

Optimizing Business Process through Fault-Tolerant Scheduling in Cloud Environments: A Comparative Study

Anil Audumbar Pise^{1,*}

¹ Cumulus Solutions, South Africa

Email: anil@cumulussolutions.co.za

Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024 ★ Corresponding author

ABSTRACT

The fault tolerance study carried out in this research explores Bidirectional Long Short-Term Memory (LSTM) and Generative Adversarial Networks (GAN) to improve cloud computing dependability and functionality. Being an integral part of the range of business operations, cloud-computing fundamentals of resource provisioning and fault tolerance have a bearing on the overall cost dynamics, ROI, and OpEx. Reliability covers such issues as hardware failures, configuration problems, and other network issues that may have financial implications and even lead to revenue loss and failure to meet service-level agreements (SLA). The work develops a novel GAN–BiLSTM model for the accurate prediction of faults and the enhancement of recovery management, resulting in resource efficiency and cost-of-capital reduction (CapEx). Evaluation criteria involve deadline guarantee ratio, average task delay, and system scalability, confirming that the proposed model has better financial performance than DPSO and ANFIS. By cutting wastage of resources and increasing energy capacity in a system, the model displays attractive cost reduction and operating effectiveness for cloud service providers. In the simulation, important results of the model are demonstrated in business continuity and financial risk reduction, as well as in maintaining accurate and resourceful service in high-demand situations. All these developments have placed machine-learning-powered fault-tolerant systems as indispensable instruments that can also enhance profitability, resource utilisation, and sustainable competitiveness in the cloud computing business.

Keywords: Virtual Machines ▪ GAN ▪ BiLSTM ▪ Cost reduction ▪ Return on investment ▪ Operational expenditure ▪ Risk mitigation ▪ Service-level agreement ▪ Deep learning ▪ Virtual machine migration

1. INTRODUCTION

Cloud computing has emerged as a significant factor in corporate development, improving organizational productivity and financial efficiency and providing the ability to develop new products and services quickly. Cloud computing service delivery on demand means that capital-intensive resources for shared servers, storage, and applications are not purchased and maintained as fixed assets; operational expenses are minimized because resources are acquired and made available only as required, while scale and flexibility are improved [1].

It enables enterprises to remain focused on core competencies and hence enhances Return on Investment (ROI). IaaS, PaaS, and SaaS service models offer solutions to a range of requirements, making cloud computing a crucial resource for enabling competitive advantage and operational excellence.

At the same time, the financial and operational reliability of cloud computing depends on service continuity. Fault tolerance—the ability of a system to continue running when some part of it fails—is important for improving reliability, customer satisfaction, and financial returns. Vulnerabilities in hardware components, configuration settings, and networks

expose businesses to the risk of missing service-level agreements (SLAs), losing revenue, incurring extra costs, and damaging their reputation [2]. These disruptions directly affect Total Cost of Ownership (TCO) and overall profitability, magnifying the importance of strong fault-tolerance solutions for cloud service providers and companies that rely on cloud infrastructure [3].

Task resubmission, redundant models, and checkpoint-and-restart approaches are traditional but reactive methods that fail to cope with the scale of modern cloud environments [4]. They result in high costs, slow response, lower reliability, and a high volume of operations, all contrary to the objective of achieving higher efficiency. As cloud operations advance, businesses are more likely to turn to proactive fault-tolerance solutions. Such solutions must address risk and improve system dependability while maximizing resource utilization and preserving financial results [5].

To overcome these challenges, this work introduces a new fault-tolerance framework that incorporates sophisticated machine-learning algorithms, namely BiLSTM and GAN [6]. BiLSTM uses operational data to predict failure patterns accurately, while GAN builds capabilities for handling varied faults through simulation. This integration permits the detection of a fault while it is in progress and the rapid rebalancing of resources, greatly minimizing lost time and potential losses [7]. The proposed framework focuses directly on resource optimization and SLA compliance, minimizing operational expenses and enhancing the economic viability and stability of solution delivery [8].

Figure 1 represents a fault-management system in a cloud computing environment. It entails active communication by the cloud user with the server performing cloud operations. A fault-prediction module is used to predict failures before they occur. The secondary controller comes into contention when an issue arises so that execution can continue while fault-recovery mechanisms are in operation [9]. Faults affect VMs that are down through VM recovery or VM migration on separate servers to ensure that they remain available. The system synthesizes fault mechanisms for detection and resolution. By evaluating failures, this architecture emphasizes redundancy and predictive service management to offer dependable and continuous services and improve cloud infrastructure for user satisfaction [10].

The GAN-BiLSTM model is attractive for proactive fault detection because it can detect hardware faults, configuration errors, and network disruptions. This enables efficient recovery, reduces interruptions, and increases business efficiency. Cloud-resource utilization patterns directly lead to more economical use of cloud resources, reduced energy expenses, and increased SLA-compliance ratios, improving profitability and customer satisfaction [11]. The predictive nature of the model also helps prevent the over-allocation of resources to meet perceived future demand, which may otherwise lead to unnecessarily high costs.

The performance of the proposed framework has been examined through simulations comparing different financial and operational results. Parameters include the deadline guarantee factor, average task-execution time, resource usage, scalability, and energy efficiency [12]. Compared with techniques such as DPSO and ANFIS, performance analysis shows that

GAN-BiLSTM provides higher operational efficiency and lower costs in terms of system downtime and operational complexity. These enhancements help cloud service providers and enterprises sustain their market position as the need for resources and cost control grows.

In a business environment, this framework minimizes potential costs such as SLA penalties, downtime expenses, and wasted resources [13]. It therefore enhances an organization's capacity to meet customer expectations and fosters trust and market loyalty. Reduced energy consumption also supports sustainability objectives and improves organizational market positioning and brand image.

Machine-learning-based fault-tolerance mechanisms are therefore needed to address the challenges of complex future cloud environments [14]. Through reliability, scalability, and cost control, the GAN-BiLSTM solution provides a managerial approach to improving business continuity and financial security. It helps organizations obtain greater value from their cloud systems, avoid or mitigate risks, and generate improved revenues under escalating competitive pressure in the technological domain.

2. RELATED WORK

The literature discusses five practical scientific workflows for different applications: CyberShake in earthquake science, SIPHT in biology, Montage in astronomy, Epigenomics in genetics, and LIGO in gravitational physics [15]. Research details the unique structural and functional computational qualities of each workflow, together with its data, computational, and structural requirements. Pegasus WMS, a scalable workflow-management system, has been used to translate abstract descriptions of scientific workflows into executable code for distributed computing systems [16]. Its components provide a road map for building, executing, and evaluating complex scientific workflows.

Heterogeneous Event Management Middleware (HEMM) WMS was developed to provide inter-enterprise workflow event management for business applications [17]. However, the WMS does not install scientific applications and is designed only for commercial application deployment.

Dynamic Scheduling of Bag-of-Tasks-based workflows (DSB) is a scientific-workflow scheduling method that aims to minimize financial cost while accommodating user-defined deadline limitations. It improves the scheduling and allocation of Bags of Tasks (BoTs) by grouping processes according to priority constraints and data dependencies [18]. The method considers the heterogeneity and flexibility of pay-as-you-go Infrastructure-as-a-Service cloud platforms.

The Adaptive Data-Aware Scheduling (ADAS) method focuses on resource usage and process-completion time, but it does not provide fault tolerance, which is critical for workflow scheduling [19]. Two algorithms, PDC and DCCP, were developed to reduce computing costs and provide deadline-constrained scheduling. BAGS, a budget-driven algorithm for generating high-quality schedules, allocates funds to jobs and adjusts scheduling and resource provisioning using real-time environmental feedback [20]. The heuristic Deadline-Budget Workflow Scheduling (DBWS) method considers time and cost as two major constraints, although time limitations and

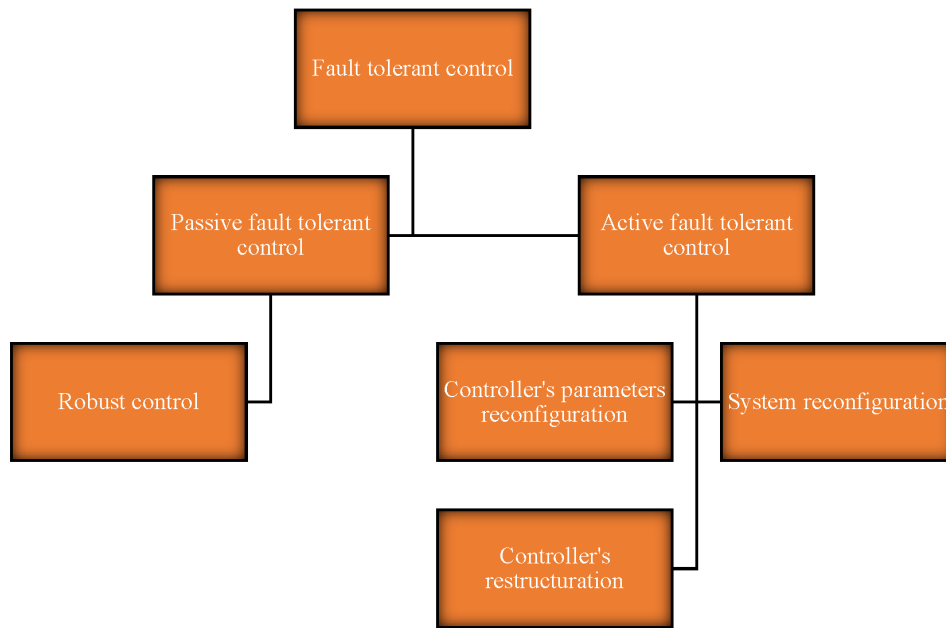


Figure 1. Fault tolerant classification.

fault tolerance still require improvement.

Scientific processes are power-hungry because they process large amounts of data and require substantial computing power [21]. A real-time dynamic scheduling system for task-based applications can reduce execution time and energy use by pairing a polynomial-time method with a multi-heuristic resource-allocation approach (MHRA). However, the inclusion of a fault-tolerant system was not treated as a crucial component.

Minimum Completion Time (MCT), Max-min, and Min-min are heuristic techniques used for task scheduling [22]. MCT prioritizes tasks with the shortest expected completion times and allocates the required resources. Max-min prioritizes large jobs on resources with the shortest runtimes, whereas Min-min prioritizes smaller activities.

The Heterogeneous Earliest Finish Time (HEFT) strategy has been proposed as a hybrid fault-tolerant approach for cloud computing systems [23]. It combines proactive and reactive strategies. Reactive approaches minimize the effect of failure on application execution, whereas proactive tactics aim to decrease failure time and increase throughput and capacity.

A fault-tolerant method is necessary when scientific workflows execute at bottleneck nodes; without it, the entire execution may be rendered useless [24]. FASTER is a dynamic, fault-tolerant scheduling tool for real-time scientific workflows. Its main elements are task reversal, horizontal and vertical scaling-up, and vertical scaling-down [25]. Nevertheless, financial and time restrictions were not considered significant obstacles to implementing scientific workflows.

Balanced and File Reuse-Replication Scheduling (BaRRS) is an approach to scheduling optimization in cloud computing systems [26]. It uses replication and data-reuse methods to improve data transfer among activities at run time and divides scientific workflows into sub-workflows to balance system usage through parallelization. Pipelining, integration, and disintegration, however, were not addressed.

Scientific-workflow applications are sets of interrelated, fine-

grained computing operations that run at different levels and require consistent service types [27]. Fault-Tolerant Clustering (FTC) mechanisms for ensembles of jobs include dynamic clustering, selective re-clustering, and dynamic re-clustering. Enhanced data-oriented scheduling strategies combine dynamic clustering with data-oriented scheduling and can reduce makespan and cost. Related systems include DSB, EDS-DC, HEMM, Pegasus WMS, ADAS, FASTER, BAGS, FTC, QFWMS, CFD, BDCWS, PDC and DCCP, DBWS, MHRA, and BaRRS. These methods address process management, integrated task management, energy-efficient scheduling, and fault-tolerant scheduling, but their constraints motivate a data-oriented, fault-tolerant, and energy-efficient workflow scheduling and management system.

3. PROPOSED WORK

The proposed system implements a new fault-tolerant architecture that includes Bidirectional Long Short-Term Memory (BiLSTM) networks and Generative Adversarial Networks (GAN) to handle important issues in cloud computing. It is designed to provide early fault detection and fault tolerance, maximizing a seamless flow of services without disruption—a factor with a major influence on business continuity and financial success.

Two controllers, one primary and one secondary, handle the proposed fault-tolerance module, as shown in Figure 2. A user's demand is received by the primary controller, which then assigns the work to the relevant virtual machine.

Algorithm 1: Fault-tolerance mechanism

- Step 1: The controller determines the task.
- Step 2: Verify whether the apparatus is available for the cloud-based assignment.
- Step 3: Integrate the training information set into the GAN.
- Step 4: Verify that the GAN can identify hardware problems.
- Step 5: The control mechanism determines the precision of

fault identification.

Step 6: A reconfigurable methodology permits the system to be reconfigured in response to recognized problems.

Different parts charged with managing the cloud system are orchestrated by the primary controller. GAN–Bidirectional LSTM is the specialized control algorithm that efficiently performs these objectives, and it is executed by the fault-prediction block. In contrast with traditional modules that focus on energy economy, scheduling, and load management, the current research installs failure-prediction modules to enhance the design.

3.1 Bidirectional LSTM

A BiLSTM network is an advanced recurrent neural network (RNN) for processing sequential data using both past and future information. It extends a traditional LSTM by combining two LSTM layers: one processes the input from beginning to end and the other processes it from end to beginning. This two-sided approach is useful for natural-language processing, speech recognition, and time-series analysis. BiLSTMs therefore provide a better representation of dependencies in sequences for tasks with sequential relationships.

The components of a BiLSTM are:

- **Forward LSTM:** analyzes the input sequence in the forward direction, from the first element to the last.
- **Backward LSTM:** analyzes a copy of the input sequence from the last element to the first.
- **Combining outputs:** combines the forward and backward LSTM outputs to produce the final output at each time step.

At every time step t , each LSTM unit uses the relevant gates and operations. The **forget gate** decides which components of the previous cell state should be forgotten. A sigmoid function produces values from 0 to 1, where 0 means forgetting and 1 means retaining information. It depends on the current input and the previous hidden state:

$$FG_t = \sigma(W_{FG}a_t + P_{FG}U_{t-1} + y_{FG}). \quad (1)$$

The **input gate** regulates which new data should enter the cell state. A sigmoid function controls the update, while tanh generates candidate values from the current input and previous hidden state:

$$IG_t = \sigma(W_{IG}a_t + P_{IG}U_{t-1} + y_{IG}), \quad (2)$$

$$\widetilde{CS}_t = \tanh(W_c a_t + P_c U_{t-1} + y_c). \quad (3)$$

The **cell-state update** integrates the forget gate, which retains or removes prior information, with the input gate, which incorporates new information. It maintains long-term dependencies that are important for autonomous application behavior:

$$CS_t = FG_t \odot CS_{t-1} + IG_t \odot \widetilde{CS}_t. \quad (4)$$

The **output gate** controls which part of the updated cell state contributes to the next hidden state. It uses the current input

and previous hidden state to provide a filtered cell state:

$$OG_t = \sigma(W_{OG}a_t + P_{OG}U_{t-1} + y_{OG}), \quad (5)$$

$$HG_t = OG_t \odot \tanh(CS_t). \quad (6)$$

Every input sequence $\{a_1, a_2, \dots, a_T\}$ is analyzed by two LSTM layers. The forward LSTM proceeds from $t = 1$ to T , updating a cell state and hidden state from preceding inputs. The backward LSTM proceeds from T to 1 and captures dependencies from future inputs. At time step t , the output is a combination of the forward and backward hidden states:

$$h_t^{\text{BiLSTM}} = \text{Combine}(\vec{h}_t, \overleftarrow{h}_t), \quad (7)$$

$$h_t^{\text{BiLSTM}} = \begin{pmatrix} \vec{h}_t \\ \overleftarrow{h}_t \end{pmatrix}, \quad (8)$$

$$h_t^{\text{BiLSTM}} = \vec{h}_t + \overleftarrow{h}_t. \quad (9)$$

Algorithm 2: BiLSTM

- Step 1: **Input:** sequence $a = [a_1, a_2, \dots, a_T]$, forward-LSTM parameters, and backward-LSTM parameters.
- Step 2: **Forward pass:** for $t = 1$ to T , compute $\vec{h}_t$ and $\vec{z}_t$ using the forward-LSTM equations.
- Step 3: **Backward pass:** for $t = T$ to 1, compute $\overleftarrow{h}_t$ and $\overleftarrow{z}_t$ using the backward-LSTM equations.
- Step 4: **Combine outputs:** for $t = 1$ to T , combine $\vec{h}_t$ and $\overleftarrow{h}_t$ to produce h_t^{BiLSTM} .

3.2 Generative Adversarial Network (GAN)

A Generative Adversarial Network is a deep-learning framework used to create new data from a given data set. It consists of two neural networks, a generator and a discriminator, that play an adversarial game. The generator outputs samples, and the discriminator attempts to distinguish real data from generated data. Through this process, the generator learns to synthesize samples that resemble real data. GANs are predominantly applied to image synthesis, enlargement of limited data sets, and video synthesis, and they have displayed excellent performance when simulating realistic data demands.

The GAN is designed to solve the following min–max optimization problem:

$$\min_G \max_D V(D, G) = \mathbb{E}_{a \sim p_{\text{data}}(a)} [\log D(a)] + \mathbb{E}_{c \sim p_z(c)} [\log(1 - D(G(c)))]. \quad (10)$$

The first term maximizes $D(a)$ for real samples, while the second minimizes $D(G(c))$ for fake samples and forces G to improve.

Algorithm 3: GAN training

- Step 1: **Input:** actual-data distribution p_{data} , noise distribution p_z , and learning rates α_D and α_G .
- Step 2: **Initialization:** initialize discriminator parameters θ_D and generator parameters θ_G .
- Step 3: **Training loop:** repeat for a predetermined number of iterations.

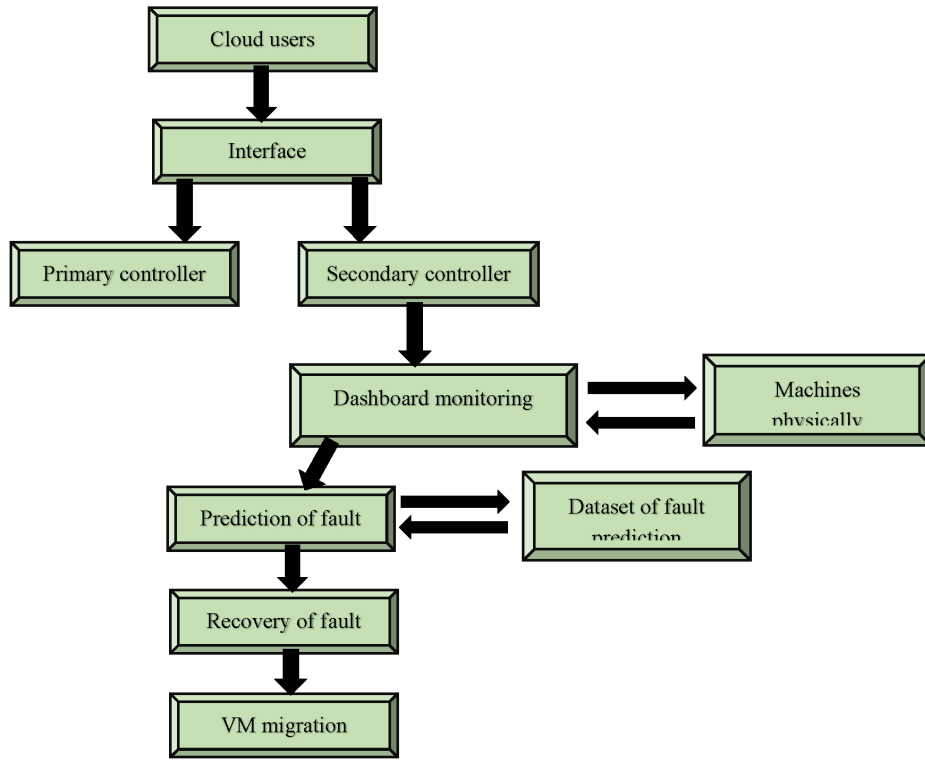


Figure 2. A GAN-Bidirectional LSTM infrastructure for a fault-tolerance technology.

- Train discriminator D : sample real data $a \sim p_{\text{data}}(a)$; sample noise $c \sim p_z(c)$ and generate fake data $\tilde{a} = G(c)$; then update θ_D to maximize

$$\mathcal{L}_D = \mathbb{E}_{a \sim p_{\text{data}}} [\log D(a)] + \mathbb{E}_{c \sim p_z} [\log(1 - D(G(c)))]. \quad (11)$$

- Train generator G : sample noise $c \sim p_z(c)$ and update θ_G to maximize

$$\mathcal{L}_G = \mathbb{E}_{c \sim p_z} [\log(1 - D(G(c)))]. \quad (12)$$

Step 4: **Repeat until convergence:** the generator eventually produces $G(c)$ that is indistinguishable from the actual-data distribution p_{data} , and D can no longer distinguish between them.

3.3 Bidirectional LSTM in GAN

The proposed architecture incorporates both GAN and bidirectional LSTM. The GAN supports adversarial training to achieve optimal performance. Compared with a high-dimensional information set, the generator network produces a simpler representation of the target domain and maps the source input. The opponent network, or discriminator, generates and evaluates output not categorized in the real-time data set and optimizes the generator-discriminator result.

The research uses a bidirectional LSTM technique to attain optimal weights and deliver accurate predicted outcomes. Training the regression model benefits the competing signal networks. To handle predicted, actual, and receptive-field categorization, the competing network employs a weight function, allowing tasks to be quantified effectively with real-time prediction and reducing errors associated with task catego-

rization. A suitable loss function is therefore required for robust training.

Figure 3 shows multiple elements, including generator G . The generator operates on a noise vector evaluated using an input distribution p_z . The input sample is transformed using LSTM layers employed on the noise vector. Finally, discriminator D categorizes the input samples according to the real distribution p_{data} and its characteristics.

The sigmoid activation function is used in the fifth and final layer of the second GAN:

$$D(a), D(G(c)) \in (0, 1). \quad (13)$$

The activation function is maximized and sample error is reduced as much as possible by forecasting against target values. The generator, in turn, reduces the discriminator's ability to identify fake samples. The discriminator's effectiveness is directly proportional to the power lost by the generator. For an input sample (x, y) , the probability estimate produced by the bidirectional-LSTM algorithm is

$$b = \text{softmax}(f(x; \theta)), \quad (14)$$

where x is the input, y is the label, and z is the class.

A cross-entropy function, as illustrated in Figure 4, determines the ideal bidirectional-LSTM weights. The fitness function is selected using the better probability estimator to improve resilience and enhance GAN-BiLSTM effectiveness.

Consider a loss function $L(y, b)$ and weights $\{w_1, w_2, \dots, w_K\}$. These weights combine the output at every stage using the loss-function value. Cross-entropy increases the probability of input-label projection, and the mean loss can be decreased by increasing the log-likelihood of the input information. The

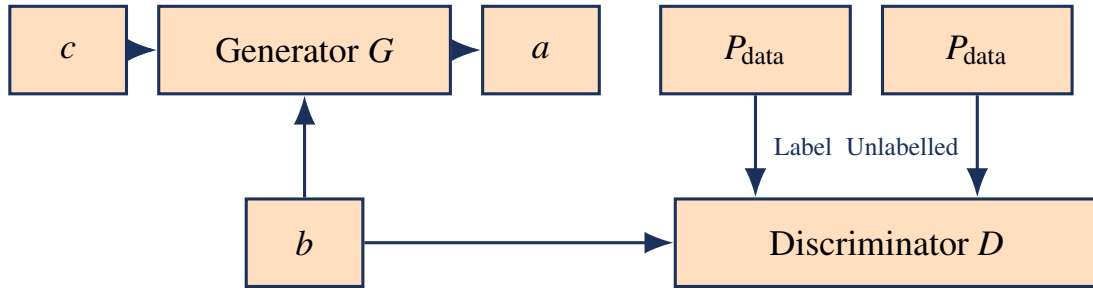


Figure 3. Generative adversarial network.

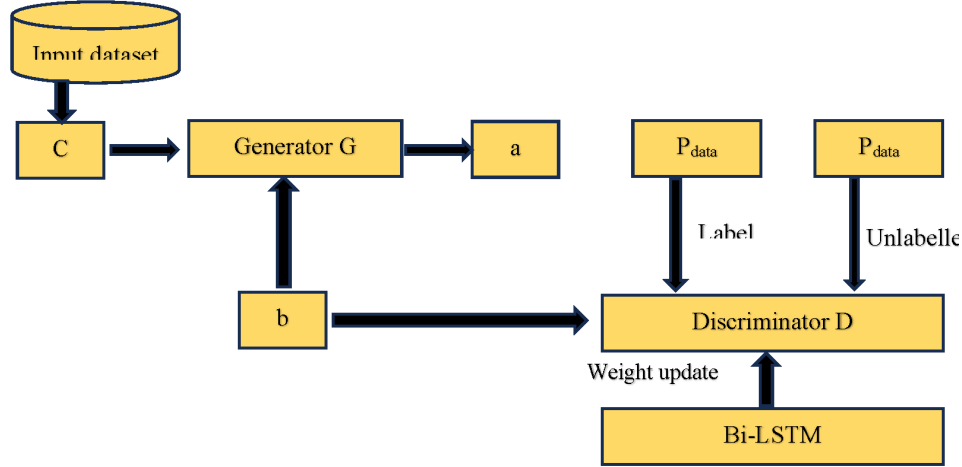


Figure 4. Training module based on the GAN-LSTM.

cross-entropy loss is defined as

$$z_E = - \sum_i [y \log(w_i) + (1 - y) \log(1 - w_i)]. \quad (15)$$

The class-based loss is

$$f = - \sum_{z=1}^M z_E(z, b) \log z_E(z, b), \quad (16)$$

where O denotes observed faults, p denotes predicted faults, M is the total number of classes, b is a binary indicator, and z is the class label. Across N training samples, the loss function is

$$\min_w f = - \sum_{i=1}^N f(y_i, b_i), \quad (17)$$

where y represents true labels or faults and b represents predicted labels or faults.

The GAN is provided with training data sets on hardware defects compiled from application logs. The trained GAN is then tested against a variety of real-time errors based on its training across different data sets. The proposed system, which combines LSTM with a bidirectional GAN network, stores outcomes from the previous stage and compares them with outcomes from the current stage, enabling a higher level of operation.

4. RESULT

The results show that the proposed GAN-BiLSTM framework improves the operational reliability and financial efficiency of cloud computing more effectively than existing methods. Performance indicators such as DGR and ATD

demonstrate SLA compliance and reduced downtime expenses, protecting revenue. The System Scalability Index and Resource Utilization Index indicate improved resource management that minimizes operating and asset costs. A low Resource Consumption Index (RCI) demonstrates reduced energy use, supporting sustainability objectives and providing cost-effective solutions for sustainable revenue.

Deadline Guarantee Ratio (DGR): the percentage of tasks finished on time in compliance with an SLA:

$$DGR = \frac{\text{Tasks completed within deadline}}{\text{Total tasks submitted}} \times 100. \quad (18)$$

Average Task Delay (ATD): the mean delay for tasks affected by system faults while migration or recovery is initiated, where n is the number of tasks:

$$ATD = \frac{1}{n} \sum_{i=1}^n (\text{Actual time}_i - \text{Expected time}_i). \quad (19)$$

System Scalability Index (SSI): the capacity of the system to maintain stable efficiency when more tasks are added:

$$SSI = \frac{\text{Total tasks}(\text{Delay sensitive} - \text{Insensitive})}{\text{Average execution time}}. \quad (20)$$

Resource Consumption Index (RCI): the ratio of resource-use activity, measured as active-host time, to the total number of tasks accomplished:

$$RCI = \frac{\text{Total active host time}}{\text{Total tasks}}. \quad (21)$$

Resource Utilization Index (RUI): the relation between the resources actually used during task performance and the total available resource time:

$$RUI = \frac{\text{Time resources are utilised}}{\text{Total active resource time}} \quad (22)$$

Operational Expenditure (OpEx): day-to-day expenses include energy use, resource administration, and fault-recovery procedures necessary to keep cloud infrastructure operating. Firms must minimize these expenses to maximize profit and maintain financial sustainability:

$$OpEx\% = \frac{\text{Baseline OpEx} - \text{Optimized OpEx}}{\text{Baseline OpEx}} \times 100. \quad (23)$$

Total Cost of Ownership (TCO): a complete cost model including all expenses incurred during cloud acquisition, deployment, and use throughout the system life cycle, including capital expenditure (CapEx) for hardware and software and operational expenditure for operation. Optimizing TCO improves ROI and supports sustainable financial savings:

$$TCO\% = \frac{\text{Baseline TCO} - \text{Optimized TCO}}{\text{Baseline TCO}} \times 100. \quad (24)$$

Accuracy: the overall correctness of the model in identifying faults:

$$Acc = \frac{\text{Correct positive} + \text{Correct negative}}{\text{Total cases}} \times 100. \quad (25)$$

Precision: the true-positive ratio relative to all predicted positive results:

$$Pre = \frac{\text{Correct positive}}{\text{Correct positive} + \text{Incorrect positive}} \times 100. \quad (26)$$

Recall: the proportion of actual positive cases classified correctly:

$$Rec = \frac{\text{Correct positive}}{\text{Correct positive} + \text{Incorrect negative}} \times 100. \quad (27)$$

Table 1. Comparison of the proposed approach with existing approaches.

Approach	Acc %	Pre %	Rec %
DPSO	88.5	89.2	84.5
ANFIS	86.5	86.2	86.2
GAN-BiLSTM	97.8	96.8	96.5

Table 1 and Figure 5 show the accuracy, precision, and recall of DPSO, ANFIS, and GAN-BiLSTM. DPSO achieves 88.5% accuracy, 89.2% precision, and 84.5% recall, indicating that it classifies many positive instances correctly but misses some that should be identified. ANFIS achieves balanced precision and recall of 86.2%, with an accuracy of 86.5%. The GAN-BiLSTM combination achieves the best results, with 97.8% accuracy, 96.8% precision, and 96.5% recall. This demonstrates the utility of GAN data augmentation and BiLSTM sequence analysis, reducing false positives

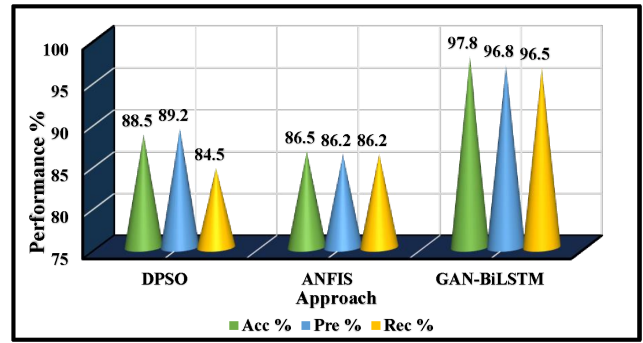


Figure 5. Comparison of the existing approaches with the proposed approach.

and false negatives in applications where context and high positive-case accuracy are priorities.

Table 2. DGR, TCO, and OpEx comparison.

Approach	DGR %	TCO %	OpEx %
DPSO	85.88	96.1	91.2
ANFIS	84.55	98.7	98.5
GAN-BiLSTM	96.44	66.2	72.1

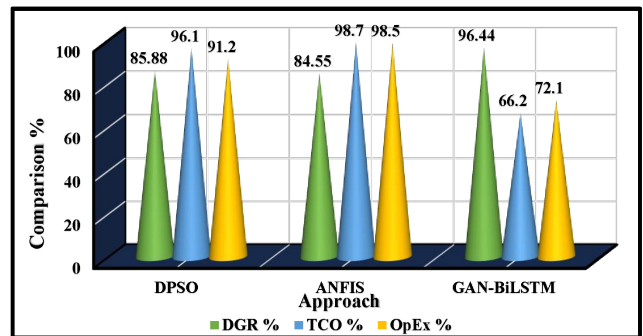


Figure 6. DGR, TCO, and OpEx comparison between existing and proposed approaches.

Table 2 and Figure 6 compare the GAN-BiLSTM model with DPSO and ANFIS across three key business metrics: Deadline Guarantee Ratio, Total Cost of Ownership, and Operational Expenditure. GAN-BiLSTM increases DGR to 96.44%, supporting higher SLA compliance and fewer penalties. Its TCO and OpEx values are 66.2% and 72.1%, respectively, showing that the model minimizes operational and capital costs and is more financially effective than the other methods.

Table 3. ATD, SSI, RCI, and RUI comparison.

Approach	ATD (ms)	SSI	RCI	RUI
DPSO	3.22	1.5	1.01	0.78
ANFIS	4.25	1.3	1.3	0.71
GAN-BiLSTM	1.01	2.1	0.7	0.9

Table 3 and Figure 7 compare GAN-BiLSTM, DPSO, and ANFIS using operational metrics. GAN-BiLSTM obtains an SSI of 2.1, demonstrating strong scalability and the ability to work with more traffic without reduced efficiency. It also achieves the best RUI, 0.9, and a lower RCI, 0.7, demonstrating better resource utilization and lower operating costs.

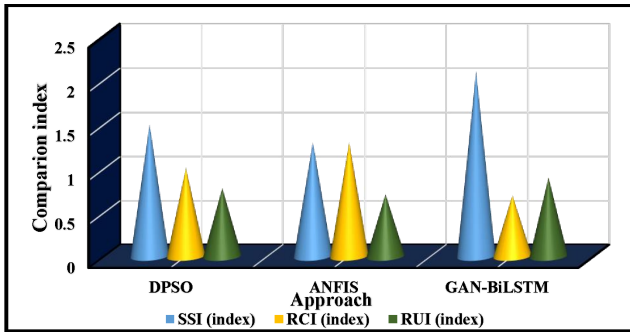


Figure 7. SSI, RCI, and RUI comparison between existing and proposed approaches.

Figure 8 shows Average Task Delay in milliseconds for DPSO, ANFIS, and GAN-BiLSTM. The proposed GAN-BiLSTM model is more efficient in minimizing task-execution delay, with a mean ATD of 1.01 ms, compared with 3.22 ms for DPSO and 4.25 ms for ANFIS. The lower ATD contributes to faster task completion, better response time, and improved system performance and productivity.

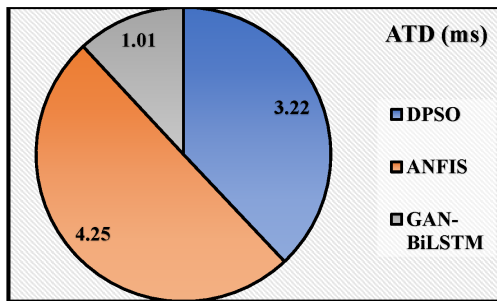


Figure 8. ATD comparison between existing and proposed approaches.

5. CONCLUSION

This research provides a novel fault-tolerance architecture that combines BiLSTM networks with GAN to solve emerging operational and financial issues in cloud computing. Because it predicts and eliminates faults in real time, the model helps meet SLA standards, controls downtime expenses, and improves service availability, all of which are critical to sustaining business operations. The framework minimizes OpEx and TCO through resource rationalization, reduced overprovisioning, and improved energy efficiency. These improvements yield higher ROI and long-term financial stability for cloud service providers and enterprises.

Dynamic resource management improves scalability and continuity of service delivery, supporting customer acquisition and retention and protecting revenue. The GAN-BiLSTM model addresses hardware failures, network breakdowns, and configuration issues more effectively than the compared methods. It therefore provides a financially sustainable and organizationally sound solution for contemporary cloud infrastructures, driving cost efficiency, sustaining competitive advantage, and promoting sustainable development as technology evolves.

Further research could investigate blockchain technology for secure fault disclosure and stronger data integrity. The model could also be enhanced with workload forecasting and edge-computing features to improve scalability and cost efficiency, making it suitable for the next generation of IoT-driven cloud

systems.

REFERENCES

- [1] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, and S. Hu, "Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort-based HEFT," *Future Generation Computer Systems*, vol. 93, pp. 278–289, 2019.
- [2] M. Zhao, Z. Han, and X. Du, "A survey of data center network topology structure," in *Proceedings of the 25th International Conference on Advanced Communication Technology (ICACT)*, 2023, pp. 303–309.
- [3] C. Acevedo, P. Hernandez, A. Espinosa, and V. Mendez, "A data-aware multiworkflow scheduler for clusters on WorkflowSim," in *Proceedings of the 2nd International Conference on Complexity, Future Information Systems and Risk*, 2017, pp. 79–86.
- [4] V. Roy and S. Shukla, "Designing efficient blind source separation methods for EEG motion artifact removal based on statistical evaluation," *Wireless Personal Communications*, vol. 108, pp. 1311–1327, 2019.
- [5] A. Sariga and J. Uthayakumar, "Type 2 fuzzy logic based unequal clustering algorithm for multi-hop wireless sensor networks," *International Journal of Wireless and Ad Hoc Communication*, vol. 1, no. 1, pp. 33–46, 2020.
- [6] I. V. Pustokhina, "Blockchain technology in the international supply chains," *International Journal of Wireless and Ad Hoc Communication*, vol. 1, no. 1, pp. 16–25, 2020.
- [7] I. Casas, J. Taheri, R. Ranjan, L. Wang, and A. Y. Zomaya, "A balanced scheduler with data reuse and replication for scientific workflows in cloud computing systems," *Future Generation Computer Systems*, vol. 74, pp. 168–178, 2017.
- [8] U. K. Jena, P. K. Das, and M. R. Kabat, "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment," *Journal of King Saud University—Computer and Information Sciences*, vol. 34, no. 6, pp. 2332–2342, 2022.
- [9] S. Sekigawa, C. Sasaki, and A. Tagami, "Toward a cloud-native telecom infrastructure: Analysis and evaluations of Kubernetes networking," in *Proceedings of the IEEE Globecom Workshops*, 2022, pp. 838–843.
- [10] K. Kiruthika, S. Gayathri, R. Hemalatha, and P. Menaga, "Design and development of mobile healthcare application for "ayurvedic" based clinical documents," *Journal of Cognitive Human-Computer Interaction*, vol. 1, no. 1, pp. 18–27, 2021.
- [11] A. G., A. Kumar, and V. R., "Query-based image retrieval using support vector machine (SVM)," *Journal of Cognitive Human-Computer Interaction*, vol. 1, no. 1, pp. 28–36, 2021.

- [12] M. Ghasemzadeh, H. Arabnejad, and J. G. Barbosa, "Deadline-budget constrained scheduling algorithm for scientific workflows in a cloud environment," in *Leibniz International Proceedings in Informatics*, vol. 70, no. 19, 2017, pp. 19.1–19.16.
- [13] V. Roy and S. Shukla, "Effective EEG motion artifacts elimination based on comparative interpolation analysis," *Wireless Personal Communications*, vol. 97, pp. 6441–6451, 2017.
- [14] H. Elhoseny and H. El-Bakry, "Utilizing service oriented architecture (SOA) in IoT smart applications," *Journal of Cybersecurity and Information Management*, vol. 0, no. 1, pp. 15–31, 2019.
- [15] A. Maseleno, "Design of optimal machine learning based cybersecurity intrusion detection systems," *Journal of Cybersecurity and Information Management*, vol. 0, no. 1, pp. 32–43, 2019.
- [16] Z. Ahmad, B. Nazir, and A. Umer, "A fault-tolerant workflow management system with quality-of-service-aware scheduling for scientific workflows in cloud computing," *International Journal of Communication Systems*, vol. 34, no. 1, pp. 1–23, 2021.
- [17] P. Banerjee, S. Roy, A. Sinha, M. M. Hassan, S. Burje, A. Agrawal, A. K. Bairagi, S. Alshathri, and W. El-Shafai, "MTD-DHJS: Makespan-optimized task scheduling algorithm for cloud computing with dynamic computational time prediction," *IEEE Access*, vol. 11, pp. 105 578–105 618, 2023.
- [18] S. Basu, M. Karuppiah, K. Selvakumar, K.-C. Li, S. K. H. Islam, M. M. Hassan, and M. Z. A. Bhuiyan, "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," *Future Generation Computer Systems*, vol. 88, pp. 254–261, 2018.
- [19] N. Singh, Y. Hamid, S. Juneja, G. Srivastava, G. Dhi-man, T. R. Gadekallu, and M. A. Shah, "Load balancing and service discovery using Docker Swarm for microservice-based big data applications," *Journal of Cloud Computing*, vol. 12, no. 1, pp. 1–9, 2023.
- [20] V. Roy *et al.*, "Network physical address based encryption technique using digital logic," *International Journal of Scientific and Technology Research*, vol. 9, no. 4, pp. 3119–3122, 2020.
- [21] K. Vinay, S. M. D. Kumar, S. Raghavendra, and K. R. Venugopal, "Cost and fault-tolerant aware resource management for scientific workflows using hybrid instances on clouds," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 10 171–10 193, 2018.
- [22] H. Arabnejad, C. Pahl, G. Estrada, A. Samir, and F. Fowley, "A fuzzy load balancer for adaptive fault tolerance management in cloud platforms," in *Proceedings of the European Conference on Service-Oriented and Cloud Computing*, 2017, pp. 109–124.
- [23] S. Shukla, V. Roy, and A. Prakash, "Wavelet based empirical approach to mitigate the effect of motion artifacts from EEG signal," in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, 2020, pp. 323–326.
- [24] J. Shah and D. Dubaria, "Building modern clouds: Using Docker, Kubernetes and Google Cloud Platform," in *Proceedings of the IEEE 9th Annual Computing and Communication Workshop and Conference*, 2019, pp. 184–189.
- [25] B. Cai, B. Wang, M. Yang, and Q. Guo, "AutoMan: Resource-efficient provisioning with tail latency guarantees for microservices," *Future Generation Computer Systems*, vol. 143, pp. 61–75, 2023.
- [26] L. Osman, O. Taiwo, A. Elashry, and A. E. Ezugwu, "Intelligent edge computing for IoT: Enhancing security and privacy," *Journal of Intelligent Systems and Internet of Things*, vol. 8, no. 1, pp. 55–65, 2023.
- [27] O. H. Embarak and R. A. Zitar, "Securing wireless sensor networks against DoS attacks in industrial 4.0," *Journal of Intelligent Systems and Internet of Things*, vol. 8, no. 1, pp. 66–74, 2023.