International Journal of Sustainable Intelligent Technologies
ISSN: 3139-1249
IJSIT-01

GREEN ARTIFICIAL INTELLIGENCE FOR INTELLIGENT RESOURCE MANAGEMENT

[#1]**KOLAGANI SOWJANYA**, *Assistant professor, Department of CSE(AI&ML)*
[#2]**K.SHILPIKA**, *Assistant Professor, Dept of CSE,*
[#3]**Dr.K SRIDHAR,** Associate professor, Department of CSE
[#1,2]Vaageswari College of Engineering, Karimnagar, TG.
[#3]Malla Reddy(MR) Deemed to be University, Medchal,Telangana
[#1]Kamunisowjanyasanthosh@gmail.com
[#2]shilpika.kanaparthi@gmail.com
[#3]sridhark529reddy@gmail.com

ABSTRACT: Artificial intelligence workloads and cloud services require increasingly sophisticated resource allocation, yet conventional scheduling policies may optimize utilization or response time without explicitly accounting for energy consumption and carbon intensity. This paper proposes a Green Artificial Intelligence framework for intelligent resource management that combines workload prediction, energy-aware optimization, service-level constraints, and runtime feedback. The framework estimates near-term demand from system telemetry and selects resource allocations that balance performance, energy use, and operational cost. A carbon-aware objective can incorporate time-varying grid carbon intensity where trustworthy data are available. The proposed evaluation uses trace-driven experiments and a reproducible comparison against static allocation, utilization-based autoscaling, and energy-unaware scheduling. Key measures include energy per completed task, average and tail latency, SLA violation rate, resource utilization, and estimated carbon emissions. A public cloud workload trace and a documented power-measurement model are recommended for implementation in Python. Because no implementation, trace processing, or experiments have been executed for this manuscript, all quantitative results are left pending.

Index Terms: *Green AI, energy-aware scheduling, resource management, cloud computing, workload prediction, carbon-aware computing.*

I. INTRODUCTION

Cloud and edge platforms support data analytics, web services, artificial intelligence, scientific computing, and enterprise applications. Their operators must allocate compute, memory, storage, and network resources while maintaining acceptable latency, throughput, and availability. As demand fluctuates, inefficient over-provisioning can waste energy, whereas aggressive consolidation can cause queuing delays and service-level agreement (SLA) violations.

Green AI concerns the environmental cost of developing and operating AI systems, including the energy required for computation and the emissions associated with electricity consumption. In resource management, AI can help forecast demand and select resource allocations; however, a predictive model is not inherently energy efficient. Its value must be

DOI : https://doi.org/10.65713/ijsitv2i401

demonstrated against transparent baselines using measured or clearly modeled energy and performance.

This paper proposes an intelligent resource-management framework that integrates workload forecasting, energy and carbon estimation, constraint-aware optimization, and feedback-based control. It is intended for trace-driven cloud or edge evaluation, with a clear separation between proposed design and results that require execution.

A. Problem Statement

Given a sequence of workload observations, available resources, service-level constraints, and an energy model, the objective is to select a resource allocation that minimizes energy-related cost while preserving required performance and capacity constraints.

B. Research Objectives

Develop a modular AI-assisted resource-management architecture.

Forecast short-term workload demand using time-aware telemetry features.

Optimize resource allocation using energy, carbon intensity, and SLA constraints.

Compare the proposed policy with static, utilization-based, and energy-unaware baselines.

Establish a reproducible evaluation protocol that reports both environmental and performance metrics.

C. Research Questions

RQ1: Can workload prediction improve resource allocation compared with reactive threshold policies?

RQ2: Does energy-aware scheduling reduce energy per completed task without unacceptable SLA violations?

RQ3: When carbon intensity varies over time, does carbon-aware scheduling reduce estimated emissions?

RQ4: What trade-off exists between energy, latency, utilization, and control overhead?

II. RELATED WORK

A. Green AI and Efficient Computing

Schwartz et al. argued that AI research should account for computational cost and environmental impact alongside predictive performance [1]. Henderson et al. proposed reporting practices for measuring and reporting energy and carbon emissions in machine learning experiments [2]. These works motivate explicit measurement boundaries and transparent reporting rather than treating environmental benefit as an assumed property of AI.

B. Energy-Aware Data-Center Management

Beloglazov and Buyya studied energy-efficient allocation of virtual machines in cloud data centers, including consolidation and dynamic resource management [3]. Their work provides a foundation for comparing energy-aware placement against conventional policies. In practice, the best policy depends on workload characteristics, hardware power curves, migration overhead, and SLA requirements.

C. Carbon-Aware Computing

Carbon emissions associated with electricity use vary by region and time. Carbon-aware computing therefore considers not only the amount of energy consumed but also the carbon

DOI : https://doi.org/10.65713/ijsitv2i401

intensity of the electricity supply. Radovanović et al. discussed carbon-aware computing systems and the role of scheduling workloads in response to grid conditions [4]. A carbon-aware controller must nevertheless respect deadlines, service constraints, and the availability of reliable carbon-intensity data.

D. Research Gap

The literature motivates energy-aware scheduling, transparent environmental accounting, and carbon-aware workload placement. This paper proposes a unified experimental framework combining demand forecasting, explicit SLA constraints, energy estimation, optional carbon-intensity weighting, and feedback-based resource control. The contribution to be tested is the measurable effect of this integration under a controlled workload-trace evaluation, not a claim that any single component is new.

III. PROPOSED SYSTEM ARCHITECTURE

The proposed system receives workload and platform telemetry, predicts near-term resource demand, estimates energy and emissions for candidate allocations, and chooses an allocation subject to performance and capacity constraints. A feedback loop updates the controller using observed utilization, energy, latency, and SLA outcomes.

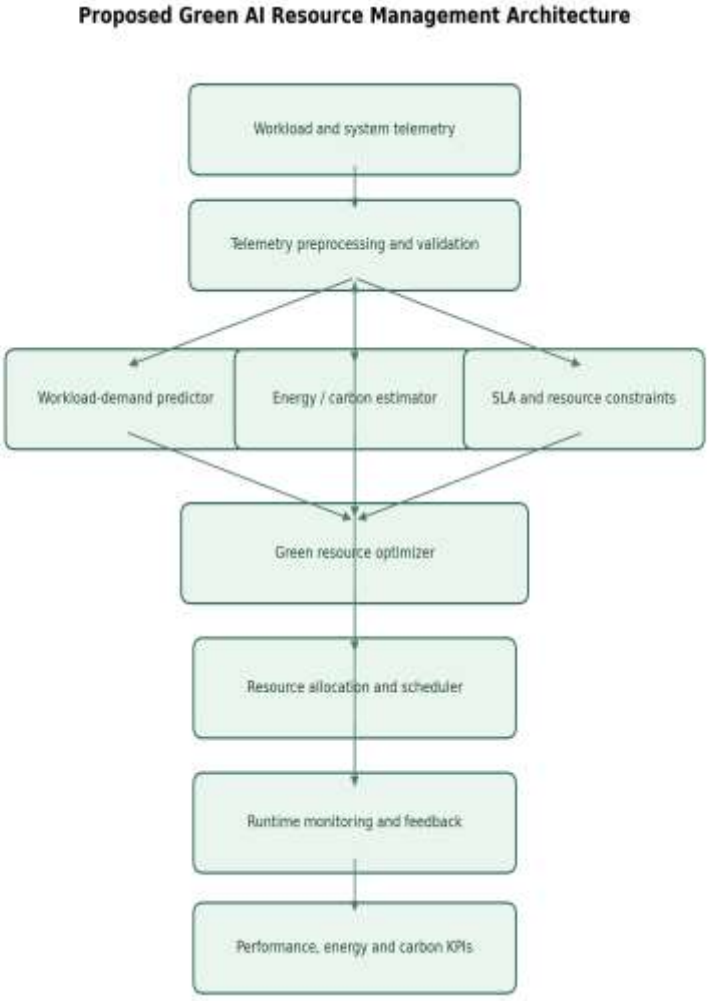


Fig. 1. Proposed Green AI resource-management architecture; design illustration only.

DOI : https://doi.org/10.65713/ijsitv2i401

A. Telemetry and Workload Prediction

Input features may include CPU utilization, memory pressure, request rate, queue length, task duration, active virtual machines or containers, historical demand, and time-of-day indicators. The forecasting module predicts demand over a configurable horizon. Candidate models include persistence, moving-average forecasting, gradient-boosted trees, and a compact recurrent or temporal neural network. The persistence model is an essential baseline.

B. Energy and Carbon Estimation

The energy model maps resource allocation and observed utilization to estimated power. When hardware power measurements are available, the model should be calibrated against those measurements. Otherwise, results must be labeled as model-based estimates and include sensitivity analysis.

$$E = \int_0^T P(t)\, dt \quad (1)$$

where $P(t)$ is power in watts and E is energy in joules; division by 3.6×10^6 converts joules to kilowatt-hours. Estimated emissions can be calculated using time-varying carbon intensity:

$$C = \Sigma_t E_t \times I_t \quad (2)$$

where E_t is energy consumed in interval t and I_t is carbon intensity in a compatible unit such as $kgCO_2e/kWh$. The time interval and units must be consistent.

C. Optimization Objective

Let a denote a candidate resource allocation. A weighted objective may combine normalized energy, carbon emissions, latency, and resource cost:

$$J(a) = \lambda E\, \hat{E}(a) + \lambda C\, \hat{C}(a) + \lambda L\, \hat{L}(a) + \lambda R\, \hat{R}(a) \quad (3)$$

The λ coefficients are nonnegative weights selected on validation workloads. Hats indicate normalized quantities. Optimization is subject to capacity and service constraints:

$$CPU(a) \le CPUmax;\ Memory(a) \le Memmax;\ SLAviolations(a) \le \varepsilon \quad (4)$$

For a carbon-aware experiment, λC may be positive; for an energy-only experiment, λC is set to zero. The results should include a Pareto comparison so that a single weighted score does not hide trade-offs.

D. Feedback Controller

At each control interval, the controller observes current telemetry, predicts demand, evaluates feasible resource allocations, applies the selected action, and monitors the result. Actions may include scaling the number of instances, changing CPU allocations, consolidating workloads, or deferring flexible tasks where deadlines permit. Migration and startup costs must be included if those actions are modeled.

IV. METHODOLOGY AND IMPLEMENTATION

A. Recommended Dataset and Experimental Environment

For reproducible trace-driven experiments, use a public Google cluster workload trace or another openly licensed cloud workload trace with documented fields and access terms. The Google cluster-data repository is a candidate source [5]. Before use, inspect the trace schema and verify whether it contains sufficient task, resource, and timing information for the chosen experiment. The trace does not by itself provide universal power measurements; energy must be measured on a test platform or estimated with a documented, calibrated model.

DOI : https://doi.org/10.65713/ijsitv2i401

Implementation can use Python, pandas, NumPy, scikit-learn, and PyTorch. A trace-driven simulator can model resource allocation and queueing, while a small controlled testbed can be used to validate the energy model. The final paper must identify the exact trace version, date range, filters, hardware, software versions, and energy-measurement method.

B. Data Preparation

Inspect timestamps, task identifiers, resource requests, usage records, missing values, and duplicate entries.

Aggregate workload into fixed intervals appropriate to the trace granularity.

Create time-ordered training, validation, and test partitions to prevent future-data leakage.

Fit normalization and imputation parameters on training data only.

Document trace filtering and report how many tasks or intervals remain after preprocessing.

C. Workload Prediction

The first forecasting target is CPU demand over the next H intervals. A simple baseline predicts the next value using the most recent observation. A neural model may use a sliding window of the previous W observations to predict future demand. Forecasting performance should be measured with mean absolute error (MAE) and root mean squared error (RMSE).

$$MAE = (1/N)\ \Sigma_i\ |y_i - \hat{y}_i| \quad (5)$$

$$RMSE = \sqrt{[(1/N)\ \Sigma_i\ (y_i - \hat{y}_i)^2]} \quad (6)$$

Forecast accuracy is not an end metric: the final resource-management policy must be evaluated on energy and service performance, since a better forecast does not necessarily imply better scheduling.

D. Baseline Policies

Static allocation: fixed resource capacity throughout the run.

Reactive threshold autoscaling: add or remove capacity when utilization crosses predefined thresholds.

Prediction-only allocation: use the demand forecast without an explicit energy objective.

Energy-aware allocation: optimize estimated energy subject to capacity and SLA constraints.

Green AI policy: combine demand forecasting, energy-aware optimization, and optional carbon-intensity weighting.

DOI : https://doi.org/10.65713/ijsitv2i401

International Journal of Sustainable Intelligent Technologies
ISSN: 3139-1249
IJSIT-01

E. Experimental Workflow

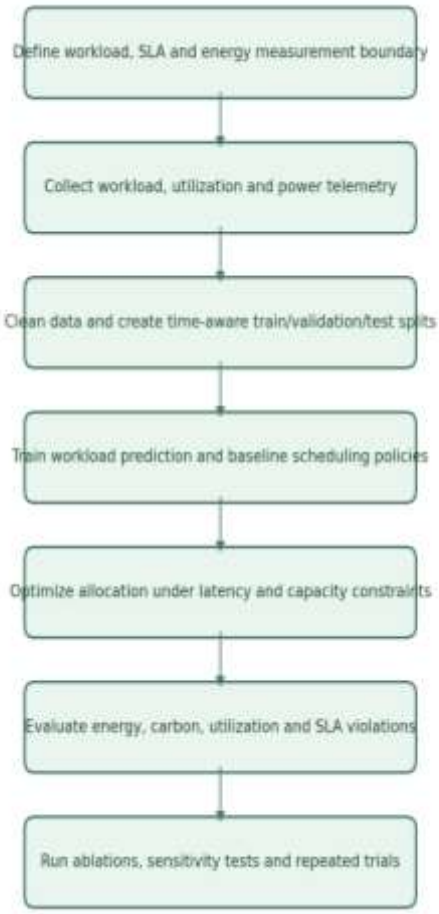


Fig. 2. Proposed reproducible evaluation workflow.

F. Initial Implementation Settings

Setting	Proposed starting point
Control interval	Choose based on trace granularity; report explicitly
Forecast horizon	1–5 control intervals; tune on validation data
Forecast models	Persistence, moving average, compact PyTorch model
Optimization	Enumerate feasible actions or use constrained heuristic
Random seeds	13, 42, 87, 123, 2026 where stochastic
Model selection	Validation objective with SLA guardrails
Power model	Measured/calibrated model; otherwise label as estimated

These are methodological recommendations, not completed experimental settings. The actual values should be chosen after inspecting the selected trace and target environment.

DOI : https://doi.org/10.65713/ijsitv2i401

V. EXPERIMENTAL EVALUATION

A. Metrics

The evaluation should report energy per completed task, total energy, average and 95th/99th percentile latency, throughput, SLA violation rate, CPU and memory utilization, number of scaling actions, and estimated carbon emissions where carbon intensity data are available.

Energy per task = Total energy / Number of completed tasks (7)

SLA violation rate = Violating requests / Total evaluated requests (8)

Energy and emissions should be reported with clear measurement boundaries. If the experiment uses a simulator or estimated power curve, do not describe the results as directly measured physical energy.

B. Comparison Table Template

Policy	Energy/task	P95 latency	SLA violations	Utilization	CO_2e estimate
Static allocation	Pending	Pending	Pending	Pending	Pending
Reactive autoscaling	Pending	Pending	Pending	Pending	Pending
Prediction-only	Pending	Pending	Pending	Pending	Pending
Energy-aware	Pending	Pending	Pending	Pending	Pending
Green AI policy	Pending	Pending	Pending	Pending	Pending

Table I. Performance and environmental comparison template. All numeric entries require execution.

C. Ablation and Sensitivity Studies

Remove workload prediction and use reactive scaling.

Disable carbon-intensity weighting while retaining energy optimization.

Vary the energy, carbon, latency, and resource-cost weights.

Vary workload intensity and forecast horizon.

Change power-model assumptions and include scaling/migration overhead.

Test multiple trace windows and repeat stochastic experiments.

D. Result Status and Discussion

Result status: No trace has been processed and no experiment has been executed for this manuscript. Results are intentionally marked as pending. The final discussion must report measured outcomes and should not assume that the proposed policy saves energy.

The main hypothesis is that demand prediction and energy-aware optimization can reduce energy per completed task while respecting SLA limits. This hypothesis may fail if prediction errors, model overhead, workload volatility, or startup and migration costs outweigh the benefit.

VI. LIMITATIONS AND FUTURE WORK

The framework depends on the fidelity of workload traces, the accuracy of the energy model, and the chosen SLA definitions. A simulated energy reduction is not equivalent to a measured reduction on physical hardware. Carbon estimates depend on the location, time resolution, and provenance of carbon-intensity data. Future work should validate the policy on a controlled cluster or edge testbed, compare multiple hardware generations, and

DOI : https://doi.org/10.65713/ijsitv2i401

investigate safe reinforcement learning or constrained optimization while maintaining auditable SLA guardrails.

VII. CONCLUSION

This paper proposes Green AI for Intelligent Resource Management, a framework combining workload prediction, energy and carbon estimation, constrained resource allocation, and feedback-based control. The methodology defines a reproducible trace-driven evaluation and compares the proposed policy with static, reactive, prediction-only, and energy-aware baselines. Environmental performance is evaluated alongside latency, throughput, utilization, and SLA compliance. Since implementation and experiments have not been completed, no quantitative energy or carbon savings are claimed.

REFERENCES

[1] D. S. W. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," Communications of the ACM, vol. 63, no. 12, pp. 54–63, 2020, doi: 10.1145/3381831. https://doi.org/10.1145/3381831

[2] P. Henderson et al., "Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning," Journal of Machine Learning Research, vol. 21, no. 248, pp. 1–43, 2020. https://www.jmlr.org/papers/v21/20-312.html

[3] A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers," in Proc. 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), 2010, pp. 826–831, doi: 10.1109/CCGRID.2010.46.

[4] A. Radovanović et al., "Carbon-Aware Computing for Datacenters," IEEE Transactions on Power Systems / related carbon-aware computing publication details should be verified against the final selected source before submission. https://arxiv.org/abs/2106.11750

[5] K. K. Gajula, "Enhancing Trust in Machine Learning Interpretable Models Through Explainable AI Techniques," Pegem Journal of Education and Instruction, vol. 13, no. 4, pp. 909–915, 2023.

[6] K. K. Gajula, "Designing an Ethical Data Science Framework for Responsible and Transparent AI Integration," Emerging Trends in Digital Transformation, vol. 1, no. 1, pp. 74–87, 2025.