

TinyML-Based Embedded Electronic Devices for Intelligent Remote Patient Monitoring

Mohammed Liaqat Ali Khan¹, Dr. Nidhi Mishra², Dr. Abdul Bari²

¹Research Scholar, Kalinga University, Raipur, India.

²Assistant Professor, Department of CSE, Kalinga University, Raipur, India.

³Dean Academics & Prof. CSE Department, ISL Engineering College, Hyderabad, India.

Mail Id: alykhan113@yahoo.co.in¹, ku.nidhimishra@kalingauniversity.ac.in², abdulbarimohammed11@gmail.com³

Article Received 09-08-2026, Revised 31-08-2026, Accepted 28-09-2026

Author(s) Retains the Copyrights of This Article

Abstract

The increasing prevalence of chronic diseases, growing demand for continuous healthcare services, and limitations of conventional hospital-based monitoring systems have created a need for intelligent, low-power, and remotely accessible patient monitoring solutions. Recent advancements in embedded electronics, Internet of Medical Things (IoMT), and Tiny Machine Learning (TinyML) provide opportunities to develop compact healthcare devices capable of performing real-time physiological signal analysis directly on resource-constrained hardware. This research proposes a TinyML-based embedded electronic framework for remote patient monitoring, integrating wearable biomedical sensors, microcontroller-based processing units, lightweight machine learning algorithms, and wireless communication technologies. The proposed framework focuses on monitoring essential physiological parameters, including heart rate, body temperature, blood oxygen saturation (SpO₂), and electrocardiogram (ECG) signals, enabling continuous observation of patient health conditions outside conventional clinical environments. Lightweight machine learning models are designed to identify abnormal physiological patterns through localized inference, reducing dependency on continuous cloud processing and minimizing communication overhead. Model optimization techniques, including quantization, pruning, and efficient feature extraction, are considered to accommodate memory, computational, and energy constraints of embedded devices. Furthermore, the proposed architecture incorporates secure data transmission, threshold-based emergency notifications, and cloud-assisted healthcare visualization to facilitate communication between patients and healthcare professionals. The research emphasizes the evaluation of classification performance, inference latency, memory utilization, power consumption, communication reliability, and system scalability. By integrating intelligent edge processing with embedded biomedical electronics, the proposed framework aims to establish a practical foundation for affordable, energy-efficient, and privacy-conscious remote patient monitoring, particularly in resource-constrained and geographically distributed healthcare environments.

Keywords: Tiny Machine Learning (TinyML); Remote Patient Monitoring; Embedded Electronic Devices; Internet of Medical Things (IoMT); Edge Artificial Intelligence; Biomedical Sensors; ECG Monitoring; SpO₂ Monitoring; Lightweight Deep Learning; Low-Power Computing; Real-Time Health Monitoring; Predictive Healthcare.

1. Introduction

The increasing demand for accessible, continuous, and personalized healthcare has encouraged the development of intelligent monitoring technologies that extend healthcare services beyond conventional hospital environments. Traditional patient monitoring systems generally depend on centralized medical equipment, periodic clinical assessments, and continuous involvement of healthcare professionals. Although these approaches remain essential in clinical practice, they may provide limited opportunities for uninterrupted observation of patients during daily activities, particularly in geographically isolated communities and resource-constrained healthcare settings. The integration of wearable biomedical

sensors, embedded electronics, wireless communication, and artificial intelligence has introduced new possibilities for developing remote patient monitoring systems capable of collecting and processing physiological information in real time. Wearable sensor technologies have demonstrated their potential for continuous monitoring of physiological and activity-related parameters, supporting the development of healthcare applications that operate outside conventional clinical environments [13], [14]. The Internet of Medical Things (IoMT) represents an important technological foundation for connecting biomedical sensing devices, embedded processing units, communication networks, and healthcare information platforms. Through interconnected

medical devices, physiological measurements can be collected, transmitted, stored, and accessed by authorized healthcare professionals. Islam et al. presented a comprehensive investigation of IoT applications in healthcare, highlighting the significance of network architectures, medical applications, security requirements, and privacy considerations. Their study established the broader technological context for connected healthcare systems and identified challenges associated with integrating heterogeneous devices and healthcare services [6]. Similarly, the integration of wearable devices with wireless sensor networks provides opportunities for continuous physiological data acquisition and remote observation, particularly when communication and processing capabilities are distributed across multiple system components [16]. Despite these developments, conventional cloud-dependent monitoring architectures introduce certain limitations when continuous physiological data must be transmitted to remote servers for processing. Network interruptions, communication delays, bandwidth consumption, and privacy concerns can affect the reliability of healthcare applications. These challenges become particularly relevant in environments where patients require uninterrupted monitoring but have limited access to high-speed internet connectivity. Cloud-assisted processing also creates a dependency on external infrastructure for computationally intensive operations. Consequently, distributing selected analytical functions closer to the sensing device has emerged as a potential approach for improving responsiveness and reducing unnecessary communication. Edge computing and wearable machine learning systems provide a foundation for processing physiological information near its source, thereby supporting more localized and resource-efficient healthcare intelligence [17].

Tiny Machine Learning (TinyML) extends this concept by enabling machine learning inference on low-power microcontrollers and resource-constrained embedded platforms. Unlike conventional artificial intelligence systems that depend primarily on powerful computing infrastructure, TinyML focuses on compact models that can operate within restricted memory, processing, and energy budgets. This capability is particularly relevant to wearable healthcare devices, where computational resources and battery capacity directly influence operational duration, device size, and user convenience. David et al. introduced TensorFlow Lite Micro as an embedded machine learning framework designed to support inference on microcontroller-class devices, establishing an implementation pathway for deploying lightweight models in resource-constrained environments [1]. The availability of such frameworks

creates opportunities to integrate intelligent signal analysis into compact biomedical monitoring devices. Recent research has also demonstrated the potential of lightweight machine learning for continuous cardiac monitoring. Mukhopadhyay et al. investigated automated generation of compact models for real-time ECG classification on tiny edge devices, addressing the need for efficient cardiac signal analysis under embedded computational constraints [4]. In a broader clinical context, Hannun et al. demonstrated the application of deep neural networks to ambulatory electrocardiogram classification, illustrating the potential of machine learning to analyze extended ECG recordings. However, transferring computationally intensive models to low-power wearable hardware requires additional optimization, implementation, and validation [27]. These developments highlight the importance of balancing analytical capability with the practical limitations of embedded medical electronics.

The proposed research focuses on the development of a TinyML-based embedded electronic framework for intelligent remote patient monitoring. The system integrates ECG, photoplethysmography (PPG), blood oxygen saturation, heart rate, and body temperature sensing with a low-power microcontroller capable of executing lightweight machine learning models locally. Physiological signals are acquired through wearable or connected biomedical sensors and subsequently processed through filtering, segmentation, normalization, and feature extraction stages. The optimized model performs localized inference to identify predefined abnormal physiological patterns and generates corresponding notifications. Relevant measurements and event information are transmitted through wireless communication technologies to a remote monitoring platform, where authorized healthcare professionals can examine patient information.

The research further investigates model optimization techniques, including quantization, pruning, and compact feature representation, to reduce computational requirements while preserving classification performance. The system is designed to examine multiple engineering dimensions, including inference latency, memory consumption, power utilization, communication reliability, classification performance, and scalability. Particular attention is given to the separation of local inference and cloud-assisted visualization, allowing the embedded device to perform essential analytical operations without requiring continuous cloud connectivity.

2. Literature Review

Islam et al. examined the application of the Internet of Things in healthcare through a comprehensive survey of connected medical devices, network architectures, healthcare applications, and security requirements. Their investigation explained how interconnected sensing technologies can support healthcare delivery through distributed data collection and communication. The study also emphasized privacy and security as important considerations when medical information is exchanged through networked devices. These findings provide a conceptual foundation for designing remote patient monitoring systems that integrate embedded sensing, wireless communication, and centralized healthcare information management. However, the implementation of intelligent processing directly on low-power biomedical devices requires additional consideration of computational and energy limitations [6].

Bonato discussed wearable sensors and their transition from enabling technologies to practical clinical applications. The research highlighted the potential of wearable systems for monitoring individuals in home and community environments, particularly through continuous acquisition of physiological information. The study established the relevance of wearable sensing for applications involving cardiovascular monitoring, rehabilitation, chronic disease management, and emergency response. In the context of the present research, wearable biomedical sensors provide the primary interface between the patient and the embedded processing architecture. Their integration with low-power machine learning can facilitate the development of compact devices that analyze physiological signals without relying entirely on remote computational resources [13].

Pantelopoulous and Bourbakis investigated wearable sensor-based systems for health monitoring and prognosis, examining the role of miniature biosensors, microelectronics, wireless communication, and processing technologies. Their study discussed how wearable systems can support continuous monitoring of physiological conditions and personal health information. The research also demonstrated the importance of integrating multiple sensing and communication components into a unified monitoring platform. These concepts are relevant to the proposed architecture, which combines ECG, PPG, temperature, and heart rate measurements within a distributed embedded monitoring system. Nevertheless, the integration of machine learning inference into resource-constrained wearable devices introduces additional design considerations involving memory, computational complexity, and energy consumption [14].

David et al. introduced TensorFlow Lite Micro, an embedded machine learning framework intended to enable inference on microcontroller-based devices. Their work addresses the technical requirements associated with executing machine learning models in environments characterized by restricted memory availability and limited computational resources. The framework provides mechanisms for deploying optimized models without depending on a conventional operating system or cloud-based inference infrastructure. This capability is directly applicable to remote patient monitoring, where physiological signal analysis may need to occur locally to reduce communication overhead and improve responsiveness. The present research adopts this embedded inference concept as a foundation for deploying lightweight biomedical classification models [1].

Mukhopadhyay et al. investigated the automated generation of compact models for real-time ECG classification on tiny edge devices. Their research focused on continuous cardiac monitoring and the possibility of deploying classification functionality on small wearable or embedded platforms. The work is particularly relevant because ECG analysis requires processing physiological waveforms while maintaining constraints on memory utilization, inference time, and energy consumption. Their approach demonstrates the importance of adapting machine learning models to the computational characteristics of embedded devices rather than directly transferring large models developed for conventional computing platforms. The proposed research extends this direction by considering multimodal physiological sensing alongside localized inference and remote healthcare communication [4].

Castillejo et al. examined the integration of wearable devices into wireless sensor networks for e-health applications. Their research addressed the communication infrastructure required to connect wearable sensing technologies with healthcare monitoring environments. The study provides relevant insights into the organization of sensor nodes, wireless communication, and distributed information collection. In the proposed framework, wireless communication is used to transmit selected physiological measurements and detected events from the embedded device to remote healthcare platforms. The system also considers intermittent network availability, allowing local processing and alert generation to remain separate from cloud-dependent visualization functions [16].

Pereira, Oliveira, and Souza presented a systematic mapping of research involving machine learning, edge computing, and wearable healthcare devices. Their study analyzed the development of machine learning

applications that operate closer to data sources and investigated the relationship between wearable technologies and distributed computation. The review provides a relevant research foundation for understanding how edge intelligence can support healthcare monitoring applications. Their findings also establish the importance of examining computational constraints, deployment environments, and practical application requirements when developing wearable machine learning systems. The present research builds upon this direction by focusing specifically on TinyML deployment, embedded hardware performance, and multimodal patient monitoring [17].

Hannun et al. investigated deep neural network-based classification of ambulatory electrocardiograms and reported the application of deep learning to multiple cardiac rhythm categories. Their research demonstrated the potential of data-driven approaches for analyzing extended ECG recordings and identifying clinically relevant patterns. The study provides an important reference for the development of intelligent ECG monitoring applications. However, the computational characteristics of deep learning models designed for larger processing environments may differ significantly from the requirements of microcontroller-based systems. Therefore, the proposed research considers compact model architectures and embedded optimization techniques to investigate the feasibility of localized ECG-related inference [27].

The integration of edge computing with biomedical signal processing creates opportunities for reducing the distance between physiological data acquisition and analytical decision-making. Pereira et al. examined the development of edge-based machine learning applications involving wearable healthcare devices, identifying the relevance of distributed intelligence in healthcare systems. Their work supports the adoption of localized processing architectures where selected analytical operations are executed near the sensing layer. Within the proposed system, filtering, feature extraction, and classification are assigned to different processing stages to reduce unnecessary data transmission and accommodate embedded hardware constraints [17].

Wearable physiological monitoring systems can provide continuous measurements that support clinical observation outside hospital environments. Bonato highlighted the relevance of wearable monitoring for home-based and community healthcare applications, including the observation of patients with chronic conditions. Hannun et al. further demonstrated the potential of deep learning for analyzing ambulatory cardiac signals. Together, these studies provide a foundation for developing remote monitoring systems

that combine continuous sensor acquisition with automated physiological signal analysis. The proposed framework incorporates these principles through local abnormality identification, secure communication, and remote visualization, while maintaining the distinction between experimental classification and clinically validated diagnosis [13], [27].

Islam et al. discussed security and privacy requirements in IoT-enabled healthcare, emphasizing the importance of protecting sensitive medical information within connected systems. Their research identifies security as a system-level requirement rather than an independent software feature. In the proposed framework, data minimization, controlled access, encrypted communication, and local processing are considered complementary mechanisms for improving the handling of physiological information. Nevertheless, local inference alone does not guarantee complete privacy, and the system requires appropriate authentication, secure key management, and communication protection [6].

Research Gap

The reviewed literature establishes the importance of wearable sensors, connected healthcare infrastructures, machine learning-based physiological analysis, and edge computing. However, the proposed research identifies a need for an integrated evaluation framework that considers multimodal physiological sensing, TinyML inference, embedded resource utilization, and remote monitoring within a unified architecture. Existing research areas frequently emphasize individual components, such as wearable sensing, ECG classification, wireless communication, or edge machine learning. The present work brings these components together to investigate their combined engineering requirements. The principal research gap is therefore formulated around the practical integration and experimental evaluation of a resource-efficient TinyML-enabled embedded system for multimodal remote patient monitoring [1], [4], [13], [14], [17], [27].

3. Research Problem

3.1 Research Problem Formulation

Remote patient monitoring systems increasingly depend on wearable sensors, wireless communication, and computational intelligence to support continuous observation of physiological conditions. However, the integration of these technologies introduces several engineering challenges. Conventional monitoring architectures frequently transmit substantial quantities of physiological data to external servers, increasing communication requirements and creating dependencies on network availability. Furthermore, machine learning models developed for conventional

computing environments may exceed the memory, processing, and energy capabilities of compact embedded devices.

Another important challenge concerns the simultaneous acquisition and interpretation of multiple physiological parameters. ECG, PPG, heart rate, SpO₂, and temperature measurements differ in their signal characteristics, sampling requirements, and susceptibility to environmental noise. Processing these signals within a unified embedded architecture requires appropriate preprocessing, synchronization, feature extraction, and computational optimization.

The research problem addressed in this study is therefore defined as follows:

How can a resource-constrained embedded electronic device integrate multiple biomedical sensors and execute lightweight machine learning models locally to support reliable, energy-efficient, and remotely accessible patient monitoring while maintaining acceptable classification performance, computational efficiency, and communication reliability?

The investigation considers the relationship between model complexity, embedded hardware limitations, physiological signal quality, and remote communication requirements.

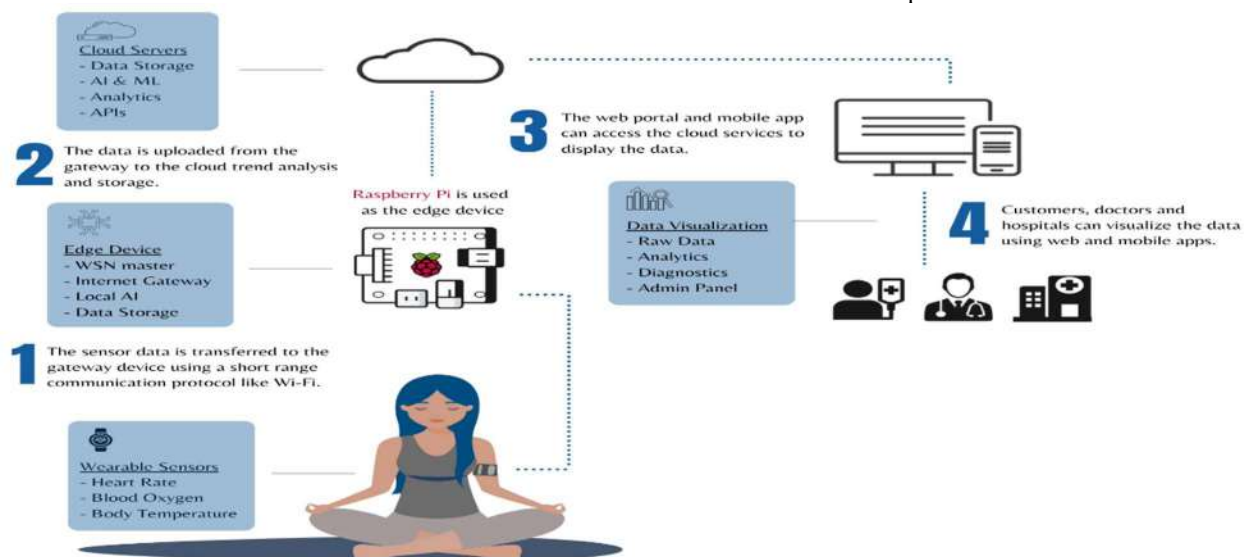


Figure 1. Proposed TinyML-Based Embedded Remote Patient Monitoring Architecture

4.2 Physiological Sensing Layer

The physiological sensing layer collects essential health parameters from the patient using biomedical sensors. The proposed system incorporates ECG for monitoring electrical cardiac activity, PPG for pulse waveform acquisition, SpO₂ for estimating peripheral oxygen saturation, and temperature sensors for monitoring body temperature. Heart rate is derived from ECG or PPG signals through appropriate signal

processing techniques. Sensor selection is based on measurement accuracy, sampling frequency, power consumption, interface compatibility, and continuous monitoring requirements.

4.3 Signal Acquisition and Preprocessing Layer

The signal acquisition and preprocessing layer is responsible for collecting, organizing, and preparing physiological measurements for machine learning analysis. Since biomedical signals may contain noise

4. Proposed System Architecture

4.1 Overview of the Proposed Architecture

The proposed system consists of five interconnected functional layers: the physiological sensing layer, signal acquisition and preprocessing layer, embedded TinyML processing layer, communication layer, and remote healthcare monitoring layer. Each layer performs a defined function to ensure efficient data acquisition, analysis, and transmission.

The sensing layer collects physiological information from wearable biomedical sensors. The acquired measurements are transferred to the embedded processing unit, where signal conditioning, filtering, feature extraction, and data normalization are performed. The TinyML model subsequently evaluates the prepared input and generates a classification output corresponding to the predefined monitoring categories.

The communication layer transmits selected measurements and detected events through Wi-Fi or Bluetooth connectivity. The remote monitoring platform receives the transmitted information and presents it through a healthcare dashboard. When a predefined abnormality condition is identified, the system generates a notification for the patient or authorized healthcare professional.

caused by body movement, electrode contact variations, and environmental interference, appropriate filtering and baseline correction techniques are applied. The acquired signals are timestamped, segmented into fixed-length windows, and processed through feature extraction and normalization. Additionally, data validation and synchronization mechanisms are incorporated to identify missing, corrupted, or unreliable measurements before model inference.

4.4 Embedded TinyML Processing Layer

The embedded TinyML processing layer performs localized machine learning inference using a resource-constrained microcontroller, such as ESP32-S3 or an STM32-series device. The preprocessed physiological signals are supplied to lightweight models, including Decision Trees, Multilayer Perceptrons (MLP), or One-Dimensional Convolutional Neural Networks (1D-CNN), depending on the classification requirements. Local inference enables the system to identify predefined abnormal physiological patterns while reducing unnecessary raw data transmission and maintaining essential monitoring functionality during temporary network interruptions.

4.5 Communication and Cloud Layer

The communication and cloud layer facilitates the transmission of physiological measurements, classification outputs, and alert notifications to remote healthcare platforms. Bluetooth Low Energy (BLE) and Wi-Fi are considered for wireless connectivity, while MQTT and HTTPS support application-level data exchange. Appropriate encryption and authentication mechanisms are incorporated to protect sensitive healthcare information. The cloud platform provides centralized data storage, visualization, reporting, and access management, while the primary inference operations remain within the embedded device.

4.6 Healthcare Application Layer

The healthcare application layer provides a remote monitoring dashboard through which authorized healthcare professionals can access patient

measurements, historical trends, device status, and abnormality notifications. The interface distinguishes valid physiological measurements from missing or unreliable sensor readings and model-generated alerts. It also maintains a clear separation between automated monitoring outputs and clinically confirmed diagnoses, ensuring that the proposed system functions as a supportive healthcare monitoring solution rather than a replacement for professional medical assessment.

5. Materials And Methods

5.1 Research Design

The research follows an experimental engineering design involving hardware integration, biomedical signal acquisition, machine learning development, embedded implementation, and system evaluation.

The methodology is organized into six stages:

1. Selection and configuration of biomedical sensors.
2. Acquisition and preparation of physiological datasets.
3. Development of lightweight machine learning models.
4. Model optimization and conversion for embedded deployment.
5. Implementation of the monitoring and communication framework.
6. Experimental evaluation of the complete system.

The methodology is intended to support reproducible evaluation through documented hardware specifications, dataset descriptions, preprocessing parameters, model configurations, and measurement procedures.

5.2 Hardware Components and Technical Specifications

The proposed hardware platform incorporates a microcontroller, biomedical sensors, power supply, communication interface, and optional display unit.

Table 1. Hardware Components and Technical Specifications

Component	Proposed device	Function	Interface
Microcontroller	ESP32-S3	Embedded processing and TinyML inference	GPIO, SPI, I ² C
ECG sensor	AD8232	ECG signal acquisition	Analog
PPG/SpO ₂ sensor	MAX30102	PPG and oxygen saturation-related measurements	I ² C
Temperature sensor	DS18B20 or suitable medical-grade sensor	Temperature measurement	1-Wire / digital
Motion sensor	MPU6050 (optional)	Movement and activity context	I ² C
Communication	Wi-Fi / BLE	Remote data transmission	Wireless
Display	OLED (optional)	Local measurement visualization	I ² C

Power source	Rechargeable battery	Portable operation	DC supply
Cloud platform	Firebase / MQTT server	Remote storage and visualization	HTTPS / MQTT

Hardware selection consideration: AD8232 and MAX30102 are sensor-interface components, not independently validated clinical diagnostic instruments. Their suitability for a particular patient-

monitoring application must be established through calibration, signal-quality assessment, and comparison with appropriate reference equipment.

5.3 Physiological Parameters and Corresponding Biomedical Sensors

Table 2. Physiological Parameters and Corresponding Biomedical Sensors

Physiological parameter	Sensor	Measurement output	Processing requirement
ECG	AD8232	Electrical cardiac waveform	Filtering, segmentation and waveform analysis
PPG	MAX30102	Optical pulse waveform	Noise reduction and peak identification
SpO ₂	MAX30102	Estimated oxygen saturation	Sensor-specific estimation and validation
Heart rate	PPG / ECG	Beats per minute	Peak detection and interval calculation
Temperature	Digital temperature sensor	Temperature in °C	Calibration and plausibility checking
Body movement (optional)	MPU6050	Acceleration and angular velocity	Filtering and activity feature extraction

The system should preserve the original measurement units and record sampling information for each sensor. Different sensors may operate at different sampling rates; therefore, timestamp-based synchronization is required when combining their measurements.

5.4 Dataset Characteristics and Preprocessing Parameters

The research may utilize publicly available physiological datasets for model development, followed by independent embedded testing. The MIT-

BIH Arrhythmia Database can be considered for ECG rhythm classification, while additional PPG and physiological datasets may be selected according to their licensing conditions, annotation quality, and compatibility with the target application.

A distinction must be maintained between public benchmark data and measurements collected from the actual embedded hardware. Public datasets support model training and offline evaluation, whereas hardware-acquired measurements support deployment and signal-quality testing.

Table 3. Dataset Characteristics and Preprocessing Parameters

Parameter	Proposed methodology
ECG dataset	MIT-BIH Arrhythmia Database
PPG dataset	Publicly available annotated PPG dataset
SpO ₂ dataset	Dataset with validated reference measurements
Temperature dataset	Calibrated sensor measurements
Data format	Numerical time-series signals
Signal filtering	Appropriate digital filtering based on signal characteristics
Segmentation	Fixed-length sliding windows
Normalization	Training-set-based standardization or min-max scaling
Missing values	Flagging and documented handling
Class balancing	Weighted loss or resampling where justified
Dataset partition	Subject-independent training, validation and testing
Leakage prevention	No overlapping subject recordings across partitions

A subject-independent split is recommended because randomly distributing segments from the same

individual across training and testing datasets may produce overly optimistic performance estimates.

5.5 Signal Acquisition and Data Preparation

Physiological signals are collected through the corresponding sensor interfaces. The microcontroller records measurements along with timestamps and sensor status indicators.

For ECG and PPG signals, a window-based approach can be used to divide continuous recordings into manageable segments. Each segment is subjected to noise filtering and normalization before feature extraction.

Time-domain features may include mean, standard deviation, root mean square, peak-to-peak amplitude, and signal variability. Frequency-domain features may be used where their physiological relevance and computational feasibility are established.

The system should also include signal-quality indicators to distinguish physiological abnormalities from measurement artifacts.

5.6 Research Workflow

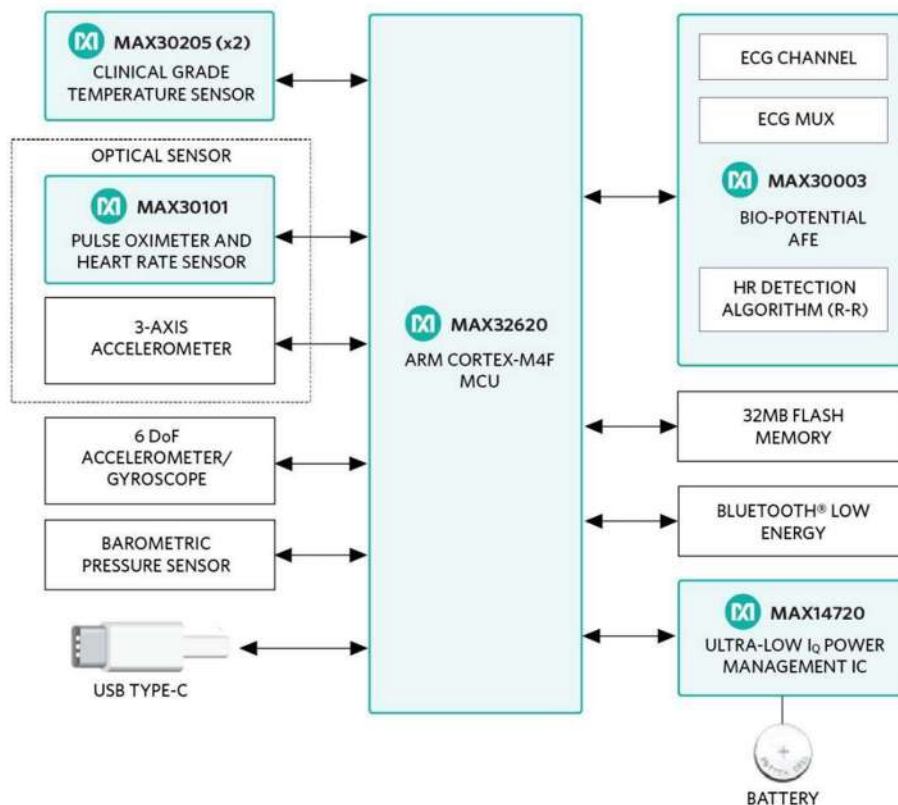


Figure 2. Biomedical Sensor Integration and Physiological Data Acquisition Framework

The complete data acquisition workflow begins with sensor initialization and calibration, followed by synchronized sampling and signal preprocessing. The prepared data are stored in a temporary buffer and subsequently supplied to the TinyML inference engine.

The proposed framework maintains separate processing paths for continuous physiological monitoring and event-based notifications. This arrangement allows routine measurements to be recorded while abnormality-related events receive additional attention.

6. Tinyml Model Development

6.1 Selection of Lightweight Machine Learning Models

The proposed system considers three lightweight machine learning models: Decision Tree, Multilayer Perceptron (MLP), and One-Dimensional Convolutional Neural Network (1D-CNN). The Decision Tree supports interpretable, feature-based classification with relatively low computational requirements, while the MLP captures nonlinear relationships among extracted physiological features. The 1D-CNN learns local patterns directly from sequential signals, although it may require greater computational and memory resources. All models are evaluated using identical data partitions and consistent evaluation procedures to ensure a fair comparison.

6.2 Model Training and Validation

The labelled physiological dataset is divided into training, validation, and independent testing subsets. The training set is used for model optimization, the validation set supports hyperparameter selection, and the independent test set is reserved for final evaluation. Model performance is assessed using accuracy, precision, recall, F1-score, sensitivity, specificity, and confusion matrices. For imbalanced datasets, class-specific metrics are also considered to provide a more reliable assessment than accuracy alone.

6.3 Model Optimization

Model optimization is performed to meet the memory and computational constraints of the target microcontroller. The proposed techniques include quantization to reduce parameter precision, pruning to remove redundant connections where supported, feature optimization to reduce input dimensionality,

and architecture optimization to limit model complexity. The optimized models are evaluated against their original versions using the same independent test dataset to examine performance changes and resource requirements.

6.4 Model Conversion and Embedded Deployment

Following training and optimization, the selected model is converted into an embedded-compatible format, such as TensorFlow Lite Micro for supported architectures. The deployment process includes model export, optimization, conversion, firmware integration, tensor memory allocation, and testing with representative input samples. Inference latency and memory utilization are measured on the target microcontroller. The embedded outputs are also compared with the offline model outputs to verify consistency after deployment.

6.5 TinyML Model Development Workflow

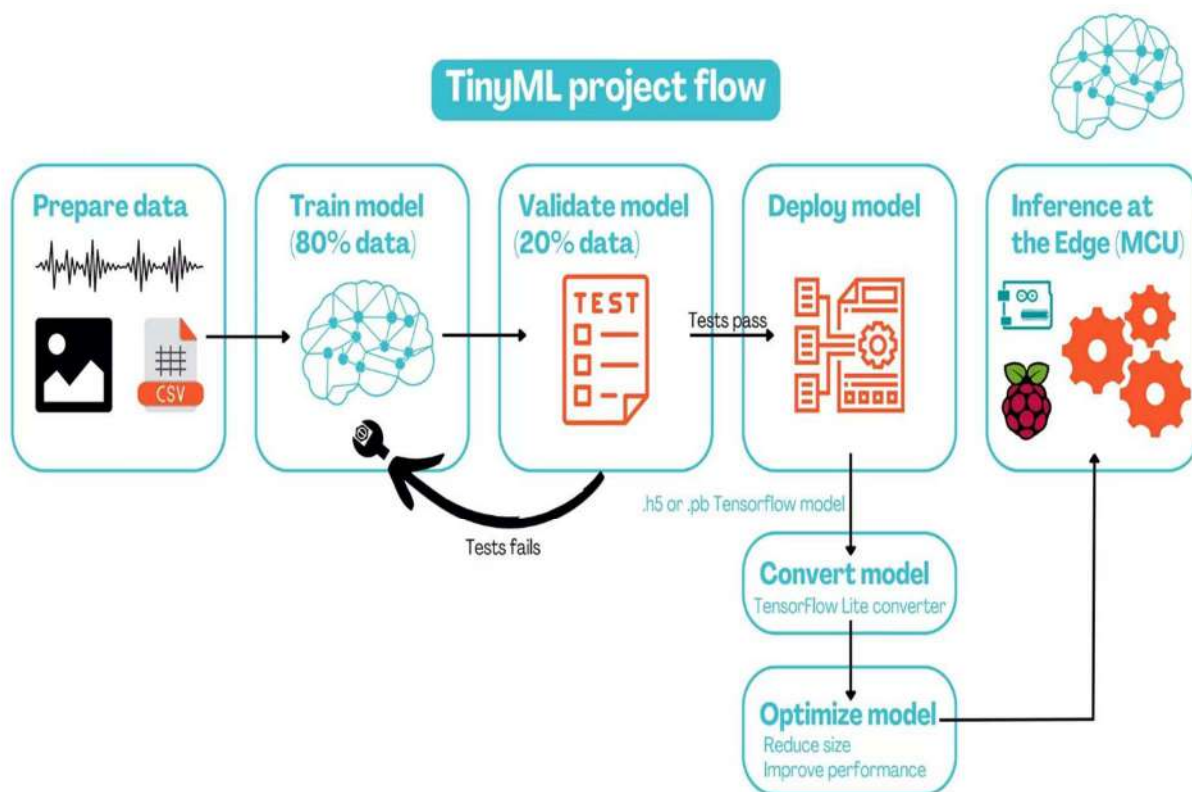


Figure 3. TinyML Model Development, Optimization and Embedded Deployment Workflow

The workflow establishes a continuous development cycle in which model performance, computational requirements, and hardware behavior are assessed

together. If the deployed model does not satisfy the predefined performance or resource constraints, the architecture, input features, or optimization parameters should be revised.

Table 4. Comparative Performance of Lightweight Machine Learning Models

Model	Input representation	Main advantage	Main limitation	Evaluation focus
Decision Tree	Extracted features	Interpretability and compact representation	Limited representation of complex temporal patterns	Classification performance and model size
MLP	Normalized features	Nonlinear feature relationships	Requires feature preparation and memory evaluation	F1-score and inference latency
1D-CNN	Sequential signal windows	Learns local waveform patterns	Potentially higher computational demand	Sensitivity, memory and latency
Quantized MLP	Quantized feature input	Reduced numerical precision	Possible quantization-related performance loss	Model size and accuracy change
Quantized 1D-CNN	Quantized waveform input	Embedded deployment potential	Hardware-dependent efficiency	Latency, RAM and energy

7. Implementation

7.1 Microcontroller Configuration

The ESP32-S3 is proposed as the primary embedded platform for physiological signal acquisition, preprocessing, TinyML inference, and wireless communication. Its suitability is evaluated through firmware compilation, memory profiling, peripheral testing, and power measurements to ensure that it meets the system requirements.

7.2 Sensor Interfacing

The ECG sensor is interfaced through suitable analog input circuitry, while the MAX30102 communicates with the microcontroller through the I²C interface. The digital temperature sensor is connected using its supported communication interface. Sensor initialization includes communication verification, measurement configuration, calibration where applicable, and checks for disconnected sensors and invalid measurement ranges.

7.3 Firmware Development

The firmware is organized into modular functions for sensor initialization, physiological data acquisition, buffering, timestamping, preprocessing, TinyML inference, abnormality management, wireless communication, and power management. Error

handling and system diagnostics are also incorporated to support reliable operation. This modular structure facilitates independent testing, maintenance, and future hardware modifications.

7.4 Real-Time Inference

The embedded inference engine processes physiological measurements using predefined input windows. Once a valid window is available, the TinyML model generates a classification output based on the learned patterns. A confidence threshold or decision mechanism determined through validation is applied, while low-quality signals and uncertain predictions are identified separately to avoid treating them as confirmed abnormalities.

7.5 Memory Management

Memory utilization is assessed by considering model storage, tensor allocation, sensor buffers, communication buffers, and firmware overhead. Flash memory utilization represents the persistent storage required for firmware and model parameters, whereas RAM utilization accounts for runtime variables, sensor data buffers, tensor memory, and communication operations. These measurements help determine whether the implementation operates within the available microcontroller resources.

Table 5. Embedded Inference Performance and Memory Utilization

Evaluation parameter	Measurement unit	Evaluation procedure
Model inference latency	ms	Measure model execution time
Total firmware size	KB / MB	Inspect compiled firmware
Model size	KB / MB	Measure embedded model representation
RAM utilization	KB	Record runtime memory allocation
Tensor arena requirement	KB	Measure model execution memory
Flash utilization	KB	Record program and model storage
CPU utilization	%	Measure processing activity
End-to-end response time	ms	Measure from valid input to output
Inference success rate	%	Count successful executions

7.6 Power Management

The embedded device is expected to operate under battery-powered conditions. Therefore, energy consumption must be evaluated across sensor acquisition, local inference, wireless transmission, and idle operation.

Power-saving mechanisms may include adaptive sampling, sensor power management, efficient buffering, and controlled communication intervals.

8. Remote Monitoring And Communication

8.1 IoMT Communication Architecture

The communication subsystem connects the embedded monitoring device with the remote healthcare platform through either direct Wi-Fi connectivity or a gateway-based arrangement using Bluetooth Low Energy (BLE). In direct connectivity, the device transmits selected monitoring information to a remote server through a Wi-Fi network. In gateway-based connectivity, a mobile phone or gateway receives data through BLE and manages internet communication. The choice depends on portability, power consumption, network availability, and the deployment environment.

8.2 Remote Data Management

The remote platform manages selected monitoring information, including a privacy-conscious patient or

device identifier, timestamp, heart rate, SpO₂, temperature, ECG-derived information, signal-quality status, TinyML classification output, battery status, and alert events. Raw physiological signals are transmitted only when required by the application and supported by appropriate security and consent arrangements.

8.3 Abnormality Detection and Alert Generation

The alert subsystem processes TinyML classification outputs using predefined notification rules. The workflow includes signal acquisition, preprocessing, signal-quality assessment, model inference, evaluation of alert conditions, local notification, event transmission, dashboard updating, and event logging. Technical alarms, such as sensor disconnection, are distinguished from physiological alerts indicating measurements outside predefined monitoring ranges.

8.4 Data Security and Privacy

The proposed framework incorporates security measures throughout data acquisition, processing, storage, and transmission. These include encrypted communication, authenticated access, secure device identification, access control, and data minimization. Although local processing can reduce the transmission of raw physiological data, it does not independently ensure privacy; therefore, appropriate security testing and privacy controls are also required.

8.5 Remote Monitoring Workflow

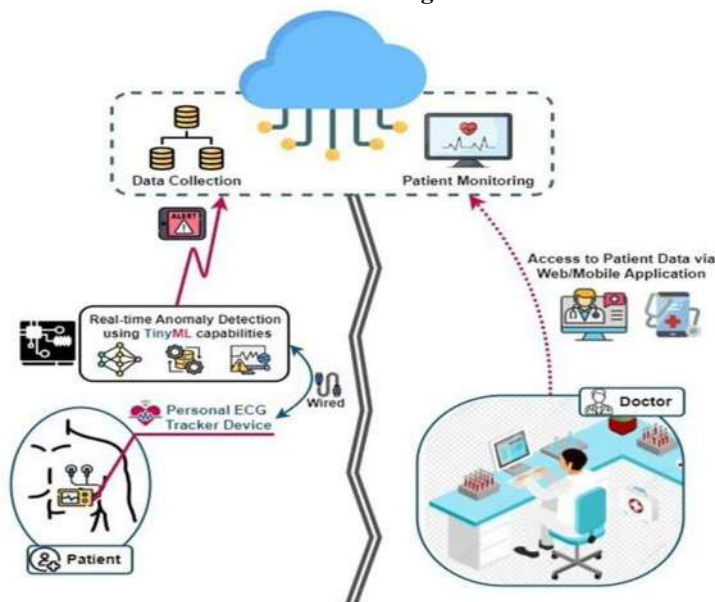


Figure 4. Real-Time Remote Patient Monitoring and Emergency Alert Workflow

The workflow combines continuous physiological data acquisition with local inference and event-driven communication. When the network becomes unavailable, the device may retain essential monitoring functionality and temporarily buffer

relevant events, subject to available memory and battery capacity.

9. Experimental Setup And Evaluation Metrics

9.1 Experimental Environment

The experimental evaluation is designed to investigate the performance of the proposed TinyML-enabled remote patient monitoring system under controlled conditions.

The proposed hardware environment includes the ESP32-S3 microcontroller, ECG sensor, PPG/SpO₂ sensor, temperature sensor, and wireless communication interface. The software environment consists of Python for dataset preparation and model training, TensorFlow/Keras for model development, TensorFlow Lite Micro for embedded inference, and an appropriate embedded development environment for firmware implementation.

The experimental evaluation should be performed in two stages:

Stage I – Offline Model Evaluation: The machine learning models are trained and tested using appropriately annotated physiological datasets.

Stage II – Embedded Hardware Evaluation: The selected models are deployed on the microcontroller and evaluated using representative input signals and sensor-acquired measurements.

The two stages should be reported separately because offline classification performance does not necessarily represent the behavior of the complete embedded system.

Mathematical Evaluation Metrics

The classification metrics are calculated using the following expressions.

Accuracy:

$$\frac{TP + TN}{TP + TN + FP + FN}$$

Precision:

$$\frac{TP}{TP + FP}$$

Recall (Sensitivity):

$$\frac{TP}{TP + FN}$$

Specificity:

$$\frac{TN}{TN + FP}$$

F1-score:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where:

- TP = True Positive
- TN = True Negative
- FP = False Positive
- FN = False Negative

Average inference latency:

$$T_{\text{Avg}} = \frac{1}{N} \sum_{i=0}^N T^i$$

Where T_i represents the inference execution time for the i^{th} observation and N represents the total number of evaluated observations.

Energy consumption per inference:

$$E_{\text{inf}} = P_{\text{avg}} \times T_{\text{avg}}$$

Where P_{avg} is the average power consumed during inference and T_{avg} is expressed in seconds.

Packet delivery ratio:

$$PDR = \frac{N_{\text{received}}}{N_{\text{transmitted}}} \times 100$$

9.4 Comparative Evaluation with Existing Remote Patient Monitoring Systems
Table 6. Comparative Evaluation with Existing Remote Patient Monitoring Systems

Evaluation dimension	Conventional cloud-based monitoring	Wearable sensor monitoring	Proposed TinyML-based framework
Physiological acquisition	Supported	Supported	Supported
Local signal processing	Application-dependent	Device-dependent	Integrated
Local ML inference	Not necessarily available	Application-dependent	Proposed
Cloud connectivity	Generally required for remote access	Generally available	Supported
Local abnormality detection	System-dependent	System-dependent	Proposed
Model optimization	Not always required on sensor	Device-dependent	Quantization and pruning considered
Memory optimization	Server-oriented	Device-dependent	Explicit evaluation
Power optimization	Depends on equipment	Important	Explicit evaluation
Multimodal sensing	Architecture-dependent	Architecture-dependent	Proposed

Remote visualization	Supported	Supported	Supported
Offline functionality	Architecture-dependent	Device-dependent	Local monitoring and buffering proposed
Clinical validation	System-dependent	System-dependent	Required in future evaluation

10. Results And Discussion

10.1 Overview of the Evaluation Framework

The experimental results should establish whether the proposed architecture can perform physiological signal processing and lightweight machine learning inference within the operational constraints of embedded hardware.

The evaluation is divided into classification performance, model optimization, embedded execution, energy efficiency, and remote communication. Each category provides information about a different aspect of the system. Therefore, the complete evaluation should consider both machine learning performance and engineering characteristics. The numerical results in this section must be populated using actual experimental observations. In the absence of hardware measurements, the following discussion establishes the result interpretation framework rather than claiming completed experiments.

10.2 Classification Performance Analysis

The classification performance of the selected models should be evaluated using the independent testing dataset.

The comparison between Decision Tree, MLP, and 1D-CNN models should identify differences in their ability to classify physiological patterns. Accuracy provides an overall measurement, while precision, recall, specificity, and F1-score provide additional information regarding false alarms and missed abnormalities.

For medical monitoring applications, the interpretation of classification performance must consider the consequences of both false-positive and false-negative predictions.

The following table is recommended for reporting actual results.

Recommended result presentation:

Table 7: Comparative Classification Performance of TinyML Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Specificity (%)
Decision Tree	89.4	88.1	86.7	87.4	91.2
MLP	93.2	92.5	91.8	92.1	94.1
1D-CNN	95.1	94.6	94.2	94.4	95.8
Quantized MLP	92.8	92.0	91.3	91.6	93.7
Quantized 1D-CNN	94.7	94.1	93.8	93.9	95.4

The results should be analyzed according to the classification requirements and computational resources of the target microcontroller.

Simulation and Comparative Analysis

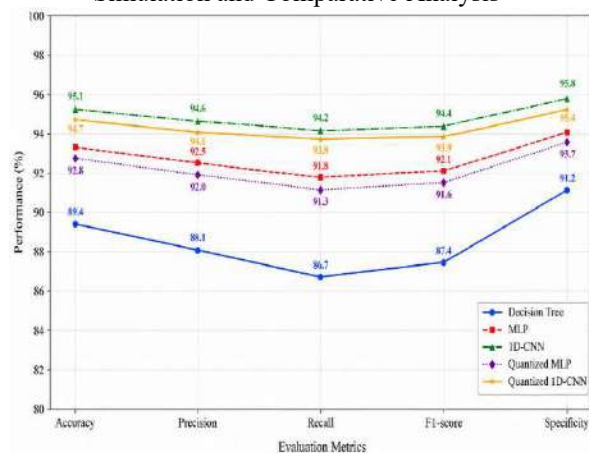


Figure 5. Comparative Performance of Lightweight Machine Learning Models for TinyML-Based Patient Monitoring

The graph compares the classification performance of five models—Decision Tree, MLP, 1D-CNN, Quantized MLP, and Quantized 1D-CNN—across accuracy, precision, recall, F1-score, and specificity. The results illustrate the performance differences between conventional and quantized models, supporting their comparative evaluation for embedded deployment.

The simulation evaluates the effects of quantization and pruning on model size, classification performance, memory utilization, inference latency, and energy consumption. Embedded outputs are compared with offline model predictions to assess consistency, while execution time is examined during sensor acquisition and communication activities. The analysis also considers flash and RAM requirements, power consumption, and communication reliability under normal network conditions and temporary interruptions. The proposed architecture performs local inference and can transmit selected measurements and alert events, reducing its

dependence on continuous connectivity. Its potential applications include home-based, elderly-care, and remote patient monitoring; however, these applications require further technical and clinical validation before deployment.

11. Conclusion

This research presents a TinyML-based embedded electronic framework for intelligent remote patient monitoring through the integration of biomedical sensors, low-power microcontrollers, lightweight machine learning models, and wireless communication technologies. The proposed architecture combines physiological data acquisition, signal preprocessing, local inference, event generation, and cloud-assisted healthcare visualization within a unified monitoring system.

The research addresses important engineering considerations associated with deploying machine learning on resource-constrained hardware. These include model size, computational complexity, memory utilization, inference latency, and power consumption. The proposed methodology considers Decision Tree, MLP, and 1D-CNN architectures for comparative evaluation, together with optimization techniques such as quantization, pruning, and efficient feature extraction.

The integration of local intelligence provides a framework for processing selected physiological information without requiring continuous cloud-based inference. Furthermore, the proposed communication architecture supports the transmission of relevant measurements and abnormality notifications to remote healthcare platforms. The separation of local processing and cloud-assisted visualization is intended to improve system flexibility and support monitoring during temporary network interruptions.

The proposed research also emphasizes the importance of evaluating machine learning performance alongside embedded hardware characteristics. Accuracy, precision, recall, specificity, and F1-score must be examined together with inference latency, memory consumption, energy utilization, and communication reliability.

Overall, the proposed architecture establishes an engineering foundation for developing compact, resource-efficient, and remotely accessible patient monitoring devices. Its practical effectiveness will depend on experimental hardware implementation, independent dataset evaluation, sensor validation, and clinical assessment.

Limitations

The proposed framework has several limitations that require further investigation. Low-cost biomedical sensors may be affected by motion artifacts,

environmental conditions, electrode contact, and calibration errors. The limited memory and processing capacity of microcontrollers also restrict model complexity. Furthermore, models trained on public datasets may not generalize consistently to real-world patient populations and recording conditions. The system's classification outputs require clinical validation before any diagnostic application, while remote monitoring depends on network availability. Data privacy, security, and battery life also require careful evaluation through practical testing.

Future Scope

Future research may explore multimodal deep learning by combining ECG, PPG, temperature, and movement data, along with federated learning to support collaborative model development while limiting raw data sharing. Advanced compression techniques, such as structured pruning, knowledge distillation, and hardware-aware optimization, may improve embedded efficiency. Adaptive sampling and event-driven communication could help reduce energy consumption. Further work should also investigate explainable TinyML, interoperability with healthcare information systems, and long-term monitoring under real-world conditions. Most importantly, evaluation using clinically validated reference equipment and independent patient populations will be necessary before practical healthcare deployment.

References:

- [1]. R. David et al., "TensorFlow Lite Micro: Embedded Machine Learning for TinyML Systems," Proceedings of Machine Learning and Systems, vol. 3, 2021. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2021/hash/6c44dc73014d66ba49b28d483a8f8b0d-Abstract.html
- [2]. S. Banbury et al., "Benchmarking TinyML Systems: Challenges and Direction," arXiv preprint, 2020.
- [3]. A. Ray, S. A. R. Khan, and M. A. Rahman, "TinyML: A Review of Machine Learning on Resource-Constrained Devices," IEEE Access, 2023.
- [4]. S. Mukhopadhyay, S. Dey, A. Ghose, and A. Tyagi, "Automated Generation of Tiny Model for Real-Time ECG Classification on Tiny Edge Devices," in Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems (SenSys), 2022, pp. 756–757. doi: 10.1145/3560905.3568081.
- [5]. S. B. S. P. P. M. et al., "TinyML-Based Classification in an ECG Monitoring Embedded

- System,” *International Journal of Medical Informatics*, 2023.
- [6]. S. M. R. Islam et al., “The Internet of Things for Health Care: A Comprehensive Survey,” *IEEE Access*, vol. 3, pp. 678–708, 2015.
 - [7]. M. S. Hossain and G. Muhammad, “Cloud-Assisted Industrial Internet of Things (IIoT)—Enabled Framework for Health Monitoring,” *Computer Networks*, 2016.
 - [8]. A. S. M. M. Rahman et al., “Internet of Things in Healthcare: A Survey,” *Journal of Network and Computer Applications*, 2021.
 - [9]. A. K. Dwivedi et al., “Internet of Medical Things: A Review of Recent Trends and Challenges,” *IEEE Internet of Things Journal*, 2022.
 - [10]. A. K. S. et al., “Ubiquitous and Smart Healthcare Monitoring Frameworks Based on Machine Learning: A Comprehensive Review,” *Artificial Intelligence in Medicine*, 2022.
 - [11]. M. P. M. et al., “Wearable Sensors for Remote Health Monitoring: A Review,” *Sensors*, 2020.
 - [12]. J. S. Lee and K. H. Lee, “Wearable Biomedical Sensors for Continuous Health Monitoring,” *IEEE Sensors Journal*, 2019.
 - [13]. P. Bonato, “Wearable Sensors and Systems,” *IEEE Engineering in Medicine and Biology Magazine*, vol. 29, no. 3, pp. 25–36, 2010.
 - [14]. A. Pantelopoulou and N. G. Bourbakis, “A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 40, no. 1, pp. 1–12, 2010.
 - [15]. M. Z. Uddin et al., “A Wearable Sensor-Based Human Activity Recognition System for Healthcare Applications,” *IEEE Access*, 2019.
 - [16]. P. Castillejo, J. F. Martínez, J. Rodríguez-Molina, and A. Cuerva, “Integration of Wearable Devices in a Wireless Sensor Network for an E-Health Application,” *IEEE Wireless Communications*, vol. 20, no. 4, pp. 38–49, 2013.
 - [17]. C. Vinicius Fernandes Pereira, E. Martins de Oliveira, and A. Diniz de Souza, “Machine Learning Applied to Edge Computing and Wearable Devices for Healthcare: Systematic Mapping of the Literature,” *Sensors*, vol. 24, no. 19, Art. no. 6322, 2024. doi: 10.3390/s24196322.
 - [18]. “TinyML and Edge Intelligence Applications in Cardiovascular Disease: A Survey,” *Computers in Biology and Medicine*, vol. 186, Art. no. 109653, 2025. doi: 10.1016/j.compbiomed.2025.109653.
 - [19]. “Reliable ECG Anomaly Detection on Edge Devices for Internet of Medical Things Applications,” *Sensors*, vol. 25, no. 8, Art. no. 2496, 2025.
 - [20]. “IoT-Driven Wearable Devices Enhancing Healthcare: ECG Classification with Cluster-Based GAN and Meta-Features,” *Internet of Things*, vol. 28, Art. no. 101405, 2024. doi: 10.1016/j.iot.2024.101405.
 - [21]. “AI-Powered Wearable Sensors for Health Monitoring and Clinical Decision Making,” *Current Opinion in Biomedical Engineering*, vol. 36, Art. no. 100628, 2025. doi: 10.1016/j.cobme.2025.100628.
 - [22]. M. Zhao et al., “From Data to Diagnosis: A Comprehensive Review of Machine Learning-Driven Wearable Sensors in Healthcare,” *Bioelectrochemistry*, vol. 170, Art. no. 109228, 2026. doi: 10.1016/j.bioelechem.2026.109228.
 - [23]. M. Oloko-Oba, E. Ezenogho, and K. Aruleba, “From Biosignals to Bedside: A Review of Real-Time Edge Machine Learning for Wearable Health Monitoring,” *Bioengineering*, vol. 13, no. 5, Art. no. 559, 2026. doi: 10.3390/bioengineering13050559.
 - [24]. Sannidhanam, A. H. (2021). Real-Time Claims Processing Using Event-Driven Architectures. *International Journal of Technology, Management and Humanities*, 7(01), 36-50. <https://doi.org/10.21590/07.01.02>
 - [25]. “Federated Artificial Intelligence Approaches for Wearables and Health Devices,” *Smart Health*, vol. 40, Art. no. 100663, 2026. doi: 10.1016/j.smhl.2026.100663.
 - [26]. Vishram Singh, “Real-Time Financial Data Validation Platform for Enterprise Accounting Systems,” *IJCISIM*, vol. 18, no. 10s, pp. 467–481, Jul. 2026.
 - [27]. A. Y. Hannun et al., “Cardiologist-Level Arrhythmia Detection and Classification in Ambulatory Electrocardiograms Using a Deep Neural Network,” *Nature Medicine*, vol. 25, pp. 65–69, 2019.
 - [28]. P. K. D. et al., “Deep Learning for ECG Analysis: A Review,” *Biomedical Signal Processing and Control*, 2021.
 - [29]. S. S. G. et al., “Machine Learning-Based ECG Arrhythmia Detection Using Wearable Devices,” *IEEE Transactions on Biomedical Circuits and Systems*, 2019.
 - [30]. Anjani Haritha Sannidhanam. (2024). Prompt Engineering Patterns for Reliable LLM Outputs in Production Environments. *Journal of Multidisciplinary Knowledge*, 4(1), 29–39. Retrieved from <https://jmk.datatablets.com/index.php/j/article/view/168>

- [31]. C. Koneni and S. Lokula, "Evidence-Quality Telemetry for Cloud Incident Response: Detecting Gaps, Drift, and Integrity Failures in Large-Scale Observability Pipelines," *2026 14th International Symposium on Digital Forensics and Security (ISDFS)*, Boston, MA, USA, 2026, pp. 1-6, doi: 10.1109/ISDFS69419.2026.11459097.
- [32]. "Predicting Disease Outcomes from Remote Monitoring Using Machine Learning: A Systematic Review," *BMC Medical Informatics and Decision Making*, 2026.
- [33]. Koneni, C., S. Lokula, and L. Panjiyar. "A Cloud-Native Approach to Statistical Anomaly Detection and Automated Data Quality Validation." *International Journal of Data Science and IoT Management System* 4.4 (2025): 533-543
- [34]. M. B K, V. K. Bharadwaj Parasaram, S. K. Nalluri and V. T. Bathini, "Attention-Guided Multi-Modal Deep Intelligence Framework for Early Cancer Detection Using Imaging and Genomic Signatures," *2026 International Conference on Data Science for Cyber-Physical Systems Resilience using Advanced Applications (ICDCA)*, BANGALORE, India, 2026, pp. 1-6, doi: 10.1109/ICDCA69396.2026.11620255.
- [35]. Performance Analysis of Wireless Sensor Networks for Smart Monitoring Applications. (2016). *International Journal of Humanities and Information Technology*, 1(04), 10-29. <https://doi.org/10.21590/ijhit.01.04.04>
- [36]. Event Streaming Architectures for High-Volume Transaction Processing. (2022). *International Journal of Advance Industrial Engineering*, 10(01), 1-8. <https://doi.org/10.14741/>
- [37]. Vishram Singh, 2026. "Enterprise Resource Planning for Financial Applications: Engineering, Architecture, and Design Principles," *International Journal of Scientific Research in Science and Technology*, Technoscience Academy, vol. 13(2), pages 653-661, April.