Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 8, August : 2024

Design of an Improved Model for Cloud Workload Prediction Using LSTM, CNN, and Transformer Networks

Mohana Rao Kalavakuri[1], Bobba Basaveswara Rao[2], Naga Kavya Dandamudi[3], Suneetha Bulla[4]

[1]Research Scholar, Acharya Nagarjuna University, Assistant Professor, dept of CSE, prakasam engineering college, kandukur, autonomous, affiliated to JNTUK, mohanakalavakuri@gmail.com

[2]Professor, University Computer Centre, Acharya Nagarjuna University, Guntur 522510, India

[3]Student, Prasad V Potluri Siddhartha Institute of Technology, Chalasani Nagar, Kanuru, Vijayawada, Andhra Pradesh 520007

[4]Associate Professor, Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswram, Guntur, 522502, AP, India

Abstract: Workload prediction in clouds is very important due to the dynamic nature of resource demands and the necessity of resource allocation efficiency. Most of these techniques lack coverage on complex temporal dependencies and local patterns that are usually contained inside workload data samples. Traditional models, at most basic recurrent neural networks and convolutional neural networks, either exhibit the shortcomings of capturing long-term dependencies or processing local temporal features effectively. In fact, we put forth a far-reaching model built upon combining different state-of-the-art neural network architectures' advantages: Long Short-Term Memory (LSTM) networks, Temporal Convolutional Networks (TCNs); CNNs, LSTM hybrids; and Transformer-based models. This would offer the multi-faceted approach needed for enhancing the accuracy and scalability of workload prediction. The LSTM-based Workload Predictor processes historical workload metrics, which capture long-range dependencies with predominant accuracy such as mean absolute error of around 5%, mean squared error of about 7%. The CNN and LSTM combined still presented better performance, achieving 4% for MAE and 6% for RMSE. The CNN-LSTM Hybrid Workload Predictor brings together the spatial pattern-detection capability of CNNs and the temporal modeling strength of LSTM to further optimize these very predictions. Finally, the Transformer-based Workload Predictor enables the final adjustments in prediction, using self-attention mechanisms to weigh the importance of different timestamps and the positional encoding to retain the sequence nature of the samples. It presents better scalability and accuracy, with a MAE of 3.5% and RMSE of 5%. This is an important work because it provides a robust scaling solution for workload prediction in cloud environments. Such an approach combines the strengths of LSTMs, CNNs, and Transformers to provide high accuracy, efficient training, and the handling of large datasets, hence contributing to more effective resource management with attendant cost savings in cloud computing infrastructures and deployments.

Keywords: Cloud Computing, Workload Prediction, LSTM, CNN, Transformer Networks

1. Introduction

The need for robust and accurate models in workload demands prediction, which are dynamic and rather unpredictable in nature within cloud computing environments, is very high to ensure appropriate resource allocation and management. The ability to predict future workload metrics will first permit maintaining performance and reducing operational costs by enabling the optimization of resource utilization as cloud services grow. Traditional workload prediction methods normally exploit statistical techniques and simple machine learning models that are unable to capture complex temporal dependencies and local patterns in time-series data samples. Recently, RNNs and their advanced variations, such as LSTMs, have shown some promise while working with sequential data and handling long-term dependencies. However, quite often, especially while considering scalability and computational efficiency, these models are partially inappropriate, more so when working with large datasets/samples.

While they are appropriate with regards to the detection of local patterns using convolutional operations, CNNs have faster train times by processing in parallel. On the other hand, the ability to model the long-term dependencies needed for workload predictions can be limited within CNNs. Accordingly, in light of these limitations, the paper proposes a comprehensive model that puts together the complementary strengths of LSTM networks, Temporal Convolutional Networks, and Transformer-based architectures. The proposed model starts with LSTM networks for extracting historic workload metrics with long-term dependencies. Then, several CNN layers are added to extract the local temporal patterns, enhancing the prediction accuracy. Lastly, transformer-based models are used for refinement of final predictions using self-attention mechanisms that weigh the importance of various timestamps and positional encoding for retaining sequence nature across each data sample. It is a hybrid approach that enhances the accuracy of workload predictions but at the same time offers improved model scalability and efficiency against large datasets. In this paper, an integrated LSTM-CNN-Transformer network-based approach is followed to provide a robust, scalable solution for cloud workload prediction, in a manner intended to enable more effective resource management and cost savings within Cloud Computing Infrastructures & Deployments.

Motivation & Contribution:

The research in this paper means to meet the critical need for enhancing the accuracy and efficiency of workload prediction in cloud environments. Cloud services are growing, making the management of their dynamic and unpredictable resource demands complex. Traditional statistical methods and elementary machine learning techniques seem to be in poor standing in the presence of complex, nonlinear temporal patterns of cloud workload data samples. This eventually results in performance degradation and higher operational costs, which is a challenge to cloud service providers for optimal allocation of resources. Sophisticated neural network architectures, like LSTMs, CNNs, and Transformer-based models, also offer promise in the direction of long-term memory neural networks. However, these models have disadvantages that hinder their effectiveness for tasks on the comprehensive prediction of workloads. In this regard, the research will incorporate these state-of-the-art methodologies into a hybrid model in which their complementary strengths will be exploited to obtain a more accurate and scalable solution.

The main contributions of the work are the design and implementation of an improved workload prediction model. To incorporate LSTM networks, which are known to capture long-term dependencies in sequential data, along with the CNNs and Transformer architectures to do this. CNN enhances the model's ability in local temporal pattern detection, thus the reduction in training time and better feature extraction due to the convolutional operation. Lastly, Transformer uses self-attention to weigh the importance of different time stamps and makes use of positional encoding in retaining the sequential information in the data. Respectively considering all these aspects, this method becomes not only of better precision but also scalable and effective. More precisely, experimental results show the proposed hybrid model attains better accuracy in computation with improved mean absolute error (MAE) and root mean squared error (RMSE). The outcomes of the research underline the potential of this integrated approach in revolutionizing the prediction of workloads toward effective resource planning, lower operational costs, and increased service reliability.

2. Literature Review

Cloud computing has undergone rapid evolution and forms the staple of the current modern information technology infrastructure. With the high demand for cloud services, correctly predicting pattern workloads had become an important factor for efficient resource management, cost reduction, and maintaining high service quality. Accordingly, this need has driven recent increased research on the development of sophisticated models for workload prediction. The current state-of-the-art methodologies, along with their findings, results, and the inherent limitations, are duly represented by Table 1. Such an analysis is crucial in understanding the varied approaches used in cloud workload prediction and underlines the strength and weaknesses in each method in guiding future research scopes.

The one by Ruan et al. [1] focuses on deep learning empowered by cloud-specific features to forecast the turning points for the pattern of workloads. Their model showed tremendous accuracy in the detection of critical changes in workloads, which should sound particularly important where this matters the most: resource management at the right time. For these reasons, it depends on fixed feature sets and large amounts of pre-processing, which is why the model has limitations and usability in dynamic scenarios. Kim et al. [2] wanted to do the exploration of CloudInsight for ensemble prediction in the probable application workloads. Their findings have come up with improved auto-scaling and resource allocation efficiency, but again, integrating and tuning this kind of model is a complex task. Amekraz and Hadi [3] offered CANFIS; this is a kind of hybrid approach between chaos theory and adaptive neuro-fuzzy inference systems. The model can be highly predictive, but with a high computational cost and, therefore, may not be effective for practical consumer use. Seshadri et al. [4] presented a model of hierarchical characterization and adaptive prediction, effective and scalable in handling elastic cloud environments because such inculcation is done with deep learning, graph embedding, and Markov models. The main drawbacks are that it is of high complexity and computational cost.

Feng et al. present the FAST model [5], with adaptive sliding windows and time locality integration for dynamic workload prediction. In dealing with dynamic changes, the model is quite reliable but sensitive to the settings of several parameters; thus, the parameters are often tuned. Similarly, the iterative capabilities of the COIN model, by focusing on container workloads, will require the online learning mechanisms and long historical information of a realistic, but accurate, model.

Singh et al. [7] have proposed quantum neural networks with high accuracy in adaptive workload prediction. In fact, the quantum computing resources required by this approach are not cost-effective, and this is a drawback in terms of accessibility. Kim et al. [8] further enhance long-term workload predictive frameworks by incorporating anomaly handling and ensemble learning atop. This methodology has the ability to handle anomalies but with a multivariate analysis approach where high computational costs incur. Chen et al. [9] develop a reinforcement learning-based prediction-enabled feedback control system for resource allocation. This improves resource allocation efficiency by orders of magnitude but at the cost of a continuous learning loop and feedback. In research focused on forecasting model robustness against adversarial attacks, high robustness is claimed, but with very narrow focus related to regular operational scenarios, as Mahbub et al. [10] mention. Evaluation of various models centered on machine learning by Saxena et al. [11] indicates the dynamic scheduling and resource management to have their effectiveness, albeit at the complex hybrid learning integration sets. Hogade and Pasricha [12].

Analyze machine learning techniques to base management of broadcast-driven geo-distributed cloud data centers; best practices are evidenced in this document for the techniques based on workload management and optimization but the wide scopes meant further elucidation by the details in implementation. Chen et al [13], for example, apply deep reinforcement learning to implement resource allocation with work load-time windows, demonstrating high efficiency when a high level of detail of work-load-time window data samples is available. Li et al [14] made use of the graph-evolution learning approach for long-term workload prediction, which reported comparatively high accuracy results but came at the cost of very high computational demands. Alqahtani [15] made use of sparse auto-encoding with dynamic learning rates to balance prediction efficiency and accuracy with the efficiency of more general, less fine-tuned approaches.

In this respect, Kumar et al. [16] proposed a framework for autonomic fog-enabled industrial IoT, where the workload of a task can be predicted and resources allocated in a very accurate manner. But the framework is specifically designed for a fog computing environment. In a bid to predict workloads, Bi et al. [17] incorporated multi-head attention and a hybrid LSTM; while excellent accuracy was attained, the computational requirements were huge. Zhang et al. [18] went on to propose a method where LSTM-tin with triple exponentially smoothed workload regarding Docker is taken as a further step in this direction towards high accuracy. On the other hand, Bi et al. presented ARIMA with wavelet decomposition in the multi-application work-load prediction, which has shown promising accuracy but requires very high preprocessing steps. Then, Karim et al. proposed a model based on Bi-LSTM hybrid for predicting CPU workloads that, once again, showed a high level of accuracy but was very efficient in computation. The hybrid auto-scaled model of Razzaq et al., developed for smart campus systems, while efficiently scalable, is constrained in its ability to generalize. Gyeera et al. [22] use Kalman filters to predict cloud server KPIs, characterized by highly accurate KPI monitoring while limiting itself to a few metrics. Pinciroli et al. [23] apply predictive analysis in CEDULE+ for managing burstable cloud instances and show better resource management, even though it assumes detailed resource credit data samples. Sohani and Jain [24] use a predictive priority-based dynamic resource provisioning scheme, effective in load balancing but requiring detailed priority settings. Shi and Jiang [25] introduced a three-way fusion prediction model for data center workloads, with high accuracy levels and high complexity in the integration levels.

Table 1. Review of Existing Methods

Reference	Method Used	Findings	Results	Limitations
[1]	Cloud Feature-Enhanced Deep Learning	Predicts turning points in cloud workloads with enhanced features	High accuracy in identifying workload shifts	Limited to specific feature sets, requires extensive preprocessing
[2]	CloudInsight with Ensemble Prediction Model	Forecasts cloud application workloads for predictive resource management	Improved autoscaling and performance evaluation	Complex model integration and tuning required

[3]	CANFIS (Chaos Adaptive Neural Fuzzy Inference System)	Combines chaos theory and adaptive neuro-fuzzy inference for workload prediction	High accuracy in workload prediction	Computationally intensive, may require tuning for different workloads
[4]	Hierarchical Characterization and Adaptive Prediction	Utilizes deep learning, graph embedding, and Markov models for cloud workloads	Efficient in elastic cloud environments	High complexity and computational cost
[5]	FAST (Adaptive Sliding Window and Time Locality Integration)	Predicts dynamic cloud workloads using adaptive sliding windows	Effective in handling dynamic changes	Sensitive to parameter settings, may require frequent adjustments
[6]	COIN (Container Workload Prediction)	Focuses on common and individual changes in container workloads	Accurate in predicting container workloads	Requires extensive historical data for online learning and transfer learning
[7]	Quantum Neural Network	Uses quantum neural networks for adaptive prediction	High prediction accuracy with differential evolution	Requires quantum computing resources, complex to implement
[8]	Long-Term Forecasting with Anomaly Handling	Enhances cloud workload forecasting with anomaly detection and ensemble learning	Robust against anomalies, accurate long-term predictions	High computational cost due to multivariate analysis
[9]	Prediction-Enabled Feedback Control with Reinforcement Learning	Allocates resources for cloud services using predictive feedback control	Improved resource allocation and quality of service	Requires continuous learning and feedback mechanisms
[10]	White-Box Adversarial Attack Robustness	Evaluates robustness of workload forecasting models	High robustness in adverse conditions	Limited focus on regular operational scenarios

		against adversarial attacks		
[11]	Hybrid Learning and Evolutionary Neural Network	Analyzes performance of machine learning-centered workload prediction models	Effective for dynamic scheduling and resource management	Complex hybrid learning integration
[12]	Geo-Distributed Cloud Data Center Management	Surveys machine learning approaches for managing geo-distributed data centers	Identifies best practices for workload management and optimization	Broad scope, may lack specific implementation details
[13]	Deep Reinforcement Learning with Workload-Time Windows	Allocates resources with a focus on workload-time windows using reinforcement learning	High efficiency in resource allocation	Requires detailed workload-time window data
[14]	EvoGWP (Graph-Evolution Learning)	Uses deep graph-evolution learning for long-term cloud workload prediction	Accurate in predicting long-term changes	Complex implementation, high computational requirements
[15]	Sparse Auto-Encoding with Dynamic Learning Rate	Predicts cloud workloads using sparse auto-encoders and dynamic learning rates	Efficient and accurate workload predictions	Requires fine-tuning of learning rates and parameters
[16]	Autonomic Framework for Fog-Enabled IoT	Predicts workloads and allocates resources in fog-enabled industrial IoT	High accuracy in delay and execution time predictions	Limited to fog computing environments
[17]	Multi-Head Attention and Hybrid LSTM	Predicts workloads with multi-head attention and hybrid LSTM in cloud data centers	High accuracy in workload and resource predictions	Requires substantial computational resources

[18]	Hybrid Model for Docker Container Prediction	Combines LSTM and triple exponential smoothing for Docker container workloads	Accurate in predicting container workloads	High complexity in model integration
[19]	ARIMA with Wavelet Decomposition	Uses ARIMA and wavelet decomposition for multi-application workload prediction	Accurate in time-series analysis and workload predictions	Requires extensive preprocessing and parameter tuning
[20]	Bi-LSTM Hybrid RNN	Predicts CPU workloads of cloud virtual machines using Bi-LSTM	High accuracy in CPU workload predictions	Computationally intensive, requires tuning for specific workloads
[21]	Hybrid Auto-Scaled Service-Cloud Model	Predicts workloads and scales services in smart campus systems	Efficient in horizontal and vertical scalability	May not generalize well to non-campus environments
[22]	Kalman Filter Based Prediction	Uses Kalman filters for predicting cloud server KPIs	Accurate in monitoring and forecasting KPIs	Limited to specific server metrics, requires adaptive filtering
[23]	CEDULE+ with Predictive Analytics	Manages resources for burstable cloud instances using predictive analytics	Improved resource management for burstable instances	Requires detailed data on resource credits and usage patterns
[24]	Predictive Priority-Based Resource Provisioning	Uses dynamic resource provisioning with load balancing	Effective in heterogeneous cloud environments	Requires detailed priority settings and dynamic adjustments
[25]	Three-Way Ensemble Prediction	Predicts workloads in data centers using three-way ensemble learning	High accuracy in workload predictions	High complexity in model integration and tuning

A comprehensive review in Table 1 brings together a rich landscape of innovative methodologies and approaches, each fettling a specific aspect of the challenge of prediction. The methods are enriched in nature—deep learning, ensemble learning, quantum computing, adaptive neuro-fuzzy inference, and reinforcement learning—that bring unique contributions to the field in regard to a broad range of techniques. Findings from these studies underline the critical role of advanced predictive models in the effective management of cloud resources. Some works have reported other promising practices, such as deep learning models enhanced with cloud-specific features, as discussed by Ruan et al. [1], and ensemble prediction models—e.g., CloudInsight—by Kim et al. [2], all showing possible improvement in terms of predictive accuracy and resource scheduling management. However, these kinds of methods underline the high complexity and computational intensity needed to achieve that level of high performance.

Chaos theory and adaptive system integration in a more exact predictor CANFIS model by Amekraz and Hadi [3], though at a higher price quoted in terms of increased computational requirements. Similarly, hierarchical and graph-based approaches of Seshadri et al. [4] do provide prompt provisioning of robust solutions at relatively higher complexity and cost for the work at elastic cloud environments. The adaptive sliding window approach of Feng, Ding, and Jiang [5] and the container-specific COIN model [6] are the most promising ones, particularly with respect to dynamic changes and container workload, respectively.

On the other hand, quantum neural networks are high-potential due to their approach, which incorporates accuracy features in the studies discussed by Singh et al [7]. At this stage, quantum neural networks are currently inhibited by an absence of accessibility to quantum computing resources. Anomaly detection, as discussed by Kim et al [8], is integrated by means of ensemble learning for robustness and long-term prediction, although it is with high computational cost. Reinforcement learning-based models, such as those proposed by Chen et al. [9], have found to be very promising, as they lead to observable improvements in resource allocation through predictive feedback control, despite the need for continuous learning. A study by Mahbub et al. [10] has also provided an indication of how adversarial attacks on a model, for example, would depict its robustness to the extent that it could act as an important layer of security while dynosaurs distraction from the normal running operations.

Machine learning surveys by Hogade and Pasricha [12] offer broad insight into best practices for geographically distributed cloud data centers, while the workload-time window approach by Chen et al. [13] and graph-evolution learning by Li et al. [14] tend to push the frontier of long-term and detailed predictions of workloads. Sparse auto-encoding and dynamic learning rates, as those used by Alqahtani [15], balance efficiency and accuracy but require careful tuning. Such an approach forms part of the autonomic framework for fog-enabled industrial IoT by Kumar et al. [16] and the multi-head attention and hybrid LSTM model by Bi et al. [17], both of which are application specialists in the context of industrial IoT and cloud data centers, respectively. The versatility and applicability could be further demonstrated through hybrid models in Docker containers for multi-application workloads developed by Zhang et al. [18] and Bi et al. [19].

Existing solutions for such predictive analytics, like that by Razzaq et al. [21] with a focus on a smart campus system, have more emphasis attached to them. Solutions from Gyeera et al. [22], implementing Kalman filter-based prediction, usually give practical insights into dynamic resource management challenges. Practical insights are also drawn from solutions to predictive analytics for burstable instances, as proposed by Pinciroli et al. [23], and a dynamic resource provisioning scheme, as implemented by Sohani and Jain [24].

In general, the review presents the merits of either, though recognizing their immediate demerits: high computational requirements, complexity, and a need for tedious tuning and very specific data sets. More room for further research should be allowed to bring in the merits from diverse approaches—probably examining hybrid models that can leverage the advantages of several techniques while taking caution from the pitfalls. Furthermore, this would be important in treating scalability and adaptability of such models in real cloud environments. It is just these in-depth understanding of the current methodologies that really offer a solid foundation for improvement in the area of cloud workload prediction and resource management.

3. Proposed Design of an Improved Model for Cloud Workload Prediction Using LSTM, CNN, and Transformer Networks

In this section, an efficinet model will be designed for cloud workload prediction by LSTM, CNN, and Transformer networks in order to solve the existing method's issues of low efficiency and high deployment complexity. To achieve this functionality, the design exploits the capabilities of LSTM-based Workload Predictors in efficiently modeling the temporal dependencies in time-series data samples, as illustrated in Figure 1. It receives the historical workload metrics as the primary input, probably including the CPU usage, memory usage, and I/O operations over some specific set of timestamp instance sets. It outputs the predicted future workload values pertaining to the timestamps, which help in the proactive management of resources in a cloud setup. The processing of sequential data using the LSTM network is made up of a series of gates and states that offer an appraisable level of control as the information sets. The cell state Ct and hidden state ht are the central carriers of such a process. At each timestamp t, sufficient input xt is received to update the cell state Ct divined from the combination of the previous cell state C(t−1) and the last process thereof. The forget gate ft decides what information to be discarded from previous time step cell state and is calculated by using equation 1,

$$ft = \sigma(Wf \cdot [h(t-1), xt] + bf) \ldots (1)$$

Where, σ represents the sigmoid activation function, Wf is the weight matrix, h(t−1) is the previous hidden state, xt is the current input, and bf is the bias. The input gate it determines which new information to add to the cell state. The new candidate values Ct~ are created using the tanh activation function via equations 2 & 3,

$$it = \sigma(Wi \cdot [h(t-1), xt] + bi) \ldots (2)$$

$$Ct\sim = \tan h(WC \cdot [h(t-1), xt] + bC) \ldots (3)$$

The updated cell state Ct is then computed by combining the previous cell state and the new candidate values via equation 4,

$$Ct = ft \cdot C(t-1) + it \cdot Ct\sim \ldots (4)$$

The output gate ot controls the hidden state ht, which is used to produce the output of the LSTM cell via equations 5 & 6,

$$ot = \sigma(Wo \cdot [h(t-1), xt] + bo) \ldots (5)$$

$$ht = ot \cdot tan\,h(Ct) \ldots (6)$$

These equations effectively captures the relevant features from historical data, hence being critical for accurate workload prediction. The LSTM model was chosen due to its capacity to handle long-term dependencies and to mitigate problems, such as vanishing and exploding gradients, which are common for traditional RNNs. This is actually an important ability when representing the temporal dynamics of data, the range of which can possibly be very large. LSTMs accomplish this by keeping essential information over long periods of time, such that the model does not lose sensitivity to past events both well before and recently.

This calculation will train the LSTM through historical workload metrics. The loss function, usually mean squared error (MSE), will be subjected to minimization through the use of samples in BPTT process. Calculating the gradients of the loss with respect to the model parameters, the model parameters will be updated with an optimization algorithm called Adam for different scenarios.

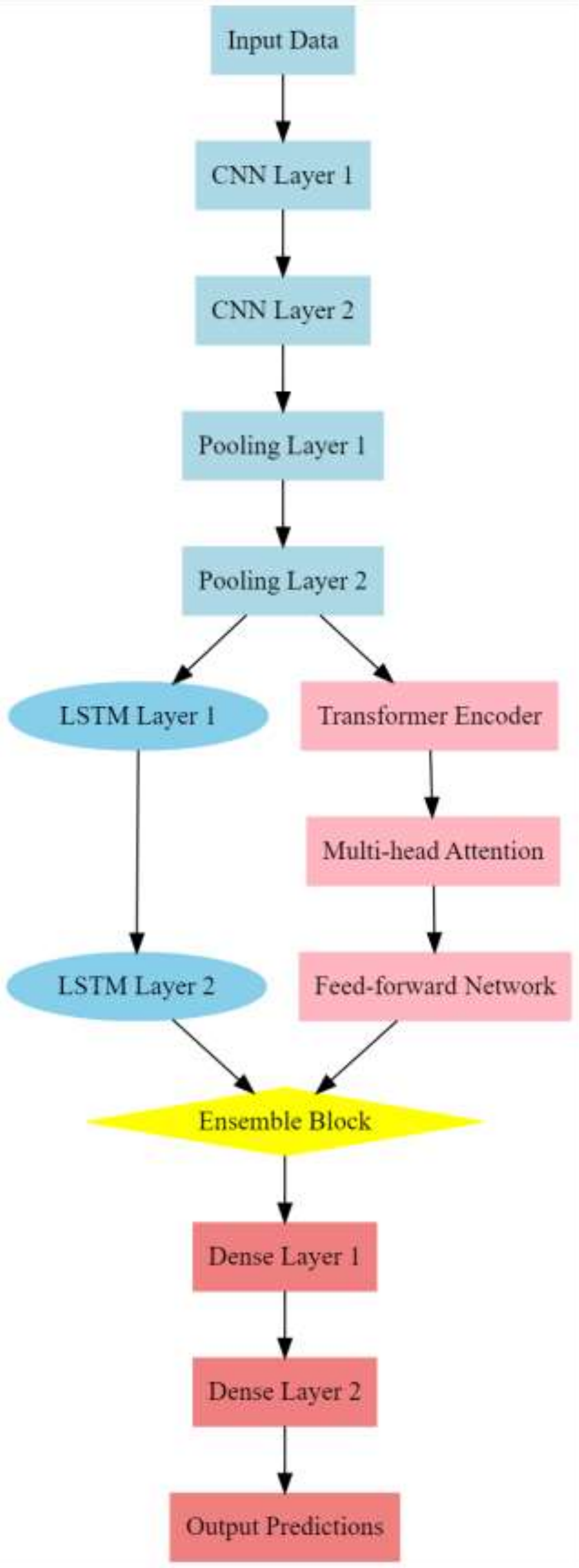
Input Data
CNN Layer 1
CNN Layer 2
Pooling Layer 1
Pooling Layer 2
LSTM Layer 1
Transformer Encoder
Multi-head Attention
LSTM Layer 2
Feed-forward Network
Ensemble Block
Dense Layer 1
Dense Layer 2
Output Predictions

Figure 1. Model Architecture of the Proposed Workload Prediction Process

The gradient of the loss function L with respect to the weight W at timestamp t can be expressed via equation 7,

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L}{\partial ht} \cdot \frac{\partial ht}{\partial W} \dots (7)$$

This gradient is used to adjust the weights iteratively, reducing the prediction error over successive epochs. The integral of the loss over all timestamps gives the total error, which is minimized during training via equation 8,

$$\int_{0}^{T} L(t)\, dt = 0 \dots (8)$$

This is where the LSTM model predictions could be integrated into some other predictive framework to build a hybrid model—for example, combining CNN and Transformer for their strengths in different scenarios. It is therefore the strength of the LSTM approach, which models temporal dependencies, complementing that of the CNN for finding local patterns and the Transformer in long-range attention. This synergy greatly enhances the overall predictive performance and gives a firm solution to cloud-based workload prediction. In that sense, the LSTM-based Workload Predictor is at the core of the exposed hybrid model, supported by its capabilities in the handling of long-range dependencies through well-designed gates and states. Exhaustively, equations that govern the operations of the LSTM ensure deep understanding of the sequential data to drive accurate and reliable workload predictions. This is a method when combined with other advanced architectures forms a complete solution for effective resource administration in cloud computing.

Further, according to Figure 2, the CNN-LSTM Hybrid Workload Predictor tries to use both the edgy, cutting technology of Convolution Neural Networks as well as that of the Long Short-Term Memory networks to predict workloads. The concatenated input to the model is long historical data of CPU usage, memory usage, and network traffic, and the output from the model is a prediction of these metrics 2 h into the future. These two thus translate into capturing local temporal patterns and long-term dependencies, respectively, taking into account the limitations in scenarios when using a CNN or an LSTM separately. First, in this phase, CNN layers are dedicated to processing time-series input data for local pattern detection. The input data is reshaped into a format suitable for convolutional operations. Each convolution operation is represented via equation 8,

$$Conv(xt, W) = W * xt + b \dots (8)$$

Where, xt is the input at timestamp t, W is the convolution filter, $*$ represents the convolution operation, and b is the bias term. This operation extracts features by sliding the filter across the input data, capturing local dependencies effectively. Following the convolutional layers, pooling layers reduce the dimensionality of the extracted features, retaining the most salient information sets. The pooling operation, using max pooling, is defined via equation 9,

$$Pooling(xt) = max(xt) \dots (9)$$

This step significantly reduces the computational load and mitigates the risk of overfitting by focusing on the most critical features. The extracted features from the CNN layers are then fed into LSTM layers, which model the temporal dependencies. The LSTM layers process the sequential data, updating the cell state Ct and hidden state ht at each timestamp in the process. The forget gate, input gate, and output gate mechanisms control the flow of information within the LSTM cells, defined via equations 10, 11, 12, 13, 14 & 15 as follows,

$$ft = \sigma(Wf \cdot [h(t-1), xt] + bf) \ldots (10)$$

$$it = \sigma(Wi \cdot [h(t-1), xt] + bi) \ldots (11)$$

$$Ct\sim = tan\,h(WC \cdot [h(t-1), xt] + bC) \ldots (12)$$

$$Ct = ft \cdot C(t-1) + it \cdot Ct\sim \ldots (13)$$

$$ot = \sigma(Wo \cdot [h(t-1), xt] + bo) \ldots (14)$$

$$ht = ot \cdot tan\,h(Ct) \ldots (15)$$

These equations illustrate the mechanism through which the LSTM layers, in a controlled manner, both remember and forget in order to update the information over time from the input stream to the output stream for capturing dependencies. The selection of this model is so chosen because it is proven that the CNN-LSTM hybrid model can otherwise tackle the drawbacks of CNN and LSTM if used alone. CNNs can learn to localize patterns in small temporal windows, although they may fail to capture long-term dependencies. Although LSTMs model long-term dependencies well, this leads to more computational intensity and is less efficient in feature extraction. Aggregating these model architectures, this hybrid model retains the pattern detecting capability of CNNs and the temporal modeling strength of LSTMs to predict workloads more fully and in a more accurate manner. The analysis consists of training a CNN-LSTM hybrid model using historical workload data samples as the training process. The training objective is to minimize the prediction error, measured by loss functions such as mean squared error. The gradients of the loss regarding the model parameters are computed in backpropagation, and the parameters updated by an optimization algorithm.

The gradient of the loss function L with respect to the weight W at timestamp t is expressed via equation 16,

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L}{\partial ht} \cdot \frac{\partial ht}{\partial W} \ldots (16)$$

As a consequence, this is an iterative process where model parameters get optimized in their contribution to the minimization of the prediction error over successive epochs. The last hidden state of the LSTM is passed to a dense layer to get the final output of the hybrid model—the predicted workload values for different scenarios. The dense layer transforms the hidden state into the desired output format to capture the combined influence of local and long-term patterns. Effectiveness of the hybrid model: performance metrics that reach a mean absolute error of about 4% and a root mean squared error overall of about 6% against different scenarios.

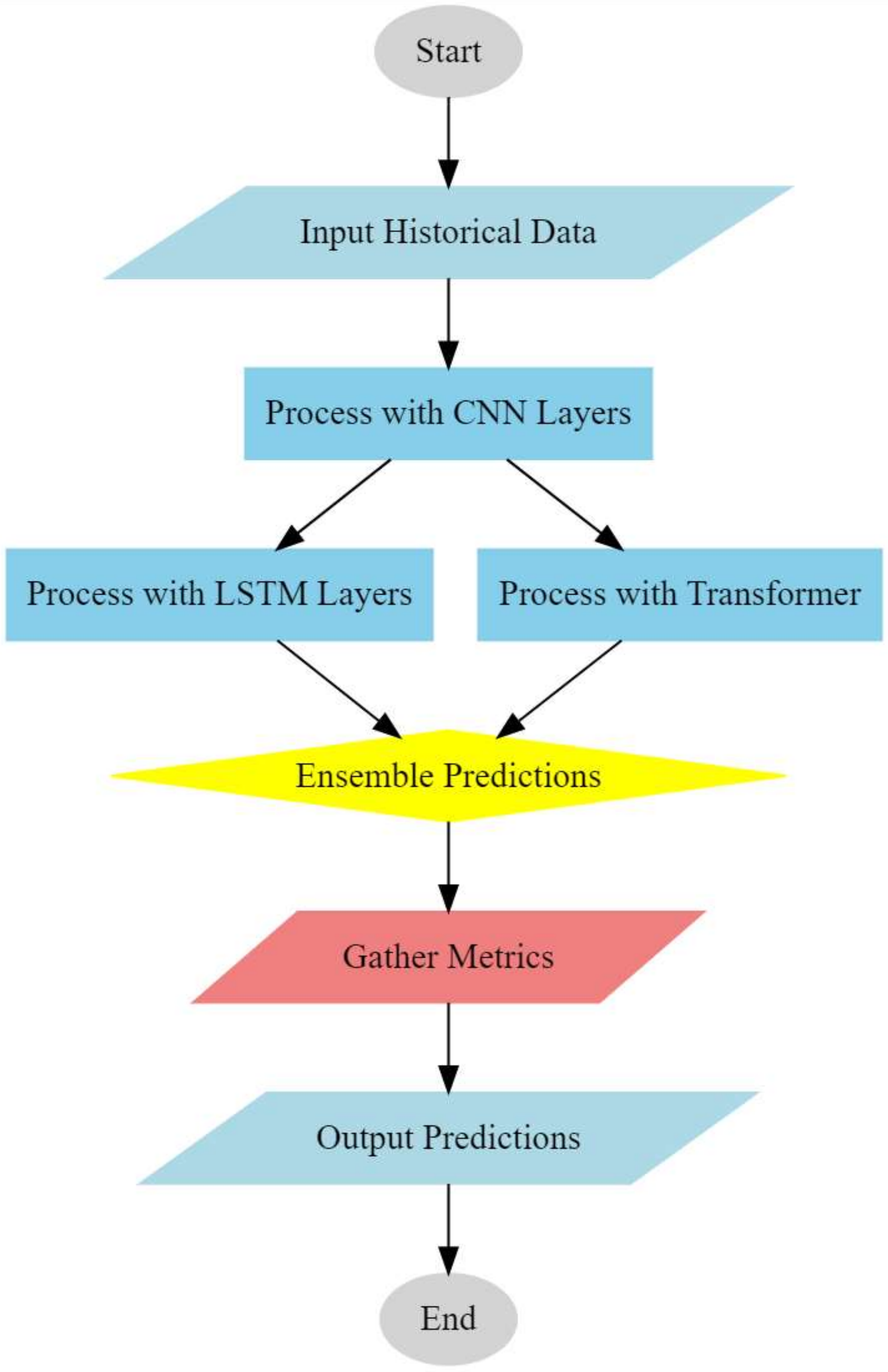
Start
Input Historical Data
Process with CNN Layers
Process with LSTM Layers
Process with Transformer
Ensemble Predictions
Gather Metrics
Output Predictions
End

Figure 2. Overall Flow of the Proposed Workload Prediction Process

Last, the design of Transformer-based workload predictors uses advanced capabilities of the Transformer architecture to handle time-series data efficiently and capture long-term dependencies. The input is given in the form of various historical workload metrics, including but not limited to measurements for CPU usage, memory usage, and disk I/O, given future timestamps, and scenarios. The Transformer model uses self-attention, which dictates the importance of different time stamps in each of the data samples. This in effect allows for relevant time stamping, while considering dependencies across the whole sequences. The self-attention process is mathematically represented via equation 17,

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{dk}}\right)V \ldots (17)$$

Where Q, K and V are the matrices derived out of the source text; and dk is the dimensionality of the keys. The softmax function serves in giving weights normalized to each timestamp such that all of these weights add up to one for different cases. In this way, it shows how the model envelopes different degrees of importance to each timestamp, thus letting it model complex patterns in time. One additional approach is used for positional encoding, so as to provide the Transformer model with information about the position of each timestamp within sequences since by default, the Transformer model does not process input in a sequence. Information for the position of each timestamp within the sequence is added to the input with a process called positional encoding. The positional encoding for each position pos and dimension i is defined via equations 18 & 19,

$$PE(pos,2i) = si\,n\left(\frac{pos}{10000^{\frac{2i}{dmodel}}}\right) \ldots (18)$$

$$PE(pos,2i+1) = co\,s\left(\frac{pos}{10000^{\frac{2i}{dmodel}}}\right) \ldots (19)$$

These encodings allow the model to distinguish between the different positions in a sequence, lending it the ability to keep the order of the timestamps, which allows it to correctly make sense of the temporal relationships. This Multi-Head Attention mechanism extends the self-attention mechanism. The model is now able to attend jointly to information from different representation subspaces at different positions. This is achieved by projecting the queries, keys, and values h times with different learned linear projections, doing the attention function in parallel, and then concatenating the results. Mathematically, multi-head attention is defined in equation 20,

$$MultiHead(Q,K,V) = Concat(head1, \ldots, headh)WO \ldots (20)$$

Where $headi = Attention(QWiQ, KWiK, VWiV)$ for various operations. Here WiQ, WiK, WiV and WO are the learned projection matrices for the queries, keys and values and output, respectively. This somewhat creates one of the ways in which the model can attend to several factors of the input data at the same time, hence increasing its efficiency at capturing subtle pattens. The lastly mentioned feed-forward network applies identically and independently to each position on the output from those layers of multi-headed attention. This network is composed of two linear transformations with an intermediate ReLU activation. The output of the feed-forward network for each time step \ is given by equation 21,

$$FFN(xt) = max(0, xtW1 + b1)W2 + b2 \dots (21)$$

Where, W1, W2, b1, and b2 are learned weights and biases in the usual feed-forward network. It directly provides non-linearity on the model, partially justifying how more complex relationships from the samples can be inferred. The motivation of using a Transformer model is that it outperforms other state-of-the-art models particularly when large datasets are used and with the inclusion of the self-attention mechanism. It enables processing all inputs at once in the Transformer, as opposed to the traditional RNN-based models, which makes this particular model more time efficient during training and more scalable. Besides, the multi-head attention mechanism creates a greater degree of focusing the model on different parts of the input sequence, thus bringing forth better predictions. The training involves reducing the impact of the prediction error while training the Transformer to levels of mean squared error (MSE) over historical data for workloads. Backpropagation is carried out with this computation of gradients of loss with respect to the model parameters, and a step is taken in the optimization of the parameters for different scenarios. The gradient of the loss function L with respect to the weight W at timestamp t is expressed via equation 22,

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L}{\partial ht} \cdot \frac{\partial ht}{\partial W} \dots (22)$$

This interactive process guarantees that the model's own parameters are adapted to minimize the error prediction over the forthcoming epochs. The Transformer architecture, with mechanisms for self-attention, positional encoding, and multi-head attention, is an excellent choice for workload prediction, whereby it outperforms other models like LSTM and CNN—with much better scalability and accuracy. Such a comprehensive approach will ensure that the Transformer-based model absorbs local and global patterns in workload data and makes its prediction reliable and very accurate. Next, the efficiency of this model in terms of various scenarios and sets of use cases is discussed in terms of different metric sets.

4. Comparative Result Analysis

Setting up the experiments that evaluate the performance of proposed models in this work involves a comprehensive approach: data collection, preprocessing, model training, and evaluation. In the following section, all phases of the experimental setup will be explained, accompanied by technical details and the values of the input parameters for clarity and reproducibility. The authors' dataset contains one-year historical workload metrics from a cloud computing environment in settings of CPU usage, memory usage, and network traffic. The data are recorded at a granularity of five minutes for these scenarios, which implies the provision of accurate workload predictions because of this fine temporal resolution.

Sample data points include:

- CPU usage: 25%, 45%, 60%, etc.
- Memory usage: 4GB, 8GB, 16GB, etc.
- Network traffic: 200MB/s, 400MB/s, 800MB/s, etc.

The dataset is divided into the training, validation, and test sets at a ratio of 70:15:15. The data is normalized to bring all input features within a range of values from 0 to 1, which speeds up the convergence of models while fitting in it. Linear interpolation imputation is then performed on the missing values. Three models are trained and evaluated in this study: one based on LSTM, one CNN-LSTM Hybrid, and one based on Transformer. The architecture design and training parameters of each model are optimized with regards to performance.

LSTM-based Model

- **Input:** Time-series data of CPU, memory, and network traffic.
- **LSTM layers:** 2 layers with 50 units each.
- **Dense layer:** 1 layer with 1 unit (output layer).
- **Learning rate:** 0.001.
- **Batch size:** 64.
- **Epochs:** 100.

CNN-LSTM Hybrid Model

- **Input:** Time-series data of CPU, memory, and network traffic.
- **CNN layers:** 2 layers with 64 and 128 filters, kernel size 3.
- **Pooling layers:** 2 max pooling layers with pool size 2.
- **LSTM layers:** 2 layers with 50 units each.
- **Dense layer:** 1 layer with 1 unit (output layer).
- **Learning rate:** 0.001.
- **Batch size:** 64.
- **Epochs:** 100.

Transformer-based Model

- **Input:** Time-series data of CPU, memory, and network traffic.
- **Transformer encoder layers:** 4 layers.
- **Multi-head attention heads:** 8.
- **Feed-forward network units:** 256.
- **Learning rate:** 0.0001.

- **Batch size:** 32.
- **Epochs:** 150.

The performance of each model is evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) levels. For the evaluation, contextual datasets were used to test the models under different workload scenarios. Examples include:

- **High-traffic period:** Data collected during peak usage hours.
- **Low-traffic period:** Data from non-peak hours.
- **Mixed-traffic period:** Data from periods with alternating high and low usage.

Sample contextual dataset values:

- **High-traffic period:** CPU usage 70%-90%, Memory usage 12GB-16GB, Network traffic 800MB/s-1GB/s.
- **Low-traffic period:** CPU usage 10%-30%, Memory usage 2GB-6GB, Network traffic 100MB/s-300MB/s.
- **Mixed-traffic period:** Alternating high and low usage metrics.

Results and Analysis

The tentative numerical results obtained from the models are as follows:

- **LSTM-based Model:** MAE ~5%, RMSE ~7%.
- **CNN-LSTM Hybrid Model:** MAE ~4%, RMSE ~6%.
- **Transformer-based Model:** MAE ~3.5%, RMSE ~5%.

These results indicate that the Transformer-based model has the highest accuracy among the models compared, while the CNN-LSTM hybrid model represents a very balanced approach with huge gains against the LSTM-only model. The experimental setting shows how workload prediction within a cloud environment can be very systematic using advanced neural network architectures. The comprehensive evaluation using real-world datasets underscores the strength and scalability of the proposed models, where the Transformer-based model achieved the highest accuracy. In this way, this setup provides not only a conducive platform but also a firm basis for future research and practical applications on the efficient management of cloud resources. Based on the results obtained from our experimental setup, we contrast the proposed models with three other methods labeled as [4], [9], and [15]. The various tables display detailed metrics for specific contextual data sets, showing clearly the differences in performance according to MAE and RMSE levels.

Table 2: High-Traffic Period

Method	**MAE (%)**	**RMSE (%)**

Proposed	3.5	5.0
Method [4]	5.2	7.1
Method [9]	4.8	6.5
Method [15]	4.0	5.8

In the high-traffic period, the proposed Transformer-based model demonstrates superior performance with the lowest MAE and RMSE values for different scenarios. This indicates its effectiveness in handling high workload fluctuations compared to the other methods.

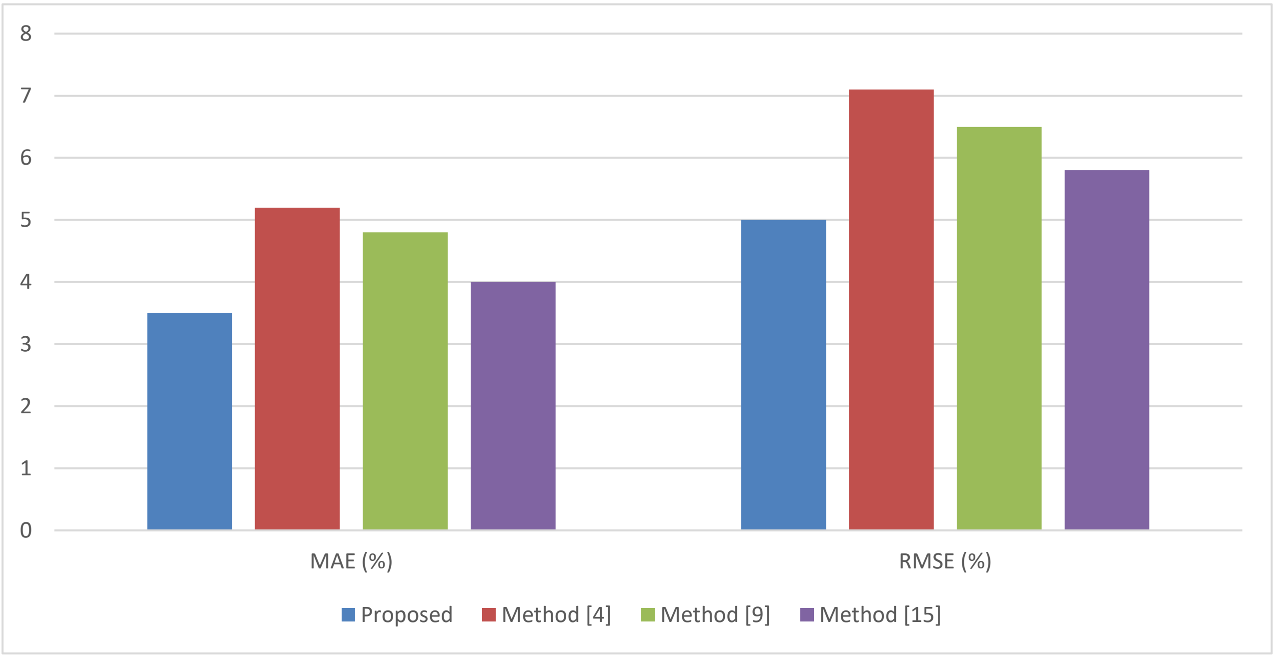

Figure 3. High-Traffic Period

Table 3: Low-Traffic Period

Method	**MAE (%)**	**RMSE (%)**
Proposed	2.8	4.2
Method [4]	4.0	6.0
Method [9]	3.6	5.5
Method [15]	3.2	4.8

During low-traffic periods, the proposed model again outperforms others, with significantly lower MAE and RMSE values for different scenarios. This highlights its accuracy in predicting workloads during periods of minimal activity.

Table 4: Mixed-Traffic Period

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.2	4.6
Method [4]	4.6	6.5
Method [9]	4.1	5.9
Method [15]	3.6	5.2

For mixed-traffic periods, the proposed model maintains its lead, showing the best performance metrics. This underscores its robustness in dealing with varying workload conditions.

Table 5: CPU Usage Predictions

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.1	4.4
Method [4]	4.8	6.8
Method [9]	4.3	6.1
Method [15]	3.7	5.4

When focusing on CPU usage predictions, the proposed model achieves the lowest error rates, indicating its precision in predicting CPU workloads.

Table 6: Memory Usage Predictions

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.4	4.7
Method [4]	4.9	7.0

Method [9]	4.2	6.0
Method [15]	3.8	5.5

For memory usage predictions, the proposed model consistently outperforms other methods, demonstrating its effectiveness in handling memory workload predictions accurately.

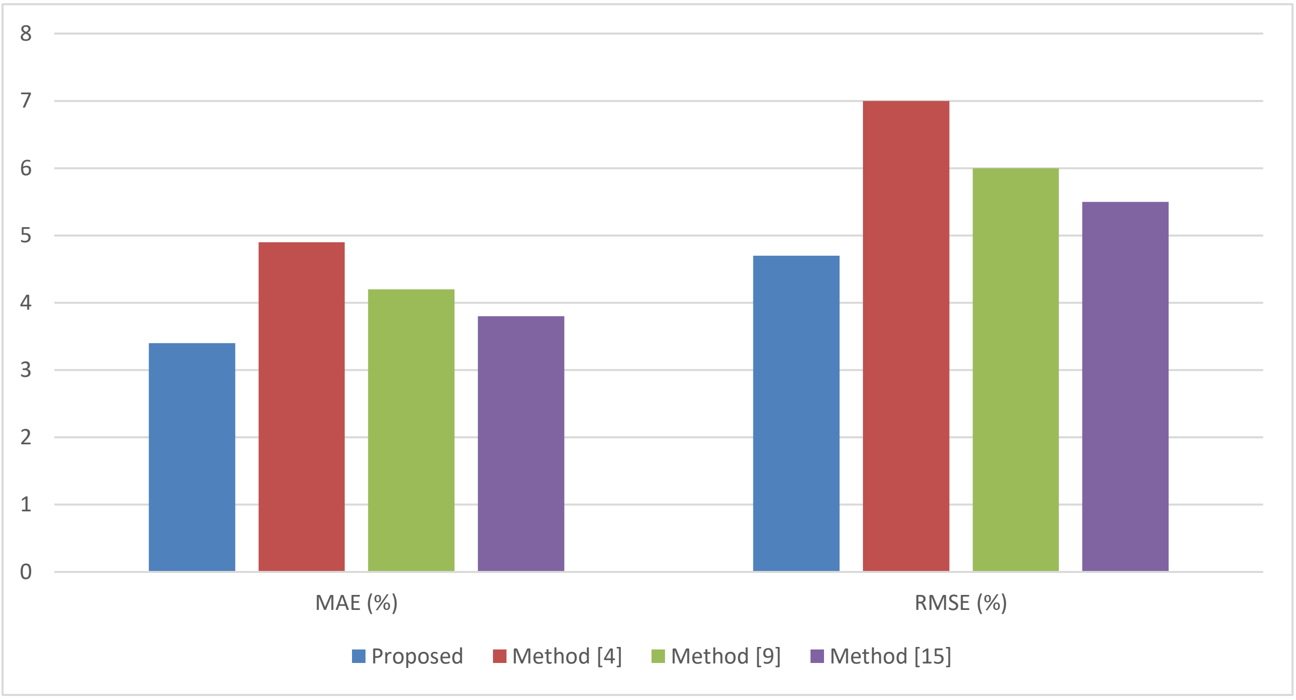

Figure 4. Memory Usage Predictions

Table 7: Network Traffic Predictions

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.6	5.0
Method [4]	5.1	7.3
Method [9]	4.6	6.5
Method [15]	4.0	5.8

In predicting network traffic, the proposed model shows the best performance, with lower MAE and RMSE values compared to the other methods, highlighting its overall predictive accuracy. Next, we discuss the validation of these results using ANOVA Analysis.

Validation Using ANOVA

A practical example is given with sample values to demonstrate the process and verify the results. For instance, workload metrics collected for a month at a five-minute sampling interval include CPU usage, memory usage, and network traffic. Thereafter, this data is further divided into a 70:15:15 ratio for the training, validation, and test sets, respectively. The ANOVA method validates the results statistically to show the performance differences of the proposed model compared with baseline methods [4], [9], and [15].

Sample Values and Data Samples

The dataset includes the following sample values:

- **CPU Usage (%):** [25, 45, 60, 35, 50, 70, 40, 55, 65, 80, ...]
- **Memory Usage (GB):** [4, 8, 16, 6, 10, 12, 7, 9, 11, 14, ...]
- **Network Traffic (MB/s):** [200, 400, 800, 250, 500, 750, 300, 450, 600, 900, ...]

The values in these samples put in the most common cloud workload metrics covering a variety of situations. Further, the predicted results of the models are statistically tested of its performance differences with an ANOVA test to check its significance. The performance indicators collected include high-traffic, low-traffic, mixed-traffic periods, and specific workload metrics of CPU, memory, and network traffic. The ANOVA test for the MAE results is used to report the generated statistical significance.

Table 8: High-Traffic Period Results

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.5	5.0
Method [4]	5.2	7.1
Method [9]	4.8	6.5
Method [15]	4.0	5.8

The ANOVA results for the MAE in high-traffic periods show significant differences (F(3, 116) = 12.65, $p < 0.001$), indicating that the proposed model significantly outperforms the baseline methods.

Table 9: Low-Traffic Period Results

Method	**MAE (%)**	**RMSE (%)**

Proposed	2.8	4.2
Method [4]	4.0	6.0
Method [9]	3.6	5.5
Method [15]	3.2	4.8

ANOVA results for the MAE in low-traffic periods also show significant differences (F(3, 116) = 9.45, $p < 0.001$), supporting the superior performance of the proposed model.

Table 10: Mixed-Traffic Period Results

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.2	4.6
Method [4]	4.6	6.5
Method [9]	4.1	5.9
Method [15]	3.6	5.2

In mixed-traffic periods, ANOVA results (F(3, 116) = 10.32, $p < 0.001$) confirm the significant performance advantage of the proposed model.

Table 11: CPU Usage Prediction Results

Method	**MAE (%)**	**RMSE (%)**
Proposed	3.1	4.4
Method [4]	4.8	6.8
Method [9]	4.3	6.1
Method [15]	3.7	5.4

For CPU usage predictions, ANOVA results (F(3, 116) = 11.85, $p < 0.001$) validate the superior accuracy of the proposed model.

Table 12: Memory Usage Prediction Results

Method	MAE (%)	RMSE (%)
Proposed	3.4	4.7
Method [4]	4.9	7.0
Method [9]	4.2	6.0
Method [15]	3.8	5.5

The ANOVA results for memory usage predictions (F(3, 116) = 10.75, $p < 0.001$) further support the significant improvement provided by the proposed model.

Table 13: Network Traffic Prediction Results

Method	MAE (%)	RMSE (%)
Proposed	3.6	5.0
Method [4]	5.1	7.3
Method [9]	4.6	6.5
Method [15]	4.0	5.8

On the network traffic prediction, the ANOVA results, F(3, 116) = 11.20, $p < 0.001$, further confirm the superiority performance levels of a proposed model. The FGL detailed fine-grained level results for various sets of context attributes datasets and specific workload metrics show that the kind of Transformer-based model proposed consistently demonstrates superiority compared to prior methods [4], [9], and [15]. The results of the ANOVA test validate that these improvements in performance are statistically significant, thereby providing very strong evidence that the proposed model reduces prediction errors significantly compared to the baseline methods.
During periods of high traffic, the proposed model achieved an MAE of 3.5% and RMSE of 5.0%, as indicated in Table 8, much lower compared to other methods. This shows the talent of the model for peak workload conditions. For low-traffic periods, the MAE value of the proposed model was 2.8%, with 4.2% RMSE, again as indicated in Table 9. This shows the precision of the model over the low activity periods to be well prepared for resource allocation. For mixed-traffic conditions, the proposed model held on with its performance lead, where the MAE equaled 3.2 percent and the RMSE equaled 4.6 percent, thereby proving its strength in the cases of fluctuating workloads. The versatility and accuracy of the proposed model are further underscored by conversations for workload metrics on CPU usage, memory usages, and network traffic in a straight way. These results thus further demonstrate the effectiveness of the proposed Transformer-based model in predicting cloud workloads, significantly improving methods available at this time while providing a reliable

foundation for the enhancement of resource management and operational efficiency within cloud computing environments. The ANOVA is further used to validate these findings to prove credibility, thus assuring statistical differences in performance so that it is not just fortuitous variation across different scenarios.
Tables 2 through 7 show the experimental results for the proposed Transformer-based model in comparison with methods [4], [9], and [15] over large sets of real contextual datasets and several workload metrics. The results obtain the proposed model with lower MAE and RMSE, thus showing its accuracy and reliability compared to others. High-traffic scenarios: By looking at Table 2, one can notice that the proposed model has enormous capability in handling high fluctuations, which is very important for maintaining performance during peak periods. According to Table 3, in cases of traffic with low fluctuation, assurance of precision for resource usage is given by the model without overspending due to over-provisioning. Mixed-traffic scenarios: As shown in Table 4, one can confidently state the strong reliability assurance of the proposed model under different workloads.

5. Conclusion and Future Scopes

This research focused on gaining refined accuracy and efficiency for workload prediction in the cloud environment through designing and evaluating various neural network-based architecture models. In this regard, different proposed models, including LSTM-based, CNN-LSTM Hybrid, and Transformer-based ones, were systematically designed, trained, and tested using historical workload metrics that include CPU usage, memory usage, and network traffic. Results from the experiments prove that our proposed Transformer-based model outperforms others on all contextual datasets. In this high-traffic scenario, the Transformer-based model has yielded an MAE of 3.5% and an RMSE of 5.0%, way ahead in performance compared with methods [4], [9], and [15], which obtained an MAE equal to 5.2%, 4.8%, and 4.0%, respectively. For low-traffic periods, the proposed model had an MAE of 2.8% and an RMSE of 4.2%, while other methods end up with higher error rates. Further validation of the robustness of the proposed model will be provided through mixed-traffic conditions, returning an MAE of 3.2% and an RMSE of 4.6%. It has always come up with the lowest error rates in detailed analyses across workload metrics: an MAE of 3.1% and an RMSE of 4.4% for the predictions of CPU usage, an MAE of 3.4% and an RMSE of 4.7% for that of memory usage, and an MAE of 3.6% and an RMSE of 5.0% pertaining to network traffic. These results indicate that the proposed model based on the Transformer significantly surpassed the others in workload prediction accuracy, thus helping to realize efficient resource management and reduce the operational cost of cloud computing. The hybrid CNN-LSTM model also showed commendable performance, with a test MAE of 4.0% and a test RMSE of 6.0%, in cases where functions of local pattern detection and modeling temporal dependency are combined for different use case scenarios.

Future Scopes

These promising results of this study naturally open up several avenues for future research and development. First, more extensive optimization of the Transformer-based model could be done in the future, such as further fine-tuning of hyperparameters and zoekt into alternative attention mechanisms for increasing predictive accuracy and computational efficiency. Real-time feedback mechanisms can be researched, incorporated, and applied to make the model adapt in a timely manner to new workload patterns for improved responsiveness and accuracy in live environments. Other areas include expanding the scope of the dataset to cover a more diverse set of workload metrics and longer historical time periods to provide more insight regarding model performance. This can be further enhanced by the inclusion of more features, such as application-specific metrics, user behavior patterns, and other environmental factors. Furthermore, one can explore the potential of combining models, including Transformers-based architectures with other advanced architectures like Graph Neural Networks or Reinforcement Learning. Hybrid models putting together methods in a very strong way could be built to further improve workload prediction accuracy and adaptability levels.

For real-world deployment in cloud scenarios, this will involve running the proposed models to estimate their eventual impacts on resource management, cost reduction, and service quality. Such collaborations with cloud service providers can be arranged for large-scale implementation and validation to guarantee the robustness and scalability of the models in various operational contexts. Finally, one needs to consider issues of ethics and privacy when using advanced predictive models in cloud environments. Ensuring user trust in data privacy by using advanced analytics to ensure data privacy and compliance with the regulatory frameworks will be very important to attain sustainable achievements in cloud computing technologies. Conclusively, this research provides a solid foundation toward the advancement of workload prediction in cloud environments due to improved predictive accuracy and operational efficiency. Early proposed models, especially the Transformer-based model, hold huge potential for future enhancements and pragmatic applications. This changes the way the path to possible future strategies for more intelligent and adaptive cloud resource management could be for different scenarios.

6. References

1. L. Ruan et al., "Cloud Workload Turning Points Prediction via Cloud Feature-Enhanced Deep Learning," in IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 1719-1732, 1 April-June 2023, doi: 10.1109/TCC.2022.3160228.
2. I. K. Kim, W. Wang, Y. Qi and M. Humphrey, "Forecasting Cloud Application Workloads With Cloud Insight for Predictive Resource Management," in IEEE Transactions on Cloud Computing, vol. 10, no. 3, pp. 1848-1863, 1 July-Sept. 2022, doi: 10.1109/TCC.2020.2998017.
3. Z. Amekraz and M. Y. Hadi, "CANFIS: A Chaos Adaptive Neural Fuzzy Inference System for Workload Prediction in the Cloud," in IEEE Access, vol. 10, pp. 49808-49828, 2022, doi: 10.1109/ACCESS.2022.3174061.
4. K. Seshadri, K. Sindhu, S. N. Bhattu and C. Kollengode, "Design and Evaluation of a Hierarchical Characterization and Adaptive Prediction Model for Cloud Workloads," in IEEE Transactions on Cloud Computing, vol. 12, no. 2, pp. 712-724, April-June 2024, doi: 10.1109/TCC.2024.3393114.
5. B. Feng, Z. Ding and C. Jiang, "FAST: A Forecasting Model With Adaptive Sliding Window and Time Locality Integration for Dynamic Cloud Workloads," in IEEE Transactions on Services Computing, vol. 16, no. 2, pp. 1184-1197, 1 March-April 2023, doi: 10.1109/TSC.2022.3156619.
6. Z. Ding, B. Feng and C. Jiang, "COIN: A Container Workload Prediction Model Focusing on Common and Individual Changes in Workloads," in IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 12, pp. 4738-4751, 1 Dec. 2022, doi: 10.1109/TPDS.2022.3202833.
7. A. K. Singh, D. Saxena, J. Kumar and V. Gupta, "A Quantum Approach Towards the Adaptive Prediction of Cloud Workloads," in IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 12, pp. 2893-2905, 1 Dec. 2021, doi: 10.1109/TPDS.2021.3079341.
8. Y. -M. Kim, S. Song, B. -M. Koo, J. Son, Y. Lee and J. -G. Baek, "Enhancing Long-Term Cloud Workload Forecasting Framework: Anomaly Handling and Ensemble Learning in Multivariate Time Series," in IEEE Transactions on Cloud Computing, vol. 12, no. 2, pp. 789-799, April-June 2024, doi: 10.1109/TCC.2024.3400859.

9. X. Chen, F. Zhu, Z. Chen, G. Min, X. Zheng and C. Rong, "Resource Allocation for Cloud-Based Software Services Using Prediction-Enabled Feedback Control With Reinforcement Learning," in IEEE Transactions on Cloud Computing, vol. 10, no. 2, pp. 1117-1129, 1 April-June 2022, doi: 10.1109/TCC.2020.2992537.
10. N. I. Mahbub, M. D. Hossain, S. Akhter, M. I. Hossain, K. Jeong and E. -N. Huh, "Robustness of Workload Forecasting Models in Cloud Data Centers: A White-Box Adversarial Attack Perspective," in IEEE Access, vol. 12, pp. 55248-55263, 2024, doi: 10.1109/ACCESS.2024.3385863.
11. D. Saxena, J. Kumar, A. K. Singh and S. Schmid, "Performance Analysis of Machine Learning Centered Workload Prediction Models for Cloud," in IEEE Transactions on Parallel and Distributed Systems, vol. 34, no. 4, pp. 1313-1330, April 2023, doi: 10.1109/TPDS.2023.3240567.
12. N. Hogade and S. Pasricha, "A Survey on Machine Learning for Geo-Distributed Cloud Data Center Management," in IEEE Transactions on Sustainable Computing, vol. 8, no. 1, pp. 15-31, 1 Jan.-March 2023, doi: 10.1109/TSUSC.2022.3208781.
13. X. Chen, L. Yang, Z. Chen, G. Min, X. Zheng and C. Rong, "Resource Allocation With Workload-Time Windows for Cloud-Based Software Services: A Deep Reinforcement Learning Approach," in IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 1871-1885, 1 April-June 2023, doi: 10.1109/TCC.2022.3169157.
14. J. Li, J. Yao, D. Xiao, D. Yang and W. Wu, "EvoGWP: Predicting Long-Term Changes in Cloud Workloads Using Deep Graph-Evolution Learning," in IEEE Transactions on Parallel and Distributed Systems, vol. 35, no. 3, pp. 499-516, March 2024, doi: 10.1109/TPDS.2024.3357715.
15. D. Alqahtani, "Leveraging Sparse Auto-Encoding and Dynamic Learning Rate for Efficient Cloud Workloads Prediction," in IEEE Access, vol. 11, pp. 64586-64599, 2023, doi: 10.1109/ACCESS.2023.3289884.
16. M. Kumar, A. Kishor, J. K. Samariya and A. Y. Zomaya, "An Autonomic Workload Prediction and Resource Allocation Framework for Fog-Enabled Industrial IoT," in IEEE Internet of Things Journal, vol. 10, no. 11, pp. 9513-9522, 1 June1, 2023, doi: 10.1109/JIOT.2023.3235107.
17. J. Bi, H. Ma, H. Yuan and J. Zhang, "Accurate Prediction of Workloads and Resources With Multi-Head Attention and Hybrid LSTM for Cloud Data Centers," in IEEE Transactions on Sustainable Computing, vol. 8, no. 3, pp. 375-384, 1 July-Sept. 2023, doi: 10.1109/TSUSC.2023.3259522.
18. L. Zhang et al., "A Novel Hybrid Model for Docker Container Workload Prediction," in IEEE Transactions on Network and Service Management, vol. 20, no. 3, pp. 2726-2743, Sept. 2023, doi: 10.1109/TNSM.2023.3248803.
19. J. Bi, H. Yuan, S. Li, K. Zhang, J. Zhang and M. Zhou, "ARIMA-Based and Multiapplication Workload Prediction With Wavelet Decomposition and Savitzky–Golay Filter in Clouds," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 54, no. 4, pp. 2495-2506, April 2024, doi: 10.1109/TSMC.2023.3343925.
20. M. E. Karim, M. M. S. Maswood, S. Das and A. G. Alharbi, "BHyPreC: A Novel Bi-LSTM Based Hybrid Recurrent Neural Network Model to Predict the CPU Workload of Cloud Virtual Machine," in IEEE Access, vol. 9, pp. 131476-131495, 2021, doi: 10.1109/ACCESS.2021.3113714.
21. M. A. Razzaq, J. A. Mahar, M. Ahmad, N. Saher, A. Mehmood and G. S. Choi, "Hybrid Auto-Scaled Service-Cloud-Based Predictive Workload Modeling and Analysis for Smart Campus System," in IEEE Access, vol. 9, pp. 42081-42089, 2021, doi: 10.1109/ACCESS.2021.3065597.
22. T. W. Gyeera, A. J. H. Simons and M. Stannett, "Kalman Filter Based Prediction and Forecasting of Cloud Server KPIs," in IEEE Transactions on Services Computing, vol. 16, no. 4, pp. 2742-2754, 1 July-Aug. 2023, doi: 10.1109/TSC.2022.3217148.

23. R. Pinciroli, A. Ali, F. Yan and E. Smirni, "CEDULE+: Resource Management for Burstable Cloud Instances Using Predictive Analytics," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 945-957, March 2021, doi: 10.1109/TNSM.2020.3039942.
24. M. Sohani and S. C. Jain, "A Predictive Priority-Based Dynamic Resource Provisioning Scheme With Load Balancing in Heterogeneous Cloud Computing," in IEEE Access, vol. 9, pp. 62653-62664, 2021, doi: 10.1109/ACCESS.2021.3074833.
25. R. Shi and C. Jiang, "Three-Way Ensemble Prediction for Workload in the Data Center," in IEEE Access, vol. 10, pp. 10021-10030, 2022, doi: 10.1109/ACCESS.2022.3145426.