IEIE Transactions on Smart Processing and Computing, vol. 13, no. 5, October 2024
https://doi.org/10.5573/IEIESPC.2024.13.5.435

Effective Scheduling Algorithm for Workload Forecasting in Fog Environment Utilizing Dual Interactive Wasserstein Generative Adversarial Network

Ravi Kumar Suggala[1,*], Suma Bharathi. M[1], P.L.V.D. Ravi Kumar[1], and NVS. Pavan Kumar[2]

[1] Assistant Professor, Department of Information Technology, Shri Vishnu Engineering College for women (A), Bhimavaram, Andhra Pradesh, India {ravikumars, suma.thota, plvdravikumarit}@svecw.edu.in

[2] Associate Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India nvspavankumar@kluniversity.in

* Corresponding Author: Ravi Kumar Suggala, ravikumars@svecw.edu.in

Received April 13, 2023; Revised September 1, 2023; Accepted October 7, 2023; Published October 30, 2024

** Regular Paper*

Abstract: A decentralized computing system distributed among data-generating hardware and the cloud is called fog computing (FC). The ability to place resources to improve performance is given to users by such a flexible structure. On the other hand, low-delay services and limited resources make it difficult to use new virtualization technologies for task scheduling with resource management in fog computing. Several studies have examined scheduling and load balancing (LB) in cloud computing (CC). Nevertheless, countless LB initiatives have been proposed in fog environments. Task scheduling using a Dual interactive Wasserstein generative adversarial network (DIWGAN) optimized with an Improved Dwarf Mongoose Optimization algorithm is proposed to classify the suitable and not suitable server for the process (ESA-WF-FE-DIWGAN). Initially, the 'Cloud-Fog Computing Dataset is used. Afterward, the dataset is fed to the Fog Resource Monitor (FRM). Here, the statistical features like storage, computing, and RAM are extracted. Subsequently, the extracted features are given to DIWGAN to classify the suitable server and the unsuitable server for the process. The scheduling process was done using the Improved Dwarf Mongoose Optimization algorithm. The proposed approach achieves 3.101%, which was a 7.12% higher make span: 24.13% and 13.04% lower total cost; 2.292% and 5.365% higher ARU compared to the existing methods.

Keywords: Fog computing, Improved dwarf mongoose optimization algorithm, Fog resource monitor

1. Introduction

Fog computing is a hybrid paradigm that emerged from recent advances in mainstream parallel with distributed computing (PDC) technologies, such as cloud and edge computing. One of the best solutions currently available for leveraging distant computational cloud resources and performing calculations near the network edge is fog computing architectures [1]. Such settings leverage containers for lightweight, flexible, and fine-grained resource sharing amongst several fog devices. On the other hand, task scheduling is challenging in heterogeneous fog environments with unpredictable workloads [2]. Due to the shift in workloads to AI, ML, and DL, modern users also require extremely low reaction times and energy usage. The resource heterogeneity makes assigning tasks in these fog environments more complicated because different tasks have different preferences for the best cost-performance trade-off.

Existing Solutions: Recent works employ dynamic scheduling approaches that adjust to varying infrastructure conditions in real time to address the difficulties of volatile tasks scheduling in diverse fog environments [3]. A lot of schedulers adjust their estimate of the "expected reward" for dynamic QoS improvement using techniques like reinforcement learning. Because of their slow learning, long scheduling delays, and brittle modelling assumptions, they not necessarily appropriate for high non-stationary applications. Several approaches use virtual queues for scheduling in constrained contexts and combine maximum weight-based tactics with belief-based exploration to get around these issues. Recent research utilizing a deterministic deep surrogate model1 and gradient-based optimization has been shown to outperform such methods. Deep neural networks (DNN) [4] deterministic surrogate models, and gradient-base optimization enable less energy consume and quick response time. It is a result of its rapid real-time adaptation to various scenarios. Such techniques might not work well in situations unfamiliar to the model during training because they do not include state-space exploration or uncertainty modeling. Ignoring the curvature information of QoS optimization surfaces can result in these methods being trapped at local optima, because these surfaces are highly non-convex [5].

New Insights: To navigate the difficulties of unobserved settings, it is imperative to take uncertainty in the model predictions into account. Uncertainty generally results from various factors, including approximations in the model, inaccurate measurements, and parameter variations over time. In particular, a deterministic mode does not provide the entire distribution because of its unknown nature or the inherent randomness of certain factors, such as network delay, temperature, hardware defects. Therefore, an Effective scheduling approach for workload forecasting in a fog environment utilizing DIWGAN is proposed.

Talaat et al. [6] suggested a Scheduling approach for load balancing in fog computing utilizing CNN-MPSO, which attains a higher make span and lower LBL. Tuli et al. [7] Suggested GOSH: TS-GOSH-FCE, which provides a higher total cost and lower ARU. Teoh et al. [8] suggested IoT with fog computing depending on predictive care methods for effectively managing assets in Industry4.0 utilizing machine learning. It attains higher ARU and lower LBL. Sharma et al. [9] suggested Two-Stage Optimum Task Scheduling for a Smart Home Environment Utilizing Fog Computing. This system provides higher total cost and lower makes span.

The primary contributions of this paper are abridged below:

- EDLB using DIWGAN is proposed, which comprises three modules: (i) fog resource monitor, (ii) DIWGAN base classifier, (iii) optimized dynamic scheduler.
- When EDLB is compared with various LB approaches from the past, it achieves high resource utilization while reducing response time.
- Thus, it is a proficient way to guarantee consistent service. As a result, EDLB is straightforward and effective in real-time fog computing structures, such as the healthcare system.
- LB for FC has introduced several methods, but they all have numerous drawbacks. EDLB overcomes these restrictions and performs well in various situations.

Remaining paper is arranged as: division 2 illustrates the proposed method, division 3 proves the results, division 4 concludes this paper.

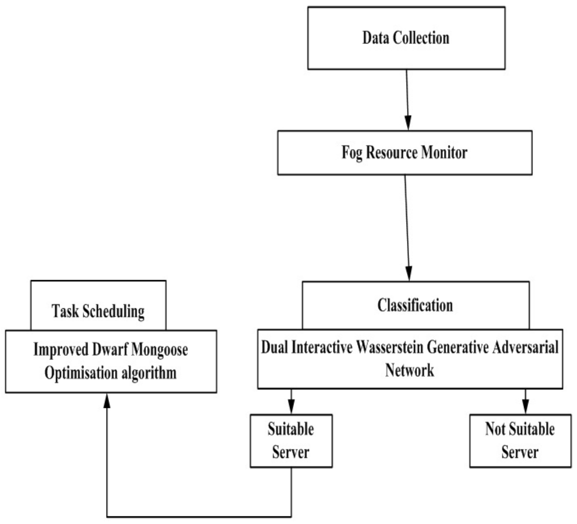
Data Collection
Fog Resource Monitor
Task Scheduling
Improved Dwarf Mongoose Optimisation algorithm
Classification
Dual Interactive Wasserstein Generative Adversarial Network
Suitable Server
Not Suitable Server

Fig. 1. ESA-WF-FE-DIWGAN task scheduling system.

2. Proposed Methodology

This segment discusses task scheduling using DIWGAN optimized with Improved Dwarf Mongoose Optimisation Algorithm (ESA-WF-FE-DIWGAN) [10]. Fig. 1 portrays the block diagram of ESA-WF-FE-DIWGAN task scheduling system. The comprehensive description of every stage is deliberated beneath:

2.1 Data Collection

Cloud-Fog computing is a hybrid computing paradigm that maximizes resource consumption and enhances overall system performance by merging the power of cloud computing with the proximity of fog computing [11]. Dynamic load balancing is a crucial part of this paradigm because it involves distributing computational tasks and network traffic efficiently across fog and cloud nodes based on workload variations and resource availability.

A Cloud-Fog Computing dataset used for load balancing typically consists of various parameters and metrics that reflect the current state of the system. These parameters include:

1. Workload characteristics: This includes the number and type of computational tasks, their processing requirements, arrival rates, and deadlines. The workload characteristics help evaluate the load on the system and determine the appropriate resource allocation.
2. Resource availability: This refers to the current status and capacity of fog and cloud nodes. It includes CPU utilization, memory usage, network bandwidth, and power consumption. These metrics are critical for assessing the available resources and making load-balancing decisions.
3. Network conditions: The dataset may also include network-related parameters, such as latency, bandwidth, and network traffic. This information helps determine the optimal placement of tasks based on the network proximity and congestion levels.

Table 1. Fog resources table.

Si	Storage (GB)	Computing (MHZ)	RAM (GB)	Type	Status
S_1	100	4000	7	Adequate, proper	Suitable
S_2	90	5000	8	Proper, not adequate	Inappropriate
S_3	200	2700	4	Adequate, not proper	Inappropriate
S_4	40	5000	6	Adequate, not proper	Inappropriate
S_5	130	5000	7	Adequate, proper	Suitable
S_6	80	6000	7	Proper, inadequate	Not suitable
S_7	120	7000	8	Adequate, proper	Suitable
	…	…	…	…	…
S_n	…	…	…	…	…

The number of data samples in a Cloud-Fog Computing dataset can vary under particular requirements and the complexity of the simulated environment. Typically, datasets used for load-balancing experiments contain a significant number of samples to capture different workload scenarios, resource states, and network conditions accurately. The dataset may provide details about the characteristics and configurations of the fog including cloud nodes. Fog nodes typically deployed at the network edge, near to the data source, and offer low-latency processing capabilities. They can be heterogeneous on the processing power, memory, and network connectivity. On the other hand, cloud nodes are located in data centers and offer vast computational resources with higher latency.

2.2 Fog Resource Monitor (FRM)

As shown in Table 1, this is in charge of observing every server resource, also recording the server data in the FRT database. FRT is situated in FRM server. FRM server is a fog server. Every fog server is categorized as appropriate or in appropriate as per the set of rules depending on storage, computing, and RAM.

The given steps determine the stages of every server are

i. Compute average storage $(AVG-STR)$.

ii. Compute average computing $(AVG-CMT)$

iii. Compute average RAM size $(AVG-RAM)$.

iv. Applying the following set of rules: (a) If $STR \geq AVG\_STR$ then $STR=1$, ELSE IF $STR < AVG\_STR$ then $STR=0$. (B) If $CMT \geq AVG\_CMT$ then $CMT=1$, ELSE IF $CMT < AVG\_CMT$ then $CMT=0$. (C) If $RAM \geq AVG\_RAM$ then $RAM=1$, ELSE IF $RAM < AVG\_RAM$ then $RAM=0$. STR, CMT, RAM are the input of DIWGAN to train the DIWGAN. The following section outlines the training process and DIWGAN CNN.

2.3 Classification using DIWGAN

This section discusses the DIWGAN, which categorizes each fog server. The data were sent to the Fog Resource Monitor (FRM), and each monitor synthesized one server. The FRT communicates to each other for sharing data from diverse inputdata storage [12]. The data distributions from FRM and the features stored database are compared using the Wasserstein distance to ensure FRM yielded accurate server information. The min–max problem between FRM can be described in Eq. (1),

$$MIN\ MAXP_{wgan}(F_{l(1)},R_{l(1)}) = -F_{X-Q_x}\begin{bmatrix} F_{l(1)}(X)+G_{Y-P_Y}\left[F_{l(1)}\left(R_{l(1)}(Y))\right)\right] \\ +\alpha F_{X-Q_j}\left[((\eta XF_{l(1)}(X)-1)^2\right] \end{bmatrix}, \quad (1)$$

The initial two terms refers Wasserstein distance assessment; X denotes input data from FRM; F_{X-Q_x} represents the collected database. G denotes the storage and R is the computing power. FRT epitomizes penalty coefficient. The network structure for the algorithm is provided as follows:

2.3.1 Generative Model

FRM consists of two paths they are contracting and an expansive path. By using an expanding path to comprehend information exchange and material deterioration, two monitors take features from the contracting path. The database fitters used to extract the features were 32, 48, 68, and 152.

2.3.2 Selector

The distance between the database and the fog resource monitor can be represented as Eq. (2)

$$D_l = \left|K_{xl} - K_{yl}\right| \ l=(1,2), \quad (2)$$

where the subscript l represents the FRM. By comparing the values, the performance of the databases was examined during each iteration. The FRT sends features to the network, and it classifies the features with the efficiency of DIWGAN in the next iteration.

2.3.3 Hybrid Loss Function

The data from FRT sends the features and is represented as Eq. (3),

$$T_2 = \frac{1}{hbd}\left|P(s)-t\right|, \tag{3}$$

where h represents the height; b is the width; d is the depth of the input sensor data. The method to improve the performance is expressed as Eq. (4)

$$q_x = \left\|\Gamma P_q - \Gamma Q_p\right\|^2 + \left\|\Gamma P_p - \Gamma Q_p\right\|^2 + \left\|\Gamma P_q - \Gamma Q_q\right\|^2, \tag{4}$$

where p and q denote gradient descent directions for input data P and Q. Γ denotes the suitable and unsuitable cauterization, resulting in a more realistic result that is expressed as Eq. (5),

$$Q_{ADV} = MIN\ MAXQ_{wgan}(F,R), \tag{5}$$

The features are transmitted to DIWGAN, which is represented in Eq. (6),

$$Q_s = \alpha_1 Q_1 + \alpha_2 Q_f + \alpha_3 Q_{data}, \tag{6}$$

The FRM can yield several databases by minimizing the features and classifying the servers into suitable and unsuitable using DIWGAN. Therefore, the classified servers are fed to the Task Scheduling process.

2.4 Task Scheduling using Improved Dwarf Mongoose Optimisation Algorithm

In this section, Task Scheduling using the Improved Dwarf Mongoose Optimisation approach (IDMO) is discussed [13]. The Improved Dwarf Mongoose Optimization Algorithm (IDMOA) offers several advantages for task scheduling and process assignment. IDMOA excels in assigning incoming processes to the most suitable server. It optimizes the allocation process by considering various factors, such as the server load, available resources, and processing capabilities. This ensures the processes are distributed effectively and efficiently, improving the overall system performance. IDMOA is highly scalable and suitable for large-scale systems and complex task-scheduling scenarios. It can handle many processes and servers while maintaining efficient process allocation. As the system grows, IDMOA can adapt and optimize the process assignment to accommodate the increased workload. IDMOA maximizes resource utilization through improved resource utilization considering the server load and available resources. It avoids overloading servers and distributes processes evenly across the available servers. This optimization results in the better utilization of system resources, reducing the idle time and increasing the overall efficiency. It exhibits robustness in handling dynamic environments and fluctuations in process arrival rates. It can adapt to changing conditions and adjust the process assignment strategy dynamically. This flexibility allows the algorithm to maintain optimal performance even in scenarios with varying workloads and server availability. IDMOA aims to minimize the response time of processes by efficiently assigning them to appropriate servers. Considering factors, such as server proximity and workload reduces the time taken to execute tasks. This leads to faster completion of processes and improved system responsiveness. IDMOA can be customized and fine-tuned to suit specific system requirements and objectives. The algorithm allows for the incorporation of different optimization criteria, such as energy efficiency or load balancing, based on the specific needs of the system. This flexibility enables the algorithm to adapt to diverse optimization goals. The incoming process must be assigned to the most suitable server by the IDMOA. When analyzing EDLB, including LB approaches, it may reduce the response time and attain higher resource usage. Therefore, this is an excellent way to confirm the endless service.

IDMOs are used for allocating the incoming tasks to a suitable server. An IDMO chooses the appropriate server from the obtainable appropriate servers in the Fog Resource Monitor.

2.4.1 Population Initialization

The IDMO populaces are initialized by a matrix of candidate dwarf mongooses (Y) that can be represented in Eq. (7),

$$Y = \begin{bmatrix} y_{1,1}\cdot & y_{1,e-1} & y_e \\ y_{2,1}\cdot & y_{2,e-1} & y_e \\ y_{m,1.} & y_{m,e-1} & y_m \end{bmatrix}, \tag{7}$$

where m is the number of dwarf mongooses. $y_{1,1}....y_{1,e-1}$ represent the storage, computing, and RAM features from the server.

2.4.2 Alpha Group

Assign the incoming process that is selected as the alpha, which controls the chaos and fits the database as expressed in Eqs. (8) and (9),

$$\beta = MIN(FIT1, FIT2...FITM), \tag{8}$$

$$Y_{j+1} = \beta + rand*(Y_j - Y_i/2), \tag{9}$$

where Y_j and Y_i are the selected dwarf mongooses.

2.4.3 The Babysitters

The dwarf mongooses (DM) replace the baby sitters, which means the information sent to servers and select the best incoming task and is represented in Eqs. (10) and (11)

$$Q = \begin{cases} roundup(0.7 * m * \dim * \left(\frac{1}{m}\right), \\ Q * m_{iter} * F when\ R \le 0 \end{cases} \qquad (10)$$

$$Y_{x+1} = \left(Y_x + rand * \left(\beta - \left(Y_x - Y_y\right)\right)/2 * br\right), \qquad (11)$$

where the best incoming task is selected, and again, the process is repeated. Thus, task scheduling is performed based on the IDMOA.

2.4.4 Termination

In this step, the task is scheduled with the help of IDMOA to iteratively repeat step 3 after selecting the process**.** Finally, the ESA-WF-FE-DIWGAN performs task scheduling with a higher ARU with a lower make span.

3. Result and Discussion

The simulation outcomes of the proposed scheduling algorithm (SA) for workload forecasting in FE utilizing the DIWGAN (ESA-WF) method is discussed. The simulation is done in JAVA using iFogSim simulator tool on Windows 10OS. The proposed ESA-WF-FE-DIWGAN method achieved feasible outcomes under several performance metrics, such as Makespan, Total cost, ARU, and LBL. The obtained results of the proposed method ESA-WF-FE-DIWGAN were analyzed with the existing methods, such as TS-LB-FE-CNN-MPSO and TS-GOSH-FCE, respectively.

3.1 Performance Measures

The performance metrics, such as Make span, Total Cost, ARU, and LBL, are examined.

3.1.1 Makespan

The total time requisite to accomplish all tasks is scaled using Eq. (12).

$$Makespan = Max\left(CT\left(ti\right)\right), \qquad (12)$$

Here CT denotes time ti completes its processing.

3.1.2 Total Cost

Assume a task is completed in a cloud-fog system; Eq. (13) shows that processing, memory use, and bandwidth are charged minimally:

$$Total\ cost = \sum_{j=0}^{m} CP\left(ti\right) + CM\left(ti\right) + CB\left(ti\right), \qquad (13)$$

Here CP denotes processing cost.

Table 2. Comparison of Makespan.

Techniques	Makespan (ms)			
	50	90	130	150
TS-LB-FE-CNN-MPSO	34.22	44.34	63.33	34.33
TS-GOSH-FCE	56.45	45.77	67.56	66.33
ESA-WF-FE-DIWGAN (proposed)	23.32	15.67	5.33	23.20

3.1.3 ARU

Every resource usage present in the ecology of fog and is represented in Eq. (14),

$$ARU = \frac{\left(BS + OL\right)}{FS_s} \times 100\%, \qquad (14)$$

Here BS denotes count of balancing FS_s ; OL represents count of overload FS_s ; FS_s indicates a count of obtainable fog servers.

3.1.4 LBL

LBL measures the load level. A count of overloaded fog servers is subdivided from the percentage of balanced fog servers represented in Eq. (15):

$$LBL = \frac{BS}{FS_s} \times 100\%, \qquad (15)$$

3.2 Performance Analysis

Tables 2-5 presents the outputs of the proposed ESA-WF-FE-DIWGAN method. The performance metrics, such as Makespan, Total cost, ARU, and LBL are analyzed. Here, the proposed ESA-WF-FE-DIWGAN technique is compared with existing methods, such as TS-LB-FE-CNN-MPSO and TS-GOSH-FCE.

Efficient dynamic load balancing performance analyzed with existing approaches by deeming the metrics. Table 2 shows a comparison of Make span.

The proposedESA-WF-FE-DIWGANmethod provides 32.21%, which was a 37.56% lower Make span for50 tasks; 25.45%, which was a 45.89% lower Make span for 90 tasks; 56.34%, which was a 21.23% lower Make span for 130 tasks; 25.45%, which was 45.89% lower Make span for 150 tasks; evaluated as the existing methods TS-LB-FE-CNN-MPSO and TS-GOSH-FCE

Table 3 compares the total cost. Here, the proposedESA-WF-FE-DIWGANmethod provided32.21%, which was a 37.56% lower entire cost for 50 tasks; 25.45%, which was 45.89% lesser entire cost for 90 tasks; 56.34%, which was a 21.23% lowerentire cost for 130 tasks; 25.45%, which was a 45.89% lower Make span for 150 tasks; these were evaluated to the existing methods TS-LB-FE-CNN-MPSO and TS-GOSH-FCE.

Table 3. Comparison of Total cost.

Techniques	Total cost (%)			
	50	90	130	150
TS-LB-FE-CNN-MPSO	34.36	56.67	76.77	78.56
TS-GOSH-FCE	35.56	67.66	67.66	67.46
ESA-WF-FE-DIWGAN (proposed)	20.33	26.44	15.33	10.55

Table 4. Comparison of ARU.

Techniques	ARU(%)			
	50	90	130	150
TS-LB-FE-CNN-MPSO	76.77	65.45	79.77	82.45
TS-GOSH-FCE	80.34	82.34	75.34	69.66
ESA-WF-FE-DIWGAN (proposed)	91.34	90.77	92.34	95.66

Table 5. Comparison of LBL.

Techniques	LBL(%)			
	50	90	130	150
TS-LB-FE-CNN-MPSO	76.34	83.34	88.56	78.77
TS-GOSH-FCE	82.45	85.45	78.77	70.22
ESA-WF-FE-DIWGAN (proposed)	95.44	98.44	92.66	93.34

Table 4 compares the ARU. Here, the proposedESA-WF-FE-DIWGANmethod provides a 56.45% and 67.68% higher ARU for 50 tasks; 73.56%, which was 76.44% higher ARU for 90 tasks; 57.57%, which was 59.3% higher ARU for 130 tasks; 67.78%, which was 45.89% higher ARU for 150 tasks; these were evaluated to the existing methods TS-LB-FE-CNN-MPSO and TS-GOSH-FCE.

The proposed approach produced better performance than the existing algorithms as greater ARU and higher LBL are attained, as listed in Tables 3 and 4. The experimental outcomes show that the proposed algorithm consumed less and reduced the number of migrations.

Table 5 compares the LBL. Here, the proposedESA-WF-FE-DIWGANmethod provided a32.21%, which was a 37.56% higher LBL for 50 tasks; 25.45%, which was a45.89% higher LBL for 90 tasks; 56.34%, which was a 21.23% higher LBL for 130 tasks; 25.45%, which was a 45.89% higher LBL for 150 tasks; these were compared with the existing methods TS-LB-FE-CNN-MPSO and TS-GOSH-FCE respectively.

Furthermore, the cost was paid for the processor, memory, and bandwidth utilization. The LBL in the ESA-WF-FE-DIWGAN approach was significantly lower than that of the previous methods (Table 5). ESA-WF-FE-DIWGAN took a variety of factors into account. ESA-WF-FE-DIWGAN has combined TS-LB-FE-CNN-MPSO and TS-GOSH-FCE, the ESA-WF-FE-DIWGAN to split the resources, including a decrease in the level of resource search that paves to effectual resource usage. DIWGAN reaches greater accuracy in resource managing activities.

Table 6. Benchmark table.

Author name	Makespan (ms)	Total cost	LBL (%)	ARU (%)
Talaat et al. [6]	34.33	78.56	78.77	82.45
Tuli et al. [7]	66.33	67.46	70.22	69.66
Teoh et al. [8]	50.14	60.14	75.12	60.14
Sharma et al. [9]	-	-	-	-
ESA-WF-FE-DIWGAN (proposed)	23.20	10.55	93.34	10.55

A decentralized computing system located amid data-producing devices and clouds is known as fog computing. Users are able to allocate resources to enhance performance to this flexible structure. Nevertheless, the application of novel virtualization for task scheduling and resource management in fog-computing is hindered by a lack of resources and low-latency services. Numerous studies have been conducted on scheduling and LB in cloud computing. On the other hand, numerous LB initiatives have been presented in the fog architectures. Some problems arise when considering how tasks are routed between fog nodes and the cloud in different physical devices. Scheduling in fog is highly challenging because of the volume and diversity of the devices. Only a few studies have been done so far. LB is a particularly attractive and significant area of study in FC because of its focus on maximizing resource utilization. LB has several difficulties, including safety with fault tolerance. Table 6 shows that The Proposed ESA-WF-FE-DIWGAN model provided 4.52%, 7.67%, and 11.94% lower make span than the existing methods, such as Talaat et al. [6], Tuli et al. [7], Teoh et al. [8]. The Proposed ESA-WF-FE-DIWGAN model provided 22.33%, 28.70%, and 14.79% lower cost than existing methods like Talaat et al. [6], Tuli et al. [7], Teoh et al. [8] respectively. The proposed ESA-WF-FE-DIWGAN method provided26.07%, 14.36%, and 14.32% lower LBL than the existing methods, such as Talaat et al. [6], Tuli et al. [7], Teoh et al. [8]. The ESA-WF-FE-DIWGAN method provided 9.80%, 14.58%, and 12.31% higher ARU than existing methods, 2022, Talaat et al. [6], Tuli et al. [7], Teoh et al. [8].

Discussion

This paper proposed EDLB using DIWGAN. EDLB comprises three primary modules: (i) fog resource monitoring, (ii) DIWGAN base classifier, (iii) Improved Dwarf Mongoose Optimization algorithm. EDLB uses a dynamic real-time scheduling approach to achieve LB in fog computing. FRM is accountable for tracking every server resource and protecting the server data. The classifier depends on DIWGAN is accountable for categorizing every fog server as appropriate or inappropriate. IDMO is accountable for choosing the best server for the incoming process. EDLB decreases the response time and reaches higher resource usage. This is a better mode to assure the uninterrupted service. Multiple FC systems have been introduced in LB, but they all have numerous drawbacks. EDLB deals with these drawbacks and achieves better performance in multiple scenarios. Compared to other LB algorithms, EDLB achieves better make span, average resource utilization, and load balancing levels. An unforeseen surge in demand on the server responsible for that task may result in an EDLB real-time task failure. When the system notices an abrupt increase in load on server hosting a task of real-time, it reschedules a vital task for different server to address this problem.

4. Conclusion

DIWGANoptimized with an Improved Dwarf Mongoose Optimisation algorithmwas implemented for scheduling the tasks (ESA-WF-FE-DIWGAN). The proposed ESA-WF-FE-DIWGAN approach was executed by Java. The proposed ESA-WF-FE-DIWGAN approach achieved12.03%, which was a 24.85% higher make span than existing methods, such as TS-LB-FE-CNN-MPSO and TS-GOSH-FCE.

Reference

[1] S. Iftikhar, M.M.M. Ahmad, S. Tuli, D. Chowdhury, M. Xu, S.S. Gill, and S. Uhlig, “HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments,” *Internet of Things,* vol. 21, p. 100667. Apr. 2023. Article (CrossRef Link)

[2] G. Shruthi, M.R. Mundada,, B.J Sowmya, and S. Supreeth, “Mayfly tayloroptimisation-based scheduling algorithm with deep reinforcement learning for dynamic scheduling in fog-cloud computing,” *Applied Computational Intelligence and Soft Computing*, vol. 198, no. 1, pp. 117273, Aug. 2022. Article (CrossRef Link)

[3] A. Pradhan, S.K. Bisoy, S Kautish, M.B. Jasser, and A.W. Mohamed, “Intelligent Decision-Making of Load Balancing Using Deep Reinforcement Learning and Parallel PSO in Cloud Environment,” *IEEE Access*, vol. 10, no. 1, pp. 76939-76952. Jul. 2022. Article (CrossRef Link).

[4] Z. Movahedi, and B. Defude, “An efficient population-based multi-objective task scheduling approach in fog computing systems,” *Journal of Cloud Computing*, vol. 10, no. 1, pp. 1-31. Dec. 2021. Article (CrossRef Link)

[5] M. Zahra, and D Bruno, A mohammad Hosseininia, “An efficient population-based multi-objective task scheduling approach in fog computing systems,” *Journal of Cloud Computing*, vol. 10, no. 1, pp. 1-1, Dec. 2021. Article (CrossRef Link)

[6] F.M. Talaat, H.A. Ali, M.S. Saraya, and A.I. Saleh, “Effective scheduling algorithmm for load balancing in fog environment using CNN and MPSO,” *Knowledge and Information Systems,* vol. 64, no. 3, pp. 773-797. Mar. 2022. Article (CrossRef Link)

[7] S. Tuli, G. Casale, and N.R. Jennings, “GOSH: Task scheduling using deep surrogate models in fog computing environments,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 2821-2833. Dec. 2021. Article (CrossRef Link)

[8] Y.K.Teoh, S.S Gill, and A.K. Parlikad, “IoT and fog computing based predictive maintenance model for effective asset management in industry 4.0 using machine learning,” IEEE Internet of Things Journal. Vol. 10, no. 3, pp. 2087-2094, Jan. 2021. Article (CrossRef Link)

[9] O. Sharma, G. Rathee, and J. Kerrache Herrera-Tapia, “Two-Stage Optimal Task Scheduling for Smart Home Environment Using Fog Computing Infrastructures,” *Applied Sciences*, vol. 13, no. 5, p. 2939. C.A Feb. 2023. Article (CrossRef Link)

[10] Z. Shi, H. Li, Q. Cao, Z. Wang, and M. Cheng, “A material decomposition method for dual‐energy CT via dual interactive Wasserstein generative adversarial networks,” Medical Physics, vol. 48, no. 6, pp. 2891-2905. Jun. 2021. Article (CrossRef Link)

[11] N. Krishnaraj, A. Daniel, K. Saini, and K. Bellam, “EDGE/FOG computing paradigm: Concept, platforms and toolchains”. In Advances in Computers (Vol. 127, no. 1, pp. 413-436). Elsevier, Jan. 2022. Article (CrossRef Link)

[12] T. Tuncer, S Dogan, and F. Ozyurt, “An automated Residual Exemplar Local Binary Pattern and iterative ReliefF based COVID-19 detection method using chest X-ray image,” Chemometrics and Intelligent Laboratory Systems, vol. 203, no. 1, pp. 104054. Aug. 2020. Article (CrossRef Link)

[13] J.O. Agushaka, A.E. Ezugwu, O.N. Olaide, O. Akinola, R.A. Zitar, and L. Abualigah, “Improved Dwarf Mongoose Optimization for Constrained Engineering Design Problems,” Journal of Bionic Engineering, vol. 1, no. 1, pp. 1-33. Mar. 2022. Article (CrossRef Link)

Ravi Kumar Suggala, He is currently working as Assistant Professor, Department of Information Technology, Shri Vishnu Engineering College for women, Bhimavaram, Andhra Pradesh, India. His research interests include machine learning, deep learning, and cloud technology.

Suma Bharathi. M, She is currently working as Assistant Professor, Department of Information Technology, Shri Vishnu Engineering College for women, Bhimavaram, Andhra Pradesh, India. Her research interests include Machine learning, deep learning, Grid computing and soft computing.

P.L.V.D. Ravi Kumar, he is currently working as Assistant Professor, Department of Information Technology, Shri Vishnu Engineering College for women, Bhimavaram, Andhra Pradesh, India. His research interests include Machine learning, deep learning, Grid computing and soft computing.

NVS Pavan Kumar Nidumolu is currently working as an Associate Professor in the Department of Computer Science and Engineering at Koneru Lakshmiah Education Foundation, Andhra Pradesh, India. He Completed his MCA degree from Andhra University, M.Tech degree in Computer Science and Engineering from JNTUK and obtained his doctoral degree in Department of Computer Science and Engineering. With over 20+ years of teaching experience he is well known for his administrative role in various positions. He is active and resourceful in various social bodies. He has over 15 of publication in various national and international reputed journals. He has received many awards for his achievements in administration. His area of interest is Machine learning and Data Science.