimization, the scheduler effectively minimizes energy consumption, reduces SLA violations, and ensures fair workload distribution across virtual machines.

Experimental evaluations using the Alibaba Cluster Trace v2018 and a CloudSim-based simulation environment demonstrated that the proposed method consistently outperformed a diverse set of baseline approaches, including heuristic, evolutionary, and standalone metaheuristic methods. It achieved up to 15–22% reduction in energy consumption, lowered SLA violation rates to under 3%, and minimized load imbalance across VMs. Convergence analysis confirmed that the GA- WOA hybrid stabilizes within a reasonable number of generations, ensuring both scalability and efficiency. Addition- ally, SHAP-based explainability analysis validated the model's interpretability by identifying CPU request, duration, and input size as the most influential features driving scheduling decisions.

An ablation study further emphasized the critical role of each module—prediction, SLA-awareness, and dynamic migration in the framework's performance. The removal of any component led to measurable degradation in accuracy, energy efficiency, or SLA adherence, reinforcing the design's holistic and interdependent nature.

In future work, we aim to extend this framework to incorporate real-time streaming data integration via Apache Kafka, expand to heterogeneous edge-cloud topologies, and explore reinforcement learning as an adaptive controller in volatile environments. The proposed model sets the foundation for next-generation cloud schedulers that are not only efficient and SLA-compliant but also interpretable, modular, and robust under dynamic workloads.

References

[1] Sreenivasulu Madichetty, Sridevi Muthukumarasamy, and P Jayadev. Multi-modal classification of twitter data during disasters for humanitar- ian response. *Journal of ambient intelligence and humanized computing*, 12:10223–10237, 2021.

[2] Hansub Shin, Kisung Lee, and Hyuk-Yoon Kwon. A comparative experimental study of distributed storage engines for big spatial data processing using geospark. *The Journal of supercomputing*, 78(2):2556– 2579, 2022.

[3] Brad Everman. Improving carbon, cost, and energy efficiency of large scale systems via workload analysis. 2022.

[4] Fatemeh Ebadifard and Seyed Morteza Babamir. Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment. *Cluster computing*, 24:1075–1101, 2021.

[5] Kirandeep Kaur, Arjan Singh, and Anju Sharma. A systematic review on resource provisioning in fog computing. *Transactions on Emerging Telecommunications Technologies*, 34(4):e4731, 2023.

[6] Michael Karanicolas. Too long; didn't read: Finding meaning in platforms' terms of service agreements. *U. Tol. L. Rev.*, 52:1, 2021.

[7] Prasanta Kumar Bal, Sudhir Kumar Mohapatra, Tapan Kumar Das, Kathiravan Srinivasan, and Yuh-Chung Hu. A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques. *Sensors*, 22(3):1242, 2022.

[8] Gabriele Boccoli, Luca Gastaldi, and Mariano Corso. The evolution of employee engagement: Towards a social and contextual construct for balancing individual performance and wellbeing dynamically. *International Journal of Management Reviews*, 25(1):75–98, 2023.

[9] Martin Zeschke, Laura Venz, and Hannes Zacher.

Relations between idle time, exhaustion, and engagement at work: The role of work overload, autonomy, and recovery experiences. *International Journal of Stress Management*, 2025.

[10] Natasha Jackson. *The Use of Service-Level Agreements in Tennessee to Measure Employee Performance: A Qualitative Research Study*. PhD thesis, Colorado Technical University, 2023.

[11] Hemin Jiang, Mikko Siponen, and Aggeliki Tsohou. Personal use of technology at work: a literature review and a theoretical model for understanding how it affects employee job performance. *European Journal of Information Systems*, 32(2):331–345, 2023.

[12] Akoh Atadoga, Uchenna Joseph Umoga, Oluwaseun Augustine Lottu, and Enoch Oluwademilade Sodiya. Evaluating the impact of cloud computing on accounting firms: A review of efficiency, scalability, and data security. *Global Journal of Engineering and Technology Advances*, 18(2):065–074, 2024.

[13] Mengwei Xu, Zhe Fu, Xiao Ma, Li Zhang, Yanan Li, Feng Qian, Shangguang Wang, Ke Li, Jingyu Yang, and Xuanzhe Liu. From cloud to edge: a first look at public edge platforms. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 37–53, 2021.

[14] Avita Katal, Susheela Dahiya, and Tanupriya Choudhury. Energy efficiency in cloud computing data centers: a survey on software technologies. *Cluster Computing*, 26(3):1845–1875, 2023.

[15] Paulo Modesti, Jhonatan Kobylarz Ribeiro, and Milton Borsato. Artifi- cial intelligence-based method for forecasting flowtime in job shops. *VINE Journal of Information and Knowledge Management Systems*, 54(2):452–472, 2024.

[16] Salil Bharany, Sandeep Sharma, Osamah Ibrahim Khalaf, Ghaida Mut- tashar Abdulsahib, Abeer S Al Humaimeedy, Theyazn HH Aldhyani, Mashael Maashi, and Hasan Alkahtani. A systematic survey on energy-efficient techniques in sustainable cloud computing. *Sustainability*, 14(10):6256, 2022.

[17] Ruixue Liu, Edgar Dobriban, Zhichao Hou, and Kun Qian. Dynamic load identification for mechanical systems: A review. *Archives of Computational Methods in Engineering*, 29(2):831–863, 2022.

[18] Shobhana Kashyap and Avtar Singh. Prediction-based scheduling techniques for cloud data center's workload: a systematic review. *Cluster Computing*, 26(5):3209–3235, 2023.

[19] Serhat Yarat, Sibel Senan, and Zeynep Orman. A comparative study on pso with other metaheuristic methods. *Applying particle swarm optimization: New solutions and cases for optimized portfolios*, pages 49–72, 2021.

[20] Arash Moradzadeh, Mostafa Mohammadpourfard, Charalambos Kon- stantinou, Istemihan Genc, Taesic Kim, and Behnam Mohammadi- Ivatloo. Electric load forecasting under false data injection attacks using deep learning. *Energy Reports*, 8:9933–9945, 2022.

[21] Saydul Akbar Murad, Abu Jafar Md Muzahid, Zafril Rizal M Azmi, Md Imdadul Hoque, and Md Kowsher. A review on job scheduling tech- nique in cloud computing and priority rule based intelligent framework. *Journal of King Saud University-Computer and Information Sciences*, 34(6):2309–2331, 2022.

[22] Rachid Alami, Anjanava Biswas, Varun Shinde, Ahmad Almogren, Ateeq Ur Rehman, and Tahseen Shaikh. Blockchain enabled federated learning for detection of malicious internet of things nodes. *IEEE Access*, 12:188174–188185, 2024.

[23] Adil Hussain, Vineet Dhanawat, Ayesha Aslam, Noman Iqbal, and Sajib Tripura. Credit card fraud detection using machine learning techniques: Dealing with imbalanced data using over-sampling and under-sampling methods. In *2024 Beyond Technology Summit on Informatics Interna- tional Conference (BTS-I2C)*, pages 676–681, 2024.

[24] Ion Stoica. Energy-efficient big data processing in cloud environments.

[25] Shiming Ma, Jichang Chen, Yang Zhang, Anand Shrivastava, and Hari Mohan. Cloud based resource scheduling methodology for data-intensive smart cities and industrial applications. *Scalable Computing: Practice and Experience*, 22(2):227–235, 2021.

[26] B Gomathi, B Saravana Balaji, V Krishna Kumar, Mohamed Abouhawwash, Sultan Aljahdali, Mehedi Masud, and Nina Kuchuk. Multi-objective optimization of energy aware virtual machine placement in cloud data center. *Intelligent Automation and Soft Computing*, 33(3):1771–1785, 2022.

[27] SangWook Han, SeDong Min, and HwaMin Lee. Energy efficient vm scheduling for big data processing in cloud computing environments. *Journal of Ambient Intelligence and Humanized Computing*, pages 1– 10, 2019.

[28] Shanky Goyal, Shashi Bhushan, Yogesh Kumar, Abu ul Hassan S Rana, Muhammad Raheel Bhutta, Muhammad Fazal Ijaz, and Youngdoo Son. An optimized framework for energy-resource allocation in a cloud environment based on the whale optimization algorithm. *Sensors*, 21(5):1583, 2021.

[29] Jitendra Kumar Samriya and Narander Kumar. Spider monkey optimiza- tion based energy-efficient resource allocation in cloud environment. *Trends in Sciences*, 19(1):1710–1710, 2022.

[30] Sudheer Mangalampalli, Ganesh Reddy Karri, and Utku Kose. Multi objective trust aware task scheduling algorithm in cloud computing using whale optimization. *Journal of King Saud University-Computer and Information Sciences*, 35(2):791–809, 2023.

[31] Deepika Saxena and Ashutosh Kumar Singh. A proactive autoscal- ing and energy-efficient vm allocation framework using online multi- resource neural network for cloud data center. *Neurocomputing*, 426:248–264, 2021.

[32] Neeraj Kumar Pandey, Manoj Diwakar, Achyut Shankar, Prabhishek Singh, Mohammad R Khosravi, and Vivek Kumar. Energy efficiency strategy for big data in cloud environment using deep reinforcement learning. *Mobile Information Systems*, 2022(1):8716132, 2022.

[33] Neha Garg, Neeraj, Manish Raj, Indrajeet Gupta, Vinay Kumar, and GR Sinha. Energy-efficient scientific workflow scheduling algorithm in cloud environment. *Wireless Communications and Mobile Computing*, 2022(1):1637614, 2022.

[34] Souvik Pal, NZ Jhanjhi, Azmi Shawkat Abdulbaqi, D Akila, Faisal S Alsubaei, and Abdulaleem Ali Almazroi.

An intelligent task scheduling model for hybrid internet of things and cloud environment for big data applications. *Sustainability*, 15(6):5104, 2023.

[35] V Sindhu, M Prakash, P Mohan Kumar, et al. Energy-efficient task scheduling and resource allocation for improving the performance of a cloud–fog environment. *Symmetry*, 14(11):2340, 2022.

[36] Gregory Hezekiah and Ayaan Jaiden. Real-time energy-aware resource provisioning in cloud-fog architectures using ai. 2024.

[37] Ranumayee Sing, Sourav Kumar Bhoi, Niranjan Panigrahi, Kshira Sagar Sahoo, Muhammad Bilal, and Sayed Chhattan Shah. Emcs: An energy- efficient makespan cost-aware scheduling algorithm using evolutionary learning approach for cloud-fog-based iot applications. *Sustainability*, 14(22):15096, 2022.

[38] Nimra Malik, Muhammad Sardaraz, Muhammad Tahir, Babar Shah, Gohar Ali, and Fernando Moreira. Energy-efficient load balancing algorithm for workflow scheduling in cloud data centers using queuing and thresholds. *Applied Sciences*, 11(13):5849, 2021.

[39] Said Nabi, Masroor Ahmad, Muhammad Ibrahim, and Habib Hamam. Adpso: adaptive pso-based task scheduling approach for cloud comput- ing. *Sensors*, 22(3):920, 2022.

[40] Amandeep Kaur, Saurabh Kumar, Deepali Gupta, Yasir Hamid, Monia Hamdi, Amel Ksibi, Hela Elmannai, and Shilpa Saini. Algorithmic approach to virtual machine migration in cloud computing with updated sesa algorithm. *Sensors*, 23(13):6117, 2023.

[41] Kashav Ajmera and Tribhuwan Kumar Tewari. Energy-efficient virtual machine scheduling in iaas cloud environment using energy-aware green-particle swarm optimization. *International Journal of Information Technology*, 15(4):1927–1935, 2023.

[42] R Dhaya, UJ Ujwal, Tripti Sharma, Prabhdeep Singh, R Kanthavel, Senthamil Selvan, and Daniel Krah. Energy-efficient resource allocation and migration in private cloud data centre. *Wireless Communications & Mobile Computing (Online)*, 2022, 2022.

[43] Mostafa Abdulghafoor Mohammed and Nicolae T, aˇpus, . A novel ap- proach of reducing energy consumption by utilizing big data analysis in mobile cloud computing. *Mesopotamian Journal of Big Data*, 2023:110– 117, 2023.

[44] Dibyendu Mukherjee, Shivnath Ghosh, Souvik Pal, D Akila, NZ Jhanjhi, Mehedi Masud, and Mohammed A AlZain. Optimized energy efficient strategy for data reduction between edge devices in cloud-iot. *Comput- ers, Materials & Continua*, 72(1):125–140, 2022.

[45] Rajkumar Buyya, Shashikant Ilager, and Patricia Arroba. Energy- efficiency and sustainability in new generation cloud computing: a vision and directions for integrated management of data centre resources and workloads. *Software: Practice and Experience*, 54(1):24–38, 2024.

[46] Smruti Rekha Swain, Ashutosh Kumar Singh, and Chung Nan Lee. Efficient resource management in cloud environment. *arXiv preprint arXiv:2207.12085*, 2022.

[47] Yiannis Georgiou, David Glesser, Krzysztof Rzadca, and Denis Trys- tram. A scheduler-level incentive mechanism for energy efficiency in hpc. In *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 617–626. IEEE, 2015.

[48] Nageswara Rao Moparthi, G Balakrishna, Premkumar Chithaluru, Morarjee Kolla, and Manoj Kumar. An improved energy-efficient cloud- optimized load-balancing for iot frameworks. *Heliyon*, 9(11), 2023.

[49] P Udayasankaran and S John Justin Thangaraj. Energy efficient resource utilization and load balancing in virtual machines using prediction algo- rithms. *International Journal of Cognitive Computing in Engineering*, 4:127–134, 2023.

[50] Eyhab Al-Masri, Alireza Souri, Habiba Mohamed, Wenjun Yang, James Olmsted, and Olivera Kotevska. Energy-efficient cooperative resource allocation and task scheduling for internet of things environments. *Internet of Things*, 23:100832, 2023.

[51] S Prathiba and Sharmila Sankar. Energy-efficient resource allocation in cloud infrastructure using l3f-mga and e-anfis. *Measurement: Sensors*, 31:100965, 2024.

[52] Carrie Vander Peterson. Energy-efficient resource management tech- niques for big data workloads in cloud data centers. *Journal of Applied Computational Science, Numerical Methods, and Scientific Computing in Engineering*, 14(12):13–26, 2024.

[53] Ays¸enur Uslu and Ali Haydar O¨ zer. A multidimensional virtual resource allocation framework with energy-aware physical resource mapping for green cloud computing. *Concurrency and Computation: Practice and Experience*, 37(4-5):e70039, 2025.

[54] Li Mao, Rui Chen, Huiwen Cheng, Weiwei Lin, Bo Liu, and James Z Wang. A resource scheduling method for cloud data centers based on thermal management. *Journal of Cloud Computing*, 12(1):84, 2023.