Volume 11, Issue 3, March – 2026 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

Deep Learning Based CPU Scheduling for Energy and Time Optimization

K. Usha Rani[1]; V. G. Kishore Kumar[2]; R. Pravin[3]; T. P. Yagna Narayanan[4]

[1]Assistant Professor, CSE Department, K.L.N. College of Engineering, Sivagangai, Tamil Nadu, India
[2,3,4]UG Scholar, CSE Department, K.L.N. College of Engineering, Sivagangai, Tamil Nadu, India

Publication Date: 2026/03/21

Abstract: Efficient CPU scheduling plays a crucial role in optimizing system performance and reducing energy consumption in modern computing environments. Conventional scheduling methods, such Round Robin (RR), Shortest Job First (SJF), and First Come First Serve (FCFS), are based on preset heuristics and are unable to dynamically adjust to changing workload patterns. This study suggests a CPU scheduling paradigm based on deep learning that maximises execution time and energy use. The suggested approach predicts the best scheduling choices by utilising past task execution data, such as arrival time, burst time, and priority metrics. To find effective task ordering techniques that reduce average waiting time, turnaround time, and total power consumption, a neural network model is developed. In terms of execution efficiency and energy savings, experimental evaluation shows notable gains over traditional scheduling strategies. The outcomes confirm how well deep learning works to enable intelligent, performance-aware, and adaptive CPU scheduling techniques for contemporary computer systems.

Keywords: *Deep Learning, CPU Scheduling, Energy Optimization, Execution Time Reduction, Multilayer Perceptron (MLP), Adam Optimizer, Neural Networks, Intelligent Scheduling, Machine Learning, System Performance Optimization.*

How to Cite: K. Usha Rani; V. G. Kishore Kumar; R. Pravin; T. P. Yagna Narayanan (2026) Deep Learning Based CPU Scheduling for Energy and Time Optimization. *International Journal of Innovative Science and Research Technology,* 11(3), 1685-1690. https://doi.org/10.38124/ijisrt/26mar972

I. INTRODUCTION

Operating systems depend heavily on effective CPU scheduling, which has a direct impact on system responsiveness, performance, and resource usage. Workload management has grown more dynamic and sophisticated as cloud computing, fog computing, edge settings, IoT systems, and GPU-based data centers have all expanded quickly. Static heuristics and preset rules are the foundation of traditional scheduling algorithms like Round Robin (RR), Shortest Job First (SJF), and First-Come-First-Serve (FCFS). Despite being computationally cheap, traditional methods are unable to handle the diverse and unexpected workloads found in contemporary distributed systems, which frequently leads to longer wait times, ineffective resource allocation, and worse Quality of Service (QoS) [10], [13], and [15].

In large-scale computing infrastructures, energy efficiency has become a significant problem beyond performance optimisation. Significant electrical power consumption by data centers, cloud-fog topologies, vehicle edge systems, and IoT platforms raises operating costs and has an adverse effect on the environment. In order to concurrently balance execution time, throughput, and power consumption, recent research has focused on energy-aware task scheduling. Energy-efficient scheduling in cloud environments, federated learning IoT systems, automated guided vehicles, and distributed edge networks has been effectively addressed by deep reinforcement learning and intelligent optimisation models [2], [5], [7], [9], [11], and [14]. The significance of adaptive learning-based strategies for managing heterogeneous computing needs is further demonstrated by CPU-GPU workload coordination and AI service scheduling [3], [12].

These developments have made deep learning a viable option for developing data-driven and adaptive CPU scheduling systems. Neural network models and reinforcement learning-based schedulers are able to dynamically modify task allocation techniques and learn optimal policies from past workload patterns [1], [6], [8], and [10]. In this regard, workload-aware characteristics like arrival time, burst time, and priority level are utilised by the suggested deep learning-based CPU scheduling architecture. The system seeks to minimise execution time while lowering energy usage using a Multilayer Perceptron (MLP) model trained with the Adam optimiser. The suggested method offers a scheduling framework that is both energy-efficient and performance-aware, making it appropriate for next-generation computing environments by incorporating intelligent decision making into task management.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

II. METHODOLOGY

The suggested approach presents a CPU scheduling system based on deep learning that is intended to concurrently optimise execution time and energy usage. The system starts by gathering past task execution data, which includes variables like priority, CPU utilisation, arrival time, and burst time. After being normalised, these characteristics are preprocessed and organised into a dataset that is appropriate for supervised learning. Training a model that can forecast the best scheduling order or priority score for incoming tasks is the aim. The scheduler can now dynamically adjust to changing workload patterns and go beyond static heuristics thanks to this data-driven approach.

Since a Multilayer Perceptron (MLP) neural network can learn non-linear correlations between input features and scheduling decisions, it is used as the primary prediction model. Multiple hidden layers with nonlinear activation functions, an output layer that generates scheduling priorities or execution order predictions, and an input layer that represents task properties make up the architecture. Backpropagation is used to train the model with the Adam optimiser, which guarantees steady weight updates and effective convergence. By minimising average waiting time and energy consumption, the loss function allows the network to learn scheduling behaviour that is sensitive to performance.

The trained MLP model functions as an intelligent decision engine that is integrated with the CPU scheduler during deployment. The model estimates an optimal scheduling score for each work based on the attributes of newly arrived tasks. Tasks are dynamically arranged based on these estimates in order to minimise waiting time, shorten turnaround time, and use less power. In order to assess consumption trends, energy estimation is integrated utilising execution time and CPU utilisation indicators. The system is evaluated utilising a variety of workload scenarios of different proportions against well-known algorithms as FCFS, SJF, and Round Robin. To verify the efficacy, scalability, and dependability of the suggested deep learning-based scheduling system, performance metrics such as average waiting time, turnaround time, throughput, and total energy consumption are examined.

III. PROCESS FLOW

A methodical and controlled workflow is followed by the suggested Deep Learning-based CPU scheduling system to guarantee optimal execution time and energy efficiency. Data collection and workload production are the first steps in the process. Arrival time, burst time, priority level, and CPU utilisation are among the task-related parameters that are gathered from either historical execution logs or simulated settings. The primary dataset for training and assessment is made up of these parameters. To make sure the model learns from both light and heavy computational scenarios, the collected data reflects a variety of workload patterns.

Data preparation is carried out in the following stage to improve the stability and performance of the model. A feature matrix is created by cleaning, normalising, and structuring raw task attributes. Normalisation makes sure that variables with disparate scales, such priority and burst time, don't have an undue impact on the model. Subsets of the dataset are then separated for testing and training. The most important factors influencing scheduling effectiveness and energy usage can be found by using feature selection approaches. In addition to lowering computational complexity, this step guarantees that the model concentrates on pertinent input properties.

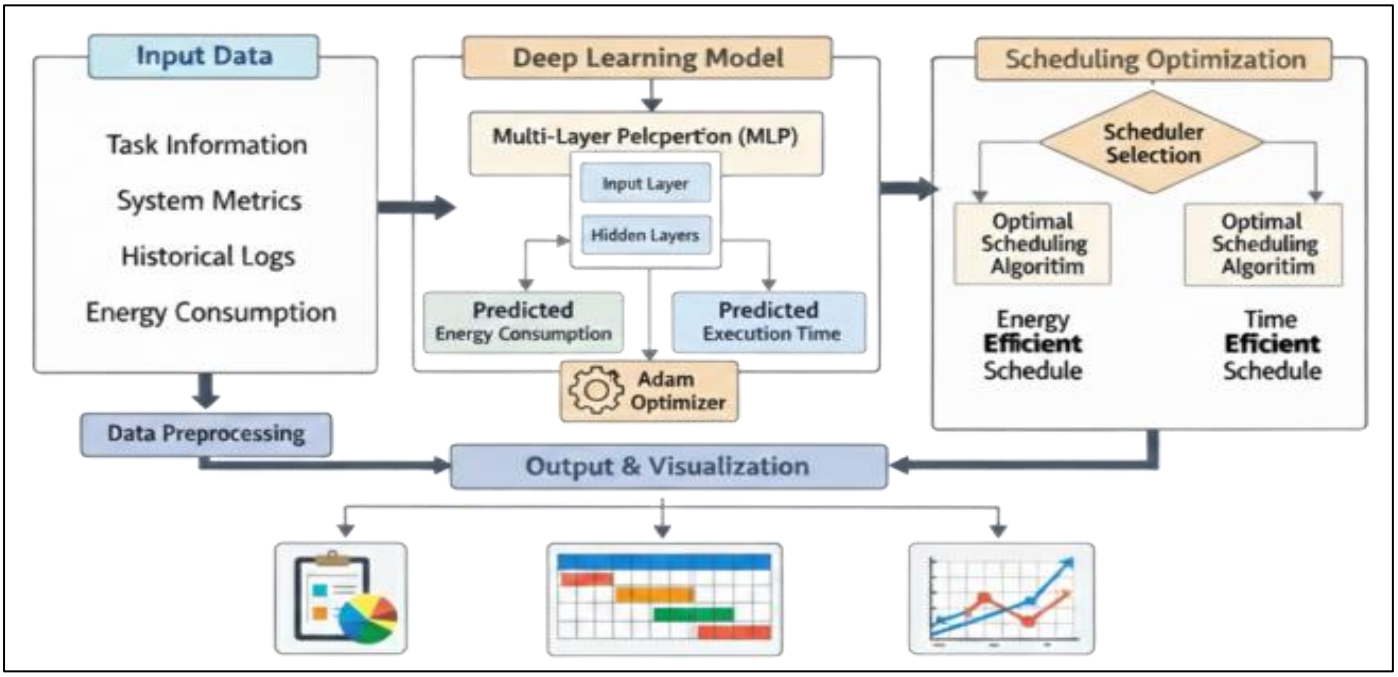

Fig 1 Process Flow

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

The Multilayer Perceptron (MLP) model is created and set up after preprocessing. Task features are sent to the input layer, and intricate nonlinear correlations between workload factors are extracted via hidden layers. Nonlinearity is introduced via activation functions like ReLU, and scheduling priority scores are produced by the output layer. The Adam optimiser, which adaptively modifies learning rates to guarantee quicker convergence and stable training, is used to train the model using backpropagation. The model is guided toward balanced optimisation by the loss function, which incorporates performance measures like predicted energy consumption and average waiting time.

Following training, the model is validated and its performance assessed using test data that has not yet been seen. For incoming jobs, the trained network forecasts the best scheduling priority. The CPU scheduling module then incorporates these anticipated values and uses the model output to dynamically rearrange the workloads. Through this connection, the conventional scheduler is changed into an intelligent decision-making tool that can adjust its behaviour to different workloads.

The trained model receives the properties of newly arrived processes in real time during runtime execution. In order to reduce execution delays and energy consumption, tasks are organised based on the scheduling scores assigned by the model. CPU active time and utilisation levels are used to estimate energy. Lastly, the suggested method is compared with conventional scheduling algorithms as FCFS, SJF, and Round Robin in order to assess system performance. Throughput, CPU utilisation, average waiting time, turnaround time, and total energy usage are among the metrics that are examined to show the effectiveness, scalability, and resilience of the deep learning-based scheduling system.

IV. ALGORITHMS USED

The suggested system analyses performance and energy optimisation by comparing a Deep Learning-based intelligent scheduler with conventional CPU scheduling algorithms. The unique advantages and disadvantages of each traditional algorithm drive the incorporation of deep learning methods.

First-Come-First-Served (FCFS) scheduling arranges procedures according to arrival order. Its main advantages are its ease of use, equitable arrival order, and minimal installation costs. It is appropriate for basic systems because it doesn't require intricate calculations. The convoy effect, which causes short processes to lag behind longer ones, impairs turnaround speed and raises average waiting times, is a problem for FCFS. It is also inflexible when it comes to changes in workload. The scheduler can prevent inefficient task sequences brought on by strict arrival-based ordering by integrating deep learning, which allows it to dynamically analyse task characteristics

Compared to FCFS, Shortest Job First (SJF) drastically lowers the average waiting time by choosing the process with the least burst time. Its ability to optimise performance under known execution times is its strongest point. However, SJF necessitates precise burst time prediction, which is sometimes unfeasible in practical systems. Longer processes may potentially starve as a result. A deep learning-based scheduler can balance short and lengthy processes to avoid hunger while preserving efficiency, learn execution patterns from past data, and make more accurate task behaviour predictions.

In time-sharing systems, Round Robin (RR) ensures fairness and enhanced responsiveness by allocating CPU time in defined time slices. Its advantages include less chance of famine and fair resource distribution. However, choosing the wrong time quantum might result in higher energy usage, more overhead, and more context switching. By dynamically adjusting to workload conditions, deep learning can optimise scheduling decisions, minimising needless context shifts and increasing energy efficiency.

CPU resources are allocated by Priority Scheduling according to predetermined priority levels. It works well for managing important or crucial activities, which makes it appropriate for real-time systems. However, it may not adjust well to changes in the workload and may starve low-priority operations. Furthermore, static priority assignment might not accurately represent the state of the system. The suggested deep learning technique can learn optimal priority patterns from data, dynamically modify scheduling choices, and get beyond the rigidity and starvation problems that arise with conventional priority-based approaches by utilising a Multilayer Perceptron (MLP) trained with the Adam optimiser.

Multilayer Perceptrons (MLPs) can learn intricate nonlinear correlations between task variables and optimal scheduling decisions, they are used as the primary predictive model in the suggested scheduling framework. Multiple hidden layers that extract significant patterns, an output layer that produces optimal scheduling scores, and an input layer that receives process parameters like arrival time, burst time, and priority make up the MLP. Using the Adam optimiser, an adaptive learning rate optimisation technique that combines the benefits of Momentum and RMSProp, the model ensures robust and effective training. Adam enables faster convergence and better performance stability by dynamically modifying weight updates according to the first and second moments of gradients. For time and energy optimisation, the MLP and Adam optimiser work together to provide precise prediction, shorter training times, and more effective scheduling.

V. SYSTEM ARCHITECTURE

The modular and tiered architecture of the suggested Deep Learning-based CPU scheduling system guarantees scalability, flexibility, and effective interaction with current scheduling systems. The Preprocessing and Feature Engineering Unit, Deep Learning Prediction Engine, Scheduling Execution Module, and Data Acquisition Module are the four main parts of the architecture. From

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

simulated or real-time settings, the Data Acquisition Module gathers task-related metrics like arrival time, burst time, priority level, and CPU utilisation. The preprocessing unit receives these inputs and uses feature structure and normalisation to get the dataset ready for training and prediction. Clean data flow and enhanced maintainability are guaranteed by this modular separation

The Deep Learning Prediction Engine, which includes a Multilayer Perceptron (MLP) trained with the Adam optimiser, is at the heart of the design. For every assignment, this engine creates optimal scheduling scores by analysing input features. In order to reduce waiting time and energy consumption, the Scheduling Execution Module then dynamically arranges processes according to the anticipated priority. Metrics including turnaround time, throughput, CPU utilisation, and overall energy consumption are also tracked by a Performance Evaluation Unit. The system architecture facilitates quantitative evaluation of enhancements made possible by astute, data-driven scheduling choices by allowing comparison with conventional algorithms such as FCFS, SJF, Round Robin, and Priority scheduling.

VI. PERFORMANCE ENHANCEMENT

Through dynamic task execution order optimisation, the suggested Deep Learning-based CPU scheduling architecture dramatically improves system performance. The intelligent scheduler predicts effective scheduling priorities by analysing workload characteristics, in contrast to traditional algorithms that depend on fixed heuristics. In a variety of workload conditions, this adaptive decision-making lowers average waiting time, turnaround time, and reaction time. Improved throughput and greater CPU utilisation result from the system's ability to learn from past execution data and spot trends that traditional approaches miss. Consistent performance even in the face of tremendous computational demand is ensured by the capacity to continuously adjust to variations in workload.

Fig 2 Performance Enhancement

By cutting down on needless CPU idle time and frequent context switching, the suggested approach improves energy performance in addition to execution efficiency. Balanced task allocation is made possible by the Multilayer Perceptron (MLP) model integration, which avoids protracted process blocking and hunger problems. Reliable scheduling predictions are produced by the Adam optimiser, which guarantees steady and precise model training. Measurable gains in execution time and energy consumption are shown when compared to FCFS, SJF, Round Robin, and Priority scheduling. All things considered, the intelligent scheduling framework offers a high-performance, scalable, and energy-conscious solution appropriate for contemporary computing environments like data centers and cloud systems.

VII. RESULT AND DISCUSSION

The efficiency of the suggested Deep Learning-based CPU scheduling system in optimising execution time and energy usage was assessed using many task sets with different workload sizes. Performance measures were assessed and contrasted with conventional scheduling algorithms like FCFS, SJF, Round Robin, and Priority scheduling. These metrics included average waiting time, turnaround time, throughput, CPU utilisation, and total energy consumption. According to experimental data, the deep learning-based scheduler reliably lowers turnaround and average waiting times for both small and big task sets. The model's ability to adapt enables it to choose effective task sequences on the fly, surpassing static heuristic-based approaches.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

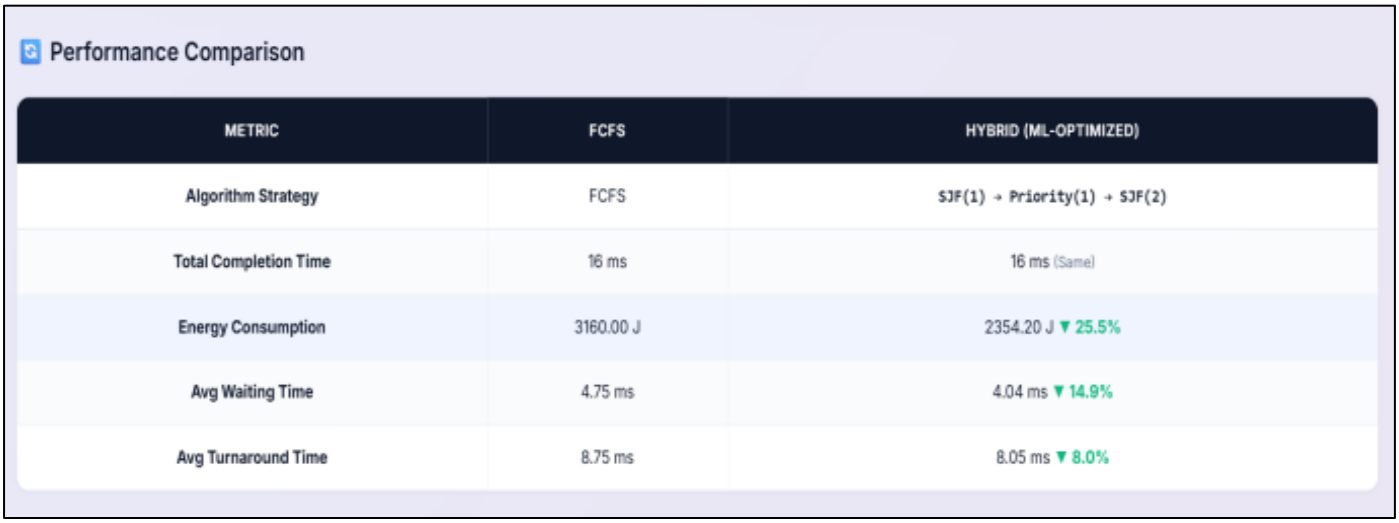

Fig 3 Hybrid Comparison

The suggested system exhibits appreciable decreases in total power usage in terms of energy efficiency because of reduced CPU idle time and regulated context switching. In contrast to Priority scheduling, which can lead to imbalance, and Round Robin, which can add overhead because of set time quanta, the intelligent scheduler keeps an execution strategy that is balanced. Regular performance gains are a result of the Multilayer Perceptron (MLP) trained with the Adam optimiser, which exhibits steady convergence and precise scheduling priority prediction. Overall, the findings confirm that deep learning can be included into CPU scheduling to improve energy optimization and computational efficiency, which makes it a viable option for contemporary high-performance and cloud computing settings.

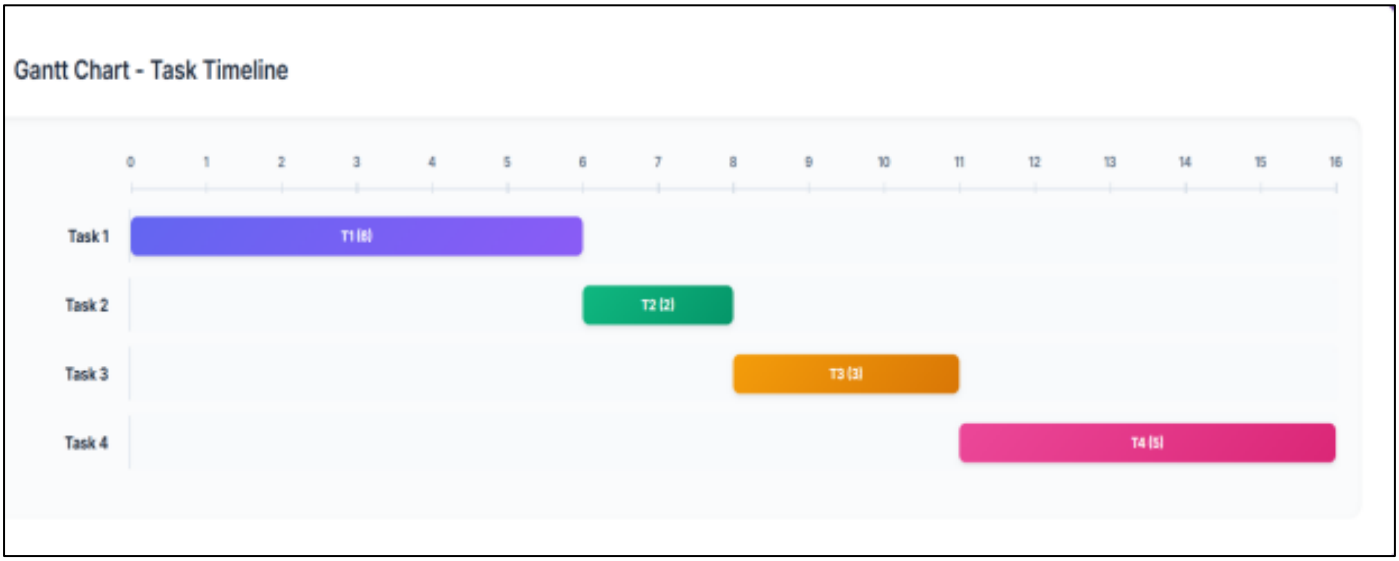

Fig 4 Gantt Chart

VIII. CONCLUSION

In order to optimise execution time and energy usage in contemporary computing systems, this article proposed a CPU scheduling framework based on deep learning. The analysis of traditional scheduling algorithms including FCFS, SJF, Round Robin, and Priority scheduling revealed both their advantages and disadvantages, such as their inability to adapt, problems with starvation, and inefficient use of energy. A Multilayer Perceptron (MLP) model trained with the Adam optimiser was incorporated into the scheduling procedure to overcome these difficulties and facilitate data-driven, intelligent decision-making.

When compared to traditional approaches, experimental evaluation showed that the suggested methodology greatly lowers average waiting time, turnaround time, and overall energy usage. The deep learning model's adaptive learning feature enables the scheduler to react to changes in workload in real time, improving system efficiency and throughput. The findings demonstrate that integrating deep learning methods into CPU scheduling improves energy efficiency and performance, making it a viable option for next-generation computing systems like data centers and cloud platforms.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar972

FUTURE ENHANCEMENT

By incorporating sophisticated neural network topologies like Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks, the suggested Deep Learning–based CPU scheduling system can be further improved to better capture temporal dependencies and sequential workload patterns. Furthermore, by using Reinforcement Learning (RL) approaches, the scheduler may interact with the system environment in real time and learn optimal policies. This will enhance scalability and robustness in dynamic computing environments by enabling the model to continuously adjust to shifting workloads without depending just on prior training data.

In order to assess the framework's viability for practical implementation, future research may also concentrate on integrating it with real-time cloud or edge computing systems. Energy optimisation may be further improved by integration with Dynamic Voltage and Frequency Scaling (DVFS) approaches. Furthermore, the model's application in massive data centers would be enhanced by expanding it to include distributed and multi-core computers. The intelligent scheduling framework for the upcoming generation of energy-conscious computing systems would be strengthened by these improvements.

REFERENCES

[1]. Prashanth Choppara, S. Sudheer Mangalampalli, “Resource Adaptive Automated Task Scheduling Using Deep Deterministic Policy Gradient in Fog Computing,” Vol. 13, 2025.

[2]. H. Janjani, T. Agarwal, M. P. Gopinath, V. Sharma, S. P. Raja, “Designing Energy-Aware Scheduling and Task Allocation Algorithms for Online Reinforcement Learning Applications in Cloud Environments,” Vol. 12, 2024.

[3]. Ying Dar Lin, Yin Tao Ling, Yuan Cheng Lai, Didik Sudyana, “Reinforcement Learning for AI as a Service: CPU-GPU Task Scheduling for Preprocessing, Training, and Inference Tasks,” Vol. 22, No. 4, pp. 3433-3448, 2025.

[4]. Velasco-Montero, Delia, Bart Goossens, Jorge Fernandez-Berni, Ángel Rodríguez-Vázquez, and Wilfried Philips. "A pipelining-based heterogeneous scheduling and energy-throughput optimization scheme for cnns leveraging apache tvm." *IEEe Access* 11 (2023): 35007-35021.

[5]. Guan, Zheng, Zengwen Wang, Yu Cai, and Xue Wang. "Deep reinforcement learning based efficient access scheduling algorithm with an adaptive number of devices for federated learning IoT systems." *Internet of Things* 24 (2023): 100980.

[6]. Jalali Khalil Abadi, Zahra, Najme Mansouri, and Mohammad Masoud Javidi. "Deep reinforcement learning-based scheduling in distributed systems: a critical review." *Knowledge and Information Systems* 66, no. 10 (2024): 5709-5782.

[7]. Li, Peisong, Ziren Xiao, Xinheng Wang, Kaizhu Huang, Yi Huang, and Honghao Gao. "EPtask: Deep reinforcement learning based energy-efficient and priority-aware task scheduling for dynamic vehicular edge computing." *IEEE Transactions on Intelligent Vehicles* 9, no. 1 (2023): 1830-1846.

[8]. Lin, Chengran, ZhengCai Cao, and MengChu Zhou. "Autoencoder-embedded iterated local search for energy-minimized task schedules of human–cyber–physical systems." *IEEE Transactions on Automation Science and Engineering* 22 (2023): 512-522.

[9]. Zhang, Lixiang, Yan Yan, and Yaoguang Hu. "Deep reinforcement learning for dynamic scheduling of energy-efficient automated guided vehicles." *Journal of Intelligent Manufacturing* 35, no. 8 (2024): 3875-3888.

[10]. Choppara, Prashanth, and Sudheer Mangalampalli. "An efficient deep reinforcement learning based task scheduler in cloud-fog environment." *Cluster Computing* 28, no. 1 (2025): 67.

[11]. Gong, Lin, Zijie Huang, Xi Xiang, and Xin Liu. "Real-time AGV scheduling optimisation method with deep reinforcement learning for energy-efficiency in the container terminal yard." *International Journal of Production Research* 62, no. 21 (2024): 7722-7742.

[12]. Chai, Sheng, and Jimmy Huang. "Dependent task scheduling using parallel deep neural networks in mobile edge computing." *Journal of Grid Computing* 22, no. 1 (2024): 27.

[13]. Hosseinzadeh, Mehdi, Elham Azhir, Jan Lansky, Stanislava Mildeova, Omed Hassan Ahmed, Mazhar Hussain Malik, and Faheem Khan. "Task scheduling mechanisms for fog computing: a systematic survey." *IEEE Access* 11 (2023): 50994-51017.

[14]. Pal, Souvik, N. Z. 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, no. 6 (2023): 5104.

[15]. Ye, Zhisheng, Wei Gao, Qinghao Hu, Peng Sun, Xiaolin Wang, Yingwei Luo, Tianwei Zhang, and Yonggang Wen. "Deep learning workload scheduling in gpu datacenters: A survey." *ACM Computing Surveys* 56, no. 6 (2024): 1-38.