Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A Data-Driven Neural Network Approach to Predict Cloud Throughput Variations

Dr. K. Subbarao
Professor & HOD

Shaik Imran, Keerthi Narendra Babu, Chennuru Siddu Sohith
Student
Department of CSE – Data Science
St. Ann's College of Engineering & Technology, Chirala, Andhra Pradesh, India.

Abstract **— Cloud computing environments experience frequent and dynamic variations in network throughput due to factors such as CPU utilization, disk I/O, network traffic, and request latency. Traditional statistical models fail to capture the complex, non-linear relationships among these parameters. This paper presents a data-driven system that employs deep learning models — Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN) — to predict short-term variations in cloud network throughput using multivariate time series data. The system analyzes five key performance metrics: network traffic, CPU utilization, disk write speed, bytes downloaded, and request latency, structured into sequences of nine time steps. Among the implemented models, LSTM achieves the best prediction accuracy with a Mean Absolute Percentage Error (MAPE) of 3.93%, followed by RNN at 4.68% and CNN at 6.70%. An interactive web application developed using Streamlit enables real-time data upload, visualization, and prediction generation. The proposed system demonstrates the effectiveness of deep learning in capturing temporal dependencies in cloud performance data, supporting proactive decision-making and efficient resource management.**

Keywords — cloud computing; throughput prediction; deep learning; LSTM; RNN; CNN; multivariate time series; Streamlit

I. INTRODUCTION

Cloud computing has emerged as a foundational technology enabling users to store, process, and transfer data over distributed systems. Platforms such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud are widely adopted, making efficient network performance a critical factor for system reliability and user satisfaction [1].

One of the foremost challenges in cloud environments is the variability of network throughput. Factors such as CPU utilization, disk operations, and network traffic interact in complex, non-linear ways that traditional statistical approaches fail to model adequately [5]. These conventional methods — including ARIMA and linear regression — typically analyze a single variable in isolation, rendering them unsuitable for the dynamic, high-dimensional nature of cloud performance data [7].

To address these shortcomings, this paper presents a multivariate, deep learning-based system for predicting short-term cloud network throughput variations. The system employs three deep learning architectures — RNN, LSTM, and CNN — trained on time series data comprising five cloud performance metrics. A Streamlit-based web application provides an accessible interface for data upload, visualization, and real-time prediction, bridging the gap between backend model processing and end-user interaction.

The contributions of this work are as follows: (i) a multivariate time series prediction framework utilizing RNN, LSTM, and CNN models; (ii) a quantitative comparison of model performance using MAPE as the evaluation metric; and (iii) a user-friendly, deployable web application for practical cloud performance monitoring.

II. LITERATURE SURVEY

Early research on cloud network performance prediction relied heavily on statistical methods. Mathis et al. [1] modelled TCP throughput using macroscopic analysis, while Hacker et al. [2] investigated the effects of parallel TCP sockets on wide-area networks. Lu et al. [3] extended this line of work by modelling and taming parallel TCP behaviour. These statistical approaches, although foundational, are inherently limited to linear relationships and univariate data.

Machine learning approaches introduced greater flexibility. Mirza et al. [5] proposed a machine learning framework for TCP throughput prediction and demonstrated improvements over statistical baselines. Papagiannaki et al. [6] performed long-term forecasting of Internet backbone traffic, while Zhou et al. [7] applied ARIMA and GARCH models for network traffic modelling. Cortez et al. [8] explored neural network-based Internet traffic forecasting, establishing early benchmarks for learning-based approaches.

More recent research has focused on deep learning for time series prediction. Trinh et al. [9] applied LSTM networks to mobile traffic prediction from raw data, reporting improved accuracy over traditional methods. Mozo et al. [10] employed CNNs for short-term data centre network traffic load forecasting, highlighting the effectiveness of convolutional feature extraction. Phanekham et al. [11] specifically investigated neural network-based prediction of cloud network infrastructure throughput. Yildirim and Akon [12], whose work serves as the primary reference for this project, demonstrated that multivariate time series prediction using deep

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

learning significantly outperforms univariate statistical approaches for cloud data transfer throughput.

The theoretical foundations of the models used in this work are drawn from Hochreiter and Schmidhuber [13] for LSTM, LeCun et al. [14] for deep learning, and Goodfellow et al. [15] for comprehensive deep learning methodology. The present system extends these prior efforts by integrating three complementary deep learning architectures within a unified, interactive prediction platform tailored for cloud environments.

TABLE I

Comparison of Existing Approaches

Method	Type	Features	Advantages	Limitations
ARIMA	Statistical	Univariate	Simple, easy to implement	Cannot handle non-linear data
Linear Reg.	Statistical	Limited	Fast computation	Low accuracy for complex data
SVM	ML	Limited	Better than statistical	Requires manual feature selection
RNN	Deep Learning	Multivariate	Handles sequential data	Poor long-term dependency
LSTM	Deep Learning	Multivariate	High accuracy, long-term data	Slightly slower training
CNN	Deep Learning	Multivariate	Fast, good feature extraction	Limited long-term modeling

III. PROPOSED METHODOLOGY

The proposed system follows a structured pipeline comprising data collection, preprocessing, feature selection, model development, and deployment. Each stage is designed to ensure data quality and model accuracy.

A. Data Collection

The dataset is sourced from cloud environments and consists of multivariate time series records capturing five cloud performance metrics at regular time intervals: Network Traffic (Mbps), CPU Utilization (%), Disk Write Speed (Mbps), Bytes Downloaded (MB), and Request Latency (ms). Table II summarises the input features. The structured multivariate format enables the system to analyse inter-parameter relationships and learn temporal patterns effectively.

TABLE II

Description of Input Features

Feature Name	Description	Unit
Network Traffic	Amount of data transferred over the network	Mbps
CPU Utilization	Percentage of CPU usage in the system	%
Disk Write Speed	Speed of writing data to disk	Mbps
Bytes Downloaded	Total data downloaded from the source	MB
Request Latency	Time taken to process a request	ms

B. Data Preprocessing

Raw data undergoes a multi-step preprocessing pipeline. Missing values are identified and handled to ensure completeness. All feature values are normalized to a common scale to prevent dominance by high-magnitude variables and to accelerate model convergence. The cleaned data is then structured into fixed-length sequences: each input sample consists of 9 time steps across 5 features, yielding an input tensor of shape (9, 5). This sequential structure is essential for enabling deep learning models to learn temporal patterns. Output labels correspond to the predicted values at the subsequent time step.

C. Feature Selection

Five features are selected based on their established influence on cloud network performance. Network Traffic directly reflects bandwidth demand. CPU Utilization indicates processing load that can cause bottlenecks. Disk Write Speed captures storage I/O constraints. Bytes Downloaded quantifies incoming data volume. Request Latency measures end-to-end processing delay. Together, these features provide a comprehensive representation of cloud system behaviour, enabling the models to learn complex interdependencies.

D. Model Development

Three deep learning architectures are developed and compared for throughput prediction.

1) Recurrent Neural Network (RNN): The RNN processes input sequences step-by-step, maintaining a hidden state that accumulates information from previous time steps. It is effective for capturing short-term sequential patterns. However, due to the vanishing gradient problem, RNN exhibits limited capacity for retaining long-term dependencies [13].

2) Long Short-Term Memory (LSTM): LSTM is an advanced variant of RNN that incorporates memory cells with gating mechanisms — input, forget, and output gates — to selectively retain or discard information across long sequences [13]. This architecture overcomes the vanishing gradient limitation and is particularly suited for time series data exhibiting long-range dependencies. LSTM achieves the highest prediction accuracy in this system.

3) Convolutional Neural Network (CNN): The CNN applies one-dimensional convolutional operations to the input time series, extracting local patterns and short-term trends through learned filters. It benefits from parallel processing, resulting in faster computation relative to recurrent architectures. While effective for feature extraction, CNN is comparatively limited in modelling long-term temporal dependencies.

All models are trained using Mean Squared Error (MSE) as the loss function and the Adam optimizer for adaptive gradient-based parameter updates. Training is conducted over multiple epochs until convergence. Performance is evaluated on held-out test data using MAPE, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

IV. SYSTEM DESIGN

A. System Architecture

The system is organized into five functional modules that interact sequentially to deliver end-to-end prediction capability.

The Data Input Module accepts user-uploaded datasets in JSON or CSV format through the web interface. The Data Preprocessing Module applies normalization and sequence construction to produce model-ready input tensors of shape (9, 5). The Model Training Module builds and trains the RNN, LSTM, and CNN architectures using historical cloud performance data. The Prediction Module accepts a nine-step input sequence and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

generates one-step-ahead predictions for all five features. The Visualization Module renders prediction outputs alongside historical data in graphical and tabular formats using Streamlit.

TABLE III

System Architecture Components

Component	Description
Data Input Module	Accepts dataset from user (JSON/CSV)
Preprocessing Module	Cleans and normalizes data into sequences
Model Module	Loads trained models (RNN, LSTM, CNN)
Prediction Module	Generates next time-step predictions
Visualization Module	Displays graphs and tabular results

B. Database Design

The system employs file-based storage using JSON and CSV formats rather than a conventional relational database. The dataset is structured in a tabular format where each row represents a time-stamped record and each column corresponds to one of the five performance features. During processing, the dataset is loaded into a Pandas DataFrame, which serves as an in-memory data management layer. This approach simplifies integration with Python-based data science libraries and supports dynamic dataset uploads without structural modification. Input data is validated prior to preprocessing to ensure completeness and consistency.

C. Model Configuration

The deep learning models are configured with consistent hyperparameters to enable fair comparison. The input shape is (9, 5) corresponding to nine time steps and five features. Training employs the MSE loss function and the Adam optimizer. Activation functions include ReLU for hidden layers and a linear or sigmoid function at the output layer. Model files are persisted in HDF5 format (.h5) and loaded at inference time by the web application.

TABLE IV

Model Configuration Parameters

Parameter	Description
Input Shape	(9, 5) — 9 time steps, 5 features
Batch Size	Number of samples per training batch
Loss Function	Mean Squared Error (MSE)
Optimizer	Adam
Activation Function	ReLU (hidden layers) / Sigmoid (output)
Model Format	HDF5 (.h5)

V. IMPLEMENTATION

A. Tools and Technologies

The system is implemented in Python, leveraging a set of well-established libraries. TensorFlow and Keras are used for designing, training, and deploying the deep learning models. Pandas handles dataset loading, manipulation, and conversion into DataFrame structures. NumPy provides efficient multi-dimensional array operations necessary for reshaping input tensors. Matplotlib generates graphical visualizations of historical and predicted performance metrics. Streamlit is used to develop the interactive web-based user interface, enabling real-time data interaction without requiring specialized frontend development expertise.

B. Application Workflow

The Streamlit application follows a structured workflow. Users begin by uploading a dataset in JSON or CSV format, after which the system displays a preview and summary statistics. The preprocessing pipeline normalizes the data and constructs input sequences. Users then select a model (RNN, LSTM, or CNN) from a sidebar menu and initiate prediction. The system loads the corresponding pre-trained model file, reshapes the input tensor to (1, 9, 5), and generates predictions for the five performance metrics. Outputs are displayed via metric cards, line graphs comparing historical and predicted values, and a downloadable results table in CSV format. The application also incorporates error handling for invalid file formats, missing data, and model loading failures.

VI. RESULTS AND DISCUSSION

The three deep learning models — RNN, LSTM, and CNN — are evaluated on the test subset of the cloud performance dataset. Prediction accuracy is measured using MAPE, where a lower value indicates higher accuracy. Table V presents the MAPE results for each model.

TABLE V

Model Performance Comparison (MAPE %)

Model	MAPE (%)	Rank
LSTM	3.93	1 (Best)
RNN	4.68	2
CNN	6.70	3

The LSTM model achieves the lowest MAPE of 3.93%, confirming its superior ability to model long-term temporal dependencies in multivariate cloud performance data. The gating mechanisms within LSTM enable selective retention of relevant historical information, which is particularly beneficial in cloud environments where performance patterns exhibit extended temporal correlations.

The RNN model attains a MAPE of 4.68%, demonstrating competent short-term prediction capability. However, its susceptibility to vanishing gradients limits its effectiveness for longer-range dependencies, resulting in a higher error compared to LSTM. The CNN model records a MAPE of 6.70%. While the CNN excels at extracting local structural patterns through convolutional filters and benefits from parallel computation, its inherently limited receptive field reduces its effectiveness for capturing long-range temporal dependencies, a limitation consistent with findings in related literature [10].

Visualization outputs generated by the Streamlit application include CPU utilization trend graphs, network traffic patterns, and model prediction charts overlaid on historical data for all three models. These visualizations enable users to intuitively assess prediction quality and system behaviour. The model comparison graph provides a clear summary of performance differences, facilitating informed model selection for deployment.

Test case evaluation demonstrates that all primary system functions operate correctly under defined conditions, including dataset upload, data validation, preprocessing, input shape verification, model selection and loading, prediction generation,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

result visualization, and output download. All ten defined test cases achieve a passing status, confirming the functional correctness of the system.

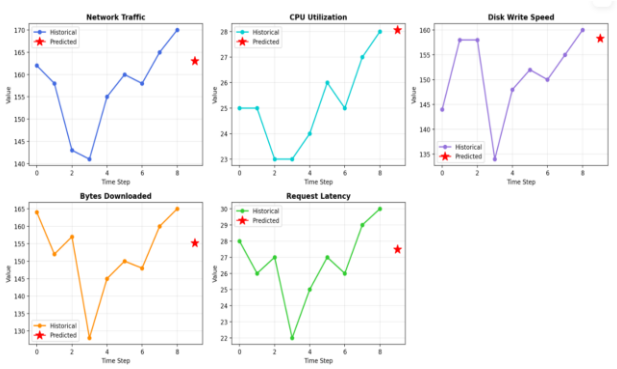

Figure 1. showing Prediction Output and Recommendation

VII. CONCLUSION

This paper has presented a data-driven deep learning system for predicting short-term variations in cloud network throughput using multivariate time series data. Three deep learning architectures — RNN, LSTM, and CNN — are implemented, trained, and evaluated on cloud performance datasets comprising five key metrics: network traffic, CPU utilization, disk write speed, bytes downloaded, and request latency.

Experimental results demonstrate that the LSTM model achieves the best prediction accuracy with a MAPE of 3.93%, outperforming RNN (4.68%) and CNN (6.70%). These results validate the hypothesis that capturing long-term temporal dependencies through LSTM gating mechanisms is critical for accurate cloud throughput prediction. The multivariate approach adopted in this work provides a more comprehensive representation of cloud system behaviour compared to univariate statistical methods.

The Streamlit-based web application delivers a practical, deployable solution for real-time cloud performance monitoring, enabling non-technical users to upload data, visualize trends, and generate predictions without specialized knowledge. The system thereby bridges the gap between advanced deep learning research and operational utility in cloud management.

Future work will explore Transformer-based architectures for enhanced long-range dependency modelling, integration with real-time cloud monitoring APIs (AWS CloudWatch, Azure Monitor), extension to multi-step forecasting horizons, and deployment on cloud infrastructure for scalable access.

ACKNOWLEDGMENT

The authors express sincere gratitude to Dr. K. Subbarao, Head of the Department of CSE–Data Science, St. Ann's College of Engineering & Technology, Chirala, for his invaluable guidance and support throughout this project. The authors also acknowledge the teaching and non-teaching staff of the department, and the management of the institution, for providing the necessary infrastructure and resources.

REFERENCES

[1] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behaviour of the TCP congestion avoidance algorithm," ACM SIGCOMM Comput. Commun. Rev., vol. 27, no. 3, pp. 67–82, Jul. 1997.

[2] T. J. Hacker, B. D. Noble, and B. D. Atley, "The end-to-end performance effects of parallel TCP sockets on a lossy wide area network," in Proc. IEEE Int. Symp. Parallel Distrib. Process. (IPDPS), Apr. 2002, pp. 434–443.

[3] D. Lu, Y. Qiao, P. A. Dinda, and F. E. Bustamante, "Modelling and taming parallel TCP on the wide area network," in Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS), Apr. 2005, p. 10.

[4] P. Millan, C. Aliaga, C. Molina, E. Dimogerontakis, and R. Meseguer, "Time series analysis to predict end-to-end quality of wireless community networks," Electronics, vol. 8, no. 5, p. 578, May 2019.

[5] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to TCP throughput prediction," IEEE/ACM Trans. Netw., vol. 18, no. 4, pp. 1026–1039, Aug. 2010.

[6] K. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-term forecasting of Internet backbone traffic: Observations and initial models," in Proc. IEEE INFOCOM, Mar./Apr. 2003, pp. 1178–1188.

[7] B. Zhou, D. He, Z. Sun, and W. H. Ng, "Network traffic modelling and prediction with ARIMA/GARCH," in Proc. HET-NETs Conf., Jul. 2005, pp. 1–10.

[8] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Internet traffic forecasting using neural networks," in Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2006, pp. 2635–2642.

[9] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using LSTM networks," in Proc. IEEE 29th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC), Sep. 2018, pp. 1827–1832.

[10] A. Mozo, B. Ordozgoiti, and S. Gomez-Canaval, "Forecasting short-term data centre network traffic load with convolutional neural networks," PLoS ONE, vol. 13, no. 2, Feb. 2018, Art. no. e0191939.

[11] D. Phanekham, S. Nair, N. Rao, and M. Truty, "Predicting throughput of cloud network infrastructure using neural networks," in Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS), May 2021, pp. 1–6.

[12] E. Yildirim and A. Akon, "Predicting short-term variations in end-to-end cloud data transfer throughput using neural networks," IEEE Access, vol. 11, pp. 78656–78670, 2023.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015.

[15] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.