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

Anil Audumbar Pise[1,*]
[1]Cumulus Solutions, South Africa
Emails: anil@cumulussolutions.co.za

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 rage for 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 agreement (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. 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 the business continuity, financial risk reduction as well as maintaining accurate and resourceful service in high demand situations. All these developments have placed the fault tolerant systems powered by machine learning as indispensable instruments that can also enhance profitability, resources 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 application are not purchases and maintained as fixed assets, operational expenses are minimized as resources are only acquired and made available as required; the scale [1] and flexibility are improved. It enables enterprises to remain focused on enterprise competencies and hence enhances on 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.

However, at the same time, reliability, financial and operational, of cloud computing depends on the continuity of its services. This paper argues that fault tolerance that is the ability of a system to continue running when some part of it fails is important in improving reliability, customer satisfaction as well as financial returns. Because of the vulnerabilities present in the hardware components, configuration settings, and network issues, businesses are at risk of missing service level agreement (SLA), [2] losing revenue, incurring extra costs, and deteriorating their reputation. Both of these disrupt directly Total Cost of Ownership (TCO) and overall profitability [3], which magnifies the importance of strong fault-tolerance solutions for both cloud service providers and companies relying on cloud underlying structures.

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

Task resubmission, redundant models, and checkpoint and restart approaches are traditional but reactive methods that fail to cope with the scale of modern [4] cloud environments. It results in high costs, slow response, less reliable and high volume of operations, which are all contrary to the objectives of achieving higher efficiency when using these methods. While cloud operation reaches the next level, businesses are more likely to turn to antecedent fault-tolerance solutions. Suboptimal solutions must address risk [5] and provide better system dependability alongside maximizing resource utilization while preserving financial results.

To overcome these challenges, this work introduces a new fault-tolerance framework that incorporates sophisticated machine learning algorithms known [6] as BiLSTM and GAN. BiLSTM uses operation data in an accurate prediction of failure patterns and builds capabilities for handling varied faults through simulation by GAN. This integration allows the detection of a fault while it is in progress and the ability [7] to quickly rebalance resources, greatly minimizing the time lost and potential losses. [8] The existence of the proposed framework and its direct focus on the optimisation of resource imperative and compliance with Service Level Agreement minimisations operational expenses hence enhances the economic viability and stability of the solutions' delivery.

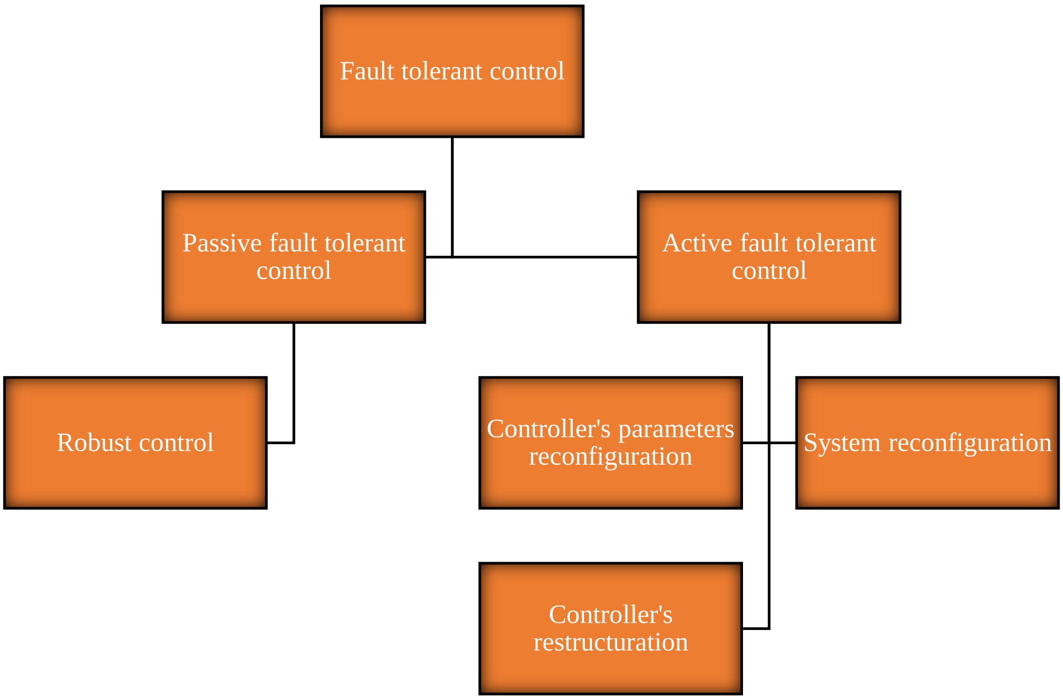


Figure 1. Fault tolerant classification

The Figure 1 represents a fault management system in cloud computing environment. It entails an active communication by the cloud user with the server that is performing the cloud operations. A fault prediction module is used to predict failures before they actually occur. The Secondary Controller comes into contention in case of an issue to continue execution while fault recovery [9] mechanisms are in play. Faults affect VMs that are down through VM recovery or VM migration on separate servers to ensure that they stay up. The system synthesizes fault technicism for detection and resolution. Evaluating failures, this architecture emphasizes redundancy and predictively [10] manage services for offering the dependable and continuous services for improving cloud infrastructure for users' satisfaction.

More so, the following aspects make the GAN-BiLSTM model attractive for proactive fault detection; it can detect hardware faults, configuration errors, and network disruptions. This in turn enables efficient measure of recovering hence reduces interruptions and increases business efficiency. The utilization patterns of cloud resources directly lead to better frugal utilization of [11] cloud resources, reduced energy expenses, and hence increased SLA compliance ratios that go a long way to increase the profitability and satisfaction levels amongst customers. However, the model also presents an advantage of predictive nature to prevent over allocation of resources in a bid to meet perceived future demand, which may lead to unnecessary high costs.

The performance of the proposed framework has been examined using simulations to compare different financial and operational results. Such parameters as the guarantee factor for the meeting deadlines, average time of task

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

execution, usage of resources in the system, scalability [12] and energy efficiency. When compared with other similar techniques like DPSO and ANFIS model, analysis of performance shows GAN-BiLSTEM has higher efficiency of operation and less costly in terms of more system down time and operational complexity. These enhancements help cloud service providers and enterprises to sustain their market position where the need for resources and cost control grows ever higher.

In business environment, this framework provides great benefit as it [13] minimizes potential costs such as SLA penalty charges, downtime expenses, and wasted resources. For this reason, it enhances an organisation's capacity to delivery on expectations of the customer and will foster trust and a loyal market. However, the reduction of energy consumption is a sustainability objective, which helps improve the organization's market positioning and brand image as well.

Therefore, we stress the need to apply machine learning-based fault-tolerance mechanisms to meet the challenges of complicated [14] cloud computing futures. Thus, due to reliability, scalability, and cost, the problem of GAN-BiLSTM provides a managerial approach to improve the continuity of businesses and their financial security. It helps organizations to get the most out of their clouds; avoid and/or mitigate risks; and generate more, better, and different revenues within a context of escalating competitive pressures in the technological realm.

2. Related Work

Discussed in the literature study are five practical scientific processes for different scientific applications, with an emphasis on the planned work on scientific workflows. A few examples of these workflows are cyberShake in the field of earthquake science, SIPHT in the field of biology, montage in the field of astronomy, epigenomics in the field of genetics, and LIGO in the field of gravitational physics. [15] The research details the unique structural and functional computational qualities of each scientific process in addition to their data and computational needs and structure. Pegasus WMS, a scalable workflow management system, was used to translate abstract descriptions of scientific workflows into executable code for distributed [16] computing systems. We designed the system's functionality after thoroughly detailing its components. In order to build, execute, and evaluate complex model scientific procedures, Pegasus WMS provides a route map.

To provide inter-enterprise workflow event management, a technology called Heterogeneous Event Management Middleware (HEMM) WMS, [17] for use in business applications. Having said that, the WMS does not install scientific applications; it is only designed for commercial application deployment.

'Dynamic Scheduling of Bag of Tasks based workflows' (DSB) is a scheduling method for scientific workflows that aims to minimize financial cost while accommodating user-defined deadline limitations. This approach improves the scheduling and allocation of Bag of [18] Tasks (BoTs) by grouping the process according to priority constraints and data dependencies. The authors take into account the characteristics of the pay-as-you-go IaaS (Infrastructure as a Service) cloud platform, which include heterogeneity and flexibility.

Resource usage and process completion time were the primary foci of the Adaptive Data Aware Scheduling (ADAS) proposed in [30]. But it doesn't have fault tolerance, which is critical for scheduling [19] workflows. In order to cut down on computing costs and provide a scheduling mechanism that is based on deadline constraints, two algorithms, PDC and DCCP, were developed.

BAGS, an acronym for "Budgetdriven Algorithm for [20] Generating high quality Schedules," allocates funds to various jobs and adjusts scheduling and resource provisioning based on real-time feedback from the environment. Time limitations and a fault tolerance mechanism are two areas that may need improvement, the heuristic method DBWS (Deadline-Budget Workflow Scheduling) was introduced for scientific workflow scheduling tasks, taking into account time and money as two major constraints.

Scientific processes are very power-hungry [21] since they process large amounts of data and need a lot of computing power. A real-time dynamic scheduling system designed for task-based applications to decrease execution time and energy usage. To accomplish these aims, a polynomial-time method was paired with MHRA, an approach for multi-heuristic resource allocation. The authors, however, failed to see the inclusion of a fault-tolerant system as a crucial component.

Minimum Completion Time (MCT), Maximum Minimum (Max-min), and Minimum-minimum [22] (Min-min) were among the heuristic techniques utilized for task scheduling. Tasks with the shortest expected completion times are given priority by the scheduling policy MCT, which then allocates the necessary resources to carry them out. Schedules with the Max-min policy prioritize big jobs on resources with the shortest runtimes, whereas those with the Min-min policy prioritize tiny activities.

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

The heterogeneous Earliest [23] Finish Time (HEFT) strategy was suggested as a hybrid fault-tolerant approach for cloud computing systems. It combines proactive and reactive strategies. There are three types of fault tolerance strategies used in cloud computing: proactive, reactive, and resubmission. The effect of failure on application execution may be minimized using reactive approaches, whereas proactive tactics aim to decrease failure time and boost throughput and capacity.

When it comes to scientific processes, a fault-tolerant method is very necessary for execution at [24] bottleneck nodes. Without it, the whole execution would be rendered useless. The FASTER algorithm is a dynamic, fault-tolerant scheduling tool for scientific procedures that run in real time. Task reversal, horizontal and vertical scaling-up methods, and [25] vertical scaling-down ways are the three main elements of FASTER.

Nevertheless, the study team did not see financial and time restrictions as significant obstacles to implementing scientific methods. One approach to scheduling optimization in cloud computing systems [26] is the BaRRS method, which stands for Balanced and file Reuse-Replication Scheduling. BaRRS uses replication and data reuse methods to enhance data transmission across activities during run-time, and it breaks scientific workflows into many sub-workflows to balance system usage via parallelization.

When it comes to scientific processes, the BaRRS algorithm is a great way to efficiently employ resources and use data reuse and replication strategies. Pipelining, integration, and disintegration are additional features of scientific procedures that the authors neglected to address. Applications in the scientific process are sets of interrelated [27]; fine-grained computing operations that run on different levels and need consistent service types. For this, somewhat computing, fault-tolerant clustering methods like the Fault-Tolerant Clustering (FTC) mechanism for ensembles of jobs are helpful. Dynamic clustering (DC), selective re-clustering (SR), and dynamic re-clustering (DR) are the three approaches.

By incorporating a data-oriented scheduling into EDS-DC and making use of a dynamic clustering fault-tolerant approach, an improved data-oriented scheduling strategy. By comparing EDS-DC to preexisting techniques, simulation findings revealed a considerable decrease in make-span and cost. A scientific workflow management system that is fault-tolerant and takes quality of service into account. Its four main parts are the workflow engine, workflow mapper, workflow scheduler, and workflow admission.

The authors laid out a comprehensive process for scientific data submission to result generation using a "cluster-based, fault-tolerant and data-intensive" (CFD) technique in a cloud context. Compared to other tactics, the CFD approach outperformed them. Budget-Deadline Constrained Workflow Scheduling (BDCWS) is a new technique that determines the optimistic spare budget by optimizing the available budget for each job based on their computational cost on the slowest virtual machine. Next, it uses the task-optimized available financial budget to regulate the range of virtual machines (VMs) selected, hence regulating the task computation cost. It subsequently creates a set of affordable VMs.

Several scientific workflow management and scheduling techniques are included in the literature study. These include DSB, EDS-DC, HEMM, Pegasus WMS, ADAS, FASTER, BAGS, FTC, QFWMS, CFD, BDCWS, PDC & DCCP, DBWS, MHRA, and BaRRS. Process management systems, integrated task management, energy-efficient scheduling, and fault-tolerant scheduling mechanisms were some of the ways in which these techniques handled and scheduled jobs for scientific process applications. Nevertheless, a scientific workflow scheduling and management system that is data-oriented, fault-tolerant, and energy-efficient is the result of these constraints.

3. Proposed Work

The present system suggests implementing a new fault tolerant status, which includes Bidirectional Long Short-Term Memory (BiLSTM), networks and Generative Adversarial Networks (GAN) for handling of important issues in cloud computing. This system is anticipated to be one that provides early fault detection as well as fault tolerance to maximize a seamless flow of services without disruption – a factor that is massively influential in the continuity of business and financial success.

There are two controllers—a main and a secondary—that handle the proposed fault tolerance module, as shown in Figure 2. A user's demand is received by the main controller, which then assigns the work to the relevant virtual machine.

Algorithm 1: Fault Tolerance mechanism

Step 1: Controller determines the task

Step 2: Verifies whether the apparatus is available to get to the cloud-based assignment.

Step 3: Integrate the training information set into GAN

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

Step 4: Verify that the GAN can identify problems with the hardware.

Step 5: The control mechanism determines the precision of fault identification.

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

Additionally, different parts charged with managing the cloud system are orchestrated or planned by the primary controller. Specifically, GAN Bidirectional LSTM is the specialized control algorithm that efficiently performs these objectives. The GAN Bidirectional LSTM is executed by the fault prediction block. In contrast to the traditional modules that focus on energy economy, organizing, and load management, the current research takes into account the installation of failure prediction modules to enhance the design.

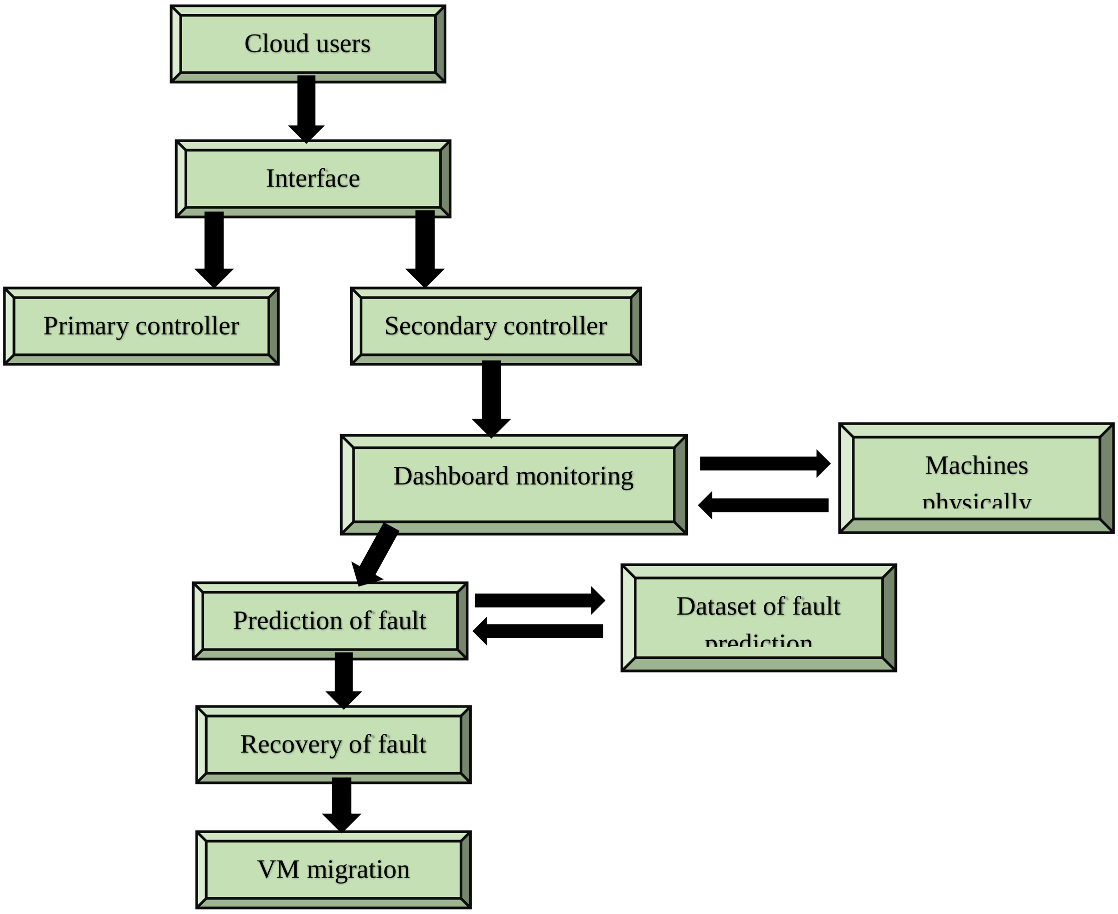


Figure 2. A GAN-Bidirectional LSTM Infrastructure for a Fault-Tolerance Technology

Bidirectional LSTM

BiLSTM network is one-step advanced recurrent neural network (RNN) meant for processing of sequential data by implementing both past and future information. It extends the traditional LSTM by combining two LSTM layers: ones that take the input from head to the toe and others that take input from toe to the head. Such a two-sided avenue helps with a range of tasks well in certain functions, such as natural language processing; speech recognition; and especially time series analysis. Thus, including these findings, BiLSTMs provide a better representation of dependencies in the defined sequences in tasks with sequential dependencies.

Components of a BiLSTM

- Forward LSTM: Just like RNN, LSTM also analyses the input sequence in the forward direction, from one to the last.
- Backward LSTM: Clone of the input sequence in the current process but with the former processed from the last time to the first time.
- Combining Outputs: Some researchers combine the forward and backward LSTMs' output and produce the final output for each time step.

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

At every time step t, every LSTM unit uses the relevant gates and operations:

Forget Gate: The forget gate decides which of previous cell state components should be forgotten. This function is a sigmoid function that they employed to estimate values ranging from 0 to 1; zero means forgetting and 1 means recall. It depends on the current input; it depends on the previous hidden state.

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

Input gate: The input gate regulates which new data should enter into the cell state. Where is suitable for control uses a sigmoid function while tanh is used for generating candidate values, it calculates a current update based on the current input and the previous hidden state.

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

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

Cell State Update: Cell state update process integrates forget gate that decides to maintain or eliminate prior information with input gate that decides to incorporate new information. It maintains long-term dependencies, which are important for the autonomous behavior of applications.

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

Output gate: The output gate controls, which part of updated cell state, will contribute to the next hidden state, which is the output. It is because of the current input and the 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 {a1, a2, …, aT} from a BiLSTM is analysed by two layers of LSTM, which are as follows:

Forward LSTM: Consequently, the forward LSTM follows a sequential process beginning with y=1 and ending with c=T, in that order. Every time iteration, from the first (before c=1) to the last (before T=), it updates a cell state c→b with data from the preceding input and calculates a so-called hidden state a→b. Depending on previous inputs, it computes a hidden state h→c and modifies a cell state Z→c at every stage.

Backward LSTM: From the most recent time step, which is T, all the way back to the first-time step, which is 1, the reverse LSTM follows the same sequence throughout the LSTM layers in the opposite direction of time. To capture input dependencies at future time steps, it determines a hidden state T ← c and modifies a cell state at location y : z ← y. It updates a cell state z←y and computes a hidden state h←y, considering dependencies from upcoming inputs.

When we get to time step t, the output is the sum (or function) of the concealed states going ahead and backward:

$$h_t^{BiLSTM} = Combine\left(\underset{h_t}{\rightarrow}, \underset{h_t}{\leftarrow}\right) \quad (7)$$

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

$$h_t^{BiLSTM} = \underset{h_t}{\rightarrow} + \underset{h_t}{\leftarrow} \quad (9)$$

Algorithm 2: BiLSTM

Step 1: Input:

- Sequence a= [a1, a2, …, aT}, Forward LSTM parameters, Backward LSTM parameters.

Step 2: Forward Pass:

- For t= 1 to T Compute h→t, z→t using forward LSTM equations.

Step 3: Backward Pass:

- For t= T to 1: Compute h←t, z←t using backward LSTM equations.

Step 4: Combine Outputs:

- For t=1 to T: Combine h→t and h←t to produce h_t^{BiLSTM}.

Generative Adversarial Network (GAN)

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

As a deep learning framework, a Generative Adversarial Network (GAN) was presented, to be used in creating new data based on a given set of data. It consists of two neural networks: a generator and a discriminator, which play a game with the nature of a confrontation. The generator outputs samples and the discriminator tries to identify the real and fake data set generated by the generator. In so doing, the generator learns to synthesize data samples in a way that they are as real as real data. They are predominantly applied for image synthesis, enlargement of limited dataset and video synthesis and have displayed excellent performance in simulation of realistic data demands.

A min-max optimization issue is what the GAN is designed to solve:

$$\min_{G} \max_{D} V(D,G) = E_{a\sim pdata}(a)[\log D(a)] + E_{c\sim pz}(c)[\log(1 - D(G(a)))] \quad (10)$$

First Term: Maximizes D(a) for real samples x.

Second Term: Minimizes D(G(c)) for fake samples G(c), forcing G to improve.

Algorithm 3: GAN training

Step1: Input

- Distributions of actual information ($p$data), noise distribution (pz), learning rates αD and αG.

Step 2: Initialization

- Put the variables θD (distinction) and θG (generators) to their initial values.

Step 3: Training Loop (Repeat for a predetermined number of times):

- Train the Discriminator D:

Sample real data a ∼ pdata(a).
Sample noise c ∼ pz(c) and generate fake data a ~ = G(c).
Update θD to maximize:

$$\mathcal{L}_D = E_{a\sim pdata}[\log D(a)] + E_{a\sim pz}[\log(1 - D(G(a)))] \quad (11)$$

- Train the Generator G:

Sample noise c ∼ pz(c).
Update θG to maximize:

$$\mathcal{L}_G = E_{c\sim pz}[\log(1 - D(G(c)))] \quad (12)$$

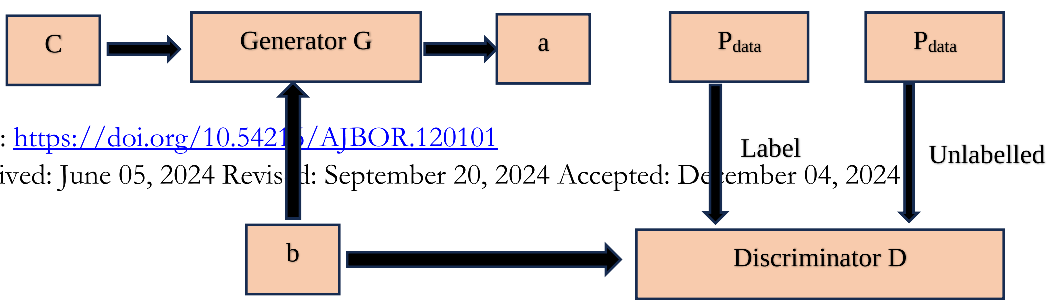
Step 4: Repeat Until Convergence

- At some point, the generator becomes so good that G(c) is identical to the distribution of actual information $p$data, and D can't tell the difference among the two.

Bidirectional LSTM in GAN

To construct the suggested technique, GAN and bidirectional LSTM are both included into the architecture. The GAN guarantees that the neural network that has been trained with the adversarial has been achieved optimal performance. In comparison to an information set with a high number of dimensions, the generator network is a simpler data set in respect to the target domain. It is believed to be the initial network that assists in mapping the source input. The opponent network, also known as the second network, tends to generate output that is not categorized in the real-time data set. On the other hand, the opponent network is responsible for optimizing the result of the discrimination-generator. For making categorization by the method of discrimination more straightforward, the final product of the generator with respect to genuine dataset samples.

The research makes use of a bi-directional LSTM technique to attain the optimal weight and deliver accurate predicted outcomes using the method. Additionally, the training of the regression model is beneficial to the network of signals of the competitors. In order to handle the predicted, actual, and receptive field categorization, the competing network employs a weight function. This allows the job to be quantified effectively with real-time prediction. This helps to eliminate mistakes that are associated with task categorization. Since this is the case, it is necessary to construct a loss function, which ought to be able to make a significant contribution to unfavourable training.



DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

Figure 3. Generative adversarial network

Figure 3 is an example of the multiple elements, one of which is a generator G. This device operates on a vector representation of noise z and is evaluated with the help of an input distribution pz. The input sample x is transformed using LSTM layers, which are employed in the noise vector z. Finally, the discriminator D tends to categorize the input samples based on the real distribution of pdata and its characteristics.

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

$$D(a), D\big(G(c)\big) \in (0,1) \qquad (13)$$

The function that activates the sample is maximized in this instance, and the error in the samples is reduced to the greatest extent possible via the process of forecasting with respect to the target values. The generator, on the other hand, diminishes the discriminator, in order to forecast the presence of bogus samples. The effectiveness of the discriminator is directly proportional to the amount of power that is lost by the generator.

Take into consideration a sample of input (x, y) in which this probability estimate is carried out by means of bidirectional LSTM algorithms as shown below:

$$b = softmax\big(f(x;\theta)\big) \qquad (14)$$

Here x= input, y = label, z= class.

A cross-entropy function, such as the one shown in Figure 4, is used to determine the ideal weights for the bidirectional LSTM. It is determined that the fitness function should be picked with the better probability estimator in order to be resilient. This results in an improvement in the effectiveness of the GAN with bidirectional LSTM with a potential weight.

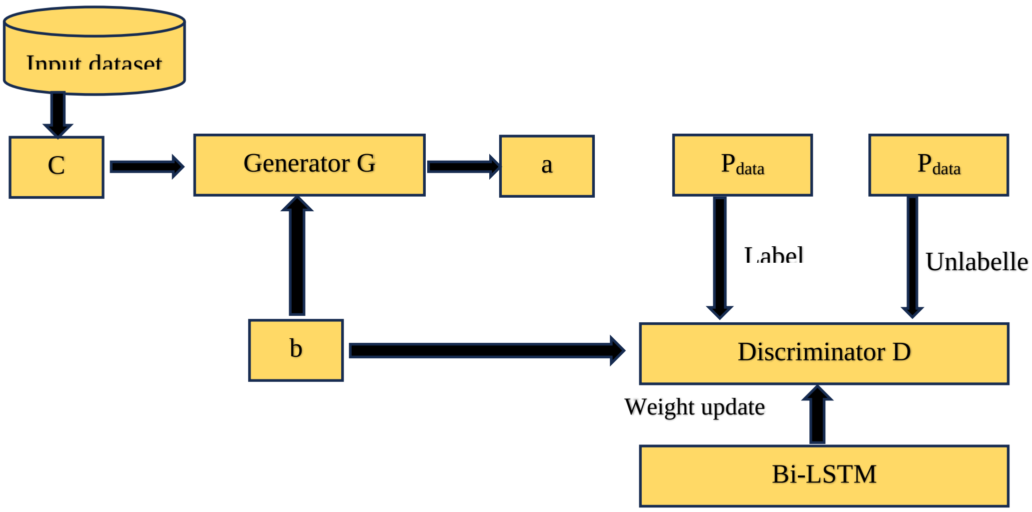


Figure 4. Training module based on the GAN-LSTM

Take into consideration a loss function. The L(y, b) and Bidirectional algorithms are employed to determine the ideal weights {w1, w2,…, wK}. These weights are then utilized for combining the output at every stage using the loss function value being utilized. Increasing the chance of input labels or labels projection is accomplished via the use of cross-entropy. It is possible to decrease the mean loss function values by increasing the log-likelihood character of the information that is being entered. The phrase that follows has been used to define the function that is responsible for cross entropy loss.

$$z_E = -\sum_i \big(y \log(w_i) + (1-y)\log(1-w_i)\big) \qquad (15)$$

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

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

Here O= observed faults, p= predicted faults, M= total classes, log = natural log, b = binary indicator, z = class label.

The following is the definition of the loss function taken across the training samples (N):

$$\min_{w} f = -\sum_{i=1}^{u} f(y_i, b_i) \quad (17)$$

Here y= true labels or fault, b = predicted labels or faults.

The generalized adversarial network (GAN) is provided with training datasets on certain hardware defects that have been compiled from a variety of log data onto real-time applications. A next step involves testing the trained GAN algorithm against a variety of real-time errors based on its training across several different trained datasets.

Based on the results of thorough investigation, on the work carried out on the acceptance of nodes on cloud, there are tiny ML-based techniques and other scheduling techniques that are implemented to optimize the capacity increase of the process that is present in the cloud node. The suggested system, which combines LSTM with a bi-directional GAN network, has the capacity to store the outcomes of the previous stage while comparing them with the outcomes of the current stage. This allows the system to operate to a higher degree than it would otherwise.

4. Result

The outcomes show that the indicated GAN-BiLSTM framework can improve the operational reliability and financial efficiency of cloud computing more effectively than the existing methods. Essential performance indicators like DGR and ATD prove SLA commitments and reduced downtime expenses that protects revenues. The System Scalability Index and Resource Utilization Index suggest that the practical framework enhances resource management to minimize operating cost and asset expense. Moreover, a low Resource Consumption Index (RCI) demonstrates energy usage reduction, which supports sustainability objectives and provides cost-effective solutions to generate sustainable revenue streams for the eternal success of the firm.

Deadline Guarantee Ratio (DGR): The percentage of tasks finished in time with compliance to Service Level Agreement (SLA).

$$DGR = \frac{Tasks\ completed\ with\ in\ deadline}{Total\ tasks\ submitted} * 100 \quad (18)$$

Average Task Delay (ATD): The mean time between failures for tasks which are impacted by system faults when migration is being done or recovery initiated, here n = number of tasks

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

System Scalability Index (SSI): The characteristic of the system to maintain a stable efficiency rate, even when more tasks are added onto the system.

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

Resource Consumption Index (RCI): Reflects the ratio between the activity of resource usage (active host time) to the total number of tasks accomplished.

$$RCI = \frac{Total\ Active\ Host\ time}{Total\ tasks} \quad (21)$$

Resource Utilization Index (RUI): The relation of the number of resources that were actually used in the course of the task performance to the number of the total available resources at the time of the performance of the task.

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

Operational Expenditure (OpEx): Day-to-day expenses, or OpEx, include things like energy use, administration of resources, and fault recovery procedures that are necessary to keep cloud infrastructure running. To maximize profits and maintain financial sustainability, it is crucial for firms to minimize operating expenses.

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

Total Cost of Ownership (TCO): TCO is a complete cost model including all costs, incurred during cloud acquisition, deployment and utilization throughout their life cycle including capital expenditure (Capex) for

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

hardware and softwares, and operational expenditure (opex) for operating. Optimizing TCO guarantees that you get a higher ROI and return on your investment hence attaining more sustainable financial savings.

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

Accuracy: Over the measure of the total slips, it calculates the general accuracy of the model in the identification of a fault.

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

Precision: High value indicates the true positive ratio in relation to predicted positive results.

$$Pre = \frac{Correct\ positive}{correct\ positive + Incorrect\ positive} * 100 \quad (26)$$

Recall: Shows how many of the actually positive cases the model was able to classify correctly.

$$Rec = \frac{Correct\ positive}{correct\ positive + Incorrect\ negative} * 100 \quad (27)$$

Table 1: Comparison of proposed approach with existing approach.

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

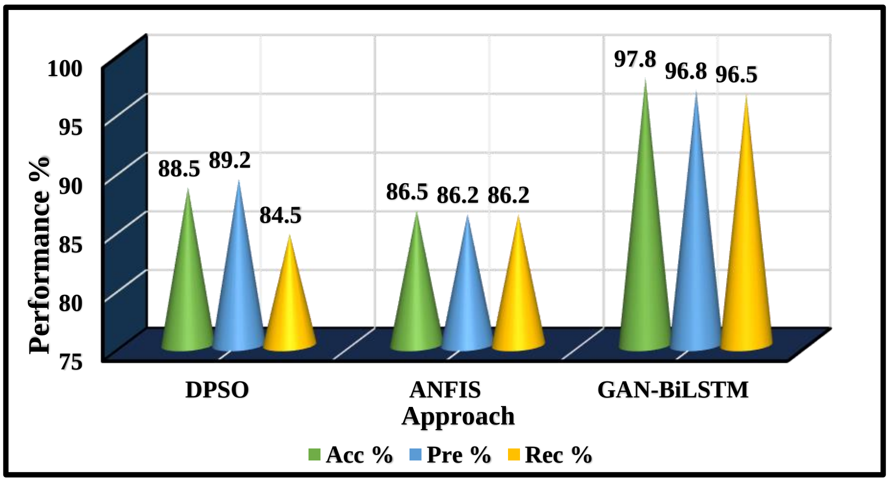


Figure 5. Illustration of comparing the existing approach with proposed approach

The Figure 5 shows the Acc, Pre, Rec results of three algorithms DPSO, ANFIS, and GAN-BiLSTM in the form of a tabular structure. From the evaluation of the proposed DPSO algorithm, we get an overall percentage accuracy of 88.5 %; Thus, the Precision is a bit high at 89.2 % but Recall is low at 84.5 % showing that the DPSO algorithm is good at correctly classifying the instances to the positive class but may miss some positive instances that should have been classified. Through ANFIS, we achieved appreciable results for both precision and recall 86.2% respectively, and thereby the accuracy 86.5 % though they have balanced slightly lower efficacy than DPSO.

The GAN-BiLSTM combination achieves better results, with the accuracy of 97.8%, precision of 96.8% and recall of 96.5%. This also clearly demonstrates its utility in exploiting the generative ability of GANs to augment data quality as well as utilization of BiLSTMs over sequences for improved performance. The higher two metrics show that using GAN-BiLSTM can seriously decrease the amount of false positives and negatives and is perfect for applications where context, together with high accuracy of positive cases is a priority.

Table 2: Comparing the DGR, TCO and OpEx with existing approach and proposed approach

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

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

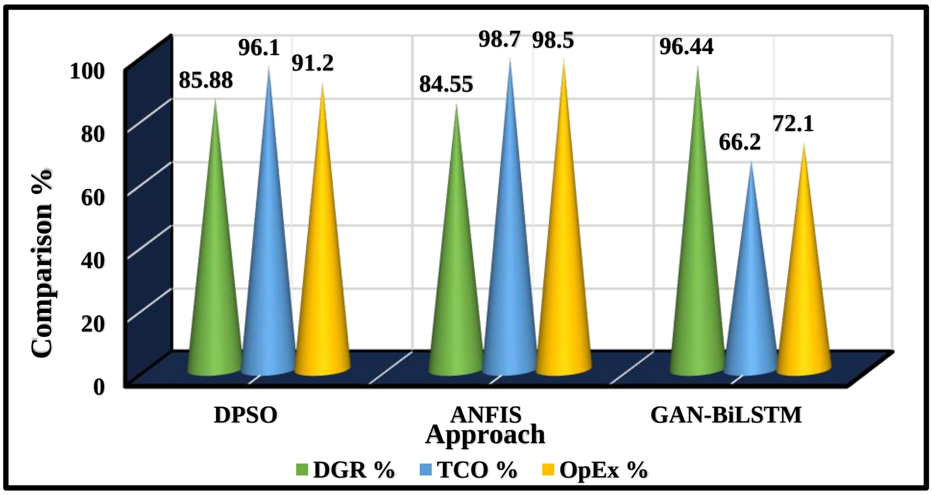


Figure 6. Illustration Comparing the DGR, TCO and OpEx with existing approach and proposed approach

The Figure 6 shows the performance of the GAN-BiLSTM model with DPSO and ANFIS across three key business metrics: There are three primary efficiency measures, which can be recommended: The Deadline Guarantee Ratio (DGR), The Total Cost of Ownership (TCO) and Operational Costs (OpEx). Dependable and relative largely superior to DPSO and ANFIS, the proposed GAN-BiLSTM model shall boost DGR to 96.44%, thereby keeping higher SLA compliance and penalties fewer. When it comes to TCO and OpEx, the proposed GAN-BiLSTM model results to 66.2% and 72.1%, respectively, fact that shows that implementing the model minimize not only the operational costs big amounts of money only for maintaining and operating certain processes but the capital expenditures as well, making it cheaper and financially effective compared to the other methods.

Table 3: Comparing the ATD, SSI, RCI and RUI with existing approach and proposed approach

Approach	ATD (ms)	SSI (index)	RCI (index)	RUI (index)
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

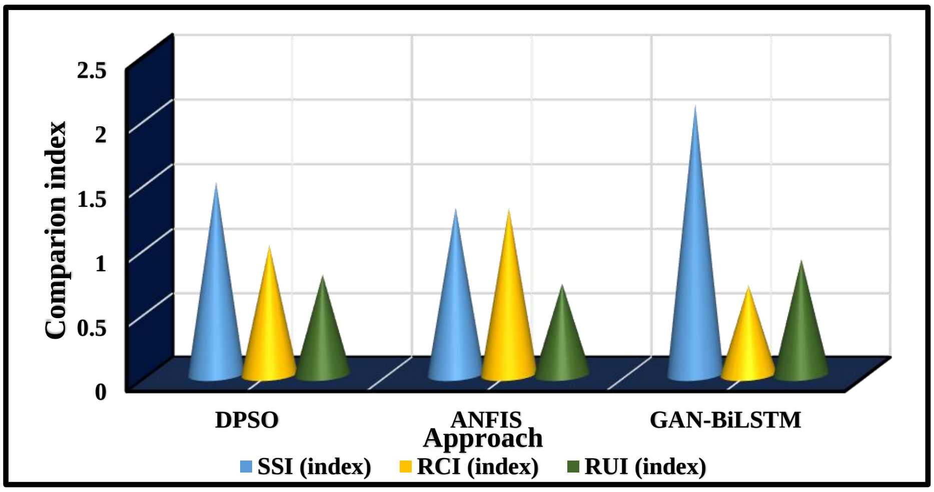


Figure 7. Illustration Comparing the SSI, RCI and RUI with existing approach and proposed approach

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

The Figure 7 compares the performance of the GAN-BiLSTM model with DPSO and ANFIS across three key operational metrics: System Scalability Index (Systems SSI), System Resource Consumption (RCI), System Resource Utilization (RUI). As for SSI, GAN-BiLSTM achieves a value of 2.1 that shows the model's outstanding scalability and its capability to work with a more significant number of traffic flows without reducing its efficiency. It also fares better than the others in RUI do by using available resources with a score of 0.9. In addition, by evaluating the RCI GAN-BiLSTM shows a better utilization of the resources and fewer operating costs in comparison with DPSO and ANFIS models.

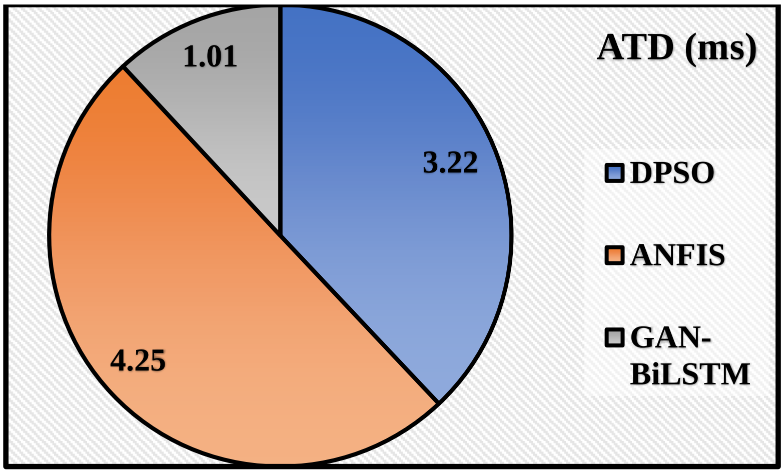


Figure 8. Illustration Comparing the ATD (ms) with existing approach and proposed approach

The Figure 8 shows the Average Task Delay (ATD) (measured in milliseconds) across three models: DPSO, ANFIS, and GAN-BiLSTM. Overall, the proposed GAN-BiLSTM model is more efficient in minimizing delay during execution of tasks when compared to DPSO and ANFIS with the mean ATD of 1.01 ms. however, defer can be computed that the DPSO model takes 3.22ms more time than FP while ANFIS model affects 4.25 ms more time than FP. The lower ATD in GAN-BiLSTM contributes for faster task completion resulting in better response time and hence better system performance and productivity.

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. Since it is able to predict and eliminate faults in real-time, the model guarantees that all sla standards are met, controls the expenses of downtime, and improves service availability, all of which are critical to sustaining business operations. The framework minimizes the OpEx and the TCO by resource rationalization and elimination of overprovisioning, as well as energy efficiency enhancement. These improvements are converted to higher return on investment (ROI) and financial stability in the long run for CSPs and enterprises. Dynamic management of resources is made possible by the system which improves scalability as well as continuity of service delivery which is critical in customer acquisition and retention in order to protect revenues. These are issues like hardware failure, network breakdown, configuration issues etc. that the GAN-BiLSTM model is able to achieve a better performance than the other method. This places it in the strategic tool kit of the enterprise that seeks to remain relevant in cost controlling and resource intense industries. Therefore, this fault-tolerance framework indicates a financially sustainable and organizationally sound solution for contemporary cloud infrastructures. They uniquely position it to drive cost efficiency, sustain competitive advantage, and promote sustainable development in pursuit of sustainability goals that underpin technical systems in organizations within the context of evolving technology.

Further research could be carried out, on how to apply blockchain technology to ensure safe disclosure of faults, and how to ensure high levels of data integrity. However, enhancing the model to include forecasting for workload as a way of optimisation and including the edge computing feature could even better enhance scale and cost environment making it suitable for the next era of IoT driven cloud system.

DOI: https://doi.org/10.54216/AJBOR.120101
Received: June 05, 2024 Revised: September 20, 2024 Accepted: December 04, 2024

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 Gener. Comput. Syst., vol. 93, pp. 278–289, Apr. 2019, doi: 10.1016/j.future.2018.10.046.

[2] M. Zhao, Z. Han, and X. Du, ''A survey of data center network topology structure,'' in Proc. 25th Int. Conf. Adv. Commun. Technol. (ICACT), Feb. 2023, pp. 303–309.

[3] C. Acevedo, P. Hernández, A. Espinosa, and V. Mendez, ''a data aware MultiWorkflow scheduler for clusters on WorkflowSim,'' in Proc. 2nd Int. Conf. Complex., Future Inf. Syst. Risk, 2017, pp. 79–86, doi: 10.5220/0006303500790086.

[4] V. Roy, S. Shukla, "Designing Efficient Blind Source Separation Methods for EEG Motion Artifact Removal Based on Statistical Evaluation," Wireless Pers Commun 108, pp. 1311–1327 (2019). https://doi.org/10.1007/s11277-019-06470-3.

[5] A. Sariga , 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 , (2020) : 33-46 (Doi : https://doi.org/10.54216/IJWAC.010102)

[6] Irina V. Pustokhina, Blockchain technology in the international supply chains, International Journal of Wireless and Ad Hoc Communication, Vol. 1 , No. 1 , (2020) : 16-25 (Doi : https://doi.org/10.54216/IJWAC.010103)

[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 Gener. Comput. Syst., vol. 74, pp. 168–178, Sep. 2017, doi: 10.1016/j.future.2015.12.005.

[8] U. K. Jena, P. K. Das, and M. R. Kabat, ''Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment,'' J. King Saud Univ., Comput. Inf. Sci., vol. 34, no. 6, pp. 2332–2342, Jun. 2022.

[9] S. Sekigawa, C. Sasaki, and A. Tagami, ''toward a cloud-native telecom infrastructure: Analysis and evaluations of kubernetes networking,'' in Proc. IEEE Globecom Workshops (GC Wkshps), Dec. 2022, pp. 838–843.

[10] Mrs.K.Kiruthika , Ms.S.Gayathri , Ms.R.Hemalatha , Ms.P.Menaga, Design and Development of Mobile Healthcare Application for "Ayurvedic" based Clinical Documents, Journal of Cognitive Human-Computer Interaction, Vol. 1 , No. 1 , (2021) : 18-27 (Doi : https://doi.org/10.54216/JCHCI.010103)

[11] Ajay G , Abhishek Kumar , Venkatesan R, Query-Based Image Retrieval using Support Vector Machine (SVM), Journal of Cognitive Human-Computer Interaction, Vol. 1 , No. 1 , (2021) : 28-36 (Doi : https://doi.org/10.54216/JCHCI.010104)

[12] M. Ghasemzadeh, H. Arabnejad, and J. G. Barbosa, ''Deadline-budget constrained scheduling algorithm for scientific workflows in a cloud environment,'' in Proc. Leibniz Int. Informat., 2017, vol. 70, no. 19, pp. 19.1–19.16, doi: 10.4230/LIPIcs.OPODIS.2016.19.

[13] V. Roy, S. Shukla. "Effective EEG Motion Artifacts Elimination Based on Comparative Interpolation Analysis." Wireless Pers Commun 97, 6441–6451 (2017). https://doi.org/10.1007/s11277-017-4846-3.

[14] Hisham Elhoseny , Hazem EL-Bakry, Utilizing Service Oriented Architecture (SOA) in IoT Smart Applications, Journal of Cybersecurity and Information Management, Vol. 0 , No. 1 , (2019) : 15-31 (Doi : https://doi.org/10.54216/JCIM.000102)

[15] Andino Maseleno, Design of Optimal Machine Learning based Cybersecurity Intrusion Detection Systems, Journal of Cybersecurity and Information Management, Vol. 0 , No. 1 , (2019) : 32-43 (Doi : https://doi.org/10.54216/JCIM.000103)

[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,'' Int. J. Commun. Syst., vol. 34, no. 1, pp. 1–23, 2021, doi: 10.1002/dac.4649.

[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. 105578–105618, 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 Gener. Comput. Syst., vol. 88, pp. 254–261, Nov. 2018, doi: 10.1016/j.future.2018.05.056.

[19] N. Singh, Y. Hamid, S. Juneja, G. Srivastava, G. Dhiman, T. R. Gadekallu, and M. A. Shah, ''Load balancing and service discovery using Docker swarm for microservice based big data applications,'' J. Cloud Comput., vol. 12, no. 1, pp. 1–9, Jan. 2023.

DOI: https://doi.org/10.54216/AJBOR.120101

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

[20] V. Roy et al., “Network Physical Address Based Encryption Technique Using Digital Logic”, International Journal of Scientific & Technology Research, Vol. 9, No. 4, 2020, Pp no.- 3119-3122.

[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 Appl., vol. 77, no. 8, pp. 10171–10193, Apr. 2018, doi: 10.1007/s11042-017-5304-7.

[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 Proc. Eur. Conf. Service-Oriented Cloud Comput. Cham, Switzerland: Springer, 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," 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, 2020, pp. 323-326, doi: 10.1109/CSNT48778.2020.9115761.

[24] J. Shah and D. Dubaria, ‘‘Building modern clouds: Using Docker, Kubernetes & Google cloud platform,’’ in Proc. IEEE 9th Annu. Comput. Commun. Workshop Conf. (CCWC), Jan. 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 Gener. Comput. Syst., vol. 143, pp. 61–75, Jun. 2023.

[26] Lobna Osman, Olutosin Taiwo, Ahmed Elashry, Absalom E. Ezugwu, Intelligent Edge Computing for IoT: Enhancing Security and Privacy, Journal of Intelligent Systems and Internet of Things, Vol. 8 , No. 1 , (2023) : 55-65 (Doi : https://doi.org/10.54216/JISIoT.080105)

[27] Ossama H. Embarak, Raed Abu Zitar, Securing Wireless Sensor Networks Against DoS attacks in Industrial 4.0, Journal of Intelligent Systems and Internet of Things, Vol. 8 , No. 1 , (2023) : 66-74 (Doi : https://doi.org/10.54216/JISIoT.080106)

DOI: https://doi.org/10.54216/AJBOR.120101

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