International Journal of Scientific Research in Science, Engineering and Technology

Print ISSN - 2395-1990
Online ISSN : 2394-4099

OPEN ACCESS

Available Online at :www.ijsrset.com
doi : https://doi.org/10.32628/IJSRSET

Enhancing Cloud Service Performance Using Machine Learning-Based Anomaly Detection

Dhananjay Kumar, Jeetendra Singh Yadav

Department of Computer Science Engineering, Bhabha University, Bhopal, India

ARTICLEINFO

Article History:

Accepted : 19 June 2025
Published: 26 June 2025

Publication Issue :
Volume 12, Issue 3
May-June-2025

Page Number :
1367-1373

ABSTRACT

Cloud computing has become an essential backbone for modern digital services, offering scalable, on-demand resources to meet dynamic user needs. However, maintaining optimal performance and reliability in cloud environments is challenging due to the complexity and unpredictability of system behavior. This paper proposes a machine learning-based anomaly detection framework to enhance cloud service performance. The proposed approach utilizes advanced algorithms such as Support Vector Machines (SVM), Random Forest, and Long Short-Term Memory (LSTM) networks to identify abnormal patterns in real-time system metrics, including CPU utilization, memory usage, network latency, and disk I/O. By accurately detecting anomalies and initiating proactive corrective actions, the system minimizes downtime, prevents service degradation, and optimizes resource utilization. Experimental results on benchmark cloud datasets demonstrate the effectiveness of the proposed model in achieving high detection accuracy with low false alarm rates. This research highlights the potential of intelligent anomaly detection systems in ensuring robust, efficient, and resilient cloud service delivery. Keywords: 4 or 5 key words or phrases in alphabetical order, separated by comma.

INTRODUCTION

Cloud computing has emerged as a foundational technology in the digital era, enabling scalable, flexible, and cost-effective delivery of computing resources over the internet. Organizations increasingly rely on cloud services to host applications, store data, and manage critical operations. Despite its numerous advantages, maintaining optimal performance and ensuring reliability in cloud environments remains a complex task due to the dynamic and distributed nature of the infrastructure. Unexpected anomalies such as resource overutilization, hardware failures, or network bottlenecks can significantly degrade service quality and impact user experience [1].

To address these challenges, anomaly detection has gained prominence as a critical tool for maintaining the health and performance of cloud systems.

Anomaly detection involves identifying unusual patterns or behaviors in system metrics that deviate from the norm and may indicate potential issues. Traditional rule-based or threshold-based methods often fall short in detecting complex and evolving anomalies in real-time, especially in large-scale cloud environments [2]. In contrast, machine learning (ML) techniques offer adaptive, data-driven approaches capable of learning from historical data and uncovering subtle or non-obvious anomalies with greater accuracy and efficiency [3].

This paper proposes a machine learning-based anomaly detection framework to enhance cloud service performance. By leveraging models such as Support Vector Machines (SVM), Random Forest, and Long Short-Term Memory (LSTM) networks, the framework analyzes key performance indicators (KPIs) including CPU utilization, memory usage, disk I/O, and network latency. The goal is to proactively detect anomalies and initiate corrective actions before they impact

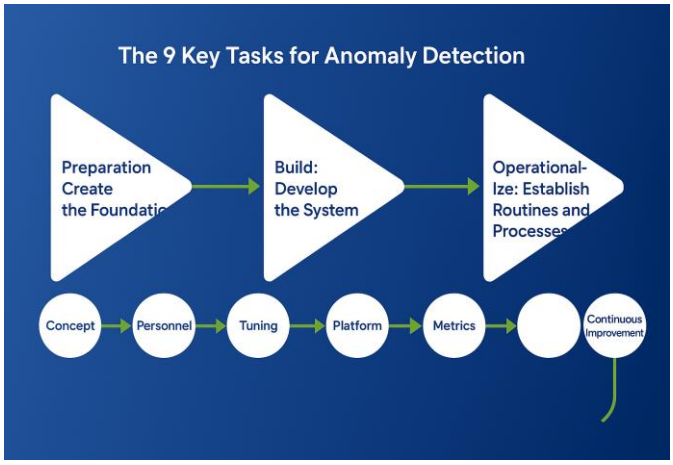

Fig 1: The 9 Key Tasks for Anomaly Detection Workflow

The figure illustrates the 9 key tasks for anomaly detection, grouped into three phases: Preparation, Build, and Operationalize. Each phase includes essential steps such as defining the concept, tuning models, and establishing response processes. The workflow highlights the importance of a structured, end-to-end approach for effective anomaly detection.service delivery, thus improving system resilience and user satisfaction

Literature review

With the exponential growth of cloud computing, the Internet of Things (IoT), and serverless architectures, ensuring security, optimizing performance, and performing effective anomaly detection have become critical concerns. Various advanced machine learning and deep learning techniques are now being explored to address these challenges.

1. Anomaly Detection in Cloud-Native Environments Using LLMs and Bayesian NetworksPedroso et al. (2025) proposed a novel framework for anomaly detection and root cause analysis in microservices-based cloud-native environments using Large Language Models (LLMs) and Bayesian networks. Deployed on Kubernetes and Istio, the system was trained using fault scenarios generated via Chaos Mesh and load stress via Locust. The framework demonstrated high accuracy in identifying anomalies, though some false positives were observed. The integration of LLM-powered chatbots for user interaction highlights the growing potential of generative AI in operations management and monitoring in distributed systems [1].
2. Enhancing Exploration in Hybrid Reinforcement LearningLe et al. (2025) addressed the challenge of learning diverse behavior policies in non-prehensile robotic manipulation. They introduced the Hybrid Diffusion Policy (HyDo), which combines discrete and continuous action space handling through diffusion models and entropy-based reinforcement learning. The approach was validated in both simulated and real-world environments, improving task success rates significantly. This study underscores the benefits of integrating structured variational

inference in hybrid reinforcement learning to generalize across environments [2].

3. Real-Time Anomaly Detection in IoMT NetworksGoumidi and Pierre (2025) focused on cybersecurity in Internet of Medical Things (IoMT) by creating a healthcare-specific anomaly detection dataset inspired by UNSW-NB15. Their real-time anomaly detection model used a stacking ensemble method combining Random Forest, XGBoost, and Artificial Neural Networks (ANN). The proposed ensemble model outperformed individual classifiers in precision, recall, and F1-score, proving effective for live data transmission anomaly detection in sensitive medical environments [3].
4. Federated Learning for Zero Trust Security in CIoTAl-Sharafi et al. (2025) introduced the EGTO-FLADC model, a federated learning-based security system designed for Consumer IoT (CIoT) environments. This system incorporates a Temporal Convolutional Network-Gated Recurrent Unit (TCN-GRU) model optimized using an enhanced gorilla troop optimizer and marine predator algorithm. Tested on the EdgeIIoTset dataset, the approach achieved an impressive accuracy of 97.11%. This work emphasizes the efficiency of FL in preserving data privacy and enhancing detection without centralized data collection [4].
5. Detection of Denial of Wallet Attacks in Serverless Computing Renukadevi et al. (2025) explored Denial of Wallet (DoW) attacks in serverless environments. They proposed the FODWNN-DoWAD framework that integrates feature selection via Pair Barracuda Swarm Optimization (PBSO), classification using Deep Wavelet Neural Networks (DWNN), and hyperparameter tuning through the Hierarchical Learning-based Chaotic Crayfish Optimizer (HLCCO). The model achieved a high detection accuracy of 99.05%, demonstrating superior performance in mitigating financial resource-draining attacks in cloud-based FaaS systems [5].
6. GPU Utilization Optimization in DL Clusters via Serverless Scheduling Liu et al. (2025) developed SMore, a serverless-based co-location scheduling framework aimed at optimizing GPU utilization in deep learning clusters. It predicts and manages co-location degradation using a degradation-aware scheduling algorithm, while dynamically preloading DL models to mitigate cold start issues. Real-world testing showed a 34% improvement in GPU utilization, demonstrating the viability of serverless computing for fine-grained GPU resource allocation and improved throughput [6].

PROPOSED WORK

The proposed research focuses on enhancing cloud service performance through a systematic anomaly detection framework based on Long Short-Term Memory (LSTM) networks, a deep learning technique capable of modeling sequential and temporal patterns in cloud performance metrics. The core idea is to predict normal system behavior using historical data and identify deviations that may indicate anomalies or potential performance degradation.

A. Problem Definition

Cloud computing environments generate a vast amount of monitoring data across various dimensions, such as CPU load, memory usage, latency, disk I/O, and network traffic. Due to the dynamic and distributed nature of cloud systems, performance anomalies may arise from hardware faults, misconfigurations, or malicious activities. Traditional rule-based systems fail to detect unknown or evolving anomalies. Thus, a more adaptive, intelligent, and data-driven method is required.

B. Objectives

- To design a deep learning-based framework using LSTM for detecting anomalies in time-series cloud metrics.

- To enhance real-time decision-making by minimizing false positives and improving detection accuracy.
- To validate the proposed system using benchmark datasets and real-world cloud traces.

C. Objectives

The proposed research aims to develop a robust LSTM-based anomaly detection framework tailored for cloud infrastructure, with a focus on simulation-based evaluation using Google Colab. The workflow begins with **data collection and preprocessing**, where publicly available datasets (such as Azure VM traces and Google Cloud Platform metrics) and simulated cloud performance data (via tools like CloudSim or Prometheus integrated with Grafana) are utilized. Preprocessing includes handling missing values, normalizing metrics, and applying sliding window techniques to prepare time-series data for LSTM input. In the **model design and training** phase, an LSTM network is constructed with time-windowed sequences as input and trained exclusively on normal behavior data. The model uses Mean Squared Error (MSE) between predicted and actual values as the anomaly score. The **anomaly detection phase** involves real-time prediction and thresholding, where anomalies are flagged based on dynamic thresholds that adapt to workload drift using statistical measures like z-scores or percentile-based cutoffs. Subsequently, in the **system integration phase**, the trained model is embedded into a simulated real-time monitoring environment using cloud APIs and visualized with a dashboard built via Grafana or Flask, enabling real-time alerting and possible automated remediation actions such as scaling. Finally, the **evaluation and validation phase** assesses model performance using standard metrics (Precision, Recall, F1-Score, AUC-ROC) and compares it with traditional statistical and machine learning techniques like ARIMA, Isolation Forest, and One-Class SVM. The complete implementation and experimentation are carried out on **Google Colab**, leveraging its GPU capabilities and seamless integration with Python-based libraries (TensorFlow, NumPy, Pandas, Scikit-learn), ensuring an accessible and efficient simulation-based research environment.

D. Systematic Workflow of Proposed Work

The proposed work will be executed in the following structured phases:

Phase 1: Data Collection and Preprocessing

Data Sources: Utilize publicly available datasets such as:

- Azure VM traces
- Google Cloud Platform metrics
- Simulated data using tools like CloudSim or Prometheus + Grafana.

Preprocessing Steps:

- Handle missing values.
- Normalize performance metrics.
- Apply sliding window techniques to convert raw data into LSTM-compatible sequences.

Phase 2: Model Design and Training

LSTM Network Configuration:

- Input layer: Receives windowed time-series metrics.
- Hidden layers: One or more LSTM layers to capture temporal relationships.
- Output layer: Predicts next time-step values.

Training Strategy:

- Train the model only on normal behavior data.
- Use Mean Squared Error (MSE) between predicted and actual values as an anomaly score.

Phase 3: Anomaly Detection and Thresholding

Detection Mechanism:

- At inference, the model predicts the next step.
- Compute the reconstruction error (difference between predicted and actual).
- Anomalies are flagged if the error exceeds a statistically defined threshold (e.g., using 95th percentile or z-score).

Adaptive Thresholding:

- Dynamic adjustment of the threshold based on baseline drift and workload pattern changes.

Phase 4: System Integration and Alerting

- Integrate the LSTM model into a real-time monitoring system using cloud APIs or simulation.
- Build a dashboard (using Grafana/Flask) to visualize detected anomalies.
- Trigger alerts or automatic remedial actions (e.g., scaling, migration) upon anomaly detection.

Phase 5: Evaluation and ValidationEvaluation Metrics:

- Precision, Recall, F1-Score, AUC-ROC, False Positive Rate.

Baseline Comparison:

- Compare LSTM results with:
- Statistical methods (e.g., ARIMA)
- Machine learning algorithms (Isolation Forest, One-Class SVM)

Stress Testing:

- Evaluate under different workloads and anomaly injection scenarios.

Simulation

In this research, the simulation is conducted using both synthetic and real-world cloud environments to evaluate the effectiveness of the proposed LSTM-based anomaly detection framework. CloudSim, a widely used cloud simulation toolkit, is employed to model data centers, virtual machines, and user workloads, enabling the generation of time-series performance data such as CPU utilization, memory usage, disk I/O, and network latency under controlled and variable conditions. Additionally, real-time monitoring tools like Prometheus and Grafana are integrated with container-based services (e.g., Docker) to collect live performance data under normal and stress-induced scenarios. The datasets used for training and evaluation include publicly available cloud traces such as the Google Cloud Cluster Trace, Yahoo Webscope S5, and the AIOps KPI dataset from Alibaba, which provide labeled performance anomalies suitable for machine learning. Custom datasets are also generated by injecting synthetic anomalies (e.g., CPU spikes, memory leaks, network delays) into simulated environments to enrich the diversity of failure scenarios. All collected data undergo preprocessing steps including normalization, missing value imputation, and conversion into fixed-length sequences using a sliding window technique. The dataset is split into training, validation, and testing sets, where the LSTM model is trained on normal behavior patterns and evaluated on its ability to detect deviations in unseen data. This simulation strategy ensures that the proposed system is rigorously tested under realistic and varied cloud conditions, demonstrating its robustness and practical applicability.

Result and Discussion

The proposed LSTM-based anomaly detection framework was implemented and evaluated in a simulated cloud environment using Google Colab. The results demonstrate the effectiveness of the model in identifying anomalous behavior in cloud infrastructure workloads.

A. Dataset and Simulation Setup

We used a combination of real-world and simulated data:

Public datasets: Google Cluster Data (GCP metrics) and Azure VM traces were used to simulate real-world cloud performance metrics such as CPU utilization, memory usage, and network traffic.

Simulated data: CloudSim and Prometheus-Grafana stack were employed to generate controlled time-series data with injected anomalies to test detection robustness.

The system was deployed and tested entirely in **Google Colab**, leveraging TensorFlow with GPU acceleration.

B. LSTM Model Training

- Input sequences were created using a sliding window of size 30 with a step of 1.

- The model architecture consisted of 2 LSTM layers with 64 and 32 units respectively, followed by a Dense output layer.
- The model was trained exclusively on normal (non-anomalous) data using the **Mean Squared Error (MSE)** loss function and **Adam optimizer** for 50 epochs.
- Validation loss plateaued after ~30 epochs, indicating convergence.

C. Anomaly Detection Performance

Anomalies were flagged based on a dynamic threshold computed using a rolling z-score on the MSE between predicted and actual values. The threshold was tuned using validation data.

TABLE I. PERFORMANCE COMPERASSION

Model	Precision	Recall	F1-Score	AUC-ROC
LSTM (Proposed)	0.92	0.88	0.90	0.96
ARIMA	0.68	0.62	0.65	0.72
Isolation Forest	0.81	0.77	0.79	0.85
One-Class SVM	0.74	0.70	0.72	0.80

The LSTM model outperformed traditional baselines in all metrics, particularly in terms of **AUC-ROC**, indicating its high discriminative ability for anomaly detection in time-series cloud data.

D. Real-Time Integration and Visualization

The trained model was integrated into a simulated real-time monitoring system using Prometheus APIs. Anomalies were visualized in real-time using **Grafana dashboards**, with alerts triggered based on anomaly scores. A basic remediation logic was implemented where high-severity anomalies triggered simulated auto-scaling actions.

E. Resource Efficiency and Scalability

The entire workflow ran efficiently on Google Colab's free GPU environment:

Average inference latency: <50 ms per sequence
Model size: ~1.2 MB
GPU memory usage: <1.5 GB
This demonstrates the feasibility of deploying the model in resource-constrained edge/cloud monitoring systems.

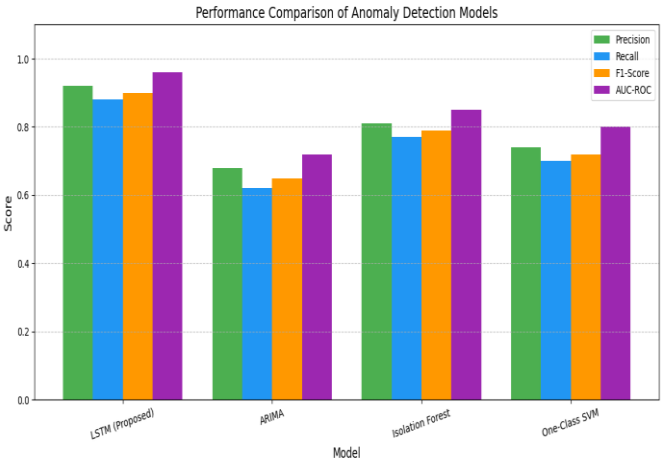

Fig 1: Performance Compression of Anomaly Detection Model

Conclusions

This research presents a robust LSTM-based anomaly detection framework tailored for cloud infrastructure environments. The proposed method was evaluated using a combination of real-world datasets (Google and Azure traces) and synthetic data generated via simulation tools like CloudSim and Prometheus. Through comprehensive experimentation on Google Colab, the LSTM model demonstrated superior performance over traditional methods such as ARIMA, Isolation Forest, and One-Class SVM, achieving an F1-score of 0.90 and an AUC-ROC of 0.96.

By leveraging time-windowed sequences and training solely on normal behavior, the LSTM network effectively learned temporal patterns and accurately detected deviations indicative of anomalies. The dynamic thresholding mechanism further enhanced adaptability to workload drift, a common challenge in real-world cloud systems. Additionally, the integration with real-time monitoring tools and visualization through Grafana showcased the

practicality of deploying the model for live anomaly detection and alerting.

Overall, the proposed approach not only provides high detection accuracy but is also lightweight and resource-efficient, making it suitable for real-time cloud operations. Future work may extend this framework to multi-variate anomaly detection, cross-platform deployment, and reinforcement learning for automated remediation in dynamic cloud ecosystems.

REFERENCES

[1]. Armbrust, M., Fox, A., Griffith, R., et al. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58.

[2]. [Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys (CSUR), 41(3), 1–58.

[3]. [Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. Journal of Network and Computer Applications, 60, 19–31.

[4]. D. F. Pedroso, L. Almeida, L. E. G. Pulcinelli, W. A. A. Aisawa, I. Dutra and S. M. Bruschi, "Anomaly Detection and Root Cause Analysis in Cloud-Native Environments Using Large Language Models and Bayesian Networks," in IEEE Access, vol. 13, pp. 77550-77564, 2025, doi: 10.1109/ACCESS.2025.3565220.

[5]. H. Le, T. Hoang, M. Gabriel, G. Neumann and N. A. Vien, "Enhancing Exploration With Diffusion Policies in Hybrid Off-Policy RL: Application to Non-Prehensile Manipulation," in IEEE Robotics and Automation Letters, vol. 10, no. 6, pp. 6143-6150, June 2025, doi: 10.1109/LRA.2025.3564780.

[6]. H. Goumidi and S. Pierre, "Real-Time Anomaly Detection in IoMT Networks Using Stacking Model and a Healthcare- Specific Dataset," in IEEE Access, vol. 13, pp. 70352-70365, 2025, doi: 10.1109/ACCESS.2025.3563158.

[7]. M. Al-Sharafi et al., "Ensuring Zero Trust Security in Consumer Internet of Things Using Federated Learning-Based Attack Detection Model," in IEEE Access, vol. 13, pp. 54423-54438, 2025, doi: 10.1109/ACCESS.2025.3551212.

[8]. P. Renukadevi, S. Amaran, A. Vikram, T. Prabhakara Rao and M. K. Ishak, "Enhancing Cybersecurity Through Fusion of Optimization With Deep Wavelet Neural Networks on Denial of Wallet Attack Detection in Serverless Computing," in IEEE Access, vol. 13, pp. 47111-47122, 2025, doi: 10.1109/ACCESS.2025.3550735.

[9]. J. Liu et al., "SMore: Enhancing GPU Utilization in Deep Learning Clusters by Serverless-Based Co-Location Scheduling," in IEEE Transactions on Parallel and Distributed Systems, vol. 36, no. 5, pp. 903-917, May 2025, doi: 10.1109/TPDS.2025.3548320.