Enabling Better QoS in Fog Computing through Adaptive Clustering and Load Balancing Strategy

[1]Saurabh ,[2]Aanjey Mani Tripathi*,[3]Rajesh Kumar Dhanaraj

[1]Research Scholar, School of Computing Science and Engineering, Galgotias University, Greater Noida, India

[2]Associate Professor, School of Computing Science and Engineering, Galgotias University, Greater Noida, India

[3]Professor, Symbiosis International University, Pune, India

Corresponding Author: Aanjey Mani Tripathi,

Abstract: Quality of Service standards have been upheld by the researchers using several methods. Major thing in the cloud traffic is the job scheduling, which is disrupted due to the increased service delay brought on by the heavy traffic. This results in an imbalanced strain on the fog environment and degrade the network performance. The suggested approach makes use of a novel model which deals with the distribution of the requests by the fog nodes i.e., load, and do the task scheduling on the cloud-based servers. In the work, authors have gone through the various existing algorithms and the implemented models for the task scheduling, and how the task scheduling is impactful in the load balancing of the fog nodes requests. The proposed system is a novel optimal load balancing method in the cloud which inspired from the various existing optimization techniques. The proposed work is compared with the existing models, and performed better in terms of Delay analysis, and the load balancing parameters like average waiting time, task carrying time and average task completion time.

Keywords: *Cloud computing, Fog nodes, Load balancing, Optimization, QoS, Resource Management.*

1. Introduction

Load balancing is a critical aspect of resource management in fog computing since it ensures the efficient distribution of resources among numerous jobs and applications. Fog computing, a subset of cloud computing, has emerged as a solution to address the increasing need for real-time data processing and low latency services. It operates at the network's edge to meet these demands. Fog computing facilitates the localization of data processing and storage near end devices, hence mitigating latency and yielding a system that is characterised by enhanced efficiency and responsiveness. Considering the increasing prevalence of interconnected devices and applications, the efficient allocation and utilisation of resources play a crucial role in maintaining a high level of Quality of Service (QoS) within fog computing settings.

1.1 Layout

The typical load balancing strategies have challenges in managing the dynamic and heterogeneous nature of fog computing systems [1]. The assurance of Quality of Service (QoS) in fog computing systems is greatly impeded by this limitation. Over the course of the last twenty years, a significant number of researchers have been actively involved in the topic of fog computing [2][3]. There exists a vast array of scholarly publications that aim to improve performance and investigate the functioning of the fog computing environment. However, there is a requirement for enhancement in the field to elevate the standards of Quality of Service (QoS). The study contributes valuable insights into various algorithms and models, including a comprehensive examination of the suggested method and technique aimed at enhancing adaptability in fog environments.

1.2 Challenges

Ensuring QoS in fog computing situations is a big concern because of the dynamic and varied nature of the system. An growth in connected devices and apps is straining the scarce resources at the network's edge, which could lead to a worsening in QoS. Fog computing settings are not well suited for traditional load balancing strategies. These algorithms frequently employ predefined criteria and fail to account for real-time changes in resource utilisation and application needs. Reduced QoS and inefficient resource allocation are the results of this.

According to the authors, the issue they are trying to solve in their research article is the dearth of a load balancing method that is efficient and flexible and can increase QoS in fog computing environments. To efficiently balance resources among multiple tasks & applications in real-time while considering the changing resource utilisation and application requirements, the authors intend to offer a new method. The following are some of the obstacles the authors have encountered when trying to implement load-balancing in fog computing environments:

Fog computing systems are made up of a variety of heterogeneous resources, all of which have different capabilities & resource limitations.

i. Workload variations: In fog computing settings, the workload might change quickly, making it challenging to decide on the best resource allocation in the moment.[5]
ii. Fog computing settings are characterised by low bandwidth, which can influence the effectiveness of the load-balancing algorithms and the system's overall QoS.[6-7]
iii. Requirements for latency: Low latency services, which fog computing systems are designed to enable, necessitate quick and effective methods for load balancing to keep everything running smoothly are assigned and used to their full potential. [8-10]

The authors aimed to work on the above-mentioned points to derive the hybrid novel approach in fog environment.

2. Literature Review

The major areas and the existing models in a fog environment are covered in this section.

2.1 Fog Computing Challenges

Due to the development of IoT and the rising need for real-time data processing, fog computing has seen a considerable increase in interest in recent years. It is viewed as a viable option for a variety of IoT applications, such as driverless vehicles, smart cities, and industrial automation [9]. Fog computing research is primarily concerned with solving problems related to resource management, security, scalability, and interoperability [10].

Despite the potential benefits of fog computing, there are still several issues that need to be resolved before it can be a real alternative to traditional cloud computing. The following are the main difficulties with fog computing:

i. Heterogeneity of edge devices and gateways: It can be challenging to create apps and services that work on all edge devices and gateways due to the heterogeneity of these devices' hardware and software capabilities. Additionally, the processing speed, memory, and storage of edge devices may be constrained, which may impair their ability to carry out complicated computational activities. Effective deployment and management of fog computing applications and services is difficult due to these constraints [11].
ii. It might be challenging to create apps and services that work on all edge devices and gateways due to the different hardware and software capabilities of these devices. Additionally, the processing speed, memory, and storage of edge devices may be constrained, which may impair their ability to carry out complicated computational activities. Effective deployment and management of fog computing applications and services is difficult due to these constraints [11].

iii. Security: Since they are frequently installed in open, unprotected environments, edge devices and gateways might be exposed to assaults. It is crucial to protect data integrity in fog computing because hostile actors may intercept data sent between edge devices and gateways [11][12].

iv. Scalability: It becomes more challenging to manage and coordinate the actions of an increasing number of edge devices and gateways, which can lead to lower fog environment performance and reliability. It is also challenging to maintain the scalability of fog computing due to its dynamic nature, where devices can appear and disappear as needed [11][13].

v. Lack of standardization: The lack of recognised standards for fog computing design makes it challenging for different fog computing implementations to work together. The requirement to create different implementations for various devices and platforms may result in additional costs for developers and organisations. [11].

With all the difficulties in a foggy environment, load balancing is currently a major concern.

2.2 QoS in Fog Computing

Quality of Service (QoS) in fog computing is determined by various considerations [20], including:

i. Bandwidth: The rate at which data is transmitted over a given period.

ii. Reliability: A measure of how likely it is that data will be delivered without errors.

iii. Security: The steps taken to guard against unauthorised access, modification, and theft of data and systems.

iv. Scalability: It refers to a system's capacity to accommodate rising service and resource demands.

v. Cost: The financial and material expenses incurred in delivering and sustaining a service.

vi. Energy Efficiency: It is the capacity of a technique to save energy although still delivering the desired results.

Various QoS-related difficulties in fog computing, such as bandwidth restrictions, resource limitations, and latency requirements, were surveyed by S. Li et al. [35], who also offered an overview of current solutions. Traffic control, resource allocation, and scheduling are just a few of the QoS metrics and methods used in fog computing. The paper gave a thorough summary of the most recent studies in this area and highlighted how fog computing could improve QoS. Authors discussed QoS optimisation for mobile fog computing systems in another research work [36]. In order to provide effective and scalable QoS optimisation, the authors suggest a novel mobile fog computing architecture that combines cloud computing, edge computing, and mobile computing. The authors talked about the many difficulties with QoS optimisation in mobile fog computing systems, such as low bandwidth and processing resources, as well as excessive latency and energy consumption. They suggested a system that balances resource limitations and QoS requirements using dynamic resource allocation and job offloading. Figure 1 depicts the fog architecture's top view layout.

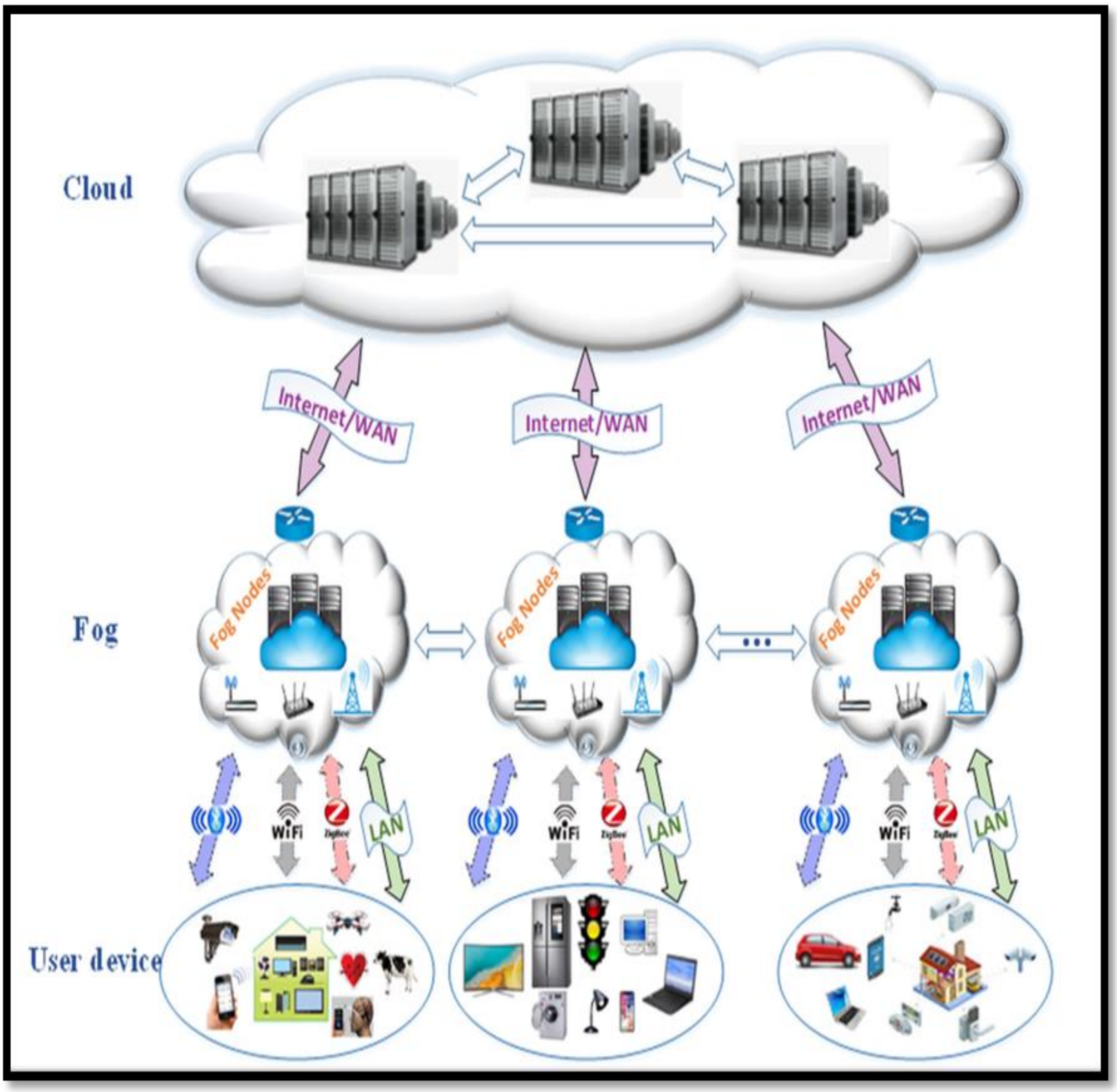

Fig 1. Fog Computing Architecture [36]

A deep reinforcement learning technique is employed in a proposed model for resource allocation in an Internet of Things (IoT) and fog computing environment, considering the Quality of Service (QoS) [37] [39]. The authors want to tackle the challenge of achieving a harmonious equilibrium between resource utilisation and quality of service (QoS) fulfilment inside the IoT-fog computing ecosystem. To attain an equilibrium between the utilisation of resources and the satisfaction of Quality of Service (QoS), the proposed methodology utilises deep reinforcement learning algorithms to dynamically allocate resources and determine resource allocation decisions. The results indicate that the proposed technique exhibits superior performance in terms of resource utilisation and satisfaction with quality of service (QoS) compared to conventional methods. Table 1 presents an overview of the examined articles, taking into consideration the working approach as well as the advantages and disadvantages of the models mentioned.

Table 1: Literature Work

Authors and paper	Methodology Adopted	Merits	Demerits
Wu. Z et al. [15]	GNN	multivariate time series is used	Only 3 parameters taken
Vaswani A, et al. [16]	Survey work	Various methods discussed	Only survey paper
Habibi et al. [20]	Survey work	Considered all parameters for networking	Model not implemented, only discussed the factors
Su Hu et al. [21]	CNN	Extracted textual data is used to make predictions about the video's subject matter.	Only related to video domain
Li. M. et al. [22]	TrGNN	Focuses on the traffic flow and congestion	Limited to the congestion method
Sriraghavendra, M. et al. [30]	DoSP, GA	Improvement in QoS	Delay
Nair, A.R., and Tanwar S. [32]	Survey paper	QoS was thoroughly covered, and several approaches were explored.	Only survey paper
Syed Mujtiba et al. [34]	Flamingo Search with GA	Fog Task scheduling Algorithm	Optimization algorithms
S. Tewari, et al. [36]	Authentication methods	Used the security enhancement mechanism in paper	Delay
Jain, V. et al. [37]	MDP	QoS is better and improved	Delay

To ensure that certain QoS standards are met, load balancing might give particular types of data priority and extra resources. The application's unique requirements as well as the system architecture influence the choice of load balancing technique.

2.3 Load balancing difficulties with fog computing

There are several obstacles associated with load balancing in fog computing:[22][23]

1. Heterogeneity: The resources and capacities of fog nodes may vary, making it challenging to distribute the task equally.

2. Dynamic Environment: It might be challenging to maintain an effective load balancing technique since the quantity and the fog's topology, including the locations of its devices and nodes, can shift rapidly.

3. Resource Constraints: Fog nodes' limited processing capabilities may result from their limited resource availability, making it difficult to properly distribute the workload among them.

4. Real-time Requirements: It is challenging to balance the workload in a way that meets real-time requirements for some fog computing applications since these needs must be met.

5. Security Concerns: Algorithms for load balancing could be attacked, hence caution must be exercised to prevent the introduction of security flaws.

To address the dynamic and resource-constrained characteristics of the fog environment, load balancing techniques in fog computing must be carefully designed and optimised. In order to make sure that the resources that are available are used effectively, this may need the usage of adaptive algorithms that can adapt to changing circumstances. Fog computing research is geared on solving problems with resource management, security, scalability, and interoperability. Load balancing is the element that affects all the parameters in addition to the ones mentioned.

2.5. Load Balancing in Fog Computing

The evaluation of various load balancing techniques for fog computing often entails comparing them using a variety of measures. The following are some of the popular load balancing methods [24]:

i. Static algorithms: In contexts involving fog computing, these methods disperse the workload by making use of previously defined rules or regulations. In the work, instead rely on pre-defined factors to determine how the workload should be allocated; the parameters include resource utilisation and network congestion.[24] The examples of frequent static-algorithms for fog computing are Round Robin [25], Least Connected [26], and Weighted Round Robin [27].

ii. Dynamic algorithms: In systems that utilise fog computing, these methods disperse the workload by making use of real-time information, such as the utilisation of resources and the congestion of networks. These methods may be more complicated and expensive to compute, but they are more efficient in terms of ensuring quality of service [24]. The common dynamic algorithms are Threshold-based [28], Prediction-based [29] and Game-theoretic [30] [38].

iii. Hybrid algorithms: These algorithms strike a good mix between the ease of use and efficacy of static and dynamic algorithms. Hybrid algorithms are developed to overcome the drawbacks of both classical and modern approaches. By fusing the straightforwardness and efficacy of static algorithms with the data-driven decision making of dynamic algorithms in real time, these algorithms hope to deliver excellent QoS [24]. Fog computing frequently makes use of hybrid algorithms, such as Hybrid Round Robin [31], Hybrid Threshold-based [32] and Hybrid Prediction-based [33].

QoS in fog computing environments can be improved with the help of hybrid algorithms, which combine the best features of static algorithms (which are easy to implement and maintain) and dynamic algorithms (which make decisions in real time based on data collected in the field). These algorithms attempt to deliver the best possible quality of service by minimising the drawbacks of static and dynamic approaches. Each algorithm has its own set of advantages and disadvantages, depending on what is needed in the fog computing setting. Static algorithms, for instance, are straightforward and effective but may not necessarily guarantee excellent quality of service. However, dynamic algorithms may be more complicated and computationally expensive, but they are more effective in guaranteeing QoS. While hybrid algorithms strike a happy medium between the two, they may still have practical limitations when it comes to dealing with complicated and ever-changing workloads.

3. Proposed Methodology

Following a thorough examination of the available literature, work on the hybrid algorithm with a focus on load-balancing parameters has been undertaken, with the desired effect of raising all QoS metrics. This proposed model's parameters were chosen with Table1's discussion and comparison of existing models in mind; the delay, a common drawback to many existing algorithms, was the primary focus.

3.1 System Model

The components of the suggested system model, to work on the fog environment is shown in the figure 2.

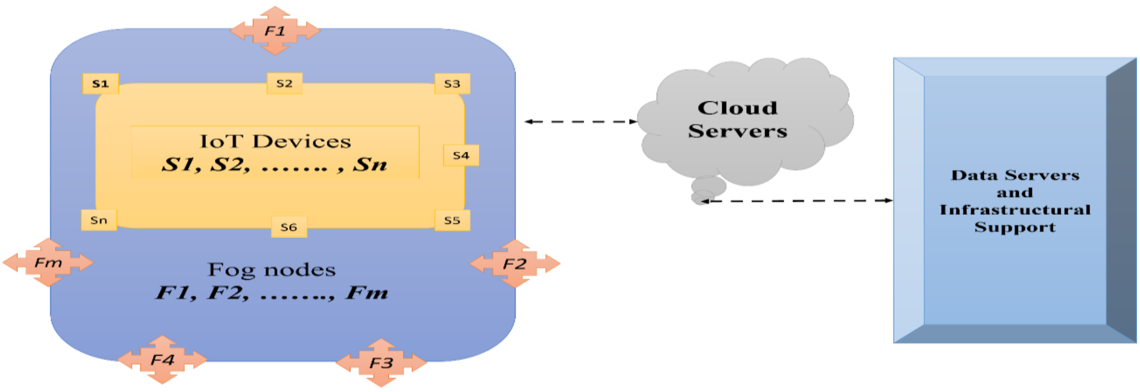

Fig 2. System Model components

Whereas the hybrid Prediction-based model in fog computing environment has a different flavor. The working modules are depicted in figure 3, and discussed as follows.

Fig 3. System Model components

3.1.1 Resource Management Unit (RMU): This part is responsible for overseeing the fog computing environment's available resources. The RMU records information about how and when resources are used, as well as how busy networks are.

3.1.2 Task Management Unit (TMU): This part oversees the tasks that must be carried out in the fog computing setting. Information about job needs, such as processing speed and data transfer capacity, is gathered by the TMU and sent on to the RMU.

3.1.3 Load Balancing Unit (LBU): The information from the RMU and TMU is sent to the Load Balancing Unit, which oversees assigning tasks. Based on the current state of the system and projected values, the LBU allocates work using the optimal set of available resources.

3.1.4 Prediction Unit (PU): Based on data from the RMU, this part predicts how resources will be used and how congested the network will be in the future. The LBU bases its decisions on predictions of resource utilisation and network congestion made by the PU.

3.1.5 Monitoring Unit (MU): This part keeps the prediction models up to date by keeping tabs on resource utilisation and network congestion. The MU reports network congestion and details on resource utilisation to the PU.

Together, the above mentioned parts of the system, analyse and estimate the performance parameters to determine which resources are most suited to carry out each activity.

3.2 Hybrid Novel Approach

In fog environment the Adaptive Algorithms are the need of an hour, which dynamically allocate the resource requests and boost the quality-of-service parameters. The algorithm keeps a close eye on the state of the system and uses both current data and forecasts to delegate work effectively. This method offers a scalable and adaptable way to enhancing quality of service in fog computing settings. The approach proposes the two main components:

i. Monitoring Resource Utilization & Network Congestion: The algorithm keeps constant tabs on how the fog computing environment is using its resources and how congested the networks are. The Prediction Unit (PU) receives data on resource utilisation and network congestion from the Monitoring Unit (MU), which performs the monitoring process.

ii. Updating Prediction Models: The PU then applies the new information to its prediction models, improving its ability to foretell resource consumption and network congestion in the future. The Load Balancing Unit (LBU) uses the modernised prediction models to allocate work effectively.

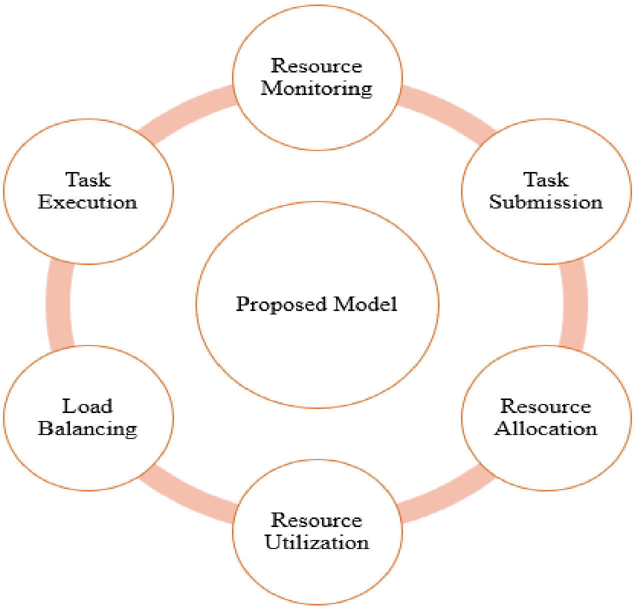

Fig 4. Modules of the Hybrid Novel Approach

When it comes to ensuring top Quality of Service (QoS) in a fog computing setting, the following phase of the Adaptive Load Balancing Algorithm, "Load Balancing," is essential. To strike a good balance between resource utilisation and network congestion, the algorithm now modifies the allocation of those resources. The programme considers both current and future data to distribute tasks to the best suitable resources.

Following are the methods to perform load balancing:

- Dynamic Resource Allocation: The algorithm makes real-time changes to the allocation of resources in response to the current state of the system and the results of the prediction models. The programme may reroute the job to a less crowded resource if network activity is heavy.
- Predictive Resource Allocation: The algorithm considers past data in order to foresee potential future outcomes, and then allocates resources accordingly. The programme considers task necessities such CPU time and data transfer rate.
- Real-time Monitoring: The system keeps tabs on things like server load and data transfer rates in order to fine-tune its forecasting abilities. The algorithm is able to adapt to changing conditions in the system and strike a good balance between resource utilisation and network congestion thanks to this constant monitoring.

To strike a good equilibrium between resource utilisation and network congestion, Algorithm 1 is derived.

***Algorithm 1: Adaptive Clustering and Load Assignment Algorithm*.**

Input: *Fog Node set Ni*

Output: *Cluster Node*

Begin

For i = 1 to N do

  Check for fog node Fi

  Assign the cluster Ci

End for

For i = 1 to N do

Check for node request Si

If (resource analyse(x) <= Threshold Value(CS))

Then (Si*RQi) is considered and Processing

Else migrate the request_query to other CS ***// Cloud Server***

End for

End

This approach achieves improved quality of clustering and the task assignment.

4. Implementation and Results

4.1 Simulation

Using load balancing and hybrid predictive scheduling, this study offered a method to enhance quality of service with fog computing. The strategy implemented in the JAVA Cloud simulation running at 2.6 GHz on an Intel i7 processor with 8 GB of RAM. The implementation values are shown in Table 2.

Table 2: Parameters and the values used in simulation.

Parameter	Value
fog servers count	8
IoT devices count	20
Fog node CPU frequency	2.9–4.2 GHz
IoT CPU frequency	16–84 MHz
Task size	400–800 KB
Task CPU cycle	1500–2500 M cycle
Bandwidth	10–20 MHz
Noise power	–100 dBm
Transmit power	60mW
Task deadline	2–0 s
Hidden layer	2
Type of layer	Fully Connected
Optimizer	Adam
Activation function	ReLu
Mini batch	64
Learning rate	0.001

4.2 Performance Evaluation and Comparison

i. Delay (D): This metric quantifies the time taken for a task to complete, starting from its initiation until its completion. This includes the processing time, response time, queuing delay and the other factors. It provides an indication of the responsiveness and efficiency of the system. In comparison to the existing systems, the proposed results are shown in Figure 5.

ii. Average Task Carrying Time (Ⱥ): Whenever the task is assigned, the time taken by the server from the task initialization to the task assigned to the respective responding node. In comparison to the existing systems, the proposed results are shown in Figure 6.

iii. Average Task Waiting Time (γ): The time taken for the task to wait from task assignment to the task completion. Basically, it is computed as the pure burst time subtracted from the task completion time. In comparison to the existing systems, the proposed results are shown in Figure 7.

iv. Average Task Completion time (£): The total time taken by the task from the time of task creation to the task completion, including delay, waiting and the task burst time. In comparison to the existing systems, the proposed results are shown in Figure 8.

The glimpse of the above-mentioned parameters are shown in table 5.

Table 3. The comparison of existing algorithms with proposed algorithm.

Parameters	Previous Algorithm [18]	Previous Algorithm [19]	Previous Algorithm [20]	DMOE [21]	Proposed Algorithm
D	21	19.23	17.8	11.2	6.9
Ꞁ	148.76	139.4620	89.244	19.2871	15.412
γ	2.133	2.0836	1.86777	2.0140	1.4915
£	261.06	251.4297	158.25	133.2391	119.212

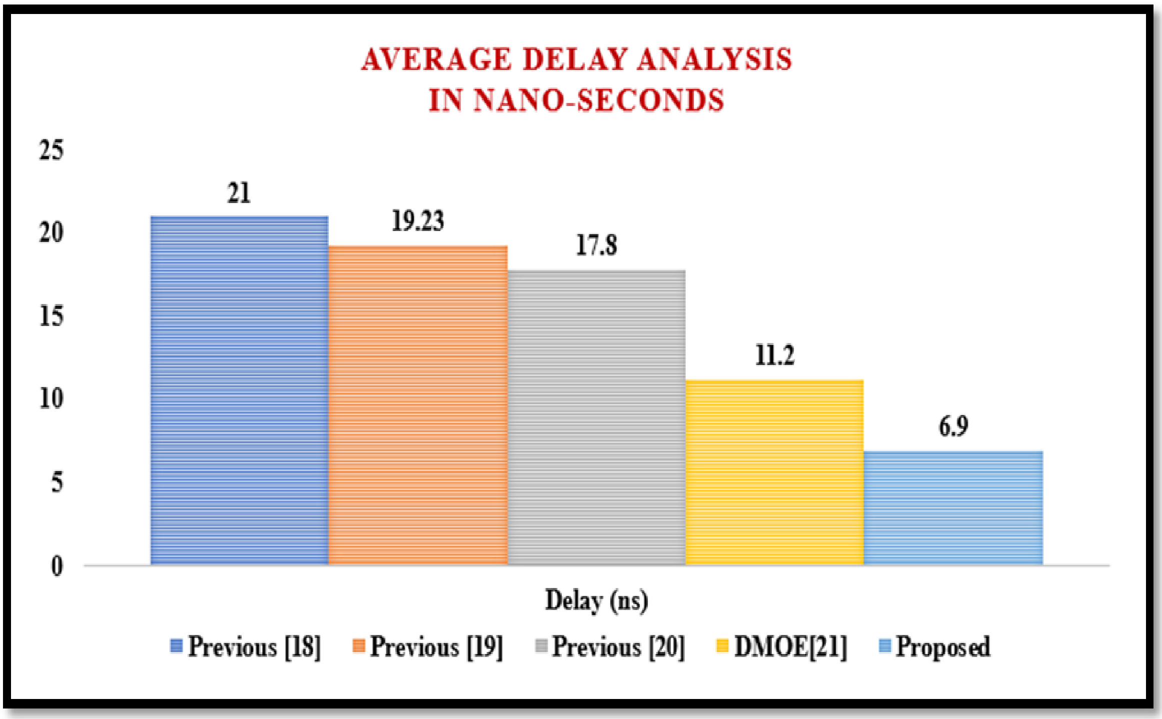

Fig 5. Delay Analysis

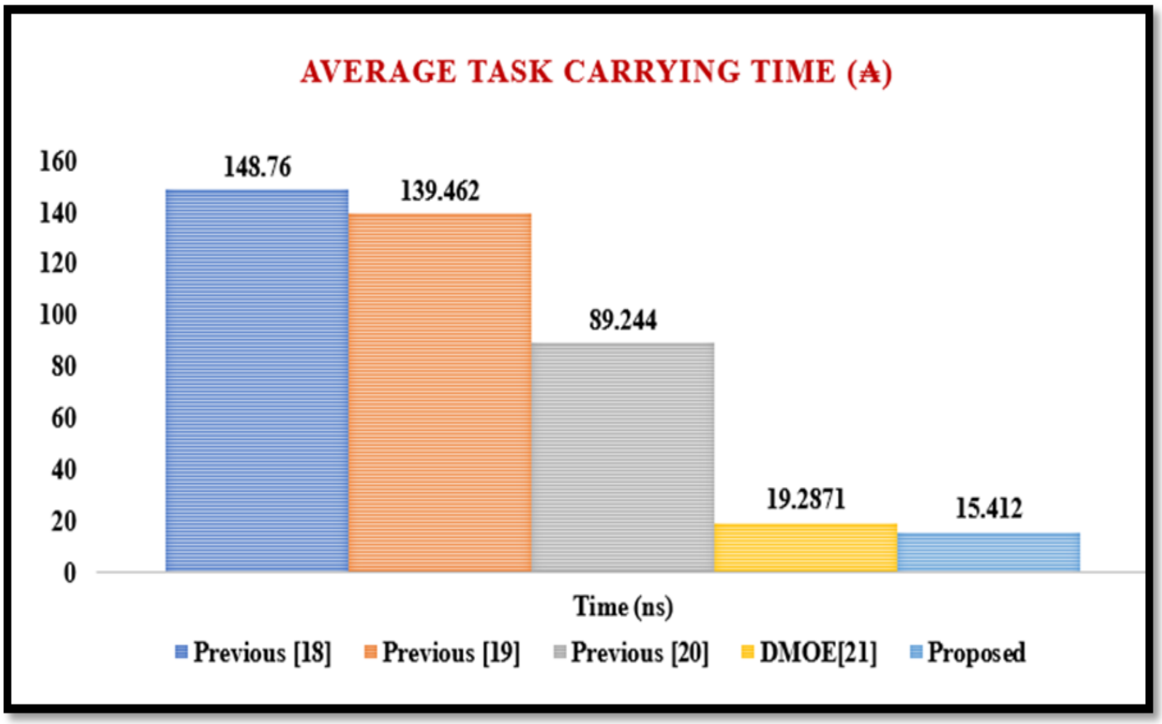

Fig 6: Average task carrying time (₳) on various algorithms.

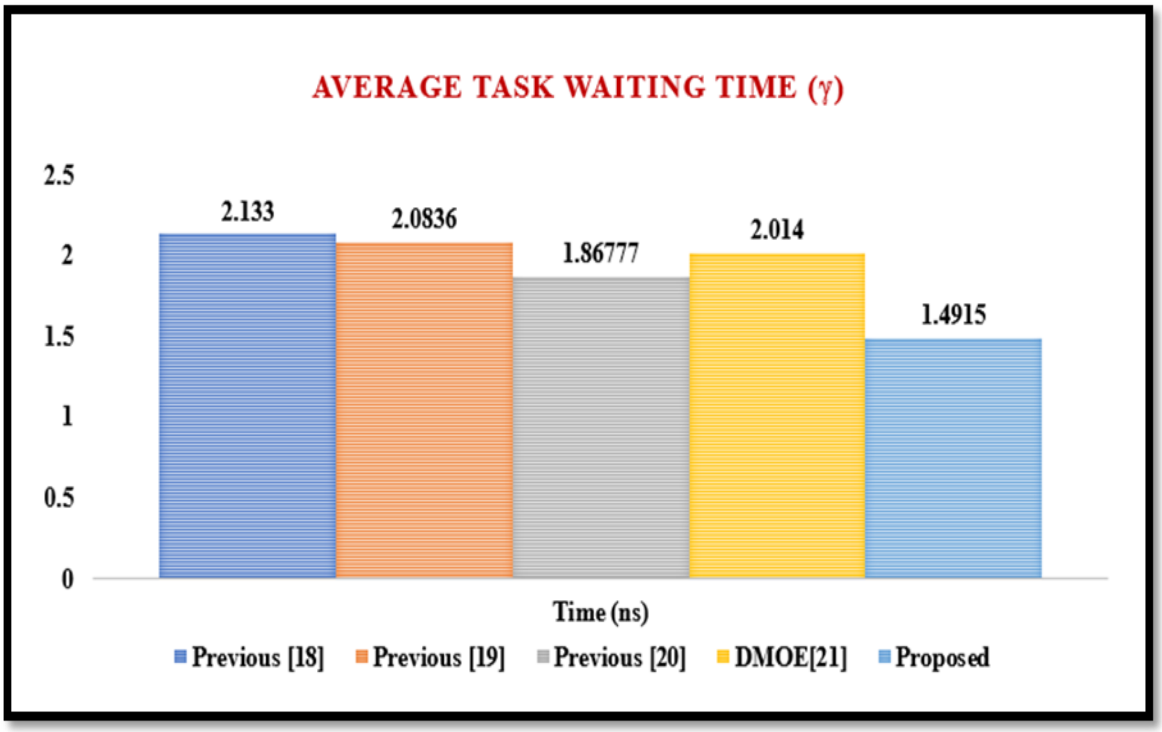

Fig 7: Average task waiting time (γ) on various algorithms.

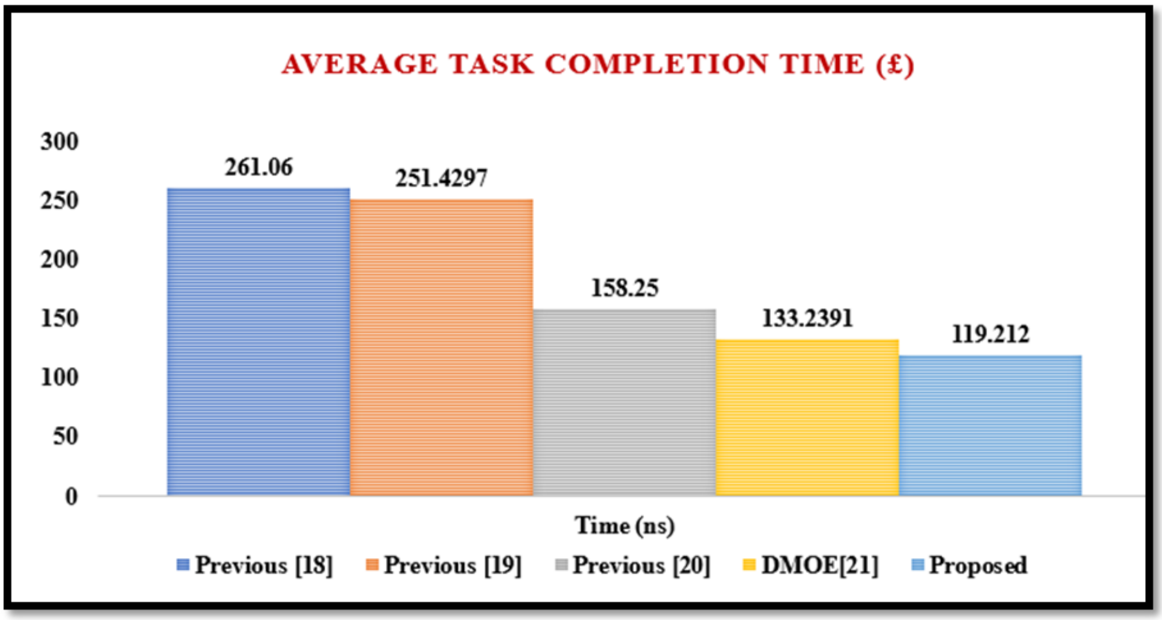

Fig 8: Average task completion time (£) on various algorithms

5. Conclusion

This work investigates the issue of ensuring QoS for applications and services in fog computing environments, which necessitates efficient load balancing. This study demonstrates a hybrid method for load balancing among heterogeneous servers. The suggested algorithm's superior performance is demonstrated by its implementation and comparison to previously developed models and previously implemented methods. The load is distributed more evenly, and other QoS parameters like delay analysis, waiting time, task carrying time, and the task completion time. Proposed model is the improved version, which is the need of an hour. In addition, when mobility is introduced to fog nodes, the same technique can be used for all the elements involved. Many of the ensuing algorithms may use hybrid models, but progress must be made towards a single strategy that works with either of topology for fog-servers and fog-nodes. As shown and detailed in the study, the proposed one currently performs better across the board.

Declarations

Ethical Approval: The content and the data used is upto the authors of the manuscript. No other entity is involved for the further ethical approval. The content is the onus of authors only.

Competing interests: The study and the paper are not supported by any financial body. The authors have not taken any support from any organization. A work is solely done by the authors.

Authors' contributions: The authors of the manuscript are agreed to share the research work and contributed equally to the paper.

Funding: No funding agency is there.

Availability of data and materials: The data used for the implementation of the results in the paper is the self-generated data on the cloud sim.

References

[1] Omoniwa, Babatunji, Riaz Hussain, Muhammad Awais Javed, Safdar Hussain Bouk, and Shahzad A. Malik. "Fog/edge computing based IoT (FECIoT): Architecture, applications, and research issues." *IEEE Internet of Things Journal* 6, no. 3 (2018): 4118-4149.

[2] Brogi, Antonio, and Stefano Forti. "QoS-aware deployment of IoT applications through the fog." *IEEE Internet of Things Journal* 4, no. 5 (2017): 1185-1192.

[3] Ni, Jianbing, Kuan Zhang, Xiaodong Lin, and Xuemin Shen. "Securing fog computing for internet of things applications: Challenges and solutions." *IEEE Communications Surveys & Tutorials* 20, no. 1 (2017): 601-628.

[4] Hussein, Mohamed K., and Mohamed H. Mousa. "Efficient task offloading for iot-based applications in fog computing using ant colony optimization." *IEEE Access* 8 (2020): 37191-37201.

[5] Nashaat, Heba, Eman Ahmed, and Rawya Rizk. "IoT Application Placement Algorithm Based on Multi-Dimensional QoS Prioritization Model in Fog Computing Environment." *IEEE Access* 8 (2020): 111253-111264.

[6] Mahmud, Redowan, Satish Narayana Srirama, Kotagiri Ramamohanarao, and Rajkumar Buyya. "Quality of Service (QoS)-aware placement of applications in Fog computing environments." *Journal of Parallel and Distributed Computing* 132 (2019): 190-203.

[7] Goudarzi, Mohammad, Huaming Wu, Marimuthu S. Palaniswami, and Rajkumar Buyya. "An application placement technique for concurrent IoT applications in edge and fog computing environments." *IEEE Transactions on Mobile Computing* (2020).

[8] Lera, Isaac, Carlos Guerrero, and Carlos Juiz. "Availability-aware service placement policy in fog computing based on graph partitions." *IEEE Internet of Things Journal* 6, no. 2 (2018): 3641-3651.

[9] Badawy, Mahmoud M., Zainab H. Ali, and Hesham A. Ali. "Qos provisioning framework for service-oriented internet of things (iot)." *Cluster Computing* (2019): 1-17.

[10] Jamil, Bushra, Mohammad Shojafar, Israr Ahmed, Atta Ullah, Kashif Munir, and Humaira Ijaz. "A job scheduling algorithm for delay and performance optimization in fog computing." Concurrency and Computation: Practice and Experience 32, no. 7 (2020): e5581.

[11] Deepa, N., and P. Pandiaraja. "E health care data privacy preserving efficient file retrieval from the cloud service provider using attribute based file encryption." Journal of Ambient Intelligence and Humanized Computing (2020): 1-11.

[12] Chen, Huankai, and Frank Z. Wang. "Spark on entropy: A reliable & efficient scheduler for low-latency parallel jobs in heterogeneous cloud" In 2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops), pp. 708-713, IEEE, 2015.

[13] Chen, W., Chen, L., Xie, Y., Cao, W., Gao, Y., Feng, X.: Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3529–3536 (2020)

[14] Ghobaei-Arani M, Shahidinejad A. A cost-efficient IoT service placement approach using whale optimization algorithm in fog computing environment. Expert Systems with Applications. 2022 Aug 15;200:117012.

[15] Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: Connecting the dots: Multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 753–763 (2020)

[16] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems 30 (2017)

[17] Du, S., Li, T., Gong, X., Horng, S.-J.: A hybrid method for traffic flow forecasting using multimodal deep learning. arXiv preprint arXiv:1803.02099 (2018)

[18] C. Wang, S. Li, T. Cheng, et al., A construction of smart city evaluation system based on cloud computing platform, Evol. Intell. 13 (10) (2019) 119–129.

[19] K. Lee, K. Kim, A performance evaluation of a geo-spatial image processing service based on open source paas cloud computing using cloud foundry on openstack, Remote Sens. 10 (8) (2018) 1274.

[20] Habibi, Pooyan, Mohammad Farhoudi, Sepehr Kazemian, Siavash Khorsandi, and Alberto Leon-Garcia. "Fog computing: a comprehensive architectural survey." IEEE Access 8 (2020): 69105-69133.

[21] Su Hu, Yinhao Xiao, "Design of cloud computing task offloading algorithm based on dynamic multi-objective evolution", Future Generation Computer Systems, Volume 122, 2021, Pages 144-148, ISSN 0167-739X, https://doi.org/10.1016/j.future.2021.04.002.

[22] Li, M., Tong, P., Li, M., Jin, Z., Huang, J., Hua, X.-S.: Traffic flow prediction with vehicle trajectories. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 294–302 (2021) Springer Nature 2021

[23] Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: A graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)

[24] Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875 (2017)

[25] Tian, C., Chan, W.K.: Spatial-temporal attention wavenet: A deep learning framework for traffic prediction considering spatial-temporal dependencies. IET Intelligent Transport Systems 15(4), 549–561 (2021)

[26] Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatialtemporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)

[27] Kong, X., Zhang, J., Wei, X., Xing, W., Lu, W.: Adaptive spatial-temporal graph attention networks for traffic flow forecasting. Applied Intelligence 52(4), 4300–4316 (2022)

[28] Oreshkin, B.N., Amini, A., Coyle, L., Coates, M.: Fc-gaga: Fully connected gated graph architecture for spatio-temporal traffic forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 9233–9241 (2021)

[29] Pan, Z., Zhang, W., Liang, Y., Zhang, W., Zheng, Y.: Spatio-temporal meta learning for urban traffic prediction. IEEE Transactions on Knowledge and Data Engineering PP(99), 1–1 (2020)

[30] Sriraghavendra, M., Chawla, P., Wu, H., Gill, S.S., Buyya, R. (2022). DoSP: A Deadline-Aware Dynamic Service Placement Algorithm for Workflow-Oriented IoT Applications in Fog-Cloud Computing Environments. In: Tiwari, R., Mittal, M., Goyal, L.M. (eds) Energy Conservation Solutions for Fog-Edge Computing Paradigms. Lecture Notes on Data Engineering and Communications Technologies, vol 74. Springer, Singapore. https://doi.org/10.1007/978-981-16-3448-2_2

[31] X. Huang, J. Wang, Y. Lan, C. Jiang, and X. Yuan, "MD-GCN: A Multi-Scale Temporal Dual Graph Convolution Network for Traffic Flow Prediction," Sensors, vol. 23, no. 2, p. 841, Jan. 2023, doi: 10.3390/s23020841. [Online]. Available: http://dx.doi.org/10.3390/s23020841

[32] Nair, A.R., Tanwar, S. (2021). Fog Computing Architectures and Frameworks for Healthcare 4.0. In: Tanwar, S. (eds) Fog Computing for Healthcare 4.0 Environments. Signals and Communication Technology. Springer, Cham. https://doi.org/10.1007/978-3-030-46197-3_3

[33] Deng, R., Lu, R., Lai, C., & Luan, T. H. (2015). Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing. In 2015 IEEE International Conference on Communications (ICC) (pp. 3909–3914). Piscataway, NJ: IEEE.

[34] Syed Mujtiba Hussain, Gh Rasool Begh, "Hybrid heuristic algorithm for cost-efficient QoS aware task scheduling in fog–cloud environment", Journal of Computational Science, Volume 64, 2022, 101828, ISSN 1877-7503, https://doi.org/10.1016/j.jocs.2022.101828.

[35] M. Mukherjee, L. Shu and D. Wang, "Survey of Fog Computing: Fundamental, Network Applications, and Research Challenges," in IEEE Communications Surveys & Tutorials, vol. 20, no. 3, pp. 1826-1857, thirdquarter 2018, doi: 10.1109/COMST.2018.2814571.

[36] Suchita Tewari, Naveen Tewari, Mukesh Joshi, Utilizing Technology and Management of Fog Computing, Cyber Technologies and Emerging Sciences, 10.1007/978-981-19-2538-2_49, (477-483), (2023).

[37] Jain, V., Kumar, B. QoS-Aware Task Offloading in Fog Environment Using Multi-agent Deep Reinforcement Learning. J Netw Syst Manage 31, 7 (2023). https://doi.org/10.1007/s10922-022-09696-y Saurabh, S., and R. K. Dhanaraj. "Improved QoS With Fog Computing Based on Adaptive Load Balancing Algorithm". International Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 5, May 2023, pp. 347-62, doi:10.17762/ijritcc.v11i5.6623

[38] Saurabh, Dhanaraj, R.K. Enhance QoS with fog computing based on sigmoid NN clustering and entropy-based scheduling. Multimed Tools Appl (2023). https://doi.org/10.1007/s11042-023-15685-3