UDC: 004.75:004.8:004.94 https://doi.org/10.15407/jai2026.02.228

O. Stefura[1], A. Petrenko[2]
[1,2] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
37 Beresteiskyi Ave., Kyiv, 03056
[1] stefura.3.1@gmail.com
[2] tolja.petrenko@gmail.com
[1] https://orcid.org/0009-0005-5962-1025
[2] https://orcid.org/0000-0001-6712-7792

INTELLIGENT METHODS FOR PROACTIVE MICROSERVICE ORCHESTRATION: ANALYSIS OF STATE-OF-THE-ART DEEP LEARNING APPROACHES

Abstract. The rapid and continuous advancement of cloud computing, coupled with increasingly stringent requirements and constraints on cloud services, continues to drive intense research activity in this field. A particularly critical area of study is resource management in cloud environments, which exerts a direct and significant impact on both Quality of Service (QoS) and operational costs. Current mainstream orchestration solutions (such as the Kubernetes Horizontal Pod Autoscaler, HPA) remain predominantly reactive in nature. Consequently, they suffer from resource inefficiencies (over-provisioning and under-provisioning), scaling latencies, and performance degradation. Thus, there is a clear necessity for robust proactive solutions capable of optimizing costs and minimizing scaling delays.

This paper provides a comprehensive analysis of state-of-the-art resource orchestration methods in cloud environments and systematizes current approaches to workload forecasting. The efficacy of hybrid models (such as ARIMA-LSTM and Prophet-LSTM) is examined, alongside an analysis of the role of Attention mechanisms and Transformers in the context of multivariate workload prediction. Particular emphasis is placed on the application of Graph Neural Networks (GNNs) as a highly promising direction for modeling complex microservice topologies and predicting latency while accounting for inter-component interactions. Finally, the bottlenecks of existing solutions are identified, and potential vectors for future research are proposed.

Keywords: cloud computing, microservices, proactive orchestration, Long Short-Term Memory (LSTM), Transformers, Graph Neural Networks (GNN), workload forecasting, Kubernetes.

1. Introduction

The rapid evolution of cloud solutions and the widespread adoption of microservice architectures have fundamentally transformed infrastructure management. Modern, large-scale, high-load systems often comprise hundreds or even thousands of interconnected services operating in dynamic environments, typically orchestrated by Kubernetes. In such a landscape, the efficient management of limited resources has become a primary challenge in maintaining system stability, both in terms of ensuring high Quality of Service (QoS) and optimizing operational expenditures (OPEX) [1].

Traditional reactive approaches, which rely on threshold-based triggers for target metrics (such as CPU and RAM utilization), prove inadequate under unpredictable and bursty traffic conditions. The most prevalent issue is the "time lag"—the interval between reaching a critical resource threshold and the full availability of newly provisioned resources. This delay frequently results in Service Level Objective (SLO) violations, increased latency, and, in many cases, cascading failures.

To address these deficiencies, the scientific community is increasingly focusing on proactive orchestration. The integration of machine learning models enables the effective forecasting of future workload levels, allowing for the preemptive allocation of necessary resources. This proactive stance effectively mitigates the inherent risks associated with the reactive model.

Recent years have seen a surge in the development of hybrid architectures that combine established machine learning models (such as LSTM and Bi-LSTM) with statistical frameworks (ARIMA/SARIMA), significantly enhancing prediction accuracy. Furthermore, there is growing interest in the application of

Graph Neural Networks (GNNs), which allow for the consideration of not only temporal patterns but also structural aspects—specifically, the complex topology of inter-service dependencies.

2. Reactive Orchestration

Reactive orchestration is a classical resource management paradigm and currently remains the industry standard across most cloud environments, particularly within the Kubernetes ecosystem [2, 3]. The core principle is based on a feedback loop, where the system responds to infrastructure state changes only after they have been detected. The popularity of this approach stems primarily from its versatility, ease of implementation, and deterministic nature.

The primary mechanisms of reactive scaling include:

- **Horizontal Scaling (Sharding):** The system periodically samples metrics and recalculates the required number of replicas. The main drawback is latency (inertia). The interval between detecting the need for a new container and its full operational readiness is a period during which the QoS degrades.
- **Vertical Autoscaling:** The number of containers remains constant, while the amount of resources (RAM/CPU) allocated to each is adjusted. Although this method helps avoid resource fragmentation, most current implementations require a component restart, leading to temporary downtime.
- **Event-Driven Scaling (e.g., KEDA):** This allows the system to respond to specific external triggers (such as Kafka queue length) rather than just classical resource metrics. While this extends system capabilities, it does not eliminate its inherently reactive nature.

A brief comparison of these methods is presented in Table 1:

Table 1. Comparison of Reactive Scaling Mechanisms

Method	Trigger	Response Time	Primary Limitation
Horizontal Scaling (HPA)	Resource Metrics (CPU/RAM)	Moderate (Minutes)	Lagged response and "cold start" issues
Vertical Scaling (VPA)	Resource Utilization	High (Node/Pod restart)	Component restart requirement (downtime)
Event-driven (KEDA)	External Events (Queue length, HTTP traffic)	Low to Moderate	Reactive nature (responds after request accumulation)

Overall, while reactive mechanisms remain pivotal in ensuring the resilience of microservice systems by enabling dynamic load balancing across identical service replicas [4], they possess several fundamental limitations. In the context of high-load and fault-sensitive systems, these constraints become critical:

- **Reactive Lag:** The delay between the emergence of a resource need and its detection, primarily dictated by the metrics polling interval.
- **"Cold Start":** The non-trivial time required for a new component to undergo initialization before it can actively process traffic.
- **Flapping/Thrashing Effect:**

A phenomenon where, if traffic fluctuates near a critical threshold, the system enters rapid, inefficient cycles of adding and removing containers.

- **Over-provisioning Risk:** Due to a lack of confidence in the responsiveness of reactive systems, developers often set excessively high resource limits or overly sensitive scaling thresholds as a safety margin.

3. Proactive Orchestration

Proactive orchestration addresses the shortcomings of the reactive model by allocating necessary resources in advance. Consequently, by the time the demand for additional capacity

arises, new service replicas are already provisioned, initialized, and ready to participate in request processing.

However, a universal "silver bullet" for precise workload forecasting does not yet exist. This limitation creates a broad landscape for research, encouraging the application of diverse methods and tactics to achieve optimal prediction accuracy and system stability.

3.1 Statistical Methods

Statistical time-series forecasting methods remain a fundamental tool in the field of microservice workload prediction due to their mathematical rigor, interpretability, and transparency. Their key advantage over deep learning models is a significantly lower demand for computational resources. However, while demonstrating high efficiency in approximating linear dependencies, statistical models cannot fully capture the complex dynamics of modern microservice systems, which are characterized by non-linear inter-component relationships and unpredictable activity spikes [5]. Consequently, they are predominantly utilized as components of hybrid models.

Even adaptive methods like Facebook Prophet, which effectively handle complex seasonality, are increasingly viewed in contemporary research as parts of hybrid architectures [6]. This approach allows for combining the mathematical stability of a statistical baseline with the power of neural networks in modeling anomalous workload fluctuations—a synergy critical for meeting Service Level Objectives (SLOs).

Currently, the role of statistical methods in workload forecasting is primarily focused on two directions:

1. **Baselines:** Utilizing statistical models as a benchmark to evaluate the performance of more complex intelligent algorithms [5].
2. **Hybridization:** Implementing combined schemes where ARIMA or ETS-class models extract linear trends and seasonality, while the regression residuals (non-linear anomalies that the model failed to identify) are predicted using Recurrent Neural Networks (LSTM/GRU), followed by result aggregation [7, 8].

The general concept of such hybrid decomposition can be represented by the following relationship:

$$Y_t = L_t + S_t + N_t + \epsilon_t$$

Where:

- L_t,S_t – represent the linear components (trends and seasonality, respectively), processed using statistical methods.
- N_t – represents non-linear disturbances, processed by the neural network architecture.
- ϵ_t – represents random noise.

The general workflow of hybrid architectures based on statistical models and RNN is shown in Figure 1.

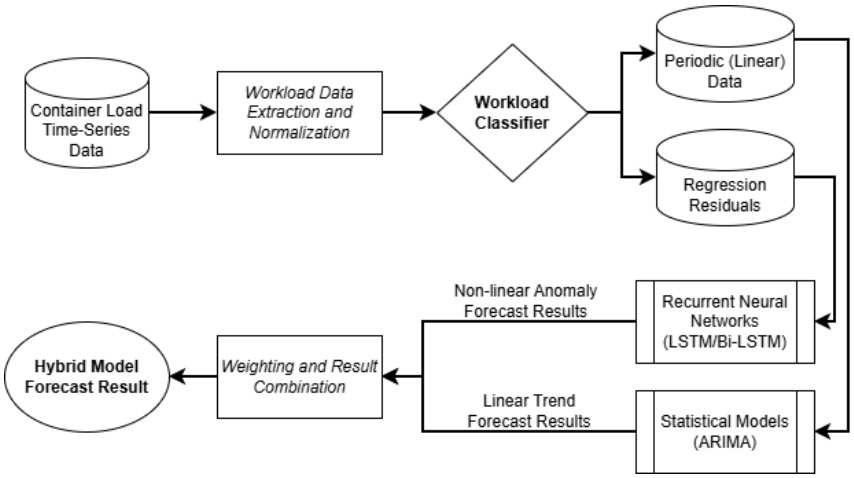

Figure 1. General architecture of hybrid forecasting approaches based on statistical models and Recurrent Neural Networks (RNNs)

However, despite the advancement of hybrid models, the standalone application of statistical methods remains relevant in distributed systems with a high degree of decomposition. At the level of individual microservices, their low computational complexity enables the implementation of proactive resource management without imposing an excessive overhead on the infrastructure [9]. This efficiency is a decisive factor in achieving high container density and optimizing operational costs within cloud environments.

3.2 Machine Learning Methods

3.2.1 Recurrent Neural Networks (RNN) and LSTM Architectures in Time-Series Forecasting

In the context of machine learning-based orchestration and automated scaling, Recurrent Neural Networks (RNNs)—particularly their widely adopted modifications, Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM)—hold a prominent position [1]. Currently, these architectures serve as the de facto standard for time-series analysis in cloud environments. They demonstrate superior efficiency compared to classical deep learning models due to their inherent ability to capture long-term dependencies within data [10].

The current research trajectory is focused on the development of hybrid models that integrate neural network architectures with statistical methods to enhance forecasting accuracy:

- **Hybridization with Statistical Methods:** Hongyuan et al. [7] proposed a workload decomposition method that separates periodic (cyclic) and non-periodic (impulse bursts) components using a classifier. The authors employ an ARIMA model to forecast linear patterns, while a Bi-LSTM network is used to process non-linear spikes. Result aggregation is performed via a dynamic weighting mechanism. This approach improved CPU utilization forecasting accuracy by 25.19% compared to standalone ARIMA and by 37.60% compared to Bi-LSTM.
- **Integration with Cloud Orchestrators:** Vasumathi et al. [11] implemented the integration of an LSTM model with the standard Kubernetes Horizontal Pod Autoscaler (HPA) mechanism. This integration enabled the transformation of reactive scaling into a proactive strategy. Experimental results indicate a 28% reduction in system response time, a 50% decrease in scaling latency, and a 15% reduction in infrastructure energy consumption.

Within the class of recurrent networks, the Gated Recurrent Unit (GRU) architecture attracts particular attention. It is positioned as a computationally efficient alternative to LSTM, which, due to its simplified structure, often provides faster convergence with comparable or even superior forecasting accuracy [12].

Another promising direction is the combination of Convolutional Neural Networks (CNNs) and recurrent networks:

1. Sabyasachi et al. [13] utilize CNNs to detect complex local patterns ("spatial" features) within workload metrics, which LSTMs subsequently interpret within a temporal context.

2. Zhang et al. [14] apply a similar approach to analyze multivariate data (task IDs, node IDs, instruction cycles). The concurrent use of CNN and Bi-LSTM reduced the Mean Squared Error (MSE) by 34–52% compared to baseline LSTM and Bi-LSTM models, confirming the validity of spatial feature extraction prior to the recurrent processing stage.

An analysis of contemporary research [11, 13-14] confirms that recurrent architectures, specifically LSTM and Bi-LSTM, are the most adaptable tools for workload modeling. Simultaneously, several key trends can be identified:

1. **The Necessity of Hybridization:** Standalone RNN models often prove insufficient for processing heterogeneous data. Effective solutions involve combining neural networks with statistical methods (e.g., ARIMA) to separate linear and non-linear traffic components [7] or utilizing CNNs for the preliminary extraction of spatial features and

correlations between various metrics (CPU, RAM, I/O) [13, 14].

2. **Transition to Proactive Management:** Integrating predictive models directly into orchestrator control loops (specifically through extensions of the standard Kubernetes HPA [11]) demonstrates significant potential in reducing scaling latencies and optimizing energy consumption.

3. **Accounting for Global Microservice Context:** Modern approaches move beyond the analysis of isolated time series by incorporating data on inter-service interactions [15] and multivariate Attention mechanisms [12], enabling more accurate prediction of cascading workloads in complex cloud-native architectures.

3.2.2 Transformers and Attention Mechanisms in Workload Forecasting

Contemporary research (2024–2026) demonstrates a distinct transition from classical recurrent networks (RNN/LSTM) toward architectures based on Attention mechanisms. This shift is driven by their superior ability to model long-range temporal dependencies within bursty workloads and complex spatial inter-service relationships.

Chauhan et al. [16] proposed a hybrid CNN-Transformer model aimed at optimizing data center energy consumption. This architecture implements a synergy between two approaches:

- **CNNs:** Responsible for extracting local patterns and spatial features from historical data.
- **Transformers:** Utilize Self-Attention mechanisms to analyze global contexts and long-term workload trends.

This approach significantly reduces the carbon footprint of cloud computing through more precise predictive resource scaling. The model achieved a Mean Squared Error (MSE) of 0.0059 and a Mean Absolute Error (MAE) of 0.0381, outperforming existing state-of-the-art (SOTA) models.

Kumar et al. [17] extend the classical MAPE (Monitor-Analyze-Plan-Execute) autoscaling paradigm by integrating a multivariate transformer (MV-Transformer) into the analysis phase. Key findings of their study include:

1. **Elastic Acceleration (EA) Metric:** The authors introduce the EA metric, which evaluates the speed and precision of system adaptation to workload changes. The MV-Transformer demonstrated a significant advantage over industry standards:
   - MV-Transformer: 2.9818
   - Bi-LSTM / LSTM: 1.3200 / 1.0230
   - Reactive Approach: 1.0000
2. **Computational Resource Optimization:** Beyond higher accuracy, the MV-Transformer proved to be less memory-intensive compared to traditional RNN models, which is a critical factor for real-time systems.

3.2.3 Graph Neural Networks (GNN) for Microservice Interaction Modeling

The integration of graph technologies into microservice architectures is a logical progression, as the topology of such systems inherently corresponds to a graph structure. The use of Graph Neural Networks (GNNs) enables a transition from the isolated analysis of individual containers to a system-centric approach that accounts for the global context of inter-component relationships.

Current research demonstrates the high efficiency of GNNs in workload forecasting and resource optimization tasks:

- **Modeling Cascading Effects:** In [18], GNNs are utilized to analyze a graph representation of the system based on distributed tracing data. The model predicts how changes in input workload affect each individual service under various resource configurations. This approach achieves CPU savings of 14–19% compared to enhanced HPA versions, and up to 36% during peak loads.
- **Spatio-Temporal Analysis:** Luo et al. [19] integrated the system state into a spatio-temporal GNN to detect "spillover" latency patterns along the call chain. By combining Attention mechanisms and Temporal Convolutional Networks (TCNs), the authors reduced the Root Mean Square Error (RMSE) by 12–15% and significantly improved the

detection accuracy of short-term workload spikes ("micro-bursts").

- **Synthetic Graphs:** A compelling methodological development involves forming synthetic graph structures based on abstract data rather than solely physical connections. In the EvoGWP architecture [20], the authors construct a graph from "shapelets" representing behavioral workload signatures. Edges in this graph reflect interference or transitions between different workload patterns, enabling the modeling of long-term pattern evolution—a critical factor for strategic cloud resource planning.

Despite their high accuracy, GNNs possess a significant drawback: substantial computational complexity, which can hinder their real-time application. The evolution of approaches in the works of Tam et al. [21, 22] clearly illustrates this challenge:

1. **Stage 1 (PERT-GNN):** In 2023, authors utilized GNNs alongside the PERT technique for latency prediction, yielding a significant advantage over classical models such as Random Forest [21].

2. **Stage 2 (FastPERT):** By 2025, the same researchers shifted away from classical GNNs in favor of the FastPERT architecture [22]. Instead of heavy graph convolution computations, they applied the concept of structural inductive bias.

This shift in focus signifies an important trend in the development of intelligent monitoring systems: a transition from universal but resource-intensive graph models toward specialized architectures where topological knowledge of microservices is embedded directly into the algorithm's structure. This allows for combining the global context of graph analysis with low-latency data processing requirements, which is critical for real-time cloud environment automation.

3.2.4 Comparative Analysis

The following table provides a comparison of the architectures discussed above, specifically focusing on parameters such as computational complexity, temporal trend modeling, and spatial topology awareness. The key advantages of each approach are highlighted.

Table 2. Comparative characteristics of machine learning methods for proactive orchestration

Architecture	Temporal Trend Modeling	Spatial Topology Awareness	Computational Complexity	Key Advantage
RNN (LSTM/GRU)	High	Low	Moderate	Efficiency in handling linear and cyclic workloads
Transformers (Attention)	Very High	Moderate	High	Processing of non-linear spikes and parallelization of computations
GNN (Graph Networks)	Moderate	Very High	Very High (Traditional)	Modeling of cascading effects and inter-microservice relationships
Hybrid (CNN-Transformer, GNN+TCN)	Very High	Very High	Variable (Depends on optimization)	Synergy of spatial and temporal analysis for maximum forecast accuracy

Each architecture offers specific advantages depending on the application domain. For instance, RNNs demonstrate high efficiency when dealing with linear or cyclic data, whereas GNNs are more suitable for modeling complex inter-service relationships. Hybrid architectures merit particular attention as they aim to integrate the strengths of diverse approaches. Such a combination potentially provides both versatility and higher forecasting accuracy; however, it presents a significant challenge: finding effective ways to optimize computational complexity to maintain the model's cost-effectiveness.

To provide an objective comparison of state-of-the-art (SOTA) models' performance, a comprehensive set of metrics is utilized. This allows for an evaluation of both the theoretical forecasting accuracy and the practical feasibility of deploying the model within real-world systems. Specifically, the analysis is based on statistical error indicators and quantitative characteristics of computational resource consumption, as detailed in Table 3.

Table 3. Summary of Model Performance Metrics

Category	Metric	Description and Purpose
Accuracy (Error Metrics)	MSE (Mean Squared Error)	Measures the average squared difference. It is highly sensitive to outliers, allowing the model to be heavily penalized for significant forecast errors.
	MAE (Mean Absolute Error)	Represents the average absolute difference between the forecast and actual data, expressed in the same units of measurement.
	RMSE (Root Mean Squared Error)	The square root of MSE. It shares the same dimensionality as the input data, which facilitates the interpretation of the error magnitude.
Resources (Efficiency)	CPU Savings (%)	The percentage reduction in processor time usage compared to a baseline solution under an identical workload.
	EA (Elastic Acceleration)	An indicator of energy/resource availability or efficiency. It defines the system's ability to perform computations and scale with minimal energy consumption.

Conclusions

The conducted analysis of modern approaches to reactive and proactive orchestration in cloud-native environments allows for the following conclusions regarding the current state of the field and its development prospects:

1. **Evolution of Forecasting Architectures:** A distinct transition is observed from homogeneous models toward hybrid neural network structures (CNN-LSTM, Bi-LSTM). Hybridization enables simultaneous accounting for linear workload trends and complex non-linear dependencies inherent in microservice traffic.

2. **Advantage of Graph-based Approaches (GNN):** An analysis of recent works (e.g., PERT-GNN and EvoGWP) confirms that for modern microservice systems, accounting for call topology is a key factor in accuracy. Traditional time-series methods often ignore cascading effects, whereas Graph Neural Networks allow for modeling inter-service relationships—a critical factor for meeting Service Level Objectives (SLOs) and optimizing resource utilization (e.g., reducing CPU usage by up to 36% [18]).

3. **Critical Bottlenecks:** Despite the high accuracy of modern Deep Learning models, the primary challenges remain:
   - *Computational Complexity:* The necessity of minimizing the overhead imposed by the forecasting algorithms themselves.
   - *Environmental Dynamism:* The problem of concept drift, where a trained model loses relevance due to changes in business logic or system topology.
   - *Energy Efficiency:* The need to integrate ecological metrics (carbon footprint) into scaling decision-making processes.

4. **Prospects for Further Research:** Future efforts should be directed toward developing adaptive graph models capable of online learning in real-time without full retraining, alongside other measures aimed at simplifying computations without compromising accuracy. To reduce overhead, a comprehensive approach is required—integrating knowledge of the target system directly into the model architecture [22], rather than merely increasing neural network capacity.

A particularly promising vector for proactive orchestration is its adaptation to edge and fog computing environments. Shifting computational logic to the edge opens

possibilities for the real-time coordination of autonomous agent swarms and unmanned aerial vehicles (UAVs). In such scenarios, network topology is highly dynamic, and node resources are constrained, rendering traditional reactive approaches ineffective. The application of GNNs and hybrid architectures in these resource-constrained environments will enable the modeling of volatile relationships between autonomous units, ensuring preemptive task distribution and connection stability. Further research should focus on developing lightweight forecasting models capable of functioning under high network volatility without losing orchestration precision.

Furthermore, there is a developing trend toward shifting the focus from the abstracted task of resource optimization to a higher-order objective: increasing energy efficiency and compliance with "green standards" within the context of large-scale data center operations.

References

1. Narváez, D., Battaglia, N., Fernández, A., & Rossi, G. (2025). Designing microservices using AI: A systematic literature review. *Software, 4*(1), 6. https://doi.org/10.3390/software4010006
2. Nguyen, T.-T., Yeom, Y.-J., Kim, T., et al. (2020). Horizontal pod autoscaling in Kubernetes for elastic container orchestration. *Sensors, 20*(16), 4621. https://doi.org/10.3390/s20164621
3. Burns, B., Beda, J., Hightower, K., & Evenson, L. (2022). *Kubernetes: Up and running: Dive into the future of infrastructure* (3rd ed.). O'Reilly Media.
4. Newman, S. (2021). *Building microservices: Designing fine-grained systems* (2nd ed.). O'Reilly Media.
5. J. Gao, H. Wang and H. Shen, "Machine Learning Based Workload Prediction in Cloud Computing," *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, Honolulu, HI, USA, 2020, pp. 1-9, doi: 10.1109/ICCCN49398.2020.9209730
6. Arslan S. 2022. A hybrid forecasting model using LSTM and Prophet for energy consumption with decomposition of time series data. *PeerJ Computer Science* 8: e1001 https://doi.org/10.7717/peerj-cs.1001
7. Zhang, H., Zhang, J., Zeng, J., & Zhu, Z. (2025). Container load prediction method based on a hybrid model of ARIMA and Bi-LSTM. *Proc. SPIE 13692, Fourth International Conference on Electronics Technology and Artificial Intelligence (ETAI 2025)*, 136923M. https://doi.org/10.1117/12.3068497
8. Wang, X., Wu, Y., Zou, W., & Zhao, X. (2025). Hybrid Time Series Forecasting for Real-Time Electricity Market Demand Using ARIMA-LSTM and Scalable Cloud-Native Architecture. *Informatica*, *49*(3). https://doi.org/10.31449/inf.v49i3.9474
9. Klymash, M., & Morhoiev, K. (2025). *Method for workload-based selection of lightweight prediction models in microservice autoscaling*. **ICTEE**, 5(2), 1–14. https://doi.org/10.23939/ictee2025.02.001
10. Ahamed, Z., et al. (2023). Technical study of deep learning in cloud computing for accurate workload prediction. *Electronics, 12*(3), 650. https://doi.org/10.3390/electronics12030650
11. Vasumathi, M. T., et al. (2025). AI-driven predictive auto-scaling for efficient microservices deployment in cloud data centers using LSTM and Kubernetes HPA. *2025 5th International Conference on Expert Clouds and Applications (ICOECA)*, 859–864. https://doi.org/10.1109/ICOECA66273.2025.00151
12. Patel, Y. S., & Bedi, J. (2023). MAG-D: A multivariate attention network based approach for cloud workload forecasting. *Future Generation Computer Systems, 142*, 376–392. https://doi.org/10.1016/j.future.2023.01.002
13. Sabyasachi, A. S., Sahoo, B. M., & Ranganath, A. (2024). Deep CNN and LSTM approaches for efficient workload prediction in cloud environment. *Procedia Computer Science, 235*, 2651–2661. https://doi.org/10.1016/j.procs.2024.04.250
14. Zhang, H., Li, J., & Yang, H. (2024). Cloud computing load prediction method based on CNN-BiLSTM model under low-carbon background. *Scientific Reports, 14*, 18004. https://doi.org/10.1038/s41598-024-68339-1
15. Mahajan, I., & Nadig, D. (2024). Enhancing workload predictions using service interactions in cloud-native microservices. *2024 IEEE 13th International Conference on Cloud Networking (CloudNet)*, 1–9. https://doi.org/10.1109/CloudNet62863.2024.10815917
16. Chauhan, R., Yang, J., & Khan, A. A. (2026). Reducing the carbon footprint of data centers: CNN-transformer-based workload prediction for green cloud computing. In *Computational Intelligence and Network Security. ICCINS 2025* (Vol. 2738, pp. 222–235). Springer. https://doi.org/10.1007/978-3-032-09572-5_22
17. Kumar, B., Verma, A., & Verma, P. (2025). A multivariate transformer-based monitor-analyze-plan-execute (MAPE) autoscaling framework for dynamic resource allocation in cloud environment. *Computing, 107*, 69. https://doi.org/10.1007/s00607-025-01426-x
18. Park, J., et al. (2024). Graph neural network-based SLO-aware proactive resource autoscaling framework for microservices. *IEEE/ACM Transactions on Networking, 32*(4), 3331–3346. https://doi.org/10.1109/TNET.2024.3393427
19. Luo, Y., et al. (2024). Integrating system state into spatio temporal graph neural network for microservice

workload prediction. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, 5521–5531. https://doi.org/10.1145/3637528.3671508

20. Li, J., et al. (2024). EvoGWP: Predicting long-term changes in cloud workloads using deep graph-evolution learning. *IEEE Transactions on Parallel and Distributed Systems*, *35*(3), 499–516. https://doi.org/10.1109/TPDS.2024.3357715

21. Tam, D. S. H., et al. (2023). PERT-GNN: Latency prediction for microservice-based cloud-native applications via graph neural networks. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*, 2155–2165. https://doi.org/10.1145/3580305.3599465

22. Tam, D. S. H., et al. (2025). FastPERT: Towards fast microservice application latency prediction via structural inductive bias over PERT networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *39*(19), 20787–20795. https://doi.org/10.1609/aaai.v39i19.34291

The article has been sent to the editors 27.04.26.
After processing 10.05.26.
Submitted for printing 30.06.26