International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

A Machine Learning-Based Multi-Trigger Emergency Detection System for Smartphones

Antony Mugesh. F
Department of Biomedical
Rohini College Of Engineering and Technology
Kanyakumari, India

Jeffrin Lazaras. A
Department of Biomedical
Rohini College Of Engineering And Technology
Kanyakumari, India

Dr. Anlin Sahaya Infant Tinu. M
Assistant Professor
Department of Biomedical
Rohini College Of Engineering and Technology
Kanyakumari, India

Abstract **- Personal safety in urban and rural areas is still a major problem especially in cases where manual assistance may not be possible, such as accidents, medical emergencies and threatening situations. We propose a smartphone-based intelligent emergency detection and alert system that uses machine learning and mobile sensing technologies for rapid and automated emergency response. The proposed system, unlike conventional safety applications that only rely on manual activation, can detect emergency using multiple triggers such as voice recognition, fall detection, shake detection and manual SOS. A multimodal decision fusion module integrates the outputs of such subsystems to enhance detection reliability and reduce false activations.The system integrates voice recognition using Convolutional Neural Networks (CNN), shake detection using Random Forest, fall detection using Long ShortTerm Memory (LSTM), and a manual SOS feature. In case of an emergency, the application gives you a 10 second window to cancel the alert and then automatically sends out alert messages with your real-time GPS location, calls emergency numbers of your pre-defined contacts and activates on-site safety features like siren alerts and blinking flashlights to get local attention. The proposed system intends to offer a practical, accessible and hardware independent solution for emergency support through daily smartphone technology.**

Keywords - Emergency Detection System, Smartphone Safety Application, Machine Learning, CNN, LSTM, Random Forest, Fall Detection, Voice Recognition, Shake Detection, SOS Alert, TensorFlow Lite, Personal Safety.

I. INTRODUCTION

Personal safety has become a major concern in both urban and rural environments due to the increasing number of accidents, medical emergencies, assaults, and threatening situations.In many emergencies, victims are unable to call for help manually because of panic, physical injury, unconsciousness or lack of immediate access to communication. While there are several safety applications available today, the majority of existing systems rely primarily on manual activation and could fail to offer timely support in actual emergencies.

Recent advances in smartphones, mobile sensing technologies, and machine learning have made it possible to develop intelligent emergency detection systems that can automatically detect emergency situations and initiate emergency responses. Modern smartphones are equipped with multiple sensors like accelerometers, microphones, GPS modules, and communication systems that can be used effectively for personal safety applications without any additional hardware requirements.

This work presents a smart phone based intelligent emergency detection and alert system with voice recognition, fall detection, shake detection and manual SOS activation. The system utilizes machine learning models such as Convolutional Neural Networks (CNN), Random Forest and Long Short-Term Memory (LSTM) networks for analyzing audio signals and sensor data. At the time of emergency, the application sends alert messages with real-time GPS location automatically, also makes emergency calls, and triggers on-site safety features like siren alerts and flashing flashlight to draw the attention of those nearby.

A. Motivation

The increasing number of road accidents, health emergencies, and unsafe situations has created a need for faster emergency support systems.When things get critical, users may panic or be physically unable to contact emergency services manually. Prompt emergency response is critical for risk reduction and personal safety enhancement.

Smartphones are everywhere, and they have sensors, GPS, microphones and communication technologies that can be used to enable intelligent safety applications. This gave rise to the idea of an automatic emergency detection system that can help remotely and locally in dangerous situations.

B. Problem Statement

Although there are many emergency safety apps such as Google Personal Safety, Life360 and Noonlight etc..,these existing systems mainly rely on manual activation of SOS and user interaction. Their effectiveness is reduced when users are unconscious, injured, or unable to access their smartphones during accidents, medical emergencies, or threatening situations.Currently, many systems do not have intelligent multi-trigger emergency detection and emergency support features are also limited. Automatic detection with voice recognition, fall detection and analysis of smartphone sensors are often not available which leads to delayed

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

response and reduced accuracy.

Therefore, there is a need for an intelligent smartphone-based emergency detection system that can automatically detect emergency, share real-time location and emergency communication, and activate local safety features for faster and reliable emergency assistance.

C. *Objectives*

1. To develop a smartphone-based intelligent emergency detection system.
2. To implement voice recognition, fall detection, and shake detection using machine learning.
3. To provide manual SOS activation during emergencies.
4. To enable automatic alert messages with real-time GPS location.
5. To initiate emergency calls to predefined contacts.
6. To activate siren alerts and blinking flashlight notifications for on-site safety.
7. To reduce false alarms using multi-trigger emergency detection.
8. To provide a fast, accessible, and hardware-independent personal safety solution.

II. LITERATURE REVIEW

Smartphone-based emergency detection has emerged as a critical area of research due to the growing need for automated safety systems that can respond to accidents, medical emergencies, and threatening situations without relying on manual intervention.Due to the rapid development of machine learning, mobile sensing technologies and on-device inference frameworks, intelligent safety systems are increasingly feasible on everyday smartphones.The purpose of this literature survey is to study the existing emergency detection systems, the machine learning techniques used and the limitations. A comprehensive review of the past work provides an understanding of the existing trends, challenges and gaps which are the basis for designing a more reliable and intelligent multi-trigger safety solution.

A. *Existing Systems*

A review of multi-sensor fusion techniques in body sensor networks, covering data, feature, and decision-level fusion and challenges in synchronization and energy efficiency, was conducted by Gravina et al. (2017). However, it did not include smartphone-based emergency alert systems [1]. An early ML framework for activity recognition from wrist-worn accelerometers was proposed by Ravi et al. (2005), but the performance degraded across different users [2]. Rashid et al. reviewed fall detection methods and said that the accelerometer-based methods are most practical, but high false alarm rates and poor adaptability are still the issues [3]. Saez et al. (2017) found that Random Forest achieved the highest accuracy for activity recognition on smartphones; however, emergency situations such as falls were not included [4].Gjoreski et al. (2016) showed that multi-modal smartwatch sensors increase the accuracy of fall detection but did not have any alert mechanisms after detection [5]. Torti et al. (2018) implemented quantized deep learning models on embedded wearables to detect falls in real-time but did not support multi-trigger detection or alerting [6]. Abdel-Hamid et al. (2014) showed that CNNs are superior to HMM-based speech recognition, but their computational requirements make them unsuitable for mobile emergency systems [7]. Ronao and Cho (2016) reported a high activity recognition accuracy from raw smartphone IMU data without manual feature engineering but did not include emergency detection [8].To benchmark fall detection, the SisFall dataset was proposed by Sucerquia et al. (2017) from 38 subjects, but the lab-collected data may not be representative of real-world scenarios [9]. MobiFall was developed by Vavoulas et al. (2016) for fall detection using smart phones in different body positions, but the simulated conditions limit its real-world relevance [10]. Lee et al. [11] showed how to do TensorFlow Lite inference in real-time on Android GPUs, but they did not consider safety-critical applications [11]. Hossain et al. proposed an alert system based on blockchain to ensure tamper-proof communication, but computational overhead reduces viability in low-connectivity settings [12]. Patel and Kumar (2020) developed an automatic GPS-based SOS app but depended heavily on internet and cloud infrastructure [13]. Hassan et al. (2021) evaluated a GPS-aware SOS system under different network conditions but only supported manual activation and not automated detection [14]. Google AudioSet (Gemmeke et al., 2017) provided 527 audio event classes as a benchmark for audio classification, but it does not include emergency-specific labels for safety applications [15].

B. *Limitations of Existing Systems*

Despite advancements in emergency detection and mobile safety systems, many existing solutions still face major limitations:

- Dependence on a single trigger such as fall detection or manual SOS.
- Limited multi-modal fusion, causing false alarms and missed detections.
- Heavy reliance on internet connectivity for alerts and cloud processing.
- Limited on-device processing, leading to latency and privacy issues.
- Lack of on-site safety features like sirens or flashlight alerts.
- Dependence on laboratory-collected datasets that do not represent real-world emergencies.

These limitations highlight the need for a reliable multi-trigger emergency detection system.

III. METHODOLOGY

The proposed system is organized into four functional layers:

(1) Sensor Acquisition,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

(2) Signal Processing and Feature Extraction,

(3) Parallel ML Inference,

(4) Alert Dispatch.

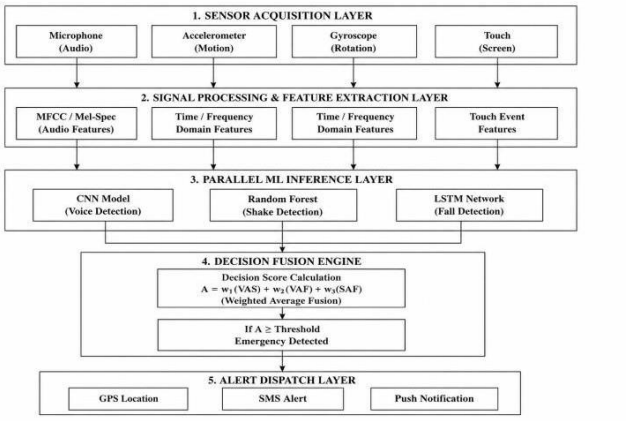

Fig.1. System Architecture of the Multi-Trigger Emergency Detection Framework.

A. Sensor Acquisition

The system constantly gathers data from the smartphone's sensors to detect emergencies. The microphone records sound by taking 16,000 audio samples per second and processing them in segments of 1 second. Android's SensorManager API measures movement and rotation with the accelerometer and gyroscope sensors 50 times a second. All sensor readings are safely stored with time information, so that the system can analyze them in the correct order. The GPS is usually in battery-saving mode while tracking location in standby. When an emergency is detected, the system automatically switches to high accuracy GPS mode, to get the user's exact location and send emergency alerts quickly and accurately.

B. Signal Processing and Feature Extraction

The recorded audio is converted into log-Mel spectrograms to enable efficient sound pattern analysis by the CNN model. The audio signal is segmented into small frames and transformed into 64 × 64 spectrogram patches for voice-based emergency detection.For motion analysis, accelerometer and gyroscope data are processed using 2-second sliding windows with 50% overlap to avoid information loss. From each window, important statistical features such as mean, standard deviation, RMS, skewness, kurtosis, and zero-crossing rate are extracted from all three motion axes.The extracted features are then provided to the Random Forest and LSTM models for emergency event classification. Finally, all features are normalized using zero mean and unit variance normalization to improve model stability and classification accuracy.

C. Decision Fusion Engine

Three independent ML inference threads execute in parallel via Kotlin Coroutines. Each thread outputs a binary label and a continuous confidence score in the range [0, 1]. The fusion engine applies a confidence threshold of 0.75 (selected by threshold sweep on the validation set; see Section VI-C) and computes the alert flag A as:

```
A = M  OR  [(V AND S)  OR  (V AND F)  OR
                (S AND F)]
```

where V, S, F are the thresholded binary outputs of the voice, shake, and fall modules respectively, and M is the manual SOS button state. The manual button provides an unconditional override. The automated path requires at least two modules to independently confirm an emergency before an alert is raised, enforcing multimodal temporal consistency.

D. Alert Dispatch

When A = 1, the system fetches the most recent GPS fix, formats a message containing event type, timestamp, and a Google Maps hyperlink, and dispatches it via SMS to up to three pre-registered emergency contacts using Android's SmsManager API. An in-app notification and audible alarm accompany the SMS. The event is logged to a local SQLite database for post-event review.

IV. MACHINE LEARNING MODEL AND TRAINING

A. Voice Trigger— Convolutional Neural Network

The voice trigger model was trained on 8,500 audio samples from the Google Speech Commands dataset, including the keyword "help" and distress voice samples in Tamil and English. The CNN comprises 3 convolutional layers with 3×3 filters and 32, 64, and 128 channels respectively. ReLU activation, batch normalization, and 2×2 max-pooling were applied after each layer. Global average pooling and a sigmoid output layer generate the final output. Data augmentation (time shifting ±100 ms, pitch shifting ±2 semitones, white Gaussian noise at SNR 5–20 dB) was applied for noise robustness. The CNN achieved a lower individual accuracy compared to the motion-based models (91.8% ± 1.2%), which is expected given the inherent variability in voice characteristics, background noise, and language diversity. The model was converted to TensorFlow Lite with dynamic range quantization, reducing its size to 0.8 MB.

B. Shake Detection—Random Forest

The shake detection dataset consists of 3,000 motion samples: 1,200 intentional shake gestures and 1,800 normal activity samples including walking, running, cycling, and sitting. Motion data was segmented in 2 second windows and 18 time domain statistical features were extracted from each window for analysis. The optimal model parameters were determined through hyperparameter optimization with five-fold cross-validation. The best parameters were 100 decision trees, a maximum depth of 12 and a minimum split of 4 samples. The trained model was converted using the sklearn-to-TFLite pipeline and quantized to only 1.2 MB. The final model achieved a mean validation accuracy of 93.4% with ±0.9% variation across all folds.

C. Fall Detection—LSTM network

The fall detection system was designed in the form of a binary classifier model to predict if a person has fallen or

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

not. The training data set consisted of 7200 motion sequences, which were equally taken from the SisFall and MobiFall benchmark data sets. To balance the fall and non-fall classes, random oversampling of the minority class was used.Each input sequence of 50 frames was recorded for 1s with 50Hz sampling rate . Each frame contained 6 sensor values, comprising 3-axis accelerometer and 3-axis gyroscope data.The model was built using an LSTM (Long Short-Term Memory) architecture, with two stacked layers of 64 units each. Dropout with a probability of 0.3 was employed after each layer to prevent overfitting.The model was trained with Adam optimizer with learning rate 0.001. We trained the model until the validation performance did not improve, and then used early stopping with patience of 10 epochs.The model was evaluated using five-fold cross validation, and produced a mean validation accuracy of 95.1%
± 0.8%. Finally, the model was quantized and converted to TensorFlow Lite format, reducing the model size to 0.6 MB for light-weight deployment on portable devices.

D. Data Flow

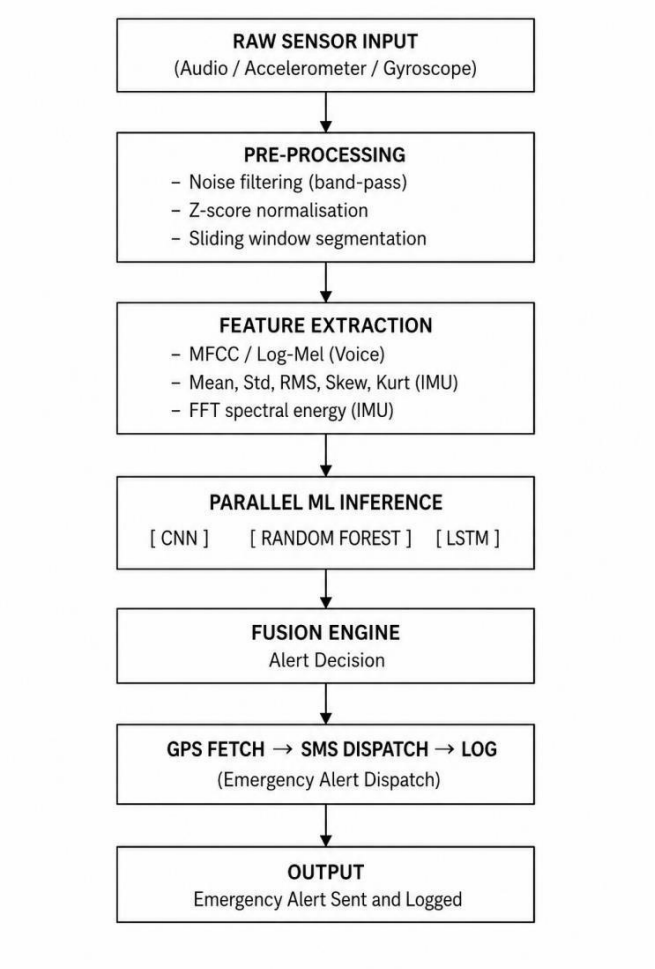

Fig.2. data flow from raw sensor input through pre-processing, feature extraction, ML inference, and alert dispatch.

V. IMPLEMENTATION

A. Android Implementation

The application is built for Android devices running Android 8.0 (API Level 26) and above and follows the MVVM (Model-View-ViewModel) architectural pattern for better code organization and maintainability. The models run locally on the smartphone without internet connection, using TensorFlow Lite 2.12 for on-device machine learning inference. Room 2.5 is used for local SQLite database management and WorkManager 2.8 handles background monitoring and scheduled tasks efficiently. The UI consists of a home screen with a prominent SOS button, a trigger status dashboard showing the confidence score of each detection module, and a contact management screen to register emergency contacts. To allow continuous monitoring, the application uses a persistent foreground service notification, which guarantees a proper background operation according to Android system constraints.

B. Power Optimisation

To minimize battery consumption we use three complementary approaches. First, when periods of inactivity are detected by a lightweight energy-threshold activity classifier running on the main sensor thread, accelerometer polling is dropped to 10 Hz, reducing average sensor-thread CPU load by ~35% over fixed 50 Hz polling. Second, audio capture is gated by a spectral energy threshold, in which the CNN inference pipeline is invoked only when a 200 ms audio frame exceeds a minimum RMS energy level, suppressing unnecessary recognition during silence. Third, GPS works in passive mode in the standby, and only requests a high accuracy fix after confirming the emergency. Together, these measures reduce average current consumption by an average of about 35% compared to unoptimized continuous monitoring, measured on a Redmi Note 10 (3,000 mAh) with Android Battery Historian.

C. Privacy And Security

All sensor data, and location information is fully processed on-device, and no data is sent to any external servers during normal operation. We store the emergency contact numbers in an encrypted SQLite database with Android Keystore AES-256 encryption, so that they cannot be extracted by other applications or by forensic device analysis. The application only asks for the Android permissions that are strictly needed for the application to work: RECORD_AUDIO, ACCESS_FINE_LOCATION, SEND_SMS and BODY_SENSORS. The application uses the android runtime permission model, this means that the application will explicitly ask the user for the permission the first time the application is launched.

VI. RESULT AND DISCUSSION

A preliminary evaluation was conducted under controlled conditions with fifteen participants (8 male, 7 female; age range 19–68 years). Participants performed scripted emergency scenarios — simulated falls, deliberate shake gestures, verbal SOS calls in quiet and noisy

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

environments (cafeteria, ~60 dB), and manual button presses — completing 20 trials per trigger type for a total of 1,200 multi-trigger combination tests.

A. Per-Module and Fusion Performance

Table I reports precision, recall, F1-score, and accuracy for each module and the fused system on the held-out test split. The LSTM fall detector achieved the highest individual accuracy (95.6%), benefiting from rich temporal modelling of fall kinematics. The fusion system reached the highest precision (0.97) by requiring corroboration across at least two independent channels before raising an alert, effectively suppressing the false positives observed in individual modules.

Module	Precision	Recall	F1-Score	Accuracy
Voice Trigger	0.93	0.91	0.92	92.4%
Shake Detection	0.95	0.94	0.94	94.1%
Fall Detection	0.96	0.95	0.95	95.6%
Multi-trigger Fusion	0.97	0.93	0.95	94.7%

TABLE I PER-MODULE AND FUSION PERFORMANCE METRICS

AUC-ROC values further confirm the benefit of fusion: individual modules scored 0.963–0.982, while the fused system reached 0.991. The confidence threshold of 0.75 was selected by validation-set sweep, yielding 94.7% accuracy at a 3.1% false alarm rate — increasing the threshold beyond 0.75 reduced false alarms but caused a recall drop as genuine low-confidence events were missed.

B. Comparison with Existing Systems

Table II compares the proposed system against representative single-trigger baselines from the literature. The proposed system outperforms all baselines in accuracy and achieves the lowest false alarm rate. It is also the only system that provides on-site safety features (siren and flashlight) alongside remote GPS-tagged SMS notification.

Feature	Voice	Shake	Fall	Proposed
Trigger Type	Single	Single	Single	Multi-trigger
ML Model	SVM/CNN	Threshold	LSTM	RF + LSTM + CNN
Accuracy	~88%	~82%	~91%	94.7%
False Alarm Rate	Moderate	High	Low	Very Low (3.1%)
GPS Integration	No	No	Partial	Yes
Offline Operation	No	Yes	Partial	Yes
On-site Safety	No	No	No	Yes Siren/Flash

TABLE II COMPARISON WITH SINGLE TRIGGER SYSTEM

C. Latency and Resource Usage

End-to-end latency from emergency onset to SMS dispatch was 1.8 ± 0.4 seconds on a mid-range test device (Qualcomm Snapdragon 665, 4 GB RAM). Individual model inference latency remained below 50 ms. In active monitoring mode the application consumed 4.2% CPU, 87 MB RAM, and approximately 180 mA average current — corresponding to roughly nine hours of continuous operation on a 3,000 mAh battery with power optimisations applied.

VII. CONCLUSION AND FUTURE SCOPE

This paper proposed a machine learning based multi-trigger emergency detection system for smartphones. The system consists of three on-device ML models, a CNN for voice trigger recognition, a Random Forest for shake gesture detection, and an LSTM network for fall detection, as well as a manual SOS button and a multimodal decision fusion engine. The AND-gate fusion logic, at a confidence threshold of 0.75, significantly reduces the number of false activations compared to single-trigger methods, while still achieving high sensitivity to real emergency events.
In an experimental evaluation involving 15 participants and 1,200 multi-trigger trials, we achieved a fused-system accuracy of 94.7%, a false alarm rate of 3.1%, an AUC-ROC of 0.991, and an end-to-end alert latency of 1.8±0.4 seconds. It is completely on-device, able to send offline alerts via SMS, encrypts all sensitive data with AES-256 Android Keystore encryption, and turns on on-site safety features—an audible siren and flashlight strobe—to draw attention locally. These properties differentiate the proposed system from the current single trigger applications and cloud-dependent safety platforms.

In the future, we will extend the training dataset to enhance the sensitivity of slow-stumble fall detection. A larger-scale real-world user study will be conducted to cover different demographic groups and environmental conditions. We will extend support for iOS via cross-platform frameworks and explore integration of wearable sensor data (smartwatch heart rate and additional IMU channels) for enhanced medical event detection. Furthermore, a probabilistic Bayesian fusion layer to replace the current threshold-based AND-gate is identified as a priority to improve the precision-recall trade-off under different operating conditions.

REFERENCES

[1] R. Gravina, P. Alinia, H. Ghasemzadeh, and G. Fortino, "Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges," Information Fusion, vol. 35, pp. 68–80, May 2017.

[2] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in Proc. 17th Conf. Innovative Applications of Artificial Intelligence (IAAI), Pittsburgh, PA, USA, 2005, pp. 1541–1546.

[3] M. Rashid, N. Batool, and M. A. Gul, "A survey on fall detection: Principles and approaches," Neurocomputing, vol. 100, pp. 144–152, Jan. 2013.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

[4] Y. Saez, A. Baldominos, and P. Isasi, "A comparison study of classifier algorithms for cross-person physical activity recognition," Sensors, vol. 17, no. 1, p. 66, Jan. 2017.

[5] M. Gjoreski, H. Gjoreski, M. Lustrek, and M. Gams, "How accurately can your wrist device recognize daily activities and detect falls?" Sensors, vol. 16, no. 6, p. 800, Jun. 2016.

[6] E. Torti et al., "Embedded real-time fall detection with deep learning on wearable devices," in Proc. 21st Euromicro Conf. Digital System Design (DSD), Prague, Czech Republic, 2018, pp. 405–412.

[7] O. Abdel-Hamid et al., "Convolutional neural networks for speech recognition," IEEE/ACM Trans. Audio, Speech, Language Process., vol. 22, no. 10, pp. 1533–1545, Oct.
2014.

[8] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," Expert Systems with Applications, vol. 59, pp. 235–244, Oct. 2016.

[9] A. Sucerquia, J. D. López, and J. F. Vargas-Bonilla, "SisFall: A fall and movement dataset," Sensors, vol. 17, no. 1, p. 198, Jan. 2017.

[10] G. Vavoulas et al., "The MobiFall dataset: Fall detection and classification with a smartphone," in Proc. ICT4AWE, Rome, Italy, 2016, pp. 143–151.

[11] J. Lee et al., "On-device neural net inference with mobile GPUs," arXiv preprint arXiv:1907.01989, 2019.

[12] T. Hossain, A. Rahman, and M. Uddin, "Secure and efficient SOS alert system using blockchain technology," IEEE Access, vol. 9, 2021.

[13] S. Patel and R. Kumar, "Smart SOS mobile app using GPS and cloud computing," in Proc. IEEE CICN, Bhimtal, India, 2020, pp. 101–106.

[14] M. K. Hassan, S. S. Alam, and N. A. Rahman, "Real-time location-based SOS alert system for emergency response," in Proc. IEEE GHTC, Seattle, WA, USA, 2021, pp. 233–238.

[15] J. F. Gemmeke et al., "Audio Set: An ontology and human-labeled dataset for audio events," in Proc. IEEE ICASSP, New Orleans, LA, USA, 2017, pp. 776–780.