



# Composition of caching and classification in edge computing based on quality optimization for SDN-based IoT healthcare solutions

Seyedeh Shabnam Jazaeri<sup>1</sup> · Parvaneh Asghari<sup>2</sup> · Sam Jabbehdari<sup>1</sup> ·  
Hamid Haj Seyyed Javadi<sup>3</sup>

Accepted: 21 April 2023 / Published online: 9 May 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

## Abstract

This paper proposes a novel approach that uses a spectral clustering method to cluster patients with e-health IoT devices based on their similarity and distance and connect each cluster to an SDN edge node for efficient caching. The proposed MFO-Edge Caching algorithm is considered for selecting the near-optimal data options for caching based on considered criteria and improving QoS. Experimental results demonstrate that the proposed approach outperforms other methods in terms of performance, achieving decrease in average time between data retrieval delays and the cache hit rate of 76%. Emergency and on-demand requests are prioritized for caching response packets, while periodic requests have a lower cache hit ratio of 35%. The approach shows improvement in performance compared to other methods, highlighting the effectiveness of SDN-Edge caching and clustering for optimizing e-health network resources.

**Keywords** Internet of Things (IoT) · Software-defined networking (SDN) · Edge computing · Healthcare · Clustering and caching

## Abbreviations

|      |                          |
|------|--------------------------|
| BSNs | Body sensor networks     |
| CCN  | Content-centric network  |
| CIB  | Content information base |
| EC   | Edge caching             |

---

✉ Parvaneh Asghari  
p\_asghari@iauctb.ac.ir

<sup>1</sup> Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran

<sup>2</sup> Department of Computer Engineering, Central Tehran Branch, Islamic Azad University, Tehran, Iran

<sup>3</sup> Department of Computer Engineering, Shahed University, Tehran, Iran

|        |                                       |
|--------|---------------------------------------|
| ECG    | Electrocardiogram                     |
| EHR    | Electronic health record              |
| EMR    | Electronic medical records            |
| FDA    | Food and Drug Administration          |
| GWO    | Grey wolf optimization                |
| HL7    | Health level 7                        |
| HDP    | Health data popularity                |
| IoT    | Internet of Things                    |
| IoMT   | Internet of Medical Things benchmark  |
| LRU    | Least recently used algorithm         |
| MFO    | Moth-flame optimization algorithm     |
| MCDs   | Multiple chronic diseases             |
| MFO-EC | Moth-flame optimization-edge caching  |
| PSO    | Particle swarm optimization algorithm |
| QoE    | Quality of experience                 |
| QoS    | Quality of service                    |
| RHMS   | Remote health monitoring systems      |
| SDN    | Software-defined networking           |
| SDWN   | Software-defined wireless network     |

## 1 Introduction

The need for healthcare services is increasing due to demographic shifts and a rise in chronic diseases. This situation is exacerbated by the temporary demand for services during disasters. Therefore, researchers are exploring how to improve healthcare service delivery networks to handle the rising demand. The healthcare industry is undergoing a digital transformation, including the use of wearable sensors to monitor patients' physiological and mobility data [1]. These sensors are connected wirelessly to a distant center that recognizes emergencies and alerts the emergency support center. Advances in technology are making therapeutic care accessible anytime, anywhere. However, transferring massive amounts of data processing and storage in the cloud is impractical and can negatively impact health-related decision-making processes [2]. Edge computing is an intermediary between the cloud and equipment, minimizing the data that needs to be sent to the cloud and providing more effective processing, analysis, and storage [3]. The deployment of IoT technologies in healthcare solutions has led to an exponential increase in data generated by medical devices, leading to a growing need for edge computing solutions that can provide low-latency, high-bandwidth communication between devices and servers [4].

The model we have considered for this problem is a content-centric network managed by the SDN central controller. The controller manages the submission and setting of flow controls, monitors of the content, and decides which edge nodes to cache which content(s) [5]. It also decides which requests will be resolved by which edge nodes or cloud servers and through which links and routes [4]. SDN offers new features to IoT-enabled networks in addition to mitigating the drawbacks of traditional networks. Increasing patient numbers, hospitals joining these networks,

medical equipment, and so forth demand SDN to support and execute the functions [2]. Caching transient data at edge nodes in e-health IoT can improve network traffic management and QoS/QoE. Edge caching allows for low-delay data to be provided for delay-sensitive applications. Popularity prediction algorithms and freshness requirements should be considered when constructing caching strategies [6]. Previous studies on edge caching and IoT caching have focused on caching transient data or content popularity [7]. A multi-criteria caching solution that supports SDN and enables caching decisions for e-health IoT data at the edge is proposed in this study.

The proposed architecture allows for the efficient sharing of medical records between patients and nearby edge devices in e-health IoT. Edge computing enables the processing and caching of electronic medical records (EMR) for more secure and efficient services compared to regular cloud computing. Spectral clustering is used to identify similarities between patients based on age, gender, chronic or acute disease, electronic health record (HER), and distance, and link clusters with the nearest SDN edge node for caching and sending transient e-health IoT data. The moth-flame optimizing caching, "MFO-Edge Caching," algorithm is used to select near-optimal content for edge caching using multi-criteria decision-making, taking into account critical characteristics such as data freshness, sensing delay, popularity, priority of patients based on the severity of illness, and energy level parameters. This SDN-based edge caching e-health IoT system improves data retrieval latency and enables quick reaction to demands for integrated health systems. Performance analysis of the proposed system shows improved access to medical records and better healthcare services. The contributions of the proposed approach are:

- **Efficient clustering of patients:** The approach uses spectral clustering to cluster patients with e-health IoT devices based on their similarity and distance, using both static and dynamic patient information. This clustering technique provides a more efficient way to group patients with similar healthcare needs.
- **Smart caching mechanism:** The approach uses a caching technique implemented at the SDN controller to determine what items need to be cached and which edge node should cache them. It gives priority to the patient's data and increases the cache hit ratio for frequently accessed information. Additionally, to maximize content diversity, each content copy is stored only once at the edge.
- **MFO algorithm for edge caching:** The approach proposes the use of the MFO algorithm for caching multiple patients' medical records, which involves decision-making with respect to multiple criteria. This algorithm is considered more logical, explicit, and effective than traditional approaches and can weight medical parameters based on illness severity and treatment policy.
- **Fair and humane medical care system:** The proposed approach prioritizes patients with more severe illnesses and reduces waiting times for them, leading to a fairer and more humane medical care system.
- It shows that the data retrieval latency that is required to communicate in an emergency cannot be accomplished using cloud technologies. The answer is provided by SDN-based edge caching e-health IoT, which allows for the creation of integrated health systems and the quick reaction to demands. Our final step was to examine the performance of the suggested system. Patients' medical records

can be accessed more quickly and better services can be provided when e-health data is available.

Centralized management and dynamic caching mechanism improve the overall performance and reliability of IoT healthcare systems. This, in turn, enables healthcare providers to prioritize and deliver better services to patients, such as real-time monitoring, early detection of health issues, timely interventions, and personalized care. The contributions of the approach are the efficient clustering of patients, the smart caching mechanism, the use of the MFO algorithm for edge caching, and the creation of a fair and humane medical care system.

The rest of the paper is as follows. In Sect. 2, we provide the related works. Motivation is explained in Sect. 3. Section 4 focuses on the problem formulation. Proposed architecture is discussed in Sect. 5. In Sect. 6, the methodology is described, also the clustering model and multi-criteria caching decision-making are formulated, and proposed algorithms are explained. The results of our performance evaluation and simulation are presented in Sect. 7. We briefly point out the limitations in healthcare system and future works in Sect. 8. Lastly, the paper is concluded in Sect. 9.

## 2 Related works

We conduct research relating to edge computing, SDN, and caching in e-health IoT. There is a brief description of the technological background and current research accomplishments of the above technologies. The following section provides an overview and analysis of recent research related to edge caching and SDN-IoT, categorized caching techniques and identified innovations, algorithms, strategies, proposed methods, simulators, and solution limitations.

The medical body area network collects triage and physiological data using sensors in urban areas to aid rescue operations. The remote health monitoring system (RHMS) aims to provide timely medical treatment using telecommunications technology in rural areas, and patients with multiple chronic diseases benefit significantly from it [8]. Additionally, edge computing is another approach to enhancing health monitoring systems, which involves using a gateway to extract ECG features and implement various edge computing services. This results in increased sensor life and energy efficiency in Body Sensor Networks (BSNs) [9]. The authors in [10–12] discuss the benefits of using edge technology in healthcare. They suggest that mobile devices may not have sufficient processing and storage capabilities to handle medical data, and propose an architecture that involves caching and storing data on edge servers. Edge technology is said to be superior to cloud technology in terms of information transmission speed, security, and processing power. The authors claim that using an edge-based system can significantly reduce the amount of information that needs to be processed in the cloud, as most of the work is done at the edge. The proposed architectures are designed to improve healthcare monitoring and enable patients to access medical data quickly and securely. In [3], the authors focus on

edge computing and caching in Internet of Medical Things (IoMT) applications and introduce a benchmark, HERMIT, that enables edge devices to analyze data.

The proposed system in [13] involves location-aware edge caching with DVS (Descriptive Video Service) technology to reduce the size of medical videos and improve the efficiency of local hospitals. Simulation results showed that the scheme with DVS camera technology can reduce transmission delay. In [14], authors suggest that E-health, due to its need for short delays and safe local storage, will require more IoT solutions in the future. Fog computing is proposed as a more relevant approach, using Content-Centric Network (CCN) technology to provide quick response, content distribution, and IoT-oriented technology. Simulation is used to evaluate the delay performance of the proposed solution. In [15], the authors propose an algorithm for optimizing fog network performance using caching placement. They classify files based on popularity and replicate them on high-level fog nodes. They also select the cluster head for caching the file.

E-health IoT service based on SDN has an application that collects data via smartphone voice control. The proposed service by using mobile applications identifies heart attack types. Cloud environments are not evaluated for the proposed application's performance, according to the authors [16]. A real-time security attack can be prevented through the use of SDN-IIoT technology. SDN can allow for efficient migration of healthcare loads to improve network performance and user experience. Consequently, medical treatments will be safer, more affordable, and better for patients. Real-time and efficient protection against security attacks is being investigated using SDN-IIoT technology [4]. In [17], a centralized SDN cache controller and clustering mechanism are used in Extended Multi-Criteria Cooperative Caching (EM3C). Cluster heads make localized decisions about where to cache content semi-hierarchically. [18] proposes a cache prefetching algorithm based on the UCBM model to speed up end-user access and the HFAP model for cache replacement to reduce cloud resource latency. [19] proposes the EM3C method for data caching in IoT environments using SDN routing to intelligently route requests and maximize resource usage while reducing cached content redundancy. [20] focuses on cache-enabled SDNs and proposes a polynomial-time algorithm to solve the joint problem of dynamic caching and adaptive video resolution, which increases QoE while improving resource consumption efficiency. [21] presents SD-RF, a replication management system for Content-Centric Networks based on SDN architecture, combining the benefits of SDN and CCN, [22] introduces RF-VS-SD-CCMN, a replication platform for video streaming in Content-Centric Mobile Networks that increases content distribution, perceived content quality, and network resource usage by considering user mobility and using offline and online methods for storage positioning and replica assignment. [23] proposes an efficient multifactorial function in the network cache decision algorithm that takes into account content size, hop counts, data latency, and node energy level, enhancing hit rate, reducing data recovery delay, and increasing information reusability in IoT settings. [24] studies caching strategies for IoT networks and shows that periodic storage reduces retrieval latency and improves cache performance, disseminating information faster and bridging the distance between data and customers. [25] proposes a forward network prediction model for visual media in the active network that actively forecasts users' demands through

strategic caching to minimize peak-data traffic demand. [26] provides a comprehensive study on the compatibility challenges of fog computing and IoT and investigates various caching schemes to optimize IoT-based fog networks. [27] presents a low-latency edge caching strategy that reduces user access latency and proposes a migration approach that balances communication and migration overhead, increasing cache reward while reducing delay and boosting revenue from service migration while lowering communication costs. [28] provides a comprehensive review of the current state of IoT Edge Caching and proposes a new taxonomy with five orthogonal features, examining five prevalent use cases for IoT and conducting an in-depth analysis and performance evaluation of edge caching in a real SHM scenario.

The papers [2, 29–33] propose various solutions to improve edge caching in fog and mobile edge computing networks. [29] proposes a D2D fog network architecture based on federated learning and QoS to estimate user preferences for content caching decision-making, while [30] proposes MagNet, a decentralized and cooperative edge caching system that utilizes Automatic Content Congregating and Mutual Assistance Group mechanisms to improve hit ratio and workload balance. [31] Discusses the advantages of SDN and edge caching in mobile edge computing networks and presents a proposed system model and caching algorithm that outperforms existing models in terms of hit rate, time delay, and traffic reduction. [2] proposes a mechanism to determine optimal content placement in a distributed edge deployment with limited storage resources to minimize data retrieval latency, and [32] proposes a data prioritization-based approach that utilizes fog computing to improve cache efficiency in ICN for IoT data. Finally, [33] proposes an intelligent traffic prediction mechanism for dynamic resource allocation using a long short-term memory neural network.

Additionally, some of the papers also propose clustering techniques as for IoT environments. Clustering helps to group together similar IoT devices or nodes to reduce communication overhead and improve data processing efficiency. In [34], the authors propose a multi-model patient similarity network (PSN) to address data heterogeneity and dimensionality in precision medicine. It uses deep learning models to analyze contextual data, generate word embeddings, and preserve temporal features. [35] proposes similarity-based clustering for IoT device classification. This paper evaluated the limitations of device classification based on traditional device fingerprinting and black-box machine learning by proposing a semantic similarity-based clustering method. The results showed that it can successfully classify a good number of to-be-classified devices with consistency and high granularity. [36] presents the design of a multi-hop similarity-based clustering framework for IoT-oriented software-defined wireless sensor networks (MSCSDNs) to improve network performance. The study emphasizes the importance of routing protocols and deployment techniques in IoT-based wireless sensor networks and highlights the benefits of using SDN-based IoT architecture for on-demand network management protocols and applications. [37] proposes an energy-efficient hybrid clustering technique (EEHCT) for IoT-based HWSN that minimizes energy consumption and distributes the network load evenly. It achieves a gain of up to 90.27% with respect to network lifetime and a statistical analysis to demonstrate the formation of energy-balanced clusters. [38] proposes an algorithm called DT-MAC to address issues related to low

computing capacity and mobility in wireless sensor networks used in various IoT-enabled applications such as healthcare and body area networks. DT-MAC utilizes the concept of minimum connected dominating set for network formation and considers node handover mechanism among virtual clusters to ensure successful message delivery and efficient energy utilization. DT-MAC is recommended for real-time applications requiring high packet delivery and response time. [23] proposes an efficient in-network caching decision algorithm for IoT, which utilizes similarity calculation to make optimal caching decisions. These studies provide guidance on selecting the best similarity measures for specific healthcare tasks and highlight the importance of choosing the right measure for accurate and effective analysis of IoT healthcare data.

Overall, these papers propose various cache management and clustering strategies, including prefetching, SDN, and periodic storage, for different IoT environments, and edge computing. These strategies aim to improve network performance, reduce latency, enhance QoE, and lower communication costs. Table 1 illustrates the contribution of the proposed solution in recent articles, as well as the caching strategy/algorithms used, limitations, and the related examined attributes to those algorithms.

Table 1 presents a summary of recent research papers that discuss the deployment of edge caching approaches for Software-Defined Networking (SDN)-based Internet of Things (IoT) networks. The table includes information on the year of publication, the approach used, the objective, the method, Caching strategy /Used algorithms, and the Limitations of each paper. The table highlights the importance of edge caching in improving the performance and efficiency of SDN-based IoT networks. The approaches discussed in the papers range from centralized caching to distributed caching and from heuristic-based algorithms to machine learning-based approaches. The objectives vary from reducing the latency and energy consumption to improving the Quality of Service (QoS) and user satisfaction. The evaluation methods include simulation, mathematical modeling, and experimental validation. The summary of Table 1 emphasizes the diversity of edge caching approaches in SDN-based IoT networks and the potential benefits they offer. It also demonstrates the need for further research in this area to address the challenges of network.

The research paper proposes a new approach to caching patient medical records in an e-health IoT environment using edge computing and SDN. The proposed solution uses a multi-criteria caching algorithm that takes into account critical characteristics like data freshness, sensing delay, content popularity, patient priority, and energy levels to make caching decisions. Additionally, spectral clustering is used to identify similarities between e-health users and link these clusters with the nearest SDN edge node to improve healthcare efficiency by making EMRs more readily available at the edge. Centralized management and dynamic caching mechanisms are crucial for IoT healthcare systems to provide better services to patients. SDN-Edge caching is a more secure option for IoT healthcare systems than other caching solutions due to its proximity to end devices, controlled access, granular control, and flexibility. With SDN-Edge caching, data transmission can be closely monitored, limiting unauthorized access. The importance of this research lies in the fact that it emphasizes the importance of fast data retrieval in emergency situations and highlights how the

**Table 1** Recent papers in edge caching approaches deployment for SDN-based IoT networks

| Authors           | Papers   | Year | Contribution of SDN the proposed solution  | Edge computing | SDN comput- ing | Cache hit probability | Latency/delay | Traffic management | Performance | Energy consumption | Caching strategy/ Used algorithms   | Limitations   |
|-------------------|--|------|--|----------------|-----------------|-----------------------|---------------|--------------------|-------------|--------------------|---|---|
| Samir et al. [31] | Cluster-Based Multi-User Caching Mechanism in 5G/6G                | 2023 | A solution to the caching issue in a configuration with distributed caching components and limited storage and backhaul capacity | *              | *               | *                     | *             | *                  | *           |                    | A Cluster-based Multi-User Multi-Server (CMUMS) caching algorithm/ Task Popular Caching (TPC) and Multi-User Multi-Server task Caching (MUMSC)                      | This requires more orientation, thorough vision, application selection, and further consideration   |
| Peng et al. [30]  | MagNet: Cooperative Edge Caching by Automatic Content Congregating | 2022 | A decentralized and cooperative edge caching system named MagNet   | *              | *               | *                     | *             | *                  | *           |                    | Machine learning-based and cooperative caching solutions/The ACC mechanism and clustering algorithm guides requests to their optimal edges to enhance the hit ratio | The latency between the two neighbors and the cached content size of the algorithm corresponds to their GT is a complex and valuable problem that is not considered |

**Table 1** (continued)

| Authors               | Papers   | Year | Contribution of SDN the proposed solution  | Edge computing | Cache hit probability | Latency/delay/Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms  | Limitations |
|-----------------------|--|------|--|----------------|-----------------------|----------------------------------|-------------|--------------------|---|-------------|
| Zyrianoff et al. [28] | IoT Edge Caching: Taxonomy, use cases and perspectives                           | 2022 | A five orthogonal features taxonomy of edge caching and its evaluated performance of edge caching deployed in a real structural health monitoring (SHM) scenario | *              | *                     | *                                | *           | *                  | Proactive and reactive edge caching strategy/The distributed Hungarian algorithm, This requires examining ways to talk about the factors that affect cache techniques |             |
| Huang et al. [29]     | Federated learning-based QoS-aware caching decisions in fog-enabled IoT networks | 2022 | A comprehensive study on fog computing, IoT's and the compatibility challenges of the two and a summary of machine learning-based storage schemes                | *              | *                     | *                                | *           | *                  | An optimal decision-making in content caching/The proposed algorithm performs clustering based on content popularity and user preference                              | –           |

**Table 1** (continued)

| Authors               | Papers   | Year | Contribution of SDN the proposed solution  | Edge computing | Cache hit probability | Latency/delay/Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms   | Limitations  |
|-----------------------|--|------|--|----------------|-----------------------|----------------------------------|-------------|--------------------|--|--|
| Zahedinia et al. [32] | Fog-based caching mechanism for IoT data in ICN using prioritization | 2022 | A data prioritization-based approach is proposed to improve cache efficiency                                       | *              | *                     | *                                |             |                    | A fog-based caching method/It considered two essential data characteristics into the prioritization mechanism, freshness and popularity and LCE method | It needs to consider the trade-off between cost and performance                            |
| Li et al. [27]        | A low-latency edge caching method                                    | 2022 | An edge caching method for effectively cache diverse content in the edge network and reduce latency, a low-latency | *              | *                     | *                                | *           |                    | Edge collaborative caching/The service migration method based on balanced communication overhead and migration overhead                                | The problem of minimizing latency is transformed into a problem of maximizing cache reward |

**Table 1** (continued)

| Authors            | Papers                                 | Year | Contribution of SDN the proposed solution  | Edge computing | Cache hit probability | Latency/delay/Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms   | Limitations  |
|--------------------|--|------|--|----------------|-----------------------|----------------------------------|-------------|--------------------|--|--|
| Ruggeri et al. [2] | Caching Popular Transient IoT Contents | 2021 | A caching orchestration system that runs as a network application on top of an SDN infrastructure                              | *              | *                     | *                                | *           | *                  | Optimal content placement through an Integer Linear Programming (ILP) problem. A heuristic algorithm | The article does not address the issue of optimizing a caching method based on a trade-off between two opposing objectives |
| Selvi et al. [33]  | URLLC-SDN-Edge computing               | 2021 | An intelligent traffic prediction mechanism for both edge cloud and centralized SDN controller for dynamic resource allocation | *              | *                     | *                                | *           | *                  | Edge caching with proactive packet drop  | The next generation of 5G networks must consider how network slicing can help to automate resource allocation even further |

Table 1 (continued)

| Authors           | Papers                            | Year | Contribution of SDN   | Edge comput- ing | Cache hit probability | Latency/delay | Traffic manage- ment | Perfor- mance | Energy con- sumption | Caching strategy/ Used algorithms   | Limitations  |
|-------------------|-----------------------------------|------|---|------------------|-----------------------|---------------|----------------------|---------------|----------------------|-------------------------------------|--|
| Zhang et al. [39] | SDN-based Caching Decision Policy | 2020 | SDN-based caching deci- sion policy outperforms the existing caching deci- sion policies in terms of the hit ratio and the aver- age number of hops | *                | *                     | *             |                      |               |                      | SDN-based cach- ing decision policy | Extending the proposed caching decision policy to mobile networks, e.g., 5G cellular networks, as a critical factor in 5G networks architecture has not been stud- ied in this article |

**Table 1** (continued)

| Authors                | Papers | Year | Contribution of SDN the proposed solution                             | Edge computing | Cache hit probability | Latency/delay | Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms                              | Limitations  |
|------------------------|--------|------|---|----------------|-----------------------|---------------|--------------------|-------------|--------------------|---|--|
| Wu et al. [18]         | HFAP   | 2020 | Optimization of heat-based cache replacement in edge computing system | *              | *                     | *             | *                  | *           |                    | File replacement based on cache Prefetching. Priority and LRU | It employs a reactive approach, which may not be suitable for centralized cache decisions. Moreover, the security of edge computing requires further investigation |
| Khodaparas et al. [19] | EM3C   | 2020 | A software-defined caching scheme for the Internet of Things          | *              | *                     | *             | *                  | *           |                    | Global cache management prevents any redundancy               | It should take mobility into account and attempt to adapt the proposed method to account for the node's movement in the environment                                |

**Table 1** (continued)

| Authors           | Papers | Year | Contribution of SDN the proposed solution   | Edge computing | Cache hit probability | Latency/delay/Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms  | Limitations  |
|-------------------|--------|------|---|----------------|-----------------------|----------------------------------|-------------|--------------------|---|--|
| Fekih et al. [21] | SD-RF  | 2020 | Proposed a framework that used for storage placement and replica assignment using both offline and online methods | *              | *                     | *                                | *           | *                  | Both offline and online for storage placement and replica allocation based on hybrid architecture<br>Cooperative Cache Management Algorithm, Myopic Cache Management Algorithm, the greedy and random algorithms<br>Distributed cache management algorithms | The model that clearly explains the duties of each module in controller management architecture has not been specified in this study |

**Table 1** (continued)

| Authors        | Papers             | Year | Contribution of SDN the proposed solution  | Edge computing | Cache hit probability | Latency/delay/Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms   | Limitations   |
|----------------|--------------------|------|--|----------------|-----------------------|----------------------------------|-------------|--------------------|--|---|
| Lv et al. [40] | EOESPA and RNOESPA | 2020 | Optimization algorithms are proposed based on SDN technology edge network service architecture | *              | *                     | *                                | *           |                    | Enumeration-Based Optimal Edge Server Placement Algorithm (EOESPA) and Ranking-based Near-optimal Edge Server Placement Algorithm (RNOESPA)<br>Optimal Enumeration Service Deployment Algorithm (OESDA), Latency Aware Heuristic Service Deployment Algorithm (LAHSDA), and Clustering Enhanced Heuristic Service Deployment Algorithm (CEHSDA)<br>K-Means cluster algorithm | It is possible to explore how to efficiently allocate resources in the setting of various mobile devices and edge servers |

**Table 1** (continued)

| Authors           | Papers        | Year | Contribution of SDN the proposed solution   | Edge computing | Cache hit probability | Latency/delay | Traffic management | Performance | Energy consumption | Caching strategy/Used algorithms  | Limitations |
|-------------------|---------------|------|---|----------------|-----------------------|---------------|--------------------|-------------|--------------------|---|-------------|
| Fekih et al. [22] | RF-VS-SD-CCMN | 2020 | Presenting a new replication framework for video streaming in Content-Centric Mobile Networks based on SDN Architecture | *              | *                     | *             | *                  | *           |                    | Interests' anticipation cooperative caching content replication Heuristic algorithm |             |

proposed system can help create integrated health systems and improve patient services. Centralized management and dynamic caching mechanisms are also essential for optimizing the system's performance, reliability, and security in IoT healthcare systems.

### 3 Motivation

Access to high-quality, affordable, and up-to-date medical services is a fundamental right that should be available to every human being. However, modern healthcare systems face numerous challenges, including the aging population crisis, chronic illnesses, and disasters that can lead to temporary shortages of healthcare services [41]. To meet the growing demand for healthcare services and address these significant problems, an efficient system is essential. Effective communication and monitoring of risk are crucial for ensuring patient safety and system efficiency.

The emergence of the Internet of Things (IoT) and cloud computing has revolutionized traditional healthcare delivery, enabling real-time applications that improve patient care. IoT and cloud-based healthcare systems have enabled easy communication between patients and clinicians, leading to increased patient participation and satisfaction [7]. Additionally, remote monitoring systems can reduce hospital stays and prevent readmissions, which in turn lowers healthcare expenditures while improving treatment outcomes [42]. As a result, there has been a surge in the number of IoT and cloud-based healthcare applications in recent years.

E-health IoT networks face numerous challenges that must be addressed to optimize network infrastructure and deliver better healthcare services to patients. One of the key challenges is the need for centralized control and management to ensure effective network optimization. Another significant challenge is managing the massive amounts of data generated by e-health IoT devices, including patient medical records, vital signs, and other health-related data. Reducing data retrieval latency and improving the quality of service for healthcare providers and patients is another major challenge that must be addressed. Moreover, ensuring reliable and secure data transmission is crucial for the successful operation of e-health IoT networks. Lastly, the challenge of prioritizing and clustering patients based on their similarity and distance to allocate and cache patient medical records using SDN-Edge caching is of utmost importance. Addressing these challenges can lead to improved healthcare services, real-time monitoring, early detection of health issues, timely interventions, and personalized care for patients.

The proposed approach aims to address these challenges by leveraging IoT, software-defined networking (SDN), edge, and cloud computing technologies to develop an IoT-enabled medical equipment system. The system can prioritize patients with more severe illnesses and reduce waiting times for them, leading to improved treatment outcomes and patient satisfaction. By using real-time remote monitoring systems, healthcare providers can quickly identify patients who need urgent care and provide timely interventions. Additionally, the use of SDN and edge computing can improve the efficiency of healthcare delivery systems, reducing costs and ensuring that resources are allocated fairly to those who need them the most.

In conclusion, the main motivation behind proposing this approach is to provide a more efficient and effective solution for IoT-based healthcare services that can meet the real-time requirements of healthcare providers and improve patient outcomes.

## 4 Problem formulation

The deployment of Internet of Things (IoT) technologies in healthcare solutions has led to an exponential increase in the amount of data generated by medical devices. In order to efficiently process and analyze this data, there is a growing need for edge computing solutions that can provide low-latency, high-bandwidth communication between devices and servers. However, the limited computing resources available at the edge make it challenging to perform complex analytics tasks, such as caching and classification, in real-time.

The objective of this research is to propose a composition of caching and classification techniques that can optimize the quality of service (QoS) for SDN-based IoT healthcare solutions. Specifically, the research aims to:

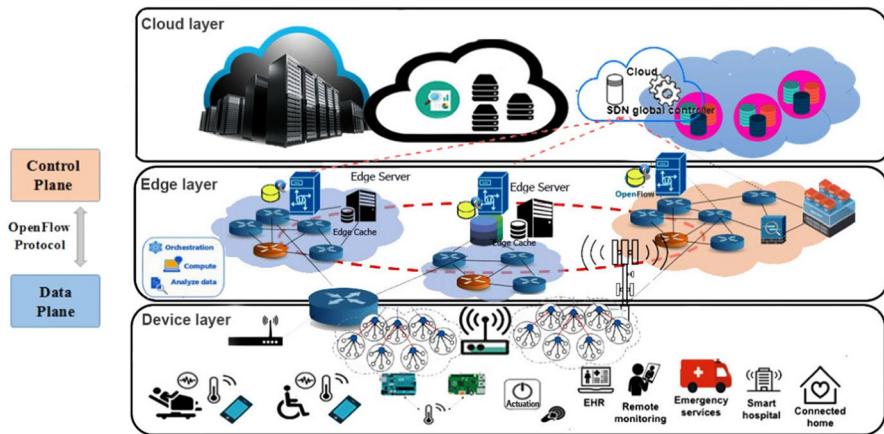
- Develop a caching mechanism that can store frequently accessed data at the edge, reducing latency and bandwidth requirements.
- Implement a classification algorithm that can accurately classify incoming data based on its type and priority.
- Optimize the performance of the caching and classification techniques by adjusting parameters.
- Evaluated the proposed solution in a simulated healthcare environment and examined how the clustering method and dynamic and centralized management of SDN affected caching IoT healthcare data at the edge of the network.

By addressing these research questions, this study aims to provide a dynamic and effective solution for optimizing QoS in SDN-based IoT healthcare solutions.

Also, this paper aims to address the challenges of managing large volumes of data generated by e-health IoT devices through the use of Software-Defined Network (SDN) technology and edge caching. The main objective of the proposed approach is to reduce the retrieval delay and enhance the Quality of Service (QoS) of e-health IoT systems. The paper presents an edge caching scheme based on a spectral clustering method that clusters patients with e-health IoT devices based on their similarity and distance. The scheme is designed to dynamically route traffic, enforce security policies, and cache data locally to reduce latency and the risk of data breaches during transmission. The proposed approach is compared with existing content-based caching strategies to demonstrate its superiority in terms of performance and scalability.

## 5 Proposed architecture

Figure 1 illustrates how SDN-Edge caching can be employed in e-health IoT system architectures based on the OpenFlow framework. To reduce the latency in e-health IoT networks, the network architecture based on SDN uses the storage

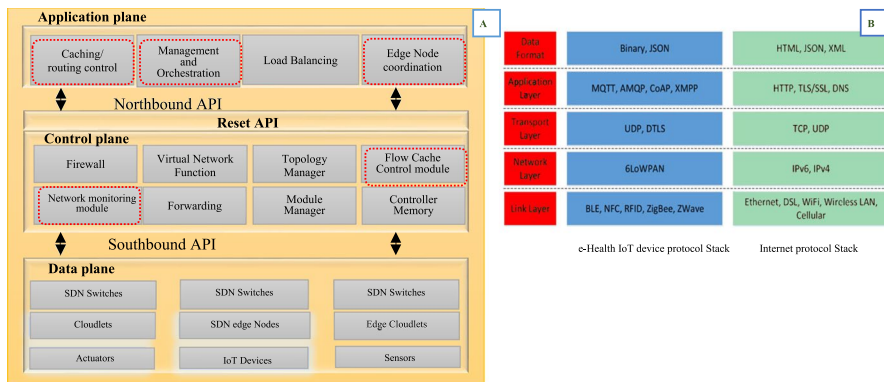


**Fig. 1** Proposed multilayer architecture of e-health IoT-based on SDN-Edge computing

and computing resources at the network edge. Using patient clustering and caching techniques, this section aims to reduce the response time after a user requests the patients' data by storing the data on edge nodes. To deliver high-speed services, OpenFlow switches are linked to SDN controllers. Because each controller is responsible for a specific geographical region in the control plane, its view of the network state will be local. The management of information is relevant to service requests and flow transmissions. To handle network compatibility and enable dynamic access and handover, the southbound interface protocols are enhanced with middleware programs. This architecture uses global SDN controller knowledge to identify the edge content location and forward the request to the SDN edge node that stores it. The protocol stack of IoT e-health is illustrated in this Fig. 2B. The application and its unique requirements have a significant impact on the choice of the most appropriate connectivity and communication technology. Wi-Fi is ideal for transferring large numbers of documents wirelessly, for instance, While BLE is best suited to communications with short range and low power.

Figure 2A depicts the architecture levels of SDN, Edge, and IoT consists of three planes containing SDN data, control, and an application plane. IoT devices use services to offload their compute and time-consuming tasks in the data plane. Based on this architecture, customized northbound apps are hosted on the northbound application plane. In addition to end-to-end caching service orchestration, control mechanisms are determined by virtualized applications [43]. The data plane, the control/management plane, and the application plane comprise the system architecture of the edge environment based on SDN. A data plane abstracts underlying network resources such as switches as part of the framework. The global SDN controller's primary responsibility is to manage the local SDN controller in the control/management plane.

By sharing operational statuses, an SDN controller changes its data plane rules. A client entity represents an application in the application plane.



**Fig. 2** **A** Architecture levels of SDN-Edge IoT, **B** e-health IoT protocol stack

The proposed three-layer structure of e-Health IoT architecture depicts in Fig. 1. It includes the device layer, edge layer and cloud layer as explained in the following [13, 42, 43].

## 5.1 Device layer

A continuous health monitoring system must have sensors operating round-the-clock. The utilization and integration of diverse biomedical data can improve clinical research, point of care, and clinical practice. In addition to real-time health data monitoring from any computer or mobile device, these smart IoT medical devices allow users to sync their data with cloud-based e-health platforms. All they have to do is connect to a gateway or an edge node through an appropriate communication protocol. The IoT e-health protocol stack is depicted in Fig. 2B. Application and individual use cases determine the near-optimal communication and connection protocol. The concepts related to e-health IoT devices can be explained as follows [2]:

- Sensors include physical and virtual sensors.

**Physical sensor:** With the use of any medical gadget that has a wired or wireless interface, an e-health system may track patients' physical well-being and digitally monitor their health. Among these are ECG/EKG monitors, monitors for blood pressure, body temperature, and glucose, pulse oximeters, hemoglobin monitors, activity monitors, smart shoes, smart garments, sleep monitors, knee sensors, skin conductance sensors, elderly monitors, medication management, food contamination detection, and early warning systems.

**Virtual sensor:** these sensors gather health information on patients as well as context about their environment through the use of software, mobile apps, and e-health services. Remote monitoring, remote consultation, medical reference, nutrition,

diagnostics, and patient health records are just a few of the various types of virtual sensors.

- **Registration:** the registration module allows physical devices to join or leave the network dynamically. In order to send and receive messages, registration is required. Devices should send registration requests to edge layers as an initial message. For authentication, devices are given a unique ID and key during the registration process.
- **Collecting data and executing commands:** data is collected by registered sensors and transferred to the edge layer. This module runs the commands received from the edge layer through actuators.

The integration of IoT in the healthcare sector has led to improved quality of life, decreased cost of care, and enhanced user experience. IoT e-health offers a comprehensive solution to health, beauty, safety, and fitness needs, seamlessly integrating different technologies, analyzing and processing data, providing customizable content and services, offering continuous monitoring, and being user-friendly. The benefits also include the reduction of costs, larger role for doctors, and global collaboration [44]. The recent trend in healthcare shows the growth and emergence of electronic medicine with IoT technology, and the advancements in cloud technologies are facilitating this progress.

## 5.2 Edge layer

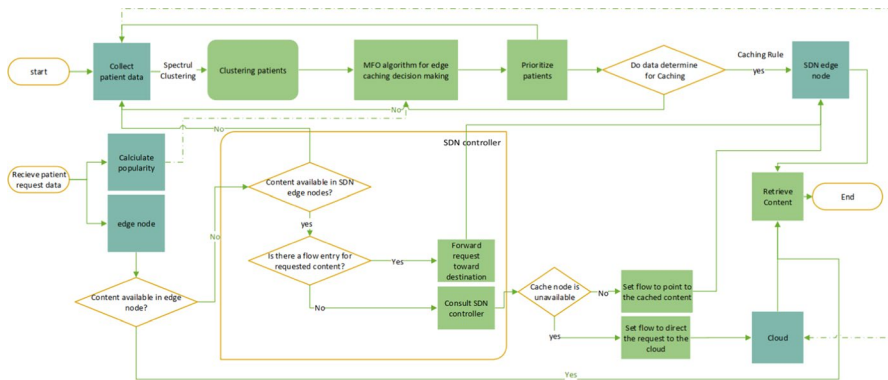
Latency is a major issue for e-health applications in the IoT. Time-sensitive data and circumstances must be analyzed and acted upon. As a result, the traditional cloud model and architecture cannot be used for analyzing sensitive bio signals, vital signs, and medical records spanning a wide specific region. A decision-making procedure that must be made quickly and with consideration of time-sensitive facts are performed on the edge nodes in the proposed e-health IoT platform. Patients and medical devices produce data closest to these edge nodes. Our main storage and computing resources, however, are the cloud. As depicts in Fig. 1 architecture, an edge node has the following characteristics [4, 42]:

- **Network connectivity:** multiple hardware interfaces and protocols are supported by edge nodes. Basically, they can communicate with any e-health device or cloud platform. Through the edge nodes, reliable and secure connections can be established through devices, networks, and across networks. This includes networking analytics, protocol translation, security, switching, and routing. By converting incoming data to the correct format, edge nodes are able to send them between different networks.
- **Exchange of data:** there is a bidirectional data flow at edge nodes. In order to store and analyze the aggregated medical data for long periods of time, the edge node transfers the data to the cloud for prolonged storage on a regular basis. The

cloud also sends commands and configuration data to it. It is important to ensure that edge nodes can be reached from the cloud.

- Translating and integrating protocols: data is sent in real-time via sensors and medical equipment to edge nodes. In spite of this, several medical tools and gadgets are available from multiple manufacturers. To be compatible with such a variety of medical equipment, these edge nodes include a multi-standard interface.
- Data annotation interoperability: in order to integrate, exchange, and retrieve electronic health records (EHR), various organizations developed different standards, such as HL7(set of international standards for the exchange, management, and integration of electronic health information between different health-care systems, devices, and applications) [45]. Health data can be processed and re-formatted by edge nodes at any time. It consists of:
  - Aggregation, filtering, formatting, encoding, and decoding: Various sources of data are incoming, including ECGs, weighing scales, blood pressures, glucose levels, pulse oximeters, etc. Medical data collected at edge nodes is filtered, aggregated, compressed, and formatted in order must be compatible with other resources and accessible.
  - Database for short-term use: incoming data can be stored on the node for a short time. The short-term database at the edge is an essential component of our proposed architecture. It is needed to store recent data samples generated by IoT devices, allowing for real-time processing and reducing latency. The short-term database serves as a buffer that temporarily stores data samples before sending them to the cloud for long-term storage.
  - Protection of data and security: to completely adhere to FDA standards for patient data protection, edge nodes provide multilayer security for authentication, encryption, and access control.
  - Assessing and notifying: incoming raw data can be evaluated and processed by edge nodes to determine whether alerts or notifications need to be tagged. Event generation and event transformation can be involved in this process.
  - Processing at the local level: health applications and edge nodes can execute a rich set of applications based on their requirements. Time-series data mining, lightweight signal processing, event processing, automated reasoning, feature extraction, etc., are all included.

Edge computing is becoming increasingly important in healthcare due to the multidimensional, high-velocity, and diverse nature of health-related data generated by sensors and devices. By creating an additional layer of computing between sensors and the cloud, edge computing can overcome bandwidth and time limitations. It enables the implementation of edge computing in e-health IoT systems, and its capabilities include advanced techniques, distributed storage, and notification services. Edge computing can simplify data management and processing, enable real-time and online analysis, reduce latency, and offload traffic to the edge from the cloud, thus lowering network latency and reducing bandwidth usage. It is not an alternative to cloud computing, but rather develops cloud computing closer to data-generating IoT devices at the network's edge. Critical information, rather than raw data, is sent



**Fig. 3** Proposed methodology for composition of caching and classification in edge computing for SDN-based IoT healthcare solutions

to the cloud, edge, or network resource identified as the best place for processing incoming data based on the speed of decision-making.

### 5.3 Cloud layer

Modules of the cloud layer include long-term storage, decision-making on a global scale, compression, and encoding. Detailed descriptions of each are provided in this section.

- **Long-term storage:** this module stores data from different edge zones permanently.
- **Decision-making on a global scale:** during data processing, feedback is sent to lower layers and useful data is permanently stored. Whenever data is collected and analyzed, patterns are identified, and knowledge is discovered from it.
- **Compression and encoding:** having these modules at both the cloud and edge layers is essential for privacy and efficiency concerns. Both parties must agree on encryption and compression approaches.

Figure 3 shows the main steps of our proposed approach. The first step involves collecting patient data from e-health IoT devices. Next, patients are clustered based on their healthcare needs using spectral clustering. Traffic flows are then categorized based on their QoS requirements using flow classification. The SDN controller implements caching to store frequently accessed data at the network edge, and a smart caching mechanism is used to determine which items need to be cached and which edge node should cache them. Patient data is given priority to increase the cache hit ratio for frequently accessed information. The MFO algorithm is used for caching multiple patients' medical records based on decision-making with respect to multiple criteria, such as illness severity and treatment policy. Finally, the proposed system's performance is evaluated through simulations and experiments to demonstrate its effectiveness in improving QoS in SDN-IoT healthcare solutions.

## 6 Methodology

Several communication challenges have arisen with the healthcare ecosystem in the recent COVID-19 pandemic situation, particularly in developing countries (Sect. 8). Patients' conditions are not being diagnosed on time due to a lack of adequate hospital accommodations, resulting in unnecessary health deterioration. In SDN, the data plane is decoupled from the control plane, allowing users to configure the controller for their own needs. The ultimate goal of all healthcare providers is to achieve high levels of patient satisfaction by promoting patient care. One solution is to use mobile devices like fitness trackers, smartphones, and sensors/actuators that are a part of the IoT [46]. The purpose of a medical center server is to enable healthcare facilities and medical professionals to quickly and effectively examine patient data. In addition, the server can serve as a tool for managing, organizing, and supporting telemedicine professionals. For telemedicine to be successful on a long-term basis, a remote medical center server is crucial. The purpose of a medical center server is to allow hospitals and physicians to analyze patient data in real time and deliver appropriate care. In addition, the server could serve as a tool for managing, organizing, and supporting telemedicine professionals. For telemedicine to be successful on a long-term basis, a remote medical center server is crucial.

In order to transmit data effectively at the edge layer, it must be adaptively tuned. In monitoring a chronically ill patient, detecting abnormal signs may require the system to boost the number of requests (priority). Various parameters and services are prioritized to determine the frequency of cloud transmissions. For instance, it is more important to prioritize rates of patient data transfer to the cloud who are suffering from chronic conditions or acute illnesses. The adaptive edge computing process improves the sensitivity and specificity of critical parameters, thereby enhancing the performance of the system.

By hosting and caching e-health IoT contents at the network edge for monitoring patients' daily lives, closer to the data consumers, such as physicians, nursing staff, doctors, medical students, office managers, and accountants, and so on; each individual would need specific information. Accountants, for example, are more concerned about invoicing details and other financial information about patients than doctors in the oncology department. Depending on their role, each user type will have its own interest space. Consequently, the user queries the same data repeatedly. It may be possible to leverage the user's workload for faster results. Caching is a good way to enable quicker retrieval of frequently used data. We may need to consider a number of aspects about caching in real-time IoT scenarios, remote healthcare, for example. Transmission delays might be reduced and patient care could be given more effectively and efficiently by utilizing the proposed solutions. The proposal can take advantage of the native SDN programmability for building routing paths and storing more effectively, as well as the built-in SDN routines for monitoring network status, by orchestrating edge caching using the SDN Controller.

To enable SDN edge nodes to cache based on the MFO-Edge caching algorithm on a regular basis, we suggested using a network application that implements a

caching strategy for caching and identifying which IoT e-health content ought to be stored in edge nodes' cache. Given these algorithms and the transient nature of particular contents in e-health IoT systems, it is significant to keep them in mind. Also, edge nodes are chosen based on their proximity to the cluster. The ability to prioritize the most crucial information and remove the less crucial information from the cache is necessary due to the confined cache capacity. The e-Health IoT data flow, on the other hand, may be separated into three groups: emergent traffic, normal traffic, and on-demand traffic.

- Emergent traffic, which also contains significant data packets, is defined as traffic that is started when a typical threshold is surpassed by nodes.
- Periodic traffic is referred to data flow that is uncritical and generated under typically healthy patient circumstances. This kind of periodic traffic includes medical information that must be provided on a regular basis or by a certain time.
- On-demand traffic, for diagnostic purposes, the clients want on-demand traffic in order to receive precise data. They are consistent with standard evaluations of patients' physiological data, which frequently reveal normal signals from those lower priority sensors.

On top of the softwarized edge architecture, we offer a caching technique that is implemented at the SDN Controller and makes use of the storage resources to detect what items need to be cached, and which edge node should do it:

- First of all, the clustering of the patients with e-health IoT devices is performed using spectral clustering, which takes into account their similarity and distance, and then each cluster is connected to the SDN edge nodes for efficient caching. We calculated similarities at the feature level, then combined them at the patient level. By using spectral clustering technique clustering the e-health IoT users according to their features. Patient similarity was estimated by applying patient information or features (static and dynamic factors).
- The second step is giving the patient's data, greater priority and increasing the cache hit ratio for the most frequently accessed information. As a result, more requests can be made for the popular data during its validity period.
- To maximize content diversity, each content copy is only stored at the edge once by SDN controller stores only one copy of each content copy at the edge. To prevent intra-domain cache redundancy, edge storage resources can be used to cache a larger number of unique contents. By using such an approach, network operators also benefit from a reduction in traffic leaving the domain to access cloud storage for content. We allocate patients' medical records to the SDN edge node for caching which is set up appropriate to the patient's life, close to the closest locations.
- Additionally, we proposed using the MFO algorithm [31] to cache multiple patients' medical records. Essentially, MFO-Edge Caching involves decision-making with respect to multiple criteria, and in operations research, it is considered. Because they can increase the quality of decisions by using a method

that is more logical, explicit, and effective than traditional approaches, the usage of nature-inspired algorithms is becoming more popular.

- A medical care system under the proposed plan is fairer and more humane than the last one. MFO-edge caching algorithm can weight medical parameters based on illness severity and treatment policy in our architecture. The system, however, prioritizes patients who have more severe illnesses and reduces waiting times for them.

## 6.1 Selection of moth-flame optimization algorithm (MFO) for dynamic edge caching

In this section, we explain the use of the moth-flame optimization algorithm (MFO) [47] in more detail and provide a justification for its selection in this study. MFO is a bionic intelligent algorithm that has shown better performance in solving complex optimization problems compared to other similar algorithms, such as the grey wolf optimization (GWO) [48] and particle swarm optimization algorithm (PSO). We chose these algorithms for comparison because they have been used in previous works for data classification in IoT networks [49–53]. The selection of MFO in this study is based on its ability to effectively solve optimization problems, specifically in selecting the best cache content. We will first provide a brief overview of the MFO algorithm, followed by an explanation of how it was used in this study.

The moth-flame optimization (MFO) algorithm is a relatively new optimization algorithm inspired by the behavior of moths and flames. It was first introduced in 2015 by Mirjalili and Lewis. The algorithm works by dividing the optimization process into two stages: the exploration stage and the exploitation stage. In the exploration stage, the algorithm searches for new regions of the search space by randomly generating new solutions. In the exploitation stage, the algorithm focuses on exploiting the best solutions found so far in order to converge toward the optimal solution.

MFO, an optimization method that imitates moth flying patterns, is one of the most potent ones. During the night, moths attempt to maintain a constant position with respect to the moon [43]. Navigation is based on a transverse orientation. The method is, however, not always useful, especially if the distance to the light source is quite great. In irregular lighting, moths attempt to preserve analogous forms that pass it in a straight line. Although the moths are closer to this light source than the moon, retaining a comparable angle to the source of light causes a dangerous or inefficient spiral flight route for them. A killing flow such as this is used to uncover optimization issues in real-world settings since the origin of light is close by. The exploration space coordinate vectors of a moth are used as variables in this method [47]. In this paper, for optimizing caching, the MFO equation is shown in Eq. (1).

$$\text{flame}_{\text{no}} = \text{round}\left(N - l * \frac{N - l}{\text{Max}It}\right) \quad (1)$$

**Algorithm 1. Moth-flame optimization (MFO) algorithm**

```

1: Begin
2:   Create a random population of moth and calculate the objective function
3:   Create a set of flames same as moth solution
4:   Update the position of the moths
5:   Change the flame size
6: End
7: Return the best solution

```

**Fig. 4** Moth-flame optimization (MFO) algorithm

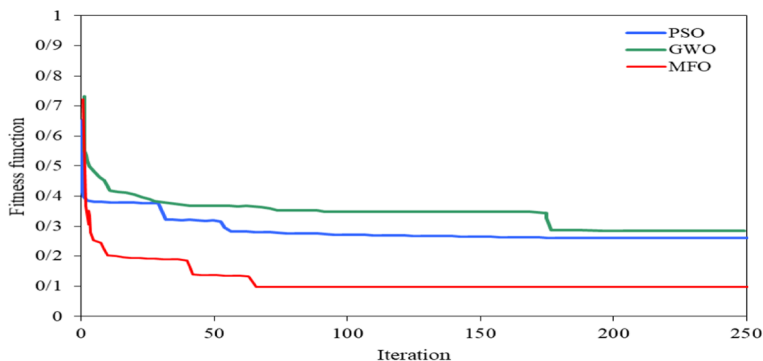
The number of repetitions indicated by  $MaxIt$  is the current number. The number of flames is  $N$  represented by the highest quantity, and the number of repeats is  $l$  indicated by the largest quantity.

Algorithm 1 summarizes the moth-flame optimization (MFO) algorithm in Fig. 4.

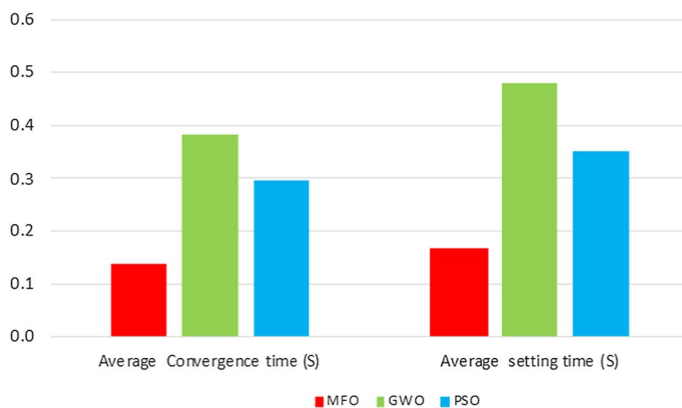
The MFO algorithm has several advantages that make it suitable for use in content selection in edge caching. First, MFO is a global optimization algorithm, which means that it is capable of finding the global optimum of a problem, rather than getting stuck in a local optimum. This makes MFO a powerful tool for content selection, as it can effectively search a large space of possible solutions to find the best content to cache at the edge. Second, MFO is a population-based algorithm, which means that it maintains a population of candidate solutions and iteratively refines them to improve their quality. This makes MFO more robust than other optimization algorithms that rely on a single solution. We chose MFO due to its ability to efficiently optimize multiple objectives, which is well suited for our proposed caching strategy that aims to minimize both latency and energy consumption. In addition, MFO has been shown to have better performance compared to other bionic intelligent algorithms such as GWO and PSO in terms of convergence time and setting time.

In terms of convergence and setting time, MFO has been shown to outperform GWO and PSO in optimization problems in our study. MFO has a fast convergence rate, which means it can quickly find the optimal solution with fewer iterations compared to GWO and PSO, Fig. 5. Moreover, MFO has shown good performance in terms of setting time, which is the time required to find the optimal solution. MFO is also less sensitive to the initial parameters, which means it is more robust and stable than other algorithms, Fig. 6. However, it is worth noting that the performance of MFO is highly dependent on the problem being solved, and it may not always be the best choice for all optimization problems.

In terms of caching, for small-scale systems, there is no significant difference between MFO, GWO, and PSO. However, as the scale of the system increases, caching algorithms may only be able to produce feasible solutions that are near optimal. In this case, MFO has been shown to outperform other algorithms in terms of finding the optimal solution. MFO has a high exploration ability, which means it can explore a large search space and find the near-optimal solution quickly. On the other hand, GWO and PSO may struggle with large-scale systems due to their limited exploration ability. Therefore, the choice of optimization algorithm may



**Fig. 5** The convergence comparison of MFO, PSO and GWO



**Fig. 6** Comparison average convergence and setting times (S) MFO, PSO and GWO in proposed approach

depend on the specific requirements and constraints of the IoT healthcare system being considered.

## 6.2 Caching mechanism in SDN-based IoT healthcare system

In our proposed solution, the caching process is carried out by the SDN controller, which has a global view of the network topology and traffic flow. The controller uses this information to dynamically route traffic and manage network resources, including caching data at the edge node. When a data packet arrives at the SDN controller, it is inspected by the controller to determine if it should be cached. This decision is based on the set criteria, which include the type of data, the source and destination of the packet, and the current network conditions. If the data packet meets the caching criteria, the controller generates a flow table entry that includes the cache action for the edge node. The flow table entry is

then installed on the appropriate switch along the path to the edge node. When subsequent data packets matching the flow table entry arrive at the switch, they are forwarded to the edge node for caching. The cached data can then be quickly retrieved by other devices on the same network, reducing latency and improving QoS.

To manage the cache, the SDN controller uses OpenFlow's flow modification messages to update the flow table entries and add or remove cached data. The controller periodically checks the cache usage and evicts old or infrequently accessed data to make room for new data. This dynamic management of the cache ensures that it is always up-to-date and optimized for efficient data retrieval. While OpenFlow does not provide explicit support for caching, its flexible architecture allows the SDN controller to configure and manage caches through flow table entries and flow modification messages. Our proposed solution leverages these capabilities to optimize the caching process and improve the performance of e-health IoT applications.

### 6.3 e-Health IoT clustering strategy

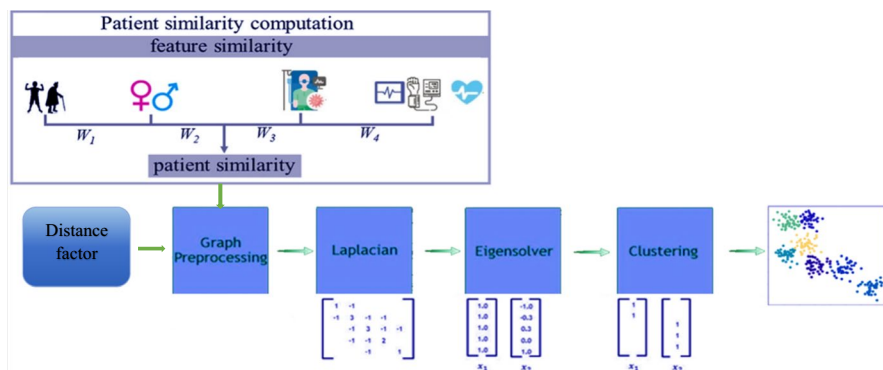
A design of electronic medical records for continuous-care patients is necessary to meet the ever-increasing demands of healthcare, and to assist patients after hospital discharge who require continuous care. To map the e-health IoT users to specific SDN edge nodes, we first use spectral clustering to identify the similarities among the users based on their features. Using both static and dynamic patient information, the similarity between patients was estimated. In order to calculate similarity at the patient level, the features were first compared and then combined into a single measure of similarity [54]. This will significantly reduce the time needed to search for data on these SDN edge nodes when they cache only data relevant to a specific cluster.

Spectral clustering is a clustering algorithm that uses the spectrum (eigenvalues and eigenvectors) of a similarity matrix to group data points into clusters [55]. The idea behind spectral clustering is to transform the data into a new space where the clusters are more easily separable.

Here's a simplified explanation of how spectral clustering works:

- Construct the similarity matrix: Compute the similarity between all pairs of data points and create a symmetric similarity matrix.
- Compute the Laplacian matrix: The Laplacian matrix is a measure of how the data points are connected in the graph.
- Compute the eigenvectors and eigenvalues of the Laplacian matrix
- Perform clustering: Use the eigenvectors and eigenvalues to perform k-means to clustering of the data points.

Overall, spectral clustering is a powerful tool for patient clustering based on e-health IoT data and edge computing, and its advantages make it a useful method



**Fig. 7** Proposed e-Health IoT clustering strategy

for healthcare professionals and researchers looking to analyze and understand large and complex patient datasets while maintaining privacy and reducing latency. On a graph, the spectral clustering procedure is shown in Fig. 7. Additionally, the graph is created by using patients who have e-Health IoT devices to represent vertices, and the edges are calculated by measuring the degree of integrity between the vertices, which in turn is affected by their similarity and Euclidean distances [56].

Vital signs are measured by a variety of medical sensors. They can be used in hospitals, homes, and other settings to monitor and identify medical issues. In telemedicine and remote health monitoring, sensors play an essential role [57]. EHR items are assigned based on vital signs; most triage systems now utilize a four-level color-coded scale, with red indicating high risk, orange indicating most urgent, yellow indicating sick but not urgent, and green indicating minor, normal). In chronic and acute diseases, patient prioritization is essential. Using a clustering system based on the emergency status of patients, we propose a prioritization system and clustering of patients [1].

In simple terms:

**Blood pressure:** The ratio of the systolic and diastolic forces exerted by the heart during its circulation in the arteries.

**Body temperature:** Determines the body's temperature.





**Heart rate:** pulse rate, measured in beats per minute. It is also important to assess the level of regularity, strength, and equality.

**Respiratory rate:** count how many times the chest rises and falls in sixty seconds.

**Oxygen saturation:** Determines the amount of oxygen in the blood by measuring peripheral capillary oxygen saturation, or SpO2.

According to Table 2, the system has four parts [1]. According to our assumption, the registered hospital (as cloud) has the entire electronic medical record (EMR) of each patient. Wearable sensors record patients' locations and monitor their health status. Wearable sensors serve two main purposes. Detecting emergency situations (e.g., high-risk) is the first step. A patient's body may produce an electronic impulse that is higher or lower than the critical value in an emergency situation. A hospital

**Table 2** Healthcare EHR items and levels

| Parameters          | Unit        | High risk  |        | Urgent  |         | Sick  |           | Normal  |
|---------------------|-------------|---|--------|--|---------|--|-----------|--|
|                     |             | Low   | High   | Low  | High    | Low  | High      |  |
| Heart rate (HR)     | Beats/min   | <= 40   | >= 131 |  | 111–130 | 41–50  | 91–100    | 51–90  |
| Blood glucose       | mg/dL       | <= 20   | >= 600 | 20–69  | 300–600 | 70–99  | 163–299   | 100–162  |
| Oxygen saturation   | %           | <= 91   |        | 92–93  |         | 94–95  |           | >= 96  |
| Body temperature    | °C          | <= 35   | >= 40  |  | 39.1–40 | 35.1–36.0  | 38.1–39.0 | 36.1–38.0  |
| Blood pressure (BP) | mmHg        | <= 90   | >= 220 |  | 91–100  | 101–110  |           | 111–219  |
| Respiratory rate    | Breaths/min | <= 8  | >= 25  |  | 21–24   | 9–10   |           | 12–20  |

and clinical professionals would then receive an emergency alert from the sensor. Medical services can be provided to patients immediately. Emergency situations are communicated to the patient's caregivers and family [1]. In many cases, the patient would also receive notifications if they needed help taking their medication. A clinical decision can be made remotely if a patient's status needs to be monitored. It is also used to record the location of patients. The caregiver or family member receives a notification regarding the subject's location through SMS, email, or messages.

Wearable sensors are used to record patients' health status. Ideally, the caching node will be located near the patient's home, workplace, or the nearest hospital to the patient. As the cloud (the registered hospital), the EMR of the patient is updated accordingly in it. Medical records would be cached on the SDN edge node. Clinical professionals are able to retrieve medical records quickly because of where the edge caches are located (close to hospitals). Because edge caches have limited storage capacity, each edge cache suggests selecting medical files that are hospitals closest to homes are ideal and most significant, workplace, and others. Ideally, physicians should be able to check patients' health without needing to ask their hospital for it. In the next section, we present a method for caching medical files.

## 6.4 Static features

$PS_{static}$  represents the patient's profile information, such as the patient's age, gender, chronic or acute disease, and EHR items. The details are as follows [34]:

### i. Feature similarity for age

As shown in Eq. (2), the patient  $i$ 's age and patient  $j$ 's age are denoted by  $Age_i$  and  $Age_j$ . A ratio of a smaller age to a larger age can be used to represent feature similarity ( $F_{age}$ ) for age [58]:

$$F_{age}(i, j) = \frac{\min(Age_i \cdot Age_j)}{\max(Age_i \cdot Age_j)} \quad (2)$$

### ii. Feature similarity for gender

As shown in Eq. (3), patients  $i$  and  $j$  were compared by the similarity feature  $F_{gender}$ , which we defined as 1 if their genders were the same and 0 if not [58].

$$F_{\text{gender}}(i \cdot j) = \begin{cases} 1. & \text{if } i \cdot j \text{ had same gender} \\ 0. & \text{otherwise} \end{cases} \quad (3)$$

### iii. Feature similarity for chronic or acute disease

Events like patients having a chronic or acute condition were represented as a Boolean value, along with other static attributes. When a patient had diabetes, for instance, we determined that if patient  $i$  and patient  $j$  had the same condition—either both had the disease or neither—the similarity feature  $F_{\text{chronic}}$  between them would have been 1 and 0, respectively, as shown in Eq. (4).

$$F_{\text{chronic}}(i \cdot j) = \begin{cases} 1. & \text{if } i \cdot j \text{ had chronic or acute disease} \\ 0. & \text{otherwise} \end{cases} \quad (4)$$

As shown in Eq. (5),  $W_j$  stands for the weights associated with the three feature similarities. In the current investigation, the experimentally allocated in [58] values for  $w_1 \cdot w_2 \cdot$  and  $w_3$  were, in that order 0.4, 0.4, 0.2.

$$PS_{\text{static}}(i \cdot j) = w_1 * F_{\text{age}}(i \cdot j) + w_2 * F_{\text{gender}}(i \cdot j) + w_3 * F_{\text{chronic}}(i \cdot j) \quad (5)$$

### iv. Dynamic feature, similarity for EHR items

EHR data was used to extract the dynamic data part of the feature Eq. (10). All  $m$  items in the EMR for each patient were continuous. For the subsequent computation, they were first normalized to  $x_i \in [0, 1]$ , where  $x_i$  stands for the patient  $i$ 's normalized feature. Calculations must be made for every pairwise patient to find the feature similarity between patients  $P_i$  and  $P_j$  ( $i, j \in 1, \dots, n$ ). According to Eqs. (6), (7) and (8), the dynamic feature similarity for EHR items ( $F_{\text{EHR}}$ ) was calculated as Eq. (9), 1 minus normalized min–max (Eq. (8)) [58]. The patient vectors  $P_e = \{P_1, P_2, \dots, P_n\}$  each have  $m$  features:  $\{x_1, x_2, \dots, x_m\}$ . These features are represented in an  $m$ -dimensional vector space.

$$\text{Patients}(x) = \begin{bmatrix} p_1 & \cdots & p_n \\ x_{i-1} & \cdots & x_{n-1} \\ \vdots & \vdots & \vdots \\ x_{m-1} & \cdots & x_{n-m} \end{bmatrix} \quad (6)$$

As a result, the patient set has  $n$  patients, whereas the feature set has  $m$  attributes.

$$d_{\text{EHR}}(i \cdot j) = \sqrt{\sum_{k=1}^m (x_{i-k} - x_{j-k})^2} \quad (7)$$

$$d' = \frac{d_{\text{EHR}}(i \cdot j) - \min(d_{\text{EHR}})}{\max(d_{\text{EHR}}) - \min(d_{\text{EHR}})} \quad (8)$$

$$F_{\text{EHR}}(i \cdot j) = 1 - d' \quad (9)$$

$$\text{PS}_{\text{dynamic}}(i \cdot j) = F_{\text{EHR}}(i \cdot j) \quad (10)$$

### • Patient similarity factor

As shown in Eq. (11), to measure weighted patient similarity (PS) between patient  $i$  and patient  $j$ , the weighted sum of the two similarities factors was used [58]:

$$\text{PS}(i \cdot j) = 0.9 * \text{PS}_{\text{static}}(i \cdot j) + 0.1 * \text{PS}_{\text{dynamic}}(i \cdot j) \quad (11)$$

### • Distance factor

Additionally, as shown in Eq. (12), the Euclidean Distance is used to determine how far the patient's IoT-enabled e-health equipment is from the SDN edge nodes.

$$\Delta(p_i \cdot p_j) = \sqrt{(x_i^2 - x_j^2) + (y_i^2 - y_j^2)} \quad (12)$$

where  $(x_i, y_i)$  represents the  $i$ th user's position and  $(x_j, y_j)$  represents the  $j$ th user's position.

As a result, based on how similar the patients are and the distance between them, the degree of integrity  $\sigma$  between them is determined. In addition, the threshold value and degree of integrity value are compared (the optimal threshold for the degree of integrity is 0.075 [55]). Upon determining that the value exceeds the threshold value, an edge between the two patients is created, forming the vertices of the graph. The degree of integrity is assessed by the weight of the edges. After the graph is created, given that the weight matrix ( $W$ ) derives its value from the weight of the edges and that the degree matrix ( $DM$ ) requires a specification in order to be generated, the weight matrix ( $W$ ) and the degree matrix ( $DM$ ) are computed. This definition is as Eq. (13):

$$\text{DM}_i = \frac{1}{m} \sum_{i=1}^m \text{PS}(i \cdot j) \quad (13)$$

The degree matrix is calculated accordingly based on Eq. (14):

$$\text{DM}_{i,j} = \begin{cases} \text{DM}_i, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

In this case, the degree matrix will be calculated as follows:

Using the Degree Matrix and Weight Matrix, we calculate the Laplacian matrix ( $L$ ), which is the difference between the two matrices, as shown in Eq. (15). For mathematical efficiency, we further normalize Laplacian Matrix using Eq. (16)

**Algorithm 2: Patient clustering pseudo-code**

```

1:  $\text{th}$ : threshold value of integrity
2:  $k$ : Number of clusters to be formed.
3:  $A$ : Matrix of eigenvectors (patients).
4: Begin
5: For each patient $_i$  (with IoT devices) do
6:   For each patient $_j$  (with IoT devices) do
7:     calculate  $(i,j)$ 
8:   end for
9: end for
10: if  $(i,j) > \text{th}$  then
11:   Draw an edge between the vertices
12: end if
13: Compute Weight Matrix (WM) and the degree matrix(DM) of graph formed
14: Compute Laplacian Matrix ( $L$ )
15:  $L = DM - PS(i,j)$ 
16: Make a matrix  $A$  of eigen vector as columns of  $A$ .
17:  $k$ -means clustering technique used to make  $k$  clusters of rows of  $A_i$  ( $A_i = \{i \in n\}$ ).
18: Output: Clusters  $\{\text{cluster}_1, \text{cluster}_2, \dots, \text{cluster}_k\}$ 
19: End

```

**Fig. 8** The pseudo-code of patient clustering algorithm

$$L = (DM_{i,j}) - P_S(i,j) \quad (15)$$

$$L = (DM_{i,j})^{-1/2} L DM_{i,j}^{-1/2} \quad (16)$$

Furthermore,  $k$  eigenvalues are chosen for  $k$  clusters as well as the corresponding vectors. The eigenvectors are considered columns in the matrix.  $K$  centroids are found using the  $k$ -means clustering algorithm, and each node is then assigned to a cluster depending on how close it is to the centroids. The degree of integrity is measured as shown in Eq. (17):

$$\sigma(i,j) = \frac{PS(i \cdot j)}{\Delta(p_i \cdot p_j)} \quad (17)$$

Algorithm 2 summarizes the method as demonstrated in Fig. 8.

## 6.5 Proposed e-health IoT data caching strategy

There are two phases to the proposed caching strategy: the decision phase and the replacement policy phase. First the decision to cache each piece of data will be made taking into account factors like freshness of data latency of sensing the content popularity and IoT node energy. In order to keep the most important data for a specified amount of time in the cache. The edge nodes assign a value based on essential characteristics of each data point. Those contents that exceed a predetermined threshold will be cached. The candidate data item will be compared against the stored contents once the cache is full. A periodic evaluation of all the cache contents will also take

**Table 3** Weight of attributes

| Attributes      | Emergency request $l_v$ | Periodic request $l_v$ | On-demand request $l_v$ | $\omega_{\text{Emergency}}$ | $\omega_{\text{Periodic}}$ | $\omega_{\text{On-demand}}$ |
|-----------------|-------------------------|------------------------|-------------------------|-----------------------------|----------------------------|-----------------------------|
| $F_{pp}$        | 5                       | 5                      | 7                       | 0.14                        | 0.14                       | 0.36                        |
| $F_D$ freshness | 9                       | 7                      | 3                       | 0.46                        | 0.36                       | 0.06                        |
| $F_{SD}$        | 7                       | 5                      | 3                       | 0.24                        | 0.14                       | 0.06                        |
| $F_{ELP}$       | 1                       | 3                      | 7                       | 0.02                        | 0.06                       | 0.36                        |
| $F_{HDP}$       | 5                       | 5                      | 5                       | 0.14                        | 0.14                       | 0.15                        |

place depending on the different applications. It will dynamically remove expired or invalid content from the cache if it is invalid.

Caching decisions are based on the attributes of the content navigating through each edge node. According to the healthcare application, each of these types of cached data has a different Quality of Information [59] value. In order to determine a cache's quality of information, we select the health data popularity (HDP), energy level parameter (ELP), delay ( $D$ ),  $D_{\text{freshness}}$  (DF), patient priority based on the severity of illness (PP) whose combined value determines its QoI. Table 3 shows that we associate priorities with each of these attributes for each request type and that these priorities determine the importance of each attribute's true value.

We suggest using a method that to use an algorithm that satisfies the edge caching problem as a nature-inspired paradigm named "MFO-EC." It is used to select the best option content for edge caching using multi-criteria moth-flame decision-making, as described in the pseudo-code of the MFO-EC algorithm. Whenever the MFO-EC algorithm selects content for caching, the controller triggers caching actions in the SDN edge nodes and a message OF PACKET OUT is sent. The selected edge nodes receive an action type, known as caching, via the Experimenter instruction, which is stored locally. When the periodically run MFO-EC algorithm determines that the requested content should be stored, an SDN edge node is located. Whenever the cache node is unavailable, the cloud must be accessed to retrieve a new copy of the requested data. If the route is not already available in the edge nodes' flow tables [2], the controller sets up the route using OF FLOW MOD messages. As the SDN controller is expected to have a broader range of processing capabilities, so it will be able to perform content selection periodically at appropriate times (Fig. 2A).

#### i. Health Data Popularity (HDP)

When many users request a data item within a short amount of time. It is considered popular. To ensure efficient data distribution, popular content should be retained for a longer amount of time in the cache. In terms of popularity, Total Requests is the total number of requests that a node has received over a given period of time and  $\text{Request}(\text{data}_q)$  is the total number of requests for the  $\text{data}_q$ , as shown in Eq. (18). Assuming Timestamp, a period of time from  $T_0$  to  $T_1$ , the SDN edge nodes in the network calculate the popularity of every content by counting each content request's number and store the results in the edge node's table. The number of

historical times  $\text{data}_k$  is requested is  $\text{Request}(\text{data}_q)$  by using the OF Experimenter extension fields [60].

$$\left| \text{Request}(\text{data}_q) \right| = \left| \text{Req}_{\text{Emergency}}(\text{data}_q) + \text{Req}_{\text{Periodic}}(\text{data}_q) + \text{Req}_{\text{on-demand}}(\text{data}_q) \right| \quad (18)$$

$$\lambda_n^{\text{req}}(\text{data}_q) = \frac{\sum_{T_0}^{T_1} \left| \text{Request}(\text{data}_q) \right|}{\text{Timestamp}} \quad (19)$$

During the timestamp, the current popularity of content  $\text{data}_k$  may be characterized as Eq. (19) [2]. We must normalize these numbers due to the differing scale of  $\lambda_n^{\text{req}}(\text{data}_q)$ . according to Eqs. (20), (21) and (22); we use the following formulae to do so [19]:

$$P_{(\text{data}_q)}^{\text{req}+} = \frac{\text{Max} \lambda_k^{\text{req}}(\text{data}_q) - \text{Min} \lambda_k^{\text{req}}(\text{data}_q)}{\text{Max} \lambda_k^{\text{req}}(\text{data}_q)} \quad (20)$$

$$P_{(\text{data}_q)}^{\text{req}-} = \frac{\text{Min} \lambda_k^{\text{req}}(\text{data}_q)}{\text{Max} \lambda_k^{\text{req}}(\text{data}_q)} \quad (21)$$

$$F_{\text{HDP}} = P_{(\text{data}_q)}^{\text{req}-} + \frac{(\lambda_n^{\text{req}}(\text{data}_q) - \text{Min} \lambda_n^{\text{req}}(\text{data}_q)) (P_{(\text{data}_q)}^{\text{req}+} - P_{(\text{data}_q)}^{\text{req}-})}{(\text{Max} \lambda_n^{\text{req}}(\text{data}_q) - \text{Min} \lambda_n^{\text{req}}(\text{data}_q))} \quad (22)$$

where  $n = 1 \dots k$ .

$F_{\text{HDP}}$  is the calculated normalized value of metric popularity for  $\text{data}_q$  on Edge node  $k$ , and  $P_{(\text{data}_q)}^{\text{req}+}$  and  $P_{(\text{data}_q)}^{\text{req}-}$  are the upper and lower bounds of these values, respectively.

## ii. Data freshness (DF)

An e-Health IoT application that relies on caching collected data would need to have high data freshness, but at the same time be able to respond quickly to variations in a patient's heart rate and blood pressure etc. A key attribute that supports the stringent quality-of-service requirement is data freshness and reliable data delivery. Data items are considered valid if they meet the freshness requirements of the application. To determine if a data item meets the desired freshness it will be evaluated when traversed to the edge node. Data freshness is calculated by Eq. (23).

$$F_{D \text{ freshness}} = \frac{\text{Time} - \text{data}_{\text{age}}}{\text{Time}} = \frac{\text{Time} - \sum_{n=1}^k (\text{data}_{\text{path}}(n) + \text{data}_{\text{caching}}(n))}{\text{Time}} \quad (23)$$

The symbol for the distance between the edge node and the source of the data is  $\text{data}_{\text{age}}$ . The data's lifetime is determined by Time.

### iii. Patient priority based on the severity of illness (PP)

Considering the parameter related to patients' priority according to their severity of illness (SOI) [61] enables to evaluate one of the factors for selecting whose e-health IoT data to cache. Each patient's SOI is determined by the diagnoses and procedures they undergo during their medical visits. Healthcare resources are consumed more frequently by patients with higher SOI (e.g., significant or extreme) in the same diagnosis-related group as those with lower SOI [13]. There are four levels of severity of illness as shows in Eq. (24). For comparison, we use the most recent priority level that a patient had. Medical records of patients with more severe illnesses have a higher priority for caching. This priority indicator is denoted by the letter  $\text{patient}_e$ .  $\text{patient}_e$  is calculated according to the equipment properties for a specific cluster. From SDN controller's registration module, the priority patient list is gathered. We assume that the patient priority factor value  $\text{patient}_e$  is as follows based on their severity of illness:

$$\text{patient}_{\text{priority}} = \begin{cases} 1. & \text{for SOI class type Minor lowest periority} \\ 2. & \text{for SOI class type Moderate medium periority} \\ 3. & \text{for SOI class type Major high periority} \\ 4. & \text{for SOI class type Severe highest periority} \end{cases} \quad (24)$$

For one cluster the priority of the patient can be obtained by Eq. (25):

$$\text{patient}_p = [\text{patient}_{p_1} \cdot \text{patient}_{p_2} \cdot \text{patient}_{p_3} \dots \text{patient}_{p_n}] \quad (25)$$

By scaling these values between zero and one, they can be normalized as follows as shown in Eq. (26) [19]:

$$F_{\text{pp}} = \text{patient}_{p_{\text{norm}}} (i) = \frac{\text{patient}_p(i) - \min(\text{patient}_n)}{\max(\text{patient}_n) - \min(\text{patient}_n)} \quad \forall i \in n \quad (26)$$

### iv. Sensing Delay (SD)

Sensing delay calculation in IoT healthcare systems involves measuring or estimating the time it takes for a sensor to detect a physiological signal, transmit it to a processing unit, and analyze it. The delay is affected by several factors, including the type of sensor, communication protocol used, and processing power of the device. To calculate the sensing delay, we need to consider the response time of the sensor, the transmission delay, and the processing delay. Once these factors have been measured or estimated, the sensing delay can be calculated by summing them up. The sensing delay is a measure of how much the consumer is willing to wait for the cached data content depending on the delay sensitivity. The use of delay-sensitive data is common in emergency situations (e.g., natural disasters and health emergencies). To capture accurate readings, sensors should be placed in different environments for varying periods of time. However, sensor nodes have a limited lifetime, so the duration of the on-time affects their lifespan as well. Longer storage of the sensed data can help the sensor node last longer if the time it takes to get the reading

is longer, which is known as sensory delay. Additionally, the propagation delay of the data would increase so that each time data was needed, it had to be transferred from sensor nodes to SDN edge nodes each time data was sought, especially if the sensor nodes were distant from the cluster head. Therefore, it's crucial to optimize the sensing delay and minimize the delay sensitivity in IoT healthcare systems to ensure timely and accurate detection of physiological signals, especially in emergency situations. As a result, the sensing delay  $\vartheta$  is taken into account Eq. (27).

$$\vartheta \propto \max (d_{s1}, d_{s2}, d_{s3}, \dots d_{sm}) \quad (27)$$

Each patient<sub>c</sub> is equipped with  $m$  sensors. The sensing delay of the sensor type  $i$  is indicated by  $d_{si}$ . Activated sensor types that deliver fresh data contribute the longest delay to the sensing delay. These value of  $F_{SD}(i)$  can also be normalized as follows by scalability between 0 and 1.

$$F_{SD}(i) = \frac{\vartheta - \min (d_s)}{\min (d_s) - \min (d_s)} \quad (28)$$

#### v. Energy level parameter (ELP)

As a result of sensor nodes' limited resources, data processing at sensor nodes has several disadvantages. The execution of complex algorithms at sensor nodes is sometimes possible, but at a high energy cost. By moving computation from the sensor nodes to the edge layer, which are typically not battery-powered. We can potentially address the aforementioned issues.

Energy (E) of an e-Health IoT node is indeed modeled as a parameter with a normalized value [0.1] (Eqs. 29 and 30), with 0 denoting the lowest level of energy and 1 denoting the highest level. Each patient is equipped with  $k$  sensors. Each device's energy consumption should be estimated so that the remaining energy levels can be adjusted accordingly. Each device's estimated energy consumption should be updated because sending and receiving data consumes energy.

$$\xi \propto \min (E_{s1} \cdot E_{s2} \cdot E_{s3} \cdot \dots \cdot E_{sK}) \quad (29)$$

$$F_{ELP}(i) = \frac{\xi - \min (E_S)}{\max (E_S) - \min (E_S)} \quad (30)$$

#### 6.5.1 Weight calculations for attributes

Using these normalized parameters, weights are examined to optimize caching decisions. Since transient data and resource constraints are among the main challenges of e-health IoT, we used the proposed method in [62] for calculating the weights for attributes. This method incorporates the concept of Weighted Product Model (WPM) to enhance the accuracy of the decision-making process. The role of each attribute will also differ depending on the application type. It is more important to

prioritize attributes related to the quality of the content ( $F_{D \text{ freshness}}$ ,  $F_{PP}$ ,  $F_{SD}$ ) than other attributes ( $F_{ELP}$  and  $F_{HDP}$ ) for emergent traffic. Conversely, for normal traffic and on-demand traffic,  $F_{ELP}$  and  $F_{HDP}$  attribute parameters are more significant than any content parameter [23]. As part of the evaluation criteria for caching, each edge node must recognize and apply this priority rule. The best way to accomplish this task is to calculate the attributes' weights. The computation is shown in Eqs. (31) and (32):

$$\omega_j = \frac{1}{m} \sum_{p=1}^m \frac{a_{jp}}{\sum_{q=1}^m a_{qp}} \quad (31)$$

$$a_{pq} = \begin{cases} 1 \cdot & \text{if } L_q = L_p \\ L_q - L_p + 1 \cdot & \text{if } (L_q - L_p) > 0 \\ \frac{1}{L_q - L_p + 1} \cdot & \text{if } (L_q - L_p) < 0 \end{cases} \quad (32)$$

The linguistic meanings  $l_v \{1.3.5.7.9\}$  describe the scale of significance (very low, low, medium, high, and very high) in order. For the pairwise comparison ' $a_{pq}$ ' of the quality's ' $p$ ' and ' $q$ ', if  $p$  is more important than  $q$  and the inverse of  $q$  is more important than that of  $p$ .  $L_v$  stands for the importance scale and  $m$  for the number of features,  $a_{pq} \{1/9.1/7.1/5.1/3.1.3.5.7.9\}$ .

### 6.5.2 Caching decision-making

Choosing the useful contents to cache is the main purpose of the caching decision. The range  $[0.1]$  for all attribute values is the same after modeling and normalization ( $F_{A(j)}$ ). According to Eqs. (33), (34) and (35), the Fitness function is often stated as follows in relation to the weight vector:

$$F_{A(j)} \in [F_{D \text{ freshness}}; F_{PP}; F_{SD}; F_{ELP}; F_{HDP}] \quad (33)$$

$$\sum_{p=1}^m \omega_j = 1 \quad (34)$$

$$\text{Fitness function } F_j = \sum_{p=1}^m \omega_p * F_{A(j)} \quad (35)$$

It has been noted that the value assigned truly describes the significance of the content. It is necessary to establish a minimum threshold in order to decide if the present content is valuable enough to reserve in the cache as a selection. The values of the  $\chi_{\min}(j)$  and  $\chi_{\max}(j)$  attribute thresholds of the different application types must be used to precisely calculate this threshold as indicated in Eq. (36) since they are essential to the efficiency of the caching decision.

**Algorithm 3:** algorithm for content selection and edge caching using MFO-EC decision-making

**Input:**  $m$ ,  $\text{Req}_i$ ,  $\mu_j$ ,  $A(j)$ ,  $n$

1.  $\text{Req}_i$ : the request from costumer for patient IoT devices
2.  $E_k$ : SDN Edge nodes to handle IoT user requests.
3. **Cloud**: Cloud server
4. Initialize all moths (contents) content  $M_i$  randomly
5. Refresh the upcoming data item
6. Set each content's value,  $A(j)$ , using Equations (22) (23) (26) (28) (30)
7. Calculate the attribute weights, using Equations (31), (32)
8. **Output** Best content for caching
9. iteration=0;
10. **for each** (Moth  $i$  in Moth List) **do**
11.      $F(i) = M(i)$
12. **end for**
13. **while** ( $t < \text{Max-Iteration}$ ) **do** #  $T(M) = \text{false}$
14.     iteration=iteration+1
15.     Update number of flames # Based on the same equation (1)
16.     **for each** (Moth  $i$  at time iteration  $t$ ) **do**
17.          $\text{OM} = \text{Fitness}(M)$  # Based on Equation (35)
18.     **end for**
19.     **if** iteration == 1 **then** # Sorting the primary moth population and updating the flames.
20.          $F = \text{Sort}(M)$  in descending order
21.          $\text{OF} = \text{Sort}(\text{OM})$  in descending order
22.     **else** #Sort the moths in current and previous iteration. Then update flames.
23.          $F = \text{Sort}(M_{t-1}, M_t)$  in descending order
24.          $\text{OF} = \text{Sort}(\text{OM}_{t-1}, \text{OM}_t)$  in descending order
25.     **end if**
26.     **for**  $i = 1$  to  $n$  **do**
27.         **for**  $j = 1$  to  $d$  **do**
28.              $M_i = S(M_i, F_j)$  # Update mechanism
29.              $S(M_i, F_j) = D_i \cdot e^{bt} \cdot \cos(2\pi t) + F_j$  # Moths move in a spiral pattern.
30.         **end for**
31.     **end for**
32.     **end while**
33.     **return** Best-contents-to cache #to cache in SDN edge nodes
34. **End procedure**
35. **if** (SDN controller suggests content for caching) **do**
36.     **if** (there is enough storage on edge node) **do**
37.         Cache the content on SDN edge Node on  $E_k$  (nearest SDN edge node to the patient cluster)
38.     **while** (there is not enough storage) **do** #not enough space on related SDN edge node
39.         perform LRU replacement policy
40.     **end while**
41.     Cache the content on SDN edge Node;
42.     **end if**
43. **end if**
44. **for each**  $\text{Req}_i$  for  $A(j)$  on  $E_k$ , Based on SDN controller decision
45.     **if**  $A(j)$  is expired:
46.         remove( $A(j)$ )
47.     **end if**
48. **end for**
49. **end if**
50. Forward  $\text{Req}_i$  to cloud ();
51. **End**

**Fig. 9** The pseudo-code of the algorithms for content selection and request for caching

$$\mu_j = \frac{1}{m} \sum_{i=1}^m \frac{\chi_{\min}(j)}{\chi_{\max}(j)} \quad (36)$$

In conclusion as shown in Fig. 9. Algorithm 3 is used to summarize the process of proposed making caching decision.

**Table 4** Simulation parameters

| Attributes                                | Value  |
|---|--------|
| Simulation time (s)                       | 3000   |
| Patients                                  | 100    |
| IoT devices per patient                   | 6      |
| Edge nodes                                | 10     |
| <i>Packet size (B)</i>                    |        |
| Emergency requests                        | 50     |
| Periodic requests                         | 150    |
| on-demand requests                        | 300    |
| Number of clusters                        | 10     |
| Serving packet requests during simulation | 1000   |
| Patient's deployment strategy             | Random |

## 7 Performance evaluation

Rapid decision-making and agile reactions are necessary as a permanent, round-the-clock remote health monitoring system in the case of multiple acute illnesses and emergencies. When data should be processed and transmitted as rapidly as feasible, raw data is sent from sensor nodes to the cloud in cloud computing. Response latencies may be unclear if the network state is unexpected. On contrast, the system is more beneficial when high priority data analyses are implemented in remote edge nodes and when crucial and urgent decisions are made locally. The processed data may then be uploaded to the cloud for archival and additional examination. In a vast sensor network, local processing at the edge layer can also lessen traffic between gateways and the cloud. Additionally, centralized administration offers control over data flow, which may assist guarantee that applications function as intended. Applications that are not deemed mission-essential can be prioritized for quick delivery over more vital data information needed for medical diagnosis. For instance, the correct services are supplied first thanks to the capacity to shape and manage data traffic [63]. Table 4 shows the list of simulation parameters. This section discusses the simulation setup and outcomes before introducing the assessment measures.

The simulation was run in a java-based simulator, depending on IoTSim [64]. The proposed scheme as depicted in Fig. 1 includes one centralized cloud server, and 10 edge nodes and 100 patients with 6 IoT devices each are used to simulate the model. The patients are distributed randomly in a two-dimensional space of 1050, 1050. The time interval for IoT devices to generate data is set to 20–45 s depending on their type. The storage size was defined as the same for all for SDN edge node 200 GB and the average of link bandwidth consider as 100 Mbps. The suggested method begins by employing spectral clustering to group the patients. The effectiveness and typical retrieval latency of the suggested strategy are compared to other ways. The spectral clustering approach, covered in Sect. 6.1, is used to group the patients. To execute spectral clustering. The patients are

regarded as the graph's vertices. The threshold value from Eq. (17) is used to calculate the degree of integrity between the two patients.

The patients are divided into 10 clusters. The age of patients is categorized in decade and each cluster is allocated to a certain SDN edge node following the suggested methodology. The requested file is checked for in the cloud if the corresponding SDN edge nodes are unable to satisfy the request for the desired data. Considering that each application type's packet data size emergency, periodic, and on-demand requests are set to 50,150, and 300 bytes, respectively. The simulations end after serving around 1000 packet requests. The packet requests are randomly generated.

The suggested scheme is compared with the content-based caching strategy [54]. This scheme is worth considering since it was recently recommended to keep the most useful content in the cache, boost cache usage efficiency, consider delaying an effective caching technique and compare it to the baseline schemes that came before it [55].

a) Proposed scheme—with SDN-Edge caching and with clustering:

The proposed scheme with both SDN-Edge caching and clustering has the lowest average time between data retrieval delays. This is because clustering reduces the number of hops between nodes, and SDN-Edge caching reduces the time it takes to retrieve data by storing frequently requested content at the edge nodes.

b) Without SDN-Edge caching and with clustering:

Without SDN-Edge caching, the average time between data retrieval delays is higher than with the proposed scheme. However, clustering still helps to reduce the number of hops between nodes, which can improve performance.

c) With SDN-Edge caching and without clustering:

With SDN-Edge caching but without clustering, the average time between data retrieval delays is lower than without SDN-Edge caching but higher than the proposed scheme with both SDN-Edge caching and clustering. This is because SDN-Edge caching alone still helps to reduce the time it takes to retrieve data, but clustering further reduces the number of hops between nodes.

d) Content-based caching strategy:

The content-based caching strategy performs worse than the proposed scheme in terms of average time between data retrieval delays. This is because it does not take into account node energy levels or cache size and does not use clustering to reduce the number of hops between nodes. However, it may still be useful in certain scenarios where content is requested infrequently and caching based on content type is more effective.

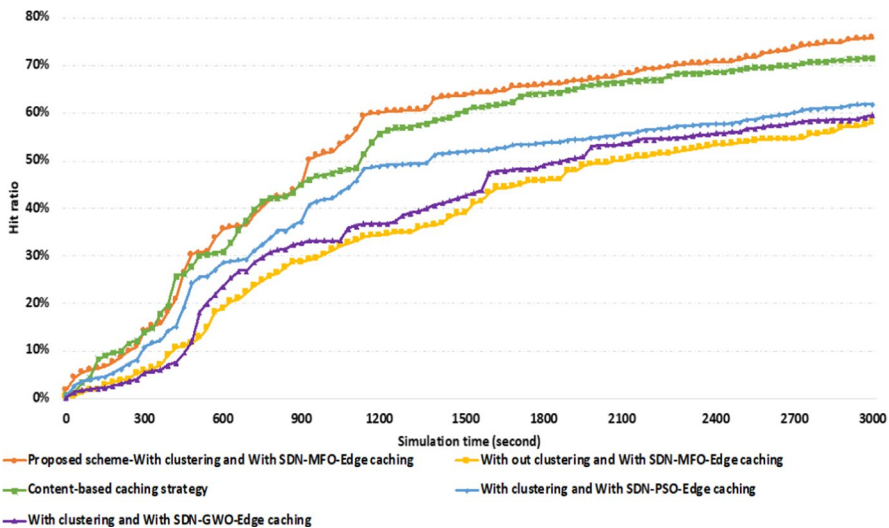


Fig. 10 The comparison of cache hit ratio

## 7.1 Cache hit rate

The probability of obtaining a cache hit from content stored on SDN edge nodes rather than the original cloud servers is known as cache hit ratio, and it is calculated and showed the percentage of cache hits to all request messages. For all of the techniques, the hit ratio could be demonstrated to increase with time. The hit ratio is the most fundamental indicator for measuring the caching decision policy's efficacy. As a result, the hit ratio may be expressed as shown in Eq. (37):

$$\text{Hit ratio} = \frac{\sum_{t=1}^T \text{total number of requests satisfied from the cache on the edge nodes } E_k}{|\text{Request}(t)|} \quad (37)$$

$|\text{Request}(t)|$  is the whole number of requests received at a period of time  $t$ . Simulation time is  $T$ . As a result, the goal is to develop a useful caching decision strategy that outperforms existing rules in terms of caching efficiency.

Figure 10 shows an increase in hit ratio over time for all tactics, with a similar LRU replacement policy. The proposed MFO-SDN-Edge caching scheme demonstrates a significant improvement in the cache hit rate, achieving 76%, which outperforms the content-based caching and "With clustering and With SDN-GWO-Edge caching" and "With clustering and With SDN-PSO-Edge caching" methods by 70.6%, 61.9%, and 59.5%, respectively. The scheme also outperforms the average hit ratio for "Without clustering and With SDN-MFO-Edge caching" and "With clustering and With SDN-MFO-Edge caching" by 37% and 57%, respectively, highlighting the effectiveness of using clustering for improving this parameter. During simulation, the proposed scheme achieved an average hit ratio of 57% for "Without clustering and With SDN-MFO-Edge caching," 36% for "caching content-based

technique," 43.7% for "With clustering and With SDN-PSO-Edge caching," and 38.9% for "With clustering and With SDN-GWO-Edge caching," as discussed in Sect. 6.2. This outcome demonstrates the superior performance of MFO over other bionic intelligent algorithms such as GWO and PSO in the proposed scheme. These findings underscore the efficacy of the clustering method and the role of SDN for orchestrating and network management, as well as MFO-caching for data selection and caching.

## 7.2 Average time between data retrieval delays

The data retrieval latency is a crucial parameter that measures how long the customer is pleased with the answer to the data requests and reflects the efficiency of caching. The cached content is often discarded in e-health IoT due to short data item lifetimes and long delay times. The data retrieval delay must therefore be investigated, which is the interval between sending the request for the data item and receiving it. In SDN edge nodes, effective caching allows the capability of responding to consumer requests where the content is frequently updated. This solution also takes into account the node energy level and cache size so that the nonpotential e-health nodes' batteries do not run out of power as well as decreasing caching delay times.

The average retrieval latency is used to determine how quickly a user can retrieve the content requested. High retrieval delay negatively affects the QoE, which emphasizes the importance of efficient caching mechanisms in e-health IoT [65]. Equation (38) can be used to compute the average retrieval delay.

$$R_{\text{delay}} = \frac{1}{|\text{Request}(t)|} * \frac{\sum_{t=1}^T \text{request travel delay}(\text{data}_K) + \text{response travel delay}(\text{data}_K)}{\lambda^{\text{request}}(\text{data}_K)} \quad (38)$$

$R_{\text{delay}}$  denotes the average retrieval delay.  $\lambda^{\text{request}}(\text{data}_K)$  The popularity of content  $\text{data}_K$  at the current time.

The delay time for retrieving data in e-health IoT is a crucial parameter that measures how long it takes for a customer to receive an answer to their data request and reflects the efficiency of caching. Effective clustering and caching in SDN edge nodes can reduce the average retrieval delay, as demonstrated in Fig. 11. Considering the edge nodes for caching and SDN orchestrating. So, by reducing the average retrieval time delay in the proposed approach the edge nodes would be able to reuse the cached content more easily. During the simulation, the proposed scheme achieved an average time between data retrieval delays of 0.019 s, which is better than the other scenarios. Specifically, the average time between data retrieval delays for the 'caching content-based technique' was 0.023 s, while it was 0.029 s for 'With clustering and With SDN-PSO-Edge caching' and 0.032 s for 'With clustering and With SDN-GWO-Edge caching'. Compared to these scenarios, the proposed scheme with SDN-MFO-Edge caching and clustering showed a decrease of approximately 10%, 27%, and 34%, respectively, in the average time between data retrieval delays. These experimental results demonstrate the effectiveness of the proposed approach in improving the retrieval time of e-health data in an IoT environment.

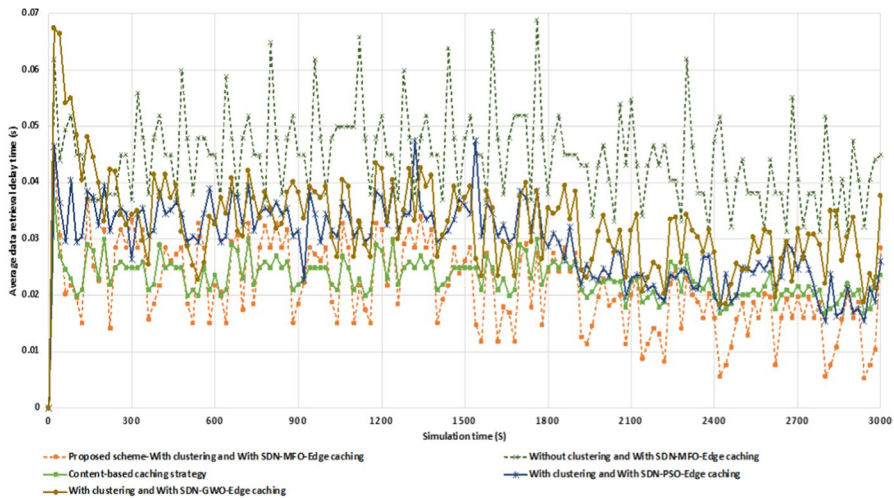


Fig. 11 Comparison of the average data retrieval delay time

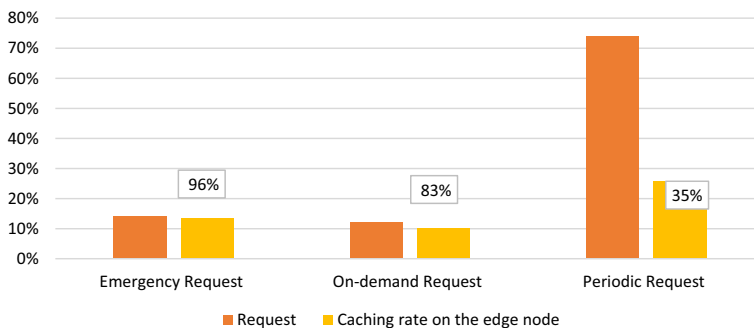


Fig. 12 Request response caching rate on the edge node

### 7.3 Caching ratio based on request the type

The study considers emergency and on-demand requests as randomly generated packet requests, while periodic requests for e-health data follow a Poisson distribution. Figure 12 shows the distribution of requests and the proportion of cached response in the associated SDN edge node according to the proposed method. In the investigated cluster, the importance and prioritization of the requests are considered, taking into account the weight of features for emergency and on-demand requests. The results show that 96% and 86% of the response for emergency and on-demand requests, respectively, are cached in the related SDN edge node. This high proportion of cached responses indicates that the proposed method is effective in addressing emergency and on-demand requests, which are critical in e-health applications.

Periodic requests are generated at regular intervals, and the data requested is often static and not time-sensitive. Therefore, caching may not be as effective in addressing these requests as it is for emergency and on-demand requests. The low proportion of response packets cached for periodic requests in the investigated cluster (35%) suggests. However, it is important to note that the effectiveness of the proposed method in addressing periodic requests may vary depending on the specific characteristics of the e-health IoT network being investigated. Overall, the results suggest that the proposed method is effective in distributing requests and caching response packets based on their importance and features, thus improving the efficiency and effectiveness of caching in e-health IoT applications.

## 8 Limitations in the healthcare system and future works

Despite the fact that IoT devices have slow processing speeds and limited computing power, the idea of SDN and edge computing in healthcare is a good one. It is important for the healthcare system to transmit data in real-time. Any delay could prove fatal. Due to rapid changes in network configurations and settings, IoT devices may encounter compatibility issues if a change occurs in a platform or environment. It will be difficult to design a topological structure model for multi-device, multi-protocol networks and methods for a healthcare network based on IoT to fulfill everyone's needs in a computing environment as it is similar to the system of IoT-based healthcare. By utilizing edge computing, research in the Internet of Healthcare Things is also gaining popularity. The IoT has benefits, but it also has significant drawbacks that need to be addressed. In this article, we describe problems that need to be solved that e-health IoT devices face [1, 6, 8, 42, 43].

- **Implementation of standards:** many vendors manufacture a variety of products for the healthcare industry. The communication protocols, data aggregation, and gateway interfaces of these E-Health devices need to be standardized by a dedicated group. Also the standardization and validation of EMRs recorded by the E-health devices must be taken into account extensively. By collaborating with researchers, information Technology and Innovation Foundation (IETF) and other organizations and standardization bodies ETSI and IPSO can form working groups to standardize devices.
- **Managing data:** IoT environment generates massive amounts of data through interconnected objects, which need to be stored and processed according to the IoT standards. Providing enough storage for all IoT data would be beneficial for some enterprises.
- **Constant observation:** as with chronic diseases, heart diseases, etc., many medical circumstances need for ongoing patient observation. Real-time data monitoring must be possible with IoT devices effectively in such situations.
- **Energy-related problems at the device level:** keeping IoT devices connected is a major challenge. Communication between devices is a major energy-consuming process. Batteries power most E-Health devices. The battery cannot be replaced easily once a sensor is attached. For such a system, a high-power

battery was needed. Globally, researchers are developing medical devices that can self-power. IoT systems could be integrated with renewable energy systems as a potential solution. As a result, these systems may help reduce the global energy crisis to some extent.

- **Effects on the environment:** To develop an e-Health system, biomedical sensors must be integrated with semiconductor-rich devices. Earth metals and other toxic chemicals are typically used in manufacturing and fabrication. The environment may be adversely affected by this. The manufacturing of sensors must therefore be controlled and regulated by a proper regulatory agency. Furthermore, biodegradable materials should be used to make sensors.
- **Keeping data confidential and secure:** a real-time monitoring system has been transformed by the integration of cloud computing. Cyberattacks have also become more prevalent in healthcare networks as a result. Several preventive measures must be taken to protect e-health IoT systems from malicious attacks. Encryption of passwords, fault tolerance, and secure booting are a few of these features.
- **Ability to scale:** a healthcare device's scalability refers to how well it adapts as its environment changes. Scalability refers to the ability of a system to work smoothly without causing any delays and to use the available resources efficiently. The scalability of a device is therefore crucial. In this way, a system can be made more efficient for both present and future purposes by combining electronics, sensors and actuators. An E-Health system shares information through the Internet between healthcare devices, sensors and actuators. It is important to manage an E-Health system efficiently due to the lack of uniformity among the connected devices.
- **Identification:** there are a lot of interactions between healthcare professionals and their patients and caregivers at any given time. Patients who deal with multiple health issues interact with multiple doctors. Therefore, identifying patients, caregivers, and doctors during a single treatment process is crucial for avoiding confusion and maintaining smooth healthcare operations.
- **Ability to self-configure:** by adding features like manual configuration. IoT devices will give their users more power. By doing so, users will be able to modify the system settings in response to changes in the environment and the application requirements.
- **Discovering new diseases:** a growing number of healthcare apps are being added with the rapid development of mobile technology, but they are all designed for a limited number of diseases. In light of this, more diseases need to be considered that were either neglected or not given adequate attention in the past. E-Health applications will be more diverse as a result.

This study presented a methodology for clustering patients and implementing caching services to reduce data retrieval delays to overcome the limitations of previous research. Yet remote healthcare system limitations need to be addressed, problems must be solved, and enhancements need to be made.

## 9 Conclusion

In conclusion, the use of SDN technology and edge computing in e-health IoT networks has a significant potential to optimize the network infrastructure and enhance the management of network resources. The proposed method in this paper utilizes SDN-Edge caching and a clustering approach to prioritize and allocate patient medical records, complementing the remote cloud's storage capacity and reducing data retrieval latency. The MFO-edge caching approach as a multi-criteria decision-making technique has shown better performance compared to other caching methods under comparison. The ability to monitor patients' daily lives closer to the data consumers benefits medical professionals, enabling real-time monitoring, early detection of health issues, timely interventions, and personalized care. The centralized management and dynamic caching mechanism of IoT healthcare systems enhance overall performance and reliability, allowing healthcare providers to deliver better services to patients.

**Author contributions** We (all authors) contributed to the study's conception and design. Material preparation, data collection, and analysis were performed by [Seyedeh Shabnam Jazaeri], and [Parvaneh Asghari]. The first draft of the manuscript was written by [Seyedeh Shabnam Jazaeri]. [Parvaneh Asghari], [Sam Jabbehdari], and [Hamid Haj Seyyed Javadi] reviewed and commented on the manuscript. We (all authors) read and approved the final manuscript.

## Declarations

**Conflict of interest** The authors declare no competing interests.

## References

1. Albahri OS, Zaidan AA, Zaidan BB, Hashim M, Albahri AS, Alsalem MA (2018) Real-time remote health-monitoring systems in a medical centre: a review of the provision of healthcare services-based body sensor information, open challenges and methodological aspects. *J Med Syst* 42:9. <https://doi.org/10.1007/s10916-018-1006-6>, 2018
2. Ruggeri G, Amadeo M, Campolo C, Molinaro A, Iera A (2021) Caching popular transient IoT contents in an SDN-based edge infrastructure. *IEEE Trans Netw Serv Manag.* <https://doi.org/10.1109/TNSM.2021.3056891>
3. Limaye A, Adegbija T (2018) HERMIT: a benchmark suite for the internet of medical things. *IEEE Internet Things J* 5(5):4212–4222. <https://doi.org/10.1109/jiot.2018.2849859>
4. Babbar H, Rani S, Alqahtani SA (2022) Intelligent edge load migration in SDN-IIoT for smart healthcare. *IEEE Trans Industr Inf* 18(11):8058–8064. <https://doi.org/10.1109/tii.2022.3172489>
5. Jazaeri S, Berangi R (2016) Survey present and future visions of Internet of Things (IoT). *Sci J Res* 8(7):8–14. <https://doi.org/10.7537/marsrj080716.02>
6. (2020) In: Health-related emergency disaster risk management (Health-EDRM). <https://doi.org/10.3390/books978-3-03936-315-5>
7. Haghi Kashani M, Madanipour M, Nikravan M, Asghari P, Mahdipour E (2021) A systematic review of IoT in healthcare: applications, techniques, and trends. *J Netw Comput Appl* 192:103164. <https://doi.org/10.1016/j.jnca.2021.103164>

8. Mohammed KI et al (2019) Real-time remote-health monitoring systems: a review on patients prioritisation for multiple-chronic diseases, taxonomy analysis, concerns and solution procedure. *J Med Syst.* <https://doi.org/10.1007/s10916-019-1362-x>
9. Gia TN, Jiang M, Rahmani AM, Westerlund T, Mankodiya K, Liljeberg P, Tenhunen H (2015) Edge computing in body sensor networks: an energy efficient approach. In: *Proc. IEEE Int. Body Sensor Netw. Conf. (BSN)*
10. Ndikumana A, Ullah S, LeAnh T, Tran NH, Hong CS (2017) Collaborative cache allocation and computation offloading in mobile edge computing. In: *2017 19th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, <https://doi.org/10.1109/apnoms.2017.8094149>
11. Klonoff DC (2017) Fog computing and edge computing architectures for processing data from diabetes devices connected to the medical internet of things. *J Diabetes Sci Technol* 11(4):647–652
12. Dubey H, Monteiro A, Constant N, Abtahi M, Borthakur D, Mahler L, Sun Y, Yang Q, Akbar U, Mankodiya K (2017) Edge computing in medical internet-of things: architecture, implementation, and applications. *Handbook of large-scale distributed computing in smart healthcare*. Springer, pp 281–321
13. Chen Z, Shikh-Bahaei T, Luff P, Shikh-Bahaei M (2018) Edge computing for low delay to cache and transfer electronic medical records. *Intell Syst Appl* 2:474–486. [https://doi.org/10.1007/978-3-030-01057-7\\_38](https://doi.org/10.1007/978-3-030-01057-7_38)
14. Guibert D, Wu J, He S, Wang M, Li J (2017) CC-Fog: toward content-centric fog networks for. In: *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, <https://doi.org/10.1109/healthcom.2017.8210830>
15. Althamary I, Huang CW, Lin P, Yang SR, Cheng CW (2018) Popularity-based cache placement for fog networks. In: *14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, IEEE
16. Ali S, Ghazal M (2017) Real-time heart attack mobile detection service (RHAMDS) an IoT use case for software defined networks. In: *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, <https://doi.org/10.1109/ccece.2017.7946780>
17. Khodaparas S, Yousefi S, Benslimane A (2019) A multi criteria cooperative caching scheme for internet of things. In: *IEEE International Conference on Communications (ICC)*, <https://doi.org/10.1109/ICC.2019.8761546>
18. Wu H, Luo Y, Li C (2020) Optimization of heat-based cache replacement in edge computing system. *J Supercomput* 77(3):2268–2301. <https://doi.org/10.1007/s11227-020-03356-1>
19. Khodaparas S, Benslimane A, Yousefi S (2020) A software-defined caching scheme for the Internet of Things. *Comput Commun* 158:178–188. <https://doi.org/10.1016/j.comcom.2020.05.002>
20. Liang C (2020) Enhancing QoE-aware wireless edge caching with software-defined wireless networks. In: *Encyclopedia of Wireless Networks*, pp 427–431. [https://doi.org/10.1007/978-3-319-78262-1\\_127](https://doi.org/10.1007/978-3-319-78262-1_127)
21. Fekih A, Gaied S, Youssef H (2020) SDN-based replication management framework for CCN networks. In: *Workshops of the International Conference on Advanced Information Networking and Applications*, Springer, pp 83–99
22. Fekih A, Gaied S, Youssef H (2020) Quality of experience aware replication framework for video streaming in content-centric mobile networks based on SDN architecture. *DiCES-N 2020 Commun Comput Inf Sci* 1348:75–94. [https://doi.org/10.1007/978-3-030-65810-6\\_5](https://doi.org/10.1007/978-3-030-65810-6_5)
23. Van Doan D, Ai Q (2018) An efficient in-network caching decision algorithm for Internet of things. *Int J Commun Syst* 31(8):1–14. <https://doi.org/10.1002/dac.3521>
24. Naeem M, Ali R, Kim B-S, Nor S, Hassan S (2018) A periodic caching strategy solution for the smart city in information-centric internet of things. *Sustainability* 10(7):2576. <https://doi.org/10.3390/su10072576>
25. Kabir A, Iqbal MS, Rathore SA, Kitindi EJ, Rehman G (2018) User aware edge caching in 5G wireless networks. *IJCSNS Int J Comput Netw Inf Secur* 18(1):25–32
26. Naeem MA, Zikria YB, Ali R, Tariq U, Meng Y, Bashir AK (2022) Cache in fog computing design, concepts, contributions, and security issues in machine learning prospective. *Digit Commun Netw.* <https://doi.org/10.1016/j.dcan.2022.08.004>
27. Li C, Qianqian C, Luo Y (2022) Low-latency edge cooperation caching based on base station cooperation in SDN based MEC. *Expert Syst Appl* 191:116252. <https://doi.org/10.1016/j.eswa.2021.116252>
28. Zyrianoff I, Trotta A, Sciullo L, Montori F, Di Felice M (2022) IoT edge caching: taxonomy, use cases and perspectives. *IEEE Internet of Things Mag* 5(3):12–18

29. Huang X, Chen Z, Chen Q, Zhang J (2022) Federated learning based QoS-aware caching decisions in fog-enabled internet of things networks. *Digit Commun Netw*. <https://doi.org/10.1016/j.dcan.2022.04.022>
30. Peng J et al (2022) MagNet: cooperative edge caching by automatic content congregating. In: *The ACM Web Conference 2022*. <https://doi.org/10.1145/3485447.3512146>
31. Samir R, El-Hennawy H, Elbadawy H (2023) Cluster-based multi-user multi-server caching mechanism in beyond 5G/6G MEC. *Sensors* 23(2):996. <https://doi.org/10.3390/s23020996>
32. Zahedinia MS, Khayyambashi MR, Bohlooli A (2022) Fog-based caching mechanism for IoT data in information centric network using prioritization. *Comput Netw* 213(4):109082. <https://doi.org/10.1016/j.comnet.2022.109082>
33. Selvi KT, Thamilselvan R (2021) Dynamic resource allocation for SDN and edge computing based 5G network. In: *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pp 19–22. <https://doi.org/10.1109/icicv50876.2021.9388468>
34. Navaz AN, El-Kassabi TH, Serhani MA, Oulhaj A, Khalil K (2022) A novel patient similarity network (PSN) framework based on multi-model deep learning for precision medicine. *J Personal Med* 12(5):768. <https://doi.org/10.3390/jpm12050768>
35. Dupont G, Leite C, dos Santos DR, Costante E, den Hartog J, Etalle S (2021) Similarity-based clustering for IoT device classification. In: *2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS)*. <https://doi.org/10.1109/coins51742.2021.9524201>
36. Shafique A, Asad M, Aslam M, Shaikat S, Cao G (2022) Multi-hop similarity-based-clustering framework for IoT-oriented software-defined wireless sensor networks. *IET Wirel Sensor Syst* 12(2):67–80. <https://doi.org/10.1049/wss2.12037>
37. Murata A, Doi T, Karwowski W (2022) Effectiveness of tactile warning and voice command for enhancing safety of drivers. *IEEE Access* 10:93854–93866. <https://doi.org/10.1109/access.2022.3204045>
38. Zeb A, Wakeel S, Rahman T, Khan I, Uddin MI, Niazi B (2022) Energy-efficient cluster formation in IoT-enabled wireless body area network. *Comput Intell Neurosci*. <https://doi.org/10.1155/2022/255859>
39. Zhang Z, Lung C-H, St-Hilaire M, Lambadaris I (2020) An SDN-based caching decision policy for video caching in information-centric networking. *IEEE Trans Multimed* 22(4):1069–1083. <https://doi.org/10.1109/tmm.2019.2935683>
40. Lv Z, Qiao L (2020) Optimization of collaborative resource allocation for mobile edge computing. *Comput Commun* 161:19–27. <https://doi.org/10.1016/j.comcom.2020.07.022>
41. Haleem A, Javaid M, Singh RP, Suman R (2021) Telemedicine for healthcare: capabilities, features, barriers, and applications. *Sens Int* 2:100117. <https://doi.org/10.1016/j.sintl.2021.100117>
42. Nikoui TS, Rahmani AM, Tabarsaied H (2019) Data management in fog computing. *Fog Edge Comput Princ Paradig*. <https://doi.org/10.1002/9781119525080.ch8>
43. Jazaeri SS, Jabbehdari S, Asghari P, Seyyed Jazvadi HH (2021) Edge computing in SDN-IoT networks: a systematic review of issues, challenges and solutions. *Cluster Comput*. <https://doi.org/10.1007/s10586-021-03311-6>
44. Jazaeri S, Berangi R (2016) Study the challenges of using and development of 5G networks. *Sci J Rep Opin* 8(7):13–19. <https://doi.org/10.7537/marsroj080716.03>
45. <http://www.hl7.org/implement/standards/rim.cfm>. Accessed 23 Feb 2022. [Online]
46. Misra S, Saha R, Ahmed N (2020) Health-flow: criticality-aware flow control for SDN-based healthcare IoT. In: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*
47. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. *Knowl-Based Syst* 89:228–249. <https://doi.org/10.1016/j.knosys.2015.07.006>
48. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf OPTIMIZER. *Adv Eng Softw* 69:46–61. <https://doi.org/10.1016/j.advengsoft.2013.12.007>
49. Keshari SK, Kansal V, Kumar S (2021) A cluster based intelligent method to manage load of controllers in SDN-IoT networks for smart cities. *Scalable Comput Pract Exp* 22(2):247–257. <https://doi.org/10.12694/scpe.v22i2.1912>
50. Azimi S, Pahl C, Shirvani M (2020) Particle swarm optimization for performance management in multi-cluster IoT edge architectures. In: *Proceedings of the 10th International Conference on Cloud Computing and Services Science*. <https://doi.org/10.5220/0009391203280337>
51. Singh T, Saxena N, Khurana M, Singh D, Abdalla M, Alshazly H (2021) Data clustering using moth-flame optimization algorithm. *Sensors* 21(12):4086. <https://doi.org/10.3390/s21124086>

52. Wan J, Chen B, Wang S, Xia M, Li D, Liu C (2018) Fog computing for energy-aware load balancing and scheduling in smart factory. *IEEE Trans Industr Inf* 14(10):4548–4556. <https://doi.org/10.1109/tii.2018.2818932>
53. Alazab M, Lakshman K, Reddy T, Pham Q-V, Reddy Maddikunta PK (2021) Multi-objective cluster head selection using fitness averaged rider optimization algorithm for IoT networks in smart cities. *Sustain Energy Technol Assess* 43:100973. <https://doi.org/10.1016/j.seta.2020.100973>
54. Doan DV, Nguyen SQ, Van TP (2021) Content-based caching strategy for ubiquitous healthcare application in WBANs. In: *Advances in Intelligent Systems and Computing*, pp 491–500
55. Gupta RN, Dhurandher SK (2020) Efficient caching method in fog computing for internet of everything. *Peer-to-Peer Netw Appl* 14(1):439–452. <https://doi.org/10.1007/s12083-020-00952-z>
56. Tashkandi A, Wiese I, Wiese L (2018) Efficient in-database patient similarity analysis for personalized medical decision support systems. *Big Data Res* 13:52–64. <https://doi.org/10.1016/j.bdr.2018.05.001>
57. da Costa CA, Pasluosta CF, Eskofier B, da Silva DB, da Rosa Righi R (2018) Internet of health things: toward intelligent vital signs monitoring in hospital wards. *Artif Intell Med* 89:61–69. <https://doi.org/10.1016/j.artmed.2018.05.005>
58. Wang N et al (2019) Measurement and application of patient similarity in personalized predictive modeling based on electronic medical records. *Biomed Eng*. <https://doi.org/10.1186/s12938-019-0718-2>
59. Al-Turjman F, Imran M, Vasilakos A (2017) Value-based caching in information-centric wireless body area networks. *Sensors*. <https://doi.org/10.3390/s17010181>
60. OpenFlow Switch Specification version 1.5.1 Open Networking Foundation (ONF). 26 March 2015. [Online]. Available: <https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf>
61. Horn SD, Horn RA, Sharkey PD (1986) The severity of illness index as a severity adjustment to diagnosis-related group. *Health Care Finan Rev* 255(11):1435. <https://doi.org/10.1001/jama.1986.03370110057008>
62. Chandavarkar BR, Guddeti RMR (2015) Simplified and improved analytical hierarchy process aid for selecting candidate network in an overlay heterogeneous networks. *Wirel Personal Commun* 83(4):2593–2606. <https://doi.org/10.1007/s11277-015-2557-1>
63. Shahid MH, Hameed AR, Ul Islam S, Khattak HA, Din IU, Rodrigues JJPC (2020) Energy and delay efficient fog computing. *Comput Commun* 154:534–541
64. Alwasel K, Jha DN, Habeeb F, Demirbag U, Rana O, Baker T, Dustdar S, Villari M, James P, Solaiman E, Ranjan R (2021) IoTSim-Osmosis: a framework for modeling and simulating IoT applications over an edge-cloud continuum. *J Syst Architect* 116:101956
65. Asmat H, Ullah F, Zareei M, Khan A, Mohamed EM (2020) Energy-efficient centrally controlled caching contents for information-centric internet of things. *IEEE Access* 8:126358–126369. <https://doi.org/10.1109/access.2020.3008193>

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.