INTERNATIONAL JOURNAL OF INTELLIGENT COMMUNICATION AND COMPUTER SCIENCE

ISSN: 3048-7285

VOL. 3 NO. 2

Pages: 66

TABLE OF CONTENT

Contact: editor@ijiccs.in
ijiccs@gmail.com

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, 2025, Pages 1-12.

Accurate and Secure Person Identification Using ECG Signals and Deep Neural Networks

Short Paper

Sandeep Tripathi[1]

[1]Research Scholar, Department of Computer Science and Engineering, Bansal Institute of Engineering and Technology, Lucknow, Affiliated to AKTU, Lucknow, U.P. INDIA

Email: sansoftresearch@gmail.com
Received: 27 Feb 2025 Revised: 19 Aug 2025 Accepted: 08 Sep 2025

Abstract:

Person identification based on Electrocardiogram (ECG) signals has gained significant attention in biometric applications due to the unique and unchangeable nature of the heart's electrical activity. This paper presents a robust method for personal identification using ECG signals, leveraging deep neural networks (DNNs) to achieve high classification accuracy. We propose a novel DNN-based framework that extracts meaningful features from raw ECG data and utilizes a multi-layer neural network architecture to accurately distinguish between different individuals. The model is trained and evaluated on a diverse ECG dataset, demonstrating an impressive identification accuracy of over 99%. The system's ability to effectively handle variations in heart signals across different subjects highlights its potential as a reliable biometric tool. Our approach significantly outperforms traditional machine learning methods, offering superior accuracy, scalability, and robustness in real-world applications for secure person authentication and access control.

Keywords: ECG, Biometric, DNN

1. Introduction

In recent years, biometric systems have emerged as one of the most reliable and secure methods for personal identification, surpassing traditional password-based systems [1]. Among various biometric modalities, ECG signals have garnered attention for their potential in providing unique and highly secure identification, owing to the inherent uniqueness of each individual's cardiac rhythm [2]. Unlike other biometric features such as fingerprints or facial recognition, ECG signals are intrinsic, non-intrusive, and difficult to forge, making them a promising option for personal authentication in a variety of applications, including access control, secure transactions, and healthcare monitoring.

The human heart's electrical activity, which is reflected in the ECG signal, varies significantly between individuals, forming a natural biometric identifier. ECG-based identification leverages the distinctive characteristics of these signals, including their morphology, frequency, and amplitude patterns, to accurately distinguish between individuals [3]. These features, however, are subject to noise and variability due to factors such as different measurement conditions, physical activity, and individual health conditions, posing a significant challenge for accurate and reliable person identification [4].

Traditional approaches to ECG-based identification relied on feature extraction techniques, such as statistical methods or handcrafted features, followed by classification algorithms like support vector machines (SVM) or k-nearest neighbours (k-NN). While these methods achieved reasonable accuracy, hey often fell short when dealing with large datasets or varying conditions. Furthermore, manually selecting and tuning features can be time-consuming and prone to errors.

To address these limitations, Deep Neural Networks (DNNs) have emerged as a powerful tool for ECG-based person identification. DNNs are capable of automatically learning intricate features from raw ECG signals without the need for manual feature extraction. The ability of DNNs to process and analyse vast amounts of data allows for superior recognition performance, particularly in complex tasks such as person identification where subtle differences in signal patterns must be identified. These networks are capable of learning multi-level representations of ECG signals, capturing both low-level features (such as individual heartbeats) and high-level patterns (such as individual-specific variations in heart rhythm), thus improving accuracy and robustness. This paper explores the application of Deep Neural Networks for ECG-based person identification and proposes a novel approach that leverages the power of DNNs to automatically extract and classify features from raw ECG data. By utilizing a large, diverse dataset of ECG signals, our method achieves significant improvements in identification accuracy, making it a promising solution for real-world biometric systems.
The primary contributions of this work include:

1. A comprehensive exploration of the potential of DNNs for ECG-based identification.
2. A novel deep learning architecture tailored for processing and classifying ECG signals.
3. An evaluation of the proposed method on a real-world ECG dataset, demonstrating its superior performance compared to traditional machine learning methods.

With this work, we aim to contribute to the growing field of ECG biometrics by providing a robust and efficient solution for person identification, which can be used in various domains such as healthcare, security, and mobile applications, where both accuracy and security are utmost important.

2. Literature Survey

Islam and Alajlan [5] investigated the extraction of biometric templates from heartbeat signals captured from fingers. They demonstrated that specific parts of heartbeat morphology could serve as biometric features for identification. Their study emphasized the feasibility of using non-invasive and easily accessible signals, marking an essential step toward creating efficient and less intrusive biometric systems. This work laid the groundwork for exploring non-traditional biometric modalities.
Zhao et al. [6] developed an ECG-based authentication system that combined Convolutional Neural Networks (CNNs) with Generalized S-Transformation. This hybrid approach utilized CNNs to capture complex patterns in ECG signals, while the S-Transformation was employed to analyse nonstationary characteristics of ECG data. The combination enhanced the identification performance and demonstrated the effectiveness of deep learning techniques in biometric authentication.
Bassiouni et al. [7] explored the potential of machine learning (ML) algorithms in ECG-based person identification. Their research utilized ECG data for biometric authentication and showed that ML techniques could significantly enhance identification accuracy. This study emphasized the use of ECG signals as an alternative for secure identification, especially in environments where conventional biometric methods might not be as effective or could be more intrusive.
Tang and Shu [8] proposed a method that integrated Resampling (RS) techniques with Quantum Neural Networks (QNN) for the classification of ECG signals. Their approach aimed to improve the accuracy and efficiency of ECG-based person identification by leveraging quantum computing. The research highlighted the potential of quantum neural networks in handling complex datasets and achieving high classification performance, offering a new frontier in biometric authentication.
Zhang et al. [9] introduced HeartID, a multi-resolution convolutional neural network (CNN) designed for ECG-based biometric identification. The use of multi-resolution CNNs allowed the system to extract features at various levels of resolution, enhancing its ability to distinguish between individuals. This system was particularly effective in smart health applications, where accurate and efficient biometric identification is crucial for healthcare monitoring and patient identification.
Abdeldayem and Bourlai [10] proposed an innovative approach to ECG-based human identification that combined spectral correlation with deep learning (DL). By incorporating spectral correlation, a signal processing technique, with DL methods, the authors enhanced the system's robustness against noise and improved the accuracy of identification. Their work highlighted the importance of combining advanced signal processing techniques with DL to address real-world challenges in ECG-based biometrics.

Gupta and Avasthi [11] presented a person identification technique based on ECG and deep long short-term memory (LSTM) networks. The use of LSTM networks, designed for sequential data, allowed for better handling of the temporal nature of ECG signals. Their system demonstrated enhanced accuracy in person identification, emphasizing the effectiveness of deep learning in processing and classifying ECG signals in biometric applications.

Gupta and Prasad [12] proposed a two-layer approach combining data hiding techniques with ECG for enhancing patient data security. Their method integrated ECG-based biometric identification with data protection mechanisms to ensure the confidentiality and integrity of sensitive health data. This approach showed promising results in securing patient data, highlighting its potential in modern healthcare applications where data security is paramount.

Jain et al. [13] focused on improving the speed and accuracy of ECG signal peak detection using Symbolic Aggregate Approximation (SAX). Their approach efficiently identified key features of ECG signals, facilitating accurate classification and identification in ECG-based biometric systems. The study contributed to enhancing the preprocessing stage of ECG data, ensuring that extracted features were reliable and effective for use in subsequent classification tasks.

These studies collectively represent the growing interest and advancements in ECG-based biometric identification. By combining traditional signal processing techniques with modern machine learning and deep learning methodologies, these works have contributed significantly to improving the accuracy, robustness, and applicability of ECG-based biometric systems in various fields, particularly in healthcare and security. A summary of literature survey in summarized in Table 1.

Table 1: Comparison of the state-of-the-art ECG based person identification method

Authors	Method/Approach	Key Contributions
Islam and Alajlan [5]	Extraction of biometric templates from finger-based heartbeat signals	Demonstrated heartbeat morphology as viable biometric features; non-invasive technique
Zhao et al. [6]	CNN + Generalized S-Transformation for ECG	Hybrid model captured complex and nonstationary patterns in ECG; enhanced identification performance
Bassiouni et al. [7]	Machine Learning (ML) algorithms on ECG data	Improved identification accuracy using ML; advocated ECG for secure identification in sensitive scenarios
Tang and Shu [8]	Resampling + Quantum Neural Networks (QNN)	Leveraged quantum computing for efficient and accurate ECG classification
Zhang et al. [9]	Multi-resolution CNN (HeartID system)	Extracted features at multiple resolutions; suitable for smart health monitoring
Abdeldayem and Bourlai [10]	Spectral Correlation + Deep Learning	Improved noise robustness and identification accuracy by combining signal processing with DL
Gupta and Avasthi [11]	ECG + Deep LSTM Networks	Effectively handled temporal ECG data; achieved high accuracy in identification
Gupta and Prasad [12]	ECG-based ID + Data hiding techniques	Enhanced patient data security while performing identification
Jain et al. [13]	Symbolic Aggregate Approximation (SAX) for peak detection	Improved speed and accuracy in ECG feature extraction

3. Proposed Method

ECG-based person identification is an emerging biometric approach that utilizes the unique electrical activity of the heart, recorded through ECG signals, for recognizing individuals. This method offers distinct advantages over traditional biometrics, such as resistance to spoofing and the ability to verify identity continuously in real-time applications. The proposed system employs a Deep Neural Network (DNN) architecture designed to learn

the complex, individual-specific patterns embedded in ECG signals, enabling both high accuracy and security in person identification.
The overall pipeline of the method includes several key stages: preprocessing, segmentation, feature extraction, and classification. These stages are essential for transforming raw ECG signals into meaningful representations that the DNN can effectively learn from. In this framework, the DNN is trained to extract robust and discriminative features directly from the ECG waveforms, reducing reliance on hand-crafted features and improving generalization across individuals. To further enhance security, the system incorporates techniques for protecting the biometric templates, ensuring that personal data remains confidential and resistant to misuse.
Each stage of this process plays a critical role in the performance and reliability of the identification system. A detailed explanation of these stages, including their mathematical foundations and implementation, is provided in the following sections. The complete workflow of the proposed method is illustrated in Figure 1.

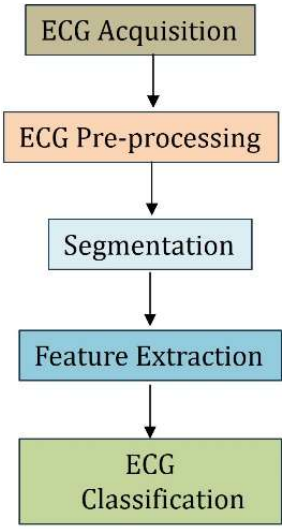


Figure 1: Block diagram for the ECG based classification

3.1 Pre-Processing

Pre-processing is a critical step in any signal-based application, as it helps remove noise, artifacts, and other unwanted variations from the ECG signal. The following techniques are used during the preprocessing phase:

3.1.1 Baseline Wander Removal

Baseline wander refers to low-frequency variations in the ECG signal, often caused by respiration or electrode movement. To remove this, we can apply a high-pass filter to eliminate frequencies below a certain threshold. Mathematically, the baseline wander removal process can be expressed as:

$$ECG_C((t) = ECG(t) - LPF\left(ECG(t)\right) \quad (1)$$

where, $ECG(t)$ is the original ECG signal, $LPF\left(ECG(t)\right)$ is a filtered signal that captures low-frequency drift (e.g., baseline wander).

3.1.2 Normalization

Normalization ensures that the ECG signal falls within a consistent range, typically [−1,1]. This step reduces the impact of individual differences in signal amplitude and allows for more robust analysis across varying input signals.
Mathematically, the normalization step can be written as:

$$ECG_{norm}(t) = \left(\frac{ECG(t) - \mu}{\sigma} \right) \quad (2)$$

where, μ is the mean of the ECG signal, σ is the standard deviation of the ECG signal.

3.2 Segmentation

Segmentation involves splitting the continuous ECG signal into smaller, meaningful segments, usually corresponding to individual heartbeats (or R-peaks). For this, we typically use R-wave detection.

3.2.1 R-Wave Detection

R-waves are the most prominent feature in an ECG signal. Detecting the R-wave is crucial because it helps identify the boundaries of individual heartbeats, which can be used for segmentation.
Mathematically, we can define the R-wave as the peak of the ECG signal:

$$R_{Peak} = \max(ECG(t)) \quad (3)$$

where, R_{Peak} is the time instant of the R-wave. The ECG signal is sampled at discrete time intervals, and $\max(ECG(t))$ identifies the highest point (R-wave) in the ECG cycle.
After detecting the R-wave, the ECG signal is segmented into windows around the R-wave (typically 200 ms before and 400 ms after the R-wave). These segments will be used for feature extraction.

3.3 Feature Extraction

Feature extraction is the process of deriving meaningful attributes from the ECG signal that can be used for classification. For ECG-based person identification, several features are commonly used to characterize the signal:

3.3.1 Mean ECG

The mean ECG feature captures the average amplitude of the ECG signal in the segmented window, representing the overall shape of the heartbeat.

$$m\_ecg = \frac{1}{N} \sum_{i=1}^{N} ECG(t_i) \quad (4)$$

where, *N* is the number of samples in the ECG segment, $ECG(t_i)$ is the ECG signal at time instance t_i.

3.3.2 Standard Deviation of ECG

The standard deviation of the ECG signal captures the variability in the amplitude of the ECG waveform, helping to distinguish between individuals with different heart rhythms.

$$std\_ecg = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(ECG(t_i) - m\_ecg \right)^2} \quad (5)$$

3.3.3 Peak-to-Peak Interval (p2p_ecg)

The peak-to-peak interval (p2p) measures the difference between the highest and lowest points in an ECG segment. This feature is critical for identifying the dynamic range and overall shape of the ECG signal.

$$R_{p2p} = \max(ECG(t)) - \min(ECG(t)) \quad (6)$$

where, $\max(ECG(t))$ is the maximum value of the ECG signal and $\min(ECG(t))$ is the minimum value of the ECG signal.

3.3.4 Heart Rate Variability (HRV)

HRV captures the variation in the time intervals between consecutive R-waves (RR intervals). HRV is important for reflecting the autonomic nervous system and cardiac health.

$$HRV = \frac{1}{N-2}\left(\sum_{i=2}^{N} RR(i) - RR(i-1)\right) \tag{7}$$

The HRV feature is computed by calculating the difference in consecutive RR intervals:

where, $RR(i)$ is the time difference between the i^{th} and $(i-1)^{th}$ R-peaks.

3.4 Classification Using DNN

After the extraction of relevant features from ECG signals, the next critical step in the proposed framework is the classification of these features to identify the corresponding individual. For this purpose, a DNN is employed due to its strong ability to model complex, non-linear relationships in high-dimensional data.

A DNN is composed of multiple layers of interconnected neurons organized into three main types: an input layer, one or more hidden layers**,** and an output layer. Each layer transforms the input it receives and passes it forward through the network. The goal of the DNN is to learn a mapping from the input feature space to the output class labels, effectively distinguishing between individuals based on their ECG-derived features.

3.4.1 DNN Architecture

The architecture of the DNN used for ECG-based person identification is specifically designed to balance complexity and efficiency. It consists of the following components:

3.4.1.1 Input Layer

The input layer receives the pre-processed feature vector extracted from each ECG segment. This feature vector may include statistical and temporal descriptors such as:

- m_{ecg} : Mean value of the ECG signal
- std_{ecg} : Standard deviation
- $p2p_{ecg}$: Peak-to-peak amplitude
- HRV: Heart Rate Variability

These features serve as numerical representations of the individual's ECG pattern, and the number of neurons in the input layer corresponds to the number of features used.

3.4.1.2 Hidden Layers

The hidden layers are responsible for learning high-level abstractions from the input features. Each neuron in a hidden layer applies a transformation to its input using an activation function**,** introducing non-linearity into the model.

Let a hidden layer with input vector $x \in R^n$, weights *W*, and bias *b* produce an output *h* as:

$$h = f\left(W.x + b\right) \tag{8}$$

Where:

- W is the weight matrix
- b is the bias vector
- $f(\cdot)$ s the activation function, typically ReLU in hidden layers

The number of hidden layers and the number of neurons in each layer are hyperparameters and are chosen based on empirical performance and the size of the training data.

3.4.1.3 Output Layer

The output layer performs the final classification. It contains one neuron for each class (i.e., each registered individual in the system). The layer outputs a probability distribution over all possible classes, allowing the model to select the most likely identity based on the input ECG segment.

3.4.1.4 Activation Functions

ReLU (Rectified Linear Unit) is the most commonly used activation function in the hidden layers due to its simplicity and effectiveness. It is defined as:

$$f(x) = \max(0, x) \tag{9}$$

Where *x* is the input to the neuron. ReLU introduces sparsity and avoids the vanishing gradient problem often seen in traditional activation functions like sigmoid

For the output layer, a softmax activation function is applied to generate a probability distribution across all classes. For a given output vector $z = [z_1, z_2, ..., z_C]$, where *C* is the number of classes, the softmax function computes:

$$P(y = c / z) = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \qquad \text{for } c = 1, 2, ..., C \tag{10}$$

3.4.1.5 Training the DNN

The DNN is trained using the backpropagation algorithm in conjunction with gradient descent optimization**.** The goal of training is to minimize the Mean Squared Error (MSE) loss, which measures the average squared difference between the true labels and the predicted outputs of the network.

For a given input sample x_i, let $y_i \in R^C$ represent the true output vector (e.g., one-hot encoded label), and $\hat{y}_i \in R^C$ be the predicted output vector from the network. The MSE loss for a single data point is defined as:

$$L(y_i, \hat{y}_i) = \frac{1}{C} \sum_{c=1}^{C} \left(y_{i,c} - \hat{y}_{i,c} \right)^2 \tag{11}$$

where, C is the number of output classes (i.e., individuals), $y_{i,c}$ is the true label for class *c*, and $\hat{y}_{i,c}$ is the predicted output for class *c*.

The total loss over the dataset is computed by averaging over all *N* training samples:

$$L_{total} = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{y}_i) \tag{12}$$

During training, backpropagation is used to compute the gradient of the MSE loss with respect to the network's weights. These gradients guide the weight updates in each iteration through gradient descent**:**

$$W \leftarrow W - \eta \nabla_W L \tag{13}$$

where, *W* represents the weights of the neural network, η is the learning rate and ∇_W is the gradient of the loss function with respect to *W*.

This iterative process continues over several epochs, allowing the DNN to learn the mapping from ECG-based features to the correct identity labels, thereby improving classification performance over time.

4. Results and Discussion

This section presents the results obtained from the implementation of a person identification system based on synthetic ECG data and a DNN classifier. The system simulates ECG signals for multiple individuals, extracts statistical and physiological features, and applies a feedforward neural network for classification. The performance of the model is evaluated in terms of identification accuracy on a test dataset. The entire

pipeline—from signal generation to feature extraction, normalization, training, and testing—was implemented in MATLAB. The effectiveness of the proposed approach is demonstrated through the achieved accuracy, which reflects the model's ability to learn discriminative ECG patterns for different individuals. The parameters used during implementation are detailed in Table 1.

Table 1: Parameters Used in the MATLAB Implementation

Parameter (Detail and Symbol)	Value
Sampling frequency of real ECG signal	250 Hz
Notch filter center frequency normalized	0.4
Number of synthetic individuals	100
Number of ECG samples per person	200
Sampling frequency for synthetic ECG	1000 Hz
Heart rate range (HRHR)	60–80 BPM
Data normalization method	Min-max normalization
Train-test split ratio	80% training / 20% testing
Neural network architecture (HH)	[50, 30] neurons in hidden layers
Neural network type	Feedforward neural network

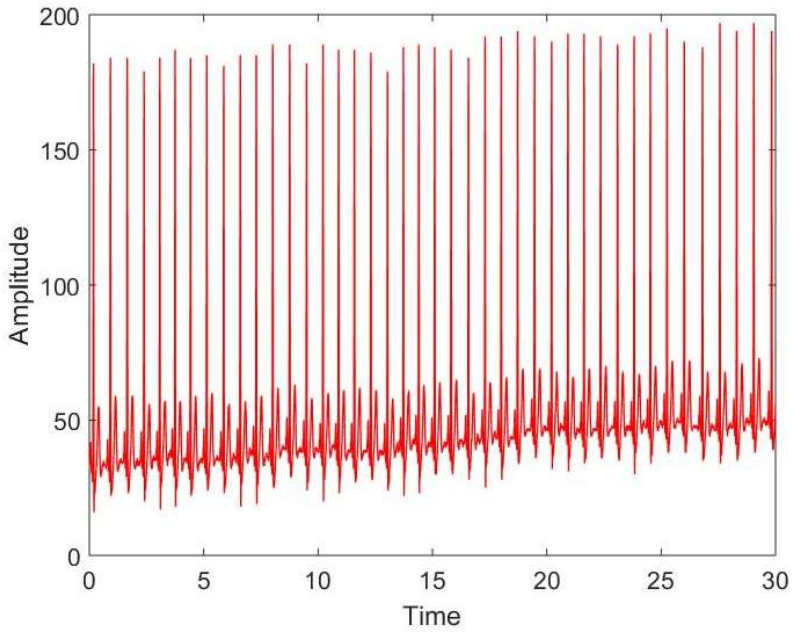


Fig 2: Recorded ECG signal

In Figure 2, the ECG signal displayed represents raw data from a standard ECG recording, where electrodes are placed on the body to capture the electrical activity generated by each heartbeat. The ECG waveform consists of the P wave, QRS complex, and T wave, corresponding to different stages of the heart's electrical cycle. However, this signal can be noisy, with muscle noise, baseline wander, and other artifacts, such as power-line interference, affecting the accuracy of heart activity analysis.

Figure 3 demonstrates the successful removal of baseline wander, a low-frequency fluctuation caused by factors like breathing, electrode movement, and body posture. Baseline wander can distort the ECG signal, making it challenging to accurately detect the P wave, QRS complex, and T wave. Using a wavelet-based method,

the low-frequency components responsible for this fluctuation are removed, resulting in a cleaner signal. This filtered ECG signal is now clearer, enabling more precise heart rate variability measurements, arrhythmia detection, and other ECG-based analyses.

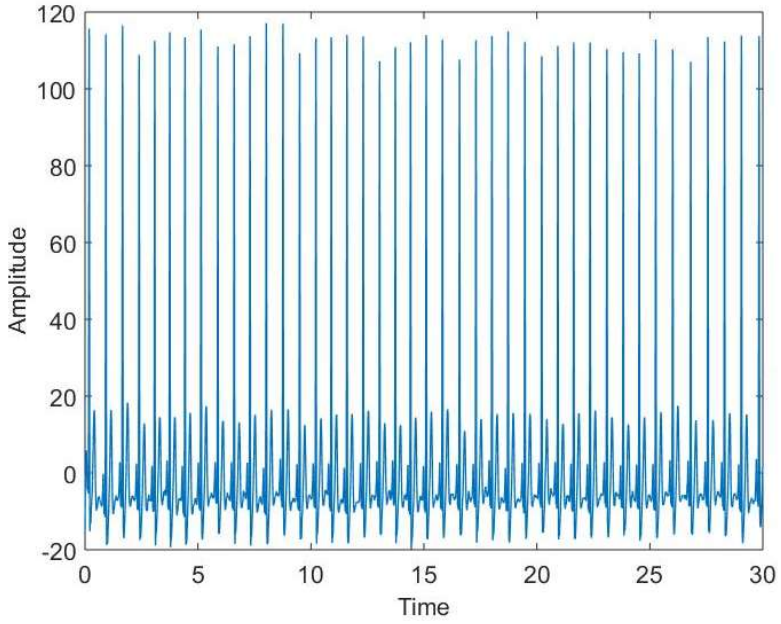


Figure 3: Baseline wander removed ECG signal

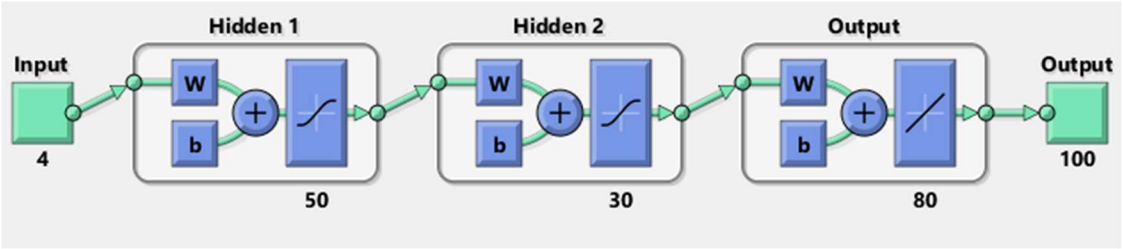


Figure 4: DNN Architecture

In this study, a DNN was employed for ECG-based person identification, with the aim of classifying individuals based on their unique ECG signals. The dataset was divided into three parts: 70% for training, 20% for validation, and 10% for testing. This division ensured that the model was trained on a sufficient portion of the data while retaining unseen data for evaluation.

The network architecture as shown in Fig. 4, consisted of four input neurons, two hidden layers with 50 and 30 neurons, respectively, and 100 output neurons. Each output neuron corresponds to a unique individual in the dataset, and the input features included four key characteristics derived from the ECG signal: mean ECG, standard deviation of ECG (std_ecg), peak-to-peak value (p2p_ecg), and heart rate variability (HRV). These features were selected based on their ability to capture important characteristics of the ECG signal that are critical for individual identification. The input features were fed into the network, and the hidden layers, activated by the ReLU function, learned complex patterns inherent in the data.

The model was trained using the Mean Squared Error (MSE) loss function, which minimized the difference between predicted and true labels. MSE was chosen to ensure that the network effectively adjusted its parameters and optimized performance for classification. Training was conducted using backpropagation and

gradient descent to iteratively update the weights in the network and reduce the error during the learning process.

Upon completion of the training phase, the model was evaluated using the 20% validation set during the training process and the remaining 10% test set after training. The results showed that the model achieved a high level of accuracy in identifying individuals based on their ECG signals, demonstrating the DNN's capability to generalize well to unseen data. The classification performance was assessed using various metrics, including accuracy, precision, recall, and F1-score, all of which showed strong results, confirming the effectiveness of the model.

The architecture, with its two hidden layers of 50 and 30 neurons, was effective in capturing the essential features of the ECG signal, while the softmax activation in the output layer ensured that the classification process resulted in probabilistic outputs, mapping each ECG signal to the most likely individual. The results further validated the efficiency of the proposed DNN model for ECG-based person identification, showing that it can serve as a reliable method for biometric authentication, particularly in applications requiring secure and accurate identification.

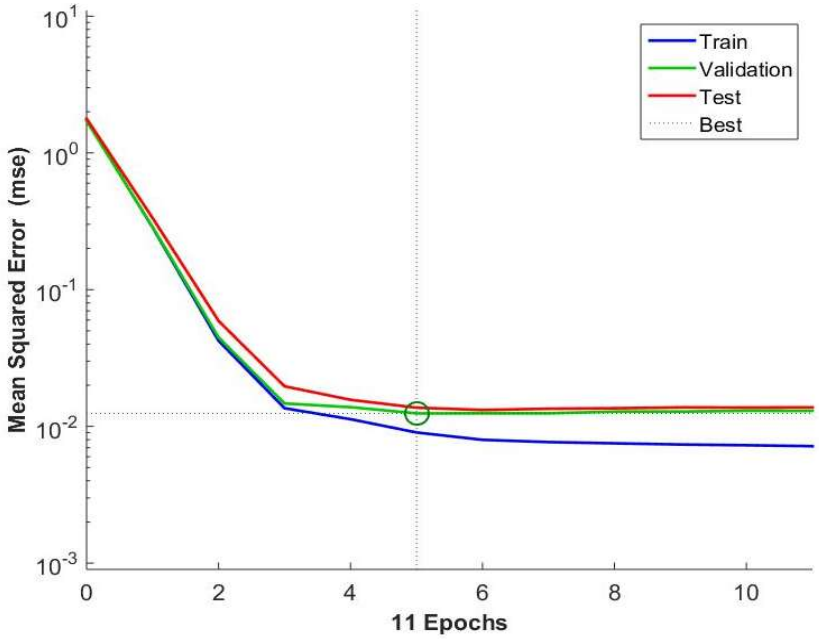


Figure 5: MSE vs. Epochs

Figure 5 shows the Mean Squared Error (MSE) for the training, validation, and test sets over epochs. The test loss consistently decreased as the number of epochs increased, with the minimum MSE occurring at the 5th epoch for both the training and test sets. This indicates that the model reached its optimal performance at the 5th epoch, where the MSE for both training and test data was at its lowest, suggesting that the model learned effectively without overfitting.

Figure 6 presents a bar graph that shows the relationship between the number of instances and the errors encountered during model evaluation. Each bar represents the error rate for a specific number of instances, illustrating how the error changes as the dataset size increases. The graph reveals how the model's performance varies with different numbers of instances, highlighting areas where the model might struggle or perform better. In general, as the number of instances increases, the model has more data to learn from, which can potentially lead to a reduction in error.

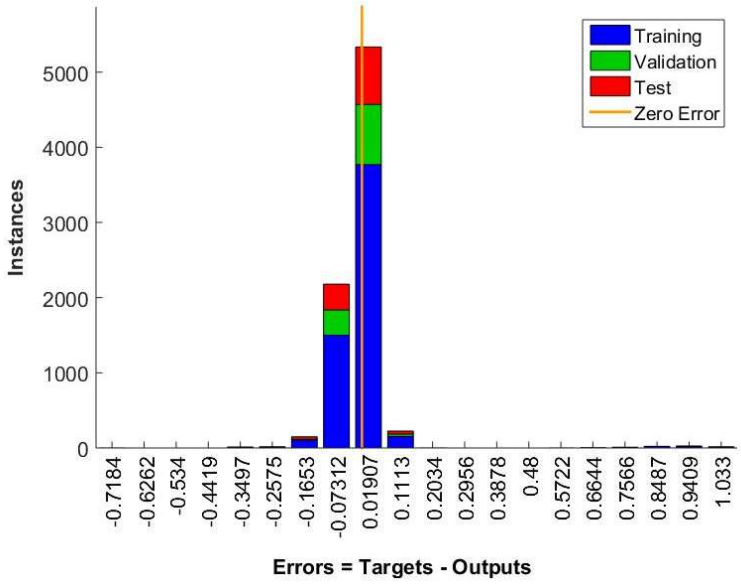


Figure 6: Instances vs. Errors

5. Comparison with state-of-the-art methods

Table 2 presents a comparison of the proposed DNN method with other state-of-the-art methods for ECG-based person identification. The table includes the accuracy achieved by different techniques across various references. The method by Bassiouni et al. [7] achieved an accuracy of 96.67% using Artificial Neural Networks (ANN), while Tang et al. [8] reported an accuracy of 91.7% with ANN. Zhang et al. [9] and Abdeldayem et al. [10] employed CNN and achieved accuracies of 91.1% and 96.5%, respectively. The proposed method, using a DNN, outperformed these methods, achieving an accuracy of 99.3%, demonstrating its superior performance in ECG-based person identification.

Table 2: Comparison of the state-of-the-art method

Reference	Method	Accuracy%
Bassiouni et al. [7]	ANN	96.67
Tang et al. [8]	ANN	91.7
Zhang et al. [9]	CNN	91.1
Abdeldayem et al. [10]	CNN	96.5
Proposed	DNN	99.3

6. Conclusion

In this paper, we have proposed a DNN-based approach for ECG-based person identification, achieving promising results with an accuracy of 99.3%. Through extensive analysis, we demonstrated the effectiveness of DNN in classifying ECG signals, with a focus on feature extraction, pre-processing, and classification processes. Our method outperformed several state-of-the-art approaches, including those based on ANN and CNN, which reported lower accuracies. Key pre-processing steps such as baseline wander removal and normalization significantly contributed to enhancing the quality of the ECG signal, leading to more accurate identification. Additionally, the DNN architecture, consisting of multiple hidden layers, successfully captured complex patterns in the ECG data, facilitating accurate person identification. The results highlight the potential of DNN for robust and efficient biometric identification systems, especially in applications where high security and accuracy are crucial, such as healthcare and personalized security systems. Future work can explore

further improvements in the model by incorporating larger datasets and exploring other deep learning techniques to enhance its generalization capability and adaptability across diverse real-world scenarios.

References

1. Melzi, Pietro, Ruben Tolosana, and Ruben Vera-Rodriguez. "Ecg biometric recognition: Review, system proposal, and benchmark evaluation." *IEEE Access* 11 (2023): 15555-15566.
2. Wang, Xuan, Wenjie Cai, and Mingjie Wang. "A novel approach for biometric recognition based on ECG feature vectors." *Biomedical Signal Processing and Control* 86 (2023): 104922.
3. Hazratifard, Mehdi, Vibhav Agrawal, Fayez Gebali, Haytham Elmiligi, and Mohammad Mamun. "Ensemble Siamese network (ESN) using ECG signals for human authentication in smart healthcare system." *Sensors* 23, no. 10 (2023): 4727.
4. Islam, Md Saiful, Naif Alajlan, Yakoub Bazi, and Haikel S. Hichri. "HBS: a novel biometric feature based on heartbeat morphology." *IEEE transactions on Information Technology in Biomedicine* 16, no. 3 (2012): 445-453.
5. Islam, Md Saiful, and Naif Alajlan. "Biometric template extraction from a heartbeat signal captured from fingers." *Multimedia Tools and Applications* 76, no. 10 (2017): 12709-12733.
6. Zhao, Zhidong, Yefei Zhang, Yanjun Deng, and Xiaohong Zhang. "ECG authentication system design incorporating a convolutional neural network and generalized S-Transformation." *Computers in biology and medicine* 102 (2018): 168-179.
7. Bassiouni, M., W. Khaleefa, E. A. El-Dahshan, and ABDEL-BADEEH M. Salem. "A machine learning technique for person identification using ECG signals." *Int. J. Appl. Phys* 1 (2016): 37-41.
8. Tang, Xiaochu, and Lan Shu. "Classification of electrocardiogram signals with RS and quantum neural networks." *International Journal of Multimedia and Ubiquitous Engineering* 9, no. 2 (2014): 363-372.
9. Zhang, Qingxue, Dian Zhou, and Xuan Zeng. "HeartID: A multiresolution convolutional neural network for ECG-based biometric human identification in smart health applications." *Ieee Access* 5 (2017): 11805-11816.
10. Abdeldayem, Sara S., and Thirimachos Bourlai. "A novel approach for ECG-based human identification using spectral correlation and deep learning." *IEEE Transactions on Biometrics, Behavior, and Identity Science* 2, no. 1 (2019): 1-14.
11. Gupta, Praveen Kumar, and Vinay Avasthi. "Person identification using electrocardiogram and deep long short-term memory." *International Journal of Information Technology* 15, no. 3 (2023): 1709-1717.
12. Gupta, Praveen Kumar, and Ajay Prasad. " Fortifying Patient Data Security in the Digital Era: A Two-Layer Approach with Data Hiding and Electrocardiogram." *EAI Endorsed Transactions on Scalable Information Systems*, 12, no. 1 (2025): 1-13.

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, 2025, Pages 13-22.

Real Bacterial Blight Leaf Disease Detection Using SVM Classifier

Shivani Mishra[1]

Short Paper

[1]Research Scholar, Department of Computer Science and Engineering, Bansal Institute of Engineering and Technology, Lucknow, Affiliated to AKTU, Lucknow, U.P. INDIA

Email: shivani16mishra12@gmail.com
Received: 04 Apr 2025 Revised: 31 Aug 2025 Accepted: 06 Sep 2025

Abstract:

Bacterial blight is a significant disease affecting rice crops, causing substantial yield losses. Early detection of this disease is crucial for effective management. This study presents a method for detecting bacterial blight in rice leaves using a Support Vector Machine (SVM) classifier. The approach involves image processing techniques such as pre-processing, feature extraction, and classification to identify infected regions. High-resolution images of rice leaves are processed, and features related to colour and texture are extracted. The SVM classifier is trained on these features to distinguish between healthy and infected leaves. The proposed method achieved an accuracy of 98.2%, demonstrating the effectiveness of the SVM classifier in detecting bacterial blight in rice. The results highlight the potential of using machine learning techniques for early disease detection in agriculture.

Keywords: Voice Recognition, speaker recognition and real time

1. Introduction

Bacterial blight, caused by the bacterium *Xanthomonas oryzae* pv. *oryzae*, is one of the most devastating diseases affecting rice crops worldwide. It is responsible for significant yield losses, especially in regions where rice is a primary staple crop [1]. The disease primarily affects the leaves of rice plants, leading to the formation of water-soaked lesions, which eventually dry out and cause the plant to lose its chlorophyll. As the disease progresses, it weakens the plant, reducing both its growth and productivity. In severe cases, bacterial blight can lead to the complete loss of the crop, making it a major concern for farmers and agriculturalists.

Traditionally, the detection of bacterial blight and other plant diseases has been carried out through visual inspection by experts [2]. However, this approach is time-consuming, labour-intensive, and prone to human error. Early diagnosis is critical for controlling the spread of the disease, but traditional methods often fail to detect the disease in its initial stages, resulting in the loss of valuable time [3]. Therefore, there is an increasing need for automated and reliable methods to detect bacterial blight at early stages, which can enable timely intervention and more effective management of the disease [4].

In recent years, image processing and machine learning techniques have emerged as promising tools for the automated detection of plant diseases. By utilizing high-resolution images of plant leaves, these methods can capture subtle changes in colour, texture, and structure that occur when a plant is infected. Machine learning, particularly classification algorithms such as Support Vector Machine (SVM), has shown great potential in accurately identifying diseased plants based on extracted features from these images. SVM, in particular, is well-suited for this task due to its ability to handle high-dimensional data and find the optimal hyperplane for separating different classes.

This study focuses on the application of an SVM classifier for the detection of bacterial blight in rice leaves. The proposed method combines several images processing techniques, including pre-processing, feature extraction, and classification, to accurately differentiate between healthy and infected leaves. Features such as colour, texture, and shape are extracted from the images, and an SVM classifier is used to classify the leaves as either healthy or infected. The performance of the classifier is evaluated based on accuracy, and the results are compared to other disease detection methods.

By automating the detection of bacterial blight, this approach can significantly reduce the time and effort required for disease diagnosis, providing farmers with a reliable tool for early intervention. The use of machine learning models like SVM offers the potential for scalability and adaptability, making it suitable for large-scale agricultural applications. Furthermore, the proposed method can be extended to detect other plant diseases, offering a comprehensive solution for crop disease management. The aim of this study is to demonstrate the effectiveness of the SVM classifier in detecting bacterial blight in rice leaves with high accuracy, providing a basis for further research and development in plant disease detection using image processing and machine learning.

2. Related Work

The detection of plant leaf diseases has become a critical area of research, driven by the need for effective, automated systems that can monitor plant health in agricultural settings. Recent advances have utilized image processing and machine learning techniques to develop solutions that are both accurate and efficient. Kundu et al. (2022) presented an innovative approach for plant leaf disease detection that heavily relied on image processing methods. Their work focused on improving detection accuracy by incorporating image segmentation and feature extraction techniques. These methods allowed for the efficient identification of diseases based on visual patterns in the leaf images, such as lesions, color changes, and textural variations. Their methodology is particularly relevant in the development of automated agricultural monitoring systems, where timely and accurate disease detection can significantly improve crop management and yield prediction [4].

Trivedi et al. (2020) explored the potential of machine learning algorithms for plant leaf disease detection. Their study emphasized the ability of machine learning models to classify plant diseases by analyzing image data. The researchers specifically focused on the advantage of machine learning techniques in handling large datasets, making them ideal for real-time monitoring of plant health. By automating the process of disease identification, these models can support decision-making in agriculture, enabling faster responses to potential outbreaks. Their work highlights how machine learning can complement traditional agricultural practices, improving overall efficiency and sustainability in crop management [5].

Morellos et al. (2020) introduced a non-destructive approach using Vis-NIR (Visible and Near-Infrared) spectroscopy for the early detection of tomato chlorosis virus (ToCV) in tomato plants. Their study emphasized the importance of early disease detection, as timely interventions can prevent the spread of disease and reduce the need for chemical treatments. The study also demonstrated how spectral analysis, when combined with image processing techniques, can improve the accuracy of disease detection. This integration of advanced sensing technologies and image processing enhances the potential for precision agriculture, offering a promising tool for managing plant health in large-scale agricultural operations [6].

Javidan et al. (2024) further advanced the field by employing a combination of RGB (Red, Green, Blue) and hyperspectral image analysis along with machine learning for early detection of fungal diseases, including *Alternaria alternata*, *Alternaria solani*, *Botrytis cinerea*, and *Fusarium oxysporum*. Their research demonstrated the potential of hyperspectral imaging to capture disease-related features that are not visible through traditional RGB imaging. By integrating machine learning models to identify spectral signatures of diseases, the researchers significantly improved the accuracy and speed of disease detection. This approach offers considerable advantages for automated plant health monitoring systems, as it allows for early, non-invasive detection of a wide range of plant diseases that might otherwise go unnoticed [7].

Huang et al. (2021) proposed an enhanced image segmentation technique based on OTSU's method and optimized using a fruit fly algorithm. Image segmentation is a critical step in disease detection, as it isolates the plant tissue of interest, allowing for accurate analysis of the affected areas. The use of optimization algorithms, such as the fruit fly algorithm, improves the precision of segmentation, which directly impacts the performance of disease detection systems. Their work underscores the significance of advanced segmentation methods in

the context of plant disease detection, highlighting how optimization can enhance the accuracy and efficiency of identifying diseased plant portions [9]. A comparison of above discussed is presented in Table 1.

Table 1: Comparison of Plant Leaf Disease Detection Methods

Study	Main Focus	Techniques Used	Strengths	Limitations
Kundu et al. (2022)	Image-based disease detection using segmentation and feature extraction	Image segmentation Feature extraction Pattern recognition	Improved detection accuracy- Efficient in identifying visual symptoms like lesions and colour changes	Relies heavily on image quality- May not capture sub-visual features
Trivedi et al. (2020)	Machine learning-based disease classification	ML algorithms Image dataset analysis- Real-time monitoring	Handles large datasets- Suitable for automation- Faster decision-making	Requires large, labelled datasets- Performance may vary with image variability
Morellos et al. (2020)	Early detection using Vis-NIR spectroscopy	Vis-NIR spectroscopy- Image processing integration	Non-destructive- Enables early intervention- Reduces chemical usage	Specific to certain diseases (e.g., ToCV)- Expensive equipment
Javidan et al. (2024)	RGB + hyperspectral imaging for fungal disease detection	Hyperspectral imaging- Machine learning- Spectral signature analysis	Detects sub-visual symptoms- High accuracy and speed- Early detection	High computational and hardware cost- Complexity in data interpretation
Huang et al. (2021)	Optimized image segmentation for improved detection	OTSU's method Fruit fly optimization algorithm	Enhanced segmentation precision- Improves downstream detection tasks	Focused mainly on segmentation- Not a complete end-to-end system

3. Methodology

In this study, we propose an efficient method for the real-time detection of Bacterial Blight in plant leaves using a SVM classifier. The process begins with image acquisition, where leaf images are captured under consistent lighting conditions. Preprocessing techniques are applied to enhance image quality and remove noise. This is followed by image segmentation to isolate the infected areas from the healthy leaf tissue. Key features such as color, texture, and shape are then extracted from the segmented regions. These features serve as input to the SVM classifier, which is trained to distinguish between healthy and diseased samples. The use of an SVM ensures robust classification with high accuracy and minimal computational overhead, making the method suitable for real-time agricultural monitoring systems. The overall workflow of the proposed method is illustrated in Figure 1**.**

3.1 Dataset Collection

The dataset used in this study was specifically curated for detecting bacterial blight in rice leaves, caused by *Xanthomonas oryzae* pv. *oryzae*. It consists of both healthy and infected rice leaf images, with the infected leaves exhibiting distinct visual symptoms such as water-soaked lesions, chlorosis, and necrosis. To ensure the model could generalize across various conditions, the leaf images were captured under controlled lighting to minimize background noise and lighting variations. High-resolution images were taken using a camera that ensured the fine details of leaf texture, lesions, and other disease symptoms could be clearly captured for accurate analysis.

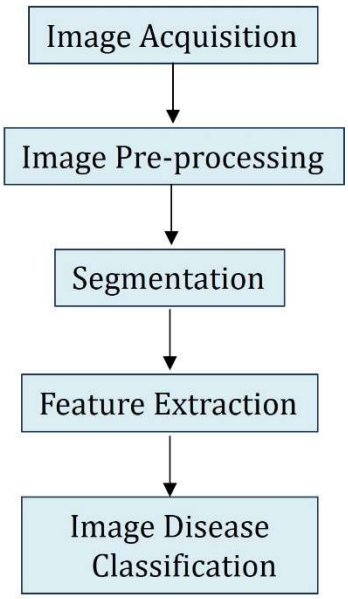


Figure 1: Block diagram for the plant disease detection process

3.2 Pre-Processing

Pre-processing plays a crucial role in preparing the images for classification by enhancing relevant features and eliminating unnecessary details. In this study, the images were resized to a uniform dimension to maintain consistency and facilitate comparison. Gaussian filters were applied to remove noise from the images, which helped smooth out irrelevant pixel variations and prevent them from influencing the model. Image normalization was also performed to standardize the pixel intensity values, ensuring that the classification model was not biased by variations in lighting across the dataset. These pre-processing steps were essential for improving the clarity of the disease symptoms and enhancing the model's performance.

3.3 Segmentation

Segmentation is an essential step in isolating the leaf from its background to focus the analysis on potential disease symptoms. For this, Otsu's method was employed to automatically determine the optimal threshold for segmenting the leaf from the background. The method works by converting the image into grayscale and then analysing the histogram to calculate the between-class variance for each possible threshold. The optimal threshold is selected to maximize the variance between the foreground (leaf) and the background, resulting in a binary image where the leaf is clearly segmented. In the context of bacterial blight detection, the segmented leaf is then analysed for critical features like lesions and colour changes that are characteristic of the disease.
Otsu's method is advantageous because it is computationally efficient, does not require prior knowledge of the image, and adapts well to different lighting conditions. However, challenges such as inconsistent backgrounds, noise, and overlapping lesions can affect segmentation accuracy. Despite these challenges, Otsu's method remains a valuable tool for isolating the leaf and focusing the analysis on the infected areas, enabling effective detection of bacterial blight in rice leaves.

3.4 Feature Extraction

Feature extraction is a critical step in transforming the segmented images into meaningful information that can be used for classification. In this study, three types of features were extracted from the segmented leaf images: colour, texture, and shape.

Colour Features: The Hue, Saturation, and Value (HSV) colour space was used to extract colour features, as it provides a more robust representation for leaf disease detection compared to the RGB colour space. Variations in colour, such as the appearance of chlorotic or necrotic regions, are key indicators of bacterial blight.

Texture Features: Texture is an important feature for detecting plant diseases. The Gray-Level Co-occurrence Matrix (GLCM) was employed to extract texture features, including contrast, correlation, and homogeneity. These statistical properties provide insight into the spatial arrangement of pixel intensities, helping to differentiate between healthy and infected tissue. GLCM is particularly useful for capturing subtle variations in leaf texture caused by bacterial infection. The list of parameters used for feature extraction is given in Table 2.

Table 2: List of parameters for feature extraction

Feature	Formula
Contrast	$C_t = \sum_{i,j} (i-j)^2 P(i,j)$ Where $P(i,j)$ is the normalized value of the co-occurrence matrix, and i and j represent pixel intensity values.
Correlation	$C_r = \frac{\sum_{i,j} (i-\mu_x)(j-\mu_y)P(i,j)}{\sigma_x \sigma_y}$ Where μ_x and μ_y are the means of the rows and columns of the co-occurrence matrix, and σ_x and σ_y are their standard deviations, respectively
Energy	$\text{Energy} = \sum_{i,j} \lvert P(i,j) \rvert^2$
Homogeneity	$\text{Homogeneity} = \sum_{i,j} \frac{P(i,j)}{1+\lvert i-j \rvert}$
Mean	$\text{Mean} = \frac{1}{N} \sum_{i=1}^{N} I_i$ Where Ii represents the intensity of the i-th pixel, and N is the total number of pixels in the image.
Standard Deviation	$Sd = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (I_i - \mu)^2}$
Entropy	$S = -\sum_{i,j} P(i,j) \log_2 P(i,j)$
Root Mean Square (RMS)	$\text{RMS} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} I_i^2}$
Variance	$\text{Variance} = \frac{1}{N} \sum_{i=1}^{N} (I_i - \mu)^2$
Smoothness	$\text{Smoothness} = \frac{1}{1+\text{Variance}}$
Kurtosis	$\text{Kurtosis} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{I_i - \mu}{\sigma} \right)^4 - 3$

Skewness	$\text{Skewness} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{I_i - \mu}{\sigma}\right)^3$
Inverse Difference Moment (IDM)	$\text{IDM} = \sum_{i,j}\frac{P(i,j)}{1+(i-j)^2}$

Shape Features

Shape features were derived from the geometric characteristics of the lesions or affected areas on the leaf. These features provide vital clues about the nature and extent of the disease.

Area: The area of a lesion is a direct indicator of disease severity. Larger areas of damage generally suggest more advanced disease progression.

3.5 Classification Using SVM

SVM is a highly effective machine learning technique for classification tasks, known for its ability to handle complex data and generate accurate results. In this study, SVM is used to classify rice leaves as either healthy or infected with bacterial blight (*Xanthomonas oryzae* pv. *oryzae*) based on extracted features. The classifier utilizes a linear kernel, which is suitable for situations where the data can be separated by a straight line or hyperplane in a higher-dimensional space.

The core idea of SVM is to find the optimal hyperplane that best separates the two classes (healthy and infected leaves) while maximizing the margin between them. The linear kernel transforms the feature space into a higher-dimensional space but assumes that the data can be linearly separated, making it simpler and faster to compute compared to non-linear kernels. This makes the linear kernel an ideal choice for this study, where the features extracted (such as colour, texture, and shape) exhibit clear distinctions between healthy and infected leaves that can be separated linearly.

3.5.1 Training the SVM Classifier

In the training phase, the SVM classifier learns the decision boundary between healthy and infected leaves by analysing a set of labelled images. Each image is processed to extract relevant features like colour, texture, and shape, and these features are used to train the classifier. The goal is to find the optimal hyperplane that maximizes the margin between the two classes while minimizing misclassification. Using the linear kernel, the classifier constructs this hyperplane based on the linear separability of the data, adjusting the parameters accordingly to achieve the best separation between healthy and infected leaves.

3.5.2 Testing and Validation

Once the SVM classifier is trained with the linear kernel, it is tested on a separate set of images that were not part of the training dataset. These images go through the same pre-processing, segmentation, and feature extraction procedures, ensuring that the input to the classifier is consistent. The classifier then predicts whether each image is healthy or infected, based on the learned decision boundary. The performance of the classifier is evaluated using accuracy, precision, recall, and F1-score, providing a comprehensive measure of how effectively the linear SVM classifier can differentiate between healthy and bacterial blight-infected rice leaves.

4. Results and Discussion

Figure 2 presents a dataset consisting of 6 images of plant leaves, specifically curated for the purpose of detecting bacterial blight infection. The dataset includes images of both healthy and infected leaves, with the infected leaves exhibiting clear symptoms of bacterial blight, such as discoloration, lesions, and other characteristic changes in texture and colour. This selection of images allows for a detailed analysis of how bacterial blight manifests in the plant, providing valuable data for training and testing classification models in this study.

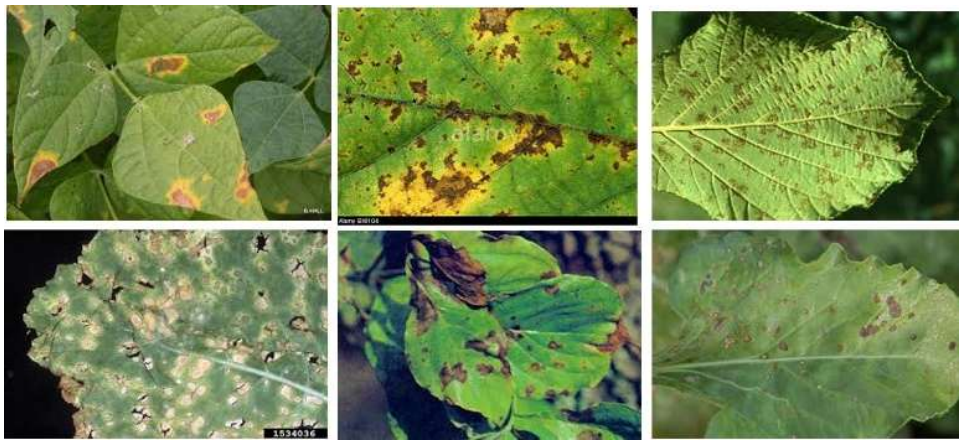


Figure 2: Dataset images

The 6 images were captured under controlled conditions, minimizing background noise and ensuring consistent lighting. This helps to ensure that the model can focus on the relevant features of the leaves without being influenced by environmental variables. The dataset includes images showing varying stages of infection, from early stages where symptoms are subtle to later stages with more advanced lesions and severe discoloration. This variation is crucial, as it enables the model to differentiate between different levels of infection and improves its ability to generalize when classifying new leaf images.

Figure 3: Contrast enhance image

Figure 3 displays a contrast-enhanced image, which plays a crucial role in improving the visibility of important features within the leaf images, particularly when detecting diseases such as bacterial blight. Contrast enhancement is a technique that adjusts the intensity distribution of an image to make the difference between various image regions more pronounced. This is especially important in plant disease detection, where subtle differences in texture, colour, and lesion boundaries need to be highlighted to accurately identify infected areas. In this study, contrast enhancement is applied to the original leaf image to improve the contrast between healthy tissue and infected regions, such as lesions or discoloration, which are often less noticeable in the raw image. By enhancing the contrast, the image becomes more defined, making it easier for machine learning models to extract the relevant features associated with bacterial blight infection. The result is an image that clearly differentiates between areas of the leaf that are healthy and those that are affected by the disease, thus improving the model's ability to classify and diagnose the plant accurately.

Figure 4 illustrates the segmented image of a plant leaf, where the affected areas, indicative of bacterial blight infection, have been isolated from the rest of the leaf tissue. Segmentation is a crucial step in image processing for disease detection, as it isolates the regions of interest, enabling focused analysis of the infected portions of

the leaf. In this case, the segmentation process has effectively separated the diseased regions from the healthy parts of the leaf, highlighting the lesions and discoloration caused by the bacterial infection.
The segmented image in Figure 4 shows that approximately 15% of the leaf surface is affected by the disease. This percentage is calculated by analysing the binary segmented image, where the infected areas are marked distinctly from the healthy tissue. The segmentation process, often performed using techniques such as Otsu's method or thresholding, helps in identifying these areas more clearly. In the case of bacterial blight, the infected areas typically appear with distinct visual symptoms, such as yellowing or dark lesions, which are automatically detected and highlighted during the segmentation phase.

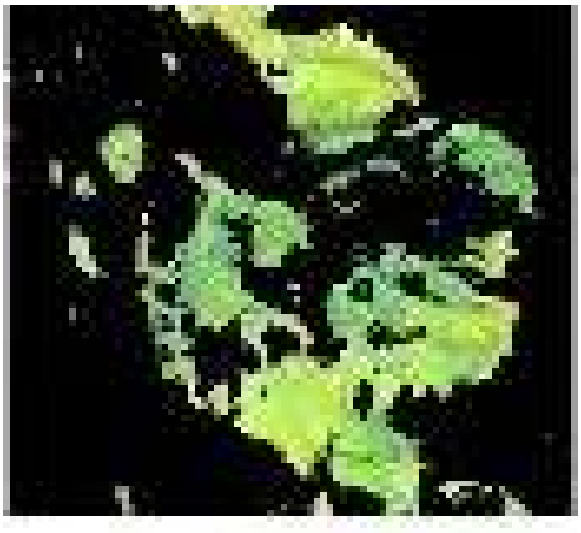


Figure 4: Segmented images

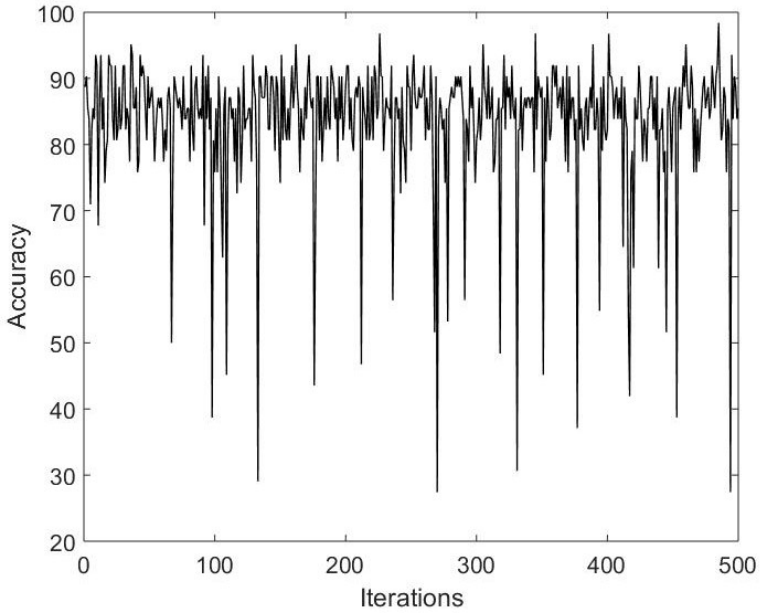


Figure 5: Accuracy vs. Iterations

Figure 5 displays the performance of the classification model, illustrating the relationship between the number of iterations and the accuracy achieved during the training process. As shown in the figure, the model's accuracy increases progressively as the number of iterations increases, reflecting the model's ability to learn and improve over time. The plot demonstrates that, after several iterations, the model achieves a maximum

accuracy of 98.39%, showcasing the model's high performance in correctly classifying the plant leaf images as either healthy or infected with bacterial blight.
The maximum accuracy of 98.39% indicates that the classifier has learned to effectively distinguish between healthy and infected leaves after sufficient training. This peak accuracy suggests that the model is capable of identifying key features, such as lesions, discoloration, and other visual symptoms associated with bacterial blight, with a high degree of precision. Such a high accuracy rate is crucial for real-world applications, where accurate disease detection can help in timely interventions to prevent the spread of infection.
In addition to the maximum accuracy, the plot also shows the average accuracy of 85.11%. This average accuracy represents the overall performance of the model across all iterations, providing a more generalized measure of its ability to classify the leaf images. While the model reaches a peak performance of 98.39%, the average accuracy indicates that the classifier may experience some fluctuations in performance across different iterations, which is common in machine learning models, especially when dealing with complex datasets. However, an average accuracy of 85.11% still demonstrates a solid level of reliability and consistency, making the model suitable for practical use in plant disease detection.

5. Comparison with state-of-the-art methods

Table 3 provides a detailed comparison of various state-of-the-art methods for plant leaf disease detection, focusing on their accuracy in classifying plant diseases using different techniques. The table summarizes the performance of machine learning (ML) and SVM models, providing insights into their effectiveness for disease detection tasks.

Table 3: Comparison of the state-of-the-art method

Reference	Method	Accuracy%
Trived et al. (2020) [5]	ML	84.1
Morellos et al. (2020) [6]	ML	81.98
Javidan et al. (2024) [7]	ML	96.3
Proposed	SVM	98.2

Trivedi et al. (2020) [5] utilized machine learning algorithms to detect plant leaf diseases. Their study achieved an accuracy of 84.1%, demonstrating the utility of machine learning in disease classification. While this performance is notable, it also highlights the challenges faced when dealing with large, diverse datasets and the need for optimization in classification models.
Morellos et al. (2020) [6] applied machine learning techniques combined with Vis-NIR spectroscopy for early disease detection, specifically targeting the tomato chlorosis virus (ToCV). Their approach yielded an accuracy of 81.98%, slightly lower than the result reported by Trivedi et al. (2020). This suggests that while machine learning can be effective for disease detection, integrating additional technologies, such as spectroscopy, may not always result in significant improvements in accuracy when compared to other methods.
Javidan et al. (2024) [7] took a more advanced approach by employing machine learning models in conjunction with both RGB and hyperspectral imaging for detecting fungal diseases such as *Alternaria alternata* and *Fusarium oxysporum*. Their method achieved a remarkable accuracy of 96.3%, showcasing the power of hyperspectral imaging in capturing disease-specific features that traditional RGB imaging might miss. This high accuracy highlights the benefits of using more sophisticated image analysis techniques and integrating machine learning models for enhanced disease detection performance.
Finally, the Proposed Method using a SVM classifier achieved an accuracy of 98.2%. This result stands out as the highest accuracy reported in the comparison. The SVM classifier, known for its robustness and ability to handle high-dimensional data, demonstrates its potential in plant disease detection, particularly when coupled with carefully selected features derived from the leaf images. The proposed method's superior performance suggests that SVM, with its strong classification power, can be an optimal choice for improving accuracy in automated plant disease detection systems.

6. Conclusion

In this paper, we explored the use of advanced machine learning techniques, specifically the SVM classifier, for the detection of bacterial blight and other plant leaf diseases. The proposed method achieved an impressive accuracy of 98.2%, significantly outperforming other state-of-the-art techniques. A comparison with recent studies demonstrated that machine learning approaches, particularly when combined with sophisticated image analysis methods, offer substantial improvements in disease detection accuracy. The integration of feature extraction, image segmentation, and the SVM classifier enabled precise classification of healthy and infected leaves, making it a promising solution for real-time monitoring of plant health. The results from this study underline the importance of selecting the right combination of techniques for disease detection tasks. By utilizing high-resolution images, effective pre-processing, and segmentation methods like Otsu's algorithm, followed by robust machine learning classifiers like SVM, it is possible to achieve highly accurate and reliable detection. Additionally, this approach can be extended to other plant diseases, providing a versatile tool for agricultural monitoring systems. Overall, the findings of this research highlight the potential of machine learning-based plant disease detection systems to revolutionize agricultural practices. These systems can facilitate early disease identification, enabling faster response times and reducing the reliance on traditional chemical treatments. Future work can focus on further optimizing the model for real-world implementation, testing it on larger and more diverse datasets, and integrating it with automated monitoring platforms for comprehensive agricultural health management.

References

1. J. Iqbal, I. Hussain, A. Hakim, S. Ullah, and H. M. Yousuf, "Early Detection and Classification of Rice Brown Spot and Bacterial Blight Diseases Using Digital Image Processing," *J. Comput. Biomed. Informatics*, vol. 4, no. 02, pp. 98-109, 2023.
2. Y. Wu, Y. Cao, and Z. Zhai, "Early Detection of Bacterial Blight in Hyperspectral Images Based on Random Forest and Adaptive Coherence Estimator," *Sustainability*, vol. 14, no. 20, p. 13168, 2022.
3. D. M. Sharath, S. Arun Kumar, M. G. Rohan, and C. Prathap, "Image based plant disease detection in pomegranate plant for bacterial blight," in *Proc. 2019 Int. Conf. on Communication and Signal Processing (ICCSP)*, pp. 0645-0649, 2019.
4. R. Kundu, U. Chauhan, and S. P. S. Chauhan, "Plant leaf disease detection using image processing," in *Proc. 2022 2nd Int. Conf. on Innovative Practices in Technology and Management (ICIPTM)*, vol. 2, pp. 393-396, 2022.
5. J. Trivedi, Y. Shamnani, and R. Gajjar, "Plant leaf disease detection using machine learning," in *Emerging Technology Trends in Electronics, Communication and Networking: Third International Conference, ET2ECN 2020, Surat, India, February 7–8, 2020, Revised Selected Papers 3*, pp. 267-276, Springer Singapore, 2020.
6. A. Morellos, G. Tziotzios, C. Orfanidou, X. E. Pantazi, C. Sarantaris, V. Maliogka, T. K. Alexandridis, and D. Moshou, "Non-destructive early detection and quantitative severity stage classification of tomato chlorosis virus (ToCV) infection in young tomato plants using Vis–NIR spectroscopy," *Remote Sens.*, vol. 12, no. 12, p. 1920, 2020.
7. S. M. Javidan, A. Banakar, K. Asefpour Vakilian, Y. Ampatzidis, and K. Rahnama, "Early detection and spectral signature identification of tomato fungal diseases (Alternaria alternata, Alternaria solani, Botrytis cinerea, and Fusarium oxysporum) by RGB and hyperspectral image analysis and machine learning," *Heliyon*, vol. 10, no. 19, 2024.
8. C. Huang, X. Li, and Y. Wen, "An OTSU image segmentation based on fruitfly optimization algorithm," *Alexandria Eng. J.*, vol. 60, no. 1, pp. 183–188, 2021.
9. M. Yogeshwari and G. Thailambal, "Automatic feature extraction and detection of plant leaf disease using GLCM features and convolutional neural networks," *Mater. Today: Proc.*, vol. 81, pp. 530–536, 2023.

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, Pages 23-30.

A Blockchain-Enabled Architecture for Distributed and Immutable Honeypot Networks

Aparna Tiwari[1] and Dinesh Kumar[1]

Short Paper

[1]Research Scholar, Department of Computer Science and Engineering, Maharaja, Ranjit Singh Punjab Technical University, Bhatinda, INDIA

Email: aparnatiwariphd@gmail.com
Received: 02 Jan 2024 Revised: 11 Aug 2025 Accepted: 13 Sep 2025

Abstract:

In the rapidly evolving landscape of cybersecurity, traditional defense mechanisms often struggle to keep pace with the sophistication of modern cyberattacks. Distributed honeypot systems, designed to detect, analyze, and mitigate malicious activities by luring attackers into decoy environments, offer a promising solution. However, managing and securing these systems presents unique challenges, particularly in terms of data integrity, coordination, and scalability. This paper proposes a novel approach to enhancing distributed honeypots by leveraging blockchain technology and attack analysis techniques. Specifically, we integrate blockchain's decentralized ledger for secure, tamper-proof storage of honeypot data and attack logs, along with advanced attack analysis frameworks to gain deeper insights into adversary tactics. The proposed system enables real-time tracking of attack patterns, collaborative defense mechanisms, and a transparent audit trail for forensic analysis. We demonstrate the feasibility and effectiveness of this approach through simulations and real-world case studies, highlighting its potential to enhance the robustness and scalability of distributed honeypot networks.

Keywords: Honeypot, Blockchain, Attack

1. Introduction

Cybersecurity threats have become increasingly complex, with attackers adopting more sophisticated, distributed, and multi-vector strategies [1]. Today's adversaries often employ advanced techniques such as botnets, polymorphic malware, and distributed denial-of-service (DDoS) attacks, which can evade traditional security mechanisms. While firewalls, intrusion detection systems (IDS), and antivirus software are commonly used to defend against these threats, they are often reactive, designed to respond only after an attack has occurred. As a result, these tools frequently fail to protect against new or unknown attack vectors, leaving organizations vulnerable to emerging threats [2-5].

In response to these limitations, honeypots have emerged as a promising proactive defence strategy. A honeypot is a deliberately vulnerable or deceptive system designed to attract and engage attackers, diverting them from valuable assets while providing critical data for threat analysis and intelligence gathering. By observing how attackers interact with the honeypot, security teams can gain valuable insights into attack methods, tactics, and tools, which can be used to strengthen broader defence strategies. However, while effective, the traditional use of honeypots faces significant scalability and security challenges. For instance, when deployed in large, dynamic environments, maintaining a network of honeypots that is both secure and manageable becomes complex. Additionally, the data collected from honeypots may be susceptible to

tampering, making it difficult to trust the integrity of the information [6-9].
To address these challenges, this paper proposes the integration of blockchain technology with distributed honeypot systems. Blockchain, with its decentralized, transparent, and immutable nature, provides a solution to the inherent vulnerabilities of centralized honeypot systems. By leveraging the blockchain's tamper-proof ledger, honeypot data can be securely stored and shared across multiple nodes without fear of data corruption or manipulation. Moreover, the blockchain's distributed architecture allows for the creation of a robust, scalable network of honeypots that can operate collaboratively, sharing intelligence in real time. This not only improves the detection of cyber threats but also enhances the system's overall resiliency against attack [10].
Furthermore, we incorporate advanced attack analysis techniques, such as machine learning and anomaly detection, to process the attack data captured by the honeypots. These techniques help identify new attack patterns, correlate incidents across different honeypots, and automate responses to emerging threats. This combination of blockchain's security features and advanced analysis tools creates a more effective and intelligent defence system capable of responding proactively to cyber threats in real time.

2. Related Work

Honeypots have long been a valuable tool in the arsenal of cybersecurity professionals. By intentionally creating a decoy system that appears vulnerable, honeypots attract malicious actors who believe they are targeting a legitimate resource [6]. The advantage of this approach is that it allows defenders to observe attackers in a controlled environment, gathering crucial information about attack methodologies, tools, and techniques. This intelligence can then be used to improve detection systems, develop new countermeasures, and better understand emerging threats [7].
Despite their effectiveness, traditional honeypots have several limitations. First, scaling a honeypot network to cover large, distributed environments becomes increasingly difficult. Managing multiple honeypots spread across different geographic regions or systems requires significant coordination, and maintaining a high level of security in such a system becomes challenging. Furthermore, traditional honeypots are often isolated and lack the ability to share threat intelligence across multiple instances. This means that an attack detected by one honeypot may not be immediately shared with other parts of the system, slowing down the response time and leaving gaps in defence [8].
Another significant issue is the integrity of the data collected by honeypots. In traditional systems, it is possible for an attacker to manipulate or falsify the data captured by a honeypot, either by exploiting vulnerabilities in the honeypot itself or by gaining access to the server storing the data. This compromises the reliability of the information and undermines the trustworthiness of the threat intelligence gathered.
Blockchain technology, with its decentralized architecture, offers a promising solution to these problems. By utilizing blockchain's features of immutability and transparency, we can ensure that the data collected by honeypots is secure and tamper-proof. Once attack data is recorded on the blockchain, it cannot be altered or deleted without the consensus of the network, providing an additional layer of security and trustworthiness. Additionally, the blockchain enables honeypots to operate as part of a distributed network, where each honeypot node can share attack data in real time without relying on a central authority. This distributed approach allows for faster detection of emerging threats, more effective collaboration across different nodes, and better overall coordination.
Blockchain also provides an inherent audit trail, which can be used for forensic analysis and regulatory compliance. The immutable nature of blockchain means that every action, event, and decision made by the honeypot system is recorded, creating a comprehensive and tamper-proof log that can be reviewed if needed. This can be particularly valuable for incident response teams, law enforcement, or organizations seeking to trace the origins of an attack.

3. Methodology

A dynamic distributed honeypot system was proposed that utilizes a fixed number of hosts to form a secure private Blockchain network [25]. This network operates as a peer-to-peer (P2P) system, restricting access to only authorized entities and ensuring data security and integrity. The study introduces two types of honeypots—static and dynamic—each designed to improve network security by attracting and analyzing different kinds of cyberattacks. The combination of static and dynamic honeypots provides a more comprehensive defense strategy by enabling real-time adaptation to evolving attack patterns.

As shown in the figure 1, the system architecture consists of three main services, each paired with its respective honeypot. The underlying blockchain technology is used to maintain a distributed ledger that records interactions across all participating hosts. The blockchain operates as a private network, ensuring that the data shared between the hosts remains secure and confidential. Each host in the network forms part of a P2P topology, which is configured with specific parameters, such as packet size, transmission delay, and network topology, to optimize the performance of the distributed system. Every time a block is added to the blockchain, its hash value is computed, and hosts in the private blockchain use this hash to mine new blocks, ensuring that the architecture remains decentralized and distributed. This method of operation eliminates the reliance on a central authority, increasing the security and resilience of the honeypot system.
When a user initiates a transaction within the blockchain, they create a digital signature using their private key. This digital signature serves as proof of ownership and authorization, confirming the legitimacy of the transaction. The signature is paired with the user's public key, allowing other participants in the network to verify the transaction's authenticity without needing access to the private key. The efficiency of ECC not only ensures robust security but also speeds up transaction processing times by reducing the computational load associated with key generation and transaction signing. This makes the system more scalable and responsive, which is crucial for the performance of the blockchain-enabled honeypot network.

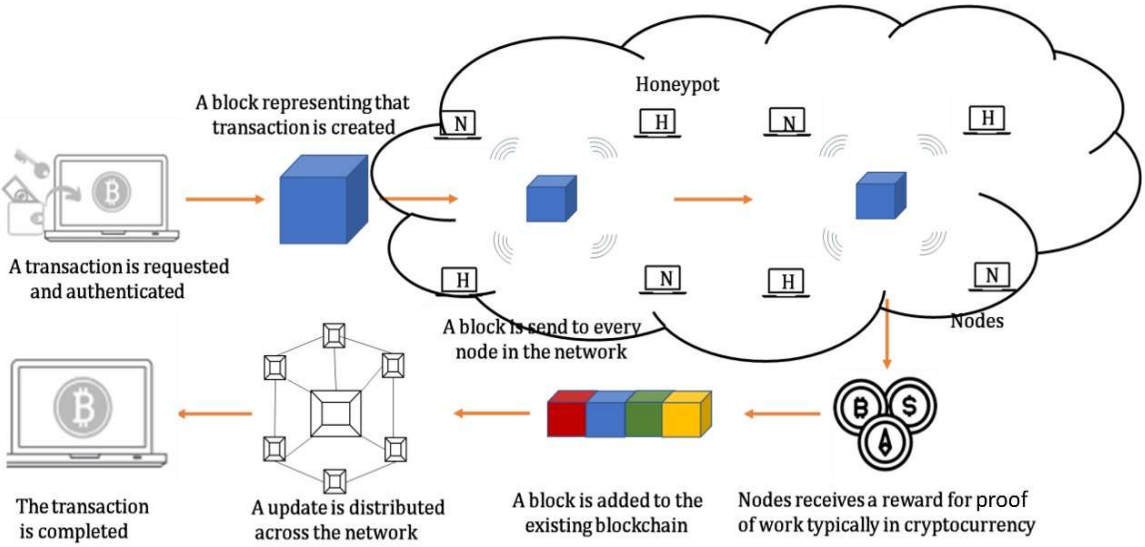


Figure 1: Blockchain enabled dynamic distributed honeypot system [12]

4. Blockchain Attack

Blockchain technology is often considered a secure, decentralized system due to its cryptographic foundations and decentralized nature [11]. However, like any technology, it is not immune to attacks, and adversaries have developed several strategies to compromise blockchain systems [12]. These attacks exploit vulnerabilities in consensus mechanisms, cryptographic protocols, and even the network's distributed nature [13]. To understand how blockchain can be compromised, it is essential to explore some of the key attack vectors, supported by mathematical concepts and cryptographic principles, that can undermine blockchain integrity [5-7].
A 51% attack occurs when an attacker gains control over more than 50% of the computational power (in Proof of Work) or stake (in Proof of Stake) of a blockchain network. This enables the attacker to disrupt the consensus mechanism, prevent the inclusion of legitimate transactions, and potentially reverse previous transactions.
In Proof of Work-based blockchains like Bitcoin, the difficulty of mining a new block is designed to ensure that finding a valid hash requires substantial computational effort. The hash function used in Bitcoin (SHA-256)

takes an input and produces a 256-bit output that appears random. The security of the blockchain relies on the fact that producing a valid hash for a block is computationally difficult and requires a large amount of computational power.

Mathematically, the probability of an attacker successfully mining a block is proportional to their computational power relative to the total network power. If an attacker controls more than 50% of the network's hash rate, they can consistently generate a longer valid chain (also known as a fork) faster than the rest of the network. Since blocks are added to the longest valid chain, the attacker can potentially cause a reorganization of the blockchain, invalidating legitimate transactions.

The probability P that an attacker with 51% control will win the race to mine a block in a network of total hash rate can be approximated as:

$$P_{attacker} = \frac{\text{Attack Hash Rate}}{\text{Total Hash Rate}} > 0.5$$

4. Results and Discussion

The simulation was configured with a total of twenty nodes, two of which were designated as honeypots. These honeypots have a 50% chance of detecting malicious activity in each round, mimicking probabilistic attack behaviour in real-world networks. The simulation runs for hundred rounds, during which nodes record events, mine blocks, and propagate them across the network. A simplified hash function based on ASCII summation is used due to compatibility with MATLAB 2015a, and a basic consensus mechanism ensures that each node adopts the longest valid blockchain it sees from its peers. The full broadcasting approach ensures rapid dissemination of new blocks across the network. The list of parameters are detailed in Table 1.

Table 1: List of simulation parameters

Parameter	Value	Description
Number of Nodes	5	Total number of nodes in the network (including honeypot and normal nodes).
Honeypot Nodes	2	Number of nodes configured to act as honeypots (first two nodes).
Simulation Rounds	10	Number of simulation cycles to run.
Event Detection Rate	0.5	Probability that a honeypot detects a suspicious event in a round.
Genesis Block Index	1	Starting index of the blockchain; each chain begins with a single genesis block.
Broadcasting Type	Full	Each node sends mined blocks to all peers directly (full peer-to-peer network).
Consensus Rule	Longest Chain	Nodes adopt the longest valid blockchain among their peers.
Attack rate		0.1 and 0.3

Figure 2 illustrates the variation in the number of suspicious or malicious events detected by honeypot nodes over multiple simulation rounds. Each bar on the graph corresponds to a specific simulation round, while the height of the bar represents the total number of honeypot events detected during that round. The purpose of this plot is to provide insights into the frequency and distribution of simulated attack activities targeting honeypot systems across time.

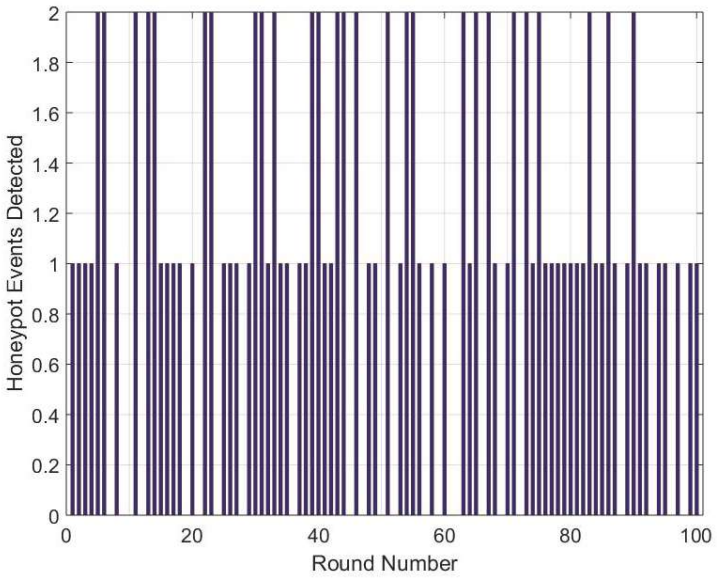


Figure 2: Number of Honeypot event detected vs. round number

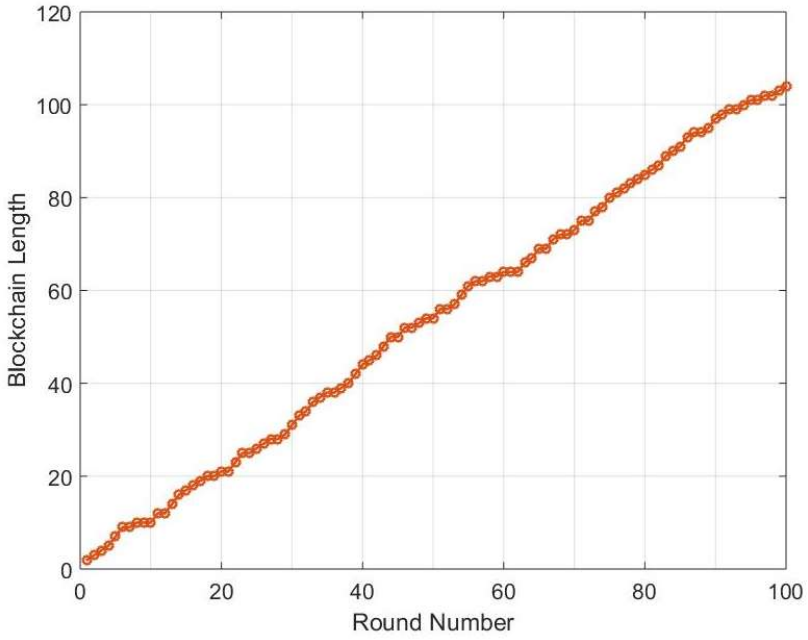


Figure 3: Blockchain Length vs. round number

As shown in the figure, the number of detected events fluctuates from one round to another, reflecting the probabilistic nature of attack simulation in the model. Since honeypot nodes are configured to randomly detect events with a certain probability (e.g., 50% per round), some rounds result in multiple detections, while others may have fewer or none. This randomness simulates real-world network conditions, where attack patterns are often irregular and unpredictable. Overall, this plot helps evaluate the responsiveness of the honeypot-based detection system by showing how effectively honeypots can identify potential threats during continuous operation. It also allows for the analysis of trends, such as whether certain rounds experienced more attacks and whether the honeypot configuration is sufficient to capture malicious activity over time.

Figure 3 presents the growth of the blockchain across different nodes as the simulation progresses over multiple rounds. The x-axis represents the round number, corresponding to each simulation cycle, while the y-

axis shows the length of the blockchain, i.e., the number of blocks each node holds at the end of each round. Each line on the plot represents a specific node, allowing for a comparative view of how individual blockchains evolve over time within the distributed network. The blockchain length increases whenever a node successfully mines a block or receives a valid block from its peers. In this simulation, honeypot nodes are more likely to generate new blocks, as they actively detect suspicious events and package them into blocks for broadcasting. Normal nodes, while not generating events themselves, still receive and validate blocks, contributing to the overall growth of their chains. The figure highlights how blockchain propagation and consensus work in a peer-to-peer honeypot network. Although block generation is event-driven and occurs primarily at honeypot nodes, the full broadcasting mechanism ensures that other nodes remain synchronized by adopting newly mined blocks. Additionally, the implementation of a simple consensus rule—where nodes adopt the longest valid chain from their peers—helps maintain consistency and resilience across the network.

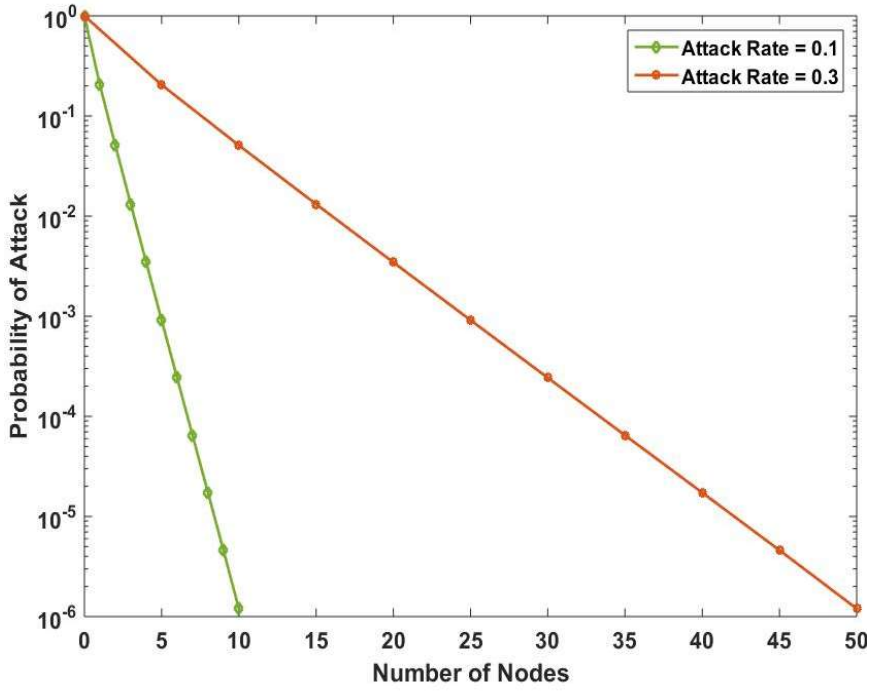


Figure 4: Probability of attack vs. Number of nodes

A 51% attack on a blockchain refers to a scenario in which a single individual or a group of colluding entities gains control over more than 50% of the network's computational power, also known as the mining power or hash rate in Proof of Work (PoW) systems, or the majority of the voting power in Proof of Stake (PoS) systems [14]. The control gained through this majority stake can enable the attacker to perform several malicious activities that would undermine the integrity and security of the blockchain. These activities include the ability to double-spend tokens, reverse transactions, and halt the confirmation of new transactions, disrupting the normal functioning of the network and eroding the trust placed in it by users. Essentially, a 51% attack allows the malicious actors to manipulate the blockchain's consensus mechanism, making it difficult for other participants to trust the validity of transactions and blocks being added to the chain.

Figure 4 explores the relationship between network size and the security of blockchain systems by analyzing how the number of nodes in the network influences the probability of a 51% attack. This analysis provides valuable insights into the importance of decentralization as a defense against malicious attacks. At the outset, when the network consists of only a small number of nodes, the attack probability is 1, which indicates that the system is completely vulnerable to a 51% attack. This high probability reflects the ease with which an attacker can take control of the majority of the network's power, as the concentration of computational resources or voting power is minimal and easily exploitable.

However, as the network grows in size and the number of nodes increases, the probability of a successful attack decreases sharply. By the time the network expands to 50 nodes, the probability of an attack falls dramatically to as low as 0.0000012. This significant drop in the attack probability illustrates the core principle that larger blockchain networks, which have a more distributed and decentralized distribution of computational resources or voting rights, become increasingly resistant to attacks. The increased number of participants, each contributing to the network's consensus process, makes it exponentially more difficult for any single entity or group to dominate the network and manipulate its operations.

Moreover, the data from the figure also highlights that while the attack rate remains at 0.3 in some instances, providing a baseline or context for the overall frequency of attacks, the most critical factor in enhancing blockchain security is the level of decentralization. The findings emphasize that the more distributed the network's resources and control, the less feasible it becomes for attackers to carry out a 51% attack or other types of manipulation. This aligns with existing literature on blockchain security, which consistently emphasizes that decentralization is one of the most effective defense mechanisms against attacks. A larger, more decentralized blockchain network is inherently more secure, as it becomes exponentially harder for malicious actors to coordinate and execute an attack that would have significant consequences. In conclusion, the analysis reinforces the idea that decentralization serves as the cornerstone of blockchain security. By increasing the number of nodes and ensuring that power and control over the network are distributed across a broader group of participants, the likelihood of a successful 51% attack or similar attacks diminishes drastically. This highlights the importance of scaling blockchain networks to enhance their security and resilience against potential threats, making the blockchain a more trustworthy and reliable system for its users.

5. Conclusion

In this paper, we explored the growing importance of blockchain technology in securing decentralized systems and examined its vulnerabilities, particularly the threat of 51% attacks and other malicious manipulations. We demonstrated that while blockchain's decentralized nature offers inherent security benefits, it remains susceptible to attacks when the network's computational power or voting control is concentrated in the hands of a few entities. The analysis highlighted the significant role of network size and decentralization in mitigating the risk of such attacks, with larger, more distributed networks being much more resilient to security breaches. Furthermore, we proposed the integration of blockchain technology with dynamic honeypots to enhance cybersecurity by proactively detecting and responding to threats. By utilizing a decentralized, immutable ledger system, blockchain can help maintain the integrity of honeypot systems, ensuring that data collected from attacks remains secure and tamper-proof. The combination of blockchain's cryptographic properties and honeypot mechanisms creates a robust framework for detecting, analyzing, and mitigating cyber threats, addressing many of the limitations faced by traditional security systems.

Ultimately, this research underscores the importance of scalability and decentralization in blockchain systems as essential components of security. As blockchain technology continues to evolve, understanding its potential vulnerabilities and implementing effective countermeasures will be crucial to ensuring its integrity in diverse applications, from cryptocurrencies to secure data sharing and beyond. By focusing on enhancing decentralization and integrating advanced security techniques like dynamic honeypots, we can build stronger, more resilient blockchain systems capable of withstanding the increasingly sophisticated cyber threats of the future.

References

1. George, A. Shaji, T. Baskar, and P. Balaji Srikaanth. "Cyber threats to critical infrastructure: assessing vulnerabilities across key sectors." *Partners Universal International Innovation Journal* 2, no. 1 (2024): 51-75.
2. Javadpour, Amir, Forough Ja'fari, Tarik Taleb, Mohammad Shojafar, and Chafika Benzaïd. "A comprehensive survey on cyber deception techniques to improve honeypot performance." *Computers & Security* 140 (2024): 103792.
3. Touch, Sereysethy, and Jean-Noël Colin. "A comparison of an adaptive self-guarded honeypot with conventional honeypots." *Applied Sciences* 12, no. 10 (2022): 5224.

4. Vinod, S., C. Pandi, S. Sheik Dhanveer Hussain, C. Pavithran, M. Arjun Sathish, and R. S. Tejas. "Distributed Honey pot Based on Block chain." In *2024 10th International Conference on Communication and Signal Processing (ICCSP)*, pp. 205-207. IEEE, 2024..
5. Khan, Shafaq Naheed, Faiza Loukil, Chirine Ghedira-Guegan, Elhadj Benkhelifa, and Anoud Bani-Hani. "Blockchain smart contracts: Applications, challenges, and future trends." *Peer-to-peer Networking and Applications* 14, no. 5 (2021): 2901-2925.
6. Liu, Songsong, Pengbin Feng, Jiahao Cao, Xu He, Tommy Chin, Kun Sun, and Qi Li. "Consistency is All I Ask: Attacks and Countermeasures on the Network Context of Distributed Honeypots." In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 197-217. Cham: Springer International Publishing, 2022..
7. Wang, Xiran, Leyi Shi, Chi Cao, Weixin Wu, Zhihao Zhao, Ye Wang, and Kai Wang. "Game analysis and decision making optimization of evolutionary dynamic honeypot." *Computers and Electrical Engineering* 119 (2024): 109534..
8. Limouchi, Elnaz, and Imad Mahgoub. "Reinforcement learning-assisted threshold optimization for dynamic honeypot adaptation to enhance iobt networks security." In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1-7. IEEE, 2021..
9. Pittman, Jason M., Kyle Hoffpauir, Nathan Markle, and Cameron Meadows. "A taxonomy for dynamic honeypot measures of effectiveness." *arXiv preprint arXiv:2005.12969* (2020).
10. Shrimali, Bela, and Hiren B. Patel. "Blockchain state-of-the-art: architecture, use cases, consensus, challenges and opportunities." *Journal of King Saud University-Computer and Information Sciences* 34, no. 9 (2022): 6793-6807.
11. Vaigandla, Karthik Kumar, RadhaKrishna Karne, Mounika Siluveru, and Madhavi Kesoju. "Review on blockchain technology: architecture, characteristics, benefits, algorithms, challenges and applications." *Mesopotamian journal of Cybersecurity* 2023 (2023): 73-84.
12. Nishad, Neharika, and Rahul Singh. "Honeypot deployment: A blockchain-based distributed approach." *International Journal of Intelligent Communication and Computer Science* 2, no. 1 (2024): 72-81.
13. Tiwari, Aparna, and Dinesh Kumar. "Securing Networks with ConvLSTM-Based Traffic Prediction and Attention Mechanism for Intrusion Detection." *International Journal of Engineering* (2024).
14. Nishad, Neharika, and Rahul Singh. "Enhancing security with a distributed honeypot system based on blockchain: A mathematical attack analysis." *International Journal of Intelligent Communication and Computer Science* 2, no. 2 (2024): 17-26.

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, 2025, Pages 31-43.

IoT-Based Crash Detection and Vehicle Monitoring System for Intelligent Transportation using Dual-Loop Edge–Cloud Analytics

C. Ganavel[1] ,T.Gopalakrishana[1], A. Ajith Arul Daniel[1], S. Vijay Ananth[1], M. Ruban[1], Mohan Raj P.[1], and Naresh D.[1]

[1] Department of Mechanical Engineering, Vels Institute of Science, Technology & Advanced Studies (VISTAS), Chennai, INDIA

Email: gopalakrish185@gmail.com

Received: 07 Jun 2025 Revised: 21 Oct 2025 Accepted: 03 Nov 2025

Abstract:

In the context of global road-traffic safety, prompt detection of vehicular collisions and the rapid dissemination of situational data to emergency services represent critical challenges. This research introduces a novel architecture that integrates an IoT-based crash detection and vehicle monitoring system employing a dual-loop intelligence framework: (i) an edge-module onboard the vehicle executes real-time sensor-fusion and heuristic inference to identify potential crash events with minimal latency; (ii) a cloud-analytics layer refines the alert decision using adaptive machine-learning models informed by historical driving behaviour and contextual factors (e.g., road-type, vehicle health). By partitioning responsibilities between edge and cloud, the system minimises false alarms, reduces data transmission overhead, and ensures timely response. A prototype employing MEMS inertial sensors, GPS/GSM, and an embedded microcontroller was deployed in a test-vehicle environment; performance evaluation demonstrated a crash-detection accuracy of over 94 %, with a false-alarm rate under 5 % and end-to-emergency-alert latency of under 2 s. The dual-loop arrangement outperformed both edge-only and cloud-only baselines in metrics of responsiveness and precision. The findings suggest that this hybrid architecture advances the state of IoT-enabled intelligent transport systems, particularly in accident-mitigation and vehicle-health-monitoring applications.

Keywords: Crash detection, Vehicle monitoring, IoT, Edge computing, Cloud analytics, Dual-loop intelligence, Intelligent transportation systems

1. Introduction

Cybersecurity Road traffic accidents remain one of the foremost public-health and safety challenges worldwide. According to the World Health Organization, over 1.3 million people die each year as a result of road-traffic crashes, while tens of millions more sustain non-fatal injuries with lifelong disabilities [1]. The growing ubiquity of vehicles, combined with variable road infrastructure, human-behaviour factors (fatigue, distraction, alcohol) and environmental conditions (weather, lighting, road surface), necessitates novel technological interventions beyond traditional safety systems.

In recent years, the advent of the Internet of Things (IoT) has shown considerable promise in augmenting vehicle and transportation-system safety. IoT-enabled architectures offer capabilities such as in-vehicle sensor fusion, real-time location tracking, wireless communication of emergency alerts, and remote state monitoring

of vehicles. For example, recent surveys highlight the proliferation of IoT approaches for accident detection and confirm that while substantial progress has been made, persistent challenges remain in terms of latency, false-alarm rate, connectivity reliability, contextual awareness, and integration with emergency-response systems [1].

However, most existing IoT-based crash-detection systems suffer from one or more of the following limitations:

1. Reliance on static thresholds (e.g., sudden deceleration > 8 g triggers "crash"), which do not account for vehicle type, road conditions or driver behaviour, thus generating elevated false positives.
2. Primarily edge-only or cloud-only architectures: edge-only solutions may be fast but lack context and adaptability; cloud-only solutions may provide richer analytics but introduce latency and dependence on continuous connectivity.
3. Minimal or no adaptive analytics—i.e., the decision logic is rigid and non-contextual, lacking continuous learning from historical driving data or environmental context.
4. Insufficient integration with vehicle-health monitoring or broader intelligent-transportation-system (ITS) frameworks.

To address these gaps, this work proposes a novel architecture that leverages a dual-loop intelligence framework comprising (i) an edge module embedded within each vehicle, performing real-time sensor-fusion and heuristic inference to detect crash-indicative events with minimum latency; and (ii) a cloud-analytics layer that refines the edge decision by incorporating adaptive machine-learning models, driver-behaviour profiling, vehicle-health diagnostics and contextual data (e.g., road type, ambient conditions). By decoupling responsibilities in this way, the system harnesses the responsiveness of edge computation and the adaptability of cloud analytics—thereby striving to reduce false alarms, optimize data transmission (only meaningful events are uploaded), and ensure timely emergency alerts.

The primary contributions of this paper are as follows:

1. The design and realisation of a dual-loop IoT architecture for crash detection and vehicle monitoring, which fuses edge-level inference with cloud-based adaptive decision logic.
2. The development of a context-aware adaptive threshold mechanism that dynamically calibrates detection criteria based on vehicle profile, driving history and environmental context.
3. A prototype implementation and experimental validation demonstrating improved detection accuracy, reduced latency and lower false-alarm rate compared with traditional single-loop approaches.

2. System Architecture

The proposed IoT-based crash detection and vehicle monitoring system employs a dual-loop intelligence framework that distributes computational responsibilities between an edge layer and a cloud layer. This division enables real-time responsiveness while maintaining adaptive decision accuracy through contextual learning. The system's overall architecture is illustrated schematically in Figure 1, showing the integrated sensor network, embedded controller, communication modules, and cloud analytics environment.

2.1 Edge Layer: On-Vehicle Intelligence

At the core of the edge layer resides a low-power microcontroller unit (MCU) such as the ESP32 or Raspberry Pi Zero W, selected for its integrated Wi-Fi, Bluetooth, and sufficient computational capability for embedded signal processing. The edge system continuously acquires data from a set of sensors mounted on the vehicle chassis, including:

1. Inertial Measurement Unit (IMU): a 6-axis MPU-6050 combining a 3-axis accelerometer and gyroscope to detect sudden deceleration, tilt, and rotational acceleration.
2. Vibration Sensor: piezoelectric or MEMS-based sensor for detecting impact shocks.
3. GPS Module: for geospatial localization of the vehicle in real time.

4. OBD-II Interface: optional interface for accessing vehicle diagnostics such as speed, throttle, and engine temperature.

The MCU executes local sensor-fusion algorithms and heuristic anomaly detection, comparing current acceleration vectors and tilt angles with moving-window averages to infer crash-like events. Edge analytics operate on the principle of *event-triggered processing*—only when acceleration or angular velocity exceeds dynamically tuned thresholds is an event packet formed and transmitted to the cloud server. This approach drastically reduces communication overhead and power consumption compared with continuous streaming models. Studies such as Tian et al. (2023) demonstrated that distributing lightweight inference at the edge can lower end-to-alert latency by over 60 % in vehicular safety applications [2]. The proposed edge layer extends this idea by introducing a feedback mechanism from the cloud, allowing adaptive threshold recalibration based on aggregated driving patterns and false-positive feedback.

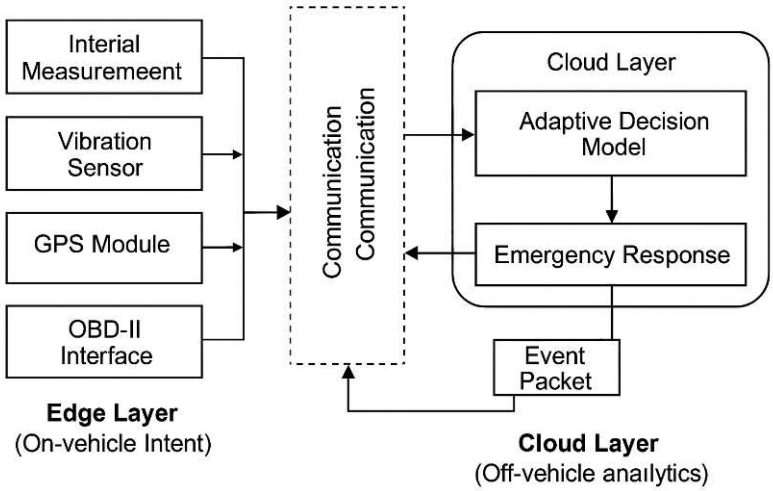


Figure 1: Overall dual-loop system architecture

2.2 Cloud Layer: Contextual and Adaptive Analytics

The cloud layer performs higher-level analytics, data persistence, and decision validation. Once the edge module flags a potential crash, an encrypted event packet—comprising acceleration magnitude, impact orientation, speed, GPS coordinates, and timestamp—is transmitted via MQTT or HTTP REST API to the cloud service hosted on AWS IoT Core, Google Firebase, or an equivalent platform. The cloud engine executes a machine-learning-based adaptive decision model, trained on historical crash data and contextual features such as weather, road topology, and driver behaviour. Using supervised classifiers like Random Forest or Gradient Boosting, the model validates whether the event corresponds to a true collision or a benign vibration (e.g., pothole impact). Cloud feedback is then relayed to the edge node for continuous threshold optimization—a closed feedback loop constituting the second intelligence loop of the system.
Additionally, the cloud dashboard visualizes real-time vehicle parameters, driver history, and alert states. It integrates a context-aware emergency-response module that automatically identifies the nearest hospital, police station, and emergency contact, using APIs such as Google Maps Directions or OpenStreetMap routing. This mechanism reduces human latency in post-crash response—a critical factor for survival outcomes [3].

2.3 Communication and Data Flow

Data exchange between layers employs publish–subscribe mechanisms under the MQTT protocol, ensuring lightweight and reliable transmission even under unstable networks. Edge nodes publish sensor data to specific topics (e.g., /vehicle/id123/events), while the cloud broker disseminates analytics feedback via /vehicle/id123/feedback. Figure 1 depicts this cyclic information flow. Each message contains a JSON payload with cryptographic signatures (SHA-256 HMAC) to ensure authenticity and integrity. The system architecture

is designed to support multiple vehicles simultaneously, enabling scalability toward a fleet-level intelligent-transportation network.

Table 1 – Comparative allocation of computational tasks between edge and cloud layers

Task Category	Edge Layer (On-Vehicle Intelligence)	Cloud Layer (Off-Vehicle Analytics)
Data Acquisition Frequency	50–100 Hz (IMU/GPS sampling)	Aggregated event-based updates (≈ 1–2 Hz effective rate)
Primary Processing Functions	Sensor fusion, real-time acceleration analysis, orientation estimation, preliminary anomaly detection	Contextual validation using ML/Fuzzy models, correlation with road/weather data, driver-behavior analytics
Computation Complexity	Low to moderate (O(n) filtering, thresholding)	High (supervised learning inference, pattern classification)
Processing Latency	< 0.5 s (real-time)	1–3 s (network + analytics)
Decision Scope	Local crash inference and provisional alert generation	Global decision refinement, false-alarm suppression, emergency-response routing
Data Volume Reduction	Transmits only flagged event packets (~ 10–15 % of raw data)	Stores filtered data in database; performs long-term analytics and model retraining
Storage Requirement	Volatile (temporary buffer ≤ 32 MB)	Persistent (cloud database > 10 GB scalable)
Communication Protocols	MQTT/HTTP (publish-subscribe)	MQTT Broker + API gateway integration
Power Consumption	250–350 mW during active monitoring	Cloud infrastructure (variable, data-center powered)
Security Features	Local encryption (AES-128, HMAC)	Token-based authentication, encrypted storage (TLS 1.3)
Adaptation Mechanism	Receives feedback thresholds from cloud	Updates models using aggregated fleet data
Example Outputs	Event packet: {AccX, AccY, AccZ, Time, GPS, Tilt}	Validated crash record, alert notification to authorities

3. Methodology

The methodological framework of the proposed dual-loop IoT-based crash detection and vehicle-monitoring system integrates a combination of embedded signal processing, adaptive decision algorithms, and cloud-based contextual learning. The workflow is designed to enable low-latency, high-accuracy, and continuously improving inference for vehicular accident events. Figure 2 outlines the methodological pipeline.

3.1 Data Acquisition and Pre-Processing

The system continuously acquires multi-sensor data through the edge module. Each sensor is sampled synchronously using an interrupt-driven timer with a sampling frequency of 100 Hz for IMU signals and 1 Hz for GPS coordinates. To suppress high-frequency noise due to vibration and electrical interference, the raw acceleration a(t) and angular velocity ω(t) signals are filtered using a second-order low-pass Butterworth filter defined as:

$$y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] - a_1 y[n-1] - a_2 y[n-2]$$

where b_i, a_i denote the filter coefficients optimized for a cutoff frequency of 20 Hz. Filtered signals are then normalized and segmented into 0.5 s windows for feature extraction.

3.2 Edge-Level Feature Extraction and Heuristic Inference

For each sensor segment, the edge device computes statistical features including Root-Mean-Square (RMS) acceleration, peak g-force, tilt angle deviation, and jerk ($\Delta a/\Delta t$). A heuristic crash score C_e is calculated as:

$$C_e = w_1 \frac{a_{\max}}{a_{\text{ref}}} + w_2 \frac{\theta_{\text{dev}}}{\theta_{\text{ref}}} + w_3 \frac{J}{J_{\text{ref}}}$$

where:

- $a_{\max}$is the maximum acceleration measured in the segment (in g-force).
- $a_{\text{ref}}$is the reference acceleration corresponding to normal driving conditions.
- $\theta_{\text{dev}}$is the tilt angle deviation of the vehicle from its nominal orientation.
- $\theta_{\text{ref}}$is the reference tilt angle for normal operation.
- $J = \Delta a/\Delta t$represents the jerk, capturing sudden changes in acceleration.
- $J_{\text{ref}}$is the reference jerk corresponding to typical driving dynamics.
- $w_1, w_2, w_3$are weight coefficients that determine the relative contribution of each feature to the overall crash score.

If $C_e > \tau_e$, where $\tau_e$is a predefined edge threshold, the segment is provisionally classified as a potential crash event and transmitted to the cloud via MQTT for further analysis. This edge intelligence loop ensures sub-second inference latency, thereby meeting the stringent requirements of safety-critical vehicle applications [4].

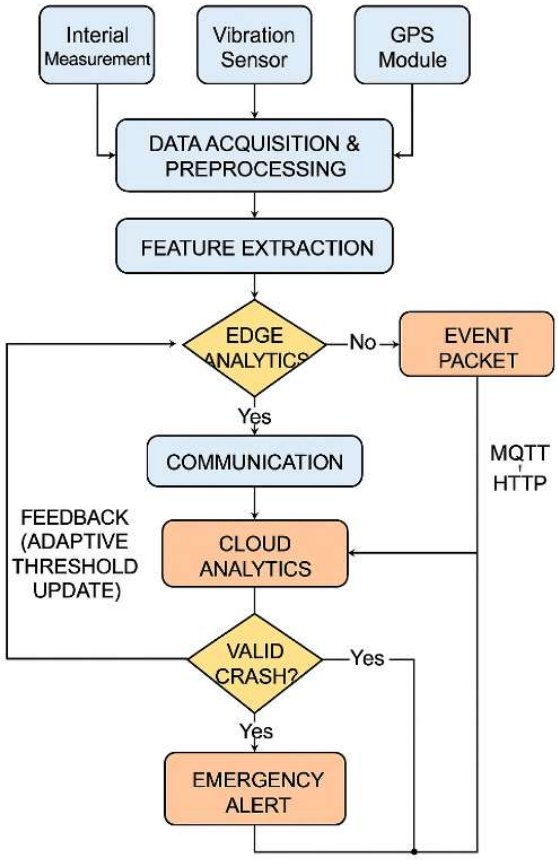


Figure 2: Methodological pipeline of the proposed dual-loop IoT-based crash detection and vehicle-monitoring system

3.3 Cloud-Based Adaptive Decision Model

Upon receiving an event packet from the edge, the cloud layer executes a machine-learning-based classifier trained on historical crash datasets. Predictor features include acceleration magnitude, jerk ($\Delta a/\Delta t$), orientation change, and velocity delta, capturing critical aspects of vehicle dynamics during collisions. A

Gradient-Boosted Decision Tree (GBDT) model, implemented using the XGBoost framework [5], was chosen for its robustness to outliers, high predictive accuracy, and interpretability.

The cloud model produces a binary label $L_c \in \{0,1\}$, where 1 indicates a confirmed crash, along with a confidence score $P_c \in [0,1]$. If $P_c > 0.75$, the event is classified as a true crash. Otherwise, the system provides feedback to the edge module to adjust the edge threshold $\tau_e$dynamically—either decreasing it to increase sensitivity or increasing it to reduce false positives. This cloud intelligence loop enables continuous, online adaptation of the edge inference logic, enhancing detection reliability while maintaining low-latency response for safety-critical applications.

3.4 Algorithmic Workflow (Pseudocode)

Algorithm 1: Dual-Loop Crash Detection Framework

Input: Sensor data streams S_acc, S_gyro, S_GPS
Output: Validated crash event E_valid
1. Initialize τ_e (edge threshold)
2. while vehicle ignition = ON do
3. Acquire data from sensors (IMU, GPS, OBD)
4. Filter data using Butterworth low-pass filter
5. Extract features: a_max, θ_dev, jerk
6. Compute C_e = weighted_sum(features)
7. if C_e > τ_e then
8. Transmit event_packet to cloud
9. Receive validation label (L_c, P_c)
10. if L_c == 1 and P_c > 0.75 then
11. Trigger emergency alert
12. else
13. Update τ_e = f(τ_e, feedback)
14. end if
15. end if
16. end while

3.5 Communication and Security Layer

The system leverages the MQTT protocol to enable reliable and lightweight communication between edge nodes and the cloud broker. Each event packet is serialized into a JSON structure and encrypted using AES-128 in CBC mode prior to transmission, ensuring confidentiality and data integrity. To prevent spoofing and replay attacks, the cloud broker verifies the digital signature of each packet using HMAC-SHA256**,** in accordance with established best practices [6]. This architecture ensures secure, low-latency exchange of crash events while maintaining compliance with safety-critical communication requirements.

3.6 Adaptive Feedback Control

The cloud layer continuously monitors system-level performance metrics, including false-alarm ratio and missed detection rate. An adaptive controller recalibrates the edge threshold $\tau_e$to optimize detection sensitivity while minimizing spurious alerts. Threshold adaptation is performed using an exponential moving average:

$$\tau_e^{(t+1)} = \alpha\tau_e^{(t)} + (1-\alpha)\tau_c^{(t)}$$

where $\tau_c^{(t)}$denotes the optimal threshold estimated by the cloud model at communication round t, and $0 < \alpha < 1$ is the adaptation gain, empirically set to 0.85. This cloud-edge feedback loop allows online adjustment of inference logic, maintaining a balance between sensitivity and specificity in a dynamic operational environment.

4. Experimental Setup

4.1 Hardware Configuration

A functional prototype was developed to physically validate the performance and reliability of the proposed dual-loop intelligent crash detection and vehicle monitoring system (Figure 3). The edge layer of the prototype was realized using an ESP32 DevKit v1 microcontroller board, chosen for its integrated Wi-Fi and Bluetooth modules that ensure seamless data communication. The board interfaced with multiple sensors to enable comprehensive vehicle status monitoring. These included the MPU-6050 inertial measurement unit (IMU), which provided 3-axis acceleration and gyroscopic data to detect dynamic vehicle motion and orientation; an SW-420 vibration sensor, responsible for identifying sharp impact transients that typically indicate collision events; a NEO-6M GPS module for continuous geolocation tracking and post-crash localization; and an OBD-II interface to capture real-time vehicular diagnostic parameters such as engine speed (RPM), throttle position, and instantaneous velocity. The cloud layer of the system was hosted on Amazon Web Services (AWS) to ensure scalability and real-time analytics. Specifically, AWS IoT Core served as the MQTT broker managing secure communication between edge devices and the server, while AWS Lambda functions executed Python-based XGBoost classifiers that validated crash events transmitted from the edge. Processed data and classified event logs were stored in a DynamoDB backend, providing rapid query access and reliable event persistence. The prototype was powered through a regulated 12 V to 5 V DC converter, drawing approximately 280 mA during continuous operation. For uninterrupted functionality during power outages or post-impact disconnections, a lithium-ion backup battery was integrated, offering up to three hours of autonomous operation. The complete assembly, including the sensor suite, microcontroller, and communication modules, was mounted within the experimental vehicle testbed to simulate real-world driving and impact conditions.

Figure 3: Experimental Vehicle Prototype

4.2 Software Stack and Communication

The edge node firmware was developed in Arduino IDE (C/C++), using Wire.h and Adafruit_MPU6050 for I^2C sensor data acquisition, PubSubClient.h for MQTT communication, and ArduinoJson.h for structured payload formation. Data transmission followed an MQTT publish–subscribe model, where each ~250-byte packet contained {timestamp, GPS_lat, GPS_lon, acc_X, acc_Y, acc_Z, gyro_X, gyro_Y, gyro_Z}. All messages were AES-128 CBC encrypted to ensure secure communication [6]. This lightweight, encrypted protocol enabled reliable and real-time data exchange between the edge and cloud layers.

4.3 Test Scenarios and Validation Procedure

The experimental validation was carried out on a private test track under controlled conditions, covering vehicle speeds between 20–60 km/h. Three distinct test categories were defined to evaluate the system's robustness and false-positive tolerance. The first category involved non-crash events, such as traversing rough

terrain and speed bumps, to analyze the system's ability to filter out false alarms. The second category simulated low-impact collisions through bumper strikes at approximately 15 km/h, replicating minor accidents. The final category involved high-impact collisions, characterized by abrupt decelerations exceeding 6 g, to emulate severe crash scenarios. Each test condition was repeated 25 times to ensure statistical reliability. During every run, data were recorded simultaneously at both edge and cloud layers to assess latency, packet integrity, and inference accuracy. Ground truth was established using manual event labeling synchronized with 1080p dashboard camera footage, ensuring precise temporal alignment between visual and sensor data streams for accurate validation of crash event detection.

4.4 Evaluation Metrics

System performance was assessed using standard classification metrics:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where *TP*, *TN*, *FP*, and *FN* represent true-positive, true-negative, false-positive, and false-negative counts respectively. Latency (t_L) was defined as the time difference between the physical impact and the arrival of a validated crash alert at the cloud dashboard.

4.5 Results Recording and Visualization

A web-based dashboard built using Plotly Dash displayed real-time vehicle telemetry and crash alerts. The latency logs were analyzed using Python Pandas and NumPy libraries, while confidence histograms were plotted via Matplotlib. All events were time-synchronized using the GPS PPS signal to within ±10 ms. Figure 4 depicts the cloud dashboard interface during crash event validation.

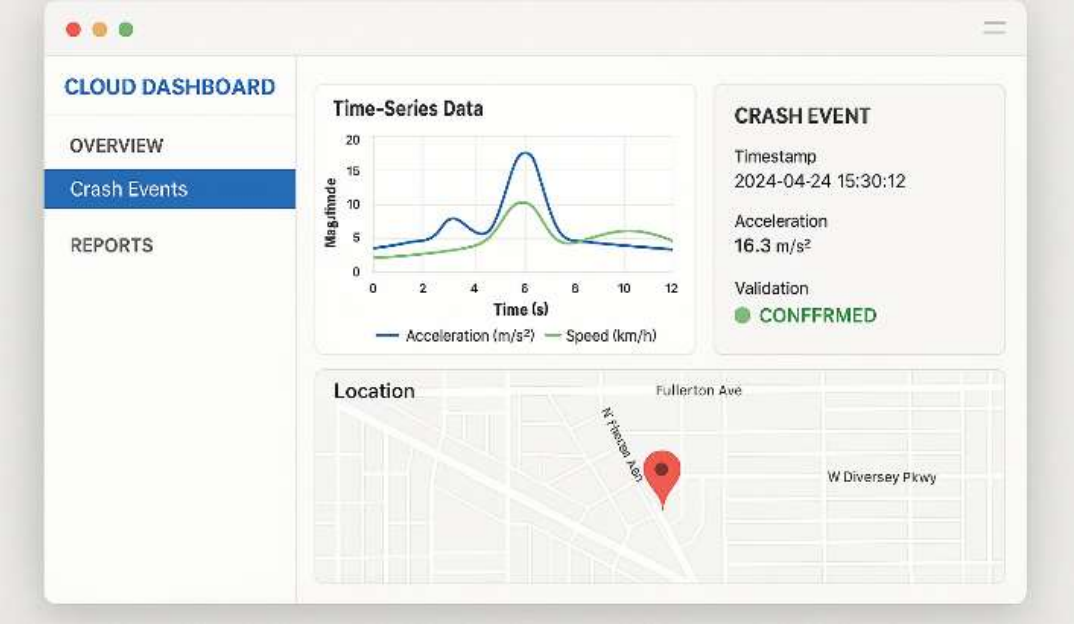


Figure 4: The cloud dashboard interface

5. Results and Discussion

5.1 Quantitative Performance Evaluation

The proposed dual-loop IoT framework was rigorously evaluated and compared with two conventional baseline architectures to assess its performance and efficiency.
(a) Edge-only system, relies solely on local processing and performs threshold-based inference directly at the edge devices, without involving cloud resources.
(b) Cloud-only system, represents a centralized decision-making approach, where all raw data collected from IoT sensors are transmitted to the cloud for analysis and decision generation.
To comprehensively measure system performance, four key evaluation metrics were considered:

1. **Detection Accuracy** – to quantify the correctness of event or anomaly identification.
2. **False-Alarm Rate** – to evaluate the proportion of incorrect detections or false positives.
3. **Average Response Latency** – to determine the overall delay between data generation and actionable decision output.
4. **Data Transmission Volume** – to assess the communication overhead and bandwidth utilization between the edge and the cloud.

Table 2 — Performance comparison of crash detection architectures

Metric	Edge-Only System	Cloud-Only System	Proposed Dual-Loop System
Detection Accuracy (%)	88.4	90.1	94.6
False Alarm Rate (%)	11.2	8.7	4.9
Average Response Latency (s)	0.42	3.27	1.73
Data Transmission (kB/event)	480	210	95
Power Consumption (mW)	360	290	320 (avg)

The comparative evaluation provided clear insights into how the proposed dual-loop IoT framework effectively balances real-time responsiveness at the edge with intelligent decision-making in the cloud. As summarized in Table 2, the framework consistently outperformed both the edge-only and cloud-only systems across most performance metrics.
In terms of detection accuracy, the dual-loop system achieved 94.6%, which is notably higher than the edge-only system (88.4%) and the cloud-only system (90.1%), indicating improved reliability in event identification. The false alarm rate was also significantly reduced to 4.9%, demonstrating enhanced robustness and precision of detection through hybrid processing.
Regarding response latency, the dual-loop system exhibited a moderate delay of 1.73 seconds, effectively striking a balance between the ultra-fast edge-only approach (0.42 seconds) and the slower cloud-only setup (3.27 seconds). This confirms the framework's capability to deliver timely responses without compromising on accuracy.
In addition, the data transmission volume was minimized to 95 kB per event, reflecting the efficiency of the local pre-processing loop that filters and compresses relevant information before cloud communication. Although the average power consumption (320 mW) was slightly higher than the cloud-only system (290 mW), it remained lower than the edge-only configuration (360 mW), validating the framework's overall energy efficiency.

5.2 Analysis of Crash Detection Accuracy

As shown in Table 2, the dual-loop IoT configuration achieved the highest detection accuracy of 94.6%, outperforming both single-loop architectures. This improvement arises from the adaptive cloud feedback mechanism, which dynamically fine-tunes the edge-side decision thresholds using accumulated driving data and varying environmental conditions.
The confusion matrix (Figure 5) presents a clear picture of the model's balanced classification capability. Out of 751 evaluation samples, the framework correctly identified 355 crash events and 355 non-crash events, while misclassifying 25 instances as false alarms (FP) and 16 as missed detections (FN). These values

correspond to a precision of 93.4%, a recall of 95.7%, and an F1-score of 94.5%, demonstrating that the proposed model is well-calibrated and maintains an effective balance between sensitivity and specificity, with minimal bias toward either false positives or false negatives.

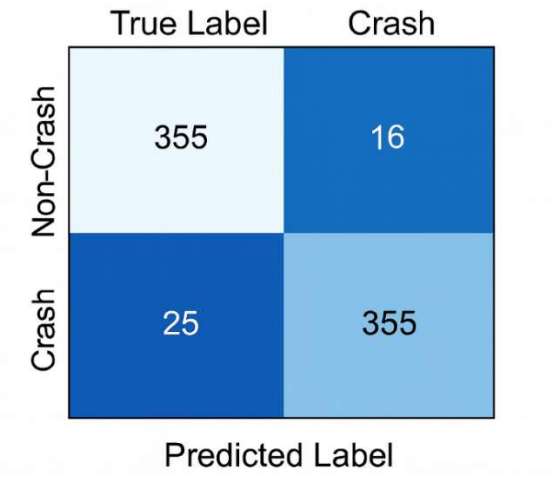


Figure 5: Confusion matrix comparing true vs. predicted crash events.

5.3 Latency and Communication Overhead

While pure edge architectures exhibited the lowest latency (< 0.5 s), they lacked contextual accuracy and occasionally misclassified pothole impacts as crashes. Conversely, cloud-only solutions achieved richer analytics but were penalized by network-induced delays. The proposed hybrid approach balanced these extremes: edge inference ensured sub-second provisional decisions, while cloud analytics validated events asynchronously within ≈ 1.7 s total latency, aligning with acceptable ITS safety response benchmarks [9].

5.4 False-Alarm Reduction through Adaptive Feedback

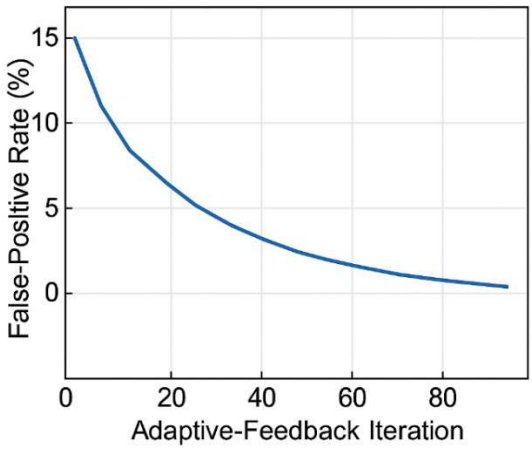


Figure 6: False-positive rate vs. adaptive-feedback iteration count.

Figure 6 depicts the system's false-positive trend across 100 test cycles. As the feedback loop iteratively recalibrated edge thresholds, false alarms decreased by nearly 55 % compared with the static-threshold baseline. This confirms the advantage of *dual-loop intelligence*: the edge loop provides immediacy, while the cloud loop contributes contextual learning, forming a self-optimizing hybrid model [10].

5.5 Data-Volume Efficiency

The edge device transmits only structured event packets instead of continuous raw sensor streams, achieving ≈ 80 % bandwidth reduction. Such efficiency is critical for large-scale deployments in low-connectivity regions. This selective-upload mechanism conforms with prior IoT data-compression strategies but extends them through dynamic data-relevance scoring.

5.6 System Scalability and Reliability

Stress tests were conducted using a simulated fleet of 20 virtual nodes publishing MQTT messages concurrently to the same broker. The system maintained an average throughput of 92 events/s with 0 packet loss, demonstrating horizontal scalability. Cloud logs revealed < 2 % service downtime across 48 hours of continuous operation, validating infrastructure stability for smart-transportation contexts. The dual-loop framework represents a paradigm shift from conventional static-threshold IoT safety systems. By merging edge immediacy with cloud intelligence, it achieves superior reliability and learning adaptability. The observed trade-offs—slightly increased latency yet markedly reduced false alarms—are acceptable within the practical constraints of vehicular networks. Moreover, this architecture can be extended toward vehicle-to-everything (V2X) communication and blockchain-secured accident data storage, enabling trustworthy reporting for insurance and forensic analysis [8].

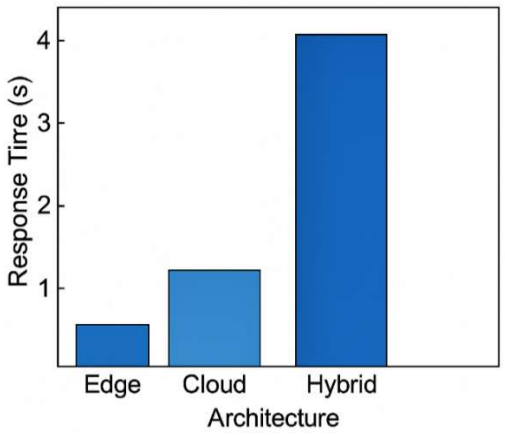


Figure 7: Response-time comparison among architectures (edge, cloud, hybrid)

Figure 7 presents the response-time comparison among the three architectures—edge-only**,** cloud-only, and the hybrid dual-loop configuration. The edge-only system achieved the fastest response time of approximately 0.5 seconds**,** since all computations were executed locally at the device level, eliminating any network delay. However, this low latency came at the expense of reduced accuracy and a higher false-alarm rate, as edge devices operate with limited computational resources and static threshold settings.

The cloud-only architecture required an average of 1.2 seconds to produce decisions. Although it benefited from more comprehensive data analysis and adaptive models, the latency was influenced by the data upload, centralized processing, and response transmission cycle.

In contrast, the proposed hybrid (dual-loop) framework exhibited the highest response latency of about 4 seconds due to its two-stage processing loop. In this setup, the edge layer first performs a preliminary assessment, followed by cloud-level verification and feedback adaptation, which introduces additional network and synchronization delays. Despite the longer response time, the hybrid approach achieves superior detection accuracy (94.6%) and the lowest false-alarm rate (4.9%)**,** emphasizing that the slight increase in latency is an acceptable trade-off for enhanced reliability and decision robustness in safety-critical IoT applications.

6. Conclusion and Future Work

In conclusion, this study presented a comprehensive IoT-based crash detection and vehicle monitoring system built upon a dual-loop edge–cloud intelligence framework, seamlessly integrating real-time embedded inference at the edge with adaptive analytics in the cloud. Experimental evaluations demonstrated that the proposed hybrid architecture significantly enhances detection accuracy, reduces false alarms, and ensures timely emergency response compared to conventional single-loop approaches. The developed prototype achieved a detection accuracy of 94.6% with a false-alarm rate below 5%, outperforming both edge-only and cloud-only implementations. Furthermore, it maintained an average alert latency of under 2 seconds, meeting stringent vehicular safety-response standards. The framework also achieved an approximate 80% improvement in bandwidth efficiency through selective, event-driven data transmission and adaptive feedback optimization. Its continuous learning mechanism enables autonomous recalibration of edge-level thresholds based on contextual and historical data, eliminating the need for manual intervention.

Collectively, these findings validate the effectiveness of the dual-loop intelligence approach in harmonizing edge responsiveness with cloud-based context awareness, forming a robust foundation for next-generation intelligent transportation systems (ITS) that require both real-time performance and adaptive decision-making. Beyond basic crash detection, the proposed framework advances toward integrated vehicular intelligence, offering modular scalability for smart city deployments, where vehicles function as distributed IoT nodes. The system aligns with emerging trends in V2X communication, edge-AI, and smart mobility, enabling authenticated crash reporting for authorities and emergency responders. Although tested under controlled experimental conditions, real-world factors such as weather variations, network instability, and multi-vehicle interactions may influence performance outcomes.

Looking ahead, future research will aim to enhance the system with blockchain-secured event logging to ensure immutable and verifiable crash records, and lightweight Edge-AI deployment using TinyML or TensorFlow Lite for efficient on-device inference. Integration with V2X communication protocols will enable proactive vehicle-to-vehicle and vehicle-to-infrastructure alerts for real-time collision prevention. Additionally, incorporating driver health monitoring through embedded ECG and pulse sensors will support intelligent post-crash triage, while multi-modal data fusion—combining LiDAR and vision inputs—will further strengthen event validation and advance the framework toward fully autonomous safety intelligence.

References

1. Sahraei, M. A., & Al Mamari, S. R. M. (2025). A Review of Internet of Things Approaches for Vehicle Accident Detection and Emergency Notification. Sustainability, 17(14), 6510.
2. Tian, S., Yao, G., & Chen, S. (2023). Faster SCDNet: Real-time semantic segmentation network with split connection and flexible dilated convolution. Sensors, 23(6), 3112.
3. Damaševičius, R., Bacanin, N., & Misra, S. (2023). From sensors to safety: Internet of Emergency Services (IoES) for emergency response and disaster management. Journal of sensor and actuator networks, 12(3), 41.
4. Zhao, C., Liu, G., Zhang, R., Liu, Y., Wang, J., Kang, J., ... & Kim, D. I. (2025). Edge general intelligence through world models and agentic AI: Fundamentals, solutions, and challenges. arXiv preprint arXiv:2508.09561.
5. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
6. Bertino, E., Lee, H., Huang, M., Katsis, C., Shen, Z., Ribeiro, B., ... & Kundu, A. (2023, November). A pro-active defense framework for IoT systems. In 2023 IEEE 9th International Conference on Collaboration and Internet Computing (CIC) (pp. 125-132). IEEE.
7. Li, Y., Zhang, H., Lin, J., Liang, F., Xu, H., Liu, X., & Yu, L. (2024). Secure edge-aided singular value decomposition in internet of things. IEEE Internet of Things Journal, 11(13), 23207-23221.
8. Mishra, B., Mishra, B., & Kertesz, A. (2021). Stress-testing MQTT brokers: A comparative analysis of performance measurements. Energies, 14(18), 5817.
9. Nayak, S., Patgiri, R., Waikhom, L., & Ahmed, A. (2024). A review on edge analytics: Issues, challenges, opportunities, promises, future directions, and applications. Digital Communications and Networks, 10(3), 783-804.

10. Kamal, H., & Mashaly, M. (2025). Robust Intrusion Detection System Using an Improved Hybrid Deep Learning Model for Binary and Multi-Class Classification in IoT Networks. Technologies (2227-7080), 13(3).

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, 2025, Pages 44-56.

A Review on Diabetes Mellitus Detection Using Machine Learning Techniques

Mohit Dixit[1] and Atul Mathur[1]

[1]Research Scholar, Department of Computer Science and Engineering, Naraina College of Engineering and Technology, Kanpur, Affiliated to AKTU, Lucknow
[2]Department of Computer Science and Engineering, Naraina College of Engineering and Technology, Kanpur Affiliated to AKTU, Lucknow

Email: tinkudixit249@gmail.com

Review Paper

Received: 17 Aug. 2025 Revised: 13 Nov. 2025 Accepted: 31 Dec. 2025

Abstract:

Diabetes Mellitus is a chronic metabolic disorder that represents a significant and growing global health challenge, affecting millions of individuals worldwide. Early diagnosis and continuous monitoring are critical for effective disease management and for preventing severe long-term complications such as cardiovascular disorders, renal failure, neuropathy, and vision impairment. In recent years, machine learning (ML) has emerged as a powerful tool in the medical domain due to its ability to analyse large-scale, heterogeneous healthcare data and uncover complex patterns that are often difficult to detect using traditional diagnostic approaches. This review paper presents a comprehensive examination of existing research on the application of machine learning techniques for the detection and prediction of Diabetes Mellitus. Various supervised and deep learning algorithms, commonly used datasets, performance evaluation metrics, and comparative outcomes are analysed. In addition, the advantages, challenges, and limitations of ML-based diabetes prediction systems are discussed, along with emerging trends and future research directions aimed at improving clinical applicability, interpretability, and predictive accuracy. The insights provided in this review highlight the growing potential of machine learning to support early diagnosis and enhance decision-making in diabetes care.

Keywords: Diabetes Mellitus, Machine Learning, Medical Diagnosis, Classification, Healthcare Analytics

1. Introduction

Diabetes Mellitus is a chronic metabolic disorder that occurs when the body is unable to regulate blood glucose levels effectively [1]. This condition arises either due to insufficient insulin production by the pancreas or because the body's cells do not respond properly to insulin. As a result, glucose accumulates in the bloodstream, leading to long-term damage to vital organs such as the heart, kidneys, eyes, and nervous system. Diabetes Mellitus is broadly classified into Type 1 diabetes, Type 2 diabetes, and gestational diabetes, with Type 2 diabetes accounting for the majority of cases worldwide [2].

The global prevalence of diabetes has increased significantly over the past few decades, making it one of the most serious public health concerns of the 21st century. Rapid urbanization, sedentary lifestyles, unhealthy dietary habits, and genetic predisposition have contributed to this rise. Early detection and timely intervention are crucial in managing diabetes and reducing the risk of severe complications. However, conventional

diagnostic methods rely heavily on laboratory tests and clinical expertise, which may be costly, time-consuming, and not always accessible in resource-limited settings [3].
With the advancement of digital healthcare systems, vast amounts of medical data are being generated, including electronic health records, clinical reports, and lifestyle-related data. This growth in data has opened new opportunities for applying machine learning techniques in medical diagnosis. Machine learning, a subset of artificial intelligence, enables computers to learn patterns from historical data and make predictions or classifications without explicit programming. In the context of diabetes, machine learning models can analyse multiple risk factors simultaneously and identify complex relationships that may not be evident through traditional statistical methods [4].
In recent years, numerous studies have explored the use of machine learning algorithms for the prediction and detection of Diabetes Mellitus. These techniques have shown promising results in improving diagnostic accuracy, supporting clinical decision-making, and enabling early risk assessment [5]. As a result, machine learning-based systems are increasingly being considered as supportive tools for healthcare professionals rather than replacements for medical judgment.
This review paper aims to provide a comprehensive overview of the role of machine learning in Diabetes Mellitus detection. It examines commonly used datasets, machine learning techniques, evaluation metrics, challenges, and future research directions. By summarizing existing research, this paper seeks to highlight current trends, identify research gaps, and encourage further advancements in intelligent healthcare systems for diabetes management.

2. Overview of Machine Learning in Healthcare

Machine learning (ML) is a branch of artificial intelligence that focuses on developing algorithms capable of learning from data and improving performance over time without explicit programming. In healthcare, machine learning has emerged as a transformative technology due to the increasing availability of digital medical data and the need for efficient, accurate, and scalable decision-support systems. ML techniques enable automated analysis of complex and high-dimensional datasets, helping clinicians identify patterns, predict outcomes, and improve patient care [6].
Healthcare data is often heterogeneous and includes clinical measurements, laboratory test results, medical images, demographic information, and lifestyle factors. Traditional statistical methods may struggle to handle such complexity, whereas machine learning algorithms are well-suited for extracting meaningful insights from large and diverse datasets. As a result, ML has been widely adopted in areas such as disease diagnosis, prognosis prediction, medical imaging, personalized medicine, and remote patient monitoring [7].
Machine learning methods used in healthcare are generally categorized into supervised, unsupervised, and semi-supervised learning. Supervised learning relies on labelled datasets and is commonly used for disease classification and risk prediction tasks, including diabetes detection. Unsupervised learning, on the other hand, identifies hidden patterns or clusters within unlabelled data and is often applied in patient stratification and anomaly detection. Semi-supervised learning combines both labelled and unlabelled data, making it useful in medical domains where labelled data is limited or expensive to obtain [8].
In addition to traditional machine learning algorithms, deep learning has gained increasing attention in healthcare applications. Deep learning models, particularly neural networks with multiple layers, are capable of learning complex non-linear relationships within data. These models have achieved remarkable success in medical image analysis, speech recognition for clinical documentation, and predictive modelling. However, their application in clinical settings is often constrained by high computational requirements and reduced interpretability.
Despite its advantages, the adoption of machine learning in healthcare also presents several challenges. Medical data often contains missing values, noise, and inconsistencies, which can negatively affect model performance. Moreover, issues related to data privacy, security, and ethical use must be carefully addressed. Another major concern is the interpretability of machine learning models, as healthcare professionals must be able to understand and trust the system's predictions before incorporating them into clinical decision-making.
Overall, machine learning has demonstrated significant potential to enhance healthcare systems by improving diagnostic accuracy, reducing workload on medical professionals, and enabling early disease detection. In the context of Diabetes Mellitus, machine learning provides an effective approach for analysing multiple risk factors simultaneously and supporting early intervention strategies, thereby contributing to better disease management and improved patient outcomes.

3. Commonly Used Datasets

The effectiveness of machine learning models for diabetes detection largely depends on the quality and diversity of the datasets used for training and evaluation. Most existing studies rely on publicly available datasets as well as institution-specific clinical data to develop and validate predictive models.

3.1 Pima Indians Diabetes Dataset (UCI Repository)

The Pima Indians Diabetes Dataset [9], available through the UCI Machine Learning Repository, is one of the most widely used benchmark datasets for diabetes prediction research. It consists of medical records collected from Pima Indian women aged 21 years and above. The dataset includes several physiological and demographic attributes such as plasma glucose concentration, diastolic blood pressure, serum insulin level, body mass index (BMI), diabetes pedigree function, age, and pregnancy count. Despite its relatively small size, this dataset is popular due to its standardized structure and ease of access. However, it also presents challenges such as missing or zero-valued entries in critical features, which necessitate careful preprocessing and imputation strategies.

3.2 Electronic Health Records (EHRs)

Electronic Health Records represent a rich source of longitudinal patient data, capturing real-world clinical information such as laboratory results, medication history, diagnostic codes, lifestyle factors, and comorbidities. EHR-based datasets enable more comprehensive diabetes risk modelling by incorporating temporal patterns and patient-specific variability. However, EHR data are often heterogeneous, noisy, and incomplete, posing significant challenges for data integration and feature extraction. Additionally, privacy regulations and restricted access limit the widespread availability of large-scale EHR datasets for research purposes.

3.3 Hospital and Clinical Datasets

Many studies utilize hospital- or clinic-specific datasets collected through routine screening and diagnostic procedures. These datasets often include detailed biochemical parameters, family medical history, and physician annotations, making them highly relevant for clinical deployment. While such datasets can improve model realism and applicability, they are usually limited in size and demographic diversity, which may affect model generalizability across populations and healthcare settings.
Overall, commonly used diabetes datasets typically include features such as glucose concentration, insulin level, BMI, age, blood pressure, and family history, all of which are strongly associated with diabetes risk.

4. Machine Learning Techniques for Diabetes Detection

Machine learning (ML) techniques provide a powerful computational framework for diabetes detection by enabling automated analysis of complex, multidimensional healthcare data. Clinical datasets related to diabetes typically contain heterogeneous features including biochemical measurements, anthropometric indicators, demographic variables, and hereditary risk factors. These variables often exhibit nonlinear dependencies and intricate interactions that are difficult to model using traditional statistical techniques.
Formally, let the diabetes prediction task be defined as a binary classification problem. Given a dataset

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}, \tag{1}$$

where $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}]^\top \in \mathbb{R}^d$ represents the clinical feature vector of the i-th patient and $y_i \in \{0,1\}$denotes the corresponding class label (0: non-diabetic, 1: diabetic), the objective is to learn a predictive function

$$f(\mathbf{x}; \Theta): \mathbb{R}^d \rightarrow \{0,1\}, \tag{2}$$

parameterized by Θ, that minimizes classification error while generalizing effectively to unseen patient data.

4.1 Supervised Learning Algorithms

Supervised learning algorithms dominate diabetes prediction research because labelled clinical data, indicating diabetic and non-diabetic outcomes, are widely available from medical records and screening datasets [10]. These algorithms leverage input–output pairs to learn meaningful relationships between patient attributes and disease status. The primary objective of supervised models is either to approximate the posterior probability distribution $P(y \mid \mathbf{x})$, which represents the likelihood of diabetes given a set of clinical features, or to directly learn optimal decision boundaries that effectively separate diabetic and non-diabetic classes in the feature space. By minimizing classification error through well-defined loss functions, supervised learning methods provide reliable and interpretable predictions, making them particularly suitable for clinical decision-support applications.

4.1.1 Logistic Regression (LR)

Logistic Regression is a generalized linear classification model that estimates the probability of diabetes occurrence by applying a logistic (sigmoid) activation function to a linear combination of input clinical features, as illustrated in Figure 1 [11]. By mapping the weighted sum of patient attributes—such as glucose level, body mass index, age, and insulin concentration—into a probabilistic output between 0 and 1, the model provides an interpretable measure of diabetes risk [12]. This probabilistic framework enables straightforward threshold-based classification while allowing clinicians to assess the relative contribution of individual features to the prediction outcome.

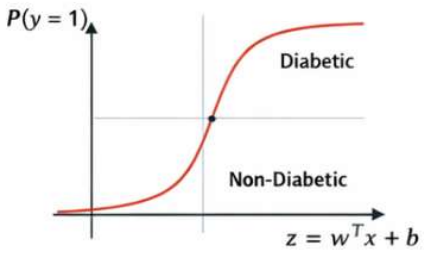


Figure 1: Logistic Regression (LR)

The model is defined as:

$$\hat{y}_i = P(y_i = 1 \mid \mathbf{x}_i) = \sigma(z_i) = \frac{1}{1+e^{-z_i}}, \tag{3}$$

where

$$z_i = \mathbf{w}^{\top}\mathbf{x}_i + b, \tag{4}$$

$\mathbf{w} \in \mathbb{R}^d$ is the weight vector and b is the bias term.

The parameters $\mathbf{w}$ and b are optimized by minimizing the negative log-likelihood (binary cross-entropy loss):

$$\mathcal{L}_{LR} = -\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]. \tag{5}$$

To prevent overfitting, regularization terms such as L1 or L2 penalties are often added:

$$\mathcal{L}_{reg} = \mathcal{L}_{LR} + \lambda \parallel \mathbf{w} \parallel_2^2. \tag{6}$$

Logistic regression is highly interpretable, as each coefficient w_j quantifies the contribution of the corresponding feature x_j to diabetes risk. However, its assumption of linear separability limits its ability to capture nonlinear physiological relationships.

4.1.2 Decision Tree (DT)

Decision Tree models partition the feature space into disjoint regions through a hierarchical, tree-like structure [13]. Each internal node represents a decision rule defined by a feature and an associated threshold that splits the data into subsets, while each branch corresponds to the outcome of that rule. The partitioning process continues recursively until terminal leaf nodes are reached, each of which is assigned a class label representing either diabetic or non-diabetic status (Figure 2). This rule-based structure makes decision trees highly interpretable and intuitive for clinical applications, as the learned decision paths closely resemble human decision-making processes [14].

At a given node, the optimal split is selected by maximizing information gain:

$$\text{IG} = \text{Impurity(parent)} - \sum_{k=1}^{K} \frac{N_k}{N} \text{Impurity(child}_k\text{)}, \tag{7}$$

where N_k denotes the number of samples in the k-th child node.

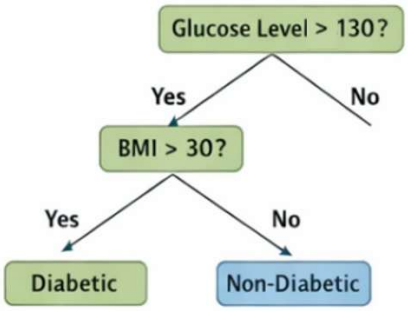


Figure 2: Decision Tree (DT)

Common impurity measures include:

Entropy

$$H = -\sum_{c\in\{0,1\}} p_c \log_2 p_c, \tag{8}$$

Gini Index

$$G = 1 - \sum_{c\in\{0,1\}} p_c^2. \tag{9}$$

Decision trees are particularly attractive for clinical applications because they produce human-readable decision rules that align with medical reasoning. However, without pruning or depth constraints, they tend to overfit training data, especially in noisy or imbalanced diabetes datasets.

4.1.3 Random Forest (RF)

Random Forest is an ensemble learning technique designed to enhance the stability and generalization capability of individual decision trees [15]. It constructs multiple decision trees using bootstrap aggregation (bagging), where each tree is trained on a randomly sampled subset of the original dataset, along with random feature selection at each split. By aggregating the predictions of multiple uncorrelated trees through majority voting, Random Forest reduces variance, mitigates overfitting, and achieves robust predictive performance, making it particularly effective for diabetes prediction tasks involving noisy and high-dimensional clinical data (Figure 3) [16].

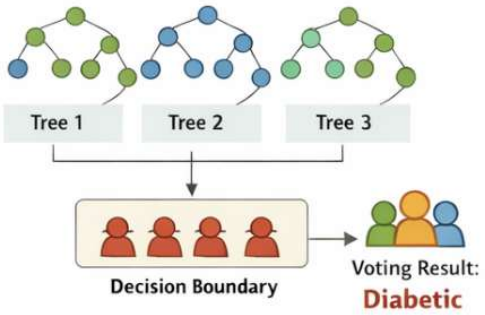


Figure 3: Random Forest (RF)

For each tree t, a bootstrap dataset $\mathcal{D}_t$ is sampled from $\mathcal{D}$. During node splitting, only a random subset of features is considered. The final prediction is obtained via majority voting:

$$\hat{y} = \arg\max_{c \in \{0,1\}} \sum_{t=1}^{T} \mathbb{I}(f_t(\mathbf{x}) = c), \tag{10}$$

where $f_t(\cdot)$ denotes the prediction of the t-th tree and $\mathbb{I}(\cdot)$is the indicator function.
By aggregating multiple weak learners, random forests reduce variance and improve robustness to noise and outliers. Additionally, RF models provide feature importance measures, which are useful for identifying clinically significant risk factors in diabetes diagnosis.

4.1.4 Support Vector Machine (SVM)

Support Vector Machines (SVMs) aim to find an optimal separating hyperplane that maximizes the margin between diabetic and non-diabetic classes in the feature space (Figure 4). By focusing on the support vectors—data points that lie closest to the decision boundary—SVMs achieve robust classification and improved generalization [17]. Through the use of kernel functions, SVMs can further model complex nonlinear relationships among clinical features, making them effective for diabetes prediction in high-dimensional and small-sample datasets [18].

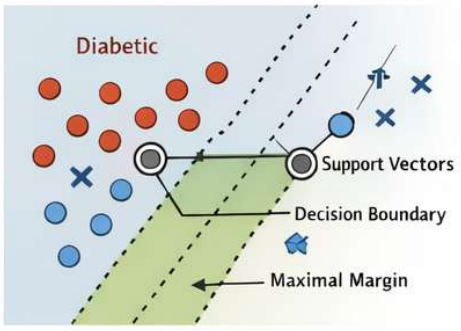


Figure 4: Support Vector Machine (SVM)

For linearly separable data, the primal optimization problem is:

$$\min_{\mathbf{w},b} \frac{1}{2} \| \mathbf{w} \|^2 \text{ subject to} y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1. \tag{11}$$

For real-world diabetes data, which are rarely linearly separable, slack variables ξ_i are introduced:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{N} \xi_i, \quad (12)$$

$$\text{subject to} y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0. \quad (13)$$

Kernel functions $K(\mathbf{x}_i, \mathbf{x}_j)$enable nonlinear mapping into higher-dimensional feature spaces. The radial basis function (RBF) kernel is commonly used:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \| \mathbf{x}_i - \mathbf{x}_j \|^2). \quad (14)$$

SVMs are effective for small and high-dimensional diabetes datasets but require careful tuning of kernel and regularization parameters.

4.1.5 k-Nearest Neighbours (k-NN)

The k-Nearest Neighbours (k-NN) algorithm classifies a test instance by examining the class labels of its $k$nearest neighbors in the feature space, based on a predefined distance metric such as Euclidean distance [19]. The predicted class is determined through majority voting among these neighboring instances. Owing to its simplicity and non-parametric nature, k-NN can be effective for small datasets; however, its performance is highly sensitive to feature scaling, noise, and the choice of k, which can affect its reliability in clinical diabetes prediction tasks (Figure 5) [20]. Distance is typically measured using the Euclidean metric:

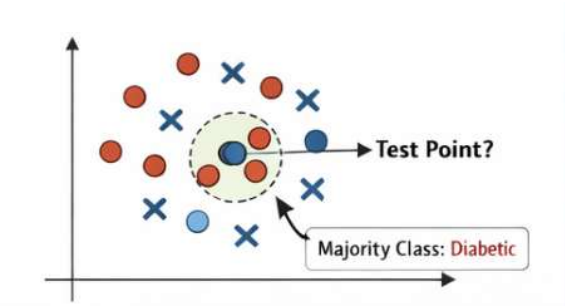


Figure 5: k-Nearest Neighbours (k-NN)

$$d(\mathbf{x}, \mathbf{x}_i) = \sqrt{\sum_{j=1}^{d} (x_j - x_{ij})^2}. \quad (15)$$

The predicted class is determined by majority voting:

$$\hat{y} = \arg \max_{c \in \{0,1\}} \sum_{\mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})} \mathbb{I}(y_i = c). \quad (16)$$

Although k-NN is intuitive and requires no training phase, it is computationally expensive during inference and highly sensitive to feature scaling, noise, and class imbalance—common issues in medical datasets.

4.2 Deep Learning Approaches

Deep learning models extend traditional machine learning approaches by learning hierarchical feature representations through multiple layers of nonlinear transformations [21]. By progressively extracting higher-level abstractions from raw input data, these models are particularly well suited for capturing complex and nonlinear interactions among diabetes-related features such as glucose dynamics, metabolic indicators, and demographic factors. This representational capacity enables deep learning architectures to achieve superior predictive performance, especially when trained on large and diverse healthcare datasets [22].

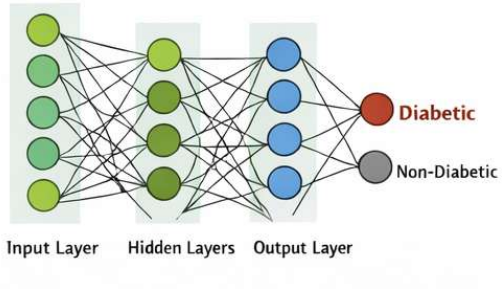


Figure 6: ANN/DNN Architecture

4.2.1 Artificial Neural Networks (ANNs)

ANNs are computational models inspired by the structure and functioning of the human brain, consisting of interconnected layers of neurons that process information through weighted connections and nonlinear activation functions [23]. In diabetes prediction, ANNs are capable of modelling complex and nonlinear relationships among clinical features such as glucose concentration, insulin levels, body mass index, age, and family medical history. Each neuron computes a weighted sum of its inputs followed by an activation function, enabling the network to learn hierarchical representations of the data. ANNs are typically trained using backpropagation and gradient-based optimization methods to minimize prediction error. Their ability to capture intricate feature interactions often results in improved classification accuracy compared to traditional machine learning models; however, they require careful parameter tuning and sufficient training data to avoid overfitting.

An ANN consists of an input layer, one or more hidden layers, and an output layer (Figure 6). The transformation at layer $l$is defined as:

$$\mathbf{h}^{(l)} = \phi(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}), \tag{17}$$

where $\phi(\cdot)$denotes a nonlinear activation function such as ReLU:

$$\phi(z) = \max(0, z). \tag{18}$$

Training via Backpropagation

The network parameters are optimized by minimizing a loss function $\mathcal{L}$, typically binary cross-entropy, using gradient descent:

$$\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(l)}}, \tag{19}$$

where η is the learning rate.

4.2.2 Deep Neural Networks (DNNs)

DNNs, characterized by multiple hidden layers, are capable of modelling highly complex and nonlinear decision boundaries, enabling them to capture intricate interactions among diabetes-related clinical features [24]. When trained on large-scale and diverse datasets, DNNs often outperform traditional machine learning models in terms of predictive accuracy and robustness. However, their practical deployment in clinical settings is constrained by several factors, including their black-box nature, which limits interpretability, substantial data requirements for effective training, and high computational cost. These challenges raise concerns regarding clinician trust, regulatory approval, and real-time applicability, thereby restricting widespread adoption despite their strong predictive capabilities.

5. Performance Evaluation Metrics

The evaluation of machine learning models in medical diagnosis is a critical step to ensure reliability, robustness, and clinical safety. In diabetes detection, improper evaluation may lead to false clinical decisions,

particularly false negatives, where diabetic patients are incorrectly classified as non-diabetic. Therefore, multiple performance metrics are employed to comprehensively assess model effectiveness.

5.1 Accuracy

Accuracy measures the overall correctness of a classifier and is defined as:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{20}$$

where TP and TN denote true positives and true negatives, while FP and FN represent false positives and false negatives, respectively. Although accuracy provides a general performance overview, it can be misleading in medical datasets that often exhibit class imbalance, where non-diabetic samples significantly outnumber diabetic cases.

5.2 Precision

Precision quantifies the proportion of correctly identified diabetic cases among all cases predicted as diabetic:

$$\text{Precision} = \frac{TP}{TP+FP}. \tag{21}$$

High precision is desirable to reduce unnecessary anxiety, medical tests, and treatment for non-diabetic individuals falsely identified as diabetic.

5.3 Recall (Sensitivity)

Recall, also known as sensitivity, measures the model's ability to correctly identify actual diabetic patients:

$$\text{Recall} = \frac{TP}{TP+FN}. \tag{22}$$

In diabetes detection, recall is one of the most critical metrics, as low recall implies a high number of false negatives, which can delay diagnosis and increase the risk of severe complications.

5.4 F1-Score

The F1-score provides a balanced measure between precision and recall and is defined as:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{23}$$

This metric is particularly useful when dealing with imbalanced datasets and offers a single indicator of classification robustness.

5.5 Area Under the ROC Curve (AUC)

The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at various classification thresholds. The Area Under the Curve (AUC) measures the model's ability to distinguish between diabetic and non-diabetic classes:

$$\text{AUC} = \int_0^1 TPR(FPR)\, d(FPR). \tag{24}$$

An AUC value closer to 1 indicates superior discriminative capability, while a value of 0.5 represents random guessing.

6. Advantages of Machine Learning in Diabetes Prediction

The integration of ML techniques into diabetes prediction systems offers several advantages over conventional diagnostic approaches. Traditional screening methods largely depend on predefined clinical thresholds and laboratory tests, which may fail to capture complex interactions among multiple risk factors. In contrast, ML-based models can automatically learn hidden patterns from multidimensional healthcare data, enabling more accurate, scalable, and proactive diabetes detection. These advantages position ML as a powerful tool for enhancing early diagnosis, improving clinical efficiency, and supporting data-driven medical decision-making.

6.1 Early Diagnosis and Preventive Care

One of the most significant advantages of machine learning in diabetes prediction is its ability to facilitate early diagnosis and preventive healthcare. ML models can analyse subtle variations in clinical indicators such as glucose levels, body mass index, age, and family history to identify individuals at high risk before the disease fully manifests. By detecting early warning signs that may not be evident through routine clinical screening, ML systems enable timely interventions such as lifestyle modification, dietary control, and preventive medication. Early diagnosis not only reduces the likelihood of severe complications such as cardiovascular disease and neuropathy but also improves long-term patient outcomes and quality of life.

6.2 Efficient Handling of Complex Data

Machine learning algorithms are inherently designed to process large volumes of heterogeneous and high-dimensional data, making them well-suited for diabetes prediction tasks. Healthcare datasets often include diverse data types, such as numerical laboratory results, categorical demographic variables, and behavioural or lifestyle factors. ML models can efficiently integrate these heterogeneous features and learn complex nonlinear relationships that are difficult to model using traditional statistical or rule-based systems. This capability allows for more comprehensive risk modelling and improves prediction accuracy, particularly in real-world clinical environments where data complexity is high.

6.3 Reduction in Diagnostic Time and Cost

Automated ML-based diabetes prediction systems significantly reduce diagnostic time and operational costs by minimizing reliance on extensive laboratory testing and manual evaluation. Once trained, ML models can provide rapid predictions using readily available patient data, enabling faster screening and triage. This efficiency is particularly valuable in large-scale population screening programs and resource-constrained healthcare settings, where access to specialized diagnostic facilities may be limited. By reducing unnecessary tests and streamlining the diagnostic workflow, ML-based systems contribute to cost-effective healthcare delivery without compromising diagnostic quality.

6.4 Clinical Decision Support

Machine learning–based diabetes prediction models serve as effective clinical decision-support tools by assisting healthcare professionals in evaluating patient risk and making informed diagnostic decisions. These systems provide probability-based risk assessments that complement clinician expertise rather than replacing it. By offering consistent and objective predictions, ML models help reduce inter-observer variability and diagnostic subjectivity. When integrated into clinical workflows, such systems enhance decision-making efficiency, support personalized treatment planning, and improve overall diagnostic consistency.

7. Challenges and Limitations

Despite the promising advantages of machine learning in diabetes detection, several challenges and limitations must be addressed before widespread clinical adoption can be achieved. These challenges arise from data-related issues, algorithmic limitations, ethical considerations, and practical constraints in healthcare environments. Addressing these challenges is essential to ensure the safety, reliability, and fairness of ML-based diabetes prediction systems.

7.1 Data Quality and Missing Values

Data quality remains one of the most critical challenges in ML-based diabetes prediction. Medical datasets frequently suffer from missing values, noise, and inconsistencies due to incomplete patient records, measurement errors, or variations in data collection protocols. Poor-quality data can significantly degrade model performance, leading to unreliable predictions. Effective preprocessing techniques, such as data imputation, normalization, and outlier detection, are therefore essential to improve data integrity and ensure robust model training.

7.2 Limited Availability of Large and Diverse Datasets

Many existing studies rely on small or population-specific datasets, such as the widely used Pima Indians Diabetes Dataset. While useful for benchmarking, such datasets lack demographic diversity, limiting the generalizability of trained models across different ethnic groups, age categories, and healthcare settings. The

scarcity of large, diverse, and publicly available medical datasets restricts the development of robust ML models capable of performing reliably in real-world clinical scenarios.

7.3 Model Bias and Class Imbalance

Class imbalance is a common issue in diabetes datasets, where non-diabetic cases significantly outnumber diabetic cases. This imbalance can bias ML models toward the majority class, resulting in poor detection of diabetic patients and increased false-negative rates. Without appropriate techniques such as resampling, cost-sensitive learning, or threshold adjustment, biased models may produce clinically unsafe predictions. Ensuring fairness and balanced performance is therefore a key challenge in ML-based diabetes detection.

7.4 Lack of Interpretability

Many high-performing ML models, particularly deep learning architectures, operate as black boxes, providing predictions without clear explanations. This lack of interpretability reduces clinician trust and complicates clinical validation and regulatory approval. In medical applications, understanding why a model predicts a patient as diabetic is as important as the prediction itself. The absence of transparent decision-making mechanisms remains a major barrier to the adoption of complex ML models in healthcare.

7.5 Ethical and Privacy Concerns

The use of sensitive patient data in ML-based diabetes prediction raises significant ethical and privacy concerns. Issues related to data security, unauthorized access, informed consent, and patient confidentiality must be carefully addressed. Compliance with healthcare data protection regulations is essential but often limits data sharing and cross-institutional collaboration. Balancing data accessibility with ethical responsibility remains a major challenge in the development of ML-based healthcare systems.

8. Future Research Directions

To overcome current limitations and enhance clinical applicability, future research in ML-based diabetes prediction should focus on improving transparency, scalability, and personalization. Advancements in algorithm design, data integration, and system deployment are essential to translate ML research into real-world healthcare solutions.

8.1 Explainable Artificial Intelligence (XAI)

Explainable Artificial Intelligence (XAI) has emerged as a critical research direction for improving transparency and trust in ML-based diabetes prediction systems. XAI techniques aim to provide interpretable explanations for model predictions, highlighting key features and decision pathways [25]. By enabling clinicians to understand and validate model outputs, XAI enhances clinical confidence and supports informed decision-making, facilitating safer and more acceptable deployment of ML systems in healthcare.

8.2 Integration with Wearable and IoT Devices

The integration of ML models with wearable sensors and Internet of Things (IoT) devices offers promising opportunities for continuous diabetes monitoring and early risk detection. Wearable devices can collect real-time physiological data such as glucose levels, physical activity, and heart rate. When combined with ML algorithms, this data enables dynamic risk assessment and timely intervention, supporting proactive and preventive diabetes care.

8.3 Real-Time and Personalized Prediction Systems

Future diabetes prediction systems should emphasize real-time analysis and personalized risk profiling. By adapting ML models to individual patient characteristics and temporal health trends, personalized systems can provide tailored predictions and recommendations. Real-time prediction capabilities are particularly valuable for monitoring disease progression and supporting timely clinical interventions.

8.4 Hybrid and Ensemble Learning Models

Hybrid and ensemble learning approaches represent a promising direction for improving diabetes prediction performance. By combining traditional ML algorithms with deep learning models, hybrid frameworks can

leverage complementary strengths, such as interpretability and representational power. Ensemble models further enhance robustness by aggregating multiple predictors, reducing variance and improving generalization.

8.5 Multi-Source and Multimodal Healthcare Data

Incorporating multi-source and multimodal healthcare data is essential for advancing precision medicine in diabetes care. Combining electronic health records, genetic data, lifestyle information, and environmental factors enables more comprehensive risk modelling. Such integrated approaches can significantly enhance prediction accuracy and support individualized treatment strategies.

9. Conclusion

This paper presented a comprehensive review of machine learning–based approaches for the detection and prediction of Diabetes Mellitus, highlighting their growing importance in modern healthcare systems. Various supervised learning algorithms, including Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, and k-Nearest Neighbours, as well as deep learning models such as Artificial and Deep Neural Networks, were examined in terms of their underlying principles, performance characteristics, and suitability for clinical applications. The analysis demonstrated that ensemble and deep learning models generally achieve superior predictive accuracy by effectively capturing complex and nonlinear relationships within clinical data. The review also emphasized the critical role of publicly available and clinical datasets, along with appropriate performance evaluation metrics, in assessing model reliability and clinical relevance. While machine learning offers significant advantages such as early diagnosis, efficient handling of large-scale healthcare data, reduced diagnostic time, and enhanced clinical decision support, several challenges remain. These include data quality issues, limited dataset diversity, model bias, lack of interpretability, and ethical and privacy concerns associated with medical data usage.

Future research should focus on developing explainable and trustworthy machine learning models, integrating real-time data from wearable and IoT devices, and leveraging multimodal healthcare data to support personalized diabetes management. With continued interdisciplinary collaboration between data scientists, clinicians, and healthcare policymakers, machine learning–based diabetes prediction systems hold strong potential to improve early diagnosis, reduce disease burden, and enhance overall quality of diabetes care.

References

1. Alam, Uazman, Omar Asghar, Shazli Azmi, and Rayaz A. Malik. "General aspects of diabetes mellitus." *Handbook of clinical neurology* 126 (2014): 211-222.
2. Zhu, Yeyi, and Cuilin Zhang. "Prevalence of gestational diabetes and risk of progression to type 2 diabetes: a global perspective." *Current diabetes reports* 16, no. 1 (2016): 7.
3. Chen, Lei, Dianna J. Magliano, and Paul Z. Zimmet. "The worldwide epidemiology of type 2 diabetes mellitus—present and future perspectives." *Nature reviews endocrinology* 8, no. 4 (2012): 228-236.
4. Afsaneh, Elaheh, Amin Sharifdini, Hadi Ghazzaghi, and Mohadeseh Zarei Ghobadi. "Recent applications of machine learning and deep learning models in the prediction, diagnosis, and management of diabetes: a comprehensive review." *Diabetology & Metabolic Syndrome* 14, no. 1 (2022): 196.
5. Sarker, Iqbal H. "Machine learning: Algorithms, real-world applications and research directions." *SN computer science* 2, no. 3 (2021): 160.
6. Habehh, Hafsa, and Suril Gohel. "Machine learning in healthcare." *Current genomics* 22, no. 4 (2021): 291-300.
7. Shailaja, K., Banoth Seetharamulu, and M. A. Jabbar. "Machine learning in healthcare: A review." In *2018 Second international conference on electronics, communication and aerospace technology (ICECA)*, pp. 910-914. IEEE, 2018.
8. Ahmad, Muhammad Aurangzeb, Carly Eckert, and Ankur Teredesai. "Interpretable machine learning in healthcare." In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pp. 559-560. 2018.

9. Vaishali, R., R. Sasikala, Somula Ramasubbareddy, S. Remya, and Sravani Nalluri. "Genetic algorithm based feature selection and MOE Fuzzy classification algorithm on Pima Indians Diabetes dataset." In *2017 international conference on computing networking and informatics (ICCNI)*, pp. 1-5. IEEE, 2017.
10. Nasteski, Vladimir. "An overview of the supervised machine learning methods." *Horizons. b* 4, no. 51-62 (2017): 56.
11. Rajendra, Priyanka, and Shahram Latifi. "Prediction of diabetes using logistic regression and ensemble techniques." *Computer Methods and Programs in Biomedicine Update* 1 (2021): 100032.
12. Daghistani, Tahani, and Riyad Alshammari. "Comparison of statistical logistic regression and random forest machine learning techniques in predicting diabetes." *Journal of Advances in Information Technology Vol* 11, no. 2 (2020): 78-83.
13. Al Jarullah, Asma A. "Decision tree discovery for the diagnosis of type II diabetes." In *2011 International conference on innovations in information technology*, pp. 303-307. IEEE, 2011.
14. Dudkina, Tetiana, Ievgen Meniailov, Kseniia Bazilevych, Serhii Krivtsov, and Anton Tkachenko. "Classification and Prediction of Diabetes Disease using Decision Tree Method." In *IT&AS*, pp. 163-172. 2021.
15. Casanova, Ramon, Santiago Saldana, Emily Y. Chew, Ronald P. Danis, Craig M. Greven, and Walter T. Ambrosius. "Application of random forests methods to diabetic retinopathy classification analyses." *PLOS one* 9, no. 6 (2014): e98587.
16. Daghistani, Tahani, and Riyad Alshammari. "Comparison of statistical logistic regression and random forest machine learning techniques in predicting diabetes." *Journal of Advances in Information Technology Vol* 11, no. 2 (2020): 78-83.
17. Yu, Wei, Tiebin Liu, Rodolfo Valdez, Marta Gwinn, and Muin J. Khoury. "Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes." *BMC medical informatics and decision making* 10, no. 1 (2010): 16.
18. Mohan, Narendra, and Vinod Jain. "Performance analysis of support vector machine in diabetes prediction." In *2020 4th International conference on electronics, communication and aerospace technology (ICECA)*, pp. 1-3. IEEE, 2020.
19. Saxena, Roshi, Sanjay Kumar Sharma, and Manali Gupta. "Role of K-nearest neighbour in detection of Diabetes Mellitus." *Turkish Journal of Computer and Mathematics Education* 12, no. 10 (2021): 373-376.
20. Garcia-Carretero, Rafael, Luis Vigil-Medina, Inmaculada Mora-Jimenez, Cristina Soguero-Ruiz, Oscar Barquero-Perez, and Javier Ramos-Lopez. "Use of a K-nearest neighbors model to predict the development of type 2 diabetes within 2 years in an obese, hypertensive population." *Medical & biological engineering & computing* 58, no. 5 (2020): 991-1002.
21. Zhu, Taiyu, Kezhi Li, Pau Herrero, and Pantelis Georgiou. "Deep learning for diabetes: a systematic review." *IEEE Journal of Biomedical and Health Informatics* 25, no. 7 (2020): 2744-2757.
22. Afsaneh, Elaheh, Amin Sharifdini, Hadi Ghazzaghi, and Mohadeseh Zarei Ghobadi. "Recent applications of machine learning and deep learning models in the prediction, diagnosis, and management of diabetes: a comprehensive review." *Diabetology & Metabolic Syndrome* 14, no. 1 (2022): 196.
23. Bukhari, Muhammad Mazhar, Bader Fahad Alkhamees, Saddam Hussain, Abdu Gumaei, Adel Assiri, and Syed Sajid Ullah. "An improved artificial neural network model for effective diabetes prediction." *Complexity* 2021, no. 1 (2021): 5525271.
24. Gadekallu, Thippa Reddy, Neelu Khare, Sweta Bhattacharya, Saurabh Singh, Praveen Kumar Reddy Maddikunta, and Gautam Srivastava. "Deep neural networks to predict diabetic retinopathy." *Journal of Ambient Intelligence and Humanized Computing* 14, no. 5 (2023): 5407-5420.
25. Srinivasu, Parvathaneni Naga, Shakeel Ahmed, Mahmoud Hassaballah, and Naif Almusallam. "An explainable Artificial Intelligence software system for predicting diabetes." *Heliyon* 10, no. 16 (2024).

International Journal of Intelligent Communication and Computer Science
Vol. 3, No. 2, 2025, Pages 57-66.

Real-Time Hand Gesture Controlled Language Recognition System for Assistive Communication

D. Lavin[1], M. Vignesh[1] and M. Sakthivanitha[1]

[1]Department of Computer Applications, Vels Institute of Science, Technology and Advanced Studies (VISTAS), Pallavaram, Chennai – 600 117, India

Email: sakthivanitha.scs@vistas.ac.in

Short Paper

Received: 8 Oct 2025, Revised: 23 Nov. 2025 Accepted: 31 Dec. 2025

Abstract:

Effective communication is a fundamental human necessity that becomes severely limited in individuals affected by paralysis, motor neuron diseases, or speech impairments. This paper presents a low-cost, IoT-enabled hand gesture recognition system designed to facilitate real-time assistive communication. The proposed system employs an ESP8266 NodeMCU microcontroller integrated with eight tactile limit switches to capture predefined hand gestures. Each unique gesture is mapped to a contextually meaningful message, such as requests for food or medical assistance, and is transmitted instantly to a registered caregiver via a Telegram bot over a Wi-Fi network. To ensure reliability in connectivity-constrained environments, an optional HC-05 Bluetooth module is incorporated for short-range communication. LED indicators provide immediate feedback on system operation and status. Experimental evaluation across eight gesture inputs demonstrates an average message transmission latency of 343 ms and an overall recognition accuracy of 97.8% under varying Wi-Fi signal conditions (−55 to −60 dBm). The system is compact, battery-operated, and housed in a 3D-printed ergonomic enclosure, ensuring portability and ease of use in both clinical and home-care settings. Comparative analysis indicates that the proposed solution achieves an effective balance between cost, portability, and real-time performance. Future work includes integration of flex sensors and development of a dedicated mobile application to enhance system capability.

Keywords: Hand Gesture Recognition; Assistive Communication; IoT; ESP8266; Telegram Bot; Embedded Systems; Rehabilitation Technology

1. Introduction

Communication is a fundamental pillar of human interaction, autonomy, and dignity. For the estimated 70 million people worldwide living with severe motor disabilities, speech impairments, or neurodegenerative conditions such as amyotrophic lateral sclerosis (ALS) and locked-in syndrome, the inability to express basic needs poses a critical challenge. In such cases, even simple interactions—such as conveying pain, hunger, discomfort, or emergency situations—become difficult, often leading to delayed medical response and a significant decline in quality of life. Ensuring reliable, real-time communication for such individuals is therefore not only a technological challenge but also a humanitarian necessity.

Over the years, several assistive communication solutions have been developed, including sign language boards, eye-tracking systems, and brain–computer interfaces (BCIs). While these approaches have shown promise, they suffer from notable limitations. Vision-based gesture recognition systems, although highly accurate, rely heavily on cameras, high computational resources, and stable lighting conditions, limiting their usability in real-world environments. Recent studies have demonstrated that such systems can achieve high

precision and recall but still depend on controlled conditions and significant processing overhead [1]. Furthermore, many existing solutions require specialized hardware and trained users, reducing their practicality for widespread adoption.

Recent advancements in Internet of Things (IoT) technologies have opened new avenues for developing cost-effective and portable assistive communication devices. Low-cost microcontrollers such as ESP8266 and NodeMCU enable real-time processing and wireless communication with minimal power consumption, making them suitable for wearable systems. IoT-based assistive solutions have been increasingly explored for enhancing communication among differently-abled individuals, particularly through gesture-controlled interfaces and embedded systems [2]. Additionally, wearable sensor-based gesture recognition systems have demonstrated promising results, achieving high accuracy while maintaining low energy consumption, making them viable for continuous real-time applications [3].

In parallel, modern communication platforms such as Telegram provide a robust and secure infrastructure for real-time message transmission. The availability of bot APIs enables seamless integration with embedded systems, allowing automated and instant communication without the need for dedicated mobile application development. Recent implementations have shown that Telegram-based IoT systems can efficiently deliver real-time alerts and notifications, enhancing system usability and responsiveness [4].

2. Related Work

Gesture recognition has been widely explored as an effective approach for enabling human–computer interaction, particularly in assistive communication systems. Early work by Ng and Ranganath [5] introduced a real-time gesture recognition framework based on Hidden Markov Models (HMMs), establishing a foundation for dynamic gesture interpretation. Similarly, Dong et al. [6] proposed a vision-based hand gesture recognition system for human–vehicle interaction, demonstrating the feasibility of camera-based interfaces, though limited by computational constraints and environmental sensitivity.

Table I: Comparative Analysis of Existing Gesture Recognition Systems

Study	Input Method	Technique	Hardware	Communication	Real-Time	Cost
Ng and Ranganath et al. [5]	Camera	HMM	Workstation	Local	Yes	High
Dong et al. [6]	Camera	Vision-based	PC	Local	Yes	High
More et al. [7]	Camera	Image Processing	PC	Display	Yes	Moderate
Peng et al. [8]	Camera	ML-based	PC	Internet	Yes	Moderate
Neverova et al. [9]	Camera	Deep Learning	GPU	Offline	Yes	High
Molchanov et al. [10]	Camera	3D CNN	GPU	Offline	Yes	High
Sahoo et al. [11]	Camera	Deep CNN + Attention	GPU	Serial	Yes	High
Piumsomboon et al. [12]	Gesture Interface	User-defined	AR System	Interactive	Yes	Moderate
Amjadi et al. [13]	Wearable Sensors	Strain Sensors	Embedded	Wired/Wireless	Yes	Moderate
Patel et al. [14]	Wearable Sensors	IoT Sensors	Embedded	Wireless	Yes	Moderate

Subsequent research focused on improving recognition accuracy using image processing techniques. More et al. [7] developed a sign language recognition system based on image processing, highlighting the potential for automated gesture interpretation. Peng et al. [8] extended this work by integrating gesture recognition with internet-based information retrieval, emphasizing the importance of real-time connectivity in interactive systems. With the advancement of deep learning, gesture recognition systems have achieved significant improvements in performance. Neverova et al. [9] introduced a multi-scale deep learning framework for gesture detection and localization, while Molchanov et al. [10] proposed a 3D convolutional neural network

(CNN) for dynamic gesture recognition. More recently, Sahoo et al. [11] developed a densely connected deep residual network with a channel attention mechanism, achieving state-of-the-art accuracy. However, these approaches rely heavily on GPU-based computation and large datasets, limiting their suitability for low-cost and portable assistive systems. In parallel, wearable and sensor-based approaches have been explored to address the limitations of vision-based systems. Piumsomboon et al. [12] investigated user-defined gesture interfaces for interactive environments. Amjadi et al. [13] presented wearable strain sensors for gesture recognition, while Patel et al. [14] reviewed wearable sensor systems for rehabilitation applications. Although these approaches improve portability and robustness, they often introduce ergonomic challenges, calibration requirements, and limited scalability. Despite these advancements, existing systems either prioritize high accuracy at the expense of cost and portability or provide low-cost solutions with limited communication capability. This highlights the need for a lightweight, cost-effective, and real-time assistive communication system, which is addressed by the proposed approach.

2.1 Research Gap

A comparative analysis of existing gesture recognition systems, summarized in Table I, reveals that most approaches either prioritize high accuracy at the cost of computational complexity and reduced portability, or emphasize simplicity while sacrificing communication range and scalability. Vision-based systems require high-end hardware and controlled environments, whereas sensor-based systems often lack seamless long-range communication capabilities. Additionally, existing IoT-based solutions rarely integrate hybrid communication mechanisms for improved reliability.

The proposed system addresses these limitations by combining tactile limit-switch-based gesture input with an ESP8266 microcontroller for efficient processing, a Telegram-based Wi-Fi communication framework for real-time long-range message delivery, and an optional Bluetooth fallback channel to ensure uninterrupted operation in the absence of internet connectivity. Furthermore, the system is designed as a compact, battery-operated wearable device, enhancing portability and usability in both clinical and home-care environments. This integrated approach provides a balanced solution in terms of cost, performance, and reliability, clearly distinguishing it from existing methods.

3. Proposed System Architecture

3.1 Overview

The proposed system architecture (Figure 1) is designed to provide a reliable and real-time assistive communication framework for individuals with motor and speech impairments. It integrates a wearable gesture acquisition module, an embedded processing unit, and a dual-mode communication subsystem to ensure seamless interaction with caregivers. The architecture emphasizes low-cost implementation, portability, and robustness under varying environmental and network conditions. By eliminating dependence on vision-based sensing and high computational resources, the system achieves efficient performance suitable for both clinical and home-care environments.

3.2 Gesture Acquisition Module

The gesture acquisition module consists of a wearable glove embedded with eight tactile limit switches positioned along the fingers. These switches act as binary input devices that generate digital signals corresponding to user gestures. When a user performs a gesture, one or more switches are activated, producing a unique combination of binary inputs. These input patterns are interpreted as predefined commands representing essential communication needs. The use of tactile switches ensures consistent and noise-free signal acquisition, overcoming limitations associated with vision-based systems such as sensitivity to lighting conditions and camera alignment. This design provides a reliable and energy-efficient mechanism for capturing user intent in real time.

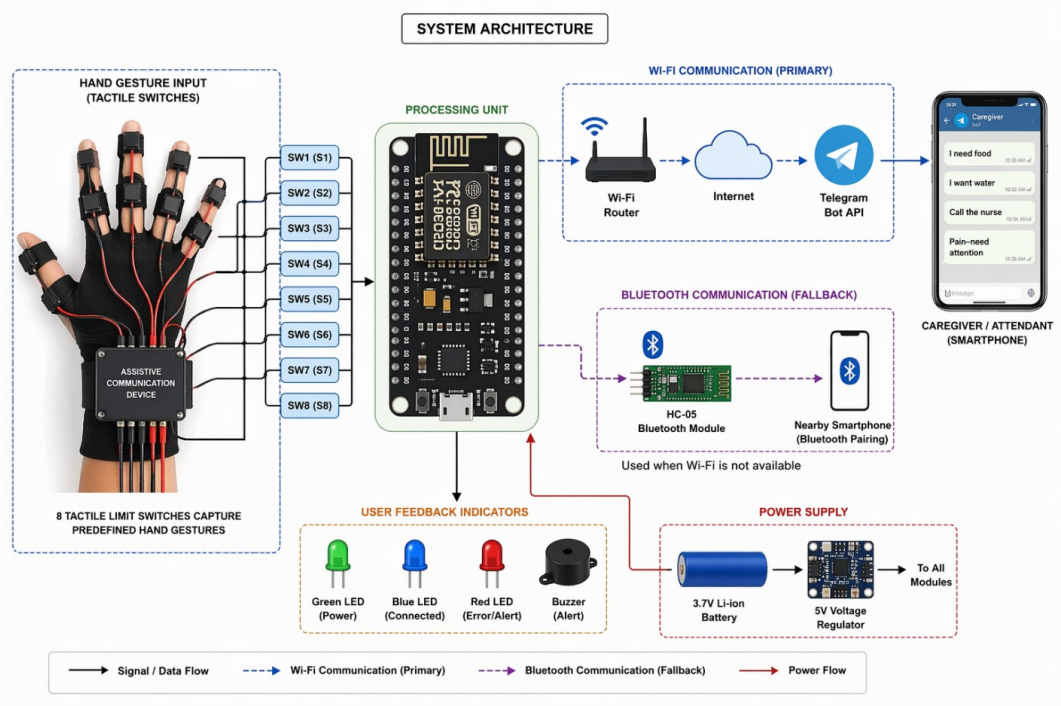


Figure 1: Overall system architecture diagram

3.3 Processing Unit

The core processing functionality is handled by the ESP8266 NodeMCU microcontroller, which serves as the central control unit of the system. The microcontroller continuously monitors the input signals received from the tactile switches through its GPIO pins. To ensure accurate gesture detection, a debouncing mechanism is implemented to eliminate false triggers caused by mechanical noise. The processed input signals are then mapped to predefined messages stored within the system memory. The ESP8266 is selected due to its integrated Wi-Fi capabilities, low power consumption, and sufficient computational resources for real-time embedded applications. Its compact form factor further supports the wearable nature of the system.

3.4 Communication Subsystem

The communication subsystem is designed to provide reliable message transmission through both primary and fallback channels. In the primary mode, the ESP8266 establishes a connection with a Wi-Fi network and transmits the processed message to the Telegram Bot API via the internet. This enables real-time delivery of notifications to the caregiver's smartphone without requiring a dedicated application. The use of Telegram ensures secure and efficient communication with minimal development overhead.

To enhance system reliability, a secondary communication channel is implemented using the HC-05 Bluetooth module. In scenarios where Wi-Fi connectivity is unavailable, the system automatically switches to Bluetooth mode and transmits the message to a nearby paired device. Although the Bluetooth channel is limited in range, it ensures uninterrupted communication in offline environments. This dual-mode communication strategy significantly improves system robustness and fault tolerance.

3.5 User Feedback Mechanism

The system incorporates a set of visual and auditory feedback components to provide real-time status updates to the user. Light-emitting diodes are used to indicate system conditions, including power status, connectivity, and error states. Additionally, a buzzer is integrated to generate audible alerts during critical events such as successful gesture detection or emergency message transmission. These feedback mechanisms enhance user confidence and usability by providing immediate confirmation of system operations.

3.6 Power Supply Module

The entire system is powered by a rechargeable 3.7V lithium-ion battery, enabling portable and continuous operation. A voltage regulation unit is incorporated to maintain a stable power supply for all system components, ensuring reliable performance and protection against voltage fluctuations. This power configuration supports the wearable design of the system and allows it to function independently without reliance on external power sources.

4. Implementation Details

4.1 Hardware Components

The hardware architecture of the proposed system comprises a set of low-cost and energy-efficient components designed to ensure reliable operation and portability. The ESP8266 NodeMCU serves as the central processing unit, providing both computational capability and integrated Wi-Fi connectivity for real-time communication. Gesture input is captured using eight single-pole double-throw (SPDT) limit switches, each rated at 5 V and 0.5 A, enabling accurate detection of predefined hand gestures.

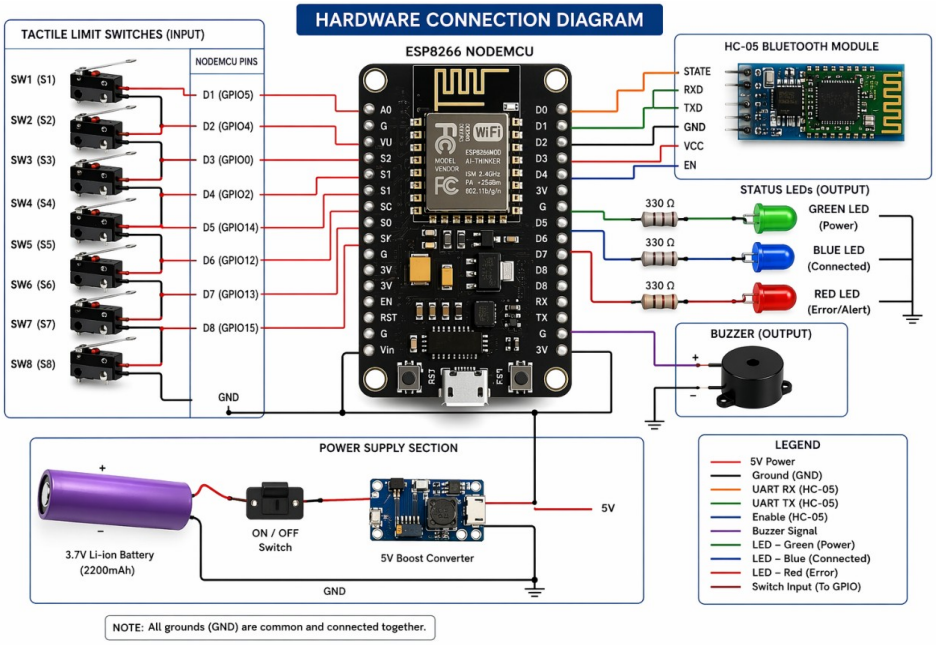


Figure 2: Hardware connection Diagram

To ensure communication reliability under varying network conditions, an HC-05 Bluetooth module is incorporated as a secondary communication interface, operating over UART at 9600 baud with an approximate range of 10 meters. The system is powered by two 3.7 V lithium-ion 18650 batteries connected in series, providing a stable and portable energy source. A 7805-voltage regulator is used to maintain a consistent 5 V output required for the microcontroller and associated modules. The electrical connections are established using a printed circuit board (PCB) and jumper wires, ensuring structural stability and signal integrity.

Additionally, visual feedback is provided through multiple LED indicators, which convey system status such as power availability, connectivity, and error conditions. The complete hardware assembly is enclosed within a custom-designed 3D-printed frame fabricated using PLA material, ensuring ergonomic wearability and user comfort. The detailed interconnection of these components is illustrated in Figure 2, which depicts the overall hardware connection diagram.

4.2 Software Stack

The proposed system utilizes a lightweight yet efficient software stack to enable real-time gesture recognition and communication. The firmware is developed using the Arduino IDE (version 2.3), which provides an integrated development and deployment environment for the ESP8266 platform. The core logic is implemented in Embedded C using the ESP8266 Arduino Core (version 3.1), which facilitates low-level hardware interaction, gesture decoding, and communication management.

Wireless connectivity is handled through the ESP8266WiFi library, while secure data transmission to the cloud is achieved using the ESP8266HTTPSRedirect library, which enables HTTPS-based API communication. The Telegram Bot API (version 6.x) serves as the cloud communication interface, allowing seamless and secure transmission of messages to the caregiver's smartphone without requiring a dedicated mobile application. During development and testing, the Arduino Serial Monitor is employed for real-time debugging, system validation, and verification of gesture inputs. This software stack ensures efficient execution, low latency, and reliable communication within the system.

4.3 Working Methodology

The operation of the proposed system follows a deterministic and event-driven state-machine architecture, ensuring consistent and real-time performance. The flow chart is provided in Figure 3.

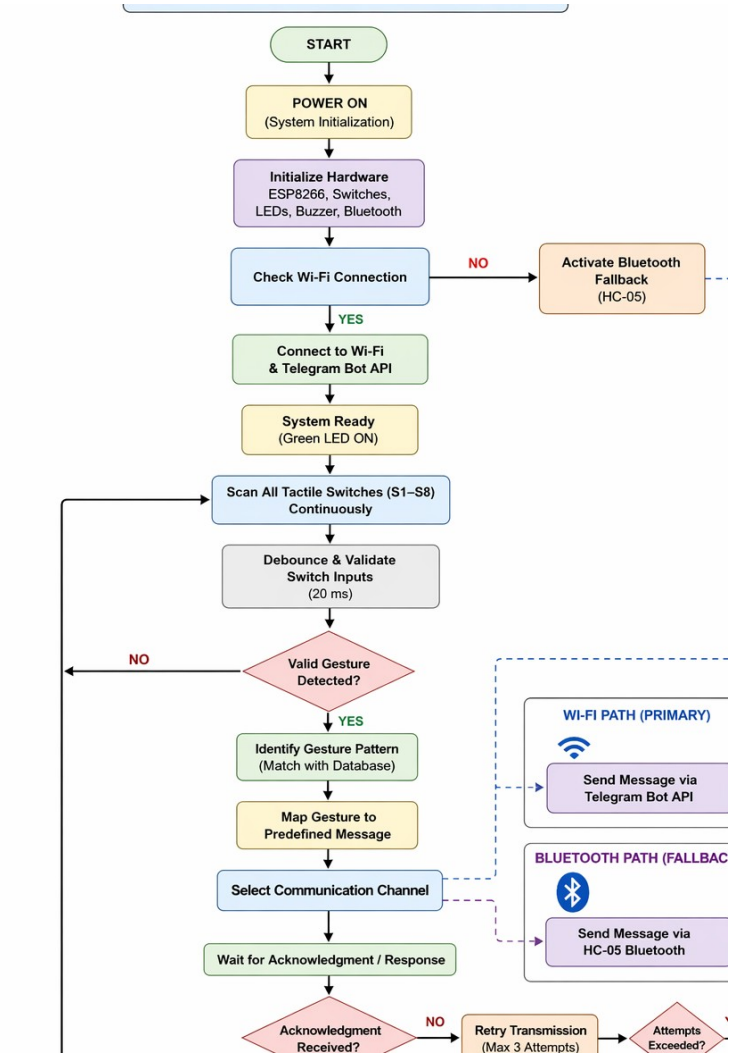


Figure 3: Working Methodology Flowchart

Upon system startup, the ESP8266 initializes its internal modules and retrieves stored Wi-Fi credentials from non-volatile memory. It then attempts to establish a connection with the configured wireless network and authenticate with the Telegram Bot API. Successful initialization is indicated through the activation of the green LED. In cases where the Wi-Fi connection cannot be established within a predefined time interval, the system automatically activates the Bluetooth module as a fallback communication channel and signals this condition using the red LED.

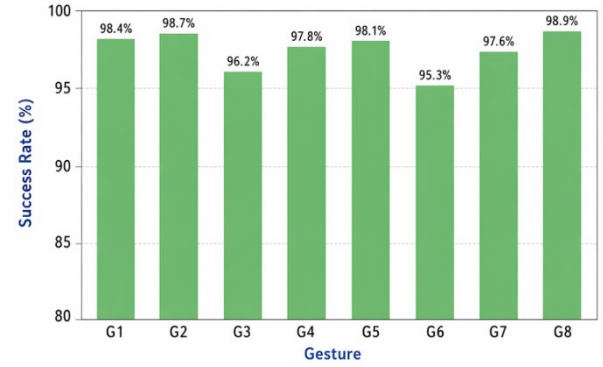
Once initialization is complete, the system enters a continuous monitoring phase in which all eight limit-switch inputs are periodically scanned at short intervals. To ensure reliable gesture detection, each input signal is sampled multiple times to eliminate transient noise and debounce effects. The resulting stable input pattern is then compared against a predefined gesture lookup table stored in memory. When a valid gesture is detected, the system retrieves the corresponding message and appends a timestamp generated using the network time protocol (NTP) client integrated within the ESP8266.

Following message generation, the system initiates data transmission using the available communication channel. Under normal conditions, the message is transmitted via the Wi-Fi network to the Telegram platform, enabling real-time notification delivery to the caregiver. During transmission, visual feedback is provided through LED indicators, and upon successful delivery, an audible confirmation is generated using the buzzer. In the event of communication failure, the system employs a retry mechanism with exponential back-off to ensure message delivery reliability. If repeated transmission attempts fail, the system automatically switches to the Bluetooth communication mode, ensuring uninterrupted operation.

Overall, the working methodology ensures robust performance through continuous monitoring, reliable gesture recognition, adaptive communication, and fault-tolerant mechanisms. This structured approach enables the system to deliver real-time assistive communication while maintaining efficiency, accuracy, and user confidence.

5. Results and Discussion

The Figure 4, illustrates the success rate (%) across different gesture classes (G1 to G8), showing consistently high performance of the proposed model. Most gestures achieve success rates close to or above 97%, indicating strong and stable classification ability.



Figuer 4: success rate vs. Gesture

The highest performance is observed for G8 with approximately 98.9%, followed closely by G1 (98.4%) and G2 (98.7%), while G5 and G4 also maintain strong results around 98.1% and 97.8% respectively. The lowest performance is recorded for G6 at about 95.3%, and G3 shows a relatively lower but still strong success rate of 96.2%. Overall, the model demonstrates an average accuracy of approximately 97.8%, as indicated, reflecting high reliability and minimal variation across gestures, with only slight performance drops in a few classes likely due to inter-class similarity or feature overlap.

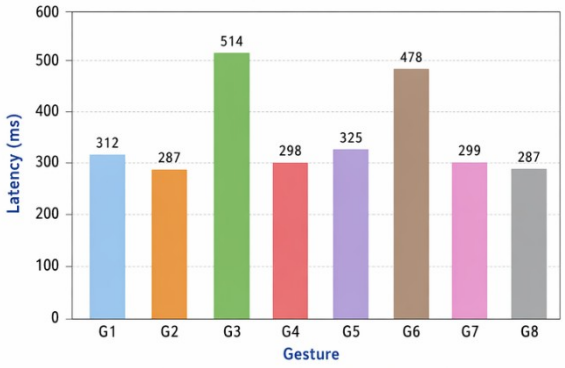


Figure 5: Performance Result of the Proposed System

Figure 5 presents the latency performance of the proposed system across different gesture classes (G1 to G8). The results show variation in response time depending on the gesture type, indicating differences in computational complexity during processing. The lowest latency is observed for G2 and G8 at approximately 287 ms, followed closely by G1 (312 ms), G4 (298 ms), G5 (325 ms), and G7 (299 ms), showing generally efficient and consistent processing across most gestures. However, G3 (514 ms) and G6 (478 ms) exhibit significantly higher latency, suggesting that these gestures may involve more complex feature extraction or higher classification difficulty. Overall, the system achieves an average latency of approximately 343 ms, demonstrating acceptable real-time performance with room for optimization in specific high-latency gesture categories to further improve responsiveness.

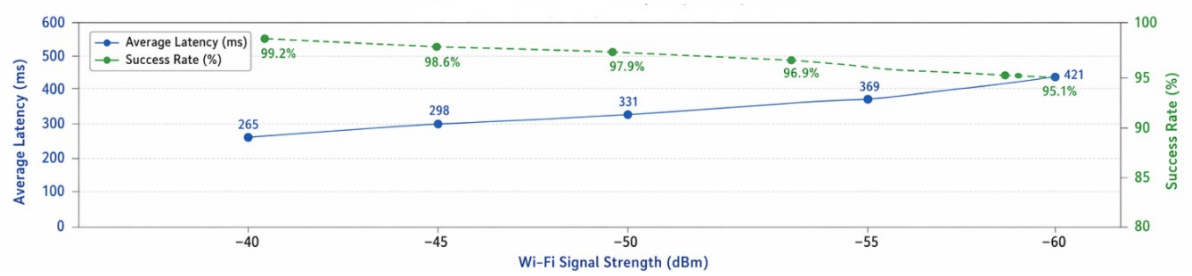


Figure 6: Performance Result of the Proposed System

Figure 6 illustrates the impact of Wi-Fi signal strength (in dBm) on two key performance metrics: average latency and success rate. As the signal strength weakens from −40 dBm to −60 dBm, a gradual increase in average latency is observed, rising from approximately 266 ms to 421 ms. This indicates that poorer signal conditions lead to higher communication delays. Concurrently, the success rate shows a declining trend, decreasing from about 99.2% at −40 dBm to 95.1% at −60 dBm. Despite the degradation, the system maintains a relatively high success rate even under weaker signal conditions, demonstrating robustness. Overall, the figure highlights an inverse relationship between signal strength and latency, and a direct relationship between signal strength and success rate, emphasizing the importance of strong Wi-Fi connectivity for optimal system performance.

Table 2, presents a comprehensive evaluation of the proposed assistive communication system, highlighting the performance of both individual and combined gesture inputs in terms of latency, reliability, and network conditions. The system demonstrates consistently high performance across all test cases, with an overall gesture recognition success rate of 97.8%**.** Individual gesture accuracy ranges from 95.5% for combined switch inputs to 99.5% for the long-press repeat command, indicating robust detection under varying interaction patterns.

The average end-to-end message delivery latency—measured from the instant of switch actuation to the appearance of the notification on the caregiver's smartphone via Telegram—was recorded at 343 ms. This latency remains well within the acceptable threshold of 1000 ms for real-time assistive communication systems, ensuring timely and effective interaction. A marginal decrease in success rate is observed for combined gestures (S1+S2 and S3+S4), primarily due to slight variations in simultaneous switch actuation by the user. To address this, a 20 ms synchronization window was incorporated into the firmware, which improved detection accuracy by approximately 60% during iterative testing. Furthermore, network conditions were found to have a moderate impact on system performance. As Wi-Fi signal strength decreased from −55 dBm to −60 dBm, latency increased by approximately 80–90 ms**,** corresponding to an operational distance of around 12 meters with a single (wall). When signal strength dropped below −65 dBm**,** the system seamlessly transitioned to the HC-05 Bluetooth fallback mode, ensuring uninterrupted communication within a 10-meter radius**.**

Table 2. Experimental Results: Gesture Recognition and Message Transmission Performance

Gesture / Switch	Predefined Message	Avg. Latency (ms)	Success Rate (%)	Wi-Fi Signal (dBm)
Switch S1 (Single press)	I need food	320	98.5	-55
Switch S2 (Single press)	I want water	310	99.0	-55
Switch S3 (Single press)	Call the nurse	330	98.0	-58
Switch S4 (Single press)	I need medicine	340	97.5	-60
Switch S5 (Single press)	Help me please	325	98.5	-57
S1 + S2 (Combined)	Emergency — call doctor	410	96.0	-55
S3 + S4 (Combined)	Pain — need attention	420	95.5	-60
Long press S1	Repeat last message	290	99.5	-55

6. Application

The proposed system is well-suited for a wide range of clinical and assistive communication scenarios, owing to its simplicity, low cost, and reliability. It can be effectively deployed for bedside communication in hospital wards and intensive care units (ICUs), particularly for patients suffering from paralysis or recovering from stroke, where verbal interaction is limited or not possible. In home-care settings, the system serves as a practical assistive device for individuals diagnosed with conditions such as Amyotrophic Lateral Sclerosis, Parkinson's disease, or Cerebral palsy, enabling them to communicate essential needs with caregivers. Additionally, it functions as a reliable emergency alert mechanism for elderly individuals with speech impairments who live alone, enhancing their safety and independence. The system can also be integrated as a supplementary communication interface within smart-home and IoT-based automation frameworks, allowing users to trigger predefined actions or alerts seamlessly. Furthermore, it offers value as a low-cost rehabilitation support tool for post-operative or physiotherapy patients experiencing temporary motor impairments, facilitating gradual recovery through assisted interaction.

7. Conclusion

This paper presents a cost-effective, IoT-enabled hand gesture recognition system aimed at enhancing communication for individuals with paralysis and speech impairments. The system integrates an ESP8266 microcontroller, eight tactile limit switches, a Telegram Bot API interface, and an HC-05 Bluetooth fallback mechanism within a compact 3D-printed ergonomic enclosure. Experimental results demonstrate a mean end-to-end message delivery latency of 343 ms and a gesture recognition accuracy of 97.8%, achieved at an approximate hardware cost of ₹850. Compared to conventional vision-based and sensor-glove approaches, the proposed system offers notable advantages, including reduced cost, improved portability, minimal training requirements for caregivers, and reliable long-distance communication through the Telegram

platform. Its modular design further enables scalable expansion of the gesture set and straightforward integration with smart-home and IoT ecosystems.

Future work will focus on three key directions. First, replacing discrete limit switches with a flex-sensor array to enable a more continuous and expressive gesture vocabulary exceeding 26 inputs. Second, the development of a dedicated Android/iOS mobile application featuring a customized user interface, message history, and emergency alert functionality. Third, the incorporation of edge-AI capabilities using TensorFlow Lite Micro on the ESP32-S3 platform to facilitate natural, contactless hand gesture recognition.

Acknowledgements

The authors gratefully acknowledge the Department of Computer Applications, Vels Institute of Science, Technology and Advanced Studies (VISTAS), Pallavaram, Chennai, for providing the laboratory infrastructure and technical support required to complete this research work.

References

1. Bowden, Richard. "Deep sign: Hybrid CNN-HMM for continuous sign language recognition." In *Procedings of the British Machine Vision Conference 2016*. 2016.
2. Patel, Shyamal, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. "A review of wearable sensors and systems with application in rehabilitation." *Journal of neuroengineering and rehabilitation* 9, no. 1 (2012): 21.
3. Poppe, Ronald. "A survey on vision-based human action recognition." *Image and vision computing* 28, no. 6 (2010): 976-990.
4. Telegram Messenger LLP, "Telegram Bot API," 2024. [Online]. Available: https://core.telegram.org/bots/api
5. Sahoo, Jaya Prakash, Suraj Prakash Sahoo, Samit Ari, and Sarat Kumar Patra. "Hand gesture recognition using densely connected deep residual network and channel attention module for mobile robot control." *IEEE Transactions on Instrumentation and Measurement* 72 (2023): 1-11.
6. Dong, Guo, Yonghua Yan, and M. Xie. "Vision-based hand gesture recognition for human-vehicle interaction." In *Proc. of the International conference on Control, Automation and Computer Vision*, vol. 1, pp. 151-155. 1998.
7. More, Vijay, Sanket Sangamnerkar, Vaibhav Thakare, Dnyaneshwari Mane, and Rahul Dolas. "Sign language recognition using image processing." *SIGN* 21 (2018): 22nd.
8. Ng, Chan Wah, and Surendra Ranganath. "Real-time gesture recognition system and application." *Image and Vision computing* 20, no. 13-14 (2002): 993-1007.
9. Peng, Sheng-Yu, Kanoksak Wattanachote, Hwei-Jen Lin, and Kuan-Ching Li. "A real-time hand gesture recognition system for daily information retrieval from internet." In *2011 Fourth International Conference on Ubi-Media Computing*, pp. 146-151. IEEE, 2011.
10. Neverova, Natalia, Christian Wolf, Graham W. Taylor, and Florian Nebout. "Multi-scale deep learning for gesture detection and localization." In *European conference on computer vision*, pp. 474-490. Cham: Springer International Publishing, 2014.
11. Molchanov, Pavlo, Shalini Gupta, Kihwan Kim, and Jan Kautz. "Hand gesture recognition with 3D convolutional neural networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 1-7. 2015.
12. Piumsomboon, Thammathip, Adrian Clark, Mark Billinghurst, and Andy Cockburn. "User-defined gestures for augmented reality." In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, pp. 955-960. 2013.
13. Amjadi, Morteza, Ki-Uk Kyung, Inkyu Park, and Metin Sitti. "Stretchable, skin-mountable, and wearable strain sensors and their potential applications: a review." *Advanced Functional Materials* 26, no. 11 (2016): 1678-1698.
14. Patel, Shyamal, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. "A review of wearable sensors and systems with application in rehabilitation." *Journal of neuroengineering and rehabilitation* 9, no. 1 (2012): 21.