# Intelligent Spam Call Classification Using Machine Learning

**Mothikumar R.**
*M.Sc. Computer Science, Vels Institute of Science, Technology and Advanced Studies (VISTAS)*
*Pallavaram, Tamil Nadu, India*
**Dr. R. Padma**
*Associate Professor, Department of Computer Science*
*Vels Institute of Science, Technology and Advanced Studies (VISTAS), Pallavaram, Tamil Nadu, India*
mothikumar.research@gmail.com

## Abstract

*The widespread adoption of automated telephony and Voice over Internet Protocol (VoIP) services has generated an unprecedented surge in malicious call traffic, placing mobile users at risk of financial exploitation and privacy violations. Conventional call-screening tools—which rely on static, number-based blacklist databases—are structurally incapable of countering contemporary threats such as AI-synthesized voice impersonation (deepfakes) and Caller ID spoofing. This study introduces a novel real-time machine learning framework designed for automated spam call detection, functioning by concurrently evaluating the network-level provenance of an incoming call and the acoustic characteristics of the transmitted voice payload. The framework is realized as a native Android application built on a Kotlin Coroutine concurrency model, enabling simultaneous lookups across multiple global spam repositories (Truecaller, Tellows, and ListaSpam) combined with on-device STIR/SHAKEN protocol validation. When a call is flagged as suspicious, the corresponding audio stream is forwarded to a Python FastAPI backend, where 40 Mel-Frequency Cepstral Coefficients (MFCCs) are computed to encode the timbral signature of the voice. A purpose-built PyTorch feedforward neural network subsequently classifies the audio as either authentic human speech or a synthetic artifact, achieving an overall test accuracy of 94.2%. By unifying caller provenance verification with deep acoustic fingerprinting into a single hybrid decision engine, the proposed system substantially lowers false positive rates and processing latency, yielding a scalable and resilient countermeasure against emerging telecommunication threats.*

**Keywords —** Spam Call Detection; Deepfake Audio; Machine Learning; Natural Language Processing; Mel-Frequency Cepstral Coefficients (MFCC); PyTorch; Kotlin Coroutines; STIR/SHAKEN

## 1. Introduction

Global telecommunications have undergone a profound transformation owing to the mass adoption of mobile devices and cloud-based telephony platforms. While these developments have democratized communication, they have simultaneously opened new vectors for cybercriminal exploitation. Among the

most prevalent of these threats is the dramatic escalation in spam, caller-ID forgery, and fraudulent voice communications.

Modern spam calls have progressed well beyond the nuisance of unsolicited telemarketing. They now constitute a serious cybersecurity hazard, facilitating mass-scale financial fraud, identity theft, and targeted social engineering. Industry telecommunications security reports indicate that billions of automated scam calls are placed every month worldwide, with malicious actors commonly impersonating banks, government bodies, or law enforcement agencies to create artificial urgency and manipulate victims into disclosing sensitive information.

Legacy countermeasures against spam calls depend primarily on static, metadata-driven mechanisms. Commercial applications and carrier-based caller identification services operate chiefly by querying crowd-sourced number databases and maintaining rigid blacklists. Although these systems provide a basic defensive layer, they are inherently reactive: a dynamically provisioned or spoofed number can completely evade detection during the early phase of an attack campaign.

The rapid advancement of Artificial Intelligence has further aggravated the threat landscape. Contemporary attackers now deploy sophisticated voice synthesis and generative AI technologies to fabricate highly convincing, human-like speech in real time. These so-called 'deepfake' voice streams can accurately replicate pitch, tone, and emotional cadence, making them nearly indistinguishable to both human listeners and rule-based detection systems.

This trajectory exposes a critical gap in existing defenses: no current production system simultaneously evaluates the network-level authenticity of a caller and the acoustic nature of the transmitted voice. This paper addresses that gap by presenting a hybrid machine learning architecture that integrates concurrent network provenance verification with AI-driven acoustic analysis into a unified, real-time classification pipeline.

## 2. Problem Statement

The core challenge in spam call detection stems from the rapidly increasing sophistication of telecom fraud. Current deployed systems suffer from a set of fundamental architectural weaknesses that significantly limit their real-world effectiveness.

### 2.1 Core Challenges

- **Zero-Hour Detection Gap:** A newly provisioned telephone number is capable of placing thousands of automated calls before it accumulates sufficient negative reports to be added to a blacklist. Throughout this window of exposure, the number is treated as legitimate by all existing screening systems.
- **Caller ID Spoofing:** Attackers exploit vulnerabilities in telecom routing protocols—including SS7 and SIP—to falsify the Caller ID metadata displayed to recipients. By masquerading as trusted or geographically familiar numbers, fraudulent calls bypass reputation-based filters entirely.
- **Absence of Content Awareness:** Existing market solutions are restricted to analyzing call metadata (i.e., the identity of the caller). None of them intercept or evaluate the actual acoustic content of the call. As a result, they cannot determine whether the voice being transmitted is that of a genuine human or a malicious AI-generated clone.

- **Vulnerability to Deepfakes:** Traditional screening systems are fundamentally ill-equipped to detect sophisticated AI-synthesized voice threats because the malicious content resides in the physical audio waveform rather than in the call routing data.
- **Elevated False Positive Rate:** Purely reputation-based filtering systems frequently produce inaccurate results, inadvertently blocking legitimate and time-sensitive communications—such as hospital alerts or delivery notifications—while simultaneously permitting spoofed attacks to pass undetected.

## 3. Related Work

Research efforts in spam call detection have historically clustered around three primary paradigms: list-based blocking, metadata heuristic analysis, and audio signal processing.

### 3.1 List-Based and Reputation Systems

The most broadly deployed approach relies on number-based blacklists, where incoming calls are matched against centralized databases of known offenders. While these systems are computationally lightweight, they are structurally reactive. Published research documents a persistent 'Time-to-Detection Gap,' confirming that blacklists remain largely ineffective against disposable VoIP numbers and against the Caller ID manipulation techniques central to modern fraud campaigns.

### 3.2 Network-Level Authentication Frameworks

To counteract caller identity spoofing, telecom regulators and carriers introduced the STIR (Secure Telephone Identity Revisited) and SHAKEN (Signature-based Handling of Asserted information using toKENs) protocol suite. These frameworks apply cryptographic tokens to digitally sign call authentication data. Nevertheless, STIR/SHAKEN has not achieved universal deployment, and it cannot detect fraudulent intent when an attacker uses a legitimately registered but compromised number [3].

### 3.3 Audio Signal Processing

Progress in digital signal processing (DSP) introduced Mel-Frequency Cepstral Coefficients (MFCCs) as a compact, perceptually meaningful representation of speech signals. MFCCs encode the short-term power spectrum of audio in a form that approximates the human auditory system. Although widely utilized in automatic speech recognition, standalone acoustic processing lacks robustness against complex environmental noise in the absence of deep learning [1].

### 3.4 Deep Learning Approaches

Recent scholarship has applied deep learning architectures—including Convolutional Neural Networks (CNNs) and feedforward Artificial Neural Networks (ANNs)—to deepfake voice detection. These models identify anomalies such as irregular pitch variation patterns and unnaturally smooth speech transitions. However, executing deep learning inference on every inbound call introduces prohibitive latency and is computationally unsustainable in real-time mobile environments [4].

### 3.5 Identified Research Gap

A critical void persists in the published literature: no existing mobile-first framework cohesively bridges high-speed concurrent network-level querying with computationally intensive deep learning audio analysis. The present work addresses this gap by proposing a hybrid architecture in which the network

verification layer functions as a rapid, low-cost primary filter, reserving AI-driven acoustic classification exclusively for calls that are ambiguous or high-risk [2].

## 4. System Architecture

The proposed framework employs a dual-layer architecture designed to optimize both detection speed and classification accuracy, operating across a mobile client tier and a scalable server-side analysis tier.

### 4.1 Mobile Application Layer (Client-Side)

The native Android application serves as the primary call interception mechanism, implemented in Kotlin and interfacing with the Android TelecomManager API to intercept incoming calls during the ringing phase—before the user is prompted to answer.

To eliminate the latency inherent in sequential database queries, the application employs a concurrent 'Race' pattern built on Kotlin Coroutines (Dispatchers.IO). The application simultaneously dispatches asynchronous API requests to global spam repositories—Truecaller, Tellows, and ListaSpam—alongside on-device STIR/SHAKEN protocol validators. Using the Kotlin select expression, results are aggregated in real time. The instant any single repository returns a confirmed spam match, all remaining background network requests are immediately cancelled, releasing HTTP connections to conserve both battery life and system resources.

### 4.2 Backend Analysis Layer (Server-Side)

When the network verification stage yields an ambiguous or elevated-risk result, the application captures a brief audio sample and transmits it via an encrypted HTTPS payload to a Python FastAPI backend. FastAPI was selected for its asynchronous request handling architecture and high throughput characteristics. The backend receives the audio, performs signal preprocessing, and executes PyTorch model inference to return a classification decision.

### 4.3 Hybrid Decision Engine

The ultimate call classification is produced by a Hybrid Decision Engine that algorithmically combines the Boolean provenance flags generated at the network layer with the Softmax confidence scores produced by the PyTorch acoustic classifier. When both signals indicate safety, the call is permitted to proceed. When mixed signals arise, the user is presented with a detailed 'Suspicious Call' alert, enabling an informed decision before answering.

## 5. Methodology

### 5.1 Dataset Description

The deep learning model was trained on a custom, class-balanced dataset comprising 31,779 labeled audio samples totalling more than 7 GB of acoustic data. The dataset is divided equally between genuine human speech recordings (Class 0: authentic) and high-fidelity AI-synthesized voice samples (Class 1: deepfake). To prevent data leakage and ensure unbiased evaluation, the corpus was partitioned using stratified sampling into training (80%), validation (10%), and held-out test (10%) subsets.

### 5.2 Feature Extraction Pipeline

Raw audio waveforms contain substantial redundancy that is computationally intractable for direct model input. To address this, the Librosa signal processing library is employed to extract 40 Mel-Frequency Cepstral Coefficients (MFCCs) from each audio sample. The extraction procedure proceeds through four sequential stages:

- Framing and Windowing: The continuous audio signal is segmented into short, overlapping analysis frames (typically 20–40 ms in duration), each weighted by a Hann window function to minimize spectral leakage.
- Fast Fourier Transform (FFT): Each frame is transformed from the time domain into the frequency domain, yielding a power spectrum that characterizes the signal's frequency content.
- Mel-Filter Bank Application: The power spectrum is mapped onto the Mel frequency scale—which approximates the non-linear frequency sensitivity of the human auditory system—by applying a bank of triangular bandpass filters that provide higher resolution in the lower-frequency ranges where vocal tract characteristics are most prominent.
- Discrete Cosine Transform (DCT): A DCT is applied to the log-compressed Mel spectrum to produce 40 decorrelated coefficients that compactly encode the spectral shape of the audio frame, effectively capturing the configuration of the vocal tract (or the algorithmic process emulating it).

### 5.3 Neural Network Architecture

A multi-layer feedforward Artificial Neural Network (ANN) was implemented using the PyTorch framework. The architecture was designed specifically for binary classification of the 1D MFCC feature vector:

- Input Layer: 40 neurons, one per extracted MFCC coefficient.
- First Hidden Layer: 128 densely connected neurons with Rectified Linear Unit (ReLU) activation.
- Second Hidden Layer: 64 densely connected neurons with ReLU activation, introducing a compression bottleneck that encourages the network to learn generalized representations.
- Output Layer: A single neuron employing a Sigmoid activation function that maps the network output to a probability value in the range [0, 1].

A classification threshold of 0.5 is applied: a posterior probability $P > 0.5$ indicates AI-generated speech, while $P \leq 0.5$ indicates authentic human speech. The model was optimized using the Adam optimizer (learning rate = 0.001) with Binary Cross-Entropy loss over 50 training epochs.

## 6. Implementation

### 6.1 Technology Stack

| Component | Technologies Utilized |
|---|---|
| **Mobile Client** | Android SDK, Kotlin, Kotlin Coroutines, TelecomManager API |
| **Backend Server** | Python 3.9+, FastAPI, Uvicorn ASGI |
| **Machine Learning** | PyTorch 2.0+, Scikit-learn, Joblib |
| **Audio Processing** | Librosa, NumPy, Pandas |
| **Spam Databases** | Truecaller API, Tellows API, ListaSpam API |

| | |
|---|---|
| **Network Auth** | STIR/SHAKEN Protocol Validators |

### 6.2 Operational Execution Flow

The system pipeline is initiated when the Android BroadcastReceiver detects an incoming call state transition. The Coroutine scope is launched, and the awaitAll() aggregation function evaluates the concurrent API responses. Should audio-based analysis be warranted, the FastAPI backend invokes librosa.load() to normalize the audio sample rate, applies a pre-fitted Joblib standard scaler to the MFCC matrix, and passes the resulting tensor to the PyTorch model operating in model.eval() (inference) mode. The predicted class label together with an associated confidence percentage are serialized as a JSON response and rendered immediately within the Android user interface.

## 7. Results and Evaluation

### 7.1 Training Performance

The neural network was trained across 50 epochs using the Adam optimizer with a learning rate of 0.001 and a mini-batch size of 64. Binary Cross-Entropy loss was used throughout to guide gradient-based parameter updates.

Training loss followed a consistent logarithmic decay trajectory, declining from 0.5998 at Epoch 10 to 0.3366 by Epoch 50. The network successfully converged without exhibiting gradient vanishing or pronounced overfitting, achieving a final training accuracy of 89.66% and demonstrating strong generalization to the held-out validation set.

### 7.2 Classification Metrics on Test Set

| Performance Metric | Score |
|---|---|
| Overall Accuracy | **94.2%** |
| Precision | **95.1%** |
| Recall (Sensitivity) | **93.3%** |
| F1-Score | **94.19%** |

The confusion matrix confirmed a highly favorable ratio of true positives to false positives. Of particular significance is the Precision score of 95.1%: in telecommunications security, minimizing false positives is the paramount objective, since a false positive results in the erroneous blocking of a legitimate caller. The elevated Precision value confirms that when the model's Sigmoid output crosses the 0.5 threshold to flag a voice as AI-generated, the prediction is highly reliable, ensuring that the system does not disrupt the user's legitimate communications.

### 7.3 Operational Latency and Efficiency

Live-environment testing on 4G/5G mobile networks confirmed substantial performance improvements compared to legacy sequential querying methods. The concurrent Kotlin Coroutine 'Race' architecture reduced total network query overhead by approximately 60% relative to sequential API fetching. Dynamic cancellation of residual HTTP requests once a spam match was confirmed maintained

the application's active memory footprint strictly below 45 MB. Average end-to-end backend response time—encompassing audio ingestion, Librosa MFCC extraction, and PyTorch inference—ranged from 200 ms to 300 ms, ensuring that the classification decision reaches the user interface before the device is answered.

## 8. Discussion

### 8.1 Funnel Architecture Versus Single-Layer Systems

Empirical testing validates the paper's core hypothesis: a dual-layer hybrid approach substantially outperforms any isolated single-layer mechanism in both accuracy and resource efficiency. Relying exclusively on network reputation APIs fails entirely against zero-hour spoofed or newly provisioned VoIP numbers. Conversely, applying deep learning audio inference to every inbound call is computationally unsustainable in a mobile environment. The proposed architecture resolves this conflict by functioning as an intelligent resource-aware funnel: known offenders are terminated immediately at the network layer at near-zero computational cost, while the MFCC extraction and PyTorch inference pipeline is invoked only for unknown or high-risk callers.

### 8.2 Interpretation of Classification Metrics

The balance between overall accuracy (94.2%) and precision (95.1%) warrants particular attention. In the context of this application, false positives are more costly than false negatives: blocking a legitimate caller—such as a hospital, emergency service, or family member—constitutes an unacceptable operational failure. The elevated precision score demonstrates that the model's classifications are highly reliable when they are made, ensuring aggressive threat filtering without disrupting day-to-day communication.

### 8.3 Resource and Latency Trade-offs on Mobile Devices

A critical success criterion for any real-time mobile security system is the ability to intercept and classify a threat before the device alerts the user. The concurrent Coroutine 'Race' architecture satisfies this requirement by reducing network overhead by 60%, dynamically releasing completed connections, and bounding memory usage below 45 MB. The 200–300 ms server response window ensures that classifications are delivered before user interaction is required.

### 8.4 Limitations in Acoustic Profiling

Despite the strong overall accuracy, live testing identified a notable limitation in complex acoustic scenarios. When the user and the AI caller spoke concurrently, the resulting merged audio stream yielded a single blended MFCC feature vector. Because the Librosa extractor processes the combined track as a single source, the PyTorch model's confidence scores occasionally degraded during such cross-talk events. This limitation underscores the necessity for speaker diarization as a pre-processing step in future production deployments.

## 9. Challenges and Limitations

Despite the strong reported accuracy, several architectural and environmental constraints must be candidly acknowledged:

- **Audio Fidelity and Network Compression:** The accuracy of the MFCC-based PyTorch classifier depends on signal clarity. Severe telecom compression artifacts, network packet loss, or high

ambient background noise can distort the frequency bands most relevant to accurate feature extraction.

- **Cross-Talk and Overlapping Speech:** Concurrent speech from both parties produces a merged acoustic profile that occasionally reduces model confidence, as noted in Section 8.4.
- **Platform Constraints:** The current implementation is restricted to Android. Porting the continuous call-interception architecture to iOS is currently constrained by Apple's strict background execution sandboxing policies.
- **Adversarial Adaptation and Concept Drift:** The continuous advancement of Generative Adversarial Networks (GANs) and diffusion-based voice synthesis models means that threat actors will progressively refine deepfake audio to evade detection, necessitating periodic model retraining on new synthetic artifacts.

## 10. Applications

The modular architecture of the proposed framework renders it adaptable across a range of real-world deployment contexts:

### 10.1 Consumer Mobile Security

Operating silently in the background via the Android TelecomManager API, the system functions as a personalized call firewall. This is particularly valuable for protecting vulnerable populations—including elderly users and those with limited technical literacy—who are disproportionately targeted by voice fraud and deepfake impersonation schemes.

### 10.2 Carrier and Network-Level Integration

The backend PyTorch inference engine can be deployed at the carrier level, integrated directly into routing hubs and Session Initiation Protocol (SIP) trunks. By sampling and analyzing audio packets at the network switch layer, carriers can proactively terminate high-confidence deepfake calls before they reach consumer devices, providing ecosystem-wide protection without requiring end-user application installation.

### 10.3 Enterprise Call Centers and Identity Verification

Financial institutions, healthcare providers, and enterprise contact centers face escalating threats from cloned-voice attacks designed to bypass biometric security or manipulate customer service agents. Integration of this framework into enterprise VoIP infrastructure enables real-time authentication of inbound callers, with automatic flagging and secondary verification triggers when synthetic acoustic artifacts are detected.

### 10.4 Regulatory Threat Intelligence

The telemetry generated by the system—specifically failed STIR/SHAKEN authentication tokens paired with corresponding MFCC acoustic signatures—can be aggregated and reported to regulatory bodies such as TRAI and the FCC. This dataset enables authorities to map VoIP spoofing campaign origin nodes and monitor the evolution of the AI voice-cloning algorithms employed by organized cybercriminal groups.

## 11. Future Work

### 11.1 Speaker Diarization and Noise Robustness

Future iterations will incorporate Speaker Diarization as an audio pre-processing stage to partition the recorded stream into separate speaker tracks prior to MFCC extraction. By isolating and analyzing only the external caller's voice, the framework will achieve significantly higher classification accuracy in noisy or cross-talk conditions.

### 11.2 On-Device (Edge) Inference

To eliminate cloud-dependent latency and guarantee voice data privacy, future development will target migration of the ML inference pipeline to the mobile device itself, leveraging PyTorch Mobile or TensorFlow Lite. On-device execution on the smartphone's Neural Processing Unit (NPU) would ensure full functionality in offline or bandwidth-constrained environments while ensuring that no voice audio is transmitted externally.

### 11.3 On-Device Caching with Android Room

An on-device local database layer using Android Room will be incorporated to cache confirmed threat signatures. Repeat-calling by previously identified scam numbers will be blocked instantaneously via a local database lookup, completely bypassing the Coroutine API race and the ML inference pipeline for known threats.

### 11.4 Continuous Online Learning

To address concept drift introduced by the continuous evolution of generative deepfake models, future enhancements will integrate continuous online learning mechanisms. The system will periodically update its model weights using newly collected samples of emerging synthetic voice artifacts, ensuring that the defense posture remains proactive rather than retroactive.

## 12. Conclusion

The proliferation of AI-synthesized voice and sophisticated Caller ID spoofing has fundamentally undermined the integrity of modern telecommunications. Legacy single-layer spam screening applications—dependent on reactive user-reported blacklists and static number databases—are structurally obsolete against adversaries capable of dynamically manipulating routing metadata and cloning human voices in real time.

This paper successfully addresses this challenge by introducing an Intelligent Spam Call Classification system built on a dual-layer hybrid defense architecture. The concurrent Kotlin Coroutine 'Race' engine reduces network latency by approximately 60% while simultaneously validating STIR/SHAKEN protocols and querying global reputation databases, instantaneously eliminating known threats. For unknown or ambiguous callers, the system transitions seamlessly to deep acoustic analysis: 40 MFCCs extracted via Librosa are evaluated by a custom PyTorch neural network, translating physical sound waves into a mathematical representation that enables detection of the high-frequency synthetic artifacts left by AI voice generators with a test accuracy of 94.2% and a precision of 95.1%.

This research demonstrates that robust telecommunication security can no longer depend on call metadata alone. By integrating network-level provenance validation with advanced acoustic fingerprinting, this work provides a scalable, resource-efficient, and transparent mechanism for defending against the next generation of voice-based digital fraud.

## References

[1] H. Jiang, S. Zhang, and Y. Wang, "Detecting Deepfake Voices: A Machine Learning Approach Using MFCC and Neural Networks," IEEE Transactions on Information Forensics and Security, vol. 18, pp. 2104–2115, 2023.

[2] A. Rahman, M. Sarkar, and H. Tasin, "Real-Time Telecommunication Spam Detection Utilizing Collaborative Global APIs and Concurrent Processing," International Journal of Computer and Telecommunications Networking, vol. 55, no. 3, pp. 112–120, 2024.

[3] L. Qing, K. Yang, and W. Tan, "A Hybrid Approach to Caller ID Spoofing Mitigation Using STIR/SHAKEN Protocols," in Proc. IEEE International Conference on Communications (ICC), 2022, pp. 450–456.

[4] T. Kinnunen, K. Lee, and H. Delgado, "Vulnerabilities of Automatic Speaker Verification Systems to Synthetic Speech Spoofing," Speech Communication, vol. 110, pp. 64–75, 2021.

[5] K. Thakur, A. Adhya, and C. Bajpai, "Concurrent Network Architectures in Mobile Environments Using Kotlin Coroutines," Journal of Mobile Software Engineering, vol. 12, no. 2, pp. 88–95, 2022.

[6] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

[7] S. Raschka, Y. Liu, and V. Mirjalili, Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python. Birmingham, UK: Packt Publishing, 2022.

[8] D. Jemerov and S. Isakova, Kotlin in Action. Shelter Island, NY: Manning Publications, 2017.