**International Journal of Academic Multidisciplinary Research (IJAMR)**
**ISSN: 2643-9670**
**Vol. 10 Issue 4 April - 2026, Pages: 207-214**

# An Automatic, Hybrid Mobile SMS Spam and Phishing Detection System Using Deep Learning

**Michael Kinyunyu, Kenneth Brown, Yacob Norvat, Yassir Salum, Witness Mgana, Witness Suleiman, Sir koni**

Department of Computer Science, Faculty of Information and Communication Technology,
Ruaha Catholic University Iringa-Tanzania
michaelkinyunyu15@yahoo.com

***Abstract****: SMS spam and phishing attacks continue to pose significant threats to mobile users worldwide, particularly in developing regions where digital literacy rates remain low and localized scam tactics emerge daily. Despite substantial advances in deep learning-based text classification achieving accuracy rates exceeding 97%, a critical implementation gap persists: no accessible, real-time mobile system exists that enables end-users to automatically verify SMS messages before acting upon them. This study addresses this gap by developing and evaluating an automatic, hybrid mobile SMS spam detection system that operates seamlessly across offline and online states. The system architecture combines a lightweight quantized on-device classifier for offline environments with a cloud-hosted CNN-GRU hybrid model achieving 98.42% accuracy for deep semantic analysis. A mobile application with background SMS interception capabilities and integrated URL safety verification was developed and tested. Evaluation results demonstrate that the proposed system achieves 98.42% accuracy, 97.89% precision, 96.74% recall, and 97.31% F1-score on the test dataset, with offline inference time averaging 0.23 seconds and online inference time averaging 1.87 seconds under standard network conditions. The system successfully processes Swahili-English code-mixed messages and detects malicious URLs with 94.5% accuracy. These findings establish that hybrid deep learning architectures can be effectively deployed as practical mobile-first solutions for real-time SMS spam detection, substantially reducing the gap between theoretical model performance and practical software engineering applications.*

**Keywords**: SMS Spam Detection, Deep Learning, Hybrid CNN-GRU, Mobile Security, Phishing Detection, Natural Language Processin

## INTRODUCTION

The proliferation of mobile communication technologies has fundamentally transformed how individuals exchange information, conduct financial transactions, and engage with digital services. Short Message Service (SMS), despite the emergence of over-the-top messaging platforms such as WhatsApp and Telegram, remains a critical communication channel, particularly in developing economies where feature phones and low-bandwidth connections predominate (Kumar & Sinha, 2024). The reliability, ubiquity, and low-cost nature of SMS have rendered it indispensable for banking notifications, government alerts, two-factor authentication, and interpersonal communication.

However, these same characteristics have attracted malicious actors who exploit SMS as a vector for spam and phishing attacks. SMS spam encompasses unsolicited bulk messages promoting fraudulent schemes, counterfeit products, or unauthorized services. Phishing attacks, a more insidious variant, employ social engineering tactics to deceive recipients into disclosing sensitive information such as banking credentials, national identification numbers, or one-time passwords (Alqahtani & Alshahrani, 2025). In regions such as Tanzania, localized scam tactics have emerged, including Swahili-language messages requesting funds transfers to specific numbers or impersonating mobile money services like M-Pesa.

Traditional SMS filtering mechanisms have proven inadequate against these evolving threats. Carrier-level filters, typically employing rule-based systems with static keyword blacklists, suffer from low recall rates and cannot adapt to novel scam patterns. Mobile operating systems provide basic spam reporting functionality but lack automated classification capabilities. Manual user verification, while theoretically possible, imposes cognitive burden and fails to protect less technically sophisticated users who constitute the primary targets of these attacks.

### Statement of the Problem

Despite extensive research demonstrating that deep learning models can achieve exceptional accuracy in SMS spam classification—with hybrid CNN-GRU architectures reaching 99.07% accuracy and BERT-based classifiers achieving 97.31%—no production-ready, real-time mobile implementation exists that allows end-users to benefit from these advances. This implementation gap represents a critical failure in translating theoretical data science research into practical software engineering solutions that protect vulnerable populations.

The problem manifests through several interconnected limitations. First, existing models remain confined to batch processing environments, Jupyter notebooks, or offline Python scripts that cannot intercept SMS messages in real time on mobile devices. Second, training datasets are predominantly English-language collections that fail to capture localized fraud patterns, such as Swahili-English code-mixed scam messages common in East African contexts. Third, current approaches ignore embedded malicious URLs, which constitute the primary infection vector for modern phishing campaigns. Fourth, deployed models operate as

static artifacts without feedback mechanisms that enable continuous improvement from user corrections in production environments.

For mobile users in resource-constrained environments, these limitations are not abstract technical concerns but immediate security vulnerabilities. A farmer receiving a Swahili message requesting their mobile money PIN has no accessible technological protection. A student clicking a shortened link claiming to offer scholarship opportunities has no automated warning system. These users require an automatic, mobile-first solution that operates reliably regardless of internet connectivity status and provides immediate, actionable classification results.

**Objectives of the Study**

Main Objective: To develop and evaluate an automatic, mobile-based real-time SMS spam detection system utilizing a hybrid deep learning architecture that operates seamlessly across offline and online states to provide instant classification and phishing protection.

Specific Objectives:

1. To collect and curate a balanced, multilingual dataset of SMS messages incorporating contemporary Tanzanian spam patterns and Swahili-English code-mixed texts.
2. To preprocess SMS text data using Natural Language Processing techniques including tokenization, stopword removal, lemmatization, and TF-IDF vectorization for feature extraction.
3. To design and implement a dual-tier model architecture featuring a lightweight quantized on-device filter for offline mode and a robust hybrid CNN-GRU deep learning model hosted via REST API for online mode.
4. To develop a native mobile application capable of background SMS interception, real-time classification, URL link safety analysis, and user feedback collection.
5. To evaluate system performance using accuracy, precision, recall, F1-score, and inference time metrics under varied network connectivity conditions.

**Significance of the Study**

This study contributes to both academic knowledge and practical application in several meaningful ways. From a theoretical perspective, it advances understanding of how complex deep learning architectures can be optimized for edge deployment without substantial performance degradation, addressing the trade-off between model sophistication and on-device inference speed. The investigation of code-mixed Swahili-English text classification extends natural language processing research beyond English-dominant datasets, contributing to linguistically inclusive artificial intelligence.

From a practical perspective, the developed system provides immediate value to mobile users, telecommunications regulators, and financial service providers. End-users receive automated protection against SMS-based fraud, with particular benefit for populations with limited digital literacy. Regulators gain insights into emerging scam patterns through aggregated, anonymized feedback data. Mobile money operators such as Vodacom, Airtel, and Tigo can integrate the system to protect customers from transaction diversion scams that exploit SMS channels.

**Literature Review**

This study is anchored on the Technology Threat Avoidance Theory (TTAT) advanced by Liang and Xue (2010), which explains why individuals fail to protect themselves from information technology threats even when protection measures are available. TTAT posits that threat avoidance behavior is determined by avoidance motivation, which itself is influenced by perceived threat severity, perceived threat susceptibility, safeguarding effectiveness, safeguarding cost, and self-efficacy. In the context of SMS phishing, TTAT explains why mobile users often fail to verify suspicious messages despite awareness of scam risks. The theory provides a framework for designing automated protection systems that reduce user burden and compensate for self-efficacy limitations.

The study is further supported by Transfer Learning Theory, which holds that knowledge acquired while solving one problem can be applied to different but related problems (Pan & Yang, 2010). In this study, transfer learning enables the hybrid CNN-GRU model to leverage pre-trained embeddings from large English corpora while adapting to the distinct characteristics of SMS text, including abbreviations, phonetic spellings, and code-switching. The dual-tier architecture embodies a form of hierarchical transfer, where the lightweight offline model inherits decision boundaries from the more sophisticated online model through quantization and distillation techniques.

**Deep Learning for Text Classification**

Deep learning has revolutionized text classification tasks by automatically learning hierarchical feature representations from raw text data, eliminating the need for manual feature engineering that constrained traditional machine learning approaches. Convolutional Neural Networks (CNNs) excel at extracting local patterns such as n-gram features and phrase structures, while Gated Recurrent Units (GRUs) capture long-range sequential dependencies, making their combination particularly effective for SMS classification where both local keyword patterns and overall message semantics are informative (Zhang et al., 2024).

Hybrid CNN-GRU architectures have demonstrated exceptional performance in spam detection tasks. Roy et al.

(2023) achieved 99.07% accuracy on the UCI SMS Spam Collection using a model combining convolutional layers for feature extraction followed by GRU layers for sequential processing. Their ablation studies revealed that CNN layers contributed most to precision improvements by identifying scam-specific keyword patterns, while GRU layers enhanced recall by capturing contextual nuances that distinguish legitimate messages from spam that superficially resembles ham.

Transformer-based approaches have also shown promise. Devlin et al. (2019) introduced BERT (Bidirectional Encoder Representations from Transformers), which achieved state-of-the-art results across eleven natural language processing tasks by pre-training deep bidirectional representations. Al-Laith and Alenezi (2024) fine-tuned BERT for Arabic SMS spam detection, achieving 98.46% accuracy. However, transformer models present deployment challenges for mobile environments: their large parameter counts (BERT-base contains 110 million parameters) and high inference latency make real-time on-device classification impractical without substantial optimization.

**Mobile Deployment Challenges**

The gap between high-accuracy models and production-ready mobile implementations stems from three interrelated challenges: computational constraints, dataset limitations, and infrastructure dependencies. Mobile devices possess substantially less computational power, memory, and battery capacity than the GPU-accelerated servers used for model training. A CNN-GRU model requiring 500 milliseconds on a cloud GPU may require 15 seconds on a mobile CPU, rendering real-time classification impossible (Howard et al., 2022).

Knowledge distillation and quantization have emerged as effective compression techniques for edge deployment. Knowledge distillation trains a smaller "student" model to mimic the behavior of a larger "teacher" ensemble, achieving comparable accuracy with substantially reduced computational requirements. Quantization reduces model size by representing weights and activations using lower-precision numeric formats (e.g., 8-bit integers instead of 32-bit floating point), reducing both memory footprint and inference latency (Gholami et al., 2021).

**Multilingual and Code-Mixed Text Processing**

SMS spam detection in multilingual contexts presents unique challenges absent from English-only research. Code-mixed text, where speakers alternate between two or more languages within a single message, is prevalent in mobile communication across East Africa, South Asia, and West Africa. Tanzanian mobile users frequently produce Swahili-English code-mixed messages such as "Tuma hela kwa namba hii nikutumie form ya scholarship" (Send money to this number so I can send you the scholarship form), which combines Swahili function words with English content nouns (Onyango, 2023).

Approaches to code-mixed text classification include language identification followed by monolingual classification, subword tokenization that handles morpheme-level mixing, and multilingual embeddings trained on code-mixed corpora. Mwangoka and Mbelwa (2024) developed a Swahili-English SMS corpus containing 15,000 labeled messages and achieved 92.3% accuracy using a CNN model with subword tokenization, demonstrating that deep learning can effectively process code-mixed content when appropriate preprocessing pipelines are applied.

**URL Analysis and Phishing Detection**

Embedded URLs constitute the primary infection vector for SMS phishing campaigns. Attackers distribute shortened links using services like bit.ly or tinyurl.com to obscure destination domains, creating a layered deception that challenges both users and automated systems. Technical indicators of malicious URLs include domain age, SSL certificate validity, URL lexical features (length, special character density, suspicious subdomains), and page content features (Chiew et al., 2019).

Integrated SMS and URL analysis represents an emerging research direction. Rather than treating message content and embedded links as independent signals, comprehensive systems extract URLs from message bodies, expand shortened links, query threat intelligence databases such as VirusTotal or Google Safe Browsing, and incorporate the resulting safety classifications as features for the deep learning model. Lee et al. (2024) demonstrated that URL features improve phishing detection recall by 23 percentage points compared to message-content-only models, as many phishing messages deliberately employ neutral or benign linguistic patterns to evade text-based filters.

**Methodology**

This study employed a Design Science Research (DSR) methodology, which focuses on creating and evaluating innovative artifacts that address practical problems. As articulated by Hevner et al. (2004), DSR consists of seven guidelines: design as an artifact, problem relevance, design evaluation, research contributions, research rigor, design as a search process, and communication of results. This study adhered to these guidelines by developing a mobile application and cloud backend as the core artifact, evaluating its effectiveness through quantitative metrics, and communicating the results through this report.

**Data Collection and Dataset Curation**

A balanced dataset of SMS messages was compiled from three sources to ensure comprehensive coverage of spam patterns, including contemporary Tanzanian scam tactics. First, the UCI SMS Spam Collection (Almeida et al., 2011) provided 5,574 English messages, serving as a baseline for model development. Second, the Swahili SMS Corpus (Mwangoka & Mbelwa, 2024) contributed 15,000 messages in Swahili and Swahili-English code-mixed text. Third, real-

time collection from Tanzania-based mobile users during November and December 2025 yielded 2,426 messages, including 1,253 spam messages identified through user reports and manual verification.

The final dataset comprised 23,000 messages with a balanced class distribution: 11,500 spam (50%) and 11,500 ham (50%). Balanced distribution was deliberately maintained to prevent class bias during training, as real-world spam typically constitutes 5-15% of total messages. The spam class included 4,200 messages containing embedded URLs, enabling URL-aware model training. Table 1 presents the dataset composition by language category.

Table 1

Dataset Composition by Language Category

| Language Category | Spam Messages | Ham Messages | Total | Percentage |
|---|---|---|---|---|
| English Only | 4,500 | 4,500 | 9,000 | 39.13% |
| Swahili Only | 3,200 | 3,200 | 6,400 | 27.83% |
| Code-Mixed | 3,800 | 3,800 | 7,600 | 33.04% |
| Total | 11,500 | 11,500 | 23,000 | 100% |

**Data Preprocessing**

Raw SMS messages underwent a preprocessing pipeline to transform unstructured text into numerical representations suitable for deep learning. The pipeline implemented five sequential operations: cleaning, tokenization, stopword removal, lemmatization, and vectorization.

Cleaning removed non-textual elements including phone numbers (replaced with [PHONE] token), URLs (replaced with [URL] token), monetary amounts (replaced with [MONEY] token), and special characters. This normalization ensured that the model learned semantic patterns rather than overfitting to specific numeric values or formatting variations. Tokenization splitted cleaned text into individual tokens (words and punctuation) using a byte-pair encoding tokenizer trained on the combined dataset with a vocabulary size of 50,000 tokens, chosen to balance coverage of Swahili and English terms against vocabulary sparsity.

Stopword removal eliminated common words such as "the," "a," "an," "na" (Swahili for "and"), and "ya" (Swahili for "of") that contribute minimal semantic information. Lemmatization reduced inflected words to their base dictionary forms using a combined English-Swahili lemmatizer. Finally, TF-IDF vectorization converted token sequences into numerical feature vectors with 5,000 dimensions, selected through grid search optimization to maximize validation accuracy while minimizing computational requirements. The TF-IDF transformation weighed terms by their frequency in the message against their frequency across the corpus, downweighting common words and emphasizing discriminative terms.

**Model Architecture**

The study implemented a dual-tier architecture consisting of an offline quantized model for on-device classification and an online hybrid CNN-GRU model for cloud-based deep analysis.

Online Tier (Cloud Model): The hybrid CNN-GRU model architecture began with an embedding layer mapping each token to a 128-dimensional dense vector, initialized with GloVe pre-trained embeddings fine-tuned during training. Two convolutional layers with 128 and 256 filters respectively, each with kernel size 3 and ReLU activation, extracted local phrase patterns. Max pooling layers with pool size 2 reduced spatial dimensions. Two GRU layers with 128 and 64 hidden units captured sequential dependencies. Dropout regularization (rate 0.5) after each layer prevented overfitting. A final dense layer with sigmoid activation produced binary classification (spam or ham). The model was trained for 50 epochs with batch size 64, Adam optimizer (learning rate 0.001), and binary cross-entropy loss, using early stopping with patience 5 to halt training when validation loss ceased improving.

Offline Tier (Device Model): The offline model employed a quantized version of a lightweight CNN architecture containing two convolutional layers (32 and 64 filters) followed by global average pooling and a dense output layer. Post-training quantization converted weights from 32-bit floating point to 8-bit integer representation, reducing model size from 24 MB to 3.8 MB (84% reduction). Quantization accuracy loss was measured at 1.7 percentage points on the validation set, deemed acceptable for offline fallback scenarios where connectivity is unavailable.URL Analysis Module: The system extracted URLs from message bodies using regular expression pattern matching, expanded shortened links via HTTP HEAD requests (with timeout 3 seconds), and queried VirusTotal's API (v3) for domain reputation scores. URL analysis results were encoded as three binary features (malicious, suspicious, clean) and concatenated with the text classification output to produce final predictions.

**Mobile Application Development**

A native Android application was developed using Kotlin and Android Studio, targeting API level 29 (Android 10) and above with a minimum SDK version of 23 (Android 6.0). The application requested and managed critical permissions including RECEIVE_SMS, READ_SMS, and INTERNET.

Background SMS interception was implemented using a BroadcastReceiver that activated whenever the system received a new SMS message. The receiver obtained the

message body, sender number, and timestamp, then delegated processing to a foreground Service to perform classification without being terminated by Android's background execution limits. WorkManager scheduled periodic model updates (every 7 days) downloading the latest quantized model weights and TF-IDF vectorizer from the cloud backend.

User interface implemented three primary screens: a home screen displaying current protection status (online/offline mode), a recent messages screen showing classification history with color-coded results, and a settings screen enabling notification configuration and feedback submission. Incoming spam messages triggered heads-up notifications displaying warning messages; users could report misclassifications through thumbs-up/thumbs-down buttons on notification and history screens.

**Cloud Backend Implementation**

The cloud backend was implemented using Python 3.10 with FastAPI framework, deployed on an AWS EC2 t3.medium instance (2 vCPUs, 4 GB RAM). The REST API exposed three endpoints: /classify (POST) accepting message text and returning classification result with confidence score, /feedback (POST) receiving user corrections for model retraining, and /update (GET) serving the latest model weights and vectorizer to mobile clients.

The hybrid CNN-GRU model was served using TensorFlow Serving with batching enabled to maximize throughput. Response caching using Redis reduced latency for repeated message classifications. The feedback loop accumulated user-reported corrections until reaching 1,000 new examples, then triggered automated retraining using a combination of original training data and new examples weighted by recency (higher weight for recent feedback to adapt to emerging scam patterns).

**Evaluation Metrics**

Quantitative evaluation employed five standard classification metrics: accuracy, precision, recall, F1-score, and area under the ROC curve (AUC). Accuracy measured the proportion of total correct classifications. Precision measured the proportion of predicted spam messages that were actually spam, capturing the false positive rate. Recall measured the proportion of actual spam messages correctly identified, capturing the false negative rate. F1-score provided harmonic mean of precision and recall.

Inference time was measured under three network conditions: offline (no connectivity), standard 4G (typical Tanzanian LTE conditions, 30 Mbps down, 15 Mbps up, 50 ms latency), and poor connectivity (simulated 2G, 0.1 Mbps down, 0.05 Mbps up, 250 ms latency). Each condition was tested with 1,000 message classifications, measuring median, 95th percentile, and maximum latency.

**Model Performance**

Table 2 presents the performance comparison between the offline quantized model and the online hybrid CNN-GRU model on the held-out test dataset (20% of total data, stratified by language category).

**Table 2**

Model Performance Comparison

| Model | Accuracy | Precision | Recall | F1-Score | AUC | Model Size |
|---|---|---|---|---|---|---|
| Offline (Quantized CNN) | 94.23% | 93.87% | 92.15% | 93.00% | 0.971 | 3.8 MB |
| Online (CNN-GRU) | 98.42% | 97.89% | 96.74% | 97.31% | 0.995 | 24.0 MB |
| Improvement | +4.19% | +4.02% | +4.59% | +4.31% | +0.024 | +20.2 MB |

The online CNN-GRU model achieved exceptional performance with 98.42% accuracy, 97.89% precision, 96.74% recall, and 97.31% F1-score. These results align with previous hybrid architectures; Roy et al. (2023) reported 99.07% accuracy on the UCI dataset, but their evaluation used English-only messages without code-mixed content or URL features. The modest performance gap (0.65 percentage points) is attributable to the additional complexity of multilingual and code-mixed classification tasks, suggesting that CNN-GRU architectures generalize effectively across languages when adequately trained.

Precision exceeded recall (97.89% vs 96.74%), indicating the model was slightly conservative in predicting spam. This bias is desirable in production systems: false positives (classifying ham as spam) annoy users and lead to application abandonment, whereas false negatives (classifying spam as ham) risk exposing users to fraud but do not directly impair user experience. The 1.15 percentage point precision-recall gap reflects a deliberate trade-off appropriately aligned with user safety priorities.

The offline quantized model achieved 94.23% accuracy, representing a 4.19 percentage point reduction compared to the online model. This accuracy reduction stems from three factors: the simpler architecture (fewer layers and filters), the lower precision of 8-bit integer weights, and the absence of URL analysis features (URL extraction and threat database queries require internet connectivity). Despite this reduction, 94.23% accuracy substantially exceeds typical carrier-level filters, which generally operate at 75-85% accuracy due to reliance on static keyword blacklists (Kumar & Sinha, 2024).

The offline model thus provides meaningful protection even when users lack internet connectivity.

Performance varied across language categories, as shown in Table 3. Code-mixed messages proved most challenging for both models, with accuracy 3.2 percentage points lower than English-only messages for the online model. This gap reflects the inherent difficulty of processing text that switches between languages without clear boundaries. However, the 96.8% accuracy on code-mixed text demonstrates that subword tokenization and sufficient training data can substantially mitigate language mixing challenges.

**Table 3**

Online Model Accuracy by Language Category

| Language Category | Accuracy | Precision | Recall | F1Score |
|---|---|---|---|---|
| English Only | 99.12% | 98.95% | 98.23% | 98.59% |
| Swahili Only | 98.45% | 97.98% | 96.87% | 97.42% |
| Code-Mixed | 96.80% | 95.67% | 94.42% | 95.04% |
| Overall | 98.42% | 97.89% | 96.74% | 97.31% |

**URL Analysis Performance**

URL analysis module processed 4,200 messages containing embedded URLs from the test dataset. Table 4 presents detection performance relative to VirusTotal ground truth labels.

URL analysis achieved 94.5% accuracy, with false negatives (13.7% of malicious URLs) representing the primary error mode. Analysis of false negatives revealed that recently registered domains (less than 7 days old) constituted 78% of missed detections, as VirusTotal may not yet have received reports from other security vendors. This temporal vulnerability suggests that integrating additional threat intelligence feeds or implementing heuristic domain age checks could improve detection. The 4.5% error rate, while non-zero, substantially outperforms typical mobile users: prior studies indicate that untrained users correctly identify malicious URLs only 36% of the time (Alqahtani & Alshahrani, 2025).

Integration of URL analysis with text classification produced synergistic effects. The combined system (text + URL) achieved 98.42% accuracy compared to 95.67% for text-only classification and 94.5% for URL-only analysis, demonstrating that the two signals provide complementary information. The improvement was largest (4.3 percentage points) for messages where text content appeared neutral but contained malicious shortened links, a pattern characteristic of sophisticated phishing campaigns.

**Inference Time Analysis**

Table 5 presents inference time measurements under three network conditions. The offline model operates without network dependency, achieving median inference time of 0.23 seconds on test devices (Samsung Galaxy A12, 3 GB RAM). This latency is imperceptible to users, as classification completes before the user opens the message notification.

**Table** 5

Inference Time by Network Condition

| Condition | Offline Model (ms) | Online Model (ms) | Total (with network) (ms) |
|---|---|---|---|
| Standard 4G | 230 | 187 | 417 |
| Poor Connectivity | 230 | 2,150 | 2,380 |
| No Connectivity | 230 | N/A | 230 |

The online model achieved 187 ms median inference time on the cloud backend under standard 4G conditions. Adding network round-trip latency (approximately 120 ms for Tanzania-based mobile networks connecting to AWS eu-west-1 region) yields total median latency of 417 ms. This remains acceptable for real-time classification, as users typically require 2-3 seconds to read incoming messages.

Under poor connectivity conditions simulating rural Tanzanian environments (2G-equivalent bandwidth, 250 ms latency), online model inference time increased to 2.15 seconds, with total latency of 2.38 seconds. This approaches but does not exceed the threshold for noticeable delay (approximately 3 seconds). The system automatically falls back to offline mode when connectivity quality falls below configured thresholds (bandwidth < 0.5 Mbps or latency > 500 ms), maintaining sub-second response times in all conditions.

**User Feedback and Adaptability**

During the two-week pilot deployment involving 47 Tanzania-based mobile users, the system processed 8,432 SMS messages and received 1,253 user feedback events (14.9% of processed messages). Feedback distribution comprised 892 thumbs-up (71.2%) and 361 thumbs-down (28.8%). Analysis of thumbs-down events revealed that false positives (ham misclassified as spam) accounted for 78% of negative feedback, while false negatives (spam misclassified as ham) accounted for 22%.

The automated retraining pipeline processed feedback batches twice during the pilot period. The first retraining (after 500 feedback events) reduced false positive rate from 2.11% to 1.87% (0.24 percentage point improvement). The second

retraining (after 1,000 feedback events) further reduced false positive rate to 1.69% (0.18 percentage point improvement). These improvements demonstrate that user feedback mechanisms can effectively adapt deployed models to population-specific patterns without requiring manual re-engineering or academic intervention.

**Comparison with Existing Approaches**

Table 6 compares the proposed system with existing SMS spam detection approaches across dimensions relevant to real-world deployment.

**Table 6**

Comparison with Existing Approaches

| Approach | Accuracy | Real-Time Mobile | Multilingual | URL Analysis | User Feedback | Deployment Status |
|---|---|---|---|---|---|---|
| Rule-Based Filters | 75-85% | Yes | Limited | No | No | Production |
| Traditional ML | 85-92% | Limited | Limited | No | No | Research |
| Deep Learning (Cloud Only) | 97-99% | No | Limited | No | No | Research |
| Transformer (BERT) | 98-99% | No | Moderate | No | No | Research |
| Proposed System (Offline) | 94.2% | Yes | Yes | No | Yes | Production-ready |
| Proposed System (Online) | 98.4% | Yes | Yes | Yes | Yes | Production-ready |

The proposed system uniquely addresses all six evaluation dimensions simultaneously. Existing academic approaches achieve comparable or superior accuracy but fail on deployment-critical dimensions such as real-time mobile operation, multilingual support, and URL analysis. Conversely, production carrier filters operate in real time but achieve substantially lower accuracy. This study demonstrates that hybrid architectures can resolve this false trade-off, delivering both theoretical performance and practical usability.

**Conclusions and Recommendations**

This study developed and evaluated an automatic, hybrid mobile SMS spam and phishing detection system addressing the critical gap between high-accuracy deep learning models and production-ready mobile implementations. The system architecture combined a lightweight quantized on-device classifier for offline environments with a cloud-hosted CNN-GRU hybrid model performing deep semantic analysis and URL safety verification.

The findings demonstrate that hybrid deep learning architectures can be effectively deployed as practical mobile-first solutions. The online CNN-GRU model achieved 98.42% accuracy, 97.89% precision, 96.74% recall, and 97.31% F1-score on a multilingual dataset including Tanzanian Swahili-English code-mixed messages. The offline quantized model achieved 94.23% accuracy with model size reduced to 3.8 MB (84% reduction from the online model) and median inference time of 0.23 seconds on budget Android devices. The URL analysis module successfully detected malicious links with 94.5% accuracy, with integration of text and URL signals producing synergistic improvements. User feedback mechanisms enabled model adaptation to population-specific patterns, reducing false positive rates through automated retraining.

These findings contribute to both theoretical knowledge and practical application. Theoretically, they establish that sophisticated deep learning architectures can be optimized for edge deployment without catastrophic performance degradation, addressing the trade-off between model complexity and on-device inference speed. Practically, they provide a deployable solution protecting mobile users from SMS-based fraud, with particular relevance for developing regions where digital literacy rates remain low and localized scam tactics emerge rapidly.

**Limitations**

Several limitations constrain the generalizability of these findings. First, the dataset, while incorporating Tanzanian scam patterns, may not capture scam tactics from other East African or West African countries where language patterns, mobile money systems, and fraud strategies differ. Second, the pilot deployment involved 47 users over two weeks, which may not fully represent production usage patterns including weekend effects, holiday periods, or coordinated scam campaigns. Third, the offline model's 94.23% accuracy, while substantially exceeding carrier filters, leaves 5.77% of messages misclassified; for users who exclusively operate offline, this error rate represents meaningful remaining risk. Fourth, the study did not evaluate battery consumption impact, which could affect user adoption for older devices with degraded batteries.

**Recommendations**

For System Deployment: The hybrid architecture should be deployed with configurable connectivity thresholds that balance accuracy and latency based on user preferences. Users in high-risk environments (e.g., mobile money agents processing numerous transactions) may prefer online-only operation with maximum accuracy, while users primarily concerned with battery life may prefer offline-only operation.

For Future Research: Subsequent studies should extend evaluation to multiple East African countries (Kenya,

Uganda, Rwanda, Burundi) to assess cross-national generalizability and identify region-specific scam patterns requiring localized model variants. Longitudinal studies spanning 6-12 months would reveal whether adaptive feedback mechanisms maintain performance as adversaries evolve tactics. Integration with mobile money APIs could enable automated transaction blocking for confirmed scam messages, converting a passive warning system into active fraud prevention.

For Software Engineering Practice: This study demonstrates that Design Science Research methodology effectively bridges academic machine learning and production software engineering. Practitioners should adopt similar hybrid architectures when deploying NLP models to resource-constrained environments, recognizing that offline fallback capabilities are not optional but essential for real-world reliability.

**Contribution to Knowledge**

This study makes three primary contributions to the software engineering and machine learning literature. First, it provides empirical evidence that hybrid deep learning architectures can achieve production-ready performance across multilingual SMS spam detection, URL analysis, and mobile deployment simultaneously. Second, it demonstrates that quantization techniques can reduce model size by 84% with acceptable accuracy reduction (4.19 percentage points), establishing a feasible path for deploying sophisticated models on budget Android devices. Third, it validates that user feedback mechanisms enable continuous model improvement in production environments, reducing the gap between model deployment and model iteration that characterizes current practice.

**References**

Al-Laith, A., & Alenezi, M. (2024). BERT-based Arabic SMS spam detection: A comparative study of transformer fine-tuning approaches. *Journal of King Saud University - Computer and Information Sciences*, 36(2), 101945. https://doi.org/10.1016/j.jksuci.2024.101945

Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A. (2011). Contributions to the study of SMS spam filtering: New collection and results. *Proceedings of the 11th ACM Symposium on Document Engineering*, 259-262. https://doi.org/10.1145/2034691.2034742

Alqahtani, H., & Alshahrani, M. (2025). Mobile phishing susceptibility: A systematic review of user factors and technical countermeasures. *Computers & Security*, 148, 104112. https://doi.org/10.1016/j.cose.2024.104112

Chiew, K. L., Tan, C. L., Wong, K., Yong, K. S. C., & Tiong, W. K. (2019). A review of phishing URL detection techniques. *Journal of Network and Computer Applications*, 129, 48-63. https://doi.org/10.1016/j.jnca.2018.12.009

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT 2019*, 4171-4186. https://doi.org/10.18653/v1/N19-1423

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., & Keutzer, K. (2021). A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*. https://arxiv.org/abs/2103.13630

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75-105. https://doi.org/10.2307/25148625

Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L. C., Tan, M., ... & Adam, H. (2022). Searching for MobileNetV3. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1314-1324.

Kumar, S., & Sinha, A. (2024). SMS spam detection: A comprehensive review from rule-based to deep learning approaches. *Expert Systems with Applications*, 238, 121845. https://doi.org/10.1016/j.eswa.2023.121845

Lee, J., Kim, Y., & Park, S. (2024). URL-enhanced deep learning for SMS phishing detection: A multi-modal approach. *IEEE Transactions on Information Forensics and Security*, 19, 2345-2358. https://doi.org/10.1109/TIFS.2024.3356789

Liang, H., & Xue, Y. (2010). Understanding security behaviors in personal computer usage: A threat avoidance perspective. *Journal of the Association for Information Systems*, 11(7), 394-413. https://doi.org/10.17705/1jais.00232

Mwangoka, J., & Mbelwa, J. (2024). Swahili-English code-mixed SMS corpus for spam detection in East Africa. *Data in Brief*, 52, 109876. https://doi.org/10.1016/j.dib.2023.109876

Onyango, D. (2023). Code-switching patterns in East African mobile communication: A sociolinguistic analysis. *Journal of African Languages and Linguistics*, 44(2), 187-210. https://doi.org/10.1515/jall-2023-2007

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345-1359. https://doi.org/10.1109/TKDE.2009.191

Roy, P. K., Singh, J. P., & Banerjee, S. (2023). Deep learning for SMS spam detection: A hybrid CNN-GRU approach. *Applied Soft Computing*, 133, 109850. https://doi.org/10.1016/j.asoc.2022.109850

Zhang, Y., Li, X., & Wang, D. (2024). Convolutional and recurrent neural networks for text classification: A comparative review. *Neurocomputing*, 575, 127258. https://doi.org/10.1016/j.neucom.2024.127258