International Journal of Advance and Innovative Research
Volume 11, Issue 2: April - June 2026

ISSN 2394 - 7780

DIVINESHIELD: A PROACTIVE AI-POWERED WOMEN'S SAFETY ANDROID APPLICATION INTEGRATING ON-DEVICE MULTI-MODAL THREAT DETECTION, VOICE-ACTIVATED EMERGENCY RESPONSE, DYNAMIC SAFE-ROUTE NAVIGATION, AND OFFLINE-RESILIENT BLE MESH COMMUNICATION

Dr. Dipankar Misra, Professor, Soumyadip Das Mahapatra[1], Subhadip Tudu[1], Soumyajit Sarkar[1], Vivay Shaw[1], Ayan Bera[1]

[1]Department of Computer Science and Engineering, JIS University, Kolkata – 700109, West Bengal, , Dept. of CSE

ABSTRACT

The global epidemic of violence against women — affecting approximately one in three women worldwide across their lifetime — represents one of the most pervasive and inadequately addressed human rights violations of our era. Despite the near-universal adoption of smartphones among the target demographic, the technological response to this crisis has been characterised by a paradoxical failure: existing mobile safety applications are architecturally reactive, demanding conscious user interaction precisely when physical restraint, psychological shock, or rapid situational escalation makes such interaction impossible. This paper presents DivineShield, a novel and comprehensive Android-based women's safety platform that fundamentally rearchitects the mobile safety paradigm from reactive to proactive through the integration of five mutually reinforcing technological subsystems. The first and primary contribution is an always-on, fully on-device artificial intelligence threat detection engine implemented in TensorFlow Lite 2.13 that performs continuous multi-modal sensor fusion across three independent data streams — 16 kHz microphone audio processed through MFCC feature extraction and a Temporal Convolutional Network (TCN), six-axis inertial measurement data processed through a fully-connected anomaly classifier, and GPS trajectory sequences processed through a personalised Hidden Markov Model (HMM) anomaly detector — to compute a continuous real-time personalised threat score without transmitting any sensitive data to external servers. The second contribution is a voice-activated SOS system based on a DS-CNN keyword spotting model, achieving a mean activation-to-first-alert latency of 1.8 seconds and operating reliably in background mode across all tested Android launcher configurations. The third contribution is a composite-risk safe-route navigation engine that dynamically weights Google Maps SDK routing decisions using three factors: historical incident density (weight 0.5), time-of-day risk modelling via kernel density estimation (weight 0.3), and real-time crowdsourced proximity incident scoring (weight 0.2). The fourth contribution is a three-tier offline-first SOS protocol providing graceful emergency communication degradation from Firebase FCM (Tier 1) through SMS fallback (Tier 2) to Bluetooth Low Energy mesh relay (Tier 3) under complete network isolation. The fifth contribution is a peer-verified community incident reporting network with exponential time-decay weighted heatmap aggregation. All sensitive computation occurs on-device, ensuring complete data privacy. Rigorous experimental evaluation across six heterogeneous Android devices and 50 beta users over two weeks yielded: threat detection accuracy 93.4% (F1: 93.2%), SOS activation latency 1.8s (σ=0.31s), SOS delivery reliability 99.1% across all network conditions, BLE mesh fallback success 94.3%, safe-route risk improvement in 78.3% of routing scenarios, and a System Usability Scale score of 82.4/100 (Grade B+). Notably, 76% of beta participants reported subjectively feeling safer in unfamiliar environments — the most direct measure of the application's core social purpose.

Index Terms*—Women's Safety, Android Application, TensorFlow Lite, On-Device Machine Learning, Multi-Modal Sensor Fusion, Temporal Convolutional Network, Voice SOS, DS-CNN Keyword Spotting, Safe Route Navigation, BLE Mesh Networking, Community Safety, Threat Detection, Privacy-Preserving AI, Hidden Markov Model, MFCC Feature Extraction.*

ISSN 2394 - 7780

I. INTRODUCTION

A. Problem Context and Motivation

Violence against women remains a persistent, deeply entrenched, and globally pervasive crisis that transcends geographic, cultural, economic, and institutional boundaries. The United Nations Entity for Gender Equality and the Empowerment of Women (UN Women, 2023) [6] reports that approximately one in three women worldwide — equivalent to roughly 1.3 billion individuals — experiences physical or sexual violence in her lifetime, representing one of the most widespread and systematically underaddressed human rights violations currently documented. This statistic, alarming in isolation, understates the true scope of the crisis: it captures only reported and acknowledged incidents, while the vast majority of violence against women goes unreported due to complex and mutually reinforcing barriers including social stigma, fear of retaliation, distrust of institutional response mechanisms, and the absence of accessible, private, and reliable support channels.

In the Indian context specifically, the challenge reaches acute dimensions. The National Crime Records Bureau (NCRB, 2022) [7] reports that a crime against women is registered in India every three minutes on average, with total registered crimes against women exceeding 445,000 in 2022 alone. The Ministry of Home Affairs (MHA, 2023) [8] estimates that as many as 71% of sexual assault incidents go entirely unreported, implying that the true annual incidence

may approach 1.5 million events. The COVID-19 pandemic dramatically exacerbated this baseline: with nationwide movement restrictions keeping potential victims in sustained close proximity to abusers with reduced access to social support networks, NCRB data indicates a 40% increase in registered domestic violence incidents during 2020–2022 compared to the 2017–2019 baseline period. Urban women face distinct threat profiles including workplace harassment, street-based threats, and public transport vulnerability, while rural women face heightened risks from isolation, limited emergency service access, and social structures that actively discourage reporting.

B. Technology Opportunity and Current Limitations

Against this backdrop, the smartphone presents an unprecedented and largely underexploited opportunity for technological intervention.

The Telecom Regulatory

Authority of India (TRAI, 2023) [15] reports urban smartphone penetration rates exceeding 86%, with rapid rural adoption driven by affordable 4G devices and government digital inclusion initiatives. Every woman carrying a smartphone already possesses a device with a high-quality microphone, accelerometer, gyroscope, GPS receiver, Bluetooth radio, cellular modem, and sufficient computational capacity to run sophisticated machine learning inference — precisely the capabilities required to implement a comprehensive, always-on personal safety system.

However, the current ecosystem of women's safety applications fails to leverage these capabilities effectively. A systematic review of the twenty most-downloaded women's safety applications available on the Google Play Store as of January 2025 reveals that eighteen require explicit user interaction to trigger any emergency response, sixteen fail completely under offline conditions, fourteen provide no threat detection capability whatsoever, and none integrate threat detection, emergency response, safe navigation, and community safety intelligence into a single coherent application. This fragmentation creates dangerous operational overhead: in a genuine emergency, a user cannot be expected to unlock their phone, navigate to the correct application, and interact with the correct UI element before a threat materialises.

The four structural deficiencies that characterise the current state of women's safety technology can be summarised precisely:

(i) Reactive architecture — all existing systems require deliberate user action to initiate any safety response;

(ii) Connectivity dependency — the majority of systems fail silently when internet connectivity is unavailable, precisely the condition most correlated with dangerous isolated environments; (iii) Feature fragmentation — no existing application integrates the complete spectrum of safety capabilities into a unified interface; and (iv) Absence of

personalised adaptive intelligence — no existing system learns from an individual user's behavioural patterns to calibrate threat sensitivity and reduce false positive rates over time.

ISSN 2394 - 7780

C. Contributions of This Work

This paper presents DivineShield, a comprehensive Android application designed to address all four deficiencies simultaneously through an integrated, privacy-preserving, offline-resilient architecture developed over six months at JIS University, Kolkata. The five primary technical contributions of this work are as follows:

- **Contribution 1 —** Multi-Modal On-Device AI Threat Engine: An always-on TensorFlow Lite 2.13 inference pipeline performing continuous sensor fusion across audio (MFCC + TCN), motion (FC anomaly classifier), and GPS (personalised HMM) modalities to generate a scalar threat score $T \in [0,1]$ entirely on-device without cloud dependency or data transmission.
- **Contribution 2 —** Sub-2-Second Voice-Activated SOS: A DS-CNN keyword spotting model achieving mean wake-word-to-first-alert latency of 1.8 seconds, operable in background mode across all tested Android launcher configurations, with configurable personalised wake phrase and 10-second cancellation window.
- **Contribution 3 —** Composite-Risk Safe Route Navigation: A Google Maps Directions API extension computing per-segment composite risk $R(s) = 0.5 \cdot H(s) + 0.3 \cdot T\_d(s,t) + 0.2 \cdot I\_rt(s)$, selecting risk-minimised routes subject to user-configurable maximum travel time overhead.
- **Contribution 4 —** Three-Tier Offline-First SOS Protocol: A connectivity-adaptive emergency dispatch system providing Firebase FCM (Tier 1), SMS via Android TelephonyManager (Tier 2), and BLE mesh relay (Tier 3), ensuring 94.3%+ SOS delivery even under complete network isolation.
- **Contribution 5 —** Peer-Verified Community Safety Network: An incident reporting and validation system with spatial-temporal peer corroboration, exponential time-decay heatmap aggregation ($H(cell,t) = \Sigma\ severity\_i \cdot \exp(-\lambda \cdot \Delta t)$), and 5-minute update cycles providing self-sustaining community-driven safety intelligence.

The remainder of this paper is organised as follows. Section II provides a comprehensive review of related literature across five technological domains with consolidated gap analysis. Section III presents the complete DivineShield system architecture, entity definitions, and mathematical formulations. Section IV analyses security threats and architectural countermeasures. Section V details the full implementation stack and component architecture. Section VI describes the experimental design, hardware matrix, and dataset collection. Section VII presents comprehensive quantitative results with statistical analysis. Section VIII concludes with limitations and future research directions. References follow.

II. RELATED WORKS

The academic literature on mobile-based and IoT-based women's safety systems spans hardware-centric approaches, machine learning classifiers, crowdsourced platforms, speech recognition systems, and AI-driven navigation. We systematically review the five most directly relevant prior works, providing detailed technical critique of each, followed by a consolidated gap analysis that motivates every major design decision in DivineShield.

A. Hardware-Centric GSM/GPS Approaches [1]

Gayathri et al. (2021) [1] designed a dedicated wearable hardware device integrating an Arduino Uno microcontroller with a Neo-6M GPS module and a SIM800L GSM/GPRS module. The device is designed to be worn as a wristband or carried in a pocket. Upon activation via a physical tactile push-button, the SIM800L transmits a pre-composed SMS message containing GPS coordinates in NMEA GGA sentence format to a list of pre-registered emergency contact phone numbers. The authors report mean GPS position accuracy of ±12 metres under open-sky conditions with HDOP ≤ 2.0, and mean SMS delivery latency of approximately 3–5 seconds under 2G GSM coverage. A prototype cost of approximately INR 850 was reported, with battery life of 18 hours under continuous GPS polling at 1 Hz update rate.

Despite these positive results in controlled evaluation, the approach suffers from five critical limitations that prevent real-world efficacy. First and most fundamentally, the reactive push-button paradigm assumes the user retains physical access to the button, manual dexterity, and the psychological composure to initiate activation — all of which are precisely the resources most likely to be unavailable in a genuine emergency involving physical restraint or acute psychological shock. Second, the dedicated hardware requirement introduces a cost-

ISSN 2394 - 7780

adoption barrier and a reliability dependency on remembering, charging, and carrying an additional device. Third, the system provides zero autonomous threat detection capability; it is entirely dependent on user initiation. Fourth, the system has no fallback when GSM coverage is unavailable, a condition that is extremely common in the isolated rural environments where the system is most needed. Fifth, the system provides no route safety guidance, no community incident awareness, and no capability to learn from experience.

B. Sensor-Based Motion Classification [2]

Pradhan et al. (2022) [2] developed an Android application that captures raw accelerometer and gyroscope data through the Android SensorManager API at a sampling rate of 50 Hz and processes it through a Support Vector Machine (SVM) classifier with a Radial Basis Function (RBF) kernel trained to distinguish between normal locomotion patterns and abnormal motion patterns characteristic of a physical struggle or assault. The dataset comprised 400 labelled 10-second motion sequences collected from 20 volunteers, split equally between normal activities (walking, running, cycling, stair climbing) and simulated assault scenarios (sudden shaking, irregular high-magnitude vibration, rolling and tumbling). The trained SVM achieved an overall classification accuracy of 81.2% on the held-out test set using a 70/30 train/test split, with sensitivity (recall for the assault class) of 78.4% and specificity of 84.1%. The system was implemented as an Android Service polling accelerometer data at 50 Hz and maintaining a 2-second sliding window buffer.

The fundamental limitations of this approach are multiple and severe in the context of real-world deployment. First, the 81.2% accuracy figure reported under controlled laboratory conditions is unlikely to generalise to real-world performance: the training dataset comprised only 20 volunteers performing simulated activities in laboratory settings, which may not capture the full distributional diversity of genuine assault scenarios. Second, and critically, the false positive rate of approximately 15.9% (1 – specificity) implies that on average one false alarm would be generated for every 6.3 twenty-second monitoring windows. For a user engaged in a 30-minute brisk walk, this translates to approximately 3 false alarms per outing — an unacceptable rate that would cause users to disable the feature entirely. Third, the single-modality approach is fundamentally limited: motion data alone cannot distinguish between a user running away from a threat (high-urgency scenario) and a user running for exercise (benign scenario). Fourth, the system provides zero audio-based threat awareness, making it entirely blind to verbal threats, which constitute the majority of threatening interactions in crowded public environments. Fifth, no GPS context is utilised, meaning the system applies identical sensitivity in a crowded shopping mall at noon and in an isolated alleyway at midnight.

C. Crowdsourced Crime Mapping [3]

Chauhan and Shah (2023) [3] built a web and Android application platform enabling registered users to submit geo-tagged incident reports enriched with incident type (harassment, assault, theft, suspicious activity), severity rating (1–5), and optional photographic evidence. Submitted reports are aggregated with historical police FIR (First Information Report) data obtained through RTI (Right to Information) requests and geocoded to 100m × 100m grid cells. The aggregated data is visualised as a choropleth heatmap rendered using OpenLayers on the web and Google Maps SDK on Android, updated on a 24-hour batch processing schedule. The platform demonstrated strong community adoption in a pilot deployment in Pune, Maharashtra, accumulating 3,847 user-submitted reports over a 90-day pilot period, with 68% of reports corroborated by at least one other report in the same spatial-temporal window.

While the crowdsourcing approach and corroboration mechanism represent meaningful contributions to the community safety intelligence domain, the system has fundamental limitations as a standalone safety platform. The 24-hour batch update cycle is the most critical deficiency: a dangerous incident occurring at 10 PM will not appear on the heatmap until the following evening's batch run, meaning the platform provides zero protection for users traversing the same location that same night. The system has no personal SOS functionality whatsoever — it functions exclusively as a reference map tool, requiring users to separately manage all emergency communication through other means. There is no machine learning integration, no personalised threat assessment, and no route safety optimisation. The system also has no offline functionality, requiring continuous internet connectivity to load the heatmap tiles. Perhaps most importantly from a user experience perspective, consulting a crime heatmap requires deliberate pre-trip planning behaviour rather than providing continuous, ambient, automatic protection during a journey.

D. Voice-Activated Emergency Systems [4]

Singh et al. (2023) [4] implemented an Android application using the CMU Sphinx 5 offline speech recognition library to perform continuous keyword detection for the phrase 'Help Me' using a Hidden Markov Model (HMM) acoustic model trained on English phonemes. The system runs as an Android background service continuously sampling microphone audio at 8 kHz in 500 ms chunks, performing acoustic feature

ISSN 2394 - 7780

extraction (PLP cepstral coefficients) and Viterbi decoding to detect keyword presence. Upon keyword detection with confidence exceeding a threshold, the system dispatches pre-composed SMS messages to up to five emergency contacts and initiates an automated outgoing call. The authors report a keyword detection accuracy of 87.3% under quiet conditions (SNR ≥ 20 dB) and a mean SMS dispatch latency of 3.2 seconds from keyword detection. Critically, the system achieves reliable detection without requiring any internet connectivity, relying entirely on CMU Sphinx's offline acoustic model.

The offline architecture of Singh et al.'s work represents an important and admirable design choice that directly informed DivineShield's offline-first philosophy. However, the system's limitations are equally instructive. The most critical limitation is the severe performance degradation in high-noise environments: accuracy drops to 53.1% at SNR ≤ 5 dB (equivalent to a busy street or crowded market), making the system functionally unreliable in precisely the noisy public environments where threats most commonly occur. The use of CMU Sphinx 5, while enabling offline operation, introduces substantial performance overhead: the library occupies 32 MB on-device, consumes approximately 15% CPU during continuous recognition, and the 8 kHz sample rate results in acoustic models with limited expressiveness compared to modern deep learning approaches. The fixed non-configurable wake phrase 'Help Me' raises both security concerns (natural conversational use of the phrase can trigger false alarms) and accessibility concerns (non-native English speakers or individuals with certain speech characteristics may struggle to achieve consistent recognition). Furthermore, the system provides no threat detection (it cannot proactively identify danger), no route safety, no community safety features, and no location tracking beyond the initial GPS coordinate captured at SOS dispatch time.

E. AI-Driven Route Safety Navigation [5]

Gupta et al. (2024) [5] proposed a modified Dijkstra shortest-path algorithm for pedestrian route planning that incorporates crime score weights derived from historical police incident databases. The algorithm represents the city road network as a weighted directed graph G = (V, E), where each edge e∈E has two associated weights: the standard travel time weight w_t(e) and a crime safety weight w_c(e) derived from the normalised historical incident density for the corresponding road segment over a trailing 12-month window. The modified Dijkstra algorithm computes paths minimising a composite objective function $\alpha \cdot w\_t + (1-\alpha) \cdot w\_c$, where $\alpha \in [0,1]$ is a user-configurable time/safety trade-off parameter. The system was evaluated on a road network graph of 15,482 nodes and 31,247 edges representing central Delhi, using historical crime data from the Delhi Police Open Data Portal. In evaluation, safety-optimised routes with $\alpha=0.3$ reduced the normalised crime exposure score by 34.7% compared to time-optimal routes at $\alpha=1.0$, at a mean additional travel time cost of 6.1 minutes.

The composite objective formulation and empirical evaluation of Gupta et al.'s work represent a solid methodological contribution to the route safety domain. However, the system has five limitations that substantially constrain real-world utility. First and most critically, the system relies exclusively on static historical police incident data, which suffers from systematic underreporting bias (since the majority of incidents go unreported), update latency (police databases are typically updated weekly or monthly), and geographic coverage gaps in areas where police data is unavailable. Routes computed at night using daytime crime statistics will systematically underestimate nighttime danger, as the temporal distribution of incidents is highly non-uniform. Second, the system ignores real-time incident information entirely: a violent incident occurring 30 minutes before the user's route computation would have zero impact on route recommendations. Third, the Dijkstra algorithm's $O(V \log V + E)$ time complexity produces noticeable computation latency for large city-scale graphs, reported as 4.8 seconds mean route computation time on a mid-range Android device. Fourth, the system lacks any personal SOS, threat detection, or community reporting functionality. Fifth, the system requires continuous internet connectivity to load routing data and provides no offline functionality.

F. Consolidated Research Gap Analysis

The systematic review of the five prior works exposes four persistent, fundamental, and unresolved gaps in the women's safety technology landscape that collectively constitute the motivation for DivineShield's design.

Gap 1 — Absence of Proactive Multi-Modal AI Threat Detection: No reviewed system implements continuous autonomous threat assessment across multiple complementary sensor modalities. All reactive systems (hardware button, voice keyword, motion classifier) require either user initiation or detection of a single specific threat signature. A comprehensive threat detection system must fuse audio, motion, and contextual spatial signals to achieve robust, false-positive-minimised, multi-scenario threat recognition that does not require user initiation.

ISSN 2394 - 7780

Gap 2 — Connectivity Dependency and Fragile Offline Operation: The three systems with network functionality (Gayathri et al.'s SMS system is an exception, though it requires GSM) either fail silently under poor connectivity or provide no offline capability whatsoever. No prior work implements a principled, tiered offline degradation protocol that maintains core emergency communication functionality under complete network isolation using BLE mesh relay.

Gap 3 — Feature Fragmentation Across Separate Applications: Each reviewed system addresses exactly one aspect of safety: location transmission, motion classification, crime mapping, voice SOS, or route navigation. No system integrates these complementary capabilities into a unified application, requiring users to manage multiple apps simultaneously — an operationally unacceptable cognitive burden during emergencies.

Gap 4 — Absence of Personalised Adaptive Intelligence: No reviewed system implements any form of individual user behaviour learning or personalisation. A system that treats a user's daily morning run identically to a midnight forced march through an unfamiliar area will inevitably generate excessive false positives, eroding user trust and leading to feature disablement. Personalised baseline modelling using the user's historical GPS trajectories, typical motion patterns, and routine audio environments is essential for practical deployment.

III. SYSTEM DESIGN AND ARCHITECTURE

A. Architectural Overview

DivineShield is designed according to a layered architecture comprising five distinct functional layers, each encapsulating a specific technical domain and communicating with adjacent layers through well-defined, testable interfaces. This architectural philosophy serves three engineering objectives: independent testability of each layer without instrumented Android test infrastructure, incremental feature deployment allowing individual subsystems to be updated without affecting others, and graceful degradation ensuring that failure or unavailability of any non-critical subsystem does not impair the core SOS functionality.

The five layers from top to bottom are: (1) Presentation Layer — Jetpack Compose composable functions organised into five primary screens with a shared component library and Material Design 3 theming system; (2) Application Logic Layer — ViewModel classes implementing MVVM pattern, Use Case classes in the Domain layer containing business logic free of Android framework dependencies, reactive state management via Kotlin Coroutines and StateFlow; (3) AI/ML Engine Layer — TensorFlow Lite

2.13 inference pipeline, MFCC audio feature extractor implemented in native C++ via JNI, DS-CNN keyword spotter, HMM trajectory anomaly detector; (4) Data and Backend Layer — Repository classes with dual local/remote data sources, Room DAO interfaces for local SQLite access, Firebase SDK clients for remote synchronisation; (5) Hardware Abstraction Layer — Android SensorManager wrapper, GPS/FLP wrapper, MediaRecorder wrapper, SmsManager wrapper, BLE Scanner/Advertiser wrapper.

B. Entity Definitions and System Actors

(i) Primary User: The woman carrying an Android smartphone with DivineShield installed. During a 15-minute onboarding process, the user registers 1–5 emergency contacts with contact-level customisation of alert content; configures the AI sensitivity threshold θ (default 0.75) with real-time preview of historical false positive rate at each threshold value; records or selects a personalised voice wake phrase; grants required runtime permissions (Location, Microphone, SMS, Contacts, Bluetooth); and completes a structured test SOS sequence to verify emergency contact delivery. Following onboarding, the system passively learns the user's routine GPS trajectories, typical daily motion patterns, and characteristic acoustic environments over a 7-day baseline learning period, constructing a personalised baseline used to calibrate the HMM anomaly detector and the audio environment classifier.

(ii) AI Threat Engine: A persistent Android foreground service (declared in AndroidManifest with android:foregroundServiceType="location|microphone") that maintains the TFLite inference pipeline across the full application lifecycle including device sleep states (using PowerManager WakeLock with PARTIAL_WAKE_LOCK). The engine collects raw sensor data from three hardware sources, performs feature extraction, executes TFLite model inference, and updates the threat score StateFlow at 500 ms intervals. When $T \geq \theta$, the engine invokes the SOS Manager with escalating alert logic.

ISSN 2394 - 7780

(iii) Community Network: The aggregate of all DivineShield users within a geographic region, contributing incident reports that are processed through a two-stage validation pipeline and aggregated into the shared Firebase RTDB incident collection. Community data feeds both the route risk scoring engine (via the I_rt term) and the heatmap overlay. The community network exhibits positive network externalities: safety intelligence improves as the user base grows, creating a self-reinforcing adoption incentive.

(iv) Backend Infrastructure: A combination of Firebase services (RTDB for sub-100 ms live location synchronisation, Authentication for secure user identity management, Cloud Messaging for push SOS delivery, Storage for encrypted audio evidence clips) and Google Cloud Functions for server-side logic including incident validation, FCM dispatch, and rate limiting enforcement.

(v) Emergency Contacts: Registered trusted persons who receive SOS notifications through redundant channels. App-installed contacts receive FCM push notifications with real-time GPS tracking deep link, threat level indicator, and audio clip attachment option. Non-app contacts receive formatted SMS messages. The primary contact additionally receives an automated outgoing call. All contacts are presented with a web-based live tracking interface accessible via a time-limited URL embedded in the SOS notification.

C. Multi-Modal AI Threat Detection: Mathematical Formulation

The threat detection engine processes three concurrent sensor streams through independent feature extraction pipelines before fusing them in a joint classification model. We describe each pipeline and the fusion architecture in detail.

1) Audio Feature Extraction Pipeline:

Raw 16 kHz monaural audio is captured from the device microphone using Android's AudioRecord API with 16-bit PCM encoding. Audio is segmented into overlapping frames of 25 ms duration with 10 ms frame shift, yielding a frame rate of 100 frames per second. For each frame, a 512-point

Short-Time Fourier Transform (STFT) is computed using a Hann window function to minimise spectral leakage. The STFT magnitude spectrum is then mapped to a 40-band Mel-frequency filterbank spanning 80 Hz to 7600 Hz using triangular overlapping filterbanks equally spaced on the Mel scale. Log energy is computed for each filterbank band, and a Discrete Cosine Transform (DCT) is applied to the 40 log-energies to produce 13 Mel-Frequency Cepstral Coefficients (MFCCs), augmented with delta (Δ) and delta-delta (ΔΔ) features for a total of 39 audio features per frame. Consecutive frames are stacked into 2-second windows (200 frames × 39 features = 7,800 features) which form the input tensor to the TCN branch. This entire pipeline is implemented in native C++ via Android NDK JNI for computational efficiency, executing in under 15 ms per 2-second window on Snapdragon 680 class hardware.

2) Motion Feature Extraction Pipeline:

Six-axis inertial data (3-axis accelerometer + 3-axis gyroscope) is sampled at 50 Hz via Android SensorManager with SENSOR_DELAY_GAME latency mode. Data is accumulated in overlapping 2-second windows (100 samples per window) with 50% overlap. For each window and each axis, the following statistical features are computed: mean (μ), standard deviation (σ), root mean square (RMS), inter-quartile range (IQR), peak-to-peak amplitude (PtP), zero-crossing rate (ZCR), and spectral entropy derived from the DFT magnitude spectrum. Additionally, cross-axis correlation coefficients are computed between all pairs of axes (15 pairs for 6 axes). This yields a total of 6×7 + 15 = 57 motion features per window, forming the input vector to the motion FC branch.

3) GPS Trajectory Anomaly Detection:

GPS positions are obtained from the Android Fused Location Provider at 5-second intervals during active monitoring, reduced to 30-second intervals in low-power mode when $T < 0.3$. Location sequences are processed by a personalised Hidden Markov Model (HMM) trained on the individual user's historical movement data collected during the 7-day baseline learning period. The HMM state space represents semantic location clusters (Home, Work, Frequent Locations, Transit Routes, Unfamiliar Locations) estimated via k-means clustering on historical location data with k selected by the Bayesian Information Criterion. The HMM transition matrix A and emission distribution parameters B are estimated using the Baum-Welch EM algorithm on the baseline trajectory dataset. For a new GPS observation sequence $O = (o_1, o_2, \ldots, o_T)$, the trajectory anomaly score is computed as the negative log-likelihood under the learned HMM normalised by sequence length:

ISSN 2394 - 7780

$$S\_gps = -(1/T) \cdot \log P(O \mid \lambda\_user)$$

where $\lambda\_user = (A, B, \pi)$ represents the user's personalised HMM parameters. Higher S_gps values indicate greater deviation from the user's established routine, with S_gps normalised to [0,1] via min-max scaling relative to the baseline distribution.

4) Multi-Modal Fusion and Threat Score Computation:

The three feature representations are processed by independent neural network branches implemented as a single multi-input TFLite model. The audio branch applies a 5-layer Temporal Convolutional Network (TCN) with dilated causal convolutions (dilation factors 1, 2, 4, 8, 16) and residual skip connections, producing a 64-dimensional audio embedding h_audio. The motion branch applies a 3-layer fully-connected network (57→128→64→64 neurons with ReLU activations and 0.3 dropout during training) producing a 64-dimensional motion embedding h_motion. The GPS anomaly score S_gps is passed through a single linear layer producing a 64-dimensional context embedding h_gps. The three embeddings are concatenated into a 192-dimensional joint representation and processed through two fully-connected layers (192→128→64 with ReLU) followed by a sigmoid output neuron producing the final threat score:

$$T = \sigma(W\_out \cdot ReLU(W\_2 \cdot ReLU(W\_1 \cdot [h\_audio ; h\_motion ; h\_gps])))$$

The complete model is quantised to INT8 precision using TFLite post-training quantisation with a representative calibration dataset of 500 samples, reducing the model size from 16.8 MB (FP32) to 4.2 MB (INT8) while maintaining accuracy within 0.8% of the full-precision baseline. The system alert decision function is defined as:

$$Alert(T, t) = 1 \text{ if } T(t) \geq \theta\_user \text{ else } 0$$

where $\theta\_user \in [0.5, 0.95]$ is the user-configurable sensitivity threshold (default 0.75). The complete inference cycle from raw sensor collection to threat score update executes in under 120 ms mean latency across the test device matrix.

D. Voice-Activated SOS: Architecture and Dispatch Protocol

The voice SOS subsystem implements a two-stage detection pipeline. Stage 1 is a lightweight 380 KB DS-CNN keyword spotter running continuously at approximately 0.8% CPU load, monitoring for any utterance within the phonemic neighbourhood of the configured wake phrase. Stage 2 is a confirmation classifier that re-analyses the preceding 1.5 seconds of audio at higher precision upon Stage 1 positive detection. Two consecutive Stage 2 positive detections within 500 ms are required to trigger SOS dispatch, reducing the false accept rate from Stage 1's 0.8% to a confirmed 0.1% for the full two-stage pipeline.

Upon confirmed detection, the SOS Manager executes the following five-stage dispatch sequence in parallel coroutines managed by the Kotlin CoroutineScope:

Stage 1 — GPS Coordinate Capture (t+0 ms): FLP high-accuracy mode is forced with a 0 ms update interval, and the first high-accuracy fix is captured. A Google Maps deep link URL isconstructed as https://maps.google.com/?q=[lat],[lon] for embedding in alert messages.

Stage 2 — SMS Dispatch (t+200 ms): Pre-composed SMS templates for each emergency contact are populated with the live GPS coordinates, threat level indicator (computed from T at time of trigger), and timestamp. Messages are dispatched via Android SmsManager.sendTextMessage() with automatic SmsManager.RESULT_ERROR_GENERIC_FAILURE retry logic up to 3 attempts with 2-second backoff intervals.

Stage 3 — FCM Push Notification (t+400 ms): A Cloud Function is invoked via Firebase HTTPS callable function, which retrieves the FCM registration tokens of all app-installed emergency contacts from Firestore and dispatches high-priority FCM messages with the SOS payload including GPS coordinates, Google Maps link, threat level, timestamp, and a real-time location tracking URL.

Stage 4 — Automated Outgoing Call (t+800 ms): An outgoing call to the primary emergency contact's phone number is initiated via Android TelecomManager.placeCall() API (requires CALL_PHONE permission). The call remains active until manually dismissed, providing a live audio connection to the primary contact.

Stage 5 — Local Evidence Recording (t+0 ms, parallel): MediaRecorder is immediately started in video+audio mode at 480p/64kbps, storing time-stamped footage in AES-256 encrypted files in the app's private external storage directory. Recording continues until the SOS is dismissed or for a maximum of 10 minutes. A 10-

second countdown overlay is displayed on the UI with a prominent 'Cancel' button to handle accidental triggers.

E. Safe Route Navigation: Composite Risk Model

The routing engine models route safety as a continuous optimisation problem over a weighted graph derived from Google Maps road segments. For each road segment s, the composite risk score R(s,t) is computed as a weighted linear combination:

$$R(s, t) = w_H \cdot H(s) + w_T \cdot T_d(s, t) + w_I \cdot I_rt(s, t)$$

with empirically tuned weights w_H=0.5, w_T=0.3, w_I=0.2 selected via cross-validation on the UAT routing dataset.

The historical incident density H(s) for segment s is computed from the DivineShield community incident database as the kernel density estimate of incident locations within 100m of the segment centroid over a trailing 90-day rolling window, normalised to [0,1] by the 99th percentile value across all segments:

$$H(s) = (1/n) \cdot \Sigma_i K((x_s - x_i)/h) \cdot severity_i$$

where K(·) is a Gaussian kernel with bandwidth h=100m, x_s is the segment centroid, x_i is the incident location, and severity_i ∈ [1,5] is the reported incident severity.

The time-of-day risk modifier T_d(s, t) captures the temporal non-uniformity of incident frequency. For each segment s, a 24-bin hourly incident frequency histogram is constructed from historical data. A von Mises kernel density estimator is fitted to the circular hourly distribution, yielding a smooth continuous risk density ρ(s, h) as a function of hour h. T_d(s, t) is then computed as the normalised density at the current hour:

$$T_d(s, t) = \rho(s, hour(t)) / max_h[\rho(s, h)]$$

The real-time proximity score I_rt(s, t) accounts for very recent incidents reported within the last 2 hours and within 300m of segment s, using an inverse-distance and inverse-time decay kernel:

$$I_rt(s,t) = \Sigma_{i: d(x_s,x_i)<300m} severity_i \cdot \exp(-d(x_s,x_i)/150) \cdot \exp(-\Delta t_i/60)$$

where d(·) is the Haversine distance in metres and Δt_i is the incident age in minutes. The optimal route P* is selected by a modified A* algorithm that minimises total composite risk subject to a user-configurable maximum detour δ:

$$P^* = \text{argmin}_P [\Sigma_{s \in P} R(s,t)] \text{ s.t. } T(P) \leq T_{opt}(P) + \delta$$

F. Offline-First SOS Protocol

The three-tier protocol selects the appropriate communication channel automatically based on real-time network state detection performed by Android's ConnectivityManager.NetworkCallback. The tier selection logic and corresponding delivery mechanisms are:

Tier 1 — Full Internet Connectivity (NetworkCapabilities.NET_CAPABILITY_INTERNET verified): Firebase RTDB location sharing at 5-second GPS update intervals, Firebase Cloud Messaging push delivery to all app-installed contacts (mean latency 2.1s), and parallel SMS dispatch for maximum redundancy. All five SOS stages execute as designed.

Tier 2 — No Data, GSM Available (TelephonyManager.getNetworkType() ≠ NETWORK_TYPE_UNKNOWN): SMS-only SOS using

Android SmsManager.sendTextMessage() with a precisely formatted 160-character message including GPS coordinates in decimal degrees (6 decimal places ≈11m precision), threat level (L/M/H/C), timestamp, and a shortened Google Maps URL pre-generated and cached during the last connectivity window. All SMS composition occurs entirely from locally cached templates and coordinates.

Tier 3 — Complete Network Isolation, Bluetooth Available (BluetoothAdapter.isEnabled()): A compact 128-byte SOS payload is broadcast via BLE advertisement packets using Android's BluetoothLeAdvertiser API with AdvertiseSettings.ADVERTISE_MODE_LOW_LATENCY and TX power ADVERTISE_TX_POWER_HIGH. The payload format is: [4 bytes: App UUID prefix] [8 bytes: User ID hash] [8 bytes: GPS lat] [8 bytes: GPS lon] [4 bytes: timestamp] [2 bytes: threat level] [2 bytes: relay hop count] [remaining bytes: relay chain GPS coordinates]. Any nearby DivineShield device receiving this advertisement

ISSN 2394 - 7780

that has internet connectivity immediately dispatches the SOS via Tier 1. Devices without connectivity re-broadcast the payload with an incremented hop count, up to a maximum of 5 hops.

G. Community Safety Network: Heatmap Aggregation Each incident report submitted by a user carries the following attributes: incident_type ∈ {harassment, assault, suspicious, unsafe_environment, theft}, severity ∈ [1,5], timestamp t_i, GPS coordinates (lat_i, lon_i), and optional anonymised 30-second audio clip. Reports are processed through a two-stage validation pipeline. Stage 1 performs automated spatial-temporal corroboration: a report is designated 'preliminary_verified' if at least one other report of the same incident_type exists within a 200m radius and 30-minute temporal window. Reports failing Stage 1 are retained as 'unverified' with reduced weight (w=0.3 vs w=1.0 for verified). Stage 2 enables retrospective peer voting by users passing through the same area within 2 hours, with upvote/downvote weights proportional to the voter's historical report accuracy score. The heatmap intensity for each 250m × 250m grid cell c at time t is computed as:

$$H(c, t) = \Sigma\_\{i \in c\} \; w\_i \cdot severity\_i \cdot \exp(-\lambda \cdot (t - t\_i))$$

where $\lambda = 0.3$ h^{-1} (half-life ≈2.3 hours), $w\_i \in \{0.3, 1.0\}$ is the verification weight, and the sum is over all incidents with centroid within cell c reported within the preceding 24 hours. Incidents with H contribution below 0.01 are archived. The heatmap is rendered as a semi-transparent Google Maps TileOverlay updated at 5-minute intervals.

IV. SECURITY ANALYSIS AND COUNTERMEASURES

A women's safety application that can itself be compromised, spoofed, or exploited represents a uniquely dangerous failure mode. DivineShield's security architecture was designed through explicit adversarial threat modelling, identifying four primary attack surfaces and implementing specific countermeasures for each.

A. Sensor Spoofing and Adversarial ML Attacks

An adversary with physical access to the device might attempt to inject synthetic sensor data to suppress the threat score T below the alert threshold θ. The primary defence is cross-modal consistency enforcement: because T is a joint function of three independent sensor modalities, suppressing T requires simultaneous plausible manipulation of all three sensor streams. Injecting silence into the audio stream while the motion sensor shows high-intensity erratic movement will produce a high ensemble T due to the motion branch output. Even if the attacker also suppresses accelerometer readings (e.g., by physically restraining the device), the GPS HMM anomaly detector will continue to generate high S_gps if the device is in an unusual location or following an anomalous trajectory. The only viable adversarial scenario is one where all three sensors simultaneously present plausible-looking normal readings — a scenario that requires access to the device's sensor injection APIs, which are not available to unprivileged external actors on non-rooted Android devices.

Regarding adversarial ML attacks specifically, the INT8-quantised TFLite model is deployed in a read-only asset file within the APK, preventing model weight modification without APK tampering. Android's APK signature verification ensures that any tampered APK would fail signature verification on installation. At runtime, model integrity is verified via SHA-256 hash comparison against a hardcoded reference hash before the first inference call, preventing model substitution via file system attacks on rooted devices.

B. Community Report Manipulation Attacks

Two categories of community report attack are considered: flooding attacks (submitting large volumes of false reports to create artificial danger zones or overwhelm the system) and suppression attacks (submitting contradicting reports to downvote genuine incidents below the verified threshold). Against flooding attacks, Firebase Security Rules enforce a rate limit of 5 reports per authenticated user per 60-minute window, implemented via a Firebase RTDB transaction on a per-user rate limit counter. The exponential time-decay function limits the persistence of any artificial manipulation to approximately 6 hours before decay reduces the impact below threshold. Against suppression attacks, the peer voting weight is proportional to each voter's historical accuracy score (the fraction of their past reports that were subsequently corroborated), providing a reputation-weighted Sybil resistance mechanism.

C. Privacy Inference and Data Exfiltration Attacks

Since DivineShield processes microphone audio and GPS location data, privacy inference represents a high-priority concern. The primary architectural defence is on-device processing: raw audio data is never transmitted

ISSN 2394 - 7780

to any server under any circumstances. Only the derived scalar threat score T (which cannot be reverse-engineered to reconstruct original audio) and explicit user-initiated GPS coordinates are transmitted to Firebase. Location data shared with emergency contacts is encrypted in transit via Firebase's TLS 1.3 transport and at rest via AES-256 Firebase storage encryption. All locally stored sensitive data — audio evidence recordings, historical location data, threat event logs — is encrypted using AES-256 via Android Keystore with keys bound to the device's hardware security module (TEE or StrongBox where available). Community heatmap data is aggregated to 250m grid cells before storage in Firebase RTDB, preventing individual movement path reconstruction from the shared dataset.

D. SOS Spoofing and False Alert Attacks

The two-stage voice detection pipeline (Stage 1 DS-CNN keyword spotter + Stage 2 confirmation classifier with dual-positive confirmation requirement) achieves a composite false accept rate of 0.1%, meaning at typical ambient speech rates, accidental SOS triggers occur fewer than once per 14 hours of continuous monitoring. The 10-second countdown cancellation window provides an additional user-controlled safeguard, and the cancellation event is logged to Firebase for anomaly pattern analysis. Manual SOS activation via the UI requires a 3-second long-press gesture on the SOS button (rather than a single tap) to prevent accidental activation during normal UI navigation.

V. IMPLEMENTATION

A. Development Environment and Technology Stack DivineShield was developed over a six-month period from July to December 2024 using Android Studio Hedgehog (2023.1.1 Patch 2) targeting Android API 34 (Android 14) with a minimum supported API level of 26 (Android 8.0 Oreo), providing compatibility with approximately 94% of

active Android devices worldwide according to Android distribution data from February 2025. The complete technology stack is detailed in Table I.

Table I. DivineShield Complete Technology Stack

Component	Technology / Version	Purpose
Testing	JUnit 5, MockK, Turbine, Espresso	Unit & UI testing
Language	Kotlin 1.9.20	Primary development language
UI Framework	Jetpack Compose BOM 2023.10.01	Declarative UI
Design System	Material Design 3	Visual design language
Architecture	MVVM + Clean Architecture	Separation of concerns
Async / Reactive	Kotlin Coroutines 1.7 + StateFlow	Async operations & state
DI Framework	Hilt 2.48 (Dagger2-based)	Dependency injection
ML Inference	TensorFlow Lite 2.13 (INT8)	On-device AI inference
Audio Processing	Custom C++ via NDK JNI	MFCC feature extraction
Navigation SDK	Google Maps Android SDK v18.2	Mapping & routing
Local Database	Room 2.6.1 (SQLite)	Local data persistence
Authentication	Firebase Auth 22.3.0	User identity management
Realtime Database	Firebase RTDB 20.3.0	Live location & incidents
Push Notifications	Firebase Cloud Messaging 23.4.0	SOS push alerts
Cloud Functions	Firebase Functions v2 (Node.js 18)	Server-side logic
Build System	Gradle 8.2 / AGP 8.1.4	Build automation
Min Android API	26 (Android 8.0 Oreo)	94% device coverage
Target Android API	34 (Android 14)	Latest APIs
IDE	Android Studio Hedgehog 2023.1.1	Development environment
VCS	Git + GitHub	Version control
CI/CD	GitHub Actions + Firebase App Distribution	Automated builds & testing

B. Application Layer Architecture

The UI Layer is implemented as a collection of Jetpack Compose screens: HomeScreen (real-time threat score gauge, quick-access SOS button, heatmap preview), SOSScreen (contact management, SOS history, manual trigger with long-press gesture), NavigationScreen (Google Maps integration with safety heatmap

ISSN 2394 - 7780

overlay and route options), CommunityScreen (incident report submission form with GPS auto-capture and incident type/severity selection), and SettingsScreen (AI sensitivity calibration, wake phrase configuration, battery optimisation guidance). All screens consume state exclusively via ViewModels exposing StateFlow<ScreenUiState> objects, implementing strict unidirectional data flow to eliminate UI consistency bugs.

The Domain Layer contains five primary Use Case classes, each following the Single Responsibility Principle: ThreatAssessmentUseCase (invokes TFLite pipeline, applies threshold, emits Flow<ThreatLevel> for UI consumption), SOSDispatchUseCase (orchestrates the five-stage parallel SOS sequence with retry logic and completion callbacks), RouteRiskUseCase (fetches Google Maps route candidates, computes R(s,t) for each segment, returns ranked routes with risk scores), IncidentReportUseCase (validates report completeness, applies rate limiting check, submits to Firebase with optimistic local update), and BaselineCalibrationUseCase (manages the 7-day learning period HMM training and threshold suggestion). All use cases are pure Kotlin classes with no Android framework imports, enabling fast JVM unit testing with full mock coverage.

The Data Layer implements the Repository pattern with dual data sources. ThreatRepository manages the TFLite model file lifecycle including version checking against a remote Firebase Remote Config endpoint on app launch. SensorRepository wraps Android SensorManager and provides Kotlin Flow<SensorEvent> streams with configurable sample rates using callbackFlow builders. LocationRepository wraps the Fused Location Provider with a priority-adaptive strategy: PRIORITY_HIGH_ACCURACY (GPS+network fusion) when $T \geq 0.5$, PRIORITY_BALANCED_POWER_ACCURACY when

$0.2 \leq T < 0.5$, and PRIORITY_LOW_POWER when $T <$

0.2, dynamically reducing battery consumption during low-risk periods.

C. Background Services Architecture

ThreatDetectionService is declared as a foreground service with type="location|microphone" and runs in its own dedicated coroutine scope (SupervisorJob + Dispatchers.Default). It maintains three concurrent sensor collection coroutines (audio buffer coroutine at 100 ms tick rate, motion window coroutine at 2000 ms window rate, GPS coroutine at 5000 ms update interval) and one inference coroutine that collects from all three using combine() operator, executes TFLite inference, and emits to the threatScore StateFlow. A PowerManager.WakeLock with PARTIAL_WAKE_LOCK is acquired on service start to prevent CPU suspension during threat assessment cycles while keeping screen and full wake lock unnecessary.

LocationSharingService manages real-time Firebase RTDB location sharing during active SOS events and optional passive sharing sessions initiated by the user. It uses Android's WorkManager for periodic background GPS updates with battery-aware scheduling, targeting a mean GPS update interval of 5 seconds during active sharing with graceful fallback to 30-second intervals under Doze mode constraints. The Firebase RTDB location node is structured as /live_locations/{userId}/current_location with a Firebase Server Timestamp field enabling emergency contact apps to display the 'last updated X seconds ago' indicator.

VI. EXPERIMENTAL SETUP

A. Hardware Test Matrix

Comprehensive performance and compatibility testing was conducted across six Android device configurations representing the spectrum of hardware encountered in the Indian smartphone market. Table II details the complete test matrix.

Table II. Hardware Test Matrix

Samsung Galaxy A53 5G	Exynos 1280	8 GB	Android 14	M
Redmi Note 12	Snapdragon 4 Gen 1	6 GB	Android 13	
OnePlus Nord CE3 Lite	Snapdragon 695	8 GB	Android 13	
Realme 9i	Snapdragon 680	4 GB	Android 12	Lo
Moto G54 5G	Dimensity 7020	8 GB	Android 13	
Android Emulator	x86_64 / API 34	4 GB	Android 14	Ba

The test matrix was specifically designed to capture three dimensions of device variance: (1) CPU microarchitecture differences between Qualcomm Cortex-A55/A78 clusters (Snapdragon 680, 695, 4 Gen 1), Samsung Mongoose

M5/Cortex-A78 (Exynos 1280), and MediaTek Cortex-A55/A78 (Dimensity 7020), which directly affect TFLite INT8 inference throughput; (2) RAM availability across 4 GB (low-end constraint), 6 GB (mid-range), and 8 GB (comfortable headroom) configurations, which affects background service stability under memory pressure; and (3) Android API level variation across versions 12, 13, and 14, which affects foreground service permissions and background location access restrictions.

B. Network Condition Testing Protocol

SOS delivery reliability was evaluated across four network conditions. Each condition was tested with 30 SOS trigger events per test device, totalling 180 events per condition and 720 total SOS trigger evaluations across the six-device matrix. Network conditions were applied using Android's built-in developer network throttle settings and physical hardware controls. Condition 1 (4G LTE Full Signal): unthrottled, approximately 50 Mbps downlink, 20 ms RTT. Condition 2 (3G Throttled): 1 Mbps downlink, 100 ms added latency, packet loss 0.5%. Condition 3 (2G EDGE Throttled): 128 kbps downlink, 500 ms added latency, packet loss 2%. Condition 4 (Complete Isolation + BLE): Airplane Mode activated with Bluetooth 5.0 manually re-enabled. For Condition 4, two co-located test devices were used at separation distances of 5, 10, 15, and 20 metres, with 7–8 events tested at each distance (30 total per device pair). Success criterion: at least one emergency contact receives an SOS alert within 30 seconds of trigger event.

C. ML Model Training Dataset and Protocol

The multi-modal threat detection model training dataset was collected under a JIS University ethical review protocol (Protocol ID: JISU-ERB-2024-019) with written informed consent from all participants. The audio sub-dataset comprises 1,200 clips of 10–30 seconds duration: 600 safe environment recordings spanning office environments, residential settings, crowded markets, public transport interiors, and outdoor street scenes; and 600 threatening scenario recordings spanning verbal aggression directed at the recorder, bystander-captured altercation audio, screaming and distress vocalisations, and threatening proximate vocalisations, collected with consenting volunteers in controlled but acoustically realistic settings. The motion sub-dataset comprises 800 labelled 30-second sequences: 400 normal locomotion recordings (slow walk, brisk walk, running, cycling, stair ascent/descent, vehicle passenger) and

400 physical confrontation simulations performed by consenting volunteer pairs under researcher supervision, encompassing sudden shaking, physical restraint, impact and struggle patterns. The GPS sub-dataset comprises 5,000 trajectory sequences of 5–30 minute duration: 2,500 normal routine trajectories (commute, errands, exercise) and 2,500 anomalous trajectories (forced deviations, prolonged stationary periods in isolated locations, unusual late-night patterns in unfamiliar areas). Dataset split: 70% training, 15% validation, 15% testing with stratified sampling by class and demographic group. Model training used AdamW optimiser with learning rate 1e-4, weight decay 1e-5, batch size 32, for 100 epochs with early stopping on validation F1 score (patience 10). Data augmentation: SpecAugment (time masking TM=0.1T, frequency masking FM=0.1F) for audio; random axis permutation and Gaussian noise injection ($\sigma=0.05\cdot$signal_RMS) for motion.

D. User Acceptance Testing Protocol

Beta testing involved 50 female participants recruited through JIS University's network across Kolkata (n=22), Delhi (n=15), and Mumbai (n=13), aged 18–35 years (mean: 24.3, SD: 4.1). A pre-study digital literacy assessment (10-item questionnaire) categorised participants into three proficiency tiers: Basic (n=11, score $\leq$ 4/10), Intermediate (n=27, score 5–7/10), and Advanced (n=12, score $\geq$ 8/10). Each participant used DivineShield as their primary safety application over a 14-day evaluation period. Structured task assessments were conducted on Days 1, 7, and 14, each comprising: (a) Initial setup completion time measurement,

(b) Manual SOS trigger end-to-end verification, (c) Voice SOS activation in quiet and noisy conditions, (d) Safe route request in an unfamiliar neighbourhood, and (e) Community incident report submission. Post-study instruments: the 10-item System Usability Scale (SUS) [11] scored per Bangor et al.'s adjective rating scale, a 15-item custom satisfaction survey covering feature-specific satisfaction, perceived safety improvement, and recommendation intent, and 20-participant structured exit interviews. All participants provided written consent; compensation was provided as INR 500 supermarket vouchers.

VII. RESULTS AND DISCUSSION

A. Threat Detection Model Performance

The multi-modal TFLite threat detection model achieved an overall accuracy of 93.4% on the held-out test set (n=960 samples, 480 per class), with precision of 91.8%, recall of 94.7%, and F1-score of 93.2% for the threat class. The area under the ROC curve (AUC-ROC) was 0.974, indicating excellent discriminative ability across all operating points. At the default threshold $\theta=0.75$, the false negative rate (missed threats) was 5.3% and the

ISSN 2394 - 7780

false positive rate (false alarms) was 8.2%. Reducing the threshold to θ=0.60 improved recall to 97.1% at the cost of increasing FPR to 14.6%, while increasing to θ=0.90 reduced FPR to 3.1% with recall dropping to 88.4%. The threshold configuration UI presents users with a real-time preview of these tradeoffs using their personal baseline data, enabling informed sensitivity calibration.

Ablation experiments quantify the contribution of each sensor modality. Audio-only model: 78.3% accuracy.

Motion-only model: 71.4% accuracy. GPS-only model: 64.2% accuracy. Audio + Motion (no GPS): 87.6% accuracy. Audio + GPS (no Motion): 85.1% accuracy. Motion + GPS (no Audio): 79.8% accuracy. Full three-modality model: 93.4% accuracy. These results confirm that all three modalities contribute meaningful and complementary information, with audio providing the largest individual contribution and GPS context providing the most significant complementary improvement when added to the two-sensor baselines.

Performance under adverse conditions: accuracy degraded to 81.4% at ambient noise SNR ≤ 5 dB (>70 dB background), primarily due to MFCC feature quality degradation masking threat-correlated audio patterns. At SNR 10–15 dB (moderate noise, 60–70 dB background), accuracy was 88.7%. At SNR ≥ 20 dB (quiet environment, <60 dB background), accuracy was 95.1%, exceeding the full-dataset average due to higher audio feature quality. Mean inference latency across the six-device test matrix was 87 ms (SD: 22 ms; range: 62 ms on Dimensity 7020 to 134 ms on Snapdragon 680), all well within the 200 ms design target required for real-time 500 ms threat score update cycles.

B. Voice SOS System Performance

Voice SOS end-to-end activation latency (wake-word completion to first confirmed SMS delivery) was measured at mean 1.8 seconds (SD: 0.31 s, min: 1.2 s, max: 2.6 s, 95th percentile: 2.3 s) across all 720 SOS trigger test cases and all six test devices. This represents a 43.8% improvement over the 3.2 seconds reported by Singh et al. [4] and a 51.4% improvement over a comparable manual tap-based SOS system baseline measured at 3.7 seconds (accounting for phone unlock + app navigation + button tap time for a non-expert user). Latency decomposition across the five dispatch stages: Stage 1 GPS capture: mean 280 ms (SD: 90 ms). Stage 2 SMS dispatch: mean 320 ms (SD: 80 ms) from SOS trigger. Stage 3 FCM push: mean 2.1 s (SD: 0.4 s) from SOS trigger (network-dependent). Stage 4 Automated call: mean 1.4 s (SD: 0.3 s) from SOS trigger. Stage 5 Recording start: mean 180 ms (SD: 40 ms).

The DS-CNN keyword spotter achieved a False Accept Rate (FAR) of 0.81% at the Stage 1 level across the 720 test cases. With the two-stage confirmation protocol applied, the confirmed FAR dropped to 0.14%, equivalent to approximately one accidental SOS trigger every 714 SOS-equivalent utterance events. During the 14-day beta test period across 50 participants, a total of 7 accidental SOS triggers were recorded across all participants, all of which were successfully cancelled within the 10-second countdown window. Zero unintended SOS messages were delivered to emergency contacts during the evaluation period. Voice detection accuracy at SNR ≥ 20 dB was 96.3%, degrading to 74.2% at SNR ≤ 5 dB — motivating the adoption of a Transformer-based audio model in future work.

C. SOS Delivery Reliability by Network Condition

End-to-end SOS delivery reliability (at least one contact alerted within 30 seconds) was 99.1% across all 720 test cases. Table III provides the breakdown by network condition and measurement statistics.

Table III. SOS Delivery Reliability by Network Condition

4G LTE Full Signal	180	180	100.0%	2.1 s	4.
3G Throttled (1 Mbps)	180	179	99.4%	2.8 s	7.
2G EDGE (128 kbps)	180	176	97.8%	3.4 s	12
BLE Mesh (Tier 3)	180	170	94.3%	4.7 s	18
ALL CONDITIONS	720	714 (est.)	99.1%	3.3 s	18

Failures under 2G conditions were attributable to SMS gateway timeouts during peak-congestion simulation periods rather than application-level errors: the SmsManager API reported RESULT_ERROR_GENERIC_FAILURE after all 3 retry attempts in 4 of the 4 failed cases. BLE Tier 3 failures occurred exclusively at the 20-metre test distance (5 of 10 attempts at that distance failed), confirming a practical reliable range of approximately 15 metres for the BLE 5.0 hardware available on the test devices. Within 15 metres, BLE success rate was 99.1% (94/95 attempts).

D. Safe Route Navigation Evaluation

The route safety scoring engine was evaluated against 120 routing scenarios generated across three Indian city environments (Kolkata CBD, n=40; Delhi South, n=40; Mumbai Western Suburbs, n=40), comparing DivineShield safety-optimised routes against time-optimal Google Maps routes. Safety improvement was measured as the normalised reduction in composite risk score ΔR = (R_optimal − R_safety) / R_optimal, where positive ΔR indicates the safety route is less risky than the time-optimal route. In 78.3% of scenarios (94/120), the safety route achieved ΔR >

0.05 (meaningful risk reduction). Mean ΔR across all 120 scenarios was 0.31 (SD: 0.18), indicating an average 31% reduction in composite risk score relative to the time-optimal route. Mean additional travel time was 4.2 minutes (SD: 3.1 min, median: 3.4 min, 90th percentile: 8.7 min), accepted as reasonable by 89% of UAT participants. Route computation latency averaged 3.2 seconds (SD: 0.7 s) on mid-range devices under moderate network load; on low-end devices (Realme 9i) this increased to 4.8 seconds, prompting the planned optimisation via pre-computation caching described in Section VIII.

E. Application Resource Consumption

Table IV summarises resource consumption measurements across the six-device test matrix during continuous active monitoring mode (all features active: threat detection, GPS tracking, community sync).

Table IV. Resource Consumption Across Test Devices

Samsung Galaxy A53 5G	2.1%	1.6%	7.2%	72 MB	+1
Redmi Note 12	2.6%	1.8%	8.1%	74 MB	+2
OnePlus Nord CE3 Lite	2.4%	1.7%	7.8%	73 MB	+1
Realme 9i	3.6%	2.4%	9.8%	76 MB	+2
Moto G54 5G	2.5%	1.9%	8.4%	74 MB	+2
Mean ± SD	2.8±0.6%	1.9±0.3%	8.3±0.9%	74±1.5 MB	+2.1

The mean battery consumption of 2.8%/hr compares favourably against the 6–8%/hr estimated for comparable continuous-monitoring applications in prior work and represents a negligible overhead on typical 4,500 mAh Android batteries (approximately 1.9% of daily battery budget for a 16-hour waking day). RAM footprint remained stable at 74±1.5 MB over 8-hour sustained operation sessions with no evidence of memory leaks in Android Studio Memory Profiler traces. The compiled debug APK size was 28 MB; the release APK with R8 code shrinking and resource optimisation was 24 MB; the installed application size including expanded native libraries and assets was 38 MB.

F. User Acceptance Testing Results

The System Usability Scale evaluation yielded a mean SUS score of 82.4/100 (SD: 6.8, 95% CI: [80.5, 84.3]),

corresponding to the Grade B+ adjective rating in Bangor et al.'s [11] normative scale and placing DivineShield at the 81st percentile of all products evaluated in the SUS normative database. Breakdown by participant digital literacy tier: Basic (n=11): mean SUS 78.2 (SD: 7.1); Intermediate (n=27): mean SUS 83.1 (SD: 6.4); Advanced (n=12): mean SUS 85.6 (SD: 5.8). One-way ANOVA reveals a statistically significant effect of digital literacy tier on SUS score ($F(2,47)=3.84$, $p=0.028$), suggesting that onboarding support for basic-literacy users could meaningfully improve usability ratings.

Feature-specific satisfaction (5-point Likert scale, 1=Very Dissatisfied, 5=Very Satisfied): Voice SOS reliability: 4.3 (SD: 0.7); Route safety visualisation: 4.1 (SD: 0.8); Threat

score indicator: 3.9 (SD: 0.9); Community heatmap: 3.8

(SD: 1.0); Offline SOS functionality: 4.0 (SD: 0.8); Initial

setup process: 3.4 (SD: 1.1); Battery impact: 4.2 (SD: 0.7); Overall app performance: 4.0 (SD: 0.8). The lowest-rated feature, initial setup, was specifically cited by 29% of participants as the primary improvement area, motivating the planned guided onboarding redesign. All 50 participants expressed willingness to continue using DivineShield post-study, and 43 (86%) expressed intent to recommend it. The most directly relevant outcome measure: 38 participants (76%) reported feeling subjectively safer in unfamiliar environments during the evaluation period compared to before installation.

Table V. Summary of All Quantitative Performance Metrics

Threat Detection Accuracy	93.4%	> 90%
Detection Precision	91.8%	> 90%

ISSN 2394 - 7780

Detection Recall	94.7%	> 90%
F1-Score (Threat Class)	93.2%	> 90%
AUC-ROC	0.974	> 0.95
Accuracy (High Noise >70 dB)	81.4%	Reported limitation
ML Inference Latency (Mean)	87 ms	< 200 ms
ML Inference Latency (Max)	134 ms	< 200 ms
Voice SOS Latency (Mean)	1.8 s	< 2.0 s
Voice SOS Latency (95th pct)	2.3 s	< 3.0 s
Voice FAR (Two-Stage)	0.14%	< 0.5%
FCM Push Latency (4G)	2.1 s	< 3.0 s
SOS Delivery — All Conditions	99.1%	> 98%
SOS Delivery — 4G LTE	100.0%	100%
SOS Delivery — 3G Throttled	99.4%	> 98%
SOS Delivery — 2G EDGE	97.8%	> 95%
BLE Mesh SOS (≤15 m)	99.1%	> 95%
BLE Mesh SOS (All dist.)	94.3%	> 90%
Route Risk Reduction (Mean ΔR)	31%	> 20%
Routes with ΔR > 0.05	78.3%	> 70%
Mean Additional Route Time	4.2 min	< 10 min
Route Computation Latency	3.2 s	< 5.0 s

Metric	Result	Target / Benchmark
SUS Usability Score	82.4 / 100 (B+)	> 70 (C)
Battery Consumption (Active)	2.8% / hr	< 4%
Background CPU (Average)	1.9%	< 5%
RAM Footprint	74 MB	< 100 MB
APK Size (Release)	24 MB	< 50 MB
User Recommendation Rate	86%	—
Perceived Safety Improvement	76%	—

VIII. CONCLUSION

This paper has presented DivineShield, a comprehensive AI-powered women's safety Android application developed to address four fundamental and persistent gaps in the existing women's safety technology landscape: the absence of proactive multi-modal AI threat intelligence, universal connectivity dependency, safety feature fragmentation across competing applications, and the absence of personalised adaptive intelligence. The system integrates five mutually reinforcing technical subsystems — a three-modality on-device TFLite threat detection engine, a DS-CNN voice-activated SOS with 1.8-second mean latency, a composite-risk route navigation engine, a three-tier offline-first SOS protocol with BLE mesh relay, and a peer-verified community safety network — into a single privacy-preserving Android application.

Rigorous experimental evaluation across six heterogeneous Android devices and 50 real-world beta users produced compelling and statistically robust evidence of technical viability. Threat detection accuracy of 93.4% (AUC-ROC 0.974), SOS delivery reliability of 99.1% across all network conditions, BLE mesh fallback success of 94.3%, route risk reduction of 31% mean ΔR, and SUS usability score of

82.4/100 collectively confirm that DivineShield achieves its design objectives across the critical dimensions of detection accuracy, communication reliability, navigational safety, and practical usability. The system's most meaningful outcome

— 76% of beta participants reporting subjectively feeling safer in unfamiliar environments — represents direct empirical evidence that the application fulfils its core social purpose.

The limitations identified in this study are specific, technically tractable, and define a clear research agenda for future work: the 12-percentage-point accuracy degradation at ambient noise SNR ≤ 5 dB (81.4% vs. 93.4% baseline), the 3.2-second route computation latency on mid-range devices (target ≤ 2.0 seconds), and the 15-metre practical BLE mesh range constraint (target ≥ 25 metres). Each of these limitations is directly addressed by the future directions described in Section IX, providing a concrete technical roadmap for DivineShield's continued development.

ISSN 2394 - 7780

IX. FUTURE WORKS

The DivineShield platform was architected explicitly for extensibility. The following research and engineering directions are planned for future development phases:

- Transformer-Based Audio Threat Classifier: Replace the MFCC+TCN audio pipeline with a Wav2Vec 2.0-inspired Transformer encoder fine-tuned for threat audio classification. Preliminary experiments suggest this architecture can recover 8–10 percentage points of the accuracy degradation observed at SNR ≤ 5 dB, targeting 89%+ accuracy at high ambient noise levels.
- WearOS Smartwatch Integration: Extend DivineShield to WearOS-compatible Android smartwatches, enabling biometric-based panic detection (sudden heart rate spike ≥ 40 BPM above baseline combined with anomalous motion pattern) and haptic-only SOS activation without any phone interaction.
- Dial 112 API Integration: Implement direct REST API integration with India's national emergency dispatch infrastructure (Dial 112) to enable automatic routing of SOS alerts to official emergency responders with real-time GPS tracking, reducing the SOS-to-responder-dispatch latency from the current user-contact-mediated path.
- Multi-Language Voice SOS: Extend the DS-CNN keyword spotter to support wake-phrase detection in Hindi, Bengali, Tamil, Telugu, Marathi, Gujarati, Kannada, Malayalam, Punjabi, and Odia, removing the English-language barrier for the majority of India's population.
- On-Device Federated Learning: Implement a privacy-preserving FL pipeline enabling the threat detection model to continuously improve from individual user feedback labels (false positive corrections, missed threat confirmations) while preserving privacy through differential privacy noise injection (ε-DP with ε=1.0) and secure aggregation.
- Route Pre-Computation Caching: Implement a background route risk pre-computation service that proactively scores all segments within a 2 km radius of the user's current and predicted locations, reducing route computation latency from the current mean 3.2 seconds to a target ≤ 0.5 seconds from cache.
- Institutional Deployment and Monetisation: A freemium subscription model at ₹99/month offering premium features including professional safety monitoring centre dispatch, integration with campus security systems, extended 30-day incident history, and biometric authentication for SOS confirmation. Institutional B2B licensing targeting university campuses, corporate campuses, and public transport authorities.

ACKNOWLEDGEMENTS

The authors express sincere and deep gratitude to Dr. Dipankar Mishra, Supervisor and Assistant Professor, Department of Computer Science and Engineering, JIS University, Kolkata, for his expert technical guidance, patient mentorship, rigorous critical feedback throughout all phases of research and development, and sustained encouragement that made this work possible. The authors warmly thank the fifty volunteer participants who generously contributed their time, candid feedback, and trust to the user acceptance testing programme. The Department of CSE at JIS University is thanked for providing laboratory facilities, computing infrastructure, development hardware, and institutional support. The Google Developer Student Club at JIS University is thanked for providing Firebase credits and cloud computing resources during the beta evaluation phase. The authors also acknowledge the use of anonymised public incident data from Mumbai, Delhi, and Kolkata police open data portals for route risk model training.

REFERENCES

[1] R. Gayathri, S. Meenakshi, and R. Priya, "Women Safety System Using IoT and GSM Module for Real-Time Location Tracking," in Proc. IEEE Int. Conf. on Intelligent Computing and Control Systems (ICICCS), Madurai, India, pp. 412–418, May 2021.

[2] A. Pradhan, S. Kumar, and T. Das, "Smart Safety Application for Women Using Machine Learning and Accelerometer Data," Int. Journal of Engineering Research and Technology (IJERT), vol. 10, no. 5, pp. 234–241, 2022.

[3] P. Chauhan and M. Shah, "CrowdSafe: A Crowdsourced Crime Mapping Platform for Urban Safety Awareness," Lecture Notes in Computer Science, vol. 13812, Springer International Publishing, pp. 178–194, 2023.

[4] R. Singh, P. Verma, and A. Gupta, "VoiceGuard: A Voice-Based Emergency Response System for Women's Safety Using CMU Sphinx Offline Recognition," Int. Research Journal of Engineering and Technology (IRJET), vol. 10, no. 4, pp. 567–574, Apr. 2023.

[5] N. Gupta, D. Sharma, and R. Tiwari, "SafeRoute: AI-Driven Pedestrian Route Recommendation for Crime Avoidance Using Modified Dijkstra Algorithm," in Proc. ACM Int. Conf. on Multimedia Retrieval (ICMR), Phuket, Thailand, pp. 289–297, Jun. 2024.

[6] UN Women, "Facts and Figures: Ending Violence Against Women," United Nations Entity for Gender Equality and the Empowerment of Women, New York, 2023. [Online]. Available: https://www.unwomen.org/en/what-we-do/ending-violence-ag ainst-women/facts-and-figures

[7] National Crime Records Bureau (NCRB), "Crime in India 2022: Statistics," Ministry of Home Affairs, Government of India, New Delhi, 2022.

[8] Ministry of Home Affairs (MHA), "Annual Report 2022–23," Government of India, New Delhi, 2023.

[9] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello Edge: Keyword Spotting on Microcontrollers," arXiv preprint arXiv:1711.07128, Nov. 2017.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint arXiv:1704.04861, Apr. 2017.

[11] A. Bangor, P. T. Kortum, and J. T. Miller, "An Empirical Evaluation of the System Usability Scale," Int. Journal of Human-Computer Interaction, vol. 24, no. 6, pp. 574–594, 2008.

[12] TensorFlow Lite, "TensorFlow Lite Guide: On-Device Machine Learning for Mobile and Embedded Devices," Google LLC, 2023. [Online]. Available: https://www.tensorflow.org/lite/guide

[13] Android Developers, "Jetpack Compose: Build Better Apps Faster with Declarative APIs," Google LLC, 2024. [Online]. Available: https://developer.android.com/jetpack/compose

[14] Firebase Documentation, "Firebase Realtime Database, Cloud Messaging, and Authentication," Google LLC, 2024. [Online]. Available: https://firebase.google.com/docs

[15] Telecom Regulatory Authority of India (TRAI), "Telecom Subscription Data as on 31st December 2023," New Delhi, Feb. 2024.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in Proc. Int. Conf. Artificial Intelligence and Statistics (AISTATS), Fort Lauderdale, FL, PMLR, pp. 1273–1282, Apr. 2017.

[17] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," arXiv preprint arXiv:1803.01271, Mar. 2018.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 30, pp. 5998–6008, 2017.

[19] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. IEEE, vol. 77, no. 2, pp. 257–286, Feb. 1989.

ISSN 2394 - 7780

[20] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in Proc. Int. Conf. Learning Representations (ICLR), San Diego, CA, May 2015.