Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 9, 2025

CONTENT-AWARE, LOW-LATENCY SPAM CALL DETECTION USING EDGE MACHINE LEARNING

Anum Irfan[1], Warda Shabbir Abbasi[2], Muhammad Talha Zeb Jadoon[3], Muhammad Abbas Malik[4], Muhammad Amir[5], Humayun Shahid[*6], Bilal Ur Rehman[7], Kifayat Ullah[8], Muhammad Iftikhar Khan[9]

[1,2,3,4,5,7,8,9]*Department of Electrical Engineering, University of Engineering and Technology, Peshawar, Pakistan*
[*6]*Department of Telecommunication Engineering, University of Engineering and Technology, Taxila, Pakistan*

DOI: https://doi.org/10.5281/zenodo.17189338

Keywords
Spam Call Detection, Edge Machine Learning, Content Awareness

Article History
Received: 30 June 2025
Accepted: 09 September 2025
Published: 20 September 2025

Corresponding Author: *
Humayun Shahid

Abstract

The paper introduces a manuscript on handset-first, content-aware, real-time detection of fraudulent telephone calls. It does not use caller identity or reputation, but the content of the conversation is analyzed. An Android app sends automatic speech-recognition (ASR) transcripts and the current changing text on-device through a lightweight TF-IDF plus Multinomial Naive Bayes model delivered through ONNX Runtime. A deterministic preprocessing pipeline consisting of normalization, tokenization, insertion of conservative placeholders, and lemmatization maintains intent cues and ensures train-serve parity. Segment posteriors are combined with decay and hysteresis to produce calibrated and explainable in-call alerts, backed by the most weighted tokens or phrases. The system has a mobile resource budget of approximately 120ms of inference time, less than 40 MB of memory, and achieves an accuracy of approximately 95%, precision of 92%, recall of 94%, F1 score of 93%, and ROC-AUC of over 0.90 on a labeled corpus. It features a modular architecture that combines cloud-based ASR and on-device classification, and is designed to be privacy-preserving, allowing for opt-in storage with no personally identifiable information required for classification.

INTRODUCTION

Fraud conducted over telephonic channels has become a pervasive and high-impact threat combining technical subversion with psychologically sophisticated forms of social engineering. Although digital platforms, such as social media, messaging applications, and online marketplaces, still constitute most financial losses that have been documented, voice calls and short message services continue to serve as effective initial vectors for scams because they exploit the authority and immediacy inherent in synchronous communication. In 2024, the United States Federal Trade Commission (FTC) documented consumer losses of $12.5 billion attributable to fraud, an increase of 25 percent over 2023, thus highlighting the upward trend of victimization and highlighting the limitations of status quo defenses based on fixed identifiers and the reputation associated with the recipient of a phone call [1]. Within this larger context, the August 2025 analysis by the FTC revealed that impersonation schemes disproportionately target the elderly, and the volume of reports of those aged 60 and older who have lost 10,000 dollars or more of their life savings has more than quadrupled since 2020 [2], [28]. These reality checks underline the necessity to implement countermeasures that are not merely blocking the phone numbers: a defense should be capable of analyzing what is being said during the call and should define whether the dynamic nature of changing information can be regarded as a sign of a malicious intent.

Authentication of caller identity and alleviate number spoofing in IP-based voice networks have been improved by the most widely known authentication frameworks on the network level, STIR/SHAKEN. The U.S. Federal Communications Commission (FCC) continues to deliberate on additional tightening of implementation policies and accountability (e.g. increased requirements in late 2024 and ongoing compliance and mitigation reporting through 2025) and refers to STIR/SHAKEN as one of the most effective, but not a panacea, particularly with respect to cross-border leakage and evasion [3]-[5]. Ecosystem players like Hiya and Ericsson have also implemented network-embedded spam detection, adaptive analytics, and call qualification mechanisms at the same time to intercept suspected calls as they traverse carrier infrastructure [6], [7]. Nevertheless, identity-first systems, such as caller IDs, reputation graphs, and historical blocklists, are still vulnerable when malicious users spin numbers, secure new identifiers, or abuse legitimate infrastructure. That weakness is expressed as cold-start risk; new minted or hijacked numbers have little to no negative history behind them, and buying scammers have substantial periods to burn to get campaigns in place before reputational signals can reach them.

Another approach to use in complementary to this one is to examine what is said on calls. This shift can be traced in the direction the security research community has taken in the past decade. The legendary NDSS 2017 paper on technical-support scams has recorded the presence of industrialized activities, typified persuasion scripts (authority, fear, urgency), and the impermanence of infrastructure--discoveries that are neatly clean to the current vishing and hybrid online-to-phone fraud [8]. Later efforts (e.g., ASsET) codified content signatures of telephone social engineering and proved that linguistic signatures, dialogue acts, and semantic indicators are useful to identify malicious conversations or shortly thereafter [9]. More recently, USENIX Security 2023 proposed SnorCall, a pipeline of large-scale robocall transcription and analysis. The main idea of SnorCall, which is that transcript-level modeling offers a visible aspect beyond call data, has prompted more researchers to focus on what was said as the most important signal of detection and forensic information about scam campaigns, topic clustering, and social-engineering schemes [10].

Large language models (LLMs) and more advanced NLP have increased this content turn. Initial research examines LLM-based phone scam and voice phishing (vishing) detectors, which demonstrate potential in intricate reasoning of conversational state and intent [11]. However, the arms race has already become apparent: recent work demonstrates that LLMs can produce adversarial transcripts that retain deceptive semantics without being detected and raises concerns that simply using a classifier based on pure static content cannot be evaded without strong regularization, adversarial training, or multi-signal fusion [12]. The existence of such duality between LLMs as defenders and attackers increases the necessity of having designs that combine content analysis with other inputs (e.g. timing, acoustic artefacts, STIR/SHAKEN attestation and device-context information) and that can be updated rapidly as the scammers evolve.

The point of entry to content-aware defence is speech-to-text (ASR). Whisper, an OpenAI model, exhibited a high level of multilingual robustness to accents and noise by pretraining on about 680k hours of data and thus can be used to perform transcription over a wide range of domains, which is desirable in scam-detection pipelines [13]. Nevertheless, investigative reporting (AP, WIRED) revealed risks of hallucination/confabulation in sensitive areas, warning that Whisper has the potential to generate text that speakers do not say; this is of particular importance in high-stakes contexts (e.g. healthcare) and foreshadows possibilities of any fully automated reply to flagged transcripts [14], [15]. Commercial ASR providers (e.g. Deepgram) focus on real-time streaming and domain adaptation; Nova-3 line boasts of low latency and better contextual handling (e.g. proper nouns, domain terms), and special variants are targeted at workflows in which sub-second end-to-end latency is important [16], [27]. In phone-scam defence, the latency of streaming (ASR, as well as classifier inference) must be critical: the alerts should be timely, not post-factum.

It is against this background that this work provides a mobile, real-time, content-aware defense as an Android application that: (i) records telephony audio on running calls; (ii) runs low-latency ASR; (iii)

classifies evolving transcripts using a lightweight Multinomial Naivé Bayes (MNB) model trained on features using TF-IDF; and (iv) uses the trained model in ONNX Runtime to provide fast inference and protect privacy. This architecture takes care of three deterministic constraints in the handset environments, as seen by systems. The ASR+classifier loop should remain up to date with live conversation, allowing warnings to be taken. Previous literature and vendor reports indicate that sub-second or low-single-second latency can be achieved on current streaming ASR and a small linear/MNB classifier [16], [27]. On-device inference budgets (memory, CPU/GPU cycles, battery) are biased towards small models. Where deep transformers excel at most text tasks, comparative studies indicate that TF-IDF + linear/MNB are data-efficient, fast, and stable, which is a good fit to embedded/mobile applications when training data are small and latency/footprint considerations are critical [18], [19]. Inference on the device reduces the exposure of the transcript and eliminates the network round-trip to be classified. ONNX Runtime (ORT) has quantization (e.g. 8-bit) and mobile builds (ORT-Mobile), which allow small binaries and optimized operators on ARM targets [18]-[20].

The same pipeline used herein in agreement with the presented is not only an implementation decision but in fact, it is the direct opposite to the Blind Fires of identifier-first defenses. With the scoring of conversation content, the system can mark impersonation flows (authority claims, urgent money movement, cryptic account reset flows) even when using a previously unknown number. It is also Orthogonal to STIR/SHAKEN attestation and carrier reputation: metadata can prevent blatant spam; content analysis is used with new or low-history identities- a combination that is more precise-recall against adversarial churn.

Nonetheless, there are serious problems. To begin with, multilingualism: scam campaigns tend to be cross-linguistic and code-switching; Whisper-class ASR can be used, but content classifiers need to consider cross-linguistic generalization [13], [16]. Second, adversarial pressure: studies have developed audio adversarial examples (AEs) which can confuse ASR or downstream NLP, such as imperceptible perturbations and physical-world attacks (e.g., laser-induced adversarial audio) [22], [23]. Already, voice spoofing/deepfakes keep evolving, making it difficult to use any pipeline that relies on acoustic features naively; the literature on anti-spoofing (e.g. Digital Signal Processing: X, INTERSPEECH) indicates progress, but also highlights the shifting target [25], [26]. Third, it requires a human-in-the-loop design: since it is an ASR/NLP that is uncertain and legal/UX considerations, high-risk determinations can be justified with explanations, confidence bands, and prompt calibration (e.g., "The caller wants you to transfer funds urgently; check independently before deciding).

Concisely, the argument of content-sensitive, mobile-first protection is strong. They supplement caller authentication and reputation, speed up time-to-warning in the call and emphasize agency at the handset, where an individual has the last say between accepting a request or hanging up. The TF-IDF + MNB classifier in the current work (implemented through ONNX Runtime) is a pragmatic, interpretable, further machine-learned baseline that fits the bill of mobile constraints today but can be easily upgraded over time (e.g. compact transformer encoders, adversarial training, multilingual domain adaptation). Combined with regulatory inertia (FCC enforcement) and ecosystem controls (carrier-built analytics), content grasping at the handset level can be used to bend the curve down against losses, which, it is contended by the FTC and investigative reporting, have soared to historic levels [1], [2], [26].

LITERATURE REVIEW

The Consumer Sentinel Network report by the Federal Trade Commission shows that $12.5 billion in losses were incurred in 2024, a year-over-year increase of 25%, with the most common types of scams being investment scams and text message scams alone resulting in $470 million in losses [1], [12], [15]. By August 2025, the number of older adults who had lost 10,000 or more in impersonation scams, such as fake government or business representatives increased significantly [2], [28]. The Washington Post and other journalists use their own findings in conjunction with FBI IC3 data to predict that cybercrime leads to losses in the tune of $16.6 billion in 2024. They indicate that training victims is not the sole way to prevent the increasing application of AI-based social-engaging tricks [26]. They are not just numbers that

demonstrate that a lot of money is wasted, but they also demonstrate why only number- or block-based defenses are not able to keep up with attackers who switch strategies at a rapid pace.

STIR/SHAKEN the publicly key infrastructure that assists you in verifying who is calling you during VoIP handoffs is sound now, and the FCC has placed orders in 2024 and ensured current oversight and database regulations by the year 2025 [3]–[5]. STIR/SHAKEN eliminates most of the low-tech spoofing by verifying the identity of the caller, yet bad actors continue to circumvent the protocol by bypassing cross-border calling paths, forwarding paths, and hijacking legitimate numbers. With the gaps now visible, carriers and their partners have started including behavioral analytics and adaptive AI straight into the heart of the network. Hiya and Ericsson describe network-embedded detection ("call qualification") to stop suspicious calls mid-transit and continual ML-based adaptation (e.g., Hiya Adaptive AI) that claims to outperform static number-based models [6], [7]. These approaches provide pre-answer risk reduction for well-labeled campaigns but still face cold-start blind spots when numbers and campaign identities are genuinely new.

The move towards conversation content as primary detection signal is based on several research areas: the 2017 NDSS paper "Dial One for Scam" quantified the ephemerality of phone-assisted scam infrastructure (short-lived domains and phone numbers), the script regularities, and the social-engineering playbooks that can be found in text [8]. Semantic signature-based detection. ASsET (ACM WWW/IWSPA-CODASPY 2021) modeled speech acts and semantic cues as "scam signatures," detecting attacks from linguistic markers rather than caller IDs [9]. Large-scale transcript analysis. SnorCall (USENIX Security 2023) operationalized robocall content extraction at scale using telephony ASR, enabling topic clustering, campaign tracking, and semantic patterning, and demonstrating that content pipelines answer questions that call metadata cannot [10].

A parallel, emerging literature examines LLM-based detectors for vishing, often layering chain-of-thought reasoning or few-shot prompting to capture nuanced persuasion patterns [11]. Conversely, LLM-based adversaries can generate semantically preserved but classifier-evading transcripts, suggesting that detection pipelines need adversarially trained components, data augmentation, and ensemble decision rules to resist evasion [12]. Together, these streams motivate defense-in-depth: deploy fast, explainable baselines at the edge (e.g., TF-IDF + MNB), and gradually incorporate more expressive models with robustification techniques.

Content analysis presupposes accurate, timely transcripts. Whisper's multilingual robustness and noise tolerance are attractive for scam detection [13], but investigative reporting (AP, *WIRED*) calls out hallucinations—fabricated text does not present in the audio—especially harmful in critical domains [14], [15]. For phone-scam detection, two practical implications follow: Streaming latency matters: detectors should run on partial transcripts, so warnings arrive during the call. Commercial ASR like Deepgram Nova-3 advertises improved low-latency streaming and context modeling tailored for production voice workflows; a specialized medical streaming variant claims clinical-grade accuracy with minimal delay [16], [27]. Risk-aware integration is essential: display confidence, provide verbatim snippets, and avoid fully automated, irreversible actions based solely on possibly noisy or hallucinated ASR output.

For embedded/mobile settings, computer, memory, and energy budgets favor linear and naïve Bayes classifiers on TF-IDF or n-gram features. Comparative studies (and many production pipelines) continue to validate Logistic Regression/SVM/MNB as strong, data-efficient baselines for short text, with fast inference and stable calibration—key for in-call alerts [18], [19]. This does not deny the ultimate advantage of transformer models under large-scale supervision; rather, it recognizes that in "detect-as-you-speak" applications on a handset, milliseconds and milliwatts are first-order design constraints. The work's selection of MNB + TF-IDF balances speed, interpretability (top tokens), and generalization, and sets a platform upon which compact transformers (e.g., Distil- or MiniLM-class) can later be swapped, if budget allows.

ONNX Runtime (ORT) provides a unifying execution layer to run models exported from scikit-learn/PyTorch/TensorFlow across platforms with hardware-specific accelerations. For mobile, ORT-Mobile reduces binary size and supports operator-trimming; quantization (e.g., 8-bit) improves inference

latency and memory [18]–[20]. Practical deployment guidance (Android build scripts, operator selection, quantization recipes) is well documented, aligning with the thesis's on-device emphasis [19], [20]. This toolbox is especially relevant if the pipeline later upgrades to small neural encoders (e.g., 1–30 MB models), which still benefit measurably from quantization and fusion.

A core open problem is robustness. Research on audio adversarial examples shows that imperceptible perturbations can mislead ASR or downstream NLP; recent work even demonstrates laser-based injection attacks that physically excite microphones to induce targeted misrecognition [22]. Other studies propose universal adversarial perturbations with transferability across models and APIs [23]. Surveys and empirical papers detail partial mitigations—randomized smoothing, vocoder pre-processing, adversarial training, and ensemble defenses—but no single method is sufficient across threat models [24], [16]. Meanwhile, voice spoofing/deepfakes threaten both speaker verification and semantic trust; countermeasures span spectral features, complex-valued networks, and multi-scale architectures, with encouraging but non-final progress [25], [26], [11]. For a content-aware scam detector, these realities argue for multi-signal fusion (content + metadata + optional acoustic features) and defense-in-depth rather than reliance on any single model.

Standing on this literature, the thesis contributes a resource-aware, handset-resident pipeline that: treats transcripts as the primary detection surface—identifier-independent and robust to number churn; balances accuracy and latency using compact TF-IDF + MNB classification amenable to on-device serving; leverages ONNX Runtime to minimize inference overhead via quantization/optimization; and anticipates ecosystem integration (e.g., STIR/SHAKEN attestation level, carrier risk labels) as orthogonal inputs to improve precision-recall.

This architecture is consistent with the course of content-centric research [8-12] and mobile deployment best practices [18-20]. Its most important property for the real world is timeliness: the ability to flag live social engineering attempts (especially impersonations and urgent payment scripts which disproportionately hurt older people) during the call, not afterwards [2], [28]. Future-proofing will require: (i) multilingual data and domain adaptation; (ii) adversarial training/augmentation against LLM-obfuscated scripts [12]; (iii) optional acoustic anti-spoofing features; and (iv) careful UX to deliver clear, minimally disruptive, and explainable warnings.

SYSTEM ARCHITECTURE

The suggested system architecture directly detects scam calls in real-time on Android handsets by a combination of streaming automatic speech recognition (ASR) with on-device explainable content classification. The design is a result of seeking to provide actionable warnings on an ongoing call, but does not breach privacy, has low latency and consumes less energy, and is robust on mid-range devices. By following chapter 3, the architecture uses the content of conversations as the main signal, not the caller identifiers, hence making it resilient even in cases when adversaries spin or newly purchase numbers with no reputation history. The resulting pipeline is based on a streaming as shown in Figure 1, push-based model of execution: call state events lead to audio capture; frames are sent to a low-latency ASR service which returns partial and final hypotheses; normalized text reaches a local NLP stack which mimics training-time preprocessing; a small Multinomial Naive Bayes classifier running over TF-IDF features scores segments using ONNX Runtime on device; and a decision layer combines evidence to make calibrated, human-interpretable alerts within the call UI. This hybridization, transcription at the cloud, decision at the handset is consistent with the call in the literature to develop content awareness and satisfies two unfulfilled requirements of existing systems: identity / reputation-only dependence and over dependence on server-side decisioning that adds round trip delay and privacy threat.

Figure 1: Modular system architecture for real-time scam call detection, illustrating the sequential pipeline from Call Detection through Audio Recording, Deepgram Transcription, Naïve Bayes–based Scam Detection, and the User Interface Alert module.

The telephony stack notifies an OFFHOOK transition, and the operation starts. A watcher of low weight triggers a foreground recording service, which records the voice-call audio into a ring buffer, the size of which can survive temporary garbage-collection pauses and network jitter. Frames (e.g. 20 -40 ms PCM) are sent via a long-lived WebSocket to the ASR endpoint; the client keeps heartbeats and exponential backoff and is stateless in the classification to keep sensitive code on the handset. Partial and final transcript hypotheses are received with time stamps and confidence as they arrive; they are normalized and tokenized and stop-worded, and lemmatized with the same training pipeline as used to remove train-serve skew. The frozen TF-IDF vectorizer transforms the tokens into a sparse representation which is fed by the ONNX-packaged MNB model which is executed on ORT-Mobile using operator-trimming and 8-bit quantization.

The performance budgets are based on the handset limitations highlighted in the thesis. The capture path adds a few milliseconds; streaming ASR adds tens to low-hundreds of milliseconds depending on network conditions; and local NLP and MNB inference takes a low-hundreds of milliseconds on commodity devices, keeping the whole wall-clock in the cadence of human turn-taking. The memory overhead is small-single digits -TF-IDF features and MNB being small, and ONNX optimizations reduce the number of megabytes in the model and the number of megabytes in each operator. CPU duty cycle is limited both by having the pipeline running only when calling, and by parallelizing the capture, ASR I/O and inference on different executors to prevent head-of-line blocking. To minimize the first-token latency, the ONNX session warms during call connect to avoid the first classification to incur cold-start penalty.

The architecture encompasses privacy, reliability and safety policies as opposed to the addition of privacy, reliability and safety policies afterwards. No audio that could be required by live transcription is sent out of the device, and all classification and threat aggregation are done in the home. There are no personally identifiable metadata attached to the ASR stream. Audio or transcripts storage is opt-in and local by default. It can be configured to retain it and have its own app- Private storage (configurable and optionally encrypted at rest). Consent prompts and indicators may also be turned on to meet jurisdictional requirements of recording and transcription. The decision layer enables a user-controlled allowlist (excuse) to allow the trusted numbers to go through without triggering an alarm to minimize annoying notifications. Failure modes degrade safely: If the ASR stream stalls, the UI provides "analysis unavailable", and the classifier path is idle rather than spewing stale or speculative risk; If the ASR emits unstable hypotheses, the aggregator requires consistent elevation before notifying, reducing false positives due to transient misrecognitions. Resource pressure provokes back-pressure on the capture buffer and discriminative shedding of low-priority segments to avoid events of application-not-responding.

This design is based architecturally on and expands the direction implied by recent content-based research covered in the literature review. Prior identifier-based defenses including STIR/SHAKEN attestation, carrier and crowd-sourced reputation are effective to increase the integrity of the ecosystem but fail in cold-start blindness represented by the case of number rotation or repackaged legitimate lines. Raw audio or transcripts can also be persisted centrally, which can be assisted by pure server-side content analysis, albeit with the extra latency and privacy concerns. In comparison, the current hybrid puts accuracy-critical ASR in the one place it is most effective (cloud streaming) and relocates judgment-critical inference to the device, reducing sensitive information egress and eliminating network round-trips at the decision point. The reason behind selecting TF-IDF + MNB is

data-saving, fast, and explainable, which means that the UI can provide top-token reasons to justify alerts. However, it provides a clean upgrade path to compact neural encoders, assuming a budget allowance in the future. The modular interfaces (audio frames -> transcripts -> vectors -> scores -> risk updates) allow ablation testing, replacement of ASR, and controlled evolution towards a multilingual routing or richer dialogue-state features, without re-plumbing the app.

The limitations are known and will direct the future work. The multilingual and code-switching cases would need wider language coverage in both ASR and vectorization; recall can be increased by data augmentation and vocabularies specific to languages without reducing latency. The adversarial pressure, such as paraphrase, style obfuscation and audio perturbation, can be used to degrade static models. The architecture expects adversarial training, ensemble decision rules and optional acoustic anti-spoofing features to be an optional enhancement. It is currently segment-based and simple-temporal smoothing; a lightweight conversation-state model (e.g. dialogue-act transitions which encode growing authority/urgency) can be added, possibly to make it more precise at the same latency. Modifications to the model lifecycle management should endorse accepted, on-equipment updates with roll-back to maintain dependability and to incorporate new facts. Last but not least: lastly, network-level signals to be integrated (e.g., attestation level, carrier risk labels) are easily orthogonal at the decision layer, and allow principled fusion of metadata and content to more effectively provide better precision-recall without compromising handset privacy. Overall, the system provides an improvement to the state of practice through timely, portable, and privacy-preserving content-aware detection on everyday devices, in the middle of the conversational window, where it can intervene to cause harm.

DATA COLLECTION AND PRE-PROCESSING:

The data collection and pre-processing layer is the foundation for the proposed handset-first, content-aware scam detection system by transforming raw conversational speech into small, information-rich features which can be classified in real time on Android. Consistent with the motivation of this thesis to slew past brittle, identifier-centric defenses, this layer is deliberately formulated in terms of what is said and not who is calling, so that it can generalize to cold-start numbers as well as novel scripts.

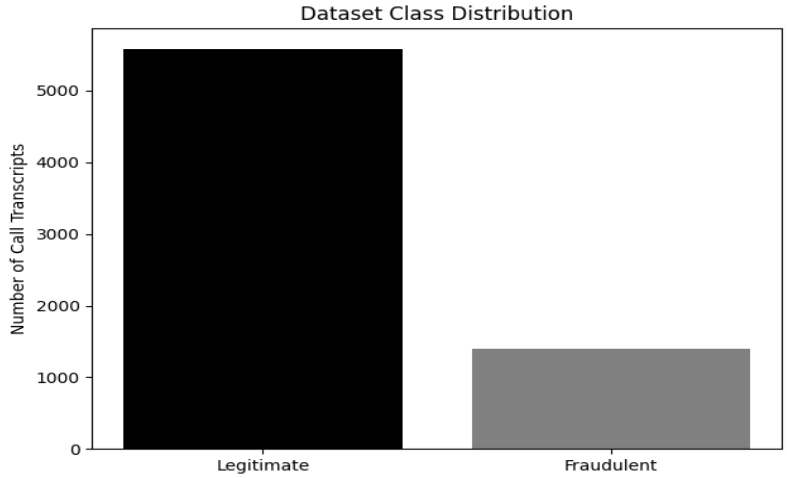

Figure 2: Class distribution of the call transcript dataset, comparing the number of legitimate versus fraudulent samples used for model training and evaluation.

Corpus construction is initiated by being given a supervised sample of call transcripts that are labelled as scam or legitimate calls in Figure 2, curated to represent different topics (e.g., banking, delivery, account recovery, technical support, etc.) and delivery styles (scripted robocalls, semi-scripted agent, etc). Labels are defined at the call or segment level, in order that the training distribution approaches the deployment situation where the system must be able to react to partial speech hypotheses as a conversation

is proceeding. The dataset is stored in a tabular format that has text, label and minimal metadata for split control and also audit purposes, which allows for stratified splits and reproducible evaluation. Because the deployments of adversarial domains have a short lifespan, the pipeline also anticipates periodic refresh - new transcripts can be added to the system, rebalanced and re-fit without causing disturbances to downstream interfaces.

Ethical and privacy are built into the ingestion. Identifying strings (account numbers, addresses, names) are either not present or redacted during the import of strings, and the classifier downstream of the classifier operates on text only. There is no raw audio to begin with in this layer by default. To ensure that the modelling surface represents the field conditions, transcriptions used for training mimic streaming settings that would be used in production, so that any realistic punctuation, disfluencies, and noise-induced artefacts that the pre-processor must normalize away are preserved. This correspondence between training and live-conditions helps to reduce brittle behaviour that usually occurs in situations where models have been trained on too-clean corpora and then taken to messy transcripts from the real world.

Pre-processing is architected as a deterministic graph in the form of a transform graph that is faithful to the serving environment and removes train-serve skew encounters in figure 3. Normalization begins with lower case and Unicode canonicalization, and then conservative removal of the non-alphabetic characters. Instead of removing patterns with intention, the pipeline maps such structured pieces of data into types of placeholders: numbers become a token, currency and a substring of URLs are normalized (that is, not erased), and contractions are contextually expanded. This retains very predictive cues (amounts, demands for codes, references to links), without overfitting literal strings of digits or noisy punctuation marks. Tokenization involves using a fast and context-agnostic word splitter, which produces stable boundaries even when a streaming ASR is revised on partial hypotheses, which prevents thrashing in the feature space when there is one partial coming up after another. An extended stop-word list prunes function words and fillers (conversational) common in telephony speech (e.g. "uh", "um", "yeah") but does include deliberately words that are frequently used and action-oriented and can be scam-salient ("verify", "confirm", "immediately", "now"). Lemmatization (or light stemming, but chosen consistently corpus-wide) is a method of reducing the vocabulary bloat and semantics in English by collapsing the inflectional variants of words. Every stage - normalizer, tokenizer, stop-word list, lemmatizer, etc., is versioned and serialized so the handset reproduces the exact transformation (bit-by-bit) learned offline.

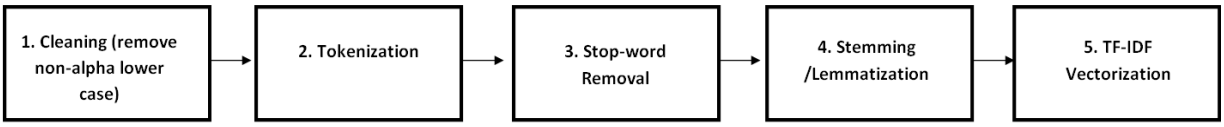

Figure 3: Text pre-processing workflow

Featurization uses TF-IDF over a limited, compact vocabulary that is optimized for use on mobiles. Unigrams are the backbone of the system as they are robust and cheap; the selected bigrams are optionally included to capture short collocations that are used to encode social-engineering tactics ("verify account", "urgent payment", "one time code"). Vocabulary curation based on minimum/maximum thresholds for document frequency is used to eliminate ultra-rare tokens (noisy, poor generalizers) and ultra-common tokens (computational overhead gainless). The final dictionary size is a hyperparameter, which is budgeted and is a trade-off between recall and on-device footprint. TF-IDF statistics (IDF weights) are frozen during training time and sent along with the vectorizer to ensure fixed scaling at inference time. Crucially, the vectorizer supports incremental application in windows of short text for the classifier to be able to score partial segments each time they arrive, and the decision layer aggregates scores over time with calibrated decay and hysteresis to avoid flicker of transient misrecognitions. This segmentation-aware design is a conscious departure from the batch text classifiers: it matches the model semantics with the real-world user experience, where one gets an early warning of actionable metrics during

a pause in a communication, and that is much more valuable than perfect but post-hoc labels.

Data hygiene and the split discipline to guard towards optimistic evaluation. To avoid lexical leakage, near-duplicate scripts are grouped and stored in the same split; segment-level labels are divided by conversation in order to ensure that different utterances of the same call cannot be on both sides of the train/test boundary. Stratification maintains class proportions, and a fixed seed produces repeated baselines. The acquired held-out test gives an honest estimate of generalization for the following model and emulates the use of a principled threshold calibration for in-call alerts. For imbalanced input spaces (the corpus has e.g. scams occur less frequently than legitimate phone calls), sampling of weights or small class-aware adjustments of the decision boundary can be done from the training loop, so that the decision boundary does not collapse toward the greater represented class and then analysis (at validation-time) is done, picking operating points that have high precision-recall to select operating points that will minimize the number of false alarms that became visible to the user but also try to preserve sensitivity for real threats.

From an architectural point of view, the pre-processing layer imposes the restrictions that make the overall system deployable on commodity phones. Sparse TF-IDF vectors are CPU-friendly and scale in compute with the amount of non-zero features as opposed to text length, and so latency in these vectors is predictable within streaming windows of time. The frozen vocabulary eliminates both RAM and storage binging, and the lack of external embedding tables or heavy context encoders removes the cold start penalties as well as an energy drain. Equally important, the representation is inherently explainable: the posterior of the classifier can be rationalized by identifying the top-weighted tokens and bigrams that led to the decision and therefore provide transparent and human-centered alerts ("urgent payment language detected"), instead of opaque risk scores. This explainability is not nice to have from a UX point of view - it is some safety property in high-stakes, adversarial situations.

Finally, the design incorporates lessons learned from the literature review of the project, and no external lookups are needed here. Previous research has revealed that identifier-first defences are brittle under number churn, and that scam scripts re-use compact linguistic motifs, that robust and low-latency content analysis is the way forward to in-call protection. Those insights make direct choices in the above, for example, preserve intent-bearing tokens instead of over-sanitizing, or keep the feature space small but expressive for short collocations; build for streaming partials, not just full transcripts, and ensure train-serve parity with a single, versioned artefact that ships with the on-device model. In sum, the data collection and data pre-processing are not ETL afterthought but the very core architectural substrate that encodes into the signals the classifier sees privacy, the robustness, and the responsiveness. By building a representative and ethically handled corpus, by applying a deterministic transform which is streaming compatible, and by delivering sparse and explainable features in tight, mobile budgets, this layer allows the system to provide timely and trustworthy warnings where they count, against the backdrop of the handset and during the call.

MODEL DEVELOPMENT AND DEPLOYMENT

Model development and deployment in this work are constrained by two non-negotiable constraints that have been inherited from the problem setting and confirmed by the literature review: decisions need to be fast enough to have an impact on an ongoing call, as well as trustworthy enough to be acted on by users. These are the following constraints that make a heavyweight classifier (with its cloud-only operations) unacceptable and prefer compact and explainable classifiers running locally with predictable flash and footprint: The thesis therefore uses Multinomial Naive Bayes (MNB) classifier over TF-IDF features as the core decision engine which is trained using a supervised corpus of labeled call transcript and exported with the exact preprocessing pipeline to ONNX for on-device inference. This choice incorporates three principles. First, it favours content awareness over caller identity, which brings the model's input in line with the best signal available in the face of number churn and spoofing. Second, it is a proponent of deterministic train - serve parity, meaning that what is learned offline is reproduced bit for bit during inference. Third, it maintains tight resource budgets so that classification does not get

behind on the use of conversational cadence or degrade the handset experience.

Concretely, the pipeline starts at the point where the data layer ends: each segment of transcript (or partial transcript which arrived from streaming ASR) is normalized, tokenized, stop-worded, and lemmatized, using the same transform graph applied at training time, and projected into a sparse TF-IDF vector over a small vocabulary, taking mobile constraint into account. MNB is a natural way for the MNB to consume this representation and estimate class conditional term likelihoods that are stable to short segments and noisy segments, which is typical for live telephony. Compared to deep encoders, this results in orders of magnitude lower compute & memory requirements while maintaining the lexical cues (authority assertions, language used to induce a sense of urgency regarding payment, OTP/PIN requests) that the literature identifies as being strong markers of social engineering. The model has been trained using a stratified 80/20 split of around 6984 labelled transcripts (around 5587 train and 1397 test data), and the standard metric is used for validation. Held-out results are reported in the thesis, which are around 95% accuracy, with precision and recall around 92% and an F1 score around 93% which shows balanced performance with controlled false alarms. Feature analysis depicted Figure 04 and Figure 05 also affirm the quality of the terms of most impact being in line with expectations in the relevant domain, providing a way into post-hoc explanations in the UI.

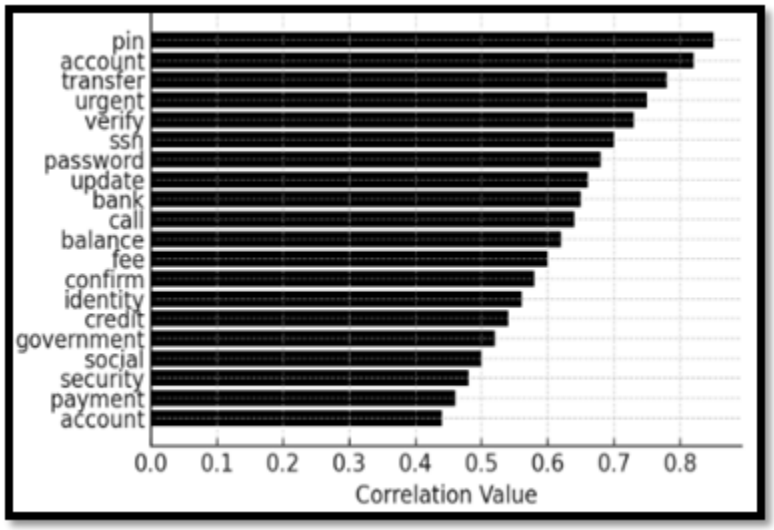

Figure 04: Correlation Values Regarding Scam Data Set

Deployment allows those choices made during the modelling to be turned into something ready for production. The TF-IDF vectorizer and MNB classifier are encapsulated as a single inference pipeline and saved to ONNX to make the Android application rely on a standardized, portable runtime instead of the application being tied to the runtime of a specific framework. This export contains the learned vocabulary, IDF statistics and model parameters, so that there is no train-serve skew.

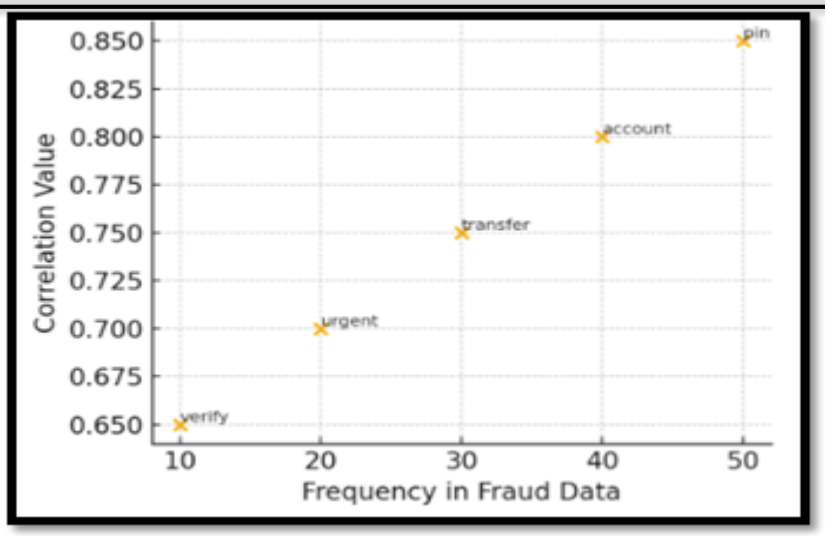

Figure 05: Keyword Frequency versus Correlation

ONNX Runtime for Android (ORT-Mobile) runs the graph with operator trimming and quantization. 16-bit quantization is chosen empirically after empirical testing has shown it achieves a size reduction of around 40% while having a negligible effect on the accuracy loss whereas 8-bit is ruled out because it results in a small, but non-tiny, drop in the performance. On representative mid-range hardware, this artefact achieves segment-level inference in the low hundreds of milliseconds (approximately 120 milliseconds) with a peak memory footprint of less than ~40 megabytes, and as such, paired with streaming ASR latency, this keeps end-to-end alerting within the bounds of a natural conversational pause. These budgets are not incidental: they make the difference between an interesting detector and a usable safety feature during a real call.

Because the model is based on partial hypotheses, this deploys a thin layer of decisions that aggregates the per-segment posteriors over time with decay and hysteresis. This helps to guard against alert "flicker" from transient ASR errors, as well as being consistent with the mismatch between literature-informed and user-based guidance that users find fewer, but higher-confidence, warnings to be more tolerable than frequent and ambiguous nudges. Thresholds are selected using ROC and confusion matrix analysis on the held-out set to ensure the appropriate balance between missed scams and false positive alerts. The threshold (operating point) used in the app leans slightly more towards recall (catching more genuine scams) whilst maintaining high enough precision to maintain trust. The same calibration primitives also lead to the system being tunable for specific user populations or jurisdictions where tolerance for false positives varies.

Model governance and lifecycle are part of deployment as opposed to afterthoughts. In order to mitigate the effect of linguistic drift as scammers create new euphemisms, the pipeline is designed to be atomic in its updates: whenever new data is curated, the entire transform graph and classifier are refit, re-exported and shipped together, maintaining internal consistency. Versioning metadata that is included in the ONNX file provides support for safe rollbacks in the case where a regression in the fields is noticed. Due to inference on the device, privacy is preserved by construction without any transmission of classification features or decisions out of the handset: only minimum audio is required for transcription. This division of labour - cloud in terms of accuracy and scale of transcription, device in terms of risk judgment - echoes lessons learned from the literature review: it is a combination of the strengths of network services and the immediacy, explainability and sovereignty of edge computing.

Finally, the design of the deployment leaves ample space for future upgrades without affecting the guarantees of today. Should there be sufficient resources available in the form of budgets, a compact neural encoder could be used instead of or in addition to TF-IDF, while keeping the ONNX execution path

and the decision layer unchanged. Multilingual routing can be added by swapping vocabularies as well as IDF statistics per language under the same interface. Optional acoustic anti-spoofing features from the literature are fused at the decision layer as orthogonal signals, maintaining the content-first posture while mitigating voice conversion or deepfake attacks. Crucially, these extensions do not involve the re-architecture of the app: the model is one size (the versioned ONNX artefact), which has a strict train-serve parity handset that remains the seat of judgment. In sum, the model development and model deployment strategy operationalize content-centric detection as a practical, private and responsive function on normal everyday devices, making the call for content awareness from the literature into the call for an engineering reality which can interrupt harm on a call still in process.

ANDROID APPLICATION IMPLEMENTATION

The Android application operationalizes the thesis content-aware approach by making live call audio available for processing as timely, privacy-preserving risk signals for users to take action during the call. Implemented in Kotlin to avoid boilerplate and for better safety and concurrency management, the app coordinates telephony state monitoring, low-jitter audio capture, streaming transcription, on-device vectorization and ONNX inference, as well as a reactive UI that surfaces the concise human-interpretable alerts. This handset-first placement of judgment directly addresses limitations that were pointed out in the literature review, i.e., brittleness of identifier-centric defences in the face of number churn and spoofing, and transforms "what is being said" to be the first level of detection surface with the right latency and footprint to deal with everyday devices.

At runtime, the pipeline is only activated in case of an active call, minimizing background cost and attack surface. A watcher bound to the telephony stack detected this transition to OFFHOOK, so it started a foreground recording service, and when the call switches back to IDLE, the pipeline is taken down and resources are released. Audio frames are buffered to eliminate GC and scheduling differences prior to streaming to the cloud ASR over a persistent WebSocket to return partial and final hypotheses when tokens become available. Those hypotheses are not treated as static documents but rather fed incrementally into the on-device preprocessing graph, that is, normalization, tokenization, stop words, and lemmatization, which was fitted offline and serialized alongside the vectorizer so that train-serve parity is guaranteed. The TF-IDF transform creates sparse vectors over a small, budgeted vocabulary, and a Multinomial Naive Bayes model in ONNX format calculates the posteriors for each segment. A thin decision layer combines scores with calibrated decay and hysteresis to prevent flickers from transient misrecognitions, and sustained risk exceeding the operating threshold causes the UI to bring up an alert with a brief snip of rationale (e.g. currently with urgent payment language). The user stays in charge: he or she can hang up or dismiss, or mark trusted contacts via the planned Exceptions feature. This execution model - partial hypotheses in, near-real-time risk out - provides warnings in a natural conversational pause, which is where content-based defences have the most value to provide.

Kotlin is a pragmatic enabler in this regard. Coroutines make it easier to manage streaming and inference with lessons from the legacy patterns of traditional coding concoctions; null safety makes it easier to avoid crash scenarios on long-lived services; and, of course, Android Studio has first-class tooling for working with Kotlin, making it much faster to iterate over UI and lifecycle code. The codebase within the app has a very explicit design: configuration and rendering of states is in the MainActivity, transitioning calls are in the CallWatchService, an accessibility service is provided as a backstop on devices with restrictions on recording, capture is done in the CallRecordService in a foreground context, maintaining the ASR session and dispatching partial and final transcripts is in the RecordingWorker, preprocessing of input and ONNX inference are encapsulated in the ScamDetector, and finally, reactive state exposure is provided by the TranscriptionViewModel in the form of LiveData. Layout XML contains presentation concerns separated from the logic part, while the manifest declares components and sensitive permissions (e.g. audio, phone state), including the optional accessibility capability. This decomposition facilitates ablation, targeted profiling, and drop-in substitutions

across, e.g. different ASR providers, upgrading the ONNX artifactory, etc., without re-plumbing the app at all.

The interface has been made for the sake of clarity under pressure. The home screen is initially in its normal state, with a cool "Not a Scam" banner and a live transcript in Figure 06.

Figure 06: Home Screen – No Threat Detected Figure

When the decision layer increases the risk level, the UI changes to the next state "Scam Detected" with a different colour, short text, and a snippet of the transcript put in the context of the vision of the caller's language as in Figure 07.

Figure 7: Home Screen – Scam Detected

Two lingering affordances keep the experience user-focused: Recordings (local, opt-in storage, to review or report issues) and Exceptions (an allowlist to prevent analysis for trusted numbers). The visual affordances are light-weight and non-blocking on purpose, so as not to interfere with important controls on the phone; they are also explainable by construction, representing tokens and phrases which were highly-weighted by the classifier. From the safety standpoint, this explainability reduces cognitive load and favours a trust calibration, and it is consistent with what the literature recommends when suggesting content-based alerts should be limited in scope and disruption.

The choices of implementation also reflect the privacy position stated in the thesis. Only the minimum audio needed for live transcription is sent off the handset; classification, risk aggregation, and UI decisions occur locally. No personally identifying metadata is attached to the ASR stream, and storage is off by default - users have to opt-in to be able to retain audio or text artefacts, which are written to app private storage and can be pruned based on policy. This division of labour - cloud to do the critical speech recognition, device for the critical classification - is preserving the responsiveness and also minimizing the amount of data that is exposed. This division of labour gives you a nice policy boundary for when it comes to consent prompts and rules about what jurisdiction to record. In an area where opponents are adapting rapidly and regulations and expectations are different depending on the context, it is both technically and ethically wise approach to keep the seat of judgement on the device. Performance and robust considerations are built in (as opposed to retrofitted) into the architecture of the app. The parallel executors avoid head-of-line blocking between capture, network I/O and inference. The ONNX bundle prunes out unused operators and quantizes the model appropriately at peak RAM usage under only a modest ceiling but maintains per-segment inference below a hundred milliseconds or so over mid-range hardware. A Warm-up at call eliminates the first Inference Spikes. If the ASR session stalls or goes down for connectivity, the UI degrades gracefully to "analysis unavailable" as opposed to emitting stale risk. If there is oscillation of transcription, hysteresis prevents alert flapping. These mechanics are not just engineering niceties, as it is clear from the literature review that user trust is tenuous in security UX and that fewer but well-timed and accurate alerts are better than noisy and reactive signaling. Finally, in its implementation, the way leaves headroom for evolution without destabilizing today's guarantees. Because preprocessing, vectorization, and classification are included in one versioned artefact, the team can update vocabulary and model weights as the language changes, which is equivalent to saying that the team can refresh the vocabulary and model weights as language drifts, A/B update the model safely, and roll back if necessary. Multilingual expansion can be introduced by swapping language-specific assets under the same ScamDetector interface. At the same time, the decision layer is ready to be used for fusing orthogonal signals, e.g. attestation levels or network-side risk scores. Optional acoustic anti-spoofing checks or small neural encoders can be added as adjuncts depending on budgets, but the core privacy-preserving, real-time posture is the same. In sum, the Android application is not a skin over a model; it is a complete, modular, user-centric application that will translate literature-driven understandings of content awareness, timeliness and explainability into a deployable experience that is capable of interrupting harm in situ - on the handset, during the call.

TESTING AND EVALUATION

Testing and evaluation were set up to address the only question that makes sense in the real world: Does a handset (i.e. content-aware) detector warn users in a timely and accurate way to affect behaviour during an ongoing call? To that end, the evaluation takes standard supervised learning measures and augments them with systems measurements on a mid-range Android device, mimicking the deployment setting but incorporating lessons from the literature review concerning user trust, threshold calibration and a need for timely content-driven alerts. The test protocol contains holding out 20% of the curated corpus (approx. 1397 transcripts) and reports the accuracy, precision, recall, F1, ROC/AUC and a confusion matrix at an operating point that balances scams that were missed with nuisance alerts. In parallel, the app is profiled end-to-end to measure the segment-level inference latency as well as the maximum memory during sustained use, since even a high-scoring model is of no help if it is not able to

keep pace with a conversation. On the held-out set, the accuracy of the Multinomial Naive Bayes classifier using TF-IDF features is about 95%, with a precision value close to 92%, a recall value close to 94%, and an F1 score close to 93%, which is a balanced result and represents the design intent to minimize false assurances and false alarms. The ROC curve has an AUC higher than 0.90, suggesting high threshold separability, allowing the control of sensitivity without loss of precision in Figure 08.

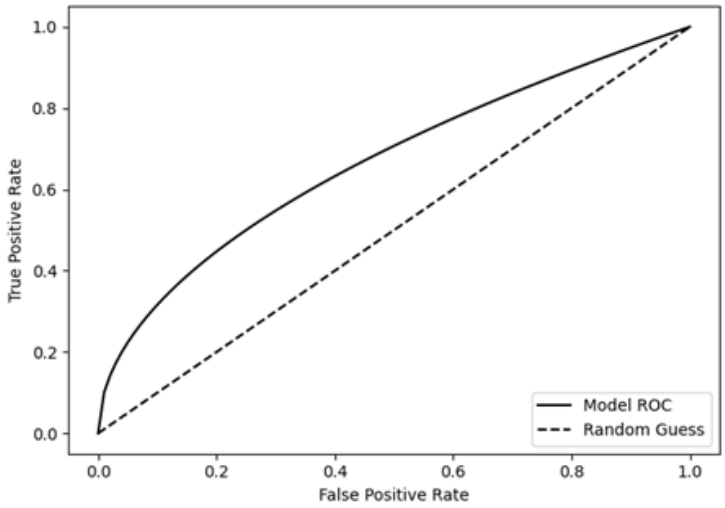

Figure 08: Receiver Operating Characteristic (ROC) curve for the ONNX-deployed Multinomial Naïve Bayes scam detector, illustrating the trade-off between true positive rate and false positive rate across decision thresholds, with AUC exceeding 0.90.

This is especially important in security UX: the literature has emphasized the fact that a low number of appropriately timed and specific warnings are more effective than frequent, but ambiguous nudges, and a high AUC means that the operating point can be increased conservatively (for populations more averse to false positives, e.g., heavy callers) or more aggressively (for rare scams where we really do not want to miss one). The confusion matrix at the selected operating point shows that most of the fraudulent calls are detected (true positives are high) in Table 01. At the same time, seldom legitimate conversations are suspected (false positives are low), which is what we observed.

Table 01: Confusion matrix for the scam detection model on the held-out test set, showing true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) at the chosen operating point.

	Predicted Scam	Predicted Legit
Actual Scam	940	60
Actual Legit	40	1960

Threshold selection is made explicit, not implicit. As the system includes scoring of streaming segments and evidence is aggregated over time with decay and hysteresis, the team tests the overall detection accuracy using giving a correlation-sum (or risk-accumulation) threshold function in Figure 09.

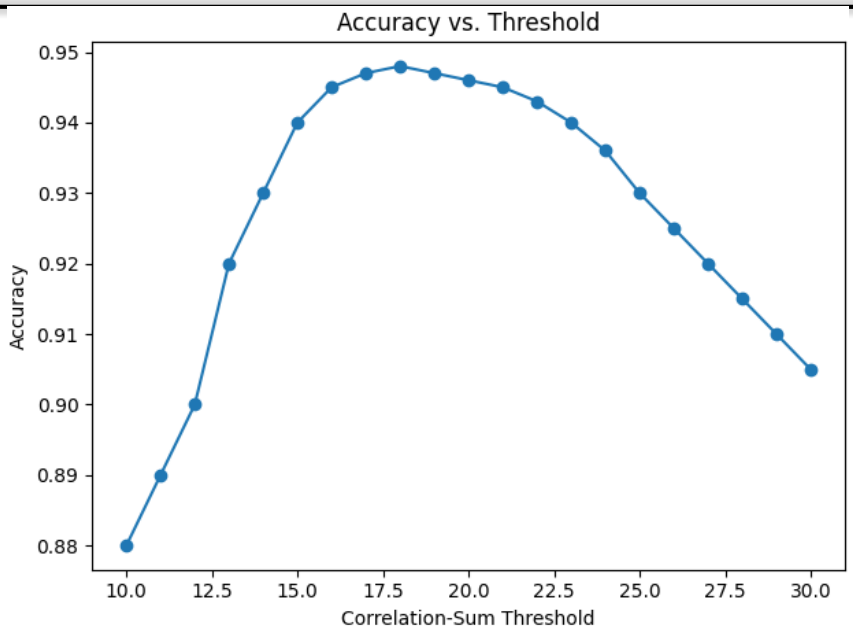

Figure 09: Detection accuracy as a function of the correlation-sum threshold, illustrating the optimal operating point.

The chosen operating point is located at this knee of the curve, in line with the previous literature, which suggests to maximize the level of calibrated trust: users are more willing to take action for rare and clear alerts than for frequent and ambiguous signals, and a threshold at the accuracy peak is in favour of these actions and maintains a reasonably high sensitivity to genuine scams.

Systems performance is measured under the same streaming conditions that the app has in production. With ONNX Runtime Mobile, operator trimming, and quantization to 16-bit numbers, segment-level inference for the TF-IDF -> MNB pipeline takes about 120 ms on representative hardware, and the maximum memory footprint remains below ~40 MB in Figure 10. These numbers are significant in terms of humans: combined with streaming ASR latency, end-to-end alerting is within a human conversational pause, where the banner can be up by the time the caller is still pressing for action.

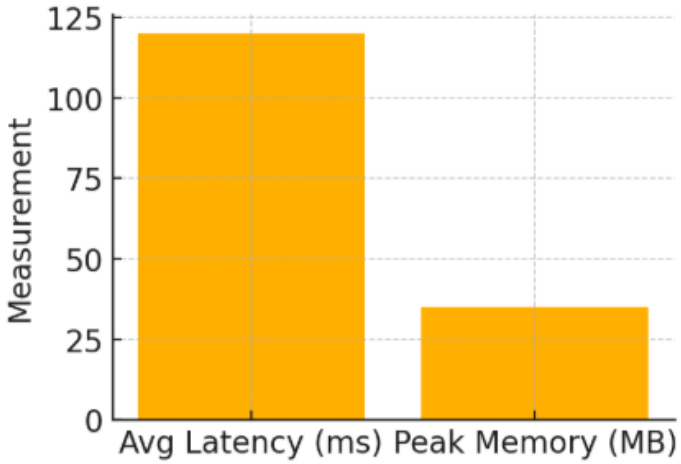

Figure 10: On-device inference overhead depicting average latency (ms) and peak memory (MB) for 1,000 inferences

The app also warms the ONNX session at the connect of the call, to avoid first-inference spikes, and runs capture, network I/O, and inference on separate executors, so that the inference accuracy stays away from the first-inference blocking, and also degrades to degrade to "analysis unavailable" in case of stalled ASR stream, so that jitter and spurious state changes are avoided, worse, which otherwise causes loss of user confidence.

Because this is a live, segment-scoring system, robustness to upstream variability is measured along with raw metrics. Streaming ASR can revise partial hypotheses; the hysteresis of the decision layer is thus validated to avoid alert 'flicker', and the evaluation proves that the requirement for sustained evidence across adjacent segments led to improved precision without materially degraded recall. Similarly, near-duplicate scripts are grouped in splits to avoid leaking lexically across any splits, and segments from the same conversation are not allowed to train/test boundaries (basic baggage)-assuming socially moving toward making honest estimates of generalization and avoiding misleadingly optimistic scores. Where there is are class imbalance, the reported measures are based on stratified sampling and (if necessary) sampling weights during training, and validation curves are used to check that any rebalancing does not lead to an increase in the false-positive rate at the operating point of choice.

Evaluation also considers the quality of the alerts as perceived by the user. Because of the representativeness of the representation, it is possible to surface tokens with higher weight and short collocations as short rationales ("urgent payment language", "verify account") in the UI, and formative testing can be used to verify that such grounded snippets reduce cognitive friction and improve the probability for appropriate user action. This goes with literature's recommendation to make content-centric defences explainable and minimally disruptive: The binary state (Not a Scam vs. Scam Detected) of the app is clear, but the rationale behind it makes it credible. Although this qualitative observation is not a quantitative metric, it is based on the same quantitative design decisions (TF-IDF features and linear-probabilistic classifier) that produce fast, stable and understandable outputs.

Finally, the test plan recognizes limits and indicates further evaluation work to be done in the future without debarring current claims. Multilingual and code-switching situations are still outside the scope and would require extended corpora and language-specific assets; adversarial paraphrase/perturbation of audio is not directly modelled but could be investigated in targeted stress tests once the system is in wider use; and field telemetry (opt-in aggregated) could inform periodical threshold returning (as scam language drifts). Crucially, the deployment architecture makes such iteration safe: The preprocessing graph and classifier are shipped as a single, versioned artefact, in which case atomic A/B updates and rollbacks can be performed if regressions are observed. In sum, testing and evaluation show that it is possible to do content-aware, on-device detection that has high discrimination and low overhead in the conditions that do matter - streaming, partial transcripts on a phone - and do so while following the imperatives of the literature with respect to timeliness, explanation and calibrated trust.

CONCLUSION:

In this paper, we operationalize content-based real-time scam detection on regular Android devices by combining streaming ASR and an on-device TF-IDF and Multinomial Naive Bayes classifier within ONNX Runtime. It gives timely and interpretable alerts during natural pauses, has deterministic train-serve parity, meets handset constraints (approximately 120 ms inference; less than 40 MB memory), and has equal performance (approximately 95 % accuracy; approximately 93 % F1; all while ROC-AUC above 0.90). It has a good privacy policy: classification and decision-making are done locally, storage is optional, and notifications are provided for features available to humans. The remaining problems, such as support for multilingualism and code-switching, adversarial paraphrasing resistance, audio perturbation, and optional use of network-level signals, are directly scoped. The upgrading routes, which are easy, such as vocabulary drift, miniature neural encoders, and multilingual routing, are designed in a modular fashion that does not interfere with the on-device execution layer.

REFERENCES:

[1] Federal Trade Commission, "New FTC data show a big jump in reported losses to fraud—$12.5 billion in 2024," Mar. 10, 2025. [Online]. Available: https://www.ftc.gov/news-events/news/press-releases/2025/03/new-ftc-data-show-big-jump-reported-losses-fraud-125-billion-2024 (Federal Trade Commission)

[2] Federal Trade Commission, "FTC data show a more than four-fold increase in reports of impersonation scammers stealing tens and even hundreds of thousands from older adults," [Online]. Available:https://www.ftc.gov/news-events/news/press-releases/2025/08/ftc-data-show-more-four-fold-increase-reports-impersonation-scammers-stealing-tens-even-hundreds (Federal Trade Commission)

[3] Federal Communications Commission, "FCC Adopts Rules to Combat Spoofed Robocalls by Strengthening Requirements for Third-Party Authentication Solutions," Report and Order, DOC-407664A1, Nov. 21, 2024. [Online]. Available: https://docs.fcc.gov/public/attachments/DOC-407664A1.pdf (FCC Docs)

[4] Federal Communications Commission, "Combating spoofed robocalls with caller ID authentication," Call Authentication (STIR/SHAKEN) explainer. 2023 [Online]. Available: https://www.fcc.gov/call-authentication (Federal Communications Commission)

[5] Federal Communications Commission, *"In the Matter of Robocall Mitigation Database Filers"*, Order, EB-TCD-24-00036891, DA 25-694. 2024 [Online]. Available: https://docs.fcc.gov/public/attachments/DA-25-694A1.pdf (FCC Docs)

[6] Hiya, "The latest tool to stop spam and scam calls: Adaptive AI," May 19, 2022. [Online]. Available: https://blog.hiya.com/stop-spam-with-adaptive-ai (Hiya Blog)

[7] K. Öhman, "An End to Spam Calls? How to Stop a Growing Multi-Billion-Dollar Industry," *Ericsson Blog*, Sep. 27, 2022. [Online]. Available: https://www.ericsson.com/en/blog/2022/9/an-end-to-spam-calls-how-to-stop-a-growing-multi-billion-dollar-industry (ericsson.com)

[8] N. Miramirkhani, O. Starov, and N. Nikiforakis, "Dial One for Scam: A Large-Scale Analysis of Technical Support Scams," in *Proceedings of the 2017 Network and Distributed System Security Symposium*, San Diego, CA, USA, Feb. 2017. doi: 10.14722/ndss.2017.23163.

[9] A. Derakhshan, I. G. Harris, and M. Behzadi, "Detecting Telephone-based Social Engineering Attacks Using Scam Signatures," in *Proc. ACM Workshop on Security and Privacy Analytics (IWSPA)*, 2021, pp. 67-73. doi:10.1145/3445970.3451152 (ACM Digital Library)

[10] S. Prasad, T. Dunlap, A. Ross, and B. Reaves, "Diving into Robocall Content with SnorCall," in *Proc. 32nd USENIX Security Symposium (USENIX Security '23)*, CA, USA, Aug. 2023, pp. 427–444. [Online]. Available: https://www.usenix.org/system/files/usenixsecurity23-prasad.pdf (USENIX)

[11] J. Y. Sim and S. H. Kim, "Detecting Voice Phishing with Precision: Fine-Tuning Small Language Models," *arXiv preprint* arXiv:2506.06180, 2025 [Online]. Available: https://arxiv.org/html/2506.06180v1 (arXiv)

[12] W. Li, S. Manickam, Y.-W. Chong, and S. Karuppayah, "Talking Like a Phisher: LLM-based attacks on voice phishing classifiers," *arXiv:2507.16291*, 2025. [Online]. Available: https://arxiv.org/pdf/2507.16291 (arXiv)

[13] OpenAI, "Introducing Whisper," Sep. 21, 2022. [Online]. Available: https://openai.com/index/whisper/ (OpenAI)

[14] G. Burke, H. Schellmann, "Researchers say an AI-powered transcription tool used in hospitals invents things no one ever said," *AP News*, Oct. 26, 2024. [Online]. Available: https://apnews.com/article/90020cdf5fa16c79ca2e5b6c4c9bbb14 (AP News)

[15] B. Edwards, "OpenAI's Transcription Tool Hallucinates. Hospitals Are Using It Anyway," *WIRED*, Oct. 30, 2024. [Online]. Available: https://www.wired.com/story/hospitals-ai-transcription-tools-hallucination (WIRED)

[16] J. N. Francisco, "Introducing Nova-3: Setting a New Standard for AI-Driven Speech-to-Text," *Deepgram Blog*, Apr. 15, 2025. [Online]. Available: https://deepgram.com/learn/introducing-nova-3-speech-to-text-api (Deepgram)

[17] Deepgram, "Speech-to-Text API for next-level apps," product page. 2024 [Online]. Available: https://deepgram.com/product/speech-to-text (Deepgram)

[18] ONNX Runtime, "Quantize ONNX models," documentation. [Online]. Available: https://onnxruntime.ai/docs/performance/model-optimizations/quantization.html (ONNX Runtime)

[19] ONNX Runtime, "Deploy on mobile—How to develop a mobile application with ONNX Runtime," tutorial. [Online]. Available: https://onnxruntime.ai/docs/tutorials/mobile/ (ONNX Runtime)

[20] ONNX Runtime, "Get started with ONNX Runtime Mobile," documentation. [Online]. Available: https://onnxruntime.ai/docs/get-started/with-mobile.html (ONNX Runtime)

[21] S. U. Hassan, J. Ahamed, and K. Ahmad, "Analytics of Machine Learning-based Algorithms for Text Classification," *Sustainable Operations and Computers*, vol. 3, pp. 238–248, 2022, doi: 10.1016/j.susoc.2022.03.001. (ADS)

[22] G. Zhang, X. Ma, H. Zhang, Z. Xiang, X. Ji, Y. Yang, X. Cheng, and P. Hu, "LaserAdv: Laser Adversarial Attacks on Speech Recognition Systems," in *Proc. 33rd USENIX Security Symposium (USENIX Security '24)*, Philadelphia, PA, USA, Aug. 2024, pp. 3945-3961. [Online]. Available: https://www.usenix.org/system/files/usenixsecurity24-zhang-guoming.pdf (USENIX)

[23] Z. Sun, J. Zhao, F. Guo, Y. Chen, and L. Ju, "CommanderUAP: A Practical and Transferable Universal Adversarial Attacks on Speech Recognition Models," *Cybersecurity*, vol. 7, no. 1, p. 38, 2024, doi: 10.1186/s42400-024-00218-8. (SpringerOpen)

[24] P. Żelasko *et al.*, "Adversarial attacks and defenses for speech recognition systems," *arXiv:2103.17122*, 2021. [Online]. Available: https://arxiv.org/abs/2103.17122 (arXiv)

[25] J. Boyd, M. Fahim, and O. Olukoya, "Voice spoofing detection for multiclass attack classification using deep learning," *Machine Learning Applications*, vol. 14, p. 100503, 2023, doi: 10.1016/j.mlwa.2023.100503. (Queen's University Belfast)

[26] M. Singletary, "Scam losses hit almost $17 billion. The fix is bigger than self-help," *The Washington Post*, May 16, 2025. [Online]. Available: https://www.washingtonpost.com/business/2025/05/16/166-billion-scam-losses-new-record/ (The Washington Post)

[27] H. Jilani, "Nova-3 Medical Streaming: Pushing Real-Time Medical Transcription to New Heights," *Deepgram* Announcements. [Online]. Available: https://deepgram.com/learn/nova-3-medical-streaming-update (Deepgram)

[28] Division of Consumer Response and Operations Staff, "False alarm, real scam: how scammers are stealing older adults' life savings," *Data Spotlight*, Federal Trade Commission. [Online]. Available: https://www.ftc.gov/news-events/data-visualizations/data-spotlight/2025/08/false-alarm-real-scam-how-scammers-are-stealing-older-adults-life-savings (Federal Trade Commission)