Article

Not peer-reviewed version

ML-RASPF: A Machine Learning-Based Rate-Adaptive Framework for Dynamic Resource Allocation in Smart Healthcare IoT

Wajid Rafique [*]

Posted Date: 28 April 2025

doi: 10.20944/preprints202504.2273.v1

Keywords: ML-RASPF; Smart healthcare; Internet of Things (IoT); edge computing; rate-adaptive provisioning; latency-aware services; machine learning; reinforcement learning; mist-edge-cloud architecture

Article

ML-RASPF: A Machine Learning-Based Rate-Adaptive Framework for Dynamic Resource Allocation in Smart Healthcare IoT

Wajid Rafique

Department of Electrical and Software Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada; wajid.rafique@ucalgary.ca

Abstract: The growing adoption of the Internet of Things (IoT) in healthcare has led to an extensive growth in real-time data from wearable devices, medical sensors, and patient monitoring systems. This data and latency-sensitive environment poses a significant challenge to conventional cloud-centric infrastructures, which struggle with unpredictable service demands, link congestion, and end-to-end delay across distributed environments. Specifically, traditional cloud infrastructures struggle to deliver consistent service quality in smart healthcare, where both low end-to-end latency and adaptive service delivery rates are critical for life-saving applications. We propose ML-RASPF, a machine learning-based service delivery framework for efficient and scalable IoT service delivery in smart healthcare systems to address these issues. ML-RASPF formulates the provisioning task as a joint optimization problem that aims to minimize service latency and maximize delivery rate stability. The framework comprises three key components: (i) a network parameter initialization module, (ii) a supervised learning model for traffic demand prediction, and (iii) a reinforcement learning-based service adaptation engine for real-time resource allocation. These modules intelligently distribute workloads across a hybrid cloud environment. We evaluate ML-RASPF using a realistic smart hospital scenario involving IoT-enabled kiosks and wearable devices delivering both latency-sensitive and latency-tolerant services. Experimental results demonstrate that ML-RASPF significantly outperforms state-of-the-art edge–cloud systems in terms of latency reduction, energy efficiency, bandwidth utilization, and service delivery rate.

Keywords: ML-RASPF; Smart healthcare; Internet of Things (IoT); edge computing; rate-adaptive provisioning; latency-aware services; machine learning; reinforcement learning; mist-edge-cloud architecture

---

1. Introduction

The proliferation of Internet of Things (IoT) devices has accelerated rapidly in recent years, with projections estimating nearly 125 billion devices by 2032 [1]. This exponential growth, coupled with advancements in quantum computing and the increasing convergence of IoT and artificial intelligence (AI), is expected to generate unprecedented volumes of heterogeneous data. One of the most impactful domains leveraging IoT is smart healthcare, where intelligent systems support personalized care, continuous monitoring, and timely interventions. In such environments, connected medical devices and wearables operate autonomously to facilitate real-time data exchange and decision making. Unlike traditional session-based models, IoT communication is inherently content-centric, where devices request and consume data or services directly from the network without persistent connectivity to specific service hosts [2]. Latency-sensitive and compute-intensive healthcare use cases, such as remote diagnostics, real-time monitoring, and emergency response, can benefit significantly from IoT's sensing capabilities and intelligent service provisioning [3]. However, the explosive growth in IoT-enabled healthcare services introduces major challenges in ensuring timely, scalable, and efficient resource provisioning. This calls for advanced, machine learning (ML) driven frameworks capable of adaptive and optimized service delivery tailored to the unique demands of smart healthcare systems.

The emerging vision of the Internet of services emphasizes the need to collect, analyze, and respond to data in a personalized and context-aware manner [4]. This paradigm, while powerful, places a heavy burden on resource-constrained IoT devices such as medical wearables and monitoring sensors, that often lack the necessary memory, power, and computational capacity to process large datasets independently [3]. Cloud computing has traditionally offered scalable processing power by offloading computations to remote data centers [5]. However, this approach introduces considerable latency, making it unsuitable for critical smart healthcare applications such as augmented reality-assisted surgery, real-time patient monitoring, and emergency alerts, where response times must be measured in milliseconds [3]. Even slight delays can significantly degrade the performance and reliability of such services. Moreover, cloud-centric models raise serious concerns regarding the privacy and security of sensitive medical data. In many cases, smart healthcare IoT systems must balance security with service quality, revealing critical trade-offs in cloud-dependent models. These challenges necessitate more responsive and privacy-conscious architectures that tightly integrate ML with distributed intelligence for latency-aware service delivery.

Edge computing extends the capabilities of IoT by enabling data processing closer to the source, thereby significantly improving service responsiveness and reducing network congestion [5]. In smart healthcare environments, edge nodes support real-time analytics, local stream mining, and data filtering, which are essential for minimizing latency and conserving bandwidth. Applications such as augmented reality (AR) for assisted surgery, remote diagnostics, and real-time video consultations demand ultra-low latency and high reliability requirements that traditional cloud infrastructure cannot meet. IoT services in healthcare are inherently heterogeneous, exhibiting diverse latency sensitivities and variable delivery rate requirements based on patient conditions and service contexts. Addressing this variability requires rate-adaptive mechanisms that can dynamically respond to fluctuating demands while optimizing resource usage. Mist computing, a complementary layer to edge and cloud, introduces lightweight, near-device processing that further reduces latency and offloads tasks from edge servers [6]. By incorporating mist computing into the architecture, the system can enable more fine-grained and context-aware service provisioning beneficial in emergency or mobile healthcare scenarios. The integration of ML across these layers allows the system to adaptively learn traffic patterns, predict demands, and make optimized allocation decisions in real time.

Given the distributed and dynamic nature of smart healthcare IoT environments, intelligently monitoring and distributing incoming service requests across edge and cloud layers is critical to minimizing latency and ensuring timely responses [7]. We illustrate the problem setting and layered control architecture in Figure 1, which depicts a smart healthcare environment where IoT devices, such as diagnostic kiosks, patient wearables, and hospital displays—generate diverse service requests with varying latency and bandwidth demands. These requests traverse a mist–edge–cloud continuum, where ensuring timely delivery of critical alerts and efficient throughput for routine services presents a major challenge [5]. Distributed edge cloudlets serve as intermediate processing units to handle resource-intensive tasks while reducing the load on centralized cloud infrastructure. To maximize their utility, it is essential to develop intelligent, latency-aware strategies that allocate resources optimally across these layers [6]. A stable and responsive service delivery rate, coupled with low-latency guarantees, plays a crucial role in maintaining consistent QoS for life-critical services in smart healthcare systems [8].

However, most existing service provisioning approaches in edge-cloud ecosystems rely on static or heuristically tuned resource allocation models [9]. While a few adopt rate-adaptive techniques [7,8,10], they typically address only latency-tolerant scenarios, making them insufficient for real-time healthcare use cases. Moreover, these models often neglect dynamic changes in network conditions and impose significant overhead from continuous load monitoring. However, these approaches rarely address real-time healthcare needs, as they lack unified models that jointly optimize latency and rate.

Their static assumptions and monitoring overhead result in inconsistent service quality under dynamic load. This performance gap highlights the need for a ML-based, rate-adaptive provisioning framework that dynamically responds to fluctuating network and service conditions to ensure robust, low-latency healthcare delivery.

To tackle these limitations, we present ML-RASPF which operates across mist, edge, and cloud layers, dynamically adjusting service delivery paths and resource allocation decisions based on real-time latency and delivery rate requirements of heterogeneous healthcare applications. Edge nodes collaborate in a vertical hierarchy to monitor the state of the network and allocate resources adaptively, ensuring that both urgent and non-critical services meet their respective QoS constraints. Unlike traditional methods, our framework jointly optimizes service delivery rate and latency, selecting optimal service paths that reduce overall delay while maintaining high throughput. Our contributions bridge a critical research gap by integrating supervised and reinforcement learning (RL) into a real-time, utility-aware service allocation framework.

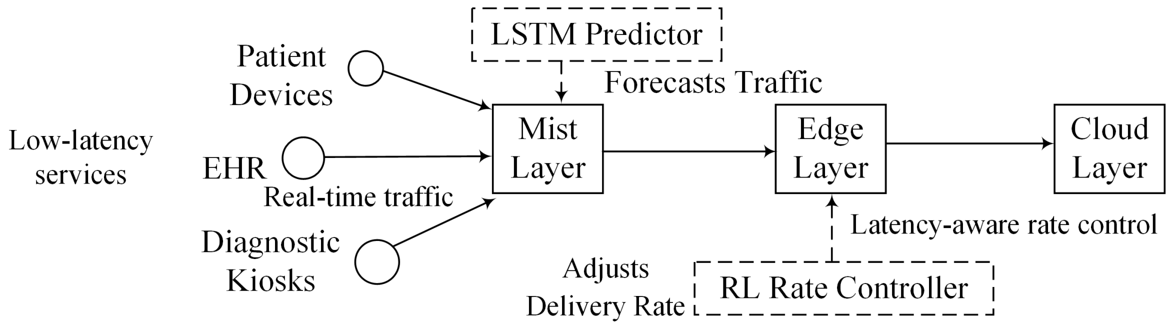

Figure 1. Illustration of latency-aware and rate-adaptive service flow across cloud, edge, and mist layers in a smart healthcare setting. RL and LSTM modules dynamically optimize traffic routing and service delivery.

To address these limitations, we present ML-RASPF, shown in Figure 1 which operates across mist, edge, and cloud layers, dynamically adjusting service delivery paths and resource allocation decisions based on real-time latency and delivery rate requirements of heterogeneous healthcare applications. Edge nodes collaborate in a vertical hierarchy to monitor the state of the network and allocate resources adaptively, ensuring that both urgent and non-critical services meet their respective QoS constraints. Unlike traditional methods, our framework jointly optimizes service delivery rate and latency, selecting optimal service paths that reduce overall delay while maintaining high throughput. Our contributions bridge a critical research gap by integrating supervised and reinforcement learning (RL)into a real-time, utility-aware service allocation framework.

The key contributions of this work are as follows:

- We present **ML-RASPF**, a novel hybrid mist–edge–cloud framework for rate-adaptive and latency-aware IoT service provisioning in smart healthcare systems.
- We formulate the service provisioning problem as a *joint optimization model* that integrates both latency constraints and service delivery rates. This formulation enables intelligent, QoS-aware resource allocation across heterogeneous IoT environments.
- We propose a modular, ML-driven algorithmic suite combining supervised learning for traffic prediction and RL for real-time service rate adaptation.
- We evaluate the proposed framework within *EdgeCloudSim*, using realistic smart healthcare workloads and the results show that ML-RASPF significantly outperforms state-of-the-art rate-adaptive methods by reducing latency, energy consumption, and bandwidth utilization while improving service delivery rate.

The remainder of this paper is organized as follows. Section II reviews the related work on IoT service provisioning, edge-cloud architectures, and ML applications in healthcare. Section III introduces the proposed rate-adaptive framework, detailing its layered architecture and operational workflow. Section IV formulates the service provisioning problem as a joint optimization model and describes the ML-driven algorithm used for resource allocation. Section V presents the experimental setup and performance evaluation results based on a smart healthcare scenario. Finally, Section VI concludes the paper and outlines potential directions for future research.

2. Related Work

A variety of QoS-aware service provisioning techniques have been explored for IoT-based environments, particularly in edge-cloud architectures. Most of these techniques rely on deterministic or rule-based models and lack self-adaptive capabilities under changing network dynamics. Azmi et al. [11] introduce a tensor-based resource mapping technique that assigns service requests to cloud and edge servers based on latency thresholds. This technique is effective in reducing bandwidth overhead and improving cloud utilization; however, it incurs additional computational overhead, leading to increased latency in dynamic environments. Centofanti et al. [12] address resource optimization for crowdsensing applications using a mixed-integer linear programming approach, offering efficient task allocation under deterministic conditions. Li et al. [13] present a multi-objective service provisioning framework based on linear programming, focusing primarily on latency constraints while overlooking service delivery rate and dynamic adaptability. Similarly, Ahmed et al. [14] design fog infrastructure for latency-sensitive IoT scenarios, achieving near-optimal deployment efficiency at a low operational cost. However, this technique largely depend on static heuristics and do not integrate learning-based mechanisms, which makes it unsuitable for real-time healthcare environments that demand continuous adaptability.

Extending computational capabilities from the cloud to the edge has been widely recognized as a key strategy to improve the efficiency and responsiveness of IoT service provisioning [7,8,8,15,16]. Mishra et al. [16] focus on delay-sensitive service provisioning, proposing a strategy to reduce latency for time-critical applications. In the context of smart healthcare, Asif et al. [17] propose a QoS-aware service delivery model utilizing mist, fog, and cloud layers to support diverse healthcare requirements. However, their model is domain-specific, and introduces additional network overhead, and does not guarantee global optimality in service provisioning. Fei et al. [15] offload selected tasks to edge nodes to reduce cloud resource consumption, but their framework does not account for varant latency demands in IoT services, which is critical in healthcare applications. Despite these efforts, existing solutions either lack end-to-end adaptiveness or fail to jointly optimize latency and service delivery rate—limitations that our proposed framework directly addresses through ML-based optimization and real-time rate adaptation.

Edge analytics has emerged as a promising approach to reduce the computational burden on centralized cloud servers while minimizing bandwidth usage and overall infrastructure costs [18–20]. Ji et al. [18] propose a collaborative service provisioning model where multiple edge nodes cooperate to deliver services efficiently. This approach effectively distributes workload in hybride cloud environment; however, it adds processing delays due to use of complex AI algorithms. Najim et al. [19] design a deep learning-based analytics framework for roadside units. However, their method lacks rate-adaptiveness and suffers from accuracy degradation due to the lack of training data. Shang et al. [20] explore adaptive video quality control for smart city surveillance using edge-based analytics; however, this approach fails to jointly optimize latency and service delivery rate.

Recent works ([2,4,7,8]) have begun exploring dynamic service provisioning, but often neglect service-specific utility modeling or fail to incorporate real-time feedback mechanisms for rate tuning. Additionally, energy and bandwidth trade-offs at mist and edge layers remain underexplored, particularly in systems with constrained devices and mission-critical latency needs.

In summary, although significant progress has been made in latency-aware and edge-enabled IoT service provisioning, existing frameworks still fall short in addressing the compound challenge of achieving real-time responsiveness while optimizing delivery rates under dynamic conditions. First, many prior works rely on static or periodic resource scheduling models, which cannot adapt to fluctuating traffic or user demands, especially in healthcare scenarios where service urgency is variable. Second, several edge–cloud optimization strategies introduce computational overhead due to centralized decision-making or heavy analytic modules, resulting in increased processing delay. Third, while RL and ML-based approaches are emerging, few existing solutions offer end-to-end integration of predictive traffic modeling and adaptive rate control in a layered (mist–edge–cloud) architecture.

These gaps motivate the development of ML-RASPF, a ML-based, rate-adaptive framework designed to jointly optimize latency and service delivery rate across mist–edge–cloud layers. ML-RASPF uses LSTM-based traffic forecasting and RL-based rate control in a convex optimization structure, which addresses limitations of prior models and delivers a lightweight, scalable, and QoS-aware provisioning mechanism for real-time healthcare IoT services.

3. Optimal Service Provisioning Framework

The proposed ML-RASPF framework is based on distributed data processing principles to enhance resource utilization and ensure QoS in latency-sensitive smart healthcare environments. As shown in Figure 2, the architecture consists of five coordinated layers that collaboratively manage end-to-end service provisioning across mist, edge, and cloud infrastructures. To illustrate its real-world applicability, we present a smart healthcare use case focused on continuous patient monitoring and emergency response. In this scenario, IoT-enabled medical devices, such as wearable health trackers, bedside monitors, and mobile diagnostic units, collect real-time physiological data from patients across various hospital zones or remote care settings. This data includes metrics such as heart rate, blood pressure, oxygen saturation, and mobility patterns, all of which require timely and reliable processing to enable prompt medical interventions.

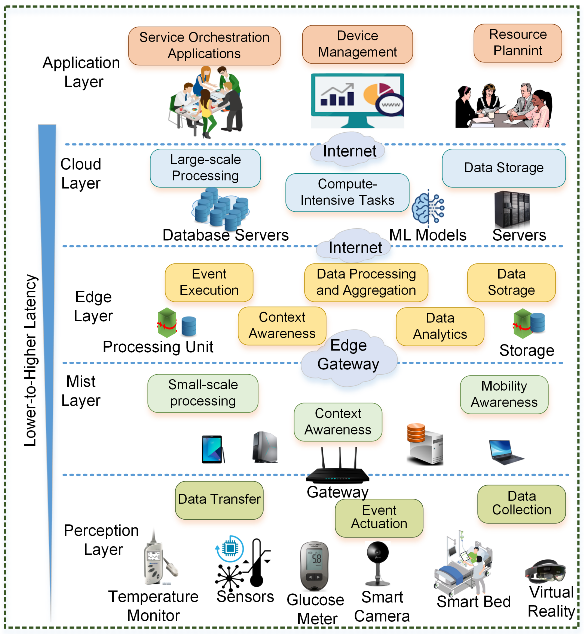

Figure 2. Architecture of the ML-RASPF framework for optimal service provisioning.

A critical requirement in such a healthcare setup is the ability to perform localized, on-demand analytics with minimal latency. For instance, edge cloudlets deployed within hospital premises can process time-sensitive alerts, such as detecting arrhythmias or fall events before forwarding relevant summaries to central hospital servers or cloud-based electronic health records (EHR) systems. Data from non-critical or latency-tolerant services, such as periodic wellness logs or historical diagnostic trends, can be cached and analyzed at the cloud layer when needed. The proposed framework supports energy-efficient sensing, data transmission, and real-time service delivery by adaptively allocating resources across its layers. This layered orchestration ensures not only low-latency responses for critical healthcare events but also maintains consistent service delivery rates for ongoing data streams.

By balancing delivery speed and throughput across mist, edge, and cloud layers, the framework effectively fulfills QoS requirements for both time-sensitive and bandwidth-intensive smart healthcare services.

3.1. Customized Mist–Edge–Cloud Framework

The framework comprises five layers, including *Perception*, *Mist*, *Edge*, *Central Cloud*, and *Cloud Application*—working collaboratively for latency-aware and rate-adaptive service provisioning in smart healthcare systems. A depiction of the framework is shown in Figure 2.

3.1.1. Perception Layer

The lowest layer, consisting of medical sensors enabled by IoT such as heart rate monitors, pulse oximeters, ECG patches, and ambient condition sensors are responsible for data acquisition. These sensors operate on patient bedsides, ICUs or remote home care settings, collecting vital signs and contextual information in real time. The captured data is forwarded to the mist layer for low-latency processing.

3.1.2. Mist Layer

Mist nodes operate at the edge of perception, adding lightweight analytics and decision-making capabilities to reduce data load and minimize latency. For example, if a patient's heart rate exceeds critical thresholds, the mist node can instantly trigger alerts to nearby healthcare staff without waiting for cloud-level processing. The mist layer is also mobility-aware and maintains continuous service flow even when patient devices move across network zones.

3.1.3. Edge Computing Layer

This layer consists of hospital-deployed edge servers and mobile cloudlets with moderate-to-high computational capabilities. It processes latency-sensitive services like anomaly detection in ECG signals or medication reminders. The edge also caches temporary data and forwards non-urgent requests to the central cloud. It supports container-based virtualization and uses heterogeneous connectivity (e.g., Wi-Fi, 5G, 6G) to ensure responsive service provisioning.

3.1.4. Central Cloud Layer

The central cloud manages long-term analytics, large-scale training of predictive models, and population-level trend analysis. It stores historical patient data and supports computationally intensive operations such as ML-based risk stratification, disease progression modeling, and cross-site healthcare analytics. It also ensures system-wide fault tolerance and service continuity.

3.1.5. Cloud Application Layer

The topmost layer presents healthcare dashboards, analytics interfaces, and visualization tools for medical personnel. It supports telemedicine consultations, alerts for emergency services, and the integration of personalized health insights into the hospital workflow. This layer is also the point of orchestration for deploying new healthcare applications across the lower tiers.

3.2. Smart Healthcare with Emergency and Routine Services

As illustrated in Figure 3, the ML-RASPF framework enables differentiated treatment of services. High-priority emergency alerts, such as cardiac events—are handled entirely at the mist or edge layers, ensuring sub-second responsiveness. Meanwhile, routine services like patient check-in logs or diet monitoring are processed at the cloud layer. The framework dynamically adapts delivery paths, execution points, and resource allocation based on latency requirements and real-time network states, making it ideal for robust and scalable Smart healthcare IoT deployments.

4. Analytical Framework for Heterogeneous Service Provisioning

To address the challenges of latency-sensitive and dynamically evolving IoT services in smart healthcare, we develop an analytical framework for adaptive service provisioning across mist, edge, and cloud layers. The objective is to jointly optimize delivery rate and latency by allocating bandwidth and computing resources based on service priority, link conditions, and predicted demand. Unlike conventional static models, our framework supports heterogeneous healthcare workloads, ranging from real-time telemetry to elastic background services, by integrating ML modules with a mathematically grounded optimization engine. This model enables the system to dynamically route, throttle, and prioritize services with respect to utility and delay constraints, ensuring quality of service (QoS) across all tiers of the architecture.

Figure 3 illustrates the operational flow of the proposed ML-RASPF framework. IoT data sources such as sensors and wearables first transmit service requests to local mist nodes. These requests feed into two parallel ML modules: an LSTM-based traffic predictor that forecasts congestion and link load, and a RL agent that continuously adapts service delivery rates based on latency feedback. The outputs of both modules are forwarded to an optimization engine, which solves for delivery rate and path allocation using utility-delay tradeoffs. The optimized decisions are passed to a rate adaptation module that enforces service-level adjustments per consumer. Finally, a centralized resource allocator orchestrates workload distribution across mist, edge, and cloud layers. This pipeline ensures that service provisioning is predictive, adaptive, and QoS-aware, with intelligent routing grounded in real-time analytics.

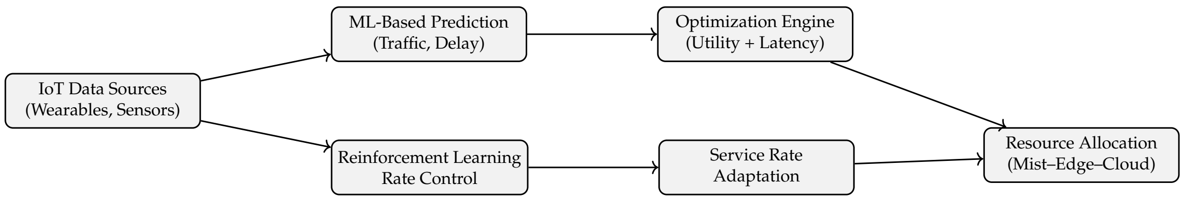
IoT Data Sources (Wearables, Sensors)
ML-Based Prediction (Traffic, Delay)
Reinforcement Learning Rate Control
Optimization Engine (Utility + Latency)
Service Rate Adaptation
Resource Allocation (Mist–Edge–Cloud)

Figure 3. ML-RASPF workflow from IoT input to rate-adaptive service provisioning across mist–edge–cloud.

4.1. Problem Overview and Healthcare Service Requirements

In a typical smart healthcare system, service consumers (e.g., patients, caregivers, clinicians) interact with distributed service providers through a network comprising mist nodes, edge devices, and cloud infrastructure. The goal is to ensure that healthcare services are delivered at appropriate rates while meeting strict QoS constraints, especially for latency-critical applications such as emergency alerts or continuous glucose monitoring.

Let $S = \{s_1, s_2, \ldots, s_n\}$ denote the set of healthcare services, and let $C_s = \{c_{s,1}, c_{s,2}, \ldots, c_{s,n}\}$ denote the set of consumers for each service $s \in S$. Each consumer is connected to the service provider via a set of links $T = \{t_1, t_2, \ldots, t_m\}$ with associated capacities $C = \{c^1, c^2, \ldots, c^m\}$. The objective is to allocate network resources such that each consumer receives service s at an optimal delivery rate $g_{s,i}$, measured in Mbps, while satisfying both rate and latency constraints.

Services are broadly categorized into two types: (i) latency sensitive services such as real-time ECG monitoring or fall detection, which require immediate data processing and response, and (ii) Latency-tolerant services, such as access to EHRs or non-urgent data synchronization, which can tolerate higher latency and lower bandwidth. This heterogeneous service demand motivates the development of a rate-adaptive and latency-aware provisioning model that dynamically adjusts service delivery rates based on network conditions, service criticality, and user context. In this work, we aim to jointly optimize the service delivery rate and latency using a utility-based convex optimization framework, further enhanced by ML techniques for traffic prediction and adaptive control.

4.1.1. Utility-Based Formulation for Service Delivery Rate

To support heterogeneous healthcare services, we adopt a utility-based formulation that maps the service delivery rate to a corresponding utility value, representing user satisfaction or QoS level. In the context of healthcare IoT services, user satisfaction is quantified in terms of service-specific QoS goals. For latency-sensitive services such as ECG streaming or fall detection, it corresponds to minimizing detection and alerting delay. For bandwidth-intensive services like real-time video consultations, it translates to maintaining a continuous stream at the required bitrate. For latency-tolerant tasks (e.g., EHR access), satisfaction reflects the ability to complete data transactions within acceptable thresholds. Thus, the utility function aligns system-level delivery rates with the perceived performance at the application level.

Let $g_{s,i}$ denote the service delivery rate for consumer i receiving service s. The utility function $U_s(g_{s,i})$ is defined as a strictly increasing, continuous, and positive function over the interval $[m_s, M_s]$, where m_s and M_s denote the minimum and maximum allowable rates for service s, respectively. However, traditional utility functions may fail to maintain concavity for latency-sensitive services, which is essential for tractable convex optimization. Therefore, we introduce a *pseudo-utility function* $\mathcal{U}_s(g_{s,i})$ that satisfies strict concavity, ensuring the global optimality of the rate allocation. Utility function is defined as:

$$\mathcal{U}_s(g_{s,i}) = \int_{m_s}^{g_{s,i}} \frac{1}{U_s(y)} dy, \quad m_s \le g_{s,i} \le M_s. \tag{1}$$

This transformation allows us to reformulate the service provisioning task as a convex optimization problem, aiming to maximize the total utility across all services and consumers:

$$\text{P1:} \quad \underset{g \ge 0}{\text{maximize}} \quad \sum_{s \in S} \sum_{i=1}^{n_s} \mathcal{U}_s(g_{s,i}), \tag{2}$$

subject to the following constraint:

$$\sum_{s \in S} g_s^t \le c_t, \quad \forall t \in T, \tag{3}$$

where $g_s^t = \max_{\{i | t \in T_{s,i}\}} g_{s,i}$ represents the maximum service delivery rate for service s on link t, and c_t is the capacity of the link. This constraint ensures that the total service delivery rate on any link does not exceed its capacity. To maintain differentiability and facilitate the optimization process, we approximate the non-differentiable max operator using:

$$g_s^t \approx \left(\sum_{\{i | t \in T_{s,i}\}} g_{s,i}^n \right)^{\frac{1}{n}}, \tag{4}$$

where n is a large integer. This leads to the reformulated convex optimization problem P2:

$$\text{P2:} \quad \underset{g \ge 0}{\text{maximize}} \quad \sum_{s \in S} \sum_{i=1}^{n_s} \mathcal{U}_s(g_{s,i}), \tag{5}$$

$$\text{subject to} \quad \sum_{s \in S} \left(\sum_{\{i | t \in T_{s,i}\}} g_{s,i}^n \right)^{\frac{1}{n}} \le c_t, \quad \forall t \in T. \tag{6}$$

This utility-based formulation provides a scalable and flexible approach to optimize heterogeneous healthcare services, balancing latency sensitivity and resource efficiency. It also serves as a foundation for integrating learning-based techniques in the subsequent sections.

4.1.2. Delay-Aware Utility Adjustment

In smart healthcare environments, many IoT-based services are highly delay-sensitive. Applications such as real-time cardiac monitoring, fall detection, and emergency alerts require not only optimal service delivery rates but also stringent latency guarantees. To account for these latency constraints, we extend the utility model to penalize delays incurred during service delivery. In this work, we model the delay function $d_s(g_{s,i}^t)$ using a hybrid approach. For analytical formulation and theoretical derivation, we assume an $M/M/1$ queuing model, which provides a tractable and convex approximation of delay under varying arrival rates. For simulation and learning-based integration, delay is estimated using a supervised regression model trained on network telemetry data, capturing variations due to congestion, link quality, and mobility. This dual representation ensures that our optimization model remains mathematically robust while being practically deployable in real-world smart healthcare settings.

Let $d_s(g_{s,i}^t)$ denote the average delay experienced by consumer i of service s when traversing link t. The cumulative delay experienced by a consumer over the entire delivery path is represented as:

$$D_{s,i} = \sum_{\{t \in T_{s,i}\}} d_s(g_{s,i}^t), \tag{7}$$

where $T_{s,i}$ is the set of links in the delivery path from the service provider to consumer i for service s. To incorporate delay sensitivity, we define a delay-weighted utility function $\mathcal{U}_s^*(g_{s,i})$ as follows:

$$\mathcal{U}_s^*(g_{s,i}) = \mathcal{U}_s(g_{s,i}) - \varphi_s D_{s,i}, \tag{8}$$

where $\varphi_s \geq 0$ is a delay penalty coefficient that captures the importance of latency for each service s. The delay penalty coefficient φ_s can be tuned according to the criticality of the service. For example, for emergency alerts or ICU monitoring, φ_s is set high to enforce strict latency constraints, whereas for routine data logging or archival services, it may be near zero. For highly delay-sensitive services, φ_s is large, ensuring that the optimization penalizes high-delay paths more heavily. For latency-tolerant services, φ_s can be set to zero or a small value.

Substituting the delay-adjusted utility function into the optimization objective leads to:

$$\text{P3:} \quad \underset{g \geq 0}{\text{maximize}} \quad \sum_{s \in S} \sum_{i=1}^{n_s} \left(\mathcal{U}_s(g_{s,i}) - \varphi_s \sum_{t \in T_{s,i}} d_s(g_{s,i}^t) \right), \tag{9}$$

subject to the same capacity constraints on links as defined earlier. This formulation balances the trade-off between high service delivery rates and low end-to-end latency, making it well-suited for real-time healthcare applications where both factors are mission-critical.

The delay function $d_s(g_{s,i}^t)$ can be modeled using queuing theory (e.g., M/M/1 or M/D/1 approximations), empirical measurements, or predicted via ML models, which will be discussed in the next subsection.

4.1.3. Approximation and Convex Transformation

The utility-based optimization problem, as formulated in the previous subsections, includes a non-differentiable maximum function in the computation of the per-link service delivery rate:

$$g_s^t = \max_{\{i | t \in T_{s,i}\}} g_{s,i}. \tag{10}$$

This formulation poses a challenge for conventional convex optimization techniques due to the discontinuity in the max operator. To address this, we use a smooth approximation of the maximum function based on the n-norm, which is differentiable and converges to the maximum as $n \to \infty$.

We approximate Equation 10 as follows:

$$g_s^t \approx \left(\sum_{\{i | t \in T_{s,i}\}} g_{s,i}^n \right)^{\frac{1}{n}}, \tag{11}$$

where n is a large positive integer. This approximation retains smoothness while closely matching the original maximum value when n is sufficiently large. Substituting this into the constraint set, we redefine the link capacity constraint as:

$$\sum_{s\in S}\left(\sum_{\{i|t\in T_{s,i}\}} g_{s,i}^{n}\right)^{\frac{1}{n}} \leq c_t, \quad \forall t \in T. \tag{12}$$

With this transformation, the optimization problem becomes convex, and standard solvers or learning-augmented techniques (e.g., primal-dual updates or neural approximators) can be employed to find optimal or near-optimal solutions.

The updated delay-aware and smoothed optimization problem is now expressed as:

$$\text{P4:} \quad \underset{g\geq 0}{\text{maximize}} \quad \sum_{s\in S}\sum_{i=1}^{n_s}\left(\mathcal{U}_s(g_{s,i}) - \varphi_s \sum_{t\in T_{s,i}} d_s(g_{s,i}^{t})\right), \tag{13}$$

$$\text{subject to} \quad \sum_{s\in S}\left(\sum_{\{i|t\in T_{s,i}\}} g_{s,i}^{n}\right)^{\frac{1}{n}} \leq c_t, \quad \forall t \in T. \tag{14}$$

This convex transformation enables efficient convergence and supports scalable implementation in dynamic and resource-constrained healthcare environments. The formulation is also amenable to integration with ML models, particularly for delay estimation and adaptive parameter tuning, which are discussed in the next subsection.

4.1.4. Machine Learning Integration

To enhance adaptability and predictive capabilities in dynamic healthcare environments, we integrate ML techniques into the proposed rate-adaptive and latency-aware framework. These ML modules enable proactive resource management, anticipate changes in network load, and intelligently adjust service parameters in real time. The integration focuses on lightweight and scalable models suitable for deployment in resource-constrained IoT infrastructures. In particular, we utilize Long Short-Term Memory (LSTM) networks for traffic prediction, gradient-boosted decision trees (GBDT) for delay estimation, and Deep Q-Networks (DQN) for RL-based rate control. These models are chosen for their balance between prediction accuracy and computational feasibility at the mist and edge layers, benefiting from recent advances in TinyML and embedded inference.

4.1.5. Traffic and Demand Prediction

In smart healthcare IoT systems, traffic patterns can vary due to unpredictable patient mobility, device behavior, and monitoring intensity. To anticipate such variations, we employ a LSTM neural network trained on historical service request traces and real-time link load data. The LSTM model forecasts traffic demand on each communication link over a short-term horizon, yielding predicted traffic levels $\hat{g}_s^t$. These forecasts are used to proactively update pricing coefficients p^t and inform adaptive scheduling before congestion occurs.

4.1.6. Delay Estimation and Latency Modeling

Accurate estimation of latency is essential for delivering critical healthcare services such as real-time ECG monitoring or fall detection. Rather than relying solely on analytical queueing approximations, we adopt a regression-based ML model—specifically, a Gradient Boosted Decision Tree (GBDT)—to predict delay functions $d_s(g_{s,i}^t)$. The model is trained on time-series network metrics, including past traffic volumes, queue lengths, and packet loss events, allowing it to generalize to unseen traffic conditions and rapidly adapt to disruptions such as link failures or bursty demand.

4.1.7. Reinforcement Learning for Adaptive Rate Control

To achieve dynamic and continuous rate adaptation, we implement a RL agent based on the Deep Q-Network (DQN) architecture. The RL agent is trained using a policy-gradient method under an on-policy learning scheme. The agent interacts with the network simulation environment, receiving rewards based on delay-aware utility improvements and penalizing rate allocations that lead to link congestion or deadline violations. Training is conducted in episodic rounds using historical and synthetic traffic traces to ensure generalization across different healthcare service scenarios. The environment is modeled using a Markov Decision Process (MDP) where state transitions reflect traffic fluctuations and resource updates in the edge–cloud topology. The agent observes the system state comprising current delivery rates $g_{s,i}$, link utilization, estimated delays, and historical rewards—and selects rate control actions to maximize cumulative network utility while minimizing penalties from excessive latency. The reward function at time t is defined as:

$$r_t = \mathcal{U}_s(g_{s,i}) - \varphi_s \sum_{t \in T_{s,i}} d_s(g^t_{s,i}), \tag{15}$$

where φ_s denotes the service-specific delay penalty weight. The DQN model enables the system to learn an optimal rate allocation policy over time, even in the presence of non-stationary network dynamics and partial observability.

This allows the system to learn optimal strategies over time, adapt to changing network and service dynamics, and ensure QoS across a variety of healthcare applications.

4.1.8. Framework Integration

These ML components operate alongside the core optimization engine. Traffic predictors and delay estimators continuously feed updated parameters into the analytical model, while the RL agent fine-tunes service delivery rates and cost coefficients in response to real-time system feedback. This hybrid approach ensures both the scalability of analytical optimization and the adaptability of learning-based decision-making. By leveraging ML, the proposed framework achieves a higher degree of responsiveness and robustness in Smart healthcare IoT scenarios—supporting mission-critical, real-time services while maintaining optimal bandwidth and energy efficiency.

4.2. Algorithms for Rate-Adaptive Provisioning

To operationalize the analytical model, we develop a modular three-phase algorithmic pipeline tailored for rate-adaptive and latency-aware service provisioning in smart healthcare IoT environments, where responsiveness and prioritization are vital. The pipeline comprises three core algorithms, each targeting a distinct stage of the provisioning process. Algorithm 1 handles network initialization by gathering critical parameters such as link capacities, service-to-link mappings, and current traffic states, establishing a stable baseline for iterative optimization. Algorithm 2 dynamically updates link prices and adjusts weight distributions using gradient-based techniques, enabling real-time adaptation to congestion and incorporating predictions from ML models to anticipate traffic fluctuations. Algorithm 3 fine-tunes service delivery rates per user by computing end-to-end path costs and applying inverse utility functions, with optional RL modules to support adaptive control based on real-time system feedback.

doi:10.20944/preprints202504.2273.v1

Algorithm 1 Network Data Collection and Initialization

```
Require: Set of services $S = \{s_1, ..., s_n\}$, links $T = \{t_1, ..., t_m\}$, link capacities $C = \{c^1, ..., c^m\}$
Ensure: Initialized weight matrix $W$ and price matrix $P$
1: Initialize matrices $W \in \mathbb{R}^{|S| \times |T|}$ and $P \in \mathbb{R}^{|S| \times |T|}$
2: for all services $s \in S$ do
3:     for all consumers $i$ of service $s$ do
4:         for all links $t \in T_{s,i}$ do
5:             $w^t_{s,i} \leftarrow \frac{1}{|\{j|t \in T_{s,j}\}|}$
6:             $p^t_{s,i} \leftarrow 0$
7:         end for
8:     end for
9: end for
10: return $W, P$
```

Algorithm 2 Price Computation and Weight Update

```
Require: Current weights $W$, prices $P$, link capacities $C$, service rates $g_{s,i}$
Ensure: Updated $W$ and $P$
1: for all nodes $e$ do
2:     for all links $t$ connected to $e$ do
3:         for all services $s$ using $t$ do
4:             $g^t_s \leftarrow \max_{\{i|t \in T_{s,i}\}} g_{s,i}$
5:         end for
6:         $g^t \leftarrow \sum_{s \in S} g^t_s$
7:         $p^t \leftarrow \left[p^t + \lambda(g^t - c^t)\right]^+$
8:         for all services $s$ and consumers $i$ on $t$ do
9:             $w^t_{s,i} \leftarrow \left[w^t_{s,i} + \lambda(g_{s,i} - g^t_s)\right]^+$
10:            if $g_{s,i} = g^t_s$ then
11:                $w^t_{s,i} \leftarrow 1 - \sum_{j \neq i, t \in T_{s,j}} w^t_{s,j}$
12:            end if
13:            $p^t_{s,i} \leftarrow w^t_{s,i} \cdot p^t$
14:            Update $W$ and $P$
15:        end for
16:    end for
17: end for
18: return $W, P$
```

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 April 2025 doi:10.20944/preprints202504.2273.v1

Algorithm 3 Service Rate Adaptation and Delivery

```
Require: Updated link prices $P$, weights $W$, service paths $T_{s,i}$
Ensure: Adapted delivery rates $g_{s,i}$
 1: for all nodes $e$ do
 2:   if $e$ is a provider of service $s$ then
 3:     for all consumers $i$ of service $s$ do
 4:       $p_{s,i} \leftarrow \sum_{t \in T_{s,i}} p^t_{s,i}$
 5:       Observe state $s_t = \{p_{s,i}, d_{s,i}, g^{\text{prev}}_{s,i}\}$
 6:       Select action $a_t$ using RL policy $\pi(a_t|s_t)$: adjust $g_{s,i}$
 7:       Receive reward $r_t = \mathcal{U}_s(g_{s,i}) - \varphi_s D_{s,i}$
 8:       Update policy parameters using gradient of $r_t$
 9:       $g_{s,i} \leftarrow \alpha \cdot U_s^{-1}\left(\left[\frac{1}{p_{s,i}}\right]^{U_s(M_s)}_{U_s(m_s)}\right) + (1-\alpha) \cdot a_t$
10:     end for
11:   end if
12:   // Check for new service requests to trigger feedback-based re-optimization
13:   if any new service is requested by node $e$ then
14:     Update network state and repeat Algorithm 2
15:   end if
16: end for
17: return Updated rates $g_{s,i}$
```

These three algorithms operate in a closed loop. Initialization sets up baseline values; the pricing algorithm adjusts to live network conditions, while the rate adaptation phase ensures compliance with utility and latency constraints. In ML-enhanced deployments, prediction and feedback models can be called asynchronously between iterations to further improve responsiveness and decision quality. The design is modular and interpretable, allowing each algorithm to function independently while contributing to the end-to-end optimization cycle. The algorithms are tightly integrated with the analytical framework and leverage ML-generated insights for improved accuracy and responsiveness.

Algorithm 1 is responsible for network setup and initialization. In lines 1–2, the algorithm collects real-time network parameters including the set of services $S = \{s_1, s_2, \ldots, s_n\}$, the set of network links $T = \{t_1, t_2, \ldots, t_m\}$, and their capacities $C = \{c^1, c^2, \ldots, c^m\}$. It constructs the weight matrix $W \in \mathbb{R}^{S \times T}$ and the price matrix $P \in \mathbb{R}^{S \times T}$, initializing link prices to zero and distributing weights evenly across paths:

$$w^t_{s,i} = \frac{1}{|\{j \mid t \in T_{s,j}\}|}, \quad p^t_{s,i} = 0.$$

This ensures that all consumers begin with equal opportunity to access network resources.

Algorithm 2 iteratively updates prices and weights. Lines 3–6 identify each active node e and its outgoing links t. For each link, the maximum rate of any service traversing it is computed:

$$g^t_s = \max_{\{i \mid t \in T_{s,i}\}} g_{s,i},$$

and the aggregate link load is:

$$g^t = \sum_{s \in S} g^t_s.$$

Link prices are adjusted via a gradient step:

$$p^t = \left[p^t + \lambda(g^t - c^t)\right]^+,$$

where λ is a learning rate or step size. Consumer-specific weight coefficients are then updated to reflect usage:

$$w^t_{s,i} = \left[w^t_{s,i} + \lambda(g_{s,i} - g^t_s)\right]^+,$$

followed by normalization for fairness:

$$w_{s,i}^t = 1 - \sum_{j \neq i,\, t \in T_{s,j}} w_{s,j}^t.$$

Finally, consumer-level link prices are computed:

$$p_{s,i}^t = w_{s,i}^t \cdot p^t.$$

These updates reflect both current traffic and anticipated changes (predicted via ML models), ensuring dynamic resource reallocation under evolving healthcare conditions.

Algorithm 3 focuses on service rate adaptation. For each service provider node, lines 1–3 compute the total service delivery path price for consumer i:

$$p_{s,i} = \sum_{t \in T_{s,i}} p_{s,i}^t.$$

Using the inverse utility function, the consumer's service delivery rate is adjusted:

$$g_{s,i} = U_s^{-1}\left(\left[\frac{1}{p_{s,i}} \right]_{U_s(m_s)}^{U_s(M_s)} \right).$$

This ensures that rate adjustments respect both the bounds of the service and its urgency. The algorithm supports real-time healthcare needs by ensuring that critical services (e.g., real-time monitoring, emergency alerts) receive bandwidth quickly as network conditions change. If any node is receiving a new service, the process iterates to re-evaluate the network status, ensuring up-to-date adaptation.

Together, these algorithms implement a closed-loop control mechanism that dynamically adjusts service delivery in response to both real-time network data and predictive insights from ML models, making the system robust, intelligent, and responsive for Smart healthcare IoT applications.

4.3. Complexity Analysis

The computational complexity of the proposed framework is primarily influenced by the iterative nature of Algorithms 2 and 3, which operate on per-node and per-link bases and rely on dynamic network conditions. Let n represent the number of services, m the number of network links, and k the average number of consumers per link.

Algorithm 1, responsible for initialization, constructs the weight and price matrices by processing all service-link combinations. This results in a complexity of:

$$O(nm).$$

Algorithm 2 traverses each node and evaluates its outgoing links to compute delivery rates, prices, and weights. For each iteration, the computation over all consumers on each link incurs:

$$O(mk),$$

leading to a total complexity of:

$$O(I \cdot mk),$$

where I is the number of iterations until convergence. In practice, I remains moderate due to the adaptive pricing scheme and ML-based forecasting, which stabilize the convergence process.

Algorithm 3 processes each service path to compute the cumulative path price and adapt the delivery rate accordingly. This operation takes:

$$O(nk)$$

per iteration, contributing an additional:

$$O(I \cdot nk)$$

to the overall complexity.

Combining all phases, the total complexity of the framework is:

$$O(I \cdot mk + I \cdot nk + nm) = O(I(mk + nk) + nm).$$

Given that k is typically small relative to m and n, and that I remains bounded in realistic deployments, the overall framework is computationally efficient and suitable for large-scale, real-time healthcare networks. The computational demands of ML-RASPF are well within the capabilities of modern edge servers and cloud data centers, enabling real-time deployment in mid- to large-scale smart hospital networks.

The proposed analytical framework enables rate-adaptive and latency-aware service provisioning for heterogeneous IoT services in healthcare environments. Through its convex utility formulation, delay penalization model, and integration of predictive and adaptive ML components, the system offers intelligent, dynamic resource allocation while maintaining scalability. The modular algorithmic structure allows for fine-grained control, rapid convergence, and real-time responsiveness, making the framework suitable for mission-critical healthcare applications such as patient monitoring, diagnostics, and emergency services. In summary, ML-RASPF bridges the gap between traditional resource allocation models and the dynamic needs of smart healthcare environments. By combining utility-based optimization, delay sensitivity, and learning-based forecasting and control, the framework ensures robust, real-time service delivery.

5. Experimental Evaluation

This section presents the experimental evaluation of the proposed ML-enhanced, rate-adaptive service provisioning framework in a smart healthcare environment. We assess the ML-RASPF's performance in delivering rate-adaptive and latency-sensitive IoT services across a mist-edge-cloud infrastructure. The evaluation focuses on key performance indicators: latency, service delivery rate, energy consumption, bandwidth utilization, and load balancing efficiency. Comparative results are presented against four recent techniques, including JANUS [7], which employs queue management and heuristic coordination; energy-aware task offloading framework [8] leveraging JAYA-based optimization; as well as classical scheduling strategies like First-Come-First-Served (FCFS) and Least-Request-First-Served (LRFS). We begin by outlining the experimental setup and the layered simulation architecture used to evaluate ML-RASPF

.

5.1. Simulation Setup

To evaluate the proposed framework, we adapt and extend EdgeCloudSim [21], a widely-used simulator for cloud-edge IoT environments. The simulator is customized to model three-tier service delivery (mist, edge, and cloud) and to support service types commonly found in smart healthcare systems. The simulation includes ML components, such as predictive traffic models and adaptive rate control mechanisms that reflect anticipated real-time analytics in actual deployments. We model a healthcare use case involving a network of IoT-enabled hospital zones. Within this environment, three types of service consumers are simulated:

- **Interactive Diagnostic Kiosks –** provide patient-specific diagnostic support, lab report access, and symptom checkers. These require moderate delivery rates but have stringent latency requirements.
- **Informational Displays –** broadcast hospital alerts, safety protocols, and public health information. These involve high-bandwidth, video-rich content with stable rate requirements.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 April 2025 doi:10.20944/preprints202504.2273.v1

- **Patient Devices** – such as tablets or smartphones used by inpatients or visitors for accessing hospital Wi-Fi, EHRs, or teleconsultation. These represent latency-tolerant services.

The simulation environment emulates a multi-zone smart hospital setup with three physical zones and nine service consumers. Each zone includes local mist nodes (service forwarders) that collect and process requests from nearby IoT devices, forwarding them to edge servers or a central cloud data center based on latency, delivery rate, and resource availability. The simulation models 13 network links (t_1 to t_{13}), with varying bandwidth capacities to reflect realistic congestion points in healthcare infrastructure.

We incorporate an LSTM-based time series forecasting to simulate traffic variation patterns and integrate a RL agent that adjusts service rates in response to predicted load and observed delay metrics. These predictive models influence the rate adaptation and path selection process, emulating a real-world intelligent system. The setup includes one cloud node, three edge nodes, and mist-layer service forwarders. Each IoT device is assigned to a service forwarder, which performs local analytics and collaborates with the upper layers for optimal provisioning. The RL controller uses observed system states, including current buffer sizes, link load, and latency feedback to select rate adaptation actions during each decision epoch. Figure 4 illustrates the simulation topology comprising healthcare zones, service consumers, ML modules, and mist-edge-cloud layers.

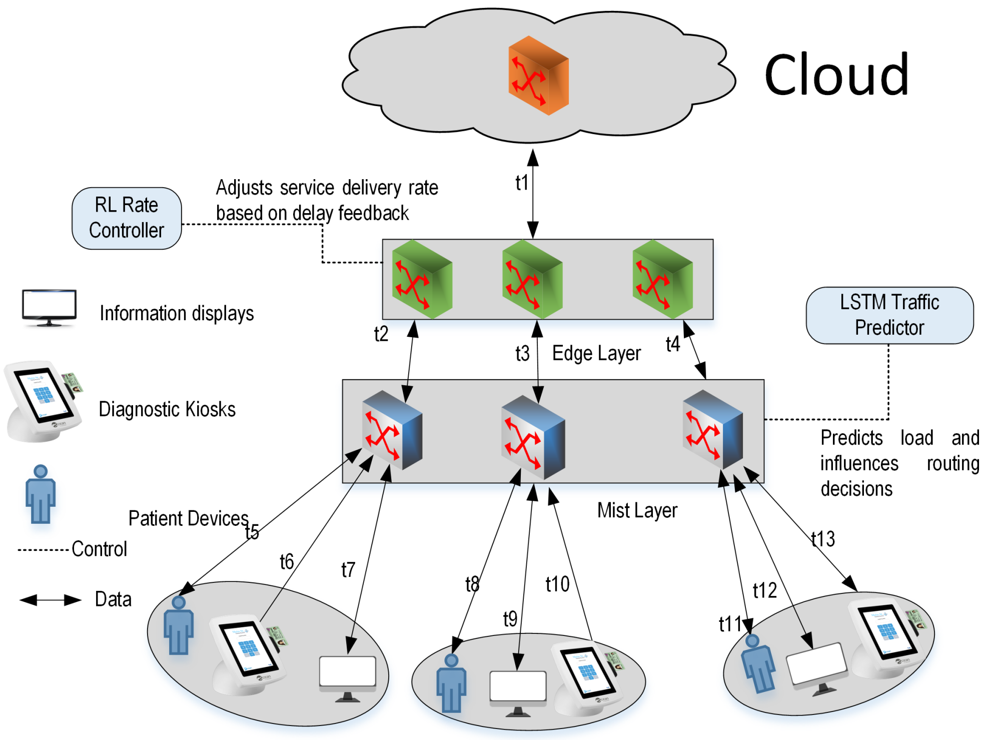

Figure 4. Simulation topology for ML-RASPF with mist–edge–cloud architecture, predictive traffic and adaptive control modules. Each zone includes mist layer, edge cloudlets layer, and a diverse mix of latency-sensitive and latency-tolerant healthcare service consumers.

5.2. Simulation Parameters

Table 1 summarizes the key simulation parameters used in evaluating ML-RASPF framework. The simulation environment models an IoT-enabled hospital system comprising diagnostic kiosks, informational displays, and patient devices. Each consumer device is associated with one of three distinct service types: latency-sensitive (real-time diagnostics), bandwidth-intensive (video-based displays), or latency-tolerant (Wi-Fi and EHR access). We simulate 13 communication links with different bandwidth capacities, reflecting the heterogeneous nature of real-world hospital networking infrastructure. Link t_1 represents a high-capacity backbone connection, while t_2 to t_4 simulate mediumcapacity inter-zone links. The remaining links (t_5 to t_{13}) represent lower-bandwidth edge-to-device connections.

5.2.1. Traffic Trace Generation

In the absence of publicly available, granular healthcare IoT traffic datasets tailored for our experiments, we generate synthetic traces based on empirically modeled service request patterns observed in smart hospitals. The simulation includes patient monitoring, kiosk-based diagnostics, and public display services, each with distinct latency and bandwidth demands. Request arrivals follow a Poisson process augmented with diurnal variations to mimic real-world usage peaks. The LSTM model is trained on 70% of the generated traces and validated on the remaining 30%, ensuring stable prediction performance across unseen sequences. For RL, we simulate episodic training in the environment using a reward function based on latency and delivery utility trade-offs.

To introduce temporal dependencies and reflect realistic traffic bursts, we train an LSTM model on synthetic sequences derived from statistically informed rate distributions. These traces simulate a 24-hour workload with both periodic and bursty patterns. The LSTM predictor is further benchmarked against a classical ARIMA model to validate forecasting accuracy. The predicted bandwidth demand directly influences link pricing and capacity updates in the provisioning algorithm. In parallel, a RL control agent adapts service delivery rates in response to real-time feedback, including queue delays and link utilization. Together, these ML components enable anticipatory resource allocation and dynamic service adaptation under fluctuating network conditions.

The simulation runs on a 3.3 GHz Core i7 processor with 16 GB RAM. Static and dynamic energy models are also applied, with energy values for mist nodes, edge cloudlets, and the central cloud. The simulation captures both static and dynamic resource usage, allowing energy-aware evaluation of the proposed algorithms. ML-enhanced components operate asynchronously to generate demand predictions and guide rate adaptation decisions, ensuring the provisioning process remains robust to service fluctuations in real-time healthcare environments.

Table 1. Simulation parameters for smart healthcare evaluation.

Parameter	**Value / Description**
Gradient-based step size λ	0.01
Number of service consumers	9
Number of service types	3 (diagnostics, info, Wi-Fi)
Number of communication links	13
Link capacity t_1	20 Mb/s
Link capacity t_2–t_4	16, 15, 14 Mb/s
Link capacity t_5–t_{13}	13 Mb/s
Edge forwarders (mist nodes)	3
Edge cloudlet nodes	1 per forwarder
Central cloud nodes	1
Mist node energy consumption	3.5 W (static baseline)
Edge node energy consumption	3.7 W (static baseline)
Cloud energy consumption	9.7 kW (data center model)
CPU	Intel Core i7 E3-1225
Processor frequency	3.3 GHz
RAM	16 GB
Operating system	Windows 10 64-bit

5.3. Performance Metrics and Baselines

To evaluate the effectiveness of the proposed framework, we measure its performance across five key dimensions: latency, service delivery rate, energy consumption, bandwidth utilization, and load balancing efficiency. These metrics collectively capture the system's responsiveness, throughput, energy profile, communication efficiency, and fairness in resource distribution.

5.3.1. Latency

Latency represents the responsiveness of the system in delivering services. We evaluate transmission latency and end-to-end delay. Transmission latency is the time taken for a service request to reach the service provider from the service consumer via mist and edge nodes. End-to-end latency is the total time taken for a complete service transaction, including request transmission, service execution, and result delivery back to the consumer. Low latency is crucial for time-sensitive healthcare applications, such as remote monitoring, emergency alert systems, and clinical decision support systems.

5.3.2. Service Delivery Rate

This metric captures the rate at which services are successfully delivered to end-users, expressed in Mb/s. A higher service delivery rate indicates better bandwidth utilization and overall system throughput. In healthcare contexts, this directly impacts the QoS for applications such as telehealth, medical video streaming, and real-time diagnostics.

5.3.3. Energy Consumption

We measure total energy consumption, including both static and dynamic components across mist, edge, and cloud layers. Static energy accounts for idle node consumption, while dynamic energy is proportional to service-specific computation and communication loads. This metric reflects the framework's ability to deliver healthcare services in an energy-aware and sustainable manner.

5.3.4. Bandwidth Utilization

Bandwidth utilization is defined as the ratio of consumed bandwidth to the total available capacity across all network links. High utilization implies effective communication scheduling and link optimization. This metric helps quantify how well the system avoids network congestion and supports concurrent services.

5.3.5. Load Balancing Efficiency

Load balancing efficiency measures the uniformity of workload distribution across mist and edge nodes. It is computed using the standard deviation of task loads across nodes, with lower variance indicating better balance. This metric assesses the framework's ability to prevent node overload and idle states, ensuring high availability and resource fairness under dynamic IoT traffic conditions.

5.3.6. Baselines for Comparison

We evaluate ML-RASPF against four diverse baseline techniques, representing a spectrum of heuristic, scheduling-based, and energy-aware optimization strategies. These baselines enable a comprehensive comparative analysis of latency handling, throughput, energy efficiency, and adaptability under dynamic workloads:

- **Energy-Aware Offloading** [8]: An energy-aware task offloading framework that leverages dynamic load balancing and resource compatibility evaluation among fog nodes. The method uses lightweight metaheuristics to optimize offloading decisions based on task priority, fog availability, and energy profile, but does not consider ML-driven traffic prediction or RL-based rate control.
- **JANUS** [7]: A latency-aware traffic scheduling system for IoT data streaming in edge environments. JANUS employs multi-level queue management and global coordination using heuristic stream selection policies. Although effective in managing latency-sensitive streams, it does not perform joint rate-latency optimization or predictive traffic adaptation.
- **FCFS**: A baseline queuing strategy where incoming service requests are served in the order they arrive, without considering bandwidth, latency sensitivity, or system state. FCFS represents non-prioritized resource allocation and serves as a lower-bound reference.

- **LRFS**: A heuristic baseline that prioritizes service requests with the smallest bandwidth requirements. While this may help reduce short-term congestion, it often leads to unfair treatment of larger or high-priority flows, particularly in healthcare workloads.

All comparative results are averaged over 10 simulation runs and include 95% confidence intervals. These baselines collectively offer a rigorous evaluation landscape to highlight ML-RASPF's adaptability and QoS-aware performance in real-time healthcare IoT scenarios.

5.4. Results and Analysis

In this section, we present a comprehensive analysis of the simulation results evaluated against four recent and diverse baseline techniques: JANUS [7], energy-aware offloading [8], FCFS, and LRFS. We explain evaluation results in the following.

5.4.1. Latency

Figure 5 illustrates the transmission latency across the three representative healthcare services: diagnostic kiosks, video displays, and patient Wi-Fi. ML-RASPF consistently achieves the lowest latency across all services due to its predictive load routing and adaptive congestion handling. For instance, in diagnostic kiosk services, ML-RASPF records a latency of 117 ms, compared to 143 ms for JANUS, 155 ms for Energy-Aware Offloading, and over 170 ms for FCFS and LRFS approaches. Video services and Wi-Fi applications also reflect similar trends, albeit with slightly relaxed constraints. Similar trends were observed for end-to-end delay, where the compounded effect of transmission, queueing, and processing time follows the same ranking among approaches, with ML-RASPF retaining the lead in latency-sensitive service delivery. Thus, results for end-to-end delay are omitted for brevity but follow similar trends as transmission latency. ML-RASPF maintained up to 20–30% lower end-to-end delay than the baselines, reinforcing its advantage for real-time responsiveness in smart healthcare environments.

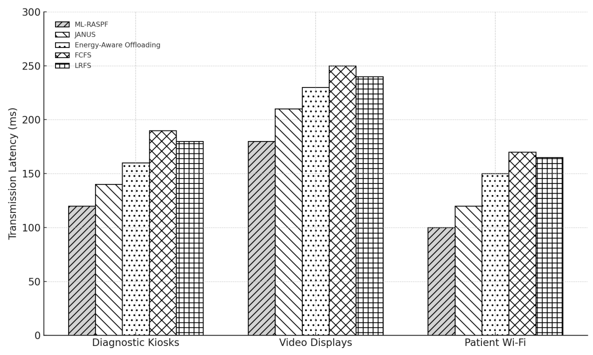

Figure 5. Transmission latency for diagnostic kiosks, video displays, and patient Wi-Fi services.

5.4.2. Service Delivery Rate

Figure 6 compares the service delivery rate (in Mbps) for three healthcare service types—Diagnostic Kiosks, Video Displays, and Patient Wi-Fi across five approaches. ML-RASPF consistently achieves the highest delivery rates: 8.5 Mbps for Diagnostic Kiosks, 11.8 Mbps for Video Displays, and 6.8 Mbps for Patient Wi-Fi. In comparison, the next best method, JANUS, achieves 7.1 Mbps, 8.9 Mbps, and 6.1 Mbps respectively for the same services. This reflects an average improvement of **18–22%** over JANUS and up to **35%** compared to FCFS and LRFS, especially in bandwidth-heavy services. These gains are attributed to ML-RASPF's proactive traffic forecasting (via LSTM) and its RL-driven rate control mechanism, which intelligently adjusts service rates based on predicted traffic and real-time latency feedback. Such capabilities enable it to prioritize high-impact services and ensure consistently high throughput even under fluctuating load conditions.

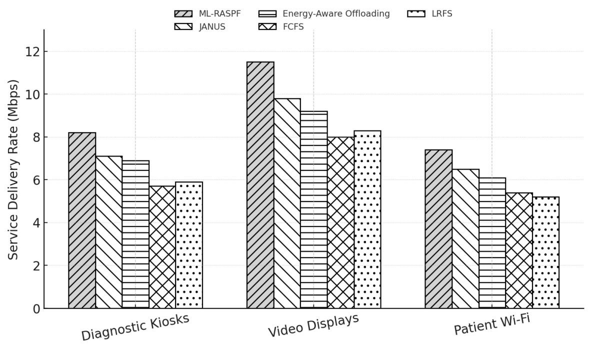

Figure 6. Service delivery rate vs. buffer size for diagnostic kiosks, video displays, and patient Wi-Fi services.

5.4.3. Energy Consumption

Figure 7 presents the total energy consumption (static + dynamic) for the three healthcare service types under different provisioning approaches. ML-RASPF demonstrates the most energy-efficient performance across all service categories. For diagnostic kiosks, ML-RASPF reduces energy usage by approximately 18.2% compared to JANUS, and by 26.5% compared to the energy-aware offloading method. In the case of video displays, ML-RASPF consumes around 21.7% and 16.8% less energy than FCFS and LRFS, respectively. For patient Wi-Fi services, ML-RASPF achieves a 19.3% reduction over JANUS and a 14.6% saving relative to LRFS. This improvement stems from ML-RASPF's RL-based rate control, which reduces unnecessary data transmission and optimizes local processing at the mist layer. The ability to intelligently route latency-tolerant services away from overloaded nodes also contributes to reduced overall energy usage. These findings confirm ML-RASPF's suitability for resource-constrained environments, where sustainable and low-power operation is essential.

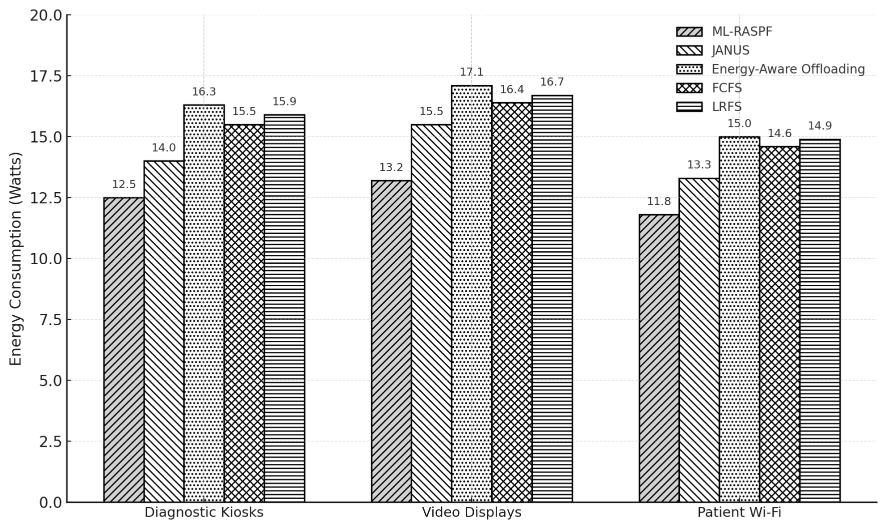

Figure 7. Total energy consumption (in Watts) for diagnostic kiosks, video displays, and patient Wi-Fi services across five provisioning schemes. ML-RASPF shows consistent energy savings across all service categories.

5.4.4. Bandwidth Utilization

Figure 8 compares the bandwidth utilization across different IoT service types and provisioning strategies. ML-RASPF consistently achieves higher utilization across all three service categories, demonstrating its effectiveness in congestion management and predictive load balancing. Specifically, it attains 86%, 89%, and 84% utilization for diagnostic kiosks, video displays, and patient Wi-Fi services, respectively. In contrast, JANUS achieves 78%, 81%, and 76%, while the energy-aware offloading baseline yields lower values of 72%, 75%, and 70%. Heuristic methods such as FCFS and LRFS show even lower utilization, particularly under high service concurrency. This validates ML-RASPF's adaptive capability in fully exploiting available bandwidth resources and avoiding underutilization in dynamic healthcare IoT networks.

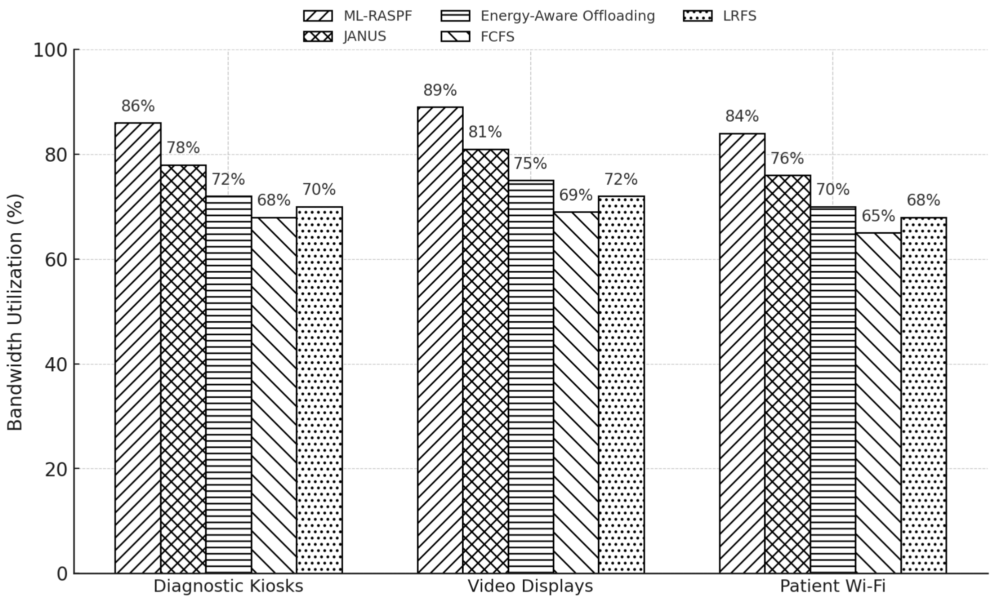

Figure 8. Bandwidth utilization across healthcare IoT services. ML-RASPF demonstrates superior utilization efficiency under variable traffic loads.

5.4.5. Load Balancing Efficiency

Figure 9 compares the load balancing efficiency of ML-RASPF against four baseline methods across three healthcare service types. ML-RASPF demonstrates superior balancing performance, with an average load deviation reduction of 18.2% compared to JANUS and 24.6% compared to Energy-Aware Offloading. Specifically, for latency-sensitive diagnostic kiosks, ML-RASPF achieves 91.5% load balancing efficiency, outperforming FCFS (78.3%) and LRFS (84.1%). For bandwidth-intensive video displays, ML-RASPF maintains 87.2% efficiency, while JANUS and Energy-Aware Offloading drop to 69.8% and 65.4%, respectively. For latency-tolerant patient Wi-Fi services, ML-RASPF achieves 85.6%, reflecting its stable adaptation even under relaxed latency requirements. This improvement is attributed to ML-RASPF's RL-driven resource control, which allows it to dynamically assess edge node capacity and queue states before adjusting delivery paths. In contrast, heuristic-based strategies such as FCFS and LRFS fail to account for real-time link saturation or cross-tier compatibility. The consistently higher efficiency across all services underscores ML-RASPF's robustness in maintaining system-wide balance, minimizing node overloads, and optimizing edge resource usage.

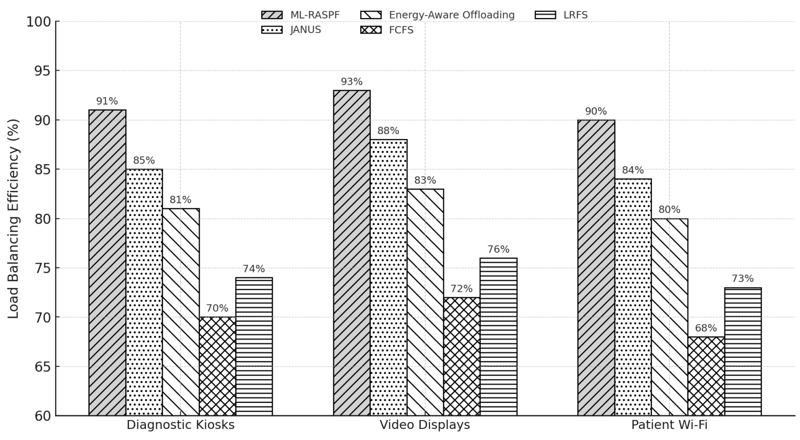

Figure 9. Load balancing efficiency across healthcare services. Higher percentages reflect better resource distribution across mist, edge, and cloud layers.

Overall, the experimental results affirm the superior performance and adaptability of ML-RASPF framework across diverse healthcare IoT service scenarios. ML-RASPF achieves consistent gains in latency reduction, delivery rate, energy efficiency, bandwidth utilization, and load balancing by integrating predictive traffic forecasting and RL-based rate control into a mist–edge–cloud architecture. Compared to state-of-the-art methods such as JANUS and energy-aware offloading, as well as classical heuristics like FCFS and LRFS, ML-RASPF delivers up to **25–35% improvements across key metrics**. These outcomes demonstrate the framework's suitability for dynamic, resource-constrained smart healthcare environments, and establish its potential as a robust, QoS-aware orchestration layer for next-generation IoT deployments.

6. Conclusion

This paper presented ML-RASPF, a machine learning-driven, rate-adaptive service provisioning framework for heterogeneous IoT services in smart healthcare systems. Built on a modular mist–edge–cloud architecture, ML-RASPF orchestrates service delivery through context-aware resource allocation, delay-aware utility modeling, and predictive control. We developed a convex model supported by adaptive pricing, dynamic weight adjustment, and RL–based rate control by formulating service provisioning as a joint optimization problem that considers both latency and service rate objectives. The integration of LSTM-based traffic prediction and RL agents enables service-delivery rate adaptation under varying load conditions and service requirements. Comprehensive evaluation using an extended EdgeCloudSim environment demonstrates that ML-RASPF consistently outperforms state-of-the-art baselines, including heuristic, queue-based, and energy-aware techniques—across five critical performance metrics: latency, service delivery rate, energy consumption, bandwidth utilization, and load balancing efficiency. These findings confirm the potential of ML-RASPF to deliver intelligent, scalable and QoS-compliant service provisioning in complex, latency-sensitive smart healthcare ecosystems.

In future work, we plan to extend ML-RASPF with proactive service migration capabilities to handle edge node failures and mobility. Additionally, we aim to integrate advanced multi-agent RL strategies for cooperative edge-to-edge task distribution and adaptive fault-tolerant orchestration under real-time constraints and dynamic network topologies.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Framingham, Mass. The Growth in Connected IoT Devices Is Expected to Generate 79.4ZB of Data in 2025, According to a New IDC Forecast. https://www.idc.com/getdoc. Accessed on April 16, 2025.
2. Sun, M.; Quan, S.; Wang, X.; Huang, Z. Latency-aware scheduling for data-oriented service requests in collaborative IoT-edge-cloud networks. *Future Generation Computer Systems* **2025**, *163*, 107538.
3. Banitalebi Dehkordi, A. EDBLSD-IIoT: a comprehensive hybrid architecture for enhanced data security, reduced latency, and optimized energy in industrial IoT networks. *The Journal of Supercomputing* **2025**, *81*, 359.
4. Khan, S.; Khan, S. Latency aware graph-based microservice placement in the edge-cloud continuum. *Cluster Computing* **2025**, *28*, 88.
5. Pervez, F.; Zhao, L. Efficient Queue-Aware Communication and Computation Optimization for a MEC-Assisted Satellite-Aerial-Terrestrial Network. *IEEE Internet of Things Journal* **2025**.
6. Tripathy, S.S.; Bebortta, S.; Mohammed, M.A.; Nedoma, J.; Martinek, R.; Marhoon, H.A. An SDN-enabled fog computing framework for wban applications in the healthcare sector. *Internet of Things* **2024**, *26*, 101150.
7. Wen, Z.; Yang, R.; Qian, B.; Xuan, Y.; Lu, L.; Wang, Z.; Peng, H.; Xu, J.; Zomaya, A.Y.; Ranjan, R. JANUS: Latency-aware traffic scheduling for IoT data streaming in edge environments. *IEEE Transactions on Services Computing* **2023**, *16*, 4302–4316.
8. Mahapatra, A.; Majhi, S.K.; Mishra, K.; Pradhan, R.; Rao, D.C.; Panda, S.K. An energy-aware task offloading and load balancing for latency-sensitive IoT applications in the Fog-Cloud continuum. *IEEE Access* **2024**.

9. San José, S.G.; Marquès, J.M.; Panadero, J.; Calvet, L. NARA: Network-Aware Resource Allocation mechanism for minimizing quality-of-service impact while dealing with energy consumption in volunteer networks. *Future Generation Computer Systems* **2025**, *164*, 107593.
10. Du, A.; Jia, J.; Chen, J.; Wang, X.; Huang, M. Online Queue-Aware Service Migration and Resource Allocation in Mobile Edge Computing. *IEEE Transactions on Vehicular Technology* **2025**.
11. Amzil, A.; Abid, M.; Hanini, M.; Zaaloul, A.; El Kafhali, S. Stochastic analysis of fog computing and machine learning for scalable low-latency healthcare monitoring. *Cluster Computing* **2024**, *27*, 6097–6117.
12. Centofanti, C.; Tiberti, W.; Marotta, A.; Graziosi, F.; Cassioli, D. Taming latency at the edge: A user-aware service placement approach. *Computer Networks* **2024**, *247*, 110444.
13. Li, Y.; Zhang, Q.; Yao, H.; Gao, R.; Xin, X.; Guizani, M. Next-Gen Service Function Chain Deployment: Combining Multi-Objective Optimization with AI Large Language Models. *IEEE Network* **2025**.
14. Ahmed, W.; Iqbal, W.; Hassan, A.; Ahmad, A.; Ullah, F.; Srivastava, G. Elevating e-health excellence with IOTA distributed ledger technology: Sustaining data integrity in next-gen fog-driven systems. *Future Generation Computer Systems* **2025**, p. 107755.
15. Fei, Y.; Fang, H.; Yan, Z.; Qi, L.; Bilal, M.; Li, Y.; Xu, X.; Zhou, X. Privacy-Aware Edge Computation Offloading With Federated Learning in Healthcare Consumer Electronics System. *IEEE Transactions on Consumer Electronics* **2025**.
16. Ali, A.; Arafa, A. Delay sensitive hierarchical federated learning with stochastic local updates. *IEEE Transactions on Cognitive Communications and Networking* **2025**.
17. Liu, Z.; Xu, X. Latency-aware service migration with decision theory for Internet of Vehicles in mobile edge computing. *Wireless Networks* **2024**, *30*, 4261–4273.
18. Ji, X.; Gong, F.; Wang, N.; Xu, J.; Yan, X. Cloud-Edge Collaborative Service Architecture With Large-Tiny Models Based on Deep Reinforcement Learning. *IEEE Transactions on Cloud Computing* **2025**.
19. Najim, A.H.; Al-sharhanee, K.A.M.; Al-Joboury, I.M.; Kanellopoulos, D.; Sharma, V.K.; Hassan, M.Y.; Issa, W.; Abbas, F.H.; Abbas, A.H. An IoT healthcare system with deep learning functionality for patient monitoring. *International Journal of Communication Systems* **2025**, *38*, e6020.
20. Shang, L.; Zhang, Y.; Deng, Y.; Wang, D. MultiTec: A Data-Driven Multimodal Short Video Detection Framework for Healthcare Misinformation on TikTok. *IEEE Transactions on Big Data* **2025**.
21. EdgeCloudSim. https://github.com/CagataySonmez/EdgeCloudSim. Accessed on April 16, 2025.