A Novel Technique for well-optimized Content Replication in Fog Computing

Khushi Joshi
Department Of Information Technology
JSS Academy of Technical Education, Noida
22csds029@jssaten.ac.in

Rohini Sharma
Department of Information Technology.
JSS Academy of Technical Education, Noida
rohini.sharma@jssaten.ac.in

Abstract- A fog computing architecture processes huge amount of data by using edge devices. The cloud computing paradigm's fog layer helps to improve quality of service (QoS) in time-sensitive applications. Because devices at the edge of the cloud are very diverse and dynamic, content replication is vital. Replication is essential for consistent service delivery in fog computing. Time-sensitive requests received scant consideration in many of the early approaches. To improve fog computing replication, this research suggests a novel content replication mechanism using KNN and Naïve Bayes classification models. The recommended approach consists of two steps: The first assigns a failure risk classification to the fog servers that are available for replication. In the second phase, we suggest size-based content replication, in which the file or content is replicated on the less likely-to-fail fog server. We have evaluated the performance using metrics like utilization, throughput, and latency. The usefulness of the proposed strategy is validated by simulation results comparing it with the Replica-3 method. Comparing this to the Replica-3 technique, better resource usage, higher throughput, and lower latency are therefore attained. By allowing the system to handle more queries in parallel across multiple computers or processors, KNN's parallel processing capabilities of CPUs and GPUs ensures better resource utilization and greatly increasing throughput as compared to Naïve Bayes model. Additionally, KNN classification offers a useful data structure that significantly reduces latency by minimizing the time required to find the nearest neighbours.

Keywords: - classification, content delivery, fault tolerance, fog Computing, fog node, fog server, KNN, Naïve Bayes, real-time applications, cloud, edge, data replication;

I. INTRODUCTION

Fog computing, often known as a little cloud, is the intermediary layer between the Internet of Things and the cloud. It was created in an effort to move cloud computing closer to the locations where data is generated. Fog computing integration with the present cloud architecture enables the delivery of highly scalable and latency-aware solutions to several regions. [1][2]. Like cloud computing, fog computing provides end users with data, storage,

processing, and application functionalities. Fog computing, as opposed to the cloud, is better suited to handle the processing demands of real-time, latency-sensitive applications, most of which make use of IoT devices and sensors that are dispersed geographically as shown in fig. 1 [3][4]. The fog network uses intelligence, as illustrated in fig. 1, to decide which data to analyze on the fog layer and which requests to send to cloud computing services. Content delivery networks, which aim to enhance user experience by decreasing latency in content delivery to users, are one of the primary applications of fog computing. [5][6]. Content delivery networks replicate the most popular files over a distributed collection of caches or caching servers in order to improve delivery performance, lessen the strain on the content source server, and accelerate user response times. [7].

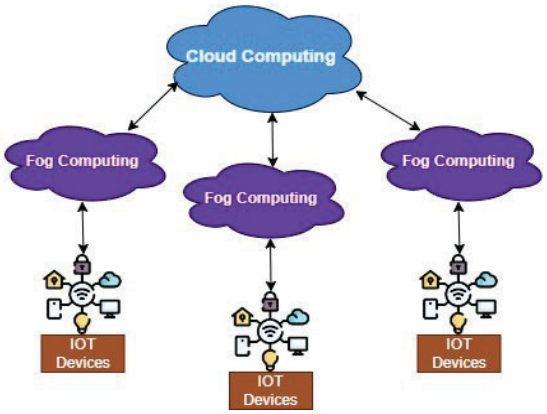


Fig. 1. Network of Fog Computing

The ability of a server to continue operating even if any of its modules fail is known as fault tolerance, or FT. There are two forms of FT: proactive and reactive. Reactive methods address errors as they arise. Replication and checkpointing are common reactive techniques [8]. Proactive methods, on the other hand, anticipate failure early on and address it [9]. Reactive techniques, on the other hand, are employed in fog computing due to their ease of use and simplicity. We compared our proposed method with the Replica-3 replication method. Since there are several important drawbacks to the Replica-3 method, such as three times the original data size was needed for high storage overhead, increased network bandwidth was used during replication, possibility of conflicting data in specific circumstances, insufficient fault tolerance for more than two node failures, resource inefficiency in comparison to more sophisticated techniques such as erasure coding, operational difficulty in preserving data balance and consistency, restricted geographic reach that might not be sufficient to fend off regional failures. Therefore, by using the reactive techniques for FT, particularly replication, the intended task keeps a copy of a file or other content on a reliable fog node or device that is less likely to fail. Also, we compared two classification models for step 1 to make step 2 more efficient: the KNN and the Naïve Bayes classification model. Their approaches are very different from one another. KNN is easy to use but computationally costly, especially when dealing with huge datasets. It classifies data points according to the majority class among their closest neighbors. Conversely, the Naive Bayes model is a probabilistic model that is quick and effective, but it is predicated on the frequently erroneous notion that characteristics are independent. Naive Bayes is favored for issues with more straight forward, probabilistic relationships, whereas KNN performs well in situations with intricate decision boundaries. The proposed procedure consists of two

2025 3rd International Conference on Disruptive Technologies (ICDT) | 979-8-3315-1958-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICDT63985.2025.10986669

Authorized licensed use limited to: UNIVERSITY OF OSLO. Downloaded on August 11,2026 at 10:07:13 UTC from IEEE Xplore. Restrictions apply.

steps. The devices are first arranged according to their failure risk using historical data from the fog server in the fog network. This classification takes into account the type of failure, the type of fog device, the documented downtime, and the anticipated use of the equipment. Using both KNN and Naïve Bayes approaches, the fog server has been split into failure groups with high and low risk. Each server has been assigned a risk rating according to the class to which it belongs. Secondly, we examine the danger label assigned to a fog server by its classification mechanism. The main points of the paper are as follows:

This study proposes a novel method to replicate files or content in order to improve the reliability of time-sensitive applications operating in fog computing. There are two stages to the suggested plan: based on information about their failure history, the proposed method's initial phase classifies the fog servers' failure risk. For the second phase, we propose a replication-based fault tolerance strategy that copies the file or content to the reliable fog server, which is less likely to fail. Throughput is one of the metrics we have used to assess performance, latency, and resource utilization. The simulation findings, which we compared with the Replica-3 approach, validate the efficacy of the proposed scheme. Still, the proposed strategy works well in comparison to another alternative.

II. Literature Survey

Content replication is one of the key components of fog computing that enhances data availability and reduces latency for end users. According to studies, effective content replication algorithms can significantly improve the performance of fog computing systems by guaranteeing that data is kept closer to its intended location. Kumari et al., for instance, emphasize the need for adaptive replication techniques that take user demand and network conditions into consideration in order to maximize resource use [7]. Sharifi et al.[8] underscore the significance of clever algorithms in managing the reproduction of content. These algorithms may dynamically adjust to real-time analytics, improving the efficiency of processes for retrieving data[9]. Itokazu and Sugawara suggest a hybrid approach that combines centralized and decentralized replication approaches to solve these issues. They claim that the performance and dependability of fog computing systems can be increased with a well-balanced approach[10]. All things considered, content replication provides benefits; yet, for optimal outcomes, implementation necessitates a thorough evaluation of the related trade-offs. One of the main elements of fog computing that improves data availability and lowers latency for end users is content replication. Effective content replication techniques can significantly enhance the performance of fog computing environments by reducing bandwidth use and access delays, as noted by Zhou et al. [11]. This is achieved by ensuring that data is retained closer to its intended location[12]. Their findings indicate that while replication might boost productivity, it also complicates data management and consistency, issues that must be appropriately resolved to avoid waste and duplication. In summary, whereas content replication in fog computing offers significant improvements in speed and user experience, it also necessitates sophisticated management strategies to reduce potential drawbacks in terms of resource consumption and data consistency [13]. It is simple to modify other mobile operating systems to match their model. An FT approach has been proposed by R. Oma et al. [14] for tree-based fog designs. They employed non-replication as well as replication tactics. The same conditions are met by several fog nodes during replication. The malfunctioning fog node gets replaced by a new one when replication is turned off. For situations including fog clouds, the authors of [15] have created a fault-tolerant scheduling method. The work [20–26] suggested single and multi-objective fault tolerance optimization strategies for fog computing, while [19] presents a reinforcement learning-based approach to increase the reliability of fog-based IoT systems. The writers have taken into account variables including price, energy, performance, and network route, as mentioned in [27].

III. Proposed Approach

Every fog server in a fog computing system is made to satisfy a user's request. If a fog server malfunctioned, obstructing or delaying the execution of user requests, customer happiness would suffer. Numerous hardware and software issues can cause a fog server to malfunction. One of the programs or processes running on the fog server is the source of the software errors among them. It is possible to predict the likelihood of a fog server failure based on historical data. This paper classifies a fog server based on its likelihood of malfunctioning by utilizing historical failure data.The aim of this work is to apply replication method is to increase the efficiency of fog computing by identifying the risk of failure of fog servers. The proposed plan includes two phases:

Step 1: The likelihood that a fog server may fail in the future is calculated by analyzing historical outage data. The dataset includes a variety of features linked to fog server failure. The type of fog device, the type of failure, the length of documented downtime, and the device's intended purpose are the characteristics that have been used in the analysis to identify hazards. A fault's cause might reveal if it is likely to happen again.For example, software problems might have been caused by a job or action that was executed on the fog server and are unlikely to recur, but physical problems might. Downtime is utilized because it appears to show the severity of the malfunction. Fig. 2 displays the probability of a fog server breakdown classified by category.

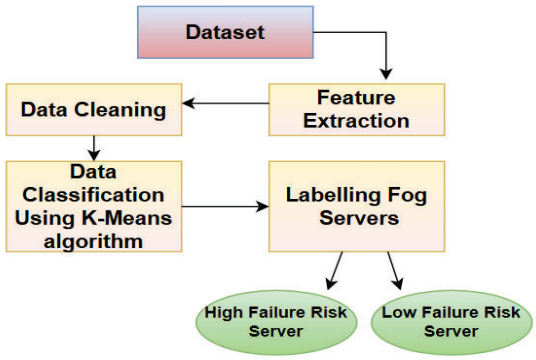


Fig. 2. Classification of Failure Risk for Fog Servers

Failure data has been categorized using the KNN and Naïve Bayes algorithm in an unsupervised manner.

Being a non-parametric technique, it does not assume anything about the distribution of the content; it can be applied to failure data from real-world sources [16, 17]. Since the data is likewise unlabelled, KNN and Naïve Bayes are utilized to divide it into two parts according to how similar their features are. Following an evaluation, the classes that had been

Authorized licensed use limited to: UNIVERSITY OF OSLO. Downloaded on August 11,2026 at 10:07:13 UTC from IEEE Xplore. Restrictions apply.

developed were classified as "high" or "low" risk. The "high/low" class fog servers have a (high/low) failure probability. The classes produced by the KNN and Naïve Bayes algorithms on the [18] dataset are displayed in Table I and Table II, together with the anticipated outcomes and an analysis of the technique's accuracy, recall, and F1-measure, respectively.

The accuracy rate is 98.81%. The discovered value is then given a label modification, either high or low, to establish the fog server's risk level.

TABLE I. A STATISTICAL ANALYSIS USING KNN

	F1 Measure	Recall	Precision
High Risk	100	100	100
Low Risk	96.4	93	100

TABLE II. A STATISTICAL ANALYSIS USING NAÏVE BAYES

	F1 Measure	Recall	Precision
High Risk	100	100	100
Low Risk	95	92	98

Step 2: In order to duplicate the contenton the dependable low risk fog server- we have employed the replication method in this phase. The goal of this work is to select low failure risk fog servers in order to efficiently and reliably maintain the file or content of a fog server with a high failure risk distributed throughout a network. First, the fog servers that have been classified as low-risk failures must be sorted. Next, we select the fog server with the low failure risk designation and see if storage is available. Copy any files or content from the high-risk fog server to the low-risk fog server if there is a high availability of storage. Fig. 3 demonstrate the use in fog computing to carry out replication. The following is a definition of the fog computing replica placement and recovery process:

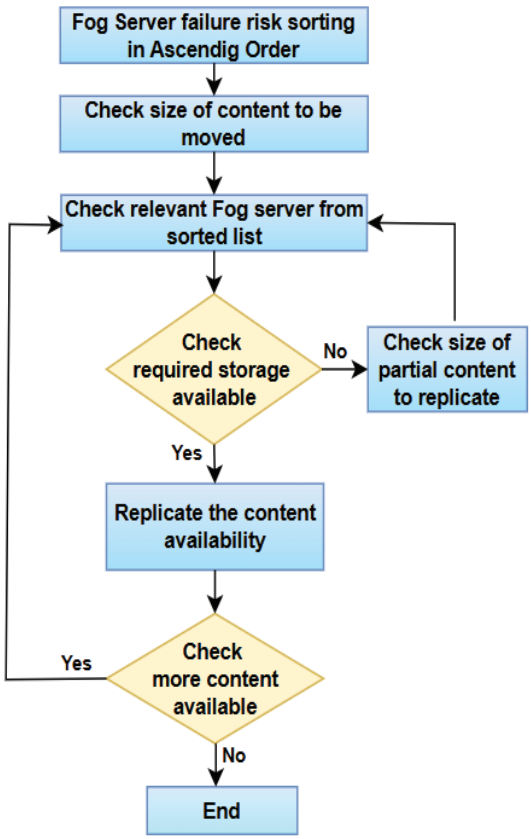


Fig. 3. Flowchart for the Suggested Technique

Algo: Contents Replication in Fog Computing

1.For each HFR server, i do;

2.For each file f in i do;

3.Select first item from the list of LFR server as s. 4.Find space that is available for replication on s. 5.Assuming that (avail_freespace[s] == 0) then

6.Keep loop going

7. Else if avail_freespace[s] ⩾ f_size / 2:

8. Set rep_s = f_size / 2

9. End if

10. While(avail_freespace [s] ≠ 0) do.

11.Move the file's replica to the LFR servers.

12. Set rep_f [i][o]=s

13. Decrement avail_freespace[s]

14. Endif

15. End else

16. Terminate for

17. Return Replica_File

IV. PERFORMANCE EVALUATION

It compares the expected approach with the Replica-3 methodology and uses simulation to assess the expected method's effectiveness using KNN and Naïve Bayes classification models.

A. Parameters

The suggested method's performance is assessed using the below parameters:

1) Resource Utilization: process of making sure cloud services is used effectively to optimize gains and reduce waste.

Resource utilization =(used time/available time)* 100 (1)

2) Throughput: the average amount of data that can genuinely go across the network in a given amount of time. Throughput = Data size / Time to transfer data (2)

3)latency: describes the lag a user encounters when sending or receiving data over the network.

Latency= Transmission Delay + Processing Delay + Propagation Delay + Queuing Delay (3)

B. Results

1) Measuring Resource Utilization: The test's objective is to determine how well fog servers are used when replicating data using the Replica-3 method and the recommended method, depicted in Fig. 4

Authorized licensed use limited to: UNIVERSITY OF OSLO. Downloaded on August 11,2026 at 10:07:13 UTC from IEEE Xplore. Restrictions apply.

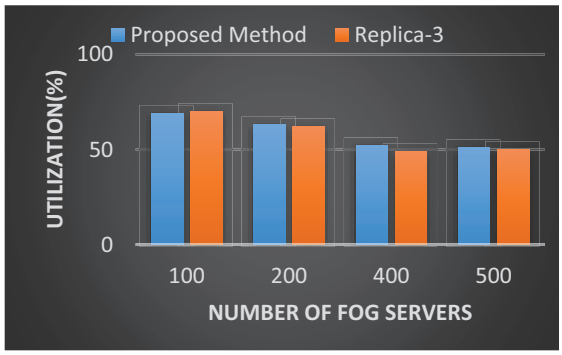


Fig. 4. Utilization comparison using KNN classification

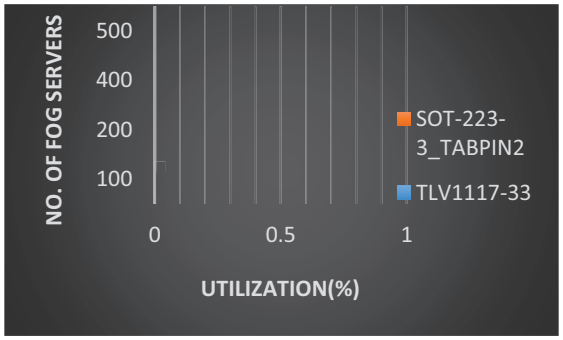


Fig. 5. Utilization comparison using Naïve Bayes classification

2) Measuring throughput: The throughput results of the recommended approach are compared with those of Replica-3 methods in terms of data transfer rate with respect to time in this study.

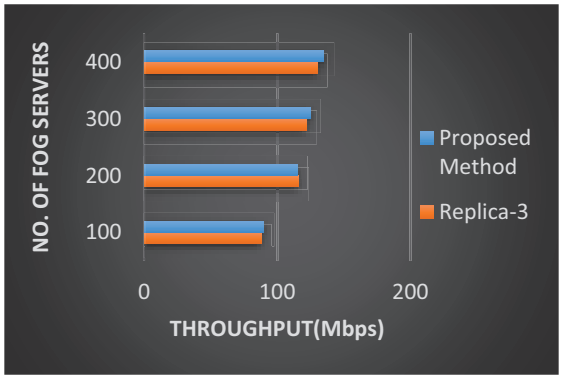


Fig. 6. Throughput comparison using KNN classification

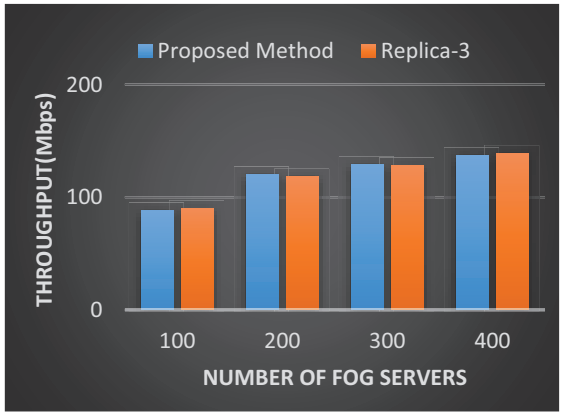


Fig. 7. Throughput comparison using Naïve Bayes classification

3) Measuring Latency: To ascertain latency, the last round of tests is conducted. We compare the proposed strategy with the Replica-3 approach in more detail. The result is displayed in Fig.6. In comparison to the Replica-3 strategy, it may be said that the recommended approach has reduced latency.

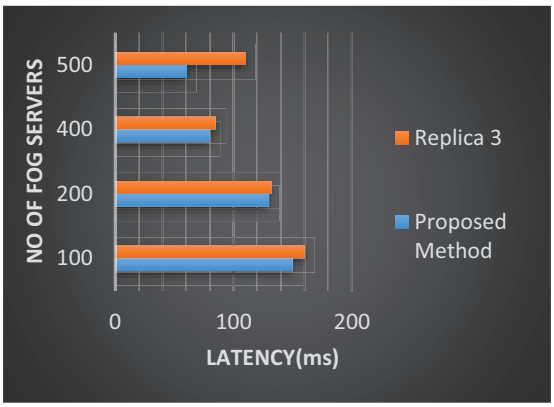


Fig. 8. Latency comparison using KNN classification

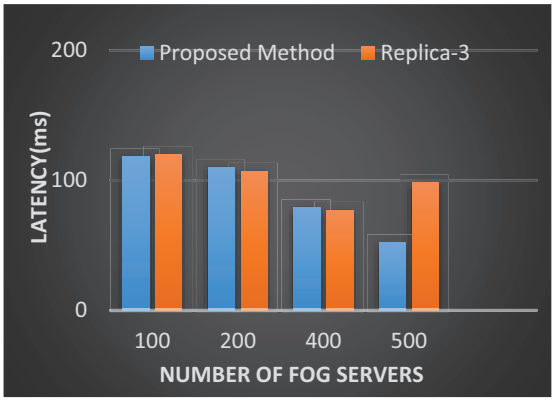


Fig. 9. Latency comparison using Naïve Bayes classification

V. Conclusion

This study suggests a new method for replicating file or content to enhance fog computing content replication. The recommended includes of two stages. The first is to use the failure history data of the fog servers to classify their failure risk using Naïve bayes and KNN classification model. We have proposed a replicationbased on content size technique for the second stage to make it more efficient by duplicating the file or content on thelow-risk failure fog server. For the performance evaluation, we considered indicators such as throughput, latency, and resource utilization. Simulation findings comparing the suggested strategy with the replica-3 method validate its utility. As a result, in comparison to the Replica-3 technique, the suggested method offers higher throughput and lower latency using KNN classification model.

VI. Future Scope

Future content replication approaches will be heavily influenced by automation, artificial intelligence, and distributed computing environments like edge and fog computing. AI-driven replication will enable the implementation of dynamic, context-aware processes that optimize data distribution in real-time, reducing costs and boosting productivity. Security will be a key area of research as well, with advanced encryption and privacy-preserving techniques ensuring data preservation during replication. As

Authorized licensed use limited to: UNIVERSITY OF OSLO. Downloaded on August 11,2026 at 10:07:13 UTC from IEEE Xplore. Restrictions apply.

the need for low-latency, high-availability systems grow, content replication techniques will become more adaptable, scalable, and crucial for enabling new technologies and applications.

REFERENCES

[1] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey, and future directions," in *Internet of Everything*, Singapore: Springer, 2018, pp. 103–130.

[2] E. Balevi and R. D. Gitlin, "Optimizing the number of fog nodes for cloud-fog-thing networks," *IEEE Access*, vol. 6, pp. 11173–11183, 2018.

[3] K. Kaur and M. Sachdeva, "Fog computing in IoT: An overview of new opportunities," in *Proc. ICETIT 2019*, 2020, pp. 59–68.

[4] J. Wan et al., "Fog computing for energy-aware load balancing and scheduling in smart factories," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4548–4556, 2018.

[5] P. G. Vinueza Naranjo, E. Baccarelli, and M. Scarpiniti, "Design and energy-efficient resource management of virtualized networked fog architectures for the real-time support of IoT applications," *J. Supercomput.*, vol. 74, no. 6, pp. 2470–2507, 2018.

[6] Y. Yang et al., "Maximal energy-efficient task scheduling for homogeneous fog networks," in *Proc. IEEE INFOCOM Workshops (INFOCOM WKSHPS)*, 2018, pp. 274–279.

[7] P. Kumari, V. Dubey, and S. Shrivastava, "Content replica placement method for fault tolerance in fog computing environment," in *Proc. IEEE World Conf. Appl. Intell. Comput. (AIC)*, 2023.

[8] "Unified fog node utilization for multiple content providers through cluster-based cooperative caching," in *Proc. IEEE 25th Int. Symp. World Wireless, Mobile, Multimedia Netw. (WoWMoM)*, 2024.

[9] F. Sharifi, Y. C. Lee, and S. Hessabi, "Evolutionary game for content cache in a mm-Wave-based vehicular fog," *IEEE Internet Things J.*, 2022.

[10] T. Itokazu and S. Sugawara, "An efficient content sharing using dynamic fog considering transition of number of mobile terminals in a city," in *Adv. Broad-Band Wireless Comput., Commun., Appl.*, 2020.

[11] Z. Su et al., "A secure content caching scheme for disaster backup in fog computing-enabled mobile social networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, Oct. 2018.

[12] K. Sarwar et al., "Efficient privacy-preserving data replication in fog-enabled IoT," *Comput. Commun.*, vol. 128, pp. 538–551, Mar. 2022.

[13] J. Taghizadeh, M. Ghobaei-Arani, and A. Shahidinejad, "An efficient data replica placement mechanism using biogeography-based optimization technique in the fog computing environment," *J. Cloud Comput.*, vol. 14, pp. 3691–3711, 2023.

[14] R. Oma et al., "Fault-tolerant fog computing models in the IoT," in *Proc. 13th Int. Conf. P2P, Parallel, Grid, Cloud, Internet Comput. (3PGCIC)*, 2018, pp. 14–25.

[15] A. Alarifi, F. Abdelsamie, and M. Amoon, "A fault-tolerant aware scheduling method for fog-cloud environments," *PLoS One*, vol. 14, no. 10, Oct. 2019.

[16] S. T. Manukumar and V. Muthuswamy, "A novel resource management framework for fog computing by using machine learning algorithm," in *Architecture and Security Issues in Fog Computing Applications*, 2020, pp. 42–52.

[17] A. Inani, C. Verma, and S. Jain, "A machine learning algorithm TSF k-NN based on automated data classification for securing mobile cloud computing model," in *Proc. IEEE 4th Int. Conf. Comput. Commun. Syst. (ICCCS)*, 2019, pp. 9–13.

[18] The dataset analyzed in this study is NERSC data, available in the CFDR repository: [Online]. Available: https://www.usenix.org/cfdr-data.

[19] R. Siyadatzadeh et al., "ReLIEF: A reinforcement learning-based real-time task assignment strategy in emerging fault-tolerant fog computing," *IEEE Internet Things J.*, Jan. 2023.

[20] S. A. Syed et al., "QoS-aware and fault tolerance-based software-defined vehicular networks using cloud-fog computing," *Sensors*, vol. 22, no. 1, Jan. 2022.

[21] Y. Ramzanpoor et al., "Multi-objective fault-tolerant optimization algorithm for deployment of IoT applications on fog computing infrastructure," *Complex Intell. Syst.*, vol. 8, no. 1, pp. 361–392, Feb. 2022.

[22] A. Reyana et al., "Opportunities of IoT in fog computing for high fault tolerance and sustainable energy optimization," *Sustainability*, vol. 15, no. 11, May 2023.

[23] P. Sharma and P. K. Gupta, "Optimization of IoT-fog network path and fault tolerance in fog computing-based environment," *Procedia Comput. Sci.*, vol. 218, pp. 2494–2503, Jan. 2023.

[24] S. Alraddady et al., "Fog computing: Strategies for optimal performance and cost effectiveness," *Electronics*, vol. 11, no. 21, Nov. 2022.

[25] C. Guerrero, I. Lera, and C. Juiz, "Genetic-based optimization in fog computing: Current trends and research opportunities," *Swarm Evol. Comput.*, Jul. 2022.

[26] P. Zhang et al., "A fault-tolerant model for performance optimization of a fog computing system," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1725–1736, Jun. 2021.

[27] Z. Zeng, L. Chang, and Y. Liu, "A fault tolerance data aggregation scheme for fog computing," *Int. J. Inf. Comput. Secur.*, vol. 17, no. 3, pp. 351–364, 2022.

[28] F. Karamimirazizi, S. M. Jameii, and A. M. Rahmani, "Data replication methods in cloud, fog, and edge computing: A systematic literature review," *Wireless Pers. Commun.*, vol. 135, pp. 531–561, 2024. doi: 10.1007/s11277-024-11082-7.

[29] C. Guerrero, I. Lera, and C. Juiz, "Optimization policy for file replica placement in fog domains," *Concurrency Comput. Pract. Exp.*, vol. 32, no. 18, e5343, 2020. doi: 10.1002/cpe.5343.

Authorized licensed use limited to: UNIVERSITY OF OSLO. Downloaded on August 11,2026 at 10:07:13 UTC from IEEE Xplore. Restrictions apply.