ECO-Hadoop: Energy Consumption Optimization System for Apache Hadoop

Zemin Chao
Harbin Institute of Technology
Harbin, China
chaozm@hit.edu.cn

Xiao Chen
Harbin Institute of Technology
Harbin, China
24B903103@stu.hit.edu.cn

Jianhe Cen
Harbin Institute of Technology
Harbin, China
24B903105@stu.hit.edu.cn

Yumeng Fu
Harbin Institute of Technology
Harbin, China
25S103495@stu.hit.edu.cn

Hongzhi Wang*
Harbin Institute of Technology
Harbin, China
wangzh@hit.edu.cn

ABSTRACT

The big data era drives the continuous development of data processing frameworks, while posing significant energy consumption challenges. This paper proposes ECO-Hadoop, a system that reduces energy consumption in Hadoop clusters by jointly optimizing worker node settings and Hadoop-level configurations. ECO-Hadoop continuously monitors the Hadoop cluster and models the relationship between the overall energy consumption of Hadoop applications and the configurations. Based on this model, parameter optimization of the Hadoop cluster is formulated as a sequential decision-making process, employing reinforcement learning to improve the platform's energy efficiency. By simultaneously adjusting the system parameters and the configurations of individual nodes, ECO-Hadoop achieves energy savings of up to 20.8% on standard benchmarks. The system also provides a user-friendly interface and comprehensive cluster monitoring capabilities, enabling users to visualize the optimization workflow.

PVLDB Reference Format:

Zemin Chao, Xiao Chen, Jianhe Cen, Yumeng Fu, and Hongzhi Wang. ECO-Hadoop: Energy Consumption Optimization System for Apache Hadoop. PVLDB, 19(12): 4750 - 4753, 2026. doi:10.14778/3827998.3828113

PVLDB Artifact Availability:

The source code, data, and/or other artifacts have been made available at https://github.com/chenxiao226/eco-hadoop.

1 INTRODUCTION

The exponential growth in data volume has intensified data processing demands, driving the widespread adoption of big data frameworks. However, energy consumption has emerged as a prominent challenge accompanying the rapid growth of the big data market, with the International Energy Agency (IEA) projecting that data center power demand could reach approximately 945 TWh by 2030 [2]. Consequently, reducing the energy consumption of these platforms is essential for lowering both computing costs and carbon emissions.

Despite the rise of alternative frameworks, Apache Hadoop remains a de facto standard in big data computing, deeply entrenched in global enterprise infrastructures. According to a 2026 industry report by IMARC Group, the global Hadoop market was valued at USD 170.6 billion in 2025 and is projected to exhibit sustained growth, driven by the indispensable need for robust, large-scale data processing [1]. Although Hadoop is a powerful tool for modern big data analytics, its energy footprint continues to grow. Therefore, developing energy-saving methods for Hadoop is necessary to address sustainability challenges.

Most research on Hadoop energy optimization has focused either on tuning worker node configurations to reduce power consumption without considering the overall Hadoop configuration, or on resource and task allocation while neglecting node configurations [3]. These isolated approaches limit the overall effectiveness of energy conservation.

To address these challenges, we present ECO-Hadoop, a system that minimizes Hadoop's energy consumption by jointly optimizing Hadoop configurations and node-level CPU frequencies. Operating transparently in the background, it automatically optimizes cluster parameters and node-level settings without requiring domain expertise.

Internally, as illustrated in Figure 1, ECO-Hadoop relies on a Monitor Component that continuously collects system statistics. A lightweight preprocessing module then handles data quality issues and abstracts metrics such as CPU frequency and utilization into discrete node states, providing a robust foundation for subsequent energy estimation. Relying on this data, a two-step energy estimator calculates the total application energy consumption by predicting both the instantaneous power of these states and their temporal changes across all nodes during execution.

In summary, we demonstrate ECO-Hadoop, an interactive system designed to optimize the energy efficiency of Hadoop clusters. Our contributions are as follows:

- We propose a joint optimization framework that simultaneously tunes both Hadoop-level configurations and node-level CPU frequencies.
- We design a learning-based architecture that integrates a 1D-CNN energy estimator with a Gated Recurrent Unit

*Corresponding author.

Proceedings of the VLDB Endowment, Vol. 19, No. 12 ISSN 2150-8097.
doi:10.14778/3827998.3828113

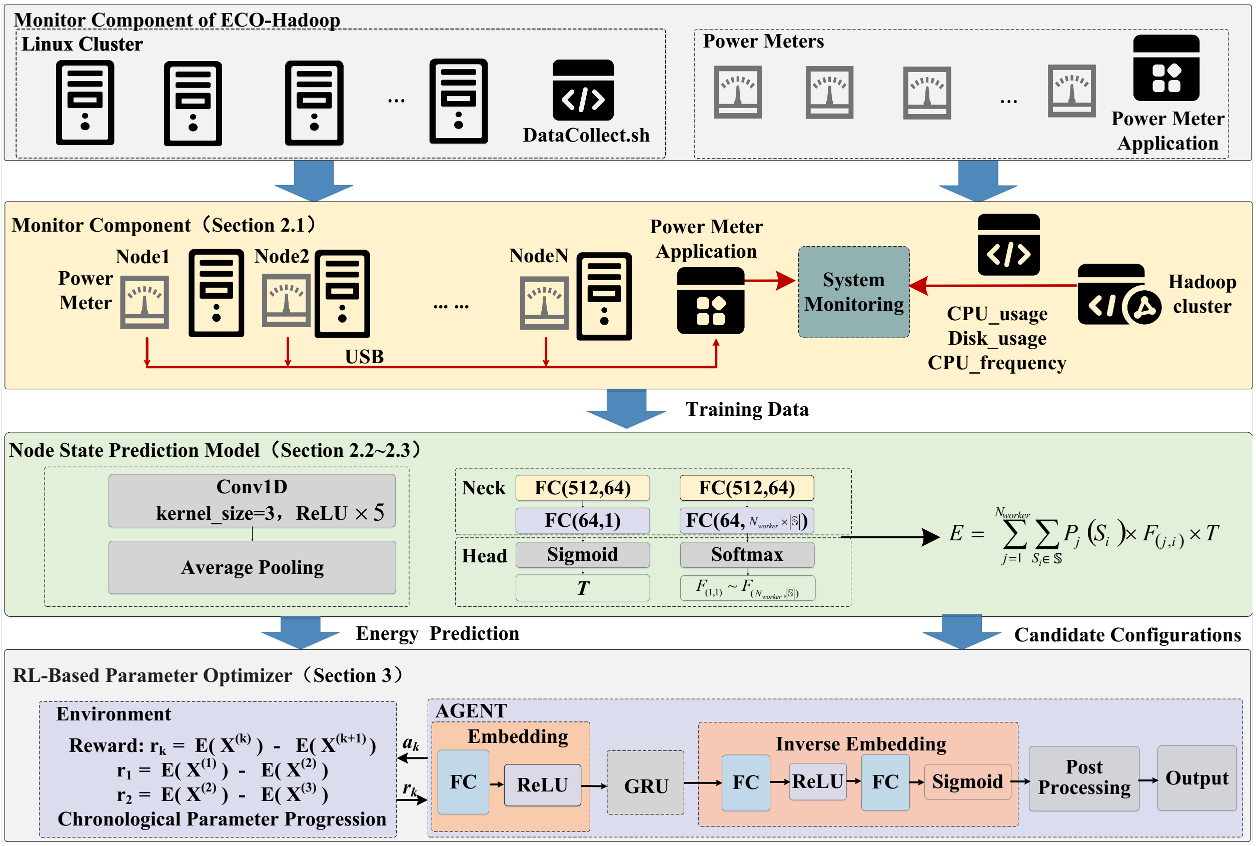


Figure 1: The framework of ECO-Hadoop.

(GRU)-based reinforcement learning (RL) agent for automated, dynamic parameter tuning.

- We provide a user-friendly, real-time dashboard that allows users to seamlessly monitor cluster states, interactively compare heuristic adjustments with RL-recommended parameters, and visualize the empirical energy savings of up to 20.8% on standard benchmarks.

2 PREDICTING ENERGY CONSUMPTION FOR HADOOP APPLICATIONS

This section introduces the monitor and energy estimator components of ECO-Hadoop, which collect system statistics and predict energy consumption to optimize Hadoop's energy efficiency.

2.1 Gathering System Information

The **Monitor Component** of ECO-Hadoop serves as the foundation for subsequent energy optimization. Power meters connected to power strips collect node-level power consumption and transmit real-time measurements to the controller node. In parallel, ECO-Hadoop gathers hardware metrics—such as CPU usage, CPU frequency, and I/O status—as well as task status and historical records for each node. For each executed Hadoop job, the system records the job's configuration and the operational states and power consumption of each node.

To mitigate data quality issues in power meter readings, ECO-Hadoop integrates a lightweight preprocessing module that automatically aligns timestamps and interpolates missing values, providing a robust foundation for accurate energy estimation.

2.2 Fitting the Power of Worker Nodes

Modeling the instantaneous power consumption of individual nodes using pure heuristics is challenging due to the complex interplay of system metrics. Instead, ECO-Hadoop employs neural networks to predict power consumption based on real-time node states. Specifically, the system monitors metrics including the node's CPU frequency, CPU utilization, and I/O status, which directly determine its instantaneous power consumption. Since nodes with similar resource utilization typically share similar power consumption profiles, ECO-Hadoop abstracts these fine-grained metrics into a finite set of discrete states to reduce modeling overhead. This discrete state space is denoted as $\mathbb{S}$.

ECO-Hadoop trains neural networks to fit the power consumption of each state in $\mathbb{S}$. Furthermore, because hardware differs across nodes, the same state may correspond to different power consumption values. For highly heterogeneous clusters, ECO-Hadoop learns a dedicated power model for each node. Assuming there are N_{worker} nodes, the power model for the j-th node is denoted as P_j. Given a node state $S_i \in \mathbb{S}$, the predicted power for the j-th $(1 \le j \le N_{\text{worker}})$

node in the cluster is denoted as $P_j(S_i)$. In data centers with homogeneous hardware configurations, power models can be shared across nodes, significantly reducing training overhead.

Because Hadoop's MapReduce paradigm alternates between CPU- and I/O-intensive phases, the collected state data are inherently imbalanced. To mitigate this sample skewness, ECO-Hadoop applies Feature Distribution Smoothing (FDS) [4] to smooth the feature vectors.

2.3 Predicting Total Energy Consumption

Since energy consumption is the time integral of power, ECO-Hadoop avoids tracking fine-grained chronological power fluctuations. Instead, it estimates the total duration each node spends in each discrete state during job execution. To this end, the **Node State Prediction Model** employs a one-dimensional convolutional neural network (1D-CNN) backbone with average pooling to extract workload features. These features are fed into a dual-branch neck: the first branch applies a sigmoid function to predict a normalized runtime ratio, which is subsequently scaled to obtain the actual total application runtime T, while the second applies a softmax function to estimate the proportion of time spent in each node state. We denote the fraction of time node j spends in state S_i as $F_{(j,i)}$.

The total time node j spends in state S_i is $T \times F_{(j,i)}$, during which it consumes power at a rate of $P_j(S_i)$. Consequently, the total cluster energy consumption $E(X)$ under configuration X is estimated as:

$$E(X) = \sum_{j=1}^{N_{\text{worker}}} \sum_{S_i \in \mathbb{S}} P_j(S_i) \times F_{(j,i)} \times T, \qquad (1)$$

where N_{worker} is the total number of worker nodes and $\mathbb{S}$ is the set of all discrete states. As discussed in Section 2.2, for clusters with homogeneous hardware, the power model $P_j(\cdot)$ can be shared across identical nodes.

3 OPTIMIZATION OF ENERGY EFFICIENCY

ECO-Hadoop utilizes the energy estimator described in Section 2.3 as an advisor. Specifically, ECO-Hadoop searches the valid parameter space for configurations that reduce cluster energy consumption. For each candidate configuration, ECO-Hadoop uses the energy estimator to predict the corresponding energy consumption. The parameter space includes the CPU frequency of each node as well as key Hadoop configurations critical to cluster performance, rendering exhaustive search infeasible. Consequently, the **RL-Based Parameter Optimizer** of ECO-Hadoop explores this space using deep reinforcement learning by formulating parameter optimization as a sequence of decisions $a_1, a_2, \ldots, a_K$. Each action a_k represents an adjustment to the current configuration $X^{(k)}$ to form the new configuration $X^{(k+1)} = X^{(k)} + a_k$. The reward r_k for action a_k is defined as the reduction in energy consumption resulting from this action. The energy reduction is estimated using the energy model from Section 2.3: $r_k = E(X^{(k)}) - E(X^{(k+1)})$.

The agent employs GRUs to generate new parameter configurations. The initial configuration $X^{(0)}$ is generated using expert-guided heuristics. At the k-th iteration, the agent determines an action a_k to adjust the current configuration $X^{(k)}$. The agent processes the current parameters $X^{(k)}$ through an embedding layer, captures temporal dependencies via a three-layer GRU with ReLU activation, and outputs feasible actions using a sigmoid-activated output projection layer followed by a post-processing module. This ensures the physical feasibility of the generated configurations, while the reward signal r_k directly quantifies the energy efficiency gain from state transitions and guides the optimization process. Finally, the optimizer outputs the parameter configuration with the lowest predicted energy consumption among all configurations explored by the agent.

To ensure policy convergence and reduce training overhead during the RL training phase, ECO-Hadoop employs a convergence-based termination mechanism. At the end of each evaluation epoch t, the system records the average energy consumption E_t under the current policy. Training automatically terminates if the relative energy reduction, defined as $(E_{t-1} - E_t)/E_{t-1}$, falls below a predefined convergence threshold $\epsilon > 0$ for θ consecutive epochs.

4 DEMONSTRATION

ECO-Hadoop features a comprehensive real-time dashboard that integrates macro-level cluster health metrics with fine-grained node-level telemetry.

Phase 1: Application Registration and Model Initialization. When a new Hadoop application is submitted, it must first be registered within the system. Users interact with the submission interface as shown in Figure 2 to specify the executable JAR file of the application. For unseen workloads, ECO-Hadoop initiates a background profiling and training phase. During this stage, the system gathers essential system statistics, enabling the underlying models to learn the application's resource utilization patterns and establish an energy baseline.

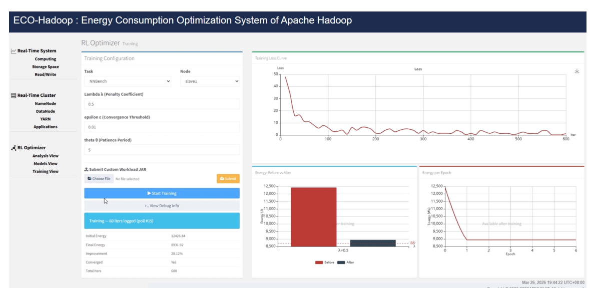


Figure 2: Registration of Hadoop Applications.

Phase 2: Performance Prediction and Interactive Parameter Optimization. Once the application is registered and profiled, users can access the optimization module. Before submitting the job to the cluster, ECO-Hadoop reports both the recommended parameters and the projected energy consumption. To facilitate interactive tuning, the interface displays the recommended parameters. Users can manually adjust parameters through an intuitive interface (Figure 3), and ECO-Hadoop reports the predicted energy consumption of these configurations in real time. This allows users to directly evaluate their heuristic adjustments against the system's recommendations. As users adjust the sliders, the system dynamically updates energy predictions for the modified parameters. Once the final configuration is confirmed, the application is launched with the user-specified settings.

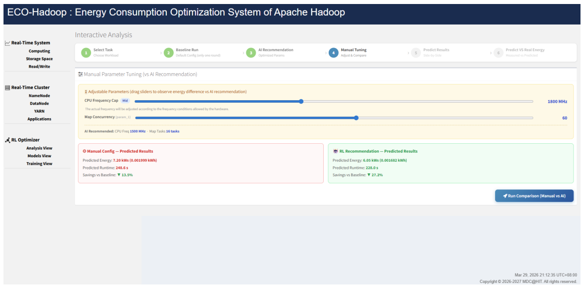


Figure 3: Interactive Parameter Optimization.

Phase 3: Execution Monitoring and Final Result Display. Upon submission, the finalized configuration is distributed to each node. During application execution, ECO-Hadoop continuously monitors the operational status of the cluster and the resource utilization of individual worker nodes. These metrics are visualized in Figure 4, providing a comprehensive summary of the application's execution. This dashboard juxtaposes the actual energy consumption under the optimized configuration against the unoptimized baseline, clearly demonstrating the energy savings achieved by ECO-Hadoop.

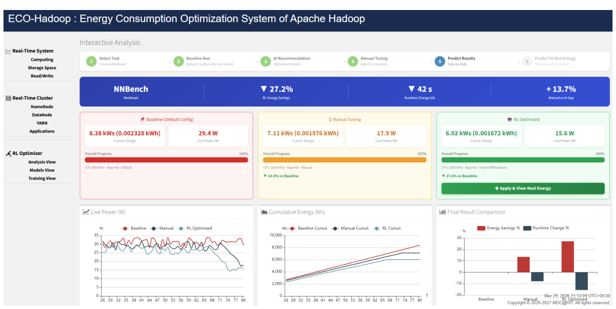


Figure 4: Comparing Total Energy Consumption.

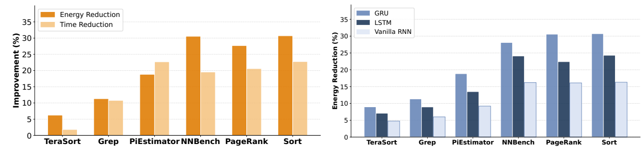
5 EVALUATION ON ENERGY CONSERVATION

To evaluate the energy optimization capabilities of ECO-Hadoop, we deployed the system on a three-node Ubuntu-based Hadoop cluster. Our experimental workloads consist of six diverse Hadoop benchmark applications. Overall, the system demonstrates substantial energy savings across all tested scenarios.

As illustrated in Figure 5(a), ECO-Hadoop achieves an average energy reduction of approximately 20.8% compared to the default cluster configurations. ECO-Hadoop also reduces application execution time by 16.3%, thereby avoiding the typical trade-off between energy efficiency and performance.

Additionally, to validate the design of our RL agent, Figure 5(b) presents an ablation study comparing energy reduction across different agent architectures. The results indicate that the GRU-based model achieves the highest energy savings, validating our algorithmic choice for the parameter optimizer.

To ensure the energy reduction stems from multidimensional optimization rather than a poorly tuned baseline, we conduct an exact Shapley value attribution analysis across parameter space. We calculate the marginal energy contribution of transitioning each parameter from a representative baseline (whose predicted energy matches the pre-optimization average) to the RL-optimized state. As illustrated in Figure 6 for NNBench, the gains are jointly driven by multiple adjustments. Similar synergistic effects are observed across all benchmarks, confirming that ECO-Hadoop actively discovers complex energy-performance trade-offs rather than merely exploiting a single suboptimal default setting.



(a) Energy and time reduction per workload.

(b) Ablation study of agent models on energy reduction.

Figure 5: Evaluation of optimization gains across six benchmark workloads. (a) ECO-Hadoop dynamically tuning configurations yields an average energy reduction of 20.8% and decreases execution time by 16.3%, effectively avoiding the typical energy-performance trade-off. (b) An ablation study confirms that the GRU-based agent consistently achieves the highest energy savings compared to LSTM and Vanilla RNN architectures.

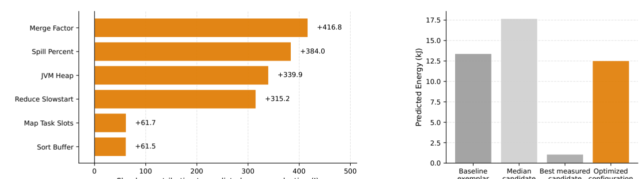


Figure 6: Shapley parameter attribution on NNBench. Optimization gains are contributed by multiple parameters.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (NSFC) under Grants 62232005 and 62202126, and by the China Postdoctoral Science Foundation under Grant 2025M774320. We appreciate the support from China Mobile Group Heilongjiang Co., Ltd., with whom this research was jointly conducted.

REFERENCES

[1] IMARC Group. 2026. Hadoop Market Size, Share And Industry Growth Report 2034. https://www.imarcgroup.com/hadoop-market Accessed: 2026-06-28.
[2] International Energy Agency (IEA). 2026. *Energy and AI-Analysis*. Technical Report. IEA. https://www.iea.org/report/energy-and-ai Accessed: 2026-06-28.
[3] Amir Masoud Rahmani, Ehsan Yazdani Chamzini, Mohsen Pourshaban, and Mehdi Hosseinzadeh. 2024. Scheduling of Big Data Workflows in the Hadoop Framework with Heterogeneous Computing Cluster. *Arabian Journal for Science and Engineering* 50, 2 (2024), 12449–12461. https://doi.org/10.1007/s13369-024-09779-9
[4] Yuzhe Yang, Kaiwen Zha, Yingcong Chen, Hao Wang, and Dina Katabi. 2021. Delving into deep imbalanced regression. In *International conference on machine learning*. PMLR, 11842–11851.