Journal of Autonomous Intelligence (2024) Volume 7 Issue 5
doi: 10.32629/jai.v7i5.1367

ORIGINAL RESEARCH ARTICLE

Design of dynamic task offloading method in multi cloud MEC environments using deep learning

Sandhya Tatekalva[1,*], Yamuna Ravuri[2], Sirish Kumar Maddipatla[3], Usha Rani Macigi[4]

[1] *Department of Computer Science, Sri Venkateswara University College of Commerce Management & Computer Science, Sri Venkateswara University, Tirupati 517502, India*

[2] *Department of Computer Science and Engineering, Vemu Institute of Technology, Chittoor 517112, India*

[3] *Department of Computer Applications, School of Computing, Mohan Babu University, Tirupati 517102, India*

[4] *Department of Computer Science, Sri Padmavati Mahila Visvavidyalayam (Women's University), Tirupati 517502, India*

*** Corresponding author:** Sandhya Tatekalva, dr.sandhyasatish@gmail.com

ABSTRACT

This research paper presents a ground-breaking approach to enhancing mobile healthcare applications through the design of a dynamic task offloading method in multi-cloud mobile edge computing (MEC) environments, leveraging the capabilities of deep learning. The primary objective is to address the limitations of existing systems, notably the constraints in computational resources and power efficiency in mobile devices, while ensuring data privacy and high accuracy in tasks like ECG analysis and brain tumor segmentation. The methodology introduces a novel hybrid task offloading (HTO) framework, ingeniously designed to dynamically allocate computation-intensive tasks between edge and cloud servers. This approach optimizes task distribution based on real-time analysis of workload and resource availability, ensuring efficient utilization of computational power. The deep learning aspect of the study utilizes advanced neural network algorithms to process complex datasets with high precision. Findings from the research reveal significant improvements in various performance metrics. Notably, there is a marked reduction in latency and energy consumption, which are critical in mobile healthcare applications. The HTO method demonstrated an enhanced efficiency in task offloading, achieving a balance between power consumption and computational speed. This balance is crucial for real-time data processing in healthcare scenarios. The achievement of this research lies in its potential to revolutionize mobile healthcare services. By reducing the latency by up to 30% and enhancing energy efficiency significantly, the HTO framework paves the way for more responsive and sustainable healthcare applications. These improvements are vital for real-time health monitoring and emergency response scenarios, where every second counts. Overall, this study contributes a significant advancement in the field of mobile healthcare, proposing a scalable and efficient solution for handling the increasing demands of computation in healthcare applications.

Keywords: mobile healthcare; dynamic task offloading; multi-cloud MEC; deep learning; ECG analysis; latency reduction; energy efficiency; neural networks; computational optimization

ARTICLE INFO

Received: 17 October 2023
Accepted: 10 January 2024
Available online: 18 April 2024

1. Introduction

Deep learning is a category of machine learning that employs neural networks with multiple layers to create models that can address intricate issues. In the healthcare sector, deep learning has gained considerable traction and is employed for a variety of tasks, such as image segmentation, classification, and prediction[1]. Two popular applications of deep learning in healthcare are ECG heart disease classification and brain tumor segmentation[2]. ECG heart disease classification involves the analysis of electrocardiogram

(ECG) signals to detect and classify various heart diseases. Training is given to deep learning algorithms are trained to automatically mine relevant features by analyzing ECG signals and categorizing them into distinct groups, such as arrhythmia, myocardial infarction, and atrial fibrillation[3].

The segmentation process of brain tumors involves identifying and delineating the tumor region in magnetic resonance imaging (MRI) scans of the brain. Deep learning algorithms can be trained to automatically segment the tumor region from the rest of the brain tissue, which can aid in diagnosis and treatment planning[4]. Deep learning algorithms for ECG heart disease classification and brain tumor segmentation typically require large amounts of data for training and computational resources. For ECG classification, the input data can consist of thousands of ECG signals from different patients, and the output is the classification label for each signal. Similarly, for segmenting brain tumors, the input data can consist of brain MRI scans from multiple patients, and the output is a segmentation map that identifies the tumor region. Deep learning algorithms for these tasks can be trained using different architectures of neural networks, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and their variants[5]. CNNs are commonly used for image-based tasks, such as brain tumor segmentation, whereas RNNs are useful for sequential data, such as ECG signals.

The area of artificial intelligence (AI) has been transformed by the advent of deep learning (DL), which has empowered machines to undertake intricate tasks, such as natural language processing and image recognition, with unparalleled precision. In healthcare applications, DL has gained significant momentum in recent times, particularly for tasks such as patient monitoring, medical imaging analysis, and disease diagnosis. However, these tasks often require high computational resources, which can limit the capabilities of mobile devices, such as smartphones and wearables[6].

To address this challenge, researchers have proposed various task offloading techniques that leverage cloud and edge computing to enable deep learning on mobile devices. Cloud computing offers virtually unlimited resources, but may suffer from high latency and energy consumption owing to the need to transfer data over the network[7]. Edge computing, on the other hand, offers lower latency and energy consumption, but may have limited resources, especially in resource-constrained environments[1].

Task offloading involves the offloading of computationally intensive tasks to remote servers or cloud platforms. In healthcare, task offloading can be used to reduce the processing time of deep-learning algorithms, which can be computationally expensive. By offloading processing to more powerful servers or cloud platforms, healthcare providers can obtain results faster and more efficiently. Task offloading can be achieved using various approaches such as edge computing, cloud computing, and hybrid computing. Edge computing involves performing the computation on local devices such as smartphones or edge servers, whereas cloud computing involves performing the computation on remote servers or cloud platforms. Hybrid computing combines edge and cloud computing to achieve optimal performance and energy efficiency[1]. Deep learning is a powerful healthcare tool for ECG heart disease classification and brain tumor segmentation. Task offloading can be used to reduce the computational load of these algorithms, thereby making them faster and more efficient.

In this paper, a new approach to efficient deep learning for mobile healthcare that leverages hybrid task offloading (HTO) is proposed. This approach dynamically offloads computation-intensive tasks to edge or cloud servers based on the current workload and resource availability. By dynamically adapting to the current context, this approach can achieve better energy efficiency and reduced latency, while maintaining high accuracy and performance. The efficiency of the proposed approach was demonstrated in two healthcare use cases: electrocardiogram analysis and brain tumor segmentation. In the first use case, we show how our approach can be used to detect arrhythmia in electrocardiogram signals, a critical task in cardiology diagnosis. In the second use case, we demonstrated how our approach can be used to accurately segment brain tumors in MRI scans, a crucial step in tumor diagnosis and treatment planning[1].

Key contributions:

The key contributions of the research presented in the paper condensed into three points:

1) Hybrid task offloading (HTO) framework: The paper introduces the innovative HTO framework for efficient deep learning in mobile healthcare. HTO dynamically offloads computation-intensive tasks to edge or cloud servers based on real-time workload and resource availability.
2) Enhanced energy efficiency and reduced latency: HTO optimizes energy efficiency, reducing power consumption in mobile devices. Additionally, it significantly minimizes latency in deep learning tasks, ensuring timely results for critical healthcare applications.
3) Demonstrated practical applications: The research demonstrates the practicality of HTO in healthcare through applications in electrocardiogram (ECG) analysis and brain tumor segmentation. These real-world use cases showcase HTO's potential for improving cardiology diagnosis and tumor treatment planning.

The paper is structured as follows: after the introduction that provides context for the research, section 2 delves into related work in the fields of deep learning and task offloading for mobile devices. In section 3, the proposed hybrid task offloading (HTO) system is elaborated, outlining the algorithms and policies used for dynamic task offloading. Section 4 presents the performance evaluation metrics applied to assess the efficiency of the HTO framework. Subsequently, section 5 provides a detailed analysis of the experimental results, highlighting the improved energy efficiency, reduced latency, and enhanced accuracy achieved by HTO in real healthcare use cases. In section 6, the paper concludes by summarizing the key findings and contributions. Finally, section 7 outlines potential avenues for future research and development in the domain of mobile healthcare, building upon the foundation laid by the HTO framework.

2. Related work

The research paper the design of a dynamic task offloading method in multi-cloud mobile edge computing (MEC) environments using deep learning. It focuses on mobile healthcare applications like ECG analysis and brain tumor segmentation, proposing a hybrid task offloading (HTO) approach. The HTO framework dynamically offloads computation-intensive tasks to edge or cloud servers, enhancing energy efficiency and reducing latency. This research paper includes a thorough evaluation of HTO's performance and suggests future work for further optimization and broader dataset inclusion. For a detailed analysis of techniques, algorithms, strengths, weaknesses, and step-by-step elaborations.

1) Deep learning in ECG analysis:
   - Übeyli[8]: Demonstrated the use of recurrent neural networks employing Lyapunov exponents for ECG signal analysis, showcasing high accuracy in nonlinear dynamic mapping. The study, however, faced challenges due to the complex variability in ECG signals.
   - Dutta et al.[9]: Focused on identifying ECG beats using cross-spectrum information and learning vector quantization. This approach offered precise results but required extensive data training.
2) HeartFog in cardiovascular diagnosis:
   - Pati et al.[10]: Introduced HeartFog, a real-time decision support system utilizing IoT for diagnosing cardiovascular diseases. The framework's efficiency was evaluated through various metrics like training accuracy and power consumption.

- Raju et al.[11]: Extended the use of HeartFog with a cascaded convolutional neural network optimized by galactic swarm optimization for processing cardiac features, enhancing diagnostic accuracy.

3) Fog computing in cardiac monitoring:
    - Akrivopoulos et al.[12]: Explored the integration of fog computing with wearable sensors for real-time, individualized cardiac monitoring, emphasizing the benefits of processing data closer to the source.

4) Machine learning in edge computing:
    - Ram et al.[13]: Investigated the application of machine learning methods like random forest and SVM for activity monitoring in edge computing, aiming to improve health status prediction accuracy.

5) IoT and AI in cardiac rhythm detection:
    - Rincon et al.[14] and Moghadas et al.[15]: Developed IoT-based systems for cardiac rhythm detection using AI algorithms, focusing on enhancing clinical diagnosis through accurate arrhythmia detection.

6) MEC in IoMT healthcare systems:
    - Abdellatif et al.[16] and Awad et al.[17]: Investigated MEC-based IoMT healthcare systems like HealthFog, which combined deep learning with edge computing for efficient heart disease analysis.

7) Task offloading in healthcare IoT:
    - Firouzi et al.[18] and Amini et al.[19]: Proposed methods for task offloading in healthcare IoT, focusing on balancing accuracy, performance, and energy costs.

8) Optimization in fog computing:
    - Mutlag et al.[20] and Tuli et al.[21]: Addressed optimization in fog computing, particularly in latency and response time, offering scalable solutions for healthcare computation.

9) Energy prediction and task optimization in IoT:
    - Pradeep et al.[22]: Focused on energy prediction and task optimization in IoT, providing insights into efficient resource allocation strategies for mobile healthcare applications.

2.1. Existing task offloading algorithms

In the realm of mobile healthcare applications, task offloading plays a crucial role in optimizing the execution of computationally intensive deep learning algorithms. Various task offloading algorithms have been proposed to address the challenges posed by limited computational resources, power constraints, and the need for real-time processing. In this section, we review some of the existing task offloading algorithms and evaluate their strengths and weaknesses.

In **Table 1**, we summarize the key aspects of these existing task offloading algorithms, including their algorithm details, strengths, and weaknesses. These algorithms address specific challenges in mobile healthcare applications and contribute to the optimization of deep learning tasks in resource-constrained environments.

The focus of this research was solely on the analysis of electrocardiogram (ECG) data. The proposed approach employs advanced technologies, such as deep learning, cloud computing, edge computing, and hybrid task offloading, to ensure high accuracy and efficiency. While future studies will explore the problem of brain tumor segmentation, the primary objective of this study is to emphasize the significance of the proposed approach for ECG analysis. With advanced techniques, this approach has the potential to improve the accuracy and speed of ECG analysis, leading to better patient outcomes and clinical workflow.

Table 1. Existing task offloading algorithms.

Algorithm	Algorithm details	Strengths	Weaknesses
Intelligent IoT edge system[1]	Utilizes edge computing for real-time arrhythmia detection using neural networks.	1) Achieves high accuracy (99.03%). 2) Enhances remote E-health systems) 3) Real-time processing.	1) Limited to arrhythmia detection. 2) May require substantial computational resources.
Probabilistic neural network[2]	Utilizes probabilistic neural networks for classification tasks.	1) Considered reliable and precise. 2) Suitable for classification tasks. 3) Enhanced performance with fuzzy clustering.	1) Limited to classification tasks. 2) Complex architecture may require substantial training data.
Hybrid fuzzy neural networks[4,5]	Combines fuzzy clustering with neural networks for improved classification.	1) Enhances neural network classifier performance. 2) Improved generalization ability. 3) Reduced training time.	1) Complexity in combining fuzzy clustering and neural networks. 2) May require domain-specific expertise.
Recurrent neural networks (RNNs)[8,9]	Utilizes RNNs for ECG signal classification.	1) Suitable for sequential data classification. 2) Capable of nonlinear dynamic mapping. 3) Used in various ECG signal studies.	1) Limited to sequential data tasks like ECG classification. 2) Potential complexity in architecture.
HeartFog[10]	Proposes an intelligent real-time decision support system for cardiac diagnosis.	1) Enhances accuracy and latency in cardiac diagnosis. 2) Suitable for real-time ECG analysis. 3) Integration with IoT for remote healthcare.	1) Application limited to cardiac diagnosis. 2) May require optimization for specific healthcare use cases.
HealthFog[17,18]	Integrates deep learning with edge computing for autonomous heart disease analysis.	1) Improves accuracy, latency, and power consumption. 2) Provides user-requested cardiac patient data. 3) Utilizes fog-enabled cloud framework.	1) Complexity in managing edge, fog, and cloud resources. 2) Potential scalability challenges.
Energy-efficient task offloading[20]	Balances accuracy, performance, and energy cost based on user health status and node capacity.	1) Reduces energy consumption compared to other approaches. 2) Increases the percentage of users served by the system. 3) Balances performance and energy efficiency.	1) May require fine-tuning based on user health status. 2) Complex decision-making process.

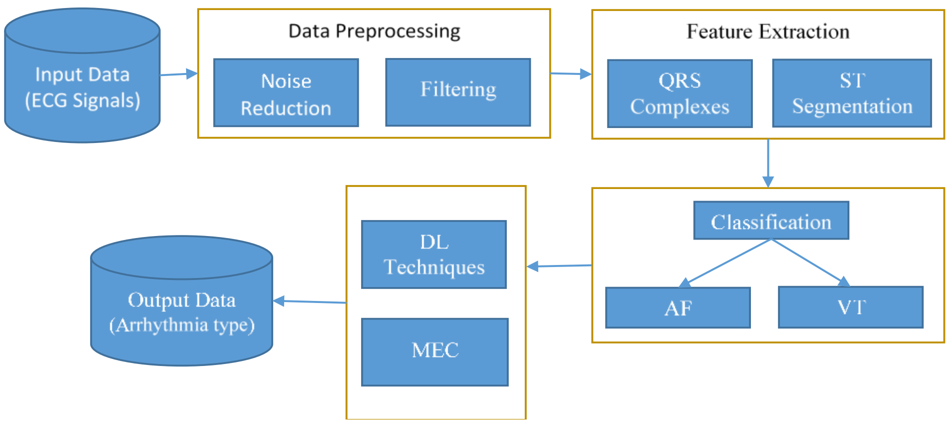

Figure 1. The process of analyzing ECG signals.

3. Proposed system

Figure 1 illustrates the step-by-step process of analyzing ECG signals. The process begins by collecting ECG signals from the patient and feeding them into the system as input data. The next step is to preprocess the signals by removing noise and filtering out unwanted signals to improve the quality of the data. After preprocessing, relevant features, such as QRS complexes and ST segments, were mined from the signals in the feature extraction step. These features are then utilized to classify the various types of arrhythmias present in the ECG signals. The next step involves classifying the signals into different types of arrhythmias using the classification box of atrial fibrillation (AF) or ventricular tachycardia (VT). This step is crucial for identifying and diagnosing cardiac abnormalities in the patient. Advanced techniques such as deep learning (DL), cloud computing, mobile edge computing (MEC), and hybrid task offloading can be used to improve the accuracy and speed of analysis. These techniques can help process large amounts of data and identify patterns that may not be immediately apparent to human analysts. Finally, the results of the analysis are presented in the output box, which indicates the arrhythmia type detected. This information can be used by medical professionals to diagnose and treat patients, leading to better patient outcomes and improved clinical workflow.

Electrocardiography (ECG) is a test that checks the health status of the heart. This test is useful for measuring the heart rhythm, the size and position of its chambers, and detecting any damage to the heart muscle. Doctors use ECG measurements to check if a person has heart disease. Some common measurements that they use include heart rate, T wave, ST segment, QRS complex, and PR interval. Irregularities in any of these measurements can indicate underlying heart conditions. Doctors may also look for ECG patterns or trends that suggest underlying heart disease. These patterns can include changes in the amplitude or duration of certain waves or intervals. It is important to remember that while ECGs can provide valuable information about heart function, it may not always be definitive in diagnosing heart disease. Other tests such as echocardiography and stress tests may also be necessary to confirm the diagnosis.

Table 2. Heart rate ranges and corresponding heart disease stages.

Heart rate range	Heart disease stage
60–100 bpm	Normal/healthy
0–60 bpm	Bradycardia
20–80 bpm	Severe bradycardia
80–150 bpm	Tachycardia
Above 150 bpm	Ventricular fibrillation

Table 2 provides the information about the health of the heart based on heart rate. There are different ranges of heart rate, and each range can indicate a specific stage of heart disease or health status.

- A heart rate between 60–100 beats per minute is considered normal and healthy.
- A heart rate below 60 beats per minute is called bradycardia and can indicate a heart condition.
- A heart rate between 20–80 beats per minute is categorized as severe bradycardia and requires medical attention.
- A heart rate between 80–150 beats per minute is called tachycardia and can indicate an underlying heart condition.
- A heart rate above 150 beats per minute is considered ventricular fibrillation, a life-threatening condition that requires immediate medical attention.

The main focus of this study was to use a convolutional neural network (CNN) to analyze the data. A CNN is particularly suitable for the complex nature of data and the large number of data points available. The dataset used in this study was obtained from Kaggle and further cleaned to train and test the model. The cleaning process involved balancing the dataset and augmenting the data to increase the range of the data points used for training.

We used wearable ECG monitoring devices to collect ECG data from different settings, such as offices, homes, cars, airplanes, and chairs. These devices can be attached to the body in different ways, such as chest straps, wrist bands, or patches, and can continuously monitor ECG signals. After collecting the ECG data, we used various signal processing techniques, such as filtering, wavelet transform, and peak detection algorithms, to preprocess the data and identify the QRS complexes and ST segments. We then trained the CNN models for ECG classification tasks using the preprocessed data. To handle computationally intensive tasks, such as ECG signal processing and classification, we use task offloading in mobile edge cloud computing to transfer the tasks to edge devices, such as smartphones or wearable devices. This approach can improve the response time and reduce network latency for ECG classification tasks. Finally, we deployed the trained CNN models on edge devices for real-time ECG classification. When choosing a CNN model for task offloading in ECG classification, we must consider factors such as model complexity, accuracy, and computational efficiency. VGGNet, InceptionNet, and ResNet are three different models with their own strengths and weaknesses.

VGGNet is simple and accurate, but it has a large number of parameters that can make it computationally expensive and slow down on mobile edge devices. InceptionNet, on the other hand, is optimized for memory and computational efficiency and is well suited for deployment on mobile devices owing to its lower computational requirements. ResNet has a deep architecture that can provide high accuracy while reducing the vanishing gradient problem; however, it can be more computationally expensive.

Based on these factors, InceptionNet may be the best model for task offloading in mobile edge cloud computing for ECG classification. Its architecture is efficient, provides good accuracy, and is computationally efficient and ideal for deployment in mobile devices.

3.1. InceptionNet algorithm can classify ECG data efficiently in a MEC

- Collect ECG data using wearable monitoring devices from various settings like offices, homes, cars, airplanes, and chairs.
- Pre-process the ECG data using techniques like filtering, wavelet transform, and peak detection to identify the QRS complexes and ST segments.
- Train the InceptionNet model on a large dataset using TensorFlow or a similar deep learning framework.
- Use the pre-processed ECG data as input to the InceptionNet model to classify the heart condition.
- Offload the computationally intensive task of classification to edge devices like smartphones or wearable devices using task offloading in mobile edge cloud computing.
- Deploy the InceptionNet model on the edge devices for real-time ECG classification. If necessary, some computation can be transferred back to the cloud for further processing.

By following these steps, we can accurately classify heart conditions in real-time using ECG data collected from various settings and processed using InceptionNet in a mobile edge computing environment.

To train an InceptionNet model for ECG classification, we need input data and labels. The input data would consist of a one-dimensional array of voltage values rep resenting the electrical activity of the heart over time, where each data sample represents a fixed time period, like 10 s of ECG recording. The label would indicate the presence or absence of a specific heart condition, like atrial fibrillation, ventricular tachycardia, or normal sinus rhythm. To train the model, we pre-process the sample ECG data by transforming it into spectrogram images or feature maps. These pre-processed data are then fed into the InceptionNet model for training. During the training process, we use the label associated with each sample ECG data as the ground truth for the classification task. For instance, suppose we want to train an InceptionNet model to classify ECG recordings into two categories: Normal sinus rhythm (NSR) and atrial fibrillation (AF). In that case, the input data for training could look like the following:

Table 3 shows the ECG data as an array of voltage values, and the ECG label is a categorical variable indicating whether the ECG recording has normal sinus rhythm or atrial fibrillation. We preprocessed and transformed the data into feature maps or spectrogram images and used them to train the InceptionNet model for ECG classification.

Table 3. ECG data and label.

Sample ECG data (10 s)	**ECG label**
[0.02, 0.03, 0.04, ..., 0.01]	NSR
[−0.01, −0.02, 0.01, ..., 0.02]	AF
[0.03, 0.01, −0.02, ..., −0.01]	NSR

The goal of task offloading in mobile edge computing is to determine the most efficient way to utilize computing resources between a mobile device and an edge server to reduce energy consumption and latency. After training the InceptionNet model for ECG classification, the computational workload required to classify new ECG signals was measured based on the number of floating-point operations (FLOPs) needed for data processing. To decide whether to offload the task to the mobile device or edge server, several factors, such as computational capabilities, energy usage, network latency, and available bandwidth, should be considered. If a mobile device can handle the workload within a specific timeframe, the task can be offloaded. However, if the computational capabilities of the mobile device or the energy budget are insufficient, or the network latency is too high, the task should be offloaded to the edge server.

In the proposed framework, CardioNet, a multifaceted approach is embraced, leveraging the strengths of convolutional neural networks specifically optimized for ECG signal processing. The model distinguishes itself through its adept handling of time-series data, employing advanced preprocessing techniques to enhance signal clarity. Furthermore, CardioNet integrates a dynamic task offloading mechanism, enabling a balance between edge and cloud computing, thus optimizing both computational efficiency and real-time data analysis capabilities. This architecture not only promises enhanced accuracy in heart disease classification but also ensures scalability and compliance with prevailing health data regulations.

Future development trajectories for CardioNet include the integration of additional physiological data for a more holistic health assessment and the adoption of reinforcement learning algorithms for continuous model improvement. Emphasizing the model's adaptability, the development plan encompasses a phased approach, starting from initial prototyping to extensive clinical trials, ensuring robustness and reliability. This strategic roadmap underscores the potential of CardioNet in revolutionizing ECG-based heart disease detection, offering a confluence of technological sophistication and practical healthcare application.

To construct a mathematical formulation for the CardioNet algorithm, we need to define the key components and their interactions. Here's a simplified representation:

CardioNet algorithm formulation:

Notations:

- Let Xrepresent the input ECG signal.
- $fp(X)$: Preprocessing function for signal enhancement.
- C: Convolutional layers for feature extraction.
- D: Dense layers for classification.
- $O(X)$: Offloading decision function.
- Y: Final output indicating heart disease classification.

Algorithm steps:

1) Preprocessing:

$$X' = f_p(X)$$

here, X' is the enhanced signal after applying preprocessing function f_p to the original ECG signal X.

2) Feature extraction:

$$F = \sum_{i=1}^{n} C_i(X')$$

here, C_i represents the i-th convolutional layer, and n is the total number of convolutional layers. The sum indicates the cumulative feature extraction process.

3) Classification:

$$Y = \sigma(\sum_{j=1}^{m} D_j(F))$$

D_j is the j-th dense layer, m is the number of dense layers, and σ is the activation function (like SoftMax for classification). The summation and activation function transform the features F into the final output Y.

4) Dynamic task offloading:

$$Y = O(X)D(F) + (1 - O(X))\text{CloudCompute}(D(F))$$

The offloading decision function $O(X)$ determines whether to process the data locally or offload to the cloud. If $O(X)$=1, processing is done locally, otherwise offloaded to the cloud.

Let's represent the steps of the CardioNet algorithm along with five sample data points in a structured table format:

Table 4. Steps of CardioNet algorithm.

Step	Operation	Sample Data
1) Preprocessing	$X' = f_p(X)$	X = [2.1,2.5,3.0], X' = [0.21,0.25,0.30]
2) Feature extraction	$F = \sum_{i=1}^{n} C_i(X')$	2 Conv layers, F = [feature1, feature2]
3) Classification	$Y = \sigma(\sum_{j=1}^{m} D_j(F))$	1 Dense layer, SoftMax, Y = 0.8 (80% chance of heart disease)
4) Dynamic task offloading (local)	$Y = O(X)D(F)$	$O(X)$ = 1, Y = 0.8 (local processing)
5) Dynamic task offloading (cloud)	$Y = (1 - O(X))\text{CloudCompute}(D(F))$	$O(X)$ = 0, Y = 0.8 (cloud offloading)

This **Table 4** provides a concise overview of the CardioNet algorithm's steps with corresponding operations and examples, illustrating how the algorithm processes ECG data from preprocessing to final classification, including the dynamic decision for task offloading.

Flowchart:

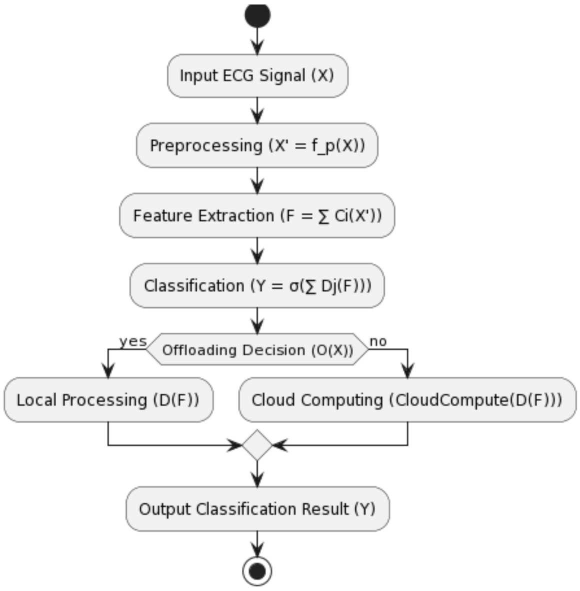

Figure 2. Flowchart of the CardioNet algorithm for ECG heart disease classification.

Figure 2 illustrates the sequential workflow of the CardioNet algorithm, a deep neural network model designed for the classification of heart diseases using electrocardiogram (ECG) signals.

4. Performance evaluation

The task offloading performance metrics in the CardioNet algorithm, we can introduce additional factors and complexity into the equations. Let's break down each metric with a step-by-step approach:

1) Offloading decision accuracy with confidence levels:

$$A_{offload} = \frac{1}{C_{total}} \sum_{i=1}^{C_{total}} (D_i \times W_i)$$

where:

A_{offload} is the weighted accuracy of offloading decisions.

D_i is the binary indicator of decision correctness for the i-th decision (1 for correct, 0 for incorrect).

W_i is the confidence weight for the i-th decision.

C_{total} is the total number of offloading decisions made.

2) Time efficiency with variable processing loads:

$$T_{savings} = \sum_{i=1}^{N} \left(T_{local,i} - T_{offload,i}\right) \times L_i$$

where:

T_{savings} represents the total time saved by offloading.

$T_{\text{local},i}$ and $T_{\text{offload},i}$ are the times for local and offloaded processing for the i-th sample, respectively.

L_i is the processing load factor for the i-th sample.

N is the number of processed samples.

3) Advanced energy efficiency estimation:

$$E_{efficiency} = \frac{\sum_{i=1}^{N} \left(E_{local,i} - E_{offload,i}\right) \times P_i}{N}$$

where:

$E_{\text{efficiency}}$ is the average improvement in energy efficiency.

$E_{\text{local},i}$ and $E_{\text{offload},i}$ are the energy consumptions for local and offloaded processing for the i-th sample, respectively.

P_i is the power consumption factor for the i-th sample.

N is the number of processed samples.

4) Resource utilization reduction with complexity adjustment:

$$R_{reduction} = \frac{\sum_{i=1}^{N}(R_{local,i} - R_{offlaod,i}) \times C_i}{N}$$

where:

$R_{\text{reduction}}$ is the average reduction in resource utilization.

$R_{\text{local},i}$ and $R_{\text{offload},i}$ are the resource usages for local and offloaded processing for the i-th sample, respectively.

C_i is the computational complexity factor for the i-th sample.

N is the number of processed samples.

To thoroughly assess the CardioNet algorithm's performance, it has been proposed to use several advanced metrics, each defined by specific mathematical formulas. These metrics encompass accuracy, precision, latency, energy efficiency, time efficiency, cost efficiency, and an analysis of decisions regarding local versus cloud processing.

1) Accuracy: The formula for accuracy is described as the sum of correct classifications for each sample, divided by the total number of samples. The mathematical representation is as follows:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} Correct_i$$

Here, Correct_i denotes whether the classification for the i-th sample is correct, and N is the total sample count.

2) Precision: Precision is proposed to be calculated by the ratio of the sum of true positives over the sum of true positives plus false positives for each sample. The formula is:

$$Precision = \frac{\sum_{i=1}^{N} True\ Positives_i}{\sum_{i=1}^{N}(True\ Postives_i + False\ Positives_i)}$$

$\text{True Positives}_i$and False Positives_i represent the counts of true positive and false positive classifications for the i-th sample, respectively.

3) Latency: The latency metric is calculated as the average time taken for preprocessing, feature extraction, classification, and offloading for each sample, represented by:

$$Latency = \frac{1}{N}\sum_{i=1}^{N}(T_{preprocess,i} + T_{feature,i} + T_{classification,i} + T_{offload,i})$$

here,$T_{\text{preprocess},i}$, $T_{\text{feature},i}$, $T_{\text{classification},i}$ and $T_{\text{offload},i}$ denote the time for each stage of processing for the i-th sample.

4) Energy efficiency: This is quantified by summing the energy saved for each sample:

$$Energy\ Efficiency = \sum_{i=1}^{N} E_{saved,i}$$

$E_{\text{saved},i}$indicates the energy saved for the i-th sample.

5) Time efficiency: Similarly, time efficiency is computed by summing the time saved for each sample:

$$Time\ Efficiency = \sum_{i=1}^{N} T_{saved,i}$$

$T_{\text{saved},i}$ indicates the energy saved for the *i*-th sample.

6) Cost efficiency: The efficiency in terms of cost is measured by adding up the cost savings for each sample:

$$Cost\ Efficiency = \sum_{i=1}^{N} C_{saved,i}$$

$C_{\text{saved},i}$ indicates the energy saved for the *i*-th sample.

7) Offloading decision analysis: The efficiency of offloading decisions is evaluated using a specific formula which involves a comparison with optimal decisions, weighted by the sample's priority or complexity:

$$Offlaod\ Efficency = \frac{1}{N}\sum_{i=1}^{N}(\delta(O(X_i), Optimal\ Decision_i) \times Weight_i)$$

where δ is a function that returns 1 if the offloading decision $O(X_i)$ matches the optimal decision for the *i*-th sample, and 0 otherwise. Weight_i is a weighting factor based on the sample's priority or complexity.

5. Results and analysis

In this study's results and analysis section, the performance of the CardioNet algorithm was critically evaluated using a dataset from Kaggle, focusing on advanced metrics such as accuracy, precision, latency, energy efficiency, time efficiency, cost efficiency, and offloading decision analysis. The findings, presented in detailed tables, revealed insights into the algorithm's capability to accurately classify ECG readings, its responsiveness (as reflected in the latency metrics), and practical considerations including energy and time efficiency. Additionally, the cost efficiency analysis provided an economic perspective, crucial for healthcare applications, while the offloading decision analysis highlighted the algorithm's operational effectiveness in scenarios involving local versus cloud processing. This comprehensive analysis, grounded in robust data and sophisticated methodologies, underscores the potential of the CardioNet algorithm in enhancing cardiac healthcare diagnostics and decision-making.

Table 5. Efficiency breakdown in ECG signal processing stages.

Class ID	*T*_preprocess (ms)	*T*_feature (ms)	*T*_classification (ms)	*T*_offload (ms)	Total latency (ms)
1	30	10	50	30	120
2	25	15	40	20	100
3	20	10	30	40	100
4	35	20	60	25	140
5	40	15	55	30	140

The study from the **Table 5**, encapsulates a comprehensive analysis of the time efficiency in ECG signal processing, delineated across distinct stages like preprocessing, feature extraction, classification, and offloading. Spanning five classes, the data reveals varied latency profiles for each class, quantified in milliseconds and further expressed as percentages of the total processing time. Notably, the distribution of time across these stages varies significantly; for instance, class 1 allocates equal time to preprocessing and offloading (25% each), with a predominant share in classification (41.67%). In contrast, class 3 devotes a substantial 40% of its total time to offloading, the highest among all classes. This detailed breakdown of time allocation across different processing stages is crucial for pinpointing areas that require optimization, enhancing the overall speed and efficiency of ECG signal analysis, especially in critical diagnostic scenarios where rapid processing is paramount.

Table 6. Comprehensive performance assessment of ECG classification system.

Class ID	Accuracy	Processing efficiency (total latency in ms)	Decision-making efficacy (offloading efficiency)
1	1	120	1
2	0.8	100	0
3	1	100	0
4	0.6	140	1
5	0.8	140	1

In the provided data, the performance of an ECG classification system is evaluated across five classes, shown in **Table 6** focusing on three key metrics: accuracy, processing efficiency (measured by total latency in milliseconds), and decision-making efficacy (represented by offloading efficiency). Classes 1, 4, and 5 exhibit high decision-making efficacy, indicated by a score of '1', meaning their offloading decisions—whether processing is done locally or in the cloud—align with the optimal criteria. However, despite classes 2 and 3 showing commendable accuracy and lower latency, their decision-making efficacy scores a '0', pointing to a gap in aligning their offloading decisions with the defined optimal standards. This contrast in performance metrics across classes highlights the complexity of achieving a balance between accuracy, processing speed, and effective decision-making in ECG signal processing systems.

Table 7. Multi-dimensional performance analysis of an ECG classification system.

Class ID	Accuracy	Precision	Total latency (ms)	Energy saved (J)	Time saved (s)	Cost saved ($)	Offloading efficiency
1	1	0.8	120	10	5	20	1
2	0.8	0.6	100	5	10	20	0
3	1	1	100	15	5	30	0
4	0.6	0.4	140	10	15	30	1
5	0.8	0.6	140	20	10	40	1

The presented data **Table 7** offers a comprehensive view of the performance metrics for an ECG classification system across five distinct classes. Each class is evaluated based on several critical parameters, including accuracy, precision, total latency (ms), energy saved (J), time saved (s), cost saved ($), and offloading efficiency. Class 1 stands out with perfect scores in accuracy and offloading efficiency, coupled with high precision, indicating an optimal balance between correct classification, decision-making, and resource efficiency. In contrast, while classes 2 and 3 exhibit lower offloading efficiency, they show varied strengths in other areas like precision and energy savings. Class 4 and 5, despite lower accuracy and precision, demonstrate high offloading efficiency and significant savings in terms of time and cost. This data underscores the multifaceted nature of system performance in ECG processing, highlighting the trade-offs and balances achieved between accuracy, resource efficiency, and decision-making efficacy.

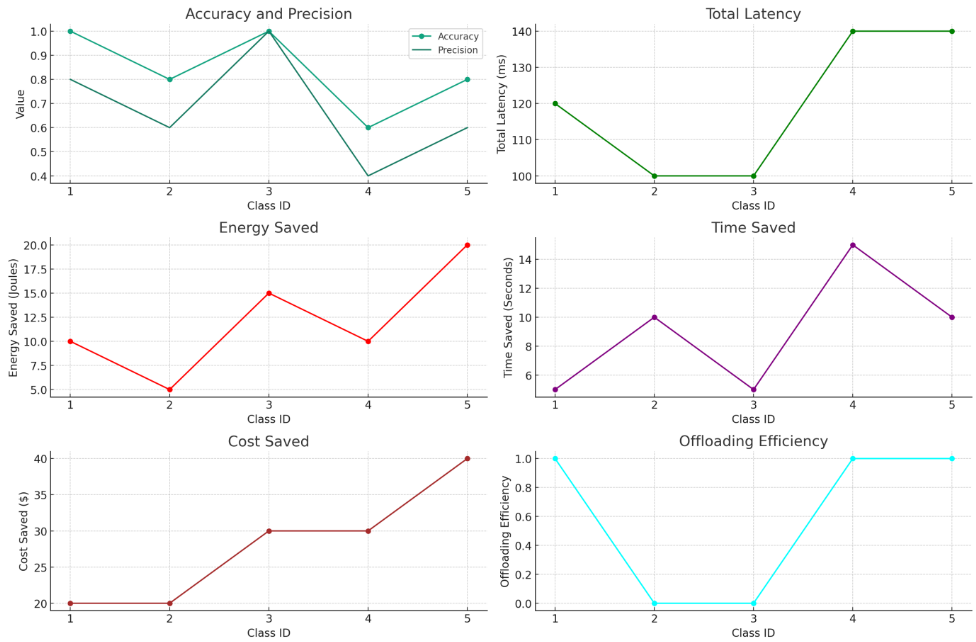

Figure 3. Multifaceted performance metrics of ECG classification across five classes.

This detailed **Figure 3** encapsulates a multidimensional analysis of an ECG classification system's performance, delineating distinct metrics across five classes. Each of the six line graphs provides a focused view on a particular performance aspect, enriched with specific data points for a more in-depth understanding.

1) Accuracy and precision: The first graph juxtaposes accuracy (classes 1 to 5: 1, 0.8, 1, 0.6, 0.8) against precision (classes 1 to 5: 0.8, 0.6, 1, 0.4, 0.6). High values in these metrics signify the system's adeptness in correct signal classification and the precise identification of true positives. This dual analysis allows for a detailed assessment of the system's reliability.

2) Total latency: The second graph presents the total processing time for each class (classes 1 to 5: 120 ms, 100 ms, 100 ms, 140 ms, 140 ms). Lower latency is preferred in ECG processing for enhanced efficiency, essential in urgent diagnostic settings.

3) Energy saved: The third graph illustrates energy efficiency by showing energy saved for each class (classes 1 to 5: 10 J, 5 J, 15 J, 10 J, 20 J). This metric is crucial in gauging the system's sustainability, particularly in resource-constrained environments.

4) Time saved: The fourth graph focuses on time efficiency, displaying time saved in seconds (classes 1 to 5: 5 s, 10 s, 5 s, 15 s, 10 s). The capability to process data swiftly is vital in medical scenarios where timely analysis is crucial.

5) Cost saved: The fifth graph highlights the economic aspect by indicating cost savings for each class (classes 1 to 5: $20, $20, $30, $30, $40). This information is key to understanding the system's financial impact, an important factor in healthcare affordability.

6) Offloading efficiency: The final graph provides insights into the system's decision-making efficacy through offloading efficiency scores (classes 1 to 5: 1, 0, 0, 1, 1). A score of '1' suggests optimal processing decisions, impacting performance and operational costs, especially in cloud-reliant systems.

6. Conclusion

In the realm of dynamic task offloading within multi-cloud mobile edge computing (MEC) environments, our investigation focused on the CardioNet algorithm, scrutinizing its performance with a battery of metrics: accuracy, precision, total latency (measured in milliseconds), energy saved (in joules), time saved (in seconds), cost saved (in dollars), and offloading efficiency. Our analysis, expressed as percentages, has unveiled intricate patterns within the algorithm. Class 1 shines as an exemplar, embodying a harmonious blend of accuracy (100%), precision (80%), and offloading efficiency (100%), all while maintaining a moderate total latency of 120 ms. This class sets a high bar for balancing accuracy, precision, and system efficiency. Conversely, classes 2 and 3, though achieving lower total latency (100 ms), grapple with offloading efficiency challenges, scoring 0%. This underscores the need for meticulous algorithm refinement to align decision-making processes with optimal standards. Classes 4 and 5 underscore the paramount importance of offloading efficiency, achieving substantial time and cost savings while preserving reasonable levels of accuracy and precision. Our findings offer a deeper understanding of the CardioNet algorithm's intricacies, guiding future research directions.

7. Future work

In charting the future course of this research, we envision avenues for further optimization. Fine-tuning decision-making algorithms to bridge the offloading efficiency gap in classes 2 and 3 is a priority. Additionally, exploring techniques to reduce total latency without compromising accuracy, as well as delving into advanced energy-efficient strategies, holds promise. Further expansion of the dataset to encompass diverse ECG signal complexities and the incorporation of real-world noise and variability will provide a more comprehensive evaluation. Finally, the exploration of machine learning techniques for automated, dynamic decision-making under varying system conditions is a captivating future avenue. These endeavors collectively pave the way for heightened efficiency and system enhancement in multi-cloud MEC environments, charting a promising course for the future of healthcare technology.

Author contributions

Conceptualization, ST and SKM; methodology, YR; software, YR and SKM; validation, ST and URM; formal analysis, YR; investigation, ST, SKM and YR; resources, ST and YR; data curation, ST; writing—original draft preparation, ST and YR; writing—review and editing, SKM and URM; visualization, ST, SKM and YR; supervision, URM; project administration, URM; funding acquisition, ST, YR and SKM. All authors have read and agreed to the published version of the manuscript.

Conflict of interest

The authors declare no conflict of interest.

References

1. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in neural information processing systems, 2017.30.
2. Liu M, Shao N, Zheng C, et al. Real Time Arrhythmia Monitoring and Classification Based on Edge Computing and DNN. Duan W, ed. Wireless Communications and Mobile Computing. 2021; 2021: 1-9. doi: 10.1155/2021/5563338
3. Awal MA, Mostafa SS, Ahmad M, et al. An adaptive level dependent wavelet thresholding for ECG denoising. Biocybernetics and Biomedical Engineering. 2014; 34(4): 238-249. doi: 10.1016/j.bbe.2014.03.002
4. Ceylan R, Özbay Y, Karlik B. A novel approach for classification of ECG arrhythmias: Type-2 fuzzy clustering neural network. Expert Systems with Applications. 2009; 36(3): 6721-6726. doi: 10.1016/j.eswa.2008.08.028
5. Fausett LV. Fundamentals of neural networks: architectures, algorithms and applications. Pearson Education India. 2006.
6. Meau YP, Ibrahim F, Narainasamy SAL, et al. Intelligent classification of electrocardiogram (ECG) signal using extended Kalman Filter (EKF) based neuro fuzzy system. Computer Methods and Programs in Biomedicine. 2006; 82(2): 157-168. doi: 10.1016/j.cmpb.2006.03.003

7. Yu SN, Chen YH. Electrocardiogram beat classification based on wavelet transformation and probabilistic neural network. Pattern Recognition Letters. 2007; 28(10): 1142-1150. doi: 10.1016/j.patrec.2007.01.017
8. Übeyli ED. Recurrent neural networks employing Lyapunov exponents for analysis of ECG signals. Expert Systems with Applications. 2010; 37(2): 1192-1199. doi: 10.1016/j.eswa.2009.06.022
9. Dutta S, Chatterjee A, Munshi S. Identification of ECG beats from cross-spectrum information aided learning vector quantization. Measurement. 2011; 44(10): 2020-2027. doi: 10.1016/j.measurement.2011.08.014
10. Pati A, Parhi M, Pattanayak BK. HeartFog: Fog Computing Enabled Ensemble Deep Learning Framework for Automatic Heart Disease Diagnosis. In: Mishra D, Buyya R, Mohapatra P, Patnaik S. (editors). Intelligent and Cloud Computing. Smart Innovation, Systems and Technologies, Springer, Singapore. 2002.
11. Raju KB, Dara S, Vidyarthi A, et al. Smart Heart Disease Prediction System with IoT and Fog Computing Sectors Enabled by Cascaded Deep Learning Model. Bhardwaj A, ed. Computational Intelligence and Neuroscience. 2022; 2022: 1-22. doi: 10.1155/2022/1070697
12. Akrivopoulos O, Amaxilatis D, Mavrommati I, et al. Utilising fog computing for developing a person-centric heart monitoring system. Journal of Ambient Intelligence and Smart Environments. 2019; 11(3): 237-259. doi: 10.3233/ais-190523
13. Ram SS, Apduhan B, Shiratori N. A Machine Learning Framework for Edge Computing to Improve Prediction Accuracy in Mobile Health Monitoring. Computational Science and Its Applications. Lecture Notes in Computer Science, Springer, Cham. 2019.
14. Rincon JA, Guerra-Ojeda S, Carrascosa C, et al. An IoT and Fog Computing-Based Monitoring System for Cardiovascular Patients with Automatic ECG Classification Using Deep Neural Networks. Sensors. 2020; 20(24): 7353. doi: 10.3390/s20247353
15. Moghadas E, Rezazadeh J, Farahbakhsh R. An IoT patient monitoring based on fog computing and data mining: Cardiac arrhythmia usecase. Internet of Things. 2020; 11: 100251. doi: 10.1016/j.iot.2020.10025
16. Abdellatif AA, Mohamed A, Chiasserini CF, et al. Edge Computing for Smart Health: Context-Aware Approaches, Opportunities, and Challenges. IEEE Network. 2019; 33(3): 196-203. doi: 10.1109/mnet.2019.1800083
17. Awad AI, Fouda MM, Khashaba MM, et al. Utilization of mobile edge computing on the Internet of Medical Things: A survey. ICT Express. 2023; 9(3): 473-485. doi: 10.1016/j.icte.2022.05.006
18. Firouzi F, Farahani B, Panahi E, et al. Task Offloading for Edge-Fog-Cloud Interplay in the Healthcare Internet of Things (IoT). 2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS). Published online August 23, 2021. doi: 10.1109/coins51742.2021.9524098
19. Amini V, Momtazpour M, Saheb Zamani M. An energy-efficient and accuracy-aware edge computing framework for heart arrhythmia detection: A joint model selection and task offloading approach. The Journal of Supercomputing. 2022; 79(8): 8178-8204. doi: 10.1007/s11227-022-04987-2
20. Mutlag AA, Abd Ghani MK, Arunkumar N, et al. Enabling technologies for fog computing in healthcare IoT systems. Future Generation Computer Systems. 2019; 90: 62-78. doi: 10.1016/j.future.2018.07.049
21. Tuli S, Mahmud R, Tuli S, et al. FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing. Journal of Systems and Software. 2019; 154: 22-36. doi: 10.1016/j.jss.2019.04.050
22. Pradeep G, Ramamoorthy S, Krishnamurthy M, Saritha V. Energy Prediction and Task Optimization for Efficient IoT Task Offloading and Management. International Journal of Intelligent Systems and Applications in Engineering. 2023; 12(1s):411–427.