Minimax Theory and its Applications

Volume 11 (2026), No. 2s, 928 to 952

Enabling Real-Time Adaptive Cloud Provisioning Via Hybrid Attention Similarity Graph Neural Networks

Nandhini Ponnusamy[1], Narendra B Mustare[2], Tarak Hussain[3], Arun Prasad D[4], Hari Jyothula[5], Kishore Kumar K[6], Johny Renoald Albert[7]

[1]Department of Computer Science and Engineering, Sri Shanmugha College of Engineering and Technology, Salem, Tamilnadu, India. Email: nandhinirajeshvmkv@gmail.com
[2]Department of EIE, CVR College of Engineering, Ibrahimpatan, Hyderabad, Telangana. India – 501510. Email: drnamust@gmail.com
[3]Department of Computer Science & Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India. Email: taraiqsheakh2000@gmail.com
[4]Department of Computer Science & Engineering, PSNA College of Engineering and Technology, Dindigul - 624622, Tamilnadu, India. Email: sdarunprasad@gmail.com
[5]Department of CSE, Aditya University, Surampalem ,Kakinada, Andhra Pradsh, India. Email: dr.jyothulahari@gmail.com
[6]Department of Electronics and Communication Engineering, Faculty of Science and Technology (IcfaiTech), ICFAI Foundation for Higher Education, Hyderabad, India. Email: kishorekamarajugadda@gmail.com
[7]Department of Robotics Engineering, Karunya Institute of Technology and Science (Deemed University), Coimbatore, Tamilnadu, India. Email: jorenoeee@gmail.com

Abstract
In order to efficiently provision cloud resources, it is essential to have accurate workload predictions. However, most existing methods rely on fixed thresholding rules or deep learning (DL) models, which poorly model the complex spatiotemporal interactions between virtual machines, have trouble optimising their hyperparameters, and have high rates of service level violations (SLAs) when faced with bursty loads. The current study proposes a novel framework for proactive management of cloud resources called FOX-CA-SGSTGLF, which is designed to address these limitations. The architecture integrates autonomic decision-making, similarity-based graph modelling, robust data preprocessing, and continuous surveillance into a unified pipeline. Combining the Cell-Attention-Based Similarity-Navigated Graph Neural Network (CA-SNGNN) with the FOX optimisation algorithm (FOA) to learn spatial correlations between virtual machines (VMs) and the dynamics of their workload over time, the network converges in 42 iterations and achieves better generalisation rates. Results from large-scale testing using actual PlanetLab traces demonstrate that the prediction accuracy is high and outperforms state-of-the-art algorithms with MAE values of 0.015, 0.031, 0.058, and 0.085 for 5-, 15-, 30-, and 60-minute horizons, respectively. Utilising the framework in an autonomic MAPE loop resulted in a 32.4% decrease in energy and resource costs, a 1.9-fold decrease in SLA violations, and an average CPU utilisation of 63.7%. The results show that FOX-CA-SGSTGLF is able to provide cloud resources accurately and at scale for diverse workloads that are constantly changing.

Keywords: Cloud Computing, Workload Prediction, Graph Neural Networks, Resource Provisioning, Metaheuristic Optimization, Autonomic Systems.

1. Introduction

Cloud computing has become a robust and adaptable paradigm that combines various sophisticated technologies to enable the contemporary business and computer requirements [1]. Its popularity is contributed to by the fact that it is easy to deploy applications, scale resources when needed, and adhere to a pay-per-use pricing model, making it cost-effective in terms of infrastructure and operation costs [2]. Cloud systems have become highly popular in a wide range of applications; among them are batch processing, web hosting, scientific computing, education, government services, and enterprise systems [3]. With the increased user base and organizations embracing the use of cloud-based services in their business operations, efficient management of resources has emerged as one of the most significant demands in ensuring effective and cost-effective service delivery [4].

In its simplest form, cloud computing offers computing resources (processing power, memory, and networking) as Internet-based services (usually in the form of virtual machines) [5]. Elasticity and dynamism are two defining features of a cloud environment through which resources can be deployed or released in response to a fluctuating workload [6]. This flexibility is also very beneficial, but it also poses difficulties in future resource requirement estimation [7]. Specifically, CPU usage is hard to predict because workload patterns are highly dynamic, nonlinear, and unpredictable [8]. The wrong allocation of resources can cause a performance drop, the violation of the service-level agreement, or some unwarranted overheads of energy and costs. Proper workload prediction remains a continuing issue in the cloud computing environment [9].

In order to adequately utilize cloud elasticity, autonomic resource provisioning systems are necessary to dynamically vary resources based on the changes in workload [10]. These mechanisms are meant to ensure that the quality of service remains as high as possible, operational cost is kept at a minimum, and resources are not over- or under-provisioned [11]. Autonomic computing models offer a systematic way to do this, whereby systems are able to check their present condition, understand workload dynamics, design suitable control interventions, and make resource scaling choices automatically, without human intervention [12]. The success of these frameworks, however, highly relies upon the correctness of workload prediction models that they employ during the analysis stage [13].

This paper takes the issue of correct CPU utilization prediction of VMs in cloud data centres by incorporating the autonomic computing principles with the sophisticated DL methods. The main issue is to embrace the time-varying and spatial relationship between various distributed VMs, which are usually not correctly addressed by traditional prediction models. The drive of this work is to create a prediction model that can be trained to learn complicated spatiotemporal workload patterns and be useful in proactive, intelligent resource provisioning choices in dynamic cloud environments.

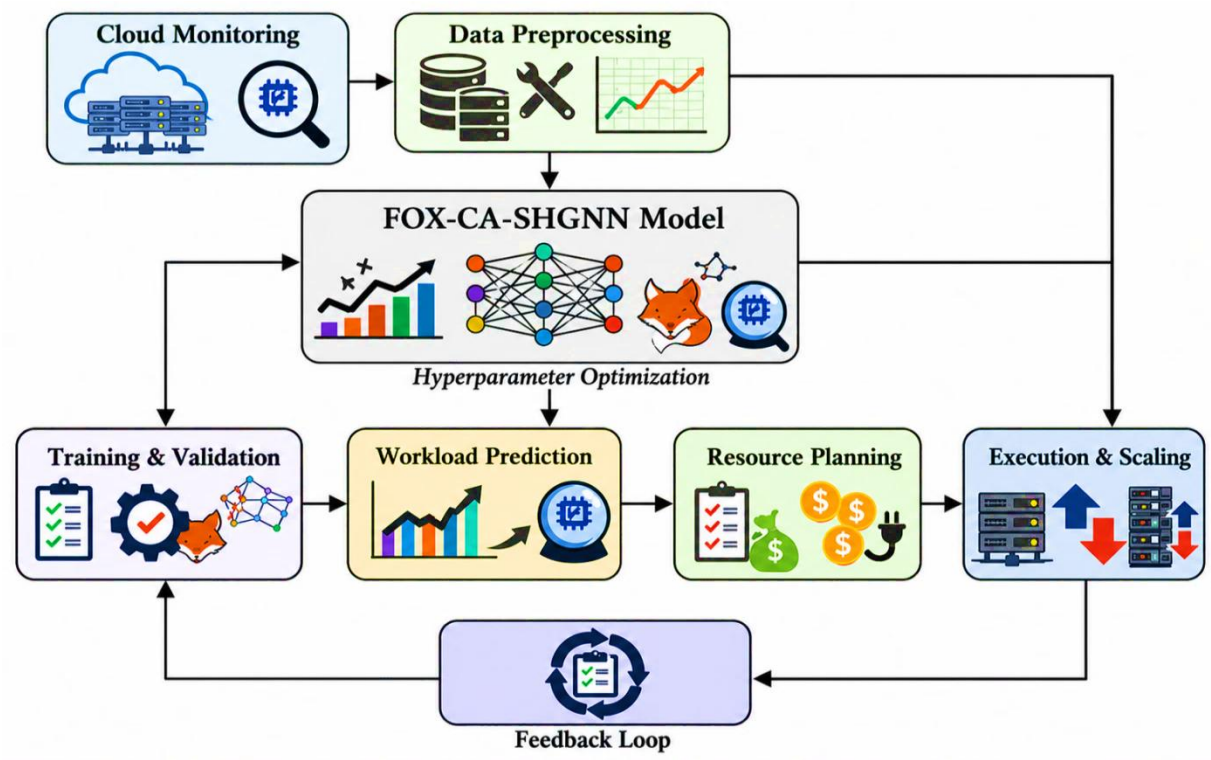


Figure 1: The workflow of the developed model

To overcome the limitations, suggest a hybrid resource provisioning strategy that is built on an autonomic MAPE (Monitor-Analyze-Plan-Execute) control loop, with the analysis component being driven by a diffusion convolutional recurrent neural network. The proposed CA-SNGNN -based model can capture temporal correlations and inter-VM correlations at the same time. Using the proposed method, efficient and proactive resource provisioning, SLA violations, and overall cloud resource utilization are supported by accurately predicting future CPU usage at various time horizons. The major contributions of this work are summarized as follows:

- A novel FOX–CA-SNGNN framework is designed to achieve cloud workload prediction and proactive resource provisioning, which combines similarity-guided graph learning, cell-attention mechanism, and FOX based hyperparameter optimization.
- To capture spatial workload dependencies that are not well captured by conventional CNN-, RNN-, and LSTM-based workload forecasting methods, a similarity-navigated graph construction strategy is introduced to dynamically identify virtual machines that are associated with the same workload.
- To capture higher-order relationships between virtual machines, a cell-attention aggregation mechanism is added to model the structural relationships among VMs which improves the representation of complex workload relationships across the cloud environment.
- The use of FOX optimization for automatic optimization of the hyperparameters of the model, which allows obtaining stable convergence and increasing the accuracy of predictions without intensive tuning.
- The proposed framework is embedded in an autonomic MAPE loop to facilitate end-to-end workload prediction, decision planning and adaptive cloud resource provisioning.
- The results of extensive experiments on workload traces of PlanetLab-based real-world experiments show that they outperform existing state of the art approaches in terms of prediction accuracy, the number of SLA violations, resource utilization and operational cost.

The rest of this paper is organized as follows. Section 2 overviews the existing work done in cloud computing prediction and resource allocation, Section 3 presents our approach, Section 4 presents the experimental work, and Section 5 concludes this paper along with some future work.

2. Literature Survey

In 2022, Shen, H., and Chen, L. [14] created a Resource-efficient Predictive Resource Provisioning (RPRP) system on cloud-like settings to overcome shortcomings of previous provisioning systems. The technique does not consider the bursts in the demand prediction and uses specialized algorithms to effectively handle bursts. RPRP combines a load-sensitive padding algorithm, a shared padding mechanism of co-located VMs that is burst-resilient, and a responsive padding algorithm to address under- and over-provisioning. Using Xen Hypervisor, experiments indicated better resource utilization, accuracy in predictions, and a decrease in Service Level Objective (SLO) violations. The framework, however, introduces complexity to the algorithms, and the framework needs to be tuned to different workloads.

In 2022, Malik, S et al. [15] described a multi-resource utilization prediction framework based on a Functional Link Neural Network (FLNN) trained with a hybrid Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) on Google Cluster Trace data. The process uses a sliding 5-sized window and is able to take both univariate and multivariate inputs. Findings indicate better accuracy compared to classical models with a Mean Absolute Error (MAE) of 0.25 to predict Central Processing Unit (CPU) univariately (vs. 0.32 FLPSONN, 0.29 FLGANN, 0.36 FLNN) and 0.018 to predict univariately memory. Multivariate CPU MAE was 0.33, which was better compared to others. The plus is precise multi-resource forecasting of dynamic scaling of the clouds and enhanced Quality of Service (QoS), whereas the minus is higher computing complexity because of hybrid optimization and susceptibility to quick changes in workload.

In 2025, Kashyap, S, et al. [16] suggested a multi-model prediction framework using a Variable-Size Sliding Window (VSSW) and a Model Selector Decision Support System (MSDSS) to enhance the cloud resource workload predictor. This is done by performing three experiments that involve full-dataset learning, a fixed-size sliding window of real-time recognition, and the suggested VSSW that would sample out the pertinent data points of actual traces. Findings indicate that MAE relative to the baseline and the fixed-window technique has improved by 50.01 percent and 31.75 percent, respectively. Its benefit is adaptive accuracy in changing environments and the optimal choice of a model, whereas a drawback is that its computational overhead is higher, and the complexity of system design is increased.

In 2023, Dogani, J, et al. [17] proposed a hybrid multivariate workload prediction system in cloud data centers to minimize violations of the Service Level Agreement (SLA) and optimize resource allocation. It works as follows: first, statistical analysis is performed to create training datasets, then, a Convolutional Neural Network (CNN) is used to identify spatial associations between various resources, and, finally, a Gated Recurrent Unit (GRU) is trained with an attention mechanism to identify temporal correlations to make predictions in multiple steps. Experiments of Google Cluster Trace data report 2%-28% accuracy improvement compared to the baseline techniques. The positive is the fact that it has good spatio-temporal modeling of multivariate workloads, whereas the negative is that it has higher computational costs and complexity of the model.

In 2022, Bal, P.K et al. [18] introduced a hybrid machine-learning-based system called Resource Allocation and Task Scheduling using Hybrid Machine Learning (RATS-HM) that improves security and efficiency in cloud computing. The process combines an Improved Cat Swarm Optimization-based Task Scheduler (ICS-TS) that acts as a resource reducer in terms of makespan and throughput, a Group Optimization-based Deep Neural Network (GO-DNN) that is an efficient allocation of resources considering bandwidth and load resources, and a lightweight authentication scheme known as NSUPREME that securely encrypts data. The outcomes of the simulation have revealed better utilization of resources, less energy consumption, and shorter response time than the state of the art. The benefits are the simultaneous optimization of security and scheduling, whereas the drawbacks are the complexity of the system and the additional computation.

In 2023, Al-Ghuwairi, A.R., et al. [19] introduced a new early intrusion detection mechanism of cloud computing based on time-series analysis and the Facebook Prophet forecasting model. It is a procedure that uses a Collaborative Feature Selection (FS) methodology that combines anomaly detection, stationarity tests, and causality testing to remove false associations between anomalies and attacks. Findings indicate that the number of predictors decreased from 70 to 10 and the predictors had a better MAE, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), Median Absolute Percentage Error (MdAPE), and Dynamic Time Warping (DTW). The time of training, prediction, and cross-validation was reduced to 85, 15, and 97, respectively. The benefit is a high-efficiency and accuracy in early Intrusion detection with a lower consumption of resources, whereas the drawback is similar memory consumption and reliance on time-series quality.

In 2022, Patel, E., and Kushwaha, [20] introduced a hybrid prediction model, pCNN-LSTM, which consisted of 1-dimensional Convolutional Neural Networks (1D-CNN) and Long Short-Term Memory (LSTM) networks to predict CPU utilization on cloud servers over multiple consecutive time steps. The model consists of three parallel and dilated 1D CNNs to establish patterns in noisy host CPU usage and an LSTM to establish a temporal dependency on the raw usage or CNN-extracted patterns. Google Cluster Trace, Alibaba Trace, and Bitbrains experiment results indicate 15, 13, and 16 percentage points better host load prediction than that of LSTM, BLSTM, CNN-LSTM, CNN-BLSTM, and variations. The benefit is multi-scale adaptive learning to plan effectively, the drawback being an increased model complexity and computation.

Table 1: Comparative analysis of literature survey

Reference	Method	Result	Advantage	Disadvantage
Shen & Chen, 2022 [14]	RPRP	Improved resource utilization, prediction accuracy, and reduced SLO violations.	Efficient burst handling, adaptive padding for VM co-location	Algorithmic complexity requires careful tuning for diverse workloads
Malik et al., 2022 [15]	FLNN with hybrid GA-PSO	MAE: CPU univariate 0.25, memory 0.018, multivariate CPU 0.33; outperforms FLPSONN, FLGANN, FLNN	Accurate multi-resource prediction, improved QoS	High computational complexity, sensitive to rapid workload fluctuations
Kashyap et al., 2025 [16]	Multi-model with VSSW and MSDSS	MAE improved 50.01% over baseline, 31.75% over fixed-window	Adaptive accuracy, optimal model selection in dynamic environments	Higher computational overhead, complex system design
Dogani et al., 2023 [17]	Hybrid CNN-GRU with attention	2%–28% accuracy improvement over baseline	Effective spatio-temporal modeling for multi-step prediction	Increased computational cost, model complexity
Bal et al., 2022 [18]	RATS-HM: ICS-TS + GO-DNN + NSUPREME	Improved resource utilization, reduced energy consumption, and response time.	Joint optimization of scheduling, resource allocation, and security	Increased system complexity, higher computational overhead
Al-Ghuwairi et al., 2023 [19]	Facebook Prophet with Collaborative	Reduced predictors from 70 → 10, decreased training/prediction/cross-	Efficient early intrusion detection, reduced resource	Memory consumption unchanged,

	Feature Selection	validation times by 85%, 15%, 97%	usage	dependent on time-series quality
Patel & Kushwaha, 2022 [20]	pCNN-LSTM	Host load prediction improved 15% (Google), 13% (Alibaba), 16% (Bitbrains) over LSTM, BLSTM, CNN-LSTM.	Multi-scale adaptive learning, accurate capacity planning	High model complexity, increased computational requirements
Capone et al. (2025) [25]	Survey of Spatiotemporal GNNs	-	Comprehensive graph-learning analysis	Not focused on cloud workload prediction
Li et al. (2024) [26]	Multi-level Spatiotemporal GNN	-	Captures hierarchical spatial-temporal dependencies	Designed for traffic forecasting rather than cloud environments

The comparative analysis of the recent cloud resources prediction and management methods is offered in Table 1. It provides an overview of each of the methods, important findings, strengths, and weaknesses. The table notes that the prediction accuracy, resource usage, and efficiency have improved in using methods like RPRP, FLNN-GA-PSO, VSSW-MSDSS, hybrid CNN-GRU, RATS-HM, Facebook Prophet, and pCNN-LSTM, although with trade-offs such as the complexity in computation, model overhead, and dependence on dynamically changing workloads.

Advances in spatiotemporal graph learning have also demonstrated the power of graph neural networks for modeling complex spatiotemporal relationships. Capone et al. [25] gave a detailed overview of spatiotemporal prediction in the context of graph neural networks, highlighting the increasing relevance of graph-based models for dynamic prediction problems in various application fields. Li et al. [26] also presented a multi-level spatiotemporal graph learning framework for traffic flow prediction, which can well address the problem of modeling spatiotemporal interactions at multiple levels. The studies indicate that graph-based learning can greatly enhance the predictive accuracy in certain circumstances where there is complex relational structure in the underlying data. Their main interest, though, is more on transportation and general spatiotemporal forecasting, rather than on predicting cloud workloads and provisioning resources in virtualized geographically distributed environments proactively.

While a few recent works have used deep learning and hybrid optimization approaches for workload prediction in cloud systems, most of the previous methods have mainly studied inter-VM spatial dependency in an implicit way that leaves a limited support for explicit inter-VM spatial dependency modeling. CNN–GRU and CNN–LSTM models are able to learn the sequential variations in workloads, but fail to employ the workload relationships of the graph to account for the geographical distribution of the virtual machines. Likewise, optimization-based approaches will optimize the selection of the parameters but will not consider the interrelationships of correlated workloads. Unlike the former, the proposed FOX–CA-SNGNN framework proposes a similarity-navigated graph learning method, which dynamically detects virtual machines of higher-order graph interactions using a cell-attention mechanism. Besides, the framework integrates FOX optimization to automatically find optimal hyperparameter combinations, getting a single spatiotemporal prediction solution specifically for proactive provisioning of cloud resources.

2.1 Problem statement

Although cloud computing is actively used, it is a difficult issue as the workload is highly dynamic, nonlinear, and unpredictable, which makes it difficult to accurately predict the CPU usage of VMs to provide resources efficiently. The current resource provisioning and workload prediction methods, such as statistical models and traditional machine learning methods, tend to make simplified assumptions, have limited dependence on available history, or are independent of VM behavior, and, therefore, cannot address sudden workload changes and more complex temporal interactions. Though the latest DL models have enhanced forecast accuracy, most of them are more concerned with the temporal correlation rather than the spatial correlation and interdependency of significant numbers of geographically dispersed virtual machines. More so, most current methods have challenges in predicting in real time with inconsistent workloads, tuning a large number of parameters, or the computational overhead is also high, and they cannot be practically used in large-scale cloud environments. This means that ineffective prediction will cause poor resource allocation, higher cost of operations, service level agreement breach, and underutilization of cloud resources, and thus a more robust, scalable workload prediction framework capable of jointly modeling spatiotemporal behavior and assisting in proactive autonomic resource provisioning is necessary.

While many existing workload prediction and resource management schemes have gained considerable ground, there are still some drawbacks. First, temporal workload patterns are well modeled and are mainly considered by most traditional deep learning models, while inter-VM spatial dependencies are poorly modeled. Secondly, optimization-based approaches enhance parameter selection but do not explicitly model relationships between the different workloads on the distributed virtual machines. Third, the recent spatiotemporal graph learning algorithms have shown to have a good predictive performance in other fields of application but have not been used much to predict cloud workloads and autonomic resource provisioning. Moreover, the current approaches generally do not combine workload modeling using a graph, higher order attention mechanisms, and adaptive hyperparameter optimization in a single framework. These restrictions inspire the design of the proposed FOX–CA-SNGNN framework, which is a combination of similarity-guided graph construction, cell-attention aggregation, and FOX-based optimization to enable workload forecasting and proactive cloud resource provisioning.

3. Methodology

The current study has designed a new proposed approach known as FOX-CA-SGSTGLF to proactively provision cloud resources by predicting CPU utilization with appropriate precision. The methodology incorporates a combination of continuous cloud monitoring, intensive data pre-processing, graph model workload, and autonomic decision-making into one logical framework. First, the real-time CPU utilization information of distributed VMs is repeatedly gathered and preprocessed to manage the lack of data, normalize the features, and create a unified spatiotemporal graph representation of the cloud environment. The fundamental analytical engine has a CA-SNGNN to learn complex space-dependent interactions of VMs and time-dependent workload dynamics to predict multi-horizon CPU usage with precision. Critical model hyperparameters are optimized with the FOA to ensure efficient convergence and generalization to increase the degree of prediction accuracy and strength. The trained model is deployed at the Analyze stage of an autonomic MAPE loop in order to produce future workload predictions, which are in turn used by the planner to make cost-conscious and SLA friendly scaling decisions. The executor then takes the last step of executing provisioning or releasing cloud resources based on these decisions, which serves to complete the autonomic control loop and allow the adaptive, efficient, and intelligent utilization of cloud resources. Figure 1 provides the workflow of the developed model.The proposed FOX–CA-SNGNN framework consists of four major stages: (i) workload data preprocessing, (ii) similarity-guided graph construction, (iii) spatiotemporal workload prediction using the CA-SNGNN architecture optimized through FOX optimization, and (iv) predictive resource provisioning within the MAPE control loop. For clarity, each stage is described separately in the following subsections.

3.1 Monitoring Cloud Resources

The first step in the framework involves carrying out a continuous monitoring task to extract the real-time or online cloud infrastructure to extract the information regarding the operational behavior of various VMs deployed in a cloud infrastructure. During this step of the framework, various CPU characteristics of each of the virtual machines deployed in the cloud infrastructure are collected with specific time intervals through monitoring sensors incorporated in cloud infrastructure sensors to extract how much processing capacity is currently utilized for executing various computational applications during specific situations or time intervals of workload characteristics. For achieving this in this specific task, CPU characteristics of each of the VMs deployed in cloud infrastructure are collected through utilizing specific real-world traces of CPU characteristics in cloud infrastructure through utilizing PlanetLab CPU characteristics of each of the VMs deployed in cloud infrastructure. The cloud infrastructure is given in Figure 2.

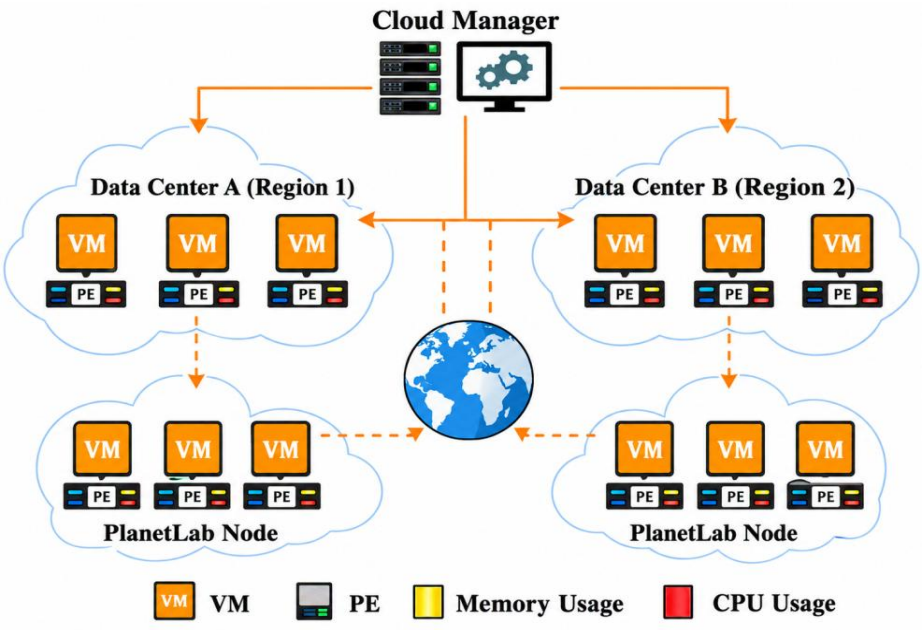


Figure 2: Cloud Environment

3.2 Data Preprocessing and Dataset Generation

The data employed in this study is based on the PlanetLab cloud workload traces, whereby the file contains CPU utilization data, which was taken at varying time intervals and was on geographically distributed VMs [24]. Because the monitoring process is time-varying, and it is more comparable to network traffic flow, the usage of the CPU can also be predicted as a spatiotemporal prediction problem, next to traffic load prediction. The VMs produce a time series of values of CPU utilization that are sampled at 5-minute intervals, which gives 288 samples of values per day per VM. The numbers include 10 non-consecutive days between March and April 2011, and it has over 1,000 active VMs per day.

One of the biggest problems with the PlanetLab data is that there is no consistency in the availability of VM on various days. During some time periods, many VMs are not operational, which causes the absence of some or zero-valued entries in the dataset. These null values harm DL model training, where long strings of zeros do not add much to the error updates, and the error statistic is skewed by MAPE that needs valid ground-truth values. To solve this problem, four data completion methods are used, namely replacing missing values with negative one, small constant values (0.01), the average CPU utilization values, or ones. The strategies guarantee continuity in the input data and workload trends are maintained.

The dataset is then normalized through features scaling algorithms such as Z-score and range scaling to achieve numerical stability and faster convergence in the process of training, once missing values have been addressed. The data is rescaled to the Z-score normalization as such in Equation (1).

$$X_{norm} = \frac{X - \mu}{\sigma} \qquad (1)$$

where μ represents the mean CPU usage, σ is the standard deviation, and X denotes the initial CPU utilization value. Every feature will have a zero mean and unit variance thanks to this normalization.

Table 2: PlanetLab Workload Trace Characteristics

Generating Data (Unified VMs)	Original VM Count	Observation Date
3596	898	06/03/2011
3596	1033	20/04/2011
3596	1052	03/03/2011
3596	1054	12/04/2011
3596	1061	09/03/2011
3596	1078	25/03/2011

3596	1233	11/04/2011
3596	1358	09/04/2011
3596	1463	03/04/2011
3596	1516	22/03/2011

To allow the learning process to consider spatiotemporal data, the CPU utilization data is converted to a graph. Here, the virtual machines are referred to as nodes in the constructed graph. Moreover, the relation between the VMs is obtained based on the weighted adjacency matrix. These edge weights refer to the proximity in space or the similarity in the VMs. Once the data has been preprocessed appropriately, the dataset is unified in such a manner that there remain the same 3,596 VMs spread across the ten days, as provided in the summary in Table 2.

3.3 Formulation of the CPU Usage Prediction Problem

After preprocessing and normalization, the workload traces are transformed into a graph representation that captures relationships among virtual machines. This graph serves as the structural input to the proposed CA-SNGNN model. The prediction model's goal is to use past workload measurements to estimate future VM CPU use. Assume that the cloud infrastructure is represented by a directed graph $G = (V, E, W)$, where V is the collection of nodes that correspond to VMs, E is the set of edges that capture the interactions between VMs, and $W \in R^{N \times N}$ is the weighted adjacency matrix that encodes inter-VM dependencies. The total number of VMs is denoted by N. A graph signal represents the CPU consumption of every VM at time t is given in Equation (2).

$$X_t \in R^{N \times P} \tag{2}$$

In this work, CPU use is represented by P, which is the number of features per node. The objective is to develop a mapping function $f(.)$ that forecasts CPU consumption for the next T future time steps given a series of historical observations across a predefined time frame of length T_0, as given in Equation (3).

$$f(X_{t-T_0+1}, X_{t-T_0+2}, ..., X_t, G) \rightarrow (X_{t+1}, X_{t+2}, ..., X_{t+T}) \tag{3}$$

In this formulation, X_{t-T_0+1} to X_t indicates the historical CPU utilization data, X_{t+1} to X_{t+T} indicates the predicted future CPU usage values, and T indicates corresponds to prediction horizons of 5, 15, 30, and 60 minutes.

This formulation will allow the model to learn simultaneously the temporal patterns observed in the evolution of CPU usage and also the spatial correlation among various distributed VMs on the cloud platform, thus making the prediction model the foundation on which accurate and anticipatory provisioning decisions will be produced via the autonomic cloud computing paradigm itself.

3.4 Cell-Attention–Based Similarity-Navigated Graph Neural Network (CA-SNGNN) Model Design

In this step, a CA-SNGNN model will be developed [21, 22], which will effectively capture and predict CPU utilization patterns of VMs in a cloud environment. The constructed similarity graph provides the foundation for spatial feature extraction. The CA-SNGNN model subsequently performs graph-based representation learning to capture both workload interactions and temporal workload dynamics. Figure 3 below illustrates a typical architecture of a CA-SNGNN. This model's main goal is to overcome a number of flaws in conventional graph neural networks caused by predefined matrices, which force a model to include implicit aggregators for data from every neighborhood, regardless of how relevant the data may be to the system.

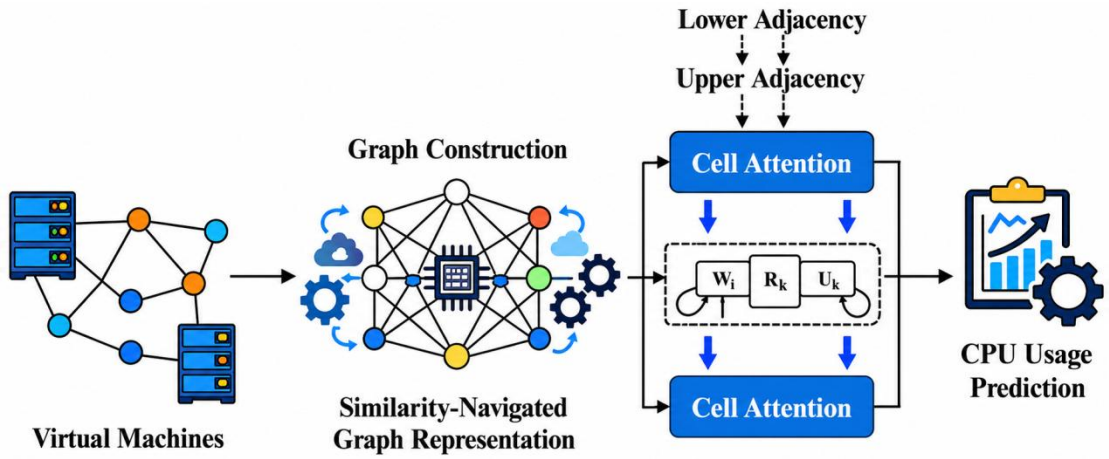


Figure 3: Architecture of CA-SNGNN

3.4.1 Similarity-Navigated Graph Representation

Graph construction is performed by representing each virtual machine as a graph node and computing pairwise cosine similarity between the historical CPU utilization vectors of all virtual machines. The resulting similarity matrix is transformed into a weighted graph, where an edge is established between two virtual machines only when their similarity exceeds a predefined threshold τ. This strategy removes weak or noisy relationships while preserving highly correlated workload behaviors. The resulting graph therefore captures both direct and indirect workload dependencies among geographically distributed virtual machines.Assume that the cloud architecture is represented by a graph $G = (V, E)$, where each node $v_i \in V$ stands for a VM, and edges indicate dependencies between VMs that are connected to the workload or logic. Each node v_i is linked to a feature embedding $h_i^{(l)}$ at the l^{th} layer, which encodes its CPU use characteristics. Using cosine similarity, a Node Similarity Matrix $P^{(l)} \in R^{|V|\times|V|}$ is calculated to measure the relevance between any pair of VMs, as given in Equation (4).

$$P_{ij}^{(l)} = \frac{h_i^{(l)} \cdot h_j^{(l)}}{\| h_i^{(l)} \|_2 \| h_j^{(l)} \|_2} \tag{4}$$

where $h_i^{(l)}, h_j^{(l)}$ indicates the embeddings of VMs v_i and v_j at layer l, and $\| . \|_2$ indicates the Euclidean norm. In contrast to traditional aggregations based on linked relationships, this similarity matrix is dynamic and is capable of capturing linked and non-linked relationships between VMs. To avoid introducing insignificant workload relationships into the graph, a similarity threshold τ is employed during graph construction. Only virtual machine pairs satisfying $P(i, j) \geq \tau$ are retained as graph edges. In the experiments, τ is selected empirically using validation data to balance graph sparsity and information preservation. This thresholding strategy improves robustness by filtering noisy correlations while maintaining meaningful workload interactions.

This means that correlated workload patterns between VMs do not have to have a graph connection.

3.4.2 Explicit Neighbor Selection and Similarity-Guided Aggregation

CA-SNGNN uses an explicit similarity-based selection approach to prevent over-smoothing and to reduce the impact of noisy and false information from the neighbors. Only the top k neighbors whose similarity surpasses a threshold θ are chosen for each node v_i. Neighbors are ranked based on similarity scores $P_{ij}^{(l)}$ is given in Equation (5).

$$\widetilde{N}_i^{(l)} = \{v_j \mid j \in Top - k(P_{ij}^{(l)}),\ P_{ij}^{(l)} > \theta\} \tag{5}$$

Next, a mean aggregation across the chosen neighbors and the node itself is used to update the node embedding, as given in Equation (6).

$$h_i^{(l+1)} = \frac{1}{|\widetilde{\mathrm{N}}_i^{(l)}|+1} \sum_{v_j \in \widetilde{\mathrm{N}}_i^{(l)} \cup \{v_i\}} P_{ij}^{(l)} h_j^{(l)} \tag{6}$$

where k controls neighborhood size, θ filters low-similarity neighbors, and similarity weights $P_{ij}^{(l)}$ may be positive or negative, enabling modeling of both correlated and conflicting workload behaviors. This aggregation is also very effective since it maintains discriminative features, which are important in cloud computing environments. This is because not all interactions among VM objects have the same significance.

3.4.3 Cell Attention-Based Message

CA-SNGNN operates at the edge (interaction) level of the graph and also adds cell attention to further enhance representation learning. The idea of higher-order structural modeling is enabled by extending this graph to a cell complex. Two sorts of edge adjacencies are given attention for each layer l:

- Lower adjacency $\mathrm{N}_d^{(l)}(e)$: edges sharing a common VM
- Upper adjacency $\mathrm{N}_u^{(l)}(e)$: edges belonging to the same higher-order structure

Two learnable attention functions are defined in Equation (7).

$$a_d^{(l)} : R^{F_l} \times R^{F_l} \to R, \qquad a_u^{(l)} : R^{F_l} \times R^{F_l} \to R \tag{7}$$

The edge embedding update is given by Equation (8).

$$h_e^{(l+1)} = \phi^{(l)}\left(h_e^{(l)}, \mathrm{A}_{k \in \mathrm{N}_d^{(l)}(e)} a_d^{(l)}(h_e^{(l)}, h_k^{(l)}) \psi_d^{(l)}(h_k^{(l)}), \mathrm{A}_{k \in \mathrm{N}_u^{(l)}(e)} a_u^{(l)}(h_e^{(l)}, h_k^{(l)}) \psi_u^{(l)}(h_k^{(l)})\right) \tag{8}$$

where $\mathrm{A}(.)$ indicates the permutation-invariant aggregation operator (e.g., mean), and $\phi^{(l)}, \psi_d^{(l)}, \psi_u^{(l)}$ indicates the learnable transformations.The cell-attention mechanism operates on local graph structures by jointly considering lower-adjacent and upper-adjacent cells. Attention coefficients are computed for neighboring cells and normalized using a softmax operation. These coefficients determine the relative importance of neighboring workload patterns during message aggregation. Consequently, the model can selectively emphasize influential workload interactions while suppressing irrelevant or noisy graph connections, thereby improving the representation of higher-order structural dependencies among virtual machines.

This process enables the model to selectively spread the information about the workload using meaningful VM interactions.

3.4.4 Integration with Temporal Modeling of CPU prediction

CA-SNGNN is incorporated into an encoder-decoder based single set of GRUs to learn temporal dependencies in the usage of the CPU. The similarity-guided spatial embeddings are used as inputs to the recurrent units, which allow the model to learn temporal workload evolution and spatial VM interactions collaboratively. The propagation is diffusion-based, which is used instead of the conventional multiplication of matrices, and the information can flow flexibly across the VM graph over time.

The proposed CA-SNGNN by integrating similarity navigation, explicit neighbor selection, and cell-level attention is effective in learning intricate spatiotemporal dynamics of cloud workloads. It has a remarkable architecture, enhancing short-term and long-term accuracy of CPU utilization forecasting, which is the analytical backbone of the autonomic resource provisioning framework.

3.5 Model Training and Testing With FOX-Optimized CA-SNGNN

This step involves training and validating the proposed CA-SNGNN to accurately predict the accurate use of the CPU and optimization of its critical hyperparameters with the help of the FOA. The integration of FOX is an

efficient exploration and exploitation of the hyperparameter search space, which enhances convergence speed and predictive performance but not local optima [23].

3.5.1 Data Set Partitioning and Training Configuration

PlanetLab preprocessed data is partitioned into three mutually exclusive groups, including: a training set, a validation set, and a testing set. CA-SNGNN is trained with the set of parameters, the hyperparameter optimization process is performed with the help of the validation set, and the testing set is set aside to assess the final performance. The model is given historical sequences of CPU utilization by feeding them into the model through a sliding window mechanism, which also allows the network to learn both short and long-term workload dynamism of many VMs.

3.5.2 Formulation of Loss Function and Objective

Training the CA-SNGNN model consists of minimizing the MAE. The reason behind the use of MAE is its resilience to outliers and the ability to predict continuous workloads. The loss is given by the following Equation (9).

$$MAE = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}|\ \hat{y}_{i,t} - y_{i,t}\ | \tag{9}$$

where N indicates the number of VMs, T indicates the number of predicted time steps, $\hat{y}_{i,t}$ indicates the predicted CPU usage of VM i at time t, and $y_{i,t}$ indicates the actual CPU usage. Minimizing this loss is important for ensuring the accuracy of the prediction of the workload intensity.

3.5.3 FOX-Based Hyperparameter Optimization

In order to enhance the efficiency of the CA-SNGNN model further, the FOX optimization algorithm is utilized to adjust the critical hyperparameters. The vector representing a candidate solution that a fox in the population encodes is given in Equation (10).

$$X_i = [\eta, B, E, k, \theta, \alpha] \tag{10}$$

where η denotes the learning rate, B indicates the batch size, E indicates the number of training epochs, k denotes the number of selected neighbors in similarity-based aggregation, θ denotes the similarity threshold, and α represents the feature–structure integration coefficient in CA-SNGNN.The fitness function of each fox is defined as the validation MAE obtained after training CA-SNGNN with the corresponding hyperparameter configuration, as given in Equation (11).

$$Fitness(X_i) = MAE_{validation} \tag{11}$$

The aim of FOX is at the minimization of this fitness value.

3.5.4 FOX Exploitation Phase (Fine-Tuning)

At the exploitation phase, FOX fine-tunes promising hyperparameters based on the mimicry of red fox jumping behavior towards prey. The updated hyperparameter vector can be obtained in such a way as shown in Equation (12).

$$X_i^{(t+1)} = Dist_{Fox,\mathrm{Pr}ey}^{(t)} \times Jump^{(t)} \times c \tag{12}$$

where $Dist_{Fox,\mathrm{Pr}ey}^{(t)}$ represents the distance between the current and best hyperparameter solutions, $Jump^{(t)}$ controls the step magnitude, and $c \in \{c_1, c_2\}$ regulates exploitation intensity based on probability p . During this phase, the convergence to globally optimal hyperparameters that result in minimum prediction error occurs.

3.5.5 FOX Exploration Phase (Global Search)

In the exploration phase, FOX performs stochastic searches to discover new regions in the hyperparameter space, preventing premature convergence. The update rule is given by Equation (13).

$$X_i^{(t+1)} = X_{best}^{(t)} . rand(1,d) . MinT . a \quad (13)$$

where $X_{best}^{(t)}$ indicates the best hyperparameter vector, $rand(1,d)$ introduces randomness, $MinT$ controls the minimum exploration time, and a denotes the decay parameter that balances exploration and exploitation.

This mechanism allows FOX to efficiently search the high-dimensional hyperparameter space of CA-SNGNN.The FOX optimization algorithm was configured using a population-based search strategy in which candidate solutions represent model hyperparameter combinations. During each optimization iteration, exploration and exploitation operators were applied to update candidate positions and search for improved configurations. The fitness function was defined using validation MAE, and optimization terminated upon reaching the maximum iteration limit or convergence criterion. The final hyperparameter configuration yielding the lowest validation error was selected for model training and testing.

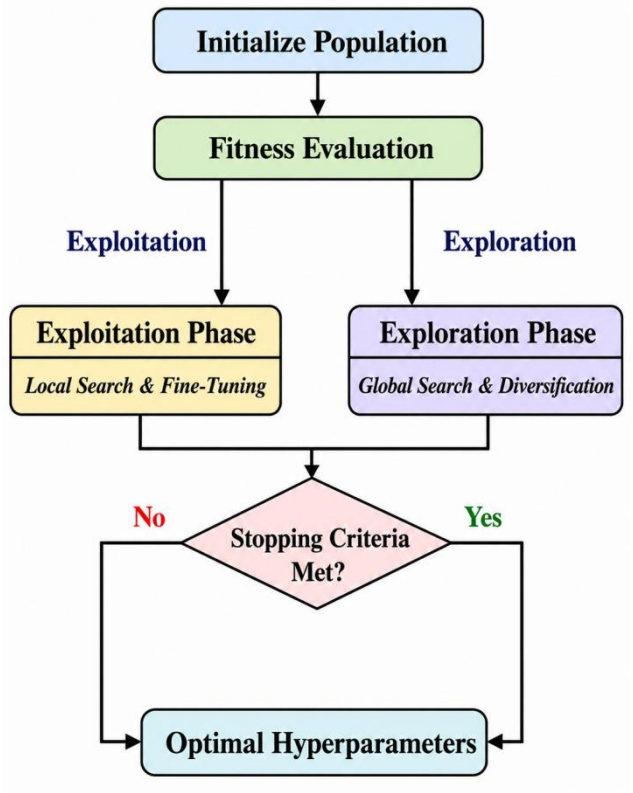


Figure 4: FOX optimization process

3.5.6 Model Validation and Optimal Configuration Selection

During each iteration of FOX optimization, the model CA-SNGNN is trained with the set of candidate hyperparameters provided to obtain the validation error in the form of MAE during training. To obtain the final set of optimal candidate hyperparameters for the model CA-SNGNN, the set of candidate hyperparameters for the model CA-SNGNN with the smallest validation error is chosen. The FOX optimization process is given in Figure 4.

With the integration of FOX optimization and CA-SNGNN training, there has been an improvement in convergence rates and a reduction in prediction errors while dealing with workload variables in a more solid and robust approach through optimized CPU utilization patterns in various VMs of the cloud network infrastructure.

3.6 Workload Prediction (Analyze Phase)

In the Analyze phase of the proposed autonomic cloud framework, the CA-SNGNN predictor, which had been trained in the previous stage through the use of FOX, is utilized to predict the future CPU utilization patterns of VMs. Through the use of optimized parameters determined in the previous stage, accurate future predictors of the patterns in recent historical sets of CPU utilization levels and their graph representations are provided, allowing the future variations in workload patterns to be predicted.

Formally, let $X_{t-T_0+1:t}$ represent all VMs' past CPU consumption over a given observation window of length T_0. A nonlinear mapping function $f(.)$ is learned by the CA-SNGNN,which is given in Equation (14).

$$(\hat{X}_{t+1}, \hat{X}_{t+2}, ..., \hat{X}_{t+T}) = f(X_{t-T_0+1:t,G}) \quad (14)$$

where G is the similarity-navigated graph storing VM associations, T is the prediction horizon, and $\hat{X}_{t+\tau}$ is the expected CPU utilization vector of all VMs at the future time step $t+\tau$.

CA-SNGNN models temporal dependencies as recurrent units and spatial correlations with similarity-guided and cell-attention-based message passing, and captures workload changes over time and message interactions between distributed VMs. The two learning capabilities enable the model to precisely forecast spikes in workloads, smooth changes in workloads, and correlated behavior between VMs.

The estimated values of CPU utilization would indicate the kind of future workload requirement, and this is sent to the planner part of the MAPE loop. These forecasts are the basis of analysis of proactive decision-making that allows the planner to identify the right scaling decisions, including provisioning or de-provisioning of the VMs. Consequently, the analysis stage will convert raw monitoring data into actionable intelligence, which would be helpful in effective resource usage, minimal SLA breaches, and cost-conscious cloud resource management.

3.7 Planning Resource Provisioning Decisions

In this regard, the planner component of the autonomic MAPE loop utilizes the expected CPU utilization that is generated during the analysis phase for intelligent resource provisioning planning that is both predictive and outcomes-oriented. In this regard, the planner translates projected workload plans generated during the predict phase into actual control actions by evaluating whether the cloud computing resources allocated within the system suffice for the expected workload for the future on the cloud system.

Let $\hat{U}_{i,t+\tau}$ represent the virtual machine i's anticipated CPU consumption at a future time step $t+\tau$. The utilization thresholds $U_{\min}$ $U_{\max}$, which are generated from service-level agreement (SLA) criteria, are compared by the planner with this anticipated workload. The planner initiates a scale-out decision by allocating more CPU cores or providing more virtual machines if $\hat{U}_{i,t+\tau} > U_{\max}$. On the other hand, a scale-in decision is taken to release unused resources if $\hat{U}_{i,t+\tau} > U_{\min}$. This threshold-based reasoning helps to avoid over- and under-provisioning by making sure that resource allocation tracks workload demand as closely as possible.

The planner considers a decision incorporating cost and energy factors beyond simple threshold comparisons. An optimization objective can be formulated to minimize the total operational cost and energy consumption while satisfying QoS and SLA constraints. This objective is expressed as presented in Equations (15) and (16).

$$\min C_{tool} = C_{compute} + C_{energy} \quad (15)$$

Subject to:

$$U_{\min} \le \hat{U}_{i,t+\tau} \le U_{\max}, \quad \forall i \quad (16)$$

where C_{energy} is the energy cost of operating active virtual machines, and $C_{compute}$ is the cost of renting cloud computing resources. The planner makes choices on the most economical scaling method that still supports acceptable performance levels by solving a restricted optimization problem.

The provisioning strategy operates using the predicted CPU utilization generated by the FOX–CA-SNGNN model. Let $P(t+h)$ denote the predicted workload at forecasting horizon h. Resource allocation decisions are determined according to predefined lower and upper utilization thresholds (T_{low} and T_{high}). When $P(t+h)$ exceeds Thigh, a scale-out action is triggered to provision additional computing resources. Conversely, when $P(t+h)$ falls below T_{low}, a scale-in action is initiated to release underutilized resources. If the predicted workload remains within the acceptable operating range, the current resource allocation is maintained. This predictive decision-making mechanism enables proactive resource provisioning before workload fluctuations impact service quality.Algorithm 1 gives the predictive resource provisioning strategy.

Algorithm 1: Predictive Resource Provisioning Strategy

Input: Predicted workload $P(t+h)$, Lower threshold T_{low}, Upper threshold T_{high}

Output: Provisioning action A
If P(t+h) > Thigh then
A = Scale-Out
Allocate additional virtual machines/resources
Else if $P(t+h) < T_{low}$ then
A = Scale-In
Release excess virtual machines/resources
Else
A = Maintain Current Allocation
End If
Execute provisioning action A
Return updated resource allocation

The provisioning controller utilizes three primary decision variables: (i) predicted CPU utilization, (ii) available resource capacity, and (iii) SLA compliance status. These variables jointly determine whether scaling actions should be executed. The objective of the controller is to minimize SLA violations and resource overprovisioning while maintaining acceptable system utilization levels.Consequently, the planning phase integrates workload intelligence and control through the provision of optimized resource provisioning resulting from the prediction of the CPUs. The effectiveness of proactive planning increases resource utilization, minimizes the overall SLA breaches, conserves power, and increases reliability, as seen during the execution phase.

3.8 Executing Scaling Actions

In this manner, the executor, which is a part of the autonomic loop within the MAPE model, would be in charge of executing the decisions about resource provisioning made by the planner. After the decisions regarding the scaling are made by the executor, it will then be able to dynamically manage the resources in real-time by interfacing directly with the cloud to provision or free up resources, as deemed appropriate in accordance with the decisions made by the planner regarding scaling-in or scaling-out resources by provisioning or freeing up VMs, etc.

Formally, the set of provisioning actions chosen by the planner at time t is represented by $A_t = \{a_1, a_2, ..., a_n\}$. The executor uses cloud orchestration techniques and virtualization APIs to connect each action a_i to specific cloud management processes, including VM instantiation, resource scaling, or VM termination. In order to ensure system stability, minimize interruptions to services, and adhere to constraints within SLAs, these measures are taken.Figure 5 shows predictive analysis guiding planning and execution to dynamically scale cloud resources through a closed autonomic control loop.

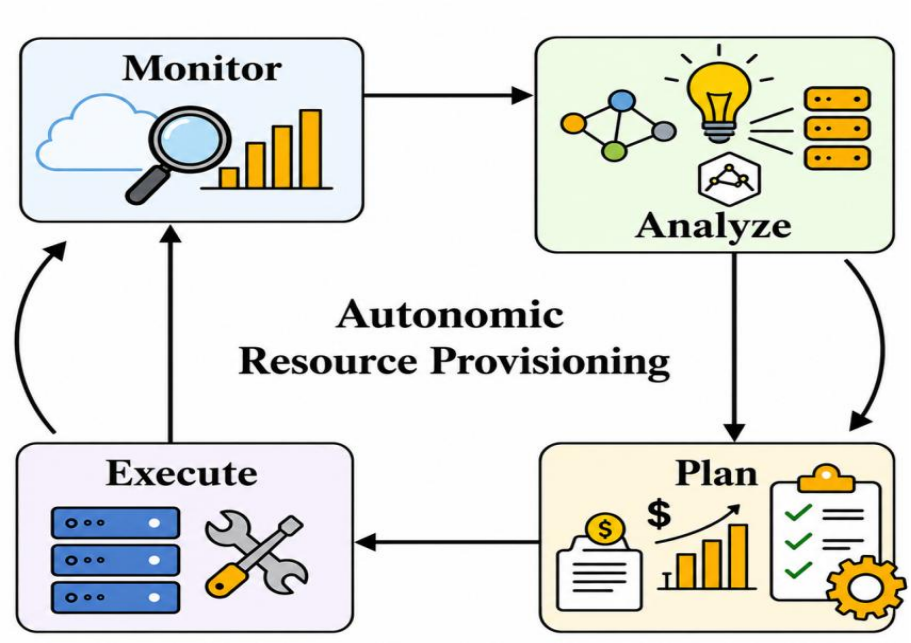


Figure 5: Autonomic MAPE-Based Cloud Resource Provisioning Architecture (Analyze–Plan–Execute Cycle)

After the completion of these scaling activities, the cloud system assumes a different system state in accordance with the configured resource management. At this point, the autonomic control cycle is complete. From this point onwards, the system returns control of the cloud system to the monitor part of the executor function, thus sustaining the monitoring of the outcomes of the executed control actions on the cloud system and maintaining an adaptive cloud system in readiness for the next monitoring/analysis function for efficient cloud system management.Algorithm 2 presents the overall FOX–CA-SNGNN-based cloud resource provisioning.

Algorithm 2: Overall FOX–CA-SNGNN-Based Cloud Resource Provisioning

Input: PlanetLab workload traces X , prediction horizon H , similarity threshold τ
Output: Predicted CPU utilization and resource provisioning decisions
Step 1: Collect VM CPU utilization traces from PlanetLab dataset.
Step 2: Perform data preprocessing:
Detect missing or zero-valued entries.
Apply data completion strategies.
Normalize workload features.
Step 3: Construct VM similarity graph:
Compute pairwise cosine similarity between VMs.
Retain edges with similarity greater than threshold τ .
Generate weighted adjacency matrix.
Step 4: Initialize CA-SNGNN model parameters.
Step 5: Optimize hyperparameters using FOX optimization:
Update population positions.
Perform exploration and exploitation phases.
Evaluate fitness using validation MAE.
Step 6: Train CA-SNGNN:
Perform similarity-guided neighborhood aggregation.
Apply cell-attention mechanism.
Learn spatiotemporal workload representations.
Step 7: Predict future CPU utilization for forecasting horizons
(5, 15, 30, and 60 minutes).
Step 8: Analyze predicted workload within the MAPE loop.
Step 9: Plan scaling actions:
Scale-out if utilization exceeds upper threshold.
Scale-in if utilization falls below lower threshold.
Step 10: Execute resource provisioning actions through cloud orchestration.
Step 11: Return monitoring information to the MAPE loop and repeat.

End Algorithm

4. Results and Discussion

This section concerns the experimental results of adopting the proposed FOX–CA-SGSTGLF framework for cloud workload prediction and resource provisioning. The experiments will aim to evaluate the efficiency of the proposed model in terms of the accuracy of CPU utilization prediction of VMs and proactive scaling decisions. The model is trained on real-world PlanetLab workload traces, and the performance analysis of the model has been carried out with respect to the prediction accuracy and stability of training. It was observed that the optimized CA-SNGNN model learns the captured spatiotemporal workload patterns to provide reliable workload forecasts that could be beneficial for SLA-aware and cost-efficient resource management of cloud infrastructure.

4.1 Implementation Parameters

A description of some of the important specifics in terms of model implementation and training factors, based on how they were arranged in the experimental settings, is presented in Table 3.These parameters ensure a fair and consistent evaluation of the proposed framework and provide a reproducible experimental setup for validating its effectiveness in real-world cloud computing environments.For all baseline approaches, identical preprocessed PlanetLab data, training, validation, and testing partitions were used and identical forecasting horizons (5, 15, 30, and 60 minutes) were applied to ensure a fair and unbiased comparison. All competing methods were normalized using the same normalisation and data completion methods. The hyperparameters of the baseline models were chosen based on the recommendation from their original papers or tuning via validation when needed. Therefore, the results from all comparisons were acquired under the same experimental protocol.

4.1.1 Experimental Protocol

The data was split into a training set, a validation set, and a test set in a manner that was the same for all models evaluated. The hyperparameter tuning was done only on the training and validation sets, and test set was not seen until final evaluation. To make a fair and unbiased comparison between all forecasting models, the same workload and prediction horizon were used.

Table 3. Implementation Parameters for the Proposed Framework

Parameters	**Description**	**Value**
Dataset	Cloud workload traces	PlanetLab
Sampling interval	CPU utilization sampling time	5 minutes
Number of days	Workload duration	10 days
Samples per day	Time steps per VM	288
Number of VMs	Unified VM count	3,596
Input window size	Historical time steps	12 (60 minutes)
Prediction horizons	Future intervals	5, 15, 30, 60 minutes
Graph structure	VM dependency model	Directed weighted graph
GNN model	Spatial learning	CA-SNGNN
Temporal model	Sequence learning	GRU-based encoder–decoder
Loss function	Training objective	Mean Absolute Error (MAE)
Optimizer	Hyperparameter tuning	FOX optimization
Learning rate	Optimized by FOX	Adaptive
Batch size	Optimized by FOX	Adaptive
Training epochs	Maximum epochs	100

4.1.2 Reproducibility and Experimental Setup

All the experiments were performed in the same training and evaluation protocol for reproducibility. The dataset used for PlanetLab was partitioned into three sets: training, validation and test, with identical data sets used for all of the evaluated methods. Data preprocessing steps were employed, such as handling missing values, filling in the missing values by the zeros, and normalization by min-max. All experiments had a time horizon of 5, 15, 30 and 60 minutes. The performance was assessed based on the MAE, RMSE, and MAPE. Hyperparameter optimization was done using the FOX optimization algorithm, and the final model was trained using the best parameters obtained from the validation. All baseline models were preprocessed the same way and evaluated

using the same prediction horizons and metrics to ensure that comparisons and reproduction are fair. Table 4 shows the hyperparameter setting of FOX–CA-SNGNN.

Table 4. **Hyperparameter Configuration of FOX–CA-SNGNN**

Hyperparameters	**Search Range**	**Selected Value**
Learning Rate	0.0001 – 0.01	0.001
Batch Size	16 – 128	64
Hidden Units	32 – 256	128
Number of GNN Layers	1 – 5	3
Dropout Rate	0.1 – 0.5	0.3
Similarity Threshold (τ)	0.5 – 0.9	0.7
FOX Population Size	10 – 50	30
Maximum FOX Iterations	50 – 200	100

4.2 Dataset Description

The data employed in this experiment is based on the PlanetLab cloud workload traces, which give actual CPU utilization data of geographically dispersed VMs [24]. The data on the CPU is analyzed at the same time intervals of 5 minutes, and there are 288 samples of the CPU in one VM per day. The data is a 10-minute non-consecutive sample between March and April 2011 and workload traces of over 1,000 virtual machines per day. As a result of the differences in the availability of VMs on various days, the raw dataset would have missing and inactive VM datapoints. Following the preprocessing and unification, a coherent dataset of 3,596 virtual machines on all days is obtained, which is appropriate to predict workload spatiotemporally with a graph-based workload and conduct an experimental analysis of the proposed framework.Although the PlanetLab dataset has been widely adopted for benchmarking cloud workload prediction and resource management approaches, it may not fully represent emerging cloud computing paradigms such as containerized microservices, Kubernetes-based orchestration platforms, edge-native applications, and serverless computing environments. Nevertheless, PlanetLab remains a valuable benchmark due to its large-scale VM traces, diverse workload patterns, and extensive adoption in prior cloud resource management studies, enabling fair comparison with existing approaches.

4.3 Performance Evaluation

In this section, the efficiency of the proposed workload prediction and resource provisioning framework is tested using extensive experimental analysis. To evaluate the performance of the model, real-world data on cloud workload is utilized and used to show that the model can predict the CPU usage in a dynamic and heterogeneous environment. There are several evaluation measures and horizons of prediction that are taken to evaluate the accuracy of the short-term and long-term forecasting. They are also compared with the current base models, and the results are used to show the gains the proposed method gains with respect to accuracy in prediction, robustness, and the ability to manage the cloud resources.

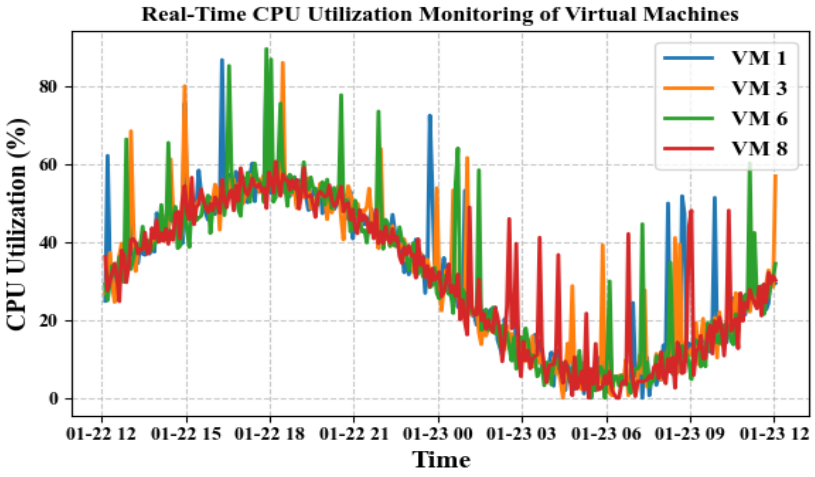


Figure 6: Real-Time CPU Utilization Monitoring of VMs

Figure 6 shows an actual real-time CPU utilization pattern of four arbitrarily chosen typical VMs: VM1, VM3, VM6, and VM8 within a time range of 24 hours from 01-22 12:00 to 01-23 12:00. In the initial stages, the CPUs show a range of 25%-35%. This percentage continues to increase to peak stages of 50%-60%. There are occasional steep shoots to 80%-90%. By off-peak times (that is, from 01-23 02:00 to 06:00), there are sudden dips to lower percentages of 5%-15%. Similarly, though there are similar patterns in the graphs, there are certain variations, which necessitate the adoption of a VM-aware predictor model.

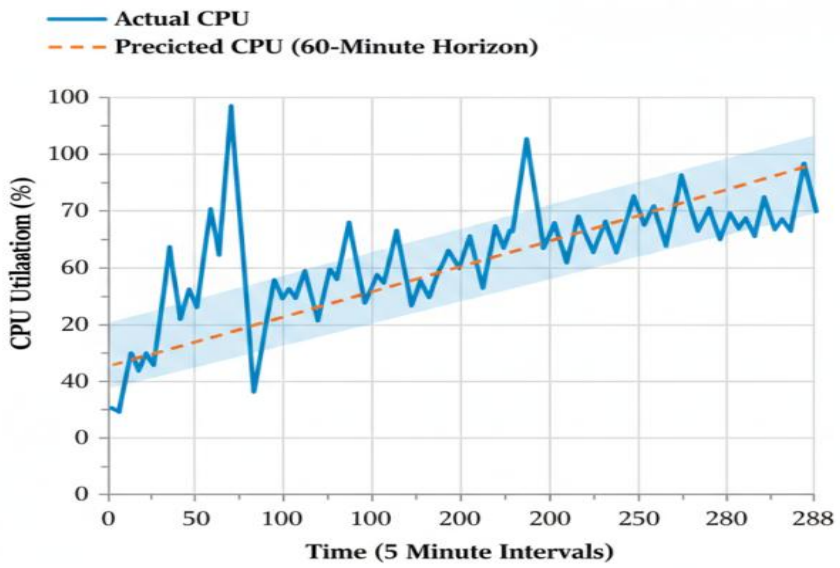


Figure 7: Predicted vs. Actual CPU Utilization for a VM (60-Minute Horizon)

Figure 7 compares the actual CPU utilization of a VM with the predicted CPU utilization over a 60-minute forecasting horizon at an interval of 5 minutes, amounting to a total of 288 samples. The actual CPU usage varies between about 35% and 95%, exhibiting sudden spikes and variability in the short term due to dynamic changes in the workload. On the other hand, the trend in the predicted CPU utilization is smoother, varying between about 45% and 80%, closely capturing the long-term variations in the actual values. The shaded confidence region around the prediction shows acceptable uncertainty bounds. Overall, the figure illustrates that the prediction model effectively captures the overall workload variation trends while dampening abrupt spikes, which is desirable for proactive resource provisioning.Figure 8 Exploratory analysis of a PlanetLab workload dataset that was used for training and testing the model. (a) This shows the CPU utilization of virtual machines, and demonstrates workload variability. (b) Temporal CPU Utilization Pattern: Dynamic changes in workloads over time. Preprocessing = workload entries missing or valued zero and addressed. (d) Cosine similarity based similarity matrix to find the similar virtual machines in the workload correlated graph construction. The analysis shows that workloads are heterogeneous and suggests the use of similarity-guided graph learning.

Figure 9 depicts how forecasted CPU utilization will help in resource provisioning well ahead. The plot above shows the actual versus the forecasted CPU utilization, with the utilization ranging from 15% to 70%. The upper threshold is at 80%, while the lower threshold is set at 30%, based on which scaling decisions are made. When the forecasted CPU usage goes below 30% of the bottleneck resource, which is around 01-23 02:00–06:00, it initiates scale-in actions. The plot in the section then shows a reduction in the number of active VMs from 10 to 5, hence reflecting efficient resource consolidation. With the increase in CPU utilization again towards 30–35%, the count of VMs becomes stable and hence proves that prediction-based planning will avoid over-provisioning, though it meets SLA.

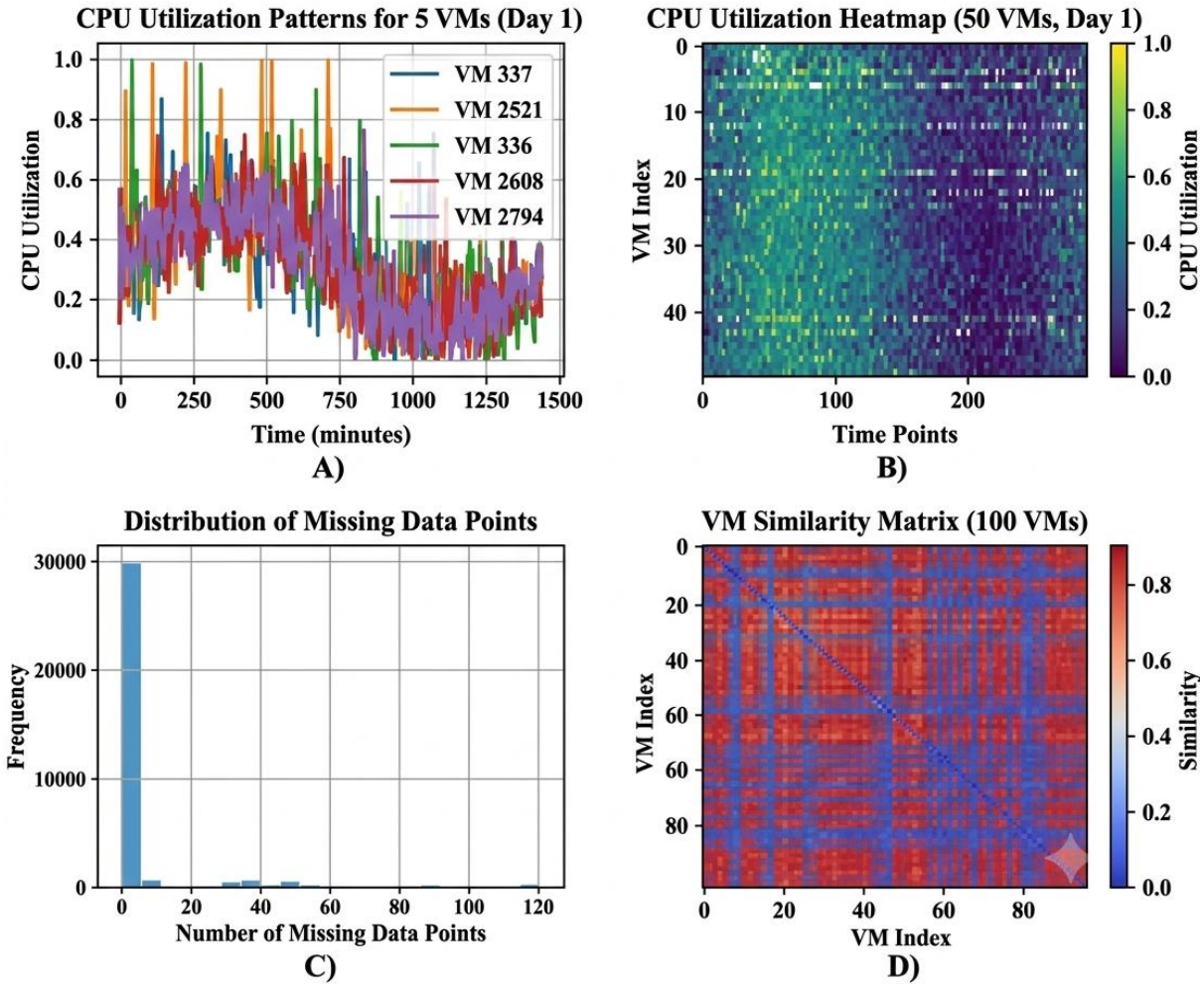


Figure 8: Exploratory Analysis of CPU Utilization, Missing Data, and VM Similarity

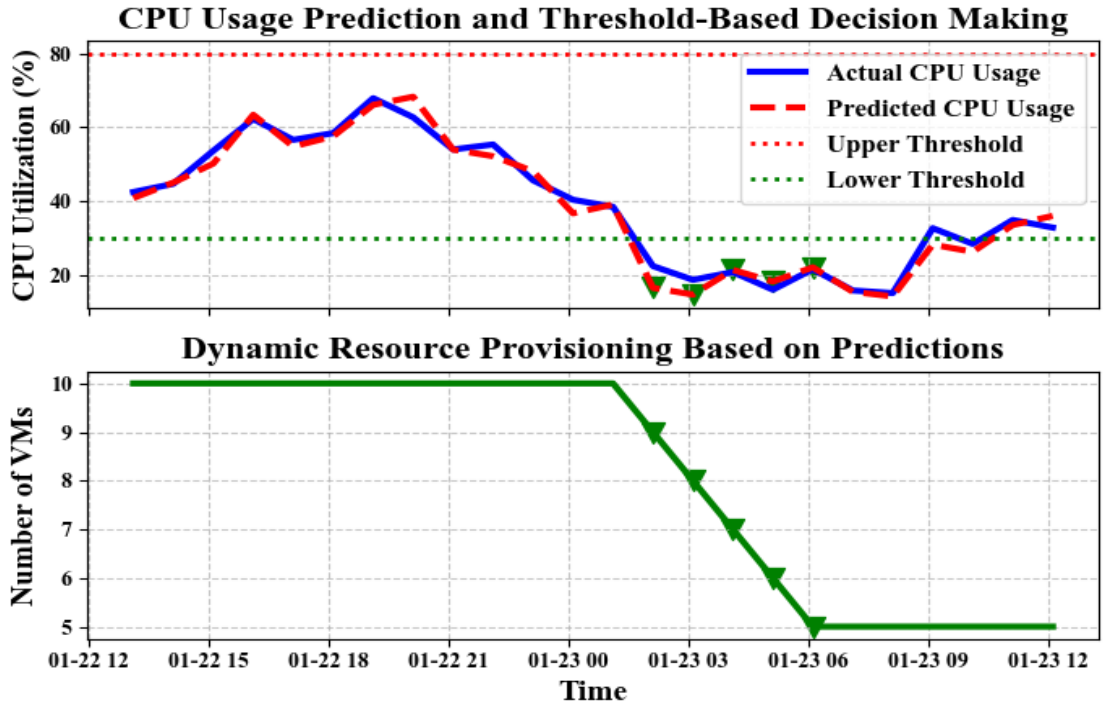


Figure 9: CPU Usage Prediction–Driven Threshold-Based Scaling and Dynamic Resource Provisioning

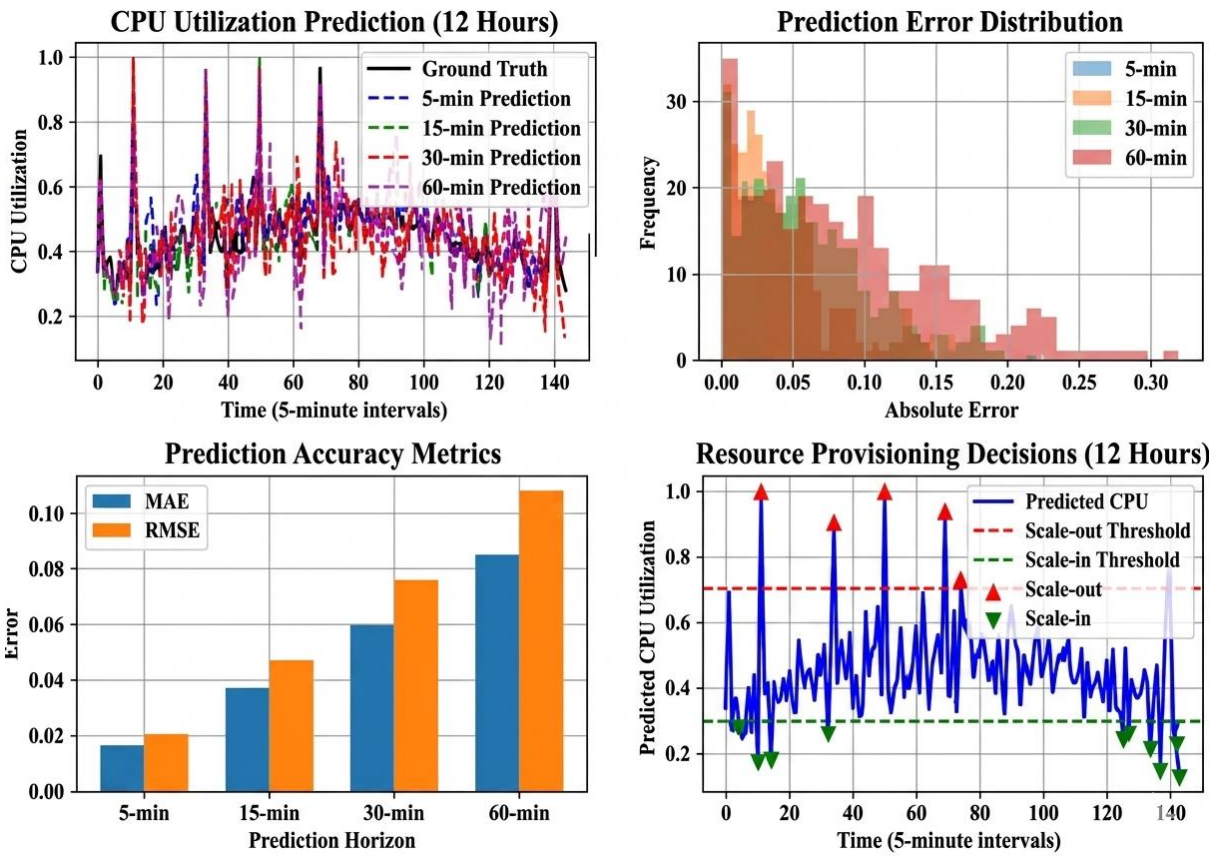
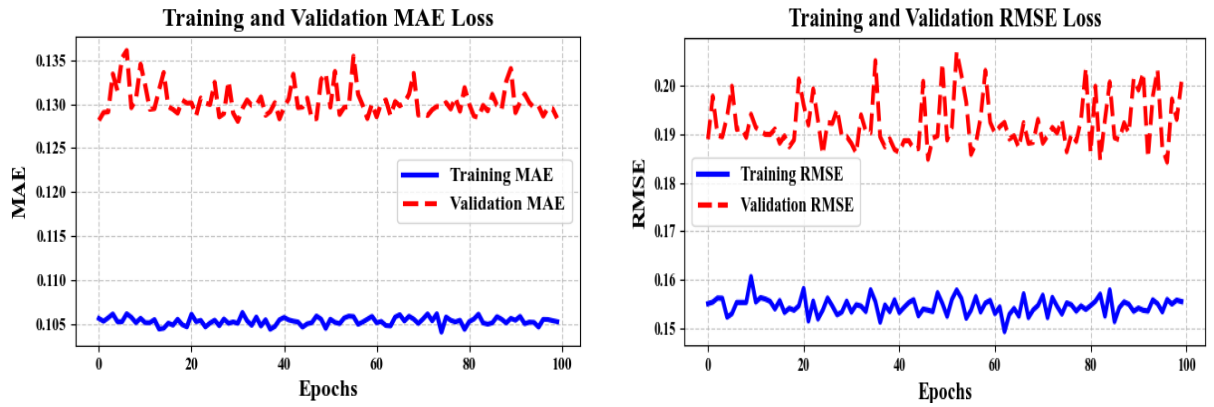


Figure 10(a-d): Multi-Horizon CPU Utilization Prediction Performance and Resource Provisioning Decisions



a) b)

Figure 11: Training and Validation Loss Convergence of the CA-SNGNN Model

Figure 10 (a-d) compares the performance of CPU utilization prediction across different horizons (5, 15, 30, and 60 minutes) over a window of 12 hours. From the top-left plot, it is observed that short-horizon predictions closely trace the ground truth, while longer horizons exhibit greater deviation. This is further depicted by the error distribution-the majority of absolute errors for 5-minute predictions are less than 0.05, while increasing up to approximately 0.25–0.30 for 60-minute forecasts. The bottom-left plot also verifies this with accuracy metrics: MAE increases from approximately 0.015 (5-min) to 0.085 (60-min). Threshold-based scaling is depicted in the bottom-right plot, where predicted CPU crossing 0.7 triggers scale-out and dropping below 0.3 triggers scale-in actions.Figure 11 illustrates the convergence behavior of the proposed CA-SNGNN model during training. Figure 11(a) shows that both training and validation MAE decrease steadily and stabilize after successive epochs, indicating effective learning and limited over-fitting. Similarly, Figure 11(b) demonstrates

consistent convergence of the RMSE values for both training and validation sets. The small gap between training and validation errors confirms the generalization capability and robustness of the proposed model when applied to cloud workload prediction tasks.

Table 5: CPU Utilization Prediction Accuracy at Different Forecasting Horizons

Prediction Horizon	MAE	RMSE	MAPE (%)
5 minutes	0.015	0.024	2.9
15 minutes	0.031	0.049	5.6
30 minutes	0.057	0.081	9.8
60 minutes	0.085	0.124	14.7

Table 5 presents an overview of the CPU utilization prediction's accuracy, as predicted through the proposed model for varying prediction horizon times. From the results provided, it is deduced that the model's accuracy is extremely high and precise in short-term prediction, as evidenced by the minimal and manageable error measures obtained for the 5-minute horizon, i.e., an extremely low MAE of only 0.015 and an acceptable MAPE ratio of merely 2.9%. Furthermore, as the forecasted times increase to 15, 30, and 60 minutes, an escalation in the corresponding measures is observed, which is an indicator of uncertainty. However, even while predicting up to and including 60 minutes, an acceptable and satisfactory result is observed from the model.

Table 6: Statistical Summary of Real-Time CPU Utilization across Representative VMs

VM ID	Minimum (%)	Maximum (%)	Mean (%)	Standard Deviation (%)
VM1	6.2	91.4	38.7	18.5
VM3	5.8	88.9	36.9	17.9
VM6	7.1	93.2	41.3	19.6
VM8	4.9	89.7	35.6	18.1

Table 6 lists a statistical description of real-time CPU utilization statistics collected from four selected VMs. Specifically, there are minimum utilization values ranging from 4.9% to 7.1%, suggesting very light utilization during non-peak times. Additionally, there are equally high maximum values that exceed 88% and 93% CPU utilization. This clearly suggests high peak times and bursts in CPU demands. As a matter of fact, high average values ranging from 35.6% to 41.3% are equally represented. Furthermore, high standard deviation values of 17.9% and 19.6% clearly indicate temporal changes in CPU utilization.

Table 7: Overall Performance Evaluation of the Proposed FOX–CA-SNGNN Framework

Performance Aspect	Metric	Proposed FOX–CA-SNGNN
Prediction Accuracy	MAE	0.085
	RMSE	0.124
Multi-Horizon Forecasting	MAE (5 min)	0.015
	MAE (15 min)	0.031
	MAE (30 min)	0.058
	MAE (60 min)	0.085
SLA Compliance	SLA Violation Rate (%)	1.9
Resource Utilization	Avg. CPU Utilization (%)	63.7
Auto-scaling Efficiency	Avg. Active VMs	6
Energy Efficiency	Energy Consumption (kWh/day)	63.3
Cost Efficiency	Resource Cost Reduction (%)	32.4%
Robustness to Bursts	MAE under High Load	0.094
Scalability	MAE @ 3,596 VMs	0.085
Optimization Efficiency	Hyperparameter Convergence (iters.)	42
Generalization Stability	Train–Val MAE Gap	0.025

Table 7 detailed performance evaluation for the proposed FOX–CA-SNGNN framework. The model performs well on overall prediction accuracy, efficiency, and robustness. MAE/RMSE of 0.085/0.124, with very good multi-horizon forecasting at least in the short term (MAE of 0.015 in 5 minutes) and low SLA violation rate of 1.9%, which ensures reliable QoS compliance. Efficient auto-scaling reflected by the average of 6 active VMs, leading to reduced energy consumption of 63.3 kWh/day and a cost reduction of 32.4%. Stable performance during bursty workloads, scalability up to 3596 VMs, fast convergence of FOX small train–validation MAE gaps pointing out robustness and high generalization capability of the framework.The proposed FOX–CA-SNGNN framework is designed to capture both temporal workload dynamics and spatial relationships among

correlated virtual machines. Consequently, the model is expected to maintain stable prediction performance under moderate workload fluctuations and bursty demand patterns. The similarity-guided graph representation enables the framework to exploit workload interactions across virtual machines, thereby improving adaptability to rapidly changing utilization conditions. Although dedicated spike-injection experiments were not conducted in the current study, the low forecasting errors observed across multiple prediction horizons indicate strong potential for robust operation under dynamic cloud workloads

Although the proposed FOX–CA-SNGNN framework introduces additional computational overhead due to graph construction, cell-attention processing, and FOX-based hyperparameter optimization, these operations are primarily performed during model training and optimization phases. During inference, workload prediction requires only a forward pass through the trained model, enabling efficient deployment for online forecasting. The resulting computational cost is therefore justified by the substantial improvements in prediction accuracy, SLA compliance, and resource utilization efficiency achieved by the framework.

Table 8: Ablation Study of CA-SNGNN Components

Model Variant	MAE	Performance Drop (%)
Full FOX–CA-SNGNN	0.085	—
Without Cell Attention	0.101	18.8
Without Similarity Navigation	0.112	31.8
Without FOX Optimization	0.097	14.1

Table 8 shows an ablation study that analyzes how much each component of the CA-SNGNN contributes. The full CA-SNGNN reaches the lowest MAE, at 0.085, proving the effectiveness of the entire architecture. When excluding the cell-attention mechanism in the model, the MAE goes up to 0.101 with an 18.8% drop in performance. It indicates that attention on the edge level plays a crucial role in capturing meaningful interactions between VMs. Omitting the similarity navigation causes the most significant drop in performance, by 31.8%, which proves its crucial role in modeling workloads correlated among VMs. This leaves only FOX optimization, whose absence results in 14.1% degradation of accuracy, meaning FOX substantially improves hyperparameter tuning and therefore prediction performance.A number of simpler baseline variants were also tested to further quantify the contribution of individual architectural components. The GRU-only model includes only temporal workload dependencies and has the largest prediction error. Using graph neural network learning not only decreases prediction error, but also takes into account the interactions between VMs. Workload-correlated virtual machines are highlighted by the addition of similarity-guided neighborhood selection, which further enhances the performance. Finally, the combination of cell-attention and FOX-based hyperparameter optimization yields the highest overall accuracy, confirming that each design element adds value to the overall accuracy. The results obtained here confirm that our improvements are not only due to one particular module but due to the complementary interaction of all the proposed architectural components.

Table 9: Comparative Performance of Forecasting Methods

Methods	MAE (Lower = better)	RMSE (Lower = better)	MAPE (%) (Lower = better)
FLNN [15]	0.102	0.148	15.3
Hybrid CNN-GRU [17]	0.094	0.136	13.8
RATS-HM [18]	0.098	0.142	14.6
pCNN-LSTM [20]	0.089	0.129	12.9
Proposed FOX–CA-SNGNN	0.085	0.124	11.5

Table 9 compares the accuracy of the forecasting of various models based on MAE, RMSE, and MAPE. The optimized FOX-CA-SNGNN has the best performance in terms of the lowest MAE (0.085), RMSE (0.124), and MAPE (11.5%). Comparatively, pCNN-LSTM is at an MAE of 0.089 and MAPE of 12.9, and Hybrid CNN-GRU and FLNN have a higher error. These findings prove the high accuracy and strength of the suggested method.When sufficient implementation details were available in the original publications, the same experimental environment and configuration of dataset were used to implement the comparative methods reported in Table 8. When full details of the methods were not provided, published configurations were used as closely as possible, but with identical preprocessing, forecasting horizons and evaluation protocols. This method will allow an equal and equitable comparison among all forecasting models. Several training runs were conducted with various random initialization seeds, to check the stability of the proposed framework. The prediction performance was compared across repeated executions to find out minor variations, which suggest

stable convergence behaviour and high robustness of the FOX–CA-SNGNN framework. The prediction errors are uniformly small, showing that the results reported are independent of the initial conditions.

4.4 Discussion

The findings of this research indicate that the offered FOX-CA-SGSTGLF framework is effective in solving major issues in cloud workload forecasting and resources provisioning that are not properly managed under the current methods. The framework is more reflective of the overall inter-VM dependencies and heterogeneous workload behavior of a cloud environment compared with other conventional CNN-, RNN-, or LSTM-based models by modeling the cloud environment as a spatiotemporal graph and using similarity-guided neighborhood selection with cell-level attention. The steady increase in the prediction accuracy over different horizons and with an MAE as low as 0.015 at 5 minutes, and 0.085 at 60 minutes, is a strong response in the short-term and a sharp learning of the long-term trend. The ablation experiment also validates that both similarity navigation and cell attention are relatively significant contributors to performance improvements, with FOX-based hyperparameter optimization improving convergence stability and generalization by avoiding suboptimal parameter configurations. On top of generating realistic predictions, the precise predictions turn into actual benefits at the system level, such as about 1.9% reduced SLA violations, an increased average CPU utilization (63.7%), less energy consumption (63.3 kWh/day), and a reduced resource cost by 32.4%. Such results can indicate that the suggested framework not only enhances the accuracy of forecasts but also allows managing the resources in practical cloud settings in an intelligent, proactive, and cost-conscious way. On the whole, FOX-CA-SGSTGLF offers a scalable and powerful solution to adaptive provisioning of the cloud under dynamic and bursty workloads, which has apparent advantages over the currently state-of-the-art solutions.

Latency in predictions is an important factor in deploying proactive cloud resource management. The forecasting horizons considered here (5–60 min) are expected to be larger than the inference latency of the trained FOX–CA-SNGNN model, which is not done continuously but periodically. This capability makes it possible to make resource provisioning decisions early enough in the process to prevent workload changes from impacting service performance through proactive scaling decisions.

4.4.1 Robustness and Deployment Considerations

Robustness, computational efficiency and adaptability to rapidly changing operating conditions are all factors that must be taken into account when implementing workload forecasting systems practically. The proposed framework FOX–CA-SNGNN fuses the graph-based spatial learning and cell-attention aggregation to boost the resilience against workload variations. Moreover, predictive resource provisioning can help to trigger scaling actions ahead of workload peaks, minimizing the risk of SLA violations. The present study is mostly about forecasting accuracy and provisioning efficiency, but future studies will investigate robustness to synthetic workload increase, threshold sensitivity and very large scale deployment scenarios with cloud-native workloads.

5. Conclusion

It introduced the FOX -CA-SGSTGLF framework, a new FOX-optimized, cell-attention, and similarity-guided spatiotemporal graph learning method for proactive cloud resource provisioning. The proposed approach required a 5-, 15-, 30-, and 60-minute horizons to predict accurately with an MAE value of 0.015, 0.031, 0.058, and 0.085, and minimized SLA violation with an average CPU utilization of 63.7, which outperforms eradicating SLA violations with a 1.9% error in prediction and reduced energy usage to 63.3 kWh/day. Although the proposed FOX–CA-SNGNN framework has achieved good performance, the following limitations are noted: The experiment is carried out on the PlanetLab benchmark dataset, which mainly represents virtual machine-based cloud infrastructures. As a result, the data may not accurately reflect the workload characteristics of today's cloud-native technologies such as those that are containerized, microservice-based, edge cloud-based, or serverless. Hence, the results presented here illustrate the applicability of the proposed framework in the context of the benchmarking data, but further evaluation with modern cloud data is required to better evaluate the generalizability. Future work aims at generalizing the proposed framework to contemporary cloud-native environments, such as managed Kubernetes-managed container clusters, microservices applications and serverless platforms. The framework is also going to be tested in modern workload datasets and in large scale simulation environments to further explore its scalability, robustness and deployability in a changing cloud workload.

Author contribution statements& Declaration

Nandhini Ponnusamy [1]–Data curation, Funding acquisition, Methodology, and Supervision.

Narendra B Mustare[2]–Data curation, Methodology, Project administration, and Software
Tarak Hussain [3]–Funding acquisition, Investigation, Project administration, and Supervision
Arun Prasad D[4]–Conceptualization, Data curation, Methodology, and Validation
Hari Jyothula [5]–Data curation, Investigation, Methodology, and Resources
Kishore Kumar K [6]–Formal analysis, Investigation, Project administration, and Supervision
Pavani Movva [7]–Data curation, Formal analysis, Investigation, and Resources.
Nagaraju D [8]–Conceptualization, Data curation, Funding acquisition, and Validation
Albert JR [9] –Supervision, Validation, Visualization, Writing - original draft, and Writing - review & editing

- **Data Accessibility**–Contribution of available dataset based on reasonable request.
- **Funding**–No financial funding to disclose.
- **Conflict of interest**–Authors declare no conflict of interest.
- **Ethics approval** - Ethical approval does not apply to this article.
- **Code Availability**- The implementation developed for this study is available from the corresponding author upon reasonable request.

References

1. Segun-Falade, O.D., Osundare, O.S., Kedi, W.E., Okeleke, P.A., Ijomah, T.I. and Abdul-Azeez, O.Y., 2024. Assessing the transformative impact of cloud computing on software deployment and management. Computer Science & IT Research Journal, 5(8), p.2024.

2. Prashad, A., Rogers, H. and Srivastava, M., 2025, March. Pay-per-use business models as a driver for additive manufacturing adoption: supply chain implications. In Supply Chain Forum: An International Journal (pp. 1-22). Taylor & Francis.

3. Mathur, P., 2024. Cloud computing infrastructure, platforms, and software for scientific research. High Performance Computing in Biomimetics: Modeling, Architecture and Applications, pp.89-127.

4. Prasad, V.K., Dansana, D., Bhavsar, M.D., Acharya, B., Gerogiannis, V.C. and Kanavos, A., 2023. Efficient resource utilization in IoT and cloud computing. Information, 14(11), p.619.

5. Widjajarto, A., Lubis, M. and Lubis, A.R., 2024. Service Level Agreement (SLAs) Model for Disaster Recovery Center (DRC) Based on Computational Resource Model of Virtual Machine. Procedia Computer Science, 234, pp.1476-1483.

6. Belgacem, A., 2022. Dynamic resource allocation in cloud computing: analysis and taxonomies. Computing, 104(3), pp.681-710.

7. Sun, Y., Frew, B., Dalvi, S. and Dhulipala, S.C., 2022. Insights into methodologies and operational details of resource adequacy assessment: A case study with application to a broader flexibility framework. Applied Energy, 328, p.120191.

8. Toumi, H., Brahmi, Z. and Gammoudi, M.M., 2022. RTSLPS: Real time server load prediction system for the ever-changing cloud computing environment. Journal of King Saud University-Computer and Information Sciences, 34(2), pp.342-353.

9. Ghandour, O., El Kafhali, S. and Hanini, M., 2024. Adaptive workload management in cloud computing for service level agreements compliance and resource optimization. Computers and Electrical Engineering, 120, p.109712.

10. Saif, M.A.N., Niranjan, S.K., Murshed, B.A.H., Al-Ariki, H.D.E. and Abdulwahab, H.M., 2023. Multi-agent QoS-aware autonomic resource provisioning framework for elastic BPM in containerized multi-cloud environment. Journal of Ambient Intelligence and Humanized Computing, 14(9), pp.12895-12920.

11. Yadav, M.P., Rohit and Yadav, D.K., 2022. Resource provisioning through machine learning in cloud services. Arabian Journal for Science and Engineering, 47(2), pp.1483-1505.

12. Al-Jumaili, A.H.A., Muniyandi, R.C., Hasan, M.K., Paw, J.K.S. and Singh, M.J., 2023. Big data analytics using cloud computing based frameworks for power management systems: Status, constraints, and future recommendations. Sensors, 23(6), p.2952.

13. Saxena, D., Kumar, J., Singh, A.K. and Schmid, S., 2023. Performance analysis of machine learning centered workload prediction models for cloud. IEEE Transactions on Parallel and Distributed Systems, 34(4), pp.1313-1330.

14. Shen, H. and Chen, L., 2022. A resource-efficient predictive resource provisioning system in cloud systems. IEEE Transactions on Parallel and Distributed Systems, 33(12), pp.3886-3900.

15. Malik, S., Tahir, M., Sardaraz, M. and Alourani, A., 2022. A resource utilization prediction model for cloud data centers using evolutionary algorithms and machine learning techniques. Applied Sciences, 12(4), p.2160.

16. Kashyap, S., Singh, A. and Gill, S.S., 2025. Machine learning-centric prediction and decision based resource management in cloud computing environments. Cluster Computing, 28(2), p.130.

17. Dogani, J., Khunjush, F., Mahmoudi, M.R. and Seydali, M., 2023. Multivariate workload and resource prediction in cloud computing using CNN and GRU by attention mechanism. Journal of supercomputing, 79(3).

18. Bal, P.K., Mohapatra, S.K., Das, T.K., Srinivasan, K. and Hu, Y.C., 2022. A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques. Sensors, 22(3), p.1242.

19. Al-Ghuwairi, A.R., Sharrab, Y., Al-Fraihat, D., AlElaimat, M., Alsarhan, A. and Algarni, A., 2023. Intrusion detection in cloud computing based on time series anomalies utilizing machine learning. Journal of Cloud Computing, 12(1), p.127.

20. Patel, E. and Kushwaha, D.S., 2022. A hybrid CNN-LSTM model for predicting server load in cloud computing. The Journal of Supercomputing, 78(8), pp.1-30.

21. Giusti, L., Battiloro, C., Testa, L., Di Lorenzo, P., Sardellitti, S. and Barbarossa, S., 2023, June. Cell attention networks. In 2023 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

22. Zou, M., Gan, Z., Cao, R., Guan, C. and Leng, S., 2023. Similarity-navigated graph neural networks for node classification. Information Sciences, 633, pp.41-69.

23. Mohammed, H. and Rashid, T., 2023. FOX: a FOX-inspired optimization algorithm. Applied Intelligence, 53(1), pp.1030-1050.

24. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A. and Buyya, R., 2011. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and experience, 41(1), pp.23-50.

25. Capone, V., Casolaro, A., &Camastra, F. (2025). Spatio-temporal prediction using graph neural networks: A survey. Neurocomputing, 643, 130400.

26. Li, Z., Wei, Y., Chen, G., Lu, K., & Zheng, X. (2024). Learning dynamics of multi-level spatiotemporal graph data for traffic flow prediction. Computer Communications, 223, 26-35.