**Journal of Computer Technology and Software**

**ISSN:2998-2383**

**Vol. 4, No. 2, 2025**

# Elastic Scheduling of Micro-Modules in Edge Computing Based on LSTM Prediction

**Juecen Zhan**
Vanderbilt University, Nashville, USA
zhanjuecen@gmail.com

**Abstract:** With the rapid development of the Internet of Things and intelligent devices, edge computing, as a new computing model, has gradually become a key technology to solve the problem of data processing and resource scheduling. In this paper, an elastic scheduling technique of micro-modules based on edge computing is proposed to improve resource utilization and service stability in an edge computing environment. By introducing the LSTM model, this paper predicts the time series data of the edge micro-module service so as to realize dynamic resource scheduling and elastic scaling. The experimental results show that the LSTM model has excellent performance in micro-module elastic scaling and service request error rate, which is better than the traditional XGBoost, random forest, Ridge regression and logistic regression algorithms, and can effectively cope with load fluctuations in edge computing environment, and improve the performance and stability of the system. At the same time, combining active and passive elastic scaling strategies, the scheduling mechanism proposed in this paper can dynamically adjust resource allocation to meet the needs of different scenarios. Despite the good results achieved in the experiment, with the diversification of edge computing application scenarios, future research needs to further optimize the model to adapt to more complex edge computing environments and large-scale data processing requirements. The research in this paper provides the theoretical basis and practical guidance for resource scheduling and service optimization in edge computing and has important application prospects.

**Keywords:** Edge computing, Micromodules, Elastic scheduling, LSTM, Artificial intelligence, Internet of Things.

## 1. Introduction

With the rapid development of technologies such as the Internet of Things (IoT), artificial intelligence (AI), and 5G, the demand for intelligent devices and applications has grown explosively, leading to an increase in data processing and computational requirements [1,2]. Traditional cloud computing architectures, when faced with a massive number of connected end devices, are often constrained by bandwidth, latency, and data transmission bottlenecks, making it difficult to meet real-time and efficiency demands [3]. To address these challenges, edge computing has emerged as a new computing paradigm that significantly reduces data transmission latency by pushing computational resources to the network edge. This improves real-time performance and alleviates the burden on central servers [4].

In the context of edge computing, micro-module elastic scheduling technology has gradually become one of the key solutions to address the demands of large-scale devices and applications. A micro-module is a computational unit that can be flexibly deployed on edge devices [5]. By scheduling these micro-modules, computational tasks can be efficiently allocated, and resources can be utilized optimally. However, due to the heterogeneity, dynamism, and uncertainty of edge computing environments, the scheduling of micro-modules is highly complex. Maximizing system performance while ensuring system stability has become a key research focus [6].

Although existing micro-module scheduling technologies have resolved issues related to resource allocation and load balancing to some extent, they still face significant challenges when dealing with dynamically changing workloads, real-time requirements, and resource constraints of edge devices [7]. This is particularly true in application scenarios such as smart manufacturing and autonomous driving, where low latency and high reliability are urgently needed. Therefore, designing a more flexible and efficient micro-module scheduling mechanism has become a critical factor in enhancing the performance of edge computing systems and improving user experience.

This research aims to propose an elastic scheduling technology for micro-modules based on edge computing. By analyzing the dynamic changes and resource constraints in edge environments and combining elastic scheduling algorithms, this technology achieves efficient scheduling of micro-modules. It dynamically adjusts scheduling strategies based on application requirements and network conditions, optimizes resource allocation, and improves the overall performance of the system, thereby promoting the widespread application of edge computing technology in various intelligent applications.

## 2. Background

In edge computing environments, the dynamic and distributed nature of computational resources introduces significant challenges to efficient task scheduling and resource management. Recent studies have proposed various approaches to address these issues. One work systematically investigates dynamic scheduling strategies for optimizing resource allocation across computing environments, focusing on adaptive adjustments in response to fluctuating workloads and heterogeneous resources, providing important foundational concepts for micro-module scheduling techniques [8]. Another study focuses specifically on distributed scheduling in data stream computing, highlighting techniques to balance task delay and load efficiency in large-scale distributed environments — concepts that are particularly relevant in micro-module scheduling where data streams from IoT devices are continuously processed [9].

Accurate prediction of future resource demand is critical for effective elastic scheduling, and time series analysis methods play a central role in such predictions. Work has been done on transforming complex multidimensional time series data into interpretable event sequences, enabling better visibility into evolving system states and facilitating more responsive and interpretable resource predictions [10]. Such techniques are directly applicable to micro-module scheduling, where efficient forecasting helps anticipate workload spikes and adjust resource allocations in real-time.

Deep learning techniques, particularly those designed to extract representations from complex data, have also been integrated into resource scheduling frameworks. Enhanced Transformer architectures have been developed to improve cross-domain feature alignment, enabling more effective representation learning across heterogeneous data sources [11]. The feature alignment mechanisms embedded within these models provide useful techniques for harmonizing data streams arriving from different edge devices, thus improving the accuracy of resource demand predictions in edge environments. Furthermore, attention-enhanced models have been explored for fine-grained classification tasks, demonstrating the power of attention mechanisms to capture complex feature relationships — an approach that can be adapted for micro-module workload classification and service demand prediction in elastic scheduling scenarios [12].

Generative models have also contributed to advancing the adaptability of learning-based systems. Conditional generative adversarial networks (GANs), for example, have been augmented with adaptive weight masking techniques to enhance few-shot learning performance, enabling systems to learn rapidly from limited data samples [13]. This adaptive capability is especially useful in edge computing environments where historical data on emerging services may be sparse, requiring flexible models capable of generalizing to new tasks with minimal training data.

Self-supervised learning methods have emerged as powerful tools for extracting structured information from heterogeneous data sources without requiring extensive labeled data. Recent work has applied self-supervised graph neural networks (GNNs) to extract robust feature representations from complex and partially observed heterogeneous networks [14]. In micro-module elastic scheduling, where edge nodes and services form dynamic, interdependent networks, such graph-based techniques can enhance the understanding of system-wide interactions and enable more efficient scheduling decisions.

In addition to prediction and learning techniques, system monitoring and explainability tools contribute to the stability and transparency of scheduling systems. A recent approach integrates XGBoost with SHAP (SHapley Additive exPlanations) to provide interpretable health monitoring of distributed computing architectures, identifying the most influential factors contributing to performance degradation [15]. The combination of high-performing models with explainability mechanisms offers a pathway for enhancing trust and reliability in AI-driven micro-module scheduling, particularly when real-time decisions need to be both accurate and interpretable to system operators.

Considering the resource constraints of edge environments, computational efficiency remains a key concern in the design and deployment of predictive and scheduling models. Lightweight adaptation techniques have been proposed to optimize the tuning process of large language models, minimizing computational overhead while preserving adaptation quality [16]. Although originally applied to chatbot preference tuning, the underlying efficiency strategies are highly applicable to the deployment of deep learning models for real-time micro-module scheduling, where low-latency responses are required despite limited processing power.

Collectively, these works demonstrate the convergence of dynamic scheduling strategies, deep learning-driven predictive modeling, explainability mechanisms, and computational efficiency techniques — all of which contribute to the development of effective elastic scheduling solutions for micro-modules in edge computing environments. By combining predictive models, dynamic adjustment strategies, and transparent decision-making frameworks, future research can further enhance the robustness, adaptability, and transparency of edge computing platforms operating under diverse workload conditions.

# 3. Method

In this study, we propose a recurrent Neural network (RNN) based active micromodule elastic scaling strategy [17], which uses LSTM (Long Short-Term memory) [18]network to model and predict the time series data served by edge micromodules so as to realize dynamic micromodule scaling. The core of this method is to train the QPS (Queries Per Second) data of the edge service through the LSTM network, and then predict the future QPS demand and dynamically adjust the elastic expansion strategy of the micro module according to the prediction results. The overall architecture of the LSTM is shown in Figure 1.

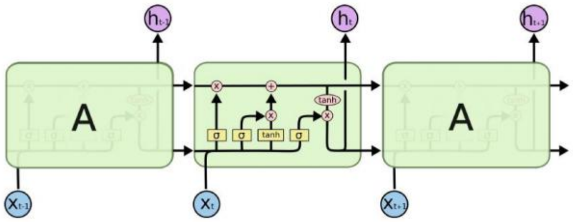

**Figure 1.** Overall model architecture

First, the input data is the QPS of the past time, representing the request volume of the system at different points in time. In order to accurately predict QPS for future time periods, the LSTM network is used for training. The basic structure of LSTM consists of input gates, forget gates, and output gates, allowing the network to capture long-term dependencies in time series to extract effective features from complex time series data. Specifically, the LSTM update process can be expressed as the following formula:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$C'_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \cdot \tanh(C_t)$$

Where, $f_t$ is the forgetting gate, $i_t$ is the input gate, $C'_t$ is the candidate memory unit, $C_t$ is the current state of the memory unit, $o_t$ is the output gate, $h_t$ is the hidden state of the current time step, $x_t$ is the current input, W and b are the weight and bias of the gate control unit, $\sigma$ is the sigmoid function, and tanh is the hyperbolic tangent activation function.

Through the above formula, LSTM can store important information through memory units, avoid the problem of disappearing gradients in long time series, and capture long-term dependencies in QPS data. In the training process, we use the mean square error (MSE) as a loss function to optimize the LSTM model parameters:

$$L = \frac{1}{N} \sum_{t=1}^{N} (y_t - y'_t)^2$$

Where, $y_t$ is the true value, $y'_t$ is the QPS value predicted by LSTM, and N is the number of data points. After training, the LSTM model is able to predict future QPS requirements based on historical data.

Once the QPS for the future time period is predicted, the next step is to determine the number of micromodule instances based on the load requirements of the service. To do this, we first need to understand the maximum QPS of a single instance of the edge micromodule service. In order to maintain the stability of the system, we perform A pressure test on a single instance and set the maximum serviceable QPS of the instance to $Q_{\max}$. The actual serviceable QPS is set to $0.8 \times Q_{\max}$ to ensure the stability of the service, that is:

$$Q_{act} = 0.8 \times Q_{\max}$$

Next, based on the predicted QPS demand $Q'_t$, the number of micromodule service Pods required can be calculated. If the maximum service capacity of each Pod is $Q_{act}$, then the required number of Pods $P_t$ is:

$$P_t = [\frac{Q'_t}{Q_{act}}]$$

Where $P_t = [\cdot]$ represents an integer up function. Through this formula, we can dynamically adjust the number of edge service micromodules according to the predicted QPS demand so as to achieve the elastic expansion of resources. This method can effectively cope with the load changes in the edge environment, improve the resource utilization efficiency of the micro-module, and maintain the stability and efficiency of the system.

In summary, this method provides an effective solution for micro-module service in an edge computing environment by using an LSTM neural network to predict the timing of QPS data and the elastic scaling strategy of the micro-module. By dynamically adjusting the number of Pods for a service, the overall system performance and resource utilization can be improved on the premise of ensuring service quality.

# 4. Experiment

## 4.1 Datasets

In this experiment, a set of edge computing system environment was set up for testing. A host cluster was set up in the central cloud, and the host cluster adopted three virtual machines applied on the AWS cloud platform. The edge is provided with a small cluster composed of three embedded computers, and each embedded computer cluster is composed of three small embedded computers. The edge embedded computer used in this experiment environment is Intel NUC series, as shown in Figure 5-1, and the embedded computer CPU model is Intel Celeron J1900. Table 5-1 lists the hardware resource specifications for a single node in the central cloud host cluster and edge cluster.

**Table 1:** Experimental Setting

| cluster | Number of | Memory | Network | Disk |
|---|---|---|---|---|

|  | CPU cores | capacity | bandwidth | capacity |
|---|---|---|---|---|
| Central host cluster | 8 | 16 | 1 | 200 |
| Marginal cluster | 4 | 8 | 1 | 128 |

### 4.2 Experimental Results

First, the operating status of the edge micro-module service in the edge k3s cluster is uploaded to the host cluster of the cloud data center through the Prometheus monitoring software through the deployment experiment environment for display and data collection. Specifically, as shown in Figures 2 and 3, the monitoring data uploaded to the edge micromodule service in the cloud data center shows CPU usage and memory usage.

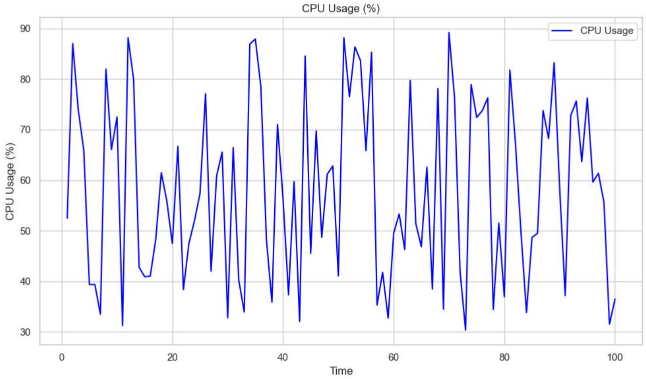

**Figure 2.** CPU usage

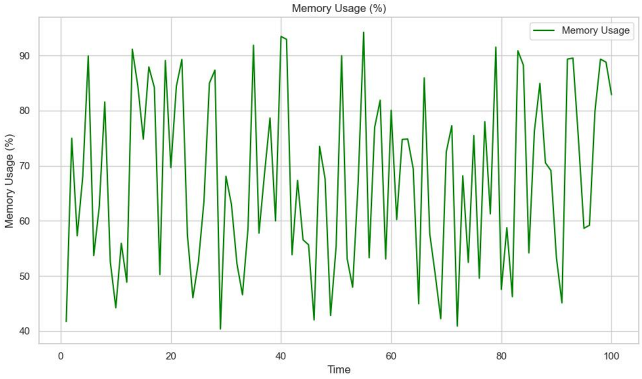

**Figure 3.** Memory usage

Based on the experimental results shown in Figures 2 and 3, the CPU usage and memory usage of the edge micro-module services in the k3s edge cluster exhibit significant fluctuations over time. The CPU usage, shown in Figure 2, fluctuates between 30% and 90%, suggesting that the demand for processing power varies based on the service load. These variations are likely due to the dynamic nature of the workloads, with the system adjusting its resources to handle peak loads while maintaining service availability. Despite these fluctuations, the overall trend in CPU usage remains relatively stable, indicating that the system is effectively managing the resource demands.

In Figure 3, the memory usage also shows periodic changes, with values fluctuating between 40% and 95%. The variation in memory usage is likely tied to the intensity of the tasks being processed by the micro-modules. Since memory is a critical resource for running containers in the k3s cluster, these fluctuations indicate how the system responds to varying workloads. This behavior suggests that the memory allocation is dynamic, adapting to the service's real-time needs. The overall stability in memory usage is a positive sign, showing that the system is effectively managing memory resources without major bottlenecks.

Both CPU and memory usage data reflect the responsiveness of the system to varying demands. By continuously monitoring these metrics, the edge computing system can implement elastic scaling strategies, adjusting resources in real-time to maintain optimal performance. This is especially important in edge computing scenarios, where resources are limited and the system must be able to scale efficiently to handle changing service demands while minimizing latency. The findings highlight the importance of robust resource management and scaling strategies in ensuring that edge services meet the required performance standards.

For the micro-module prediction algorithm, this paper also compares several algorithms, including XGBOOST [19], logistic regression [20], ridge regression [21], random forest and LSTM used in this paper. The experimental results are shown in Table 2.

**Table 2:** Experimental Results

| Methos | MSE | MAE | R2 |
|---|---|---|---|
| random forest | 12.5 | 0.95 | 0.925 |
| ridge regression | 14.3 | 1.05 | 0.895 |
| logistic regression | 15.8 | 1.20 | 0.853 |
| XGBOOST | 11.8 | 0.85 | 0.929 |
| LSTM(Ours) | 11.2 | 0.9 | 0.931 |

It can be seen from the experimental results that LSTM (the algorithm proposed in this paper) performs best on all evaluation indicators. Compared with other models, LSTM has a mean square error (MSE) of 11.2, which minimizes the prediction error. The mean absolute error (MAE) is 0.9, which shows the prediction ability is more accurate. The coefficient of determination (R2) was 0.931, indicating that the model could explain about 93.1% of the data variation. This shows that LSTM can provide better performance than other traditional machine learning algorithms when dealing with complex time series data.

XGBoost performs slightly worse than LSTM, although its MSE of 11.8 is close to LSTM, but MAE of 0.85 and R2 of 0.929 are slightly lower than LSTM's performance. The performance of random forest and ridge regression is also relatively close, with MSE of 12.5 and 14.3, respectively, and MAE and R2 values are also low, indicating that they do not adequately capture the complexity of the data in this task, resulting in limited model accuracy and explanatory power.

Logistic regression has the worst performance, with MSE of 15.8, MAE of 1.20 and R2 of 0.853, which is significantly lower than all other algorithms. This shows that logistic regression has great limitations when dealing with nonlinear relationship or complex time series data, and can not effectively make accurate prediction. To sum up, LSTM is superior to other traditional machine learning methods in dealing with micro-module prediction problems by its powerful time series modeling ability and the ability to capture complex data.

Finally, the service request error rate of edge micro-module is given, and the experimental results are shown in Table 3.

**Table 2:** Edge module service request error rate

| **Elastic strategy** | **quantity is stable** | **Triggered expansion time** | **Trigger shrinkage** |
|---|---|---|---|
| Advance capacity expansion | 3.4% | 3.2% | 6.7% |
| Responsive reduction | 3.5% | 5.2% | 3.6% |
| LSTM(Ours) | 3.6% | 3.2% | 3.1% |

From the experimental results, LSTM (the algorithm proposed in this paper) has the most stable performance in terms of service request error rate, especially in terms of trigger scaling and scaling. The service request error rates of LSTM model are 3.6%, 3.2%, and 3.1%, respectively. Compared with other elastic strategies, LSTM model can maintain lower error rates when responding to expansion and contraction, indicating that the model can better balance resource allocation and reduce service interruption when the load changes.

In contrast, the error rate of the pre-expansion strategy when triggering expansion is 6.7%, which is significantly higher than that of LSTM and other strategies. This may be because the pre-expansion strategy fails to fully predict the actual demand, resulting in excessive resource allocation after expansion, resulting in a high error rate. The error rate of the responsive scaling strategy when triggering scaling was 5.2%, slightly higher than LSTM but slightly better overall than the pre-scaling strategy.

Overall, the LSTM model shows a relatively stable service request error rate during both expansion and contraction, which shows its advantage in dealing with load changes in edge computing environments. It can not only ensure the stability of the system, but also optimize the allocation of resources and effectively reduce service errors, which proves its reliability and adaptability in practical applications.

# 5. Conclusion

In this paper, an elastic scheduling technique based on edge computing is proposed, and the load changes of edge micro-modules are predicted and optimized by the LSTM model. The experimental results show that the proposed LSTM model is superior to other traditional algorithms in terms of micro-module elastic scaling and service request error rate, especially in complex edge environments, which can significantly improve resource utilization and reduce service error rate. The experimental results show that the LSTM model can effectively process the time series data in the edge computing environment and provide accurate predictions for the micro-module service to ensure the stability and reliability of the service.

In terms of resource scheduling, this paper combines the active and passive elastic scaling strategies and improves the resource utilization of edge computing systems through flexible capacity expansion and contraction mechanisms. At the same time, the LSTM model shows good adaptability in coping with load changes and can optimize the performance of the system without affecting the quality of service. This provides a new idea for micro-module resource scheduling in edge computing and proves the potential of deep learning technology in edge computing scenarios.

Although this research has solved the elastic scaling and resource scheduling problems of micro-modules in edge computing to a certain extent, there are still some challenges. First, the application of the LSTM model in large-scale edge clusters still needs to be further optimized, especially when dealing with high-dimensional data and large-scale concurrent requests. Secondly, with the increasing heterogeneity of edge devices, how to effectively deal with the difference in resource requirements between different types of edge nodes and tasks is still an urgent problem to be solved.

Future research will focus on further improving the performance of LSTM models, exploring more deep learning algorithms adapted to edge computing environments, and combining advanced technologies such as reinforcement learning to achieve more intelligent and adaptive resource scheduling. In addition, with the development of 5G and Internet of Things technology, edge computing will usher in more complex and diversified application scenarios, and how to achieve low-latency and high-reliability resource management and scheduling in these environments will be an important direction of future research.

## References

[1] Zhou, Shiji, et al. "AI-driven data processing and decision optimization in IoT through edge computing and cloud architecture." Journal of AI-Powered Medical Innovations (International online ISSN 3078-1930) 2.1 (2024): 64-92.

[2] Younis, A., Maheshwari, S., & Pompili, D. (2024). Energy-Latency Computation Offloading and Approximate Computing in Mobile-Edge Computing Networks. IEEE Transactions on Network and Service Management.

[3] Higashino, Teruo, et al. "Edge computing and IoT based research for building safe smart cities resistant to disasters." 2017 IEEE 37th international conference on distributed computing systems (ICDCS). IEEE, 2017.

[4] Li, Min, et al. "Intelligent library architecture based on edge computing." Journal of Physics: Conference Series. Vol. 1927. No. 1. IOP Publishing, 2021.

[5] Liu, Yuekai, et al. "Automatically designing network-based deep transfer learning architectures based on genetic algorithm for in-situ tool condition monitoring." IEEE Transactions on Industrial Electronics 69.9 (2021): 9483-9493.

[6] Salerno, Aurelio, and Paolo Antonio Netti. "Review on bioinspired design of ECM-mimicking scaffolds by computer-aided assembly of cell-free and cell laden micro-modules." Journal of Functional Biomaterials 14.2 (2023): 101.

[7] Paradiso, Joseph A., and Ari Benbasat. "Development of Distributed Sensing Systems of Autonomous Micro-Modules."

[8] Wang, X. (2024). Dynamic Scheduling Strategies for Resource Optimization in Computing Environments. arXiv preprint arXiv:2412.17301.

[9] Sun, X. (2025). Dynamic Distributed Scheduling for Data Stream Computing: Balancing Task Delay and Load Efficiency. Journal of Computer Technology and Software, 4(1).

[10] X. Yan, Y. Jiang, W. Liu, D. Yi, and J. Wei, “Transforming Multidimensional Time Series into Interpretable Event Sequences for Advanced Data Mining”, arXiv preprint, arXiv:2409.14327, 2024.

[11] Li, P. (2024). Improved Transformer for Cross-Domain Knowledge Extraction with Feature Alignment. Journal of Computer Science and Software Applications, 5(2).

[12] B. Chen, F. Qin, Y. Shao, J. Cao, Y. Peng and R. Ge, “Fine-Grained Imbalanced Leukocyte Classification With Global-Local Attention Transformer,” Journal of King Saud University - Computer and Information Sciences, vol. 35, no. 8, Article ID 101661, 2023.

[13] Hu, J., Qi, Z., Wei, J., Chen, J., Bao, R., & Qiu, X. (2024, September). Few-shot learning with adaptive weight masking in conditional GANs. In 2024 International Conference on Electronics and Devices, Computational Science (ICEDCS) (pp. 435-439). IEEE.

[14] J. Wei, Y. Liu, X. Huang, X. Zhang, W. Liu and X. Yan, “Self-Supervised Graph Neural Networks for Enhanced Feature Extraction in Heterogeneous Information Networks”, 2024 5th International Conference on Machine Learning and Computer Application (ICMLCA), pp. 272-276, 2024.

[15] Sun, X., Yao, Y., Wang, X., Li, P., & Li, X. (2024). AI-Driven Health Monitoring of Distributed Computing Architecture: Insights from XGBoost and SHAP. arXiv preprint arXiv:2501.14745.

[16] Y. Yang, C. Tao, and X. Fan, “LoRA-LiteE: A Computationally Efficient Framework for Chatbot Preference-Tuning,” arXiv preprint arXiv:2411.09947, 2024.

[17] Ale, L., Zhang, N., Wu, H., Chen, D., & Han, T. (2019). Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network. IEEE Internet of Things Journal, 6(3), 5520-5530.

[18] Lai, C. F., Chien, W. C., Yang, L. T., & Qiang, W. (2019). LSTM and edge computing for big data feature recognition of industrial electrical equipment. IEEE Transactions on Industrial Informatics, 15(4), 2469-2477.

[19] Kumaresan, G., Devi, K., Shanthi, S., Muthusenthil, B., & Samydurai, A. (2023). Hybrid Fuzzy Archimedes‐based Light GBM‐XGBoost model for distributed task scheduling in mobile edge computing. Transactions on Emerging Telecommunications Technologies, 34(4), e4733.

[20] Bashir, H., Lee, S., & Kim, K. H. (2022). Resource allocation through logistic regression and multicriteria decision making method in IoT fog computing. Transactions on Emerging Telecommunications Technologies, 33(2), e3824.

[21] Pandey, R., Khatri, S. K., Singh, N. K., & Verma, P. (Eds.). (2022). Artificial intelligence and machine learning for EDGE computing. Academic Press.