Collaborative Dual-Framework Defense: CTI and LLM-Based Enhanced Smishing Detection

Li Guangliang, Kalaivani Selvaraj, Mahinderjit Singh*
School of Computer Sciences, University Sains Malaysia, 11800 Penang, Malaysia

***Abstract*—Smishing has become a severe cybersecurity threat. Attackers now use AI and social engineering to craft more sophisticated campaigns. To address this challenge, this study proposes a dual-layer detection framework. It combines cyber threat intelligence (CTI), machine learning, and a large language model (LLM). The framework uses 22 features built from 2,811 real SMS messages. These features are categorized as content-based, context-based, and Indicators of Compromise (IOC)-based features. Five machine learning models were evaluated. XGBoost, trained with a 70% training, 10% validation, and 20% test split, achieved the best performance. It had a recall of 92.08% and an F1-score of 94.66%. For borderline cases, the study experimented with 4 LLMs (including GPT-4o and LLaMA 3). They served as a semantic verification layer. All models achieved a recall rate above 98.5% and produced human-readable explanations. The study demonstrated that these 4 models are complementary verifiers rather than main classifiers. The results show that structured threat intelligence used during feature engineering improves machine learning model performance. With semantic reasoning, the framework also generates accessible reports for non-specialists. This lowers the barrier for effective smishing detection.**

***Keywords*—Smishing detection; cyber threat intelligence; XGBoost; semantic verification; large language model**

I. Introduction

Smishing remains one of the most widespread cyber threats globally, resulting in significant economic losses [1]. According to the latest report from the Anti-Phishing Working Group, both the frequency and complexity of such attacks have continued to rise recently [2].

Although most organizations train staff to avoid phishing attacks, such efforts are of limited value. Cybersecurity awareness training does not relate to autonomous detection of phishing attacks [3]. Humans struggle to recognize cyberattacks due to stress, burnout, or security fatigue. Cybercriminals exploit these weaknesses as attack points [4]. For this reason, technical methods must be stable. Technology faces two main challenges. First, AI enables phishing that is almost indistinguishable from natural writing, often powered by LLMs [5]. Another study found that AI-generated content is more effective than human-written content [6]. These contents are also easier to bypass commercial anti-phishing systems [7]. Second, while existing detection systems perform well, interpretability is still challenging for users of neural network-based techniques [8].

The purpose of CTI is to provide high-dimensional intelligence features. These help models understand deeper tactical threat characteristics, not just static content. Thus, the CTI guidance feature project upgrades content recognition to context and intent recognition. This approach systematically enhances the test model's performance and robustness. It is distinct from traditional attack identification, which only analyzes content [9], [10]. In addition, LLMs extract rich semantic representations and generate coherent, human-readable interpretations [11]. Some similar studies have been proposed in theory, but experimental evidence is lacking [12]. Combining CTI-guided feature engineering with an LLM is crucial for decision support and model transparency. There is a gap in related research on mobile platforms.

To address these issues, this study suggests a two-part smishing detection model for mobile platforms. The first part uses a CTI-driven feature engineering machine learning model for real-time detection. LLMs are then used for cross-validation and explanation. This collaborative design aims to achieve efficient detection with better precision and decision transparency.

The primary contributions of this research can be summarized as follows:

This research introduces a CTI-guided feature engineering framework, which integrates content and contextual metrics to enhance detection accuracy and model adaptability against smishing attacks.

A novel two-layer detection architecture that combines an ensemble learning-based classifier with an LLM for semantic verification.

A novel role for LLMs in smishing detection as explainable validators, assisting security analysts in improving interpretability.

The remainder of this study is structured as follows: Section II provides a comprehensive review of prior studies on smishing threats, feature engineering and detection approaches; Section III elaborates on the proposed dual-layer framework, including dataset description, feature engineering, model construction, and an LLM-driven semantic verification mechanism; Section IV presents the experimental evaluation and analyzes the model performance of the machine learning model and the two-layer integration framework. Finally, Section V concludes the study and highlights prospective directions for future research.

II. Related Work

The rising sophistication of smishing attacks, especially those enhanced by artificial intelligence (AI), represents a major challenge for cybersecurity defenses. This literature review

*Corresponding author.

addresses three aspects to provide background for this research. First, it reviews the evolution of smishing and AI-enabled attacks, focusing on unique mechanisms and emerging threats. Next, it explores how CTI can support detection and assist with feature engineering. Finally, it evaluates current phishing detection techniques, from traditional machine learning to modern LLMs, and compares their strengths and weaknesses in tackling adaptive threats.

A. Smishing Emerges and AI-Driven Threats

Smishing is a type of phishing carried out via mobile text messages. It uses social engineering content as the attack payload. The main goal is to deceive recipients and get them to click on malicious links, exposing private information [13]. Early smishing attacks used many generic messages to deceive a small group of targets. The mobile internet has since changed the threat landscape. From 2005 to 2015, attackers expanded their attacks to include channels such as SMS, instant messaging, and spoofed websites. From 2015 to 2025, attackers began using advanced automation and artificial intelligence. These tools produce highly personalized, compelling content and increase the success rate of attacks [14].

Recently, a study by [6] demonstrates, from a red-team perspective, that content generated by AI is more compelling than that written by humans. Quantitatively, AI spear-phishing agents have been shown to perform 23% more effectively than humans [15]. While attacks become smarter, detection technologies have not been upgraded in parallel, leaving gaps in effective, low-cost detection solutions.

B. CTI in Feature Engineering

CTI is a systematic security methodology focused on collecting, analyzing, and disseminating information about potential or current cyber threats, aiming to provide actionable insights to support cybersecurity decision-making [16], [17].

In the field of threat detection, CTI plays a fundamental role. While feature engineering approaches are commonly used to detect phishing, they struggle to handle AI-driven smishing that uses unstable keywords, ambiguous intentions, and changing content. CTI can be used to enhance the dimensions of features across multiple domains, URLs, IPs, and threat indicators.

A study proposed integrating PhishTank data with threat intelligence to build a hybrid deep learning model for email phishing, as noted in [12]. This study observed that cybersecurity threat databases from VirusTotal, PhishTank, and Google Safe Browsing can improve the detection of malicious URLs and phishing attempts. The study [18] developed a cyber-threat intelligence platform that aggregates data from honeypots and open-source intelligence to detect and prevent phishing attacks. The platform focuses on threat sharing. In [19], a large-scale framework was proposed for extracting text-based attack patterns from CTI reports, with a focus on classifying attack issues. The study [20] proposed an entity recognition (NER) method to extract structured STIX-based CTI features, which achieved 81.65% accuracy in STIX attack pattern classification. In [9], this study used CTI-based features derived from search engines and WHOIS records for malicious URL detection, achieving 7.8% higher accuracy and 6.7% fewer false positives with a random forest model and MLP ensemble.

Previous research has attempted to incorporate CTI to classify attack types and detect phishing emails, achieving high performance, but research on smishing combined with CTI remains sparse.

C. Detection Approaches from Machine Learning to LLMs

Artificial intelligence (AI) represents a broad category in computing technology. Machine learning is a subset of AI that learns patterns from data. Within AI, generative AI encompasses a range of capabilities, with large language models as the most important subset. While traditional machine learning offers resource savings, LLMs provide language understanding and generation capabilities. The evolution of AI technologies has also driven the gradual movement of detection methods from machine learning to large language model-based solutions. The traditional ML-based detection schemes, as well as deep learning and LLM-based approaches, have been reviewed below.

First, feature-based machine learning methods typically rely on manually engineering features such as URL length and the number of special characters, and they use this knowledge to train classification models for phishing attack identification. A study in [21] employed GPT-4o to generate 63 email simulations of phishing attack payloads and construct 47 new stylometric features for AI-generated phishing detection. Then, we employed the XGBoost model for detection, which showed the best performance. An innovative approach to extensible artificial intelligence (XAI) for feature selection, which not only improves the accuracy of phishing website identification but also enables modelling of phishing sites and delivers interpretable results, thereby fostering greater trust among stakeholders, was proposed in [22]. Additionally, a parameter-tuning approach for phishing URL detection was proposed in [23], which used three tuning strategies: data balancing, hyperparameter optimization, and feature selection. The results show that feature selection significantly improves the accuracy and performance of the gradient boosting model. These indicate that feature selection is crucial for traditional machine learning models, but there remains room for improvement in feature performance.

Deep learning-based detection methods leverage neural networks to learn hierarchical features from input data, overcoming the limitations of manual feature engineering. For example, the author in [24] proposed a deep learning framework that leverages URL features to detect phishing. This framework offers benefits in terms of speed and accuracy, but it still doesn't address the black-box problem. The hybrid GRU+CNN model proposed in [25] uses the Kaggle dataset, which contains more than 2.5 million samples of URLs labelled as "phishing." The CNN model's accuracy is 98%. However, real-time performance is not demonstrated. Although deep learning achieves high performance in some scenarios, it poses additional challenges for real-time response applications owing to its black-box nature and high computational requirements.

Recently, with the rise of LLMs, most methods have adopted LLMs as core detectors. Optimization of LLMs typically involves pre-training or fine-tuning a specialized detector on phishing data, along with prompt engineering. In [26], two LLMs were fine-tuned on 573,880 phishing and benign URLs, achieving a high accuracy of 99.86%, with performance close to

that of state-of-the-art general-purpose LLMs. However, it also confirms that generalized LLMs still perform as well as dedicated LLMs in specific domains. The study [27] used five generic LLMs evaluated across three phishing datasets, and the results show that GPT-4 Turbo is the best-performing LLM under one-shot prompting, and the interpretations are highly readable. The study [28] compared the performance of fine-tuning and prompt engineering in phishing URL detection. The results demonstrated that fine-tuning can improve prompt engineering's accuracy from 92.9% to 97.3%, but prompt engineering has the advantage of fast development. A PhishBERT pre-trained model for phishing URL detection was proposed in [29]. The results are very satisfactory, but it consumes significant infrastructure, including 10 NVIDIA V100 servers. Pre-training a model needs strong infrastructure support, while fine-tuning consumes slightly lower computational resources. Prompt engineering offers easy access and still acceptable performance. To summarize the above, the LLM for smishing detection is highly readable and suitable for intensive interpretation. However, most research focuses only on the LLM itself and does not consider the illusion of LLMs, leaving a gap in combining LLMs and ML for detection.

Therefore, although existing research has demonstrated the enormous potential of CTI for guiding feature engineering, it also leaves a clear gap for improvement in smishing detection. To address this limitation, this research explores the systematic use of structured cyber threat intelligence features for building a lightweight, accurate, and understandable smishing detection model.

III. Methodology

This study proposes a dual-layer detection framework (see Fig. 1) to address two critical challenges: the escalating sophistication of AI-driven smishing attacks and the lack of interpretability in the detection framework. Layer 1 performs rapid interception, while Layer 2 handles cross-verification and explanation.

Fig. 1. Dual synergistic defense model.

A. System Architecture

A dual-layer detection framework process flow (see Fig. 2) is proposed in this study, consisting of two core pipelines. First, this study constructed a dataset and engineered 22 CTI-driven features out of it. These features are used to train and evaluate the optimal machine learning model. Subsequently, the machine learning model's output, together with the original smishing content, is formatted into a structured prompt. This prompt is then fed into the LLMs for cross-validation and interpretation. Finally, the framework produces highly accurate, human-readable detection results.

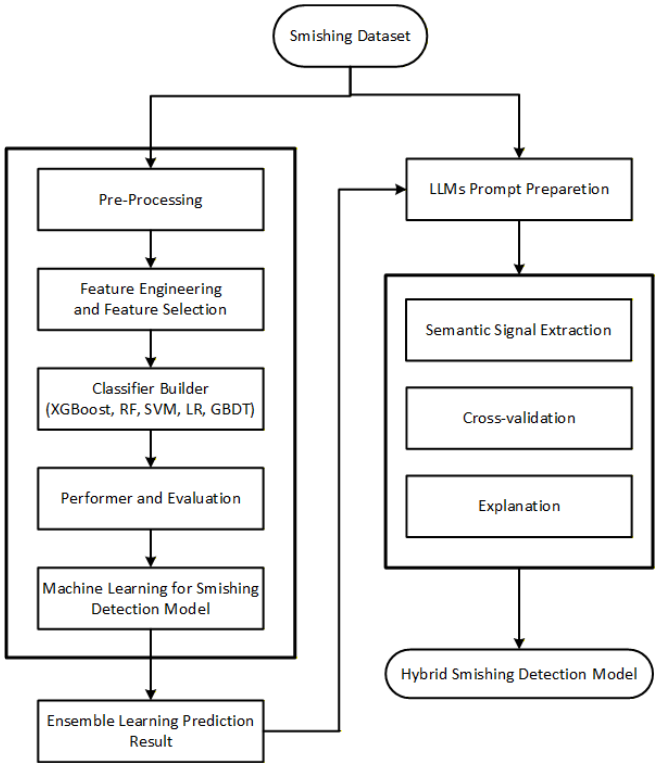

Fig. 2. Process flow of the hybrid smishing detection framework.

B. Dataset

This study uses a publicly available dataset, derived from SmishTank (https://smishtank.com/dataset). The advantage of this dataset is that it provides real-world SMS texts and their corresponding classification labels, with 22 feature dimensions. To enhance data coverage, this study employed a web crawler to collect additional samples from the source platform, expanding the dataset to 2,811 entries, as summarized in Table I.

TABLE I. Dataset Summary

Data Source	Total Samples	Legitimate	Smishing
SmishTank	2,811	1,799	1,012

To expand the number and coverage of data samples, we collected smishing records from SmishTank in real-time, as shown in Fig. 3, which displays the raw message content, associated URLs, and metadata used during dataset construction.

Upon obtaining the dataset, this study constructs the 22 features shown in Table II based on the CTI guidance and the existing literature, which categorizes IOC into three categories: content, context and IOC. CTI-driven feature set for smishing detection is given in Table II.

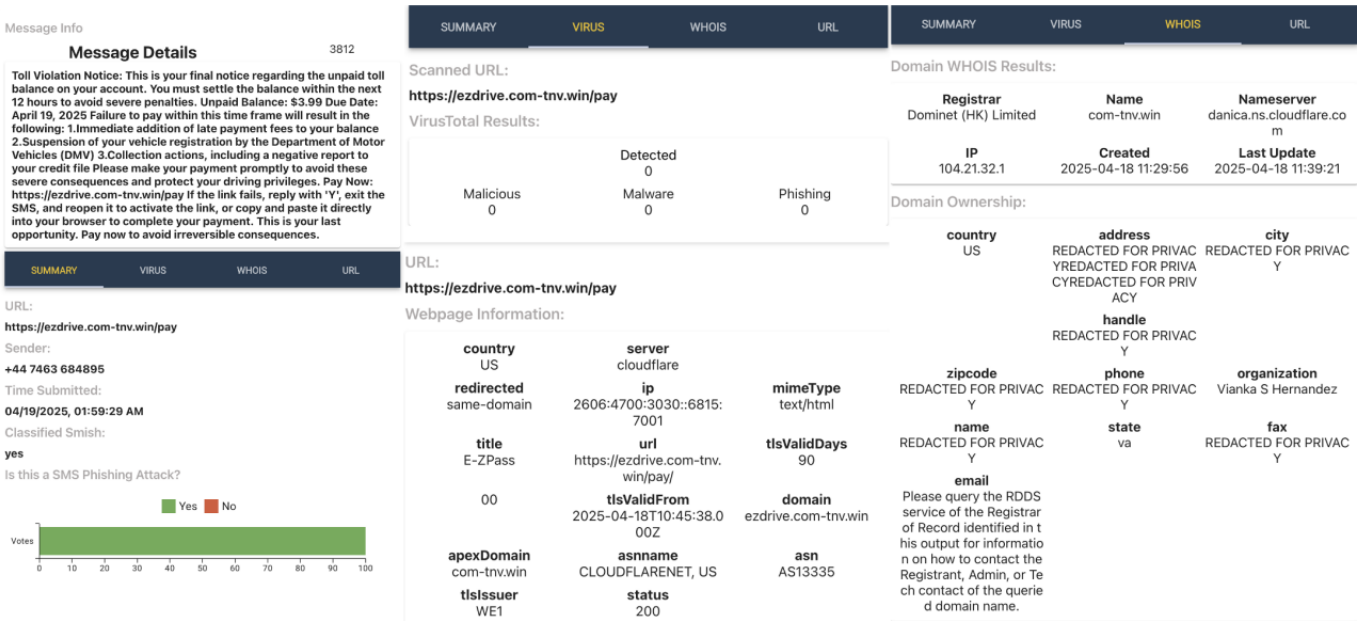

Fig. 3. Example of a smishing sample collected from SmishTank.

TABLE II. FEATURE CONSTRUCTION

ID	Feature Name	ID	Feature Name
1	sender_country_code	12	contains_financial_ref
2	sender_number_prefix	13	contains_forged_identity
3	receive_time	14	contains_account_ref
4	has_virus_malicious	15	contains_contact_instruction
5	has_virus_malware	16	has_verify_term
6	has_virus_detected	17	has_reward_term
7	tls_issuer	18	has_action_require
8	count_characters	19	has_login_term
9	has_command_tone	20	is_ip_in_url
10	has_threat_feature	21	domain_entropy
11	has_urgency_feature	22	is_url_shortener

IOC features such as has_virus_malicious, has_virus_malware, and has_virus_detected are obtained from external threat intelligence repositories, which explicitly identify malicious attacks and are therefore more reliable. Content features are a form of linguistic fingerprinting. They capture lexical and semantic patterns that differ from ordinary communication, reflecting the attacker's social engineering strategies. A key advantage of this category is its robustness; even if attackers modify domains or IP addresses, their linguistic habits and manipulation techniques are harder to conceal. Contextual features represent environmental cues that often go unnoticed individually but exhibit statistically meaningful patterns at scale, such as abnormal sending times. These three categories form a comprehensive feature set that enhances the model's ability to detect diverse smishing behaviors.

C. Machine Learning for Smishing Detection Model

This study employs five model selection methods: LR, SVM, RF, XGBoost, and GBDT. The selected classifiers represent complementary learning paradigms commonly used in security detection tasks. The linear model LR serves as the baseline, with SVM added. Tree-based and ensemble models (RF, GBDT, and XGBoost) are used to capture nonlinear interactions among CTI-driven features.

To ensure fair comparison and reproducibility, each model was configured using commonly adopted baseline parameter settings. Only limited, model-specific tuning was applied to control overfitting, prioritizing robustness and generalization over aggressive optimization.

The dataset is divided into training, validation, and test sets using a 70:10:20 ratio to balance learning capacity, parameter selection, and unbiased evaluation. Recall is emphasized as the primary metric, since missing a smishing message incurs a higher risk than issuing a false alarm. The F1 score is reported as a complementary metric to reflect the overall detection balance. Additionally, inference latency and memory consumption are evaluated to assess deployment feasibility on mobile devices. Finally, this study selects the best ML model as a detector for Layer 1 after evaluation.

D. LLM-Based Semantic Analysis and Cross-Verification

Four LLMs serve as interpreters and auxiliary detectors at Layer-2 to address the limitations of traditional machine learning models in interpretability and decision transparency. This layer is implemented in three parts: LLM prompt preparation, prompt execution, and result analysis. All four LLMs are evaluated using the same prompt and input structure to ensure comparability between models.

LLM prompt preparation: This study constructs a prompt that implements three functions in Fig. 4. Input the evaluation result of the ML for the LLM to evaluate whether it agrees or not, and give 0-100 as the likelihood of recognizing a phishing attack. Then extract the attack clues from the SMS payload and display them. In the end, explain the reason that the message is judged as a phishing attack.

```
You are a cybersecurity AI assistant.
Please analyze the following SMS message and its extracted features.
The XGBoost model classified this message as: {Phishing / Not Phishing}.
Do you agree or disagree? Provide your reasoning.
Message: "{message}"
Please rate it from 0 to 100:
  ▪ Decision: Phishing or Not Phishing, with a score (e.g., 85/100)
  ▪ Semantic Signal: Describe which signals indicate phishing or not
    phishing (around 30 words)
  ▪ Explanation: Provide a short paragraph explaining your reasoning
    behind the decision (around 50–100 words)
Example Output:
  ▪ Decision: Phishing (Score: x/100)
  ▪ Semantic Signal: Contains a suspicious shortened URL, urgent
    language about financial loss, impersonation of a legitimate
    institution, and no user-specific personalization.
  ▪ Explanation: This message creates pressure by stating an
    unauthorized transaction and includes a shortened URL (bit.ly),
    which obscures the final destination. It lacks personalization and
    uses generic terms, which are typical signs of phishing. Legitimate
    institutions usually include identifiable user information and avoid
    generic urgency tactics.
```

Fig. 4. Prompt for semantic verification.

Prompt execution: 1/5 of the dataset (2,811) is selected as the evaluation dataset. The LLM evaluator selects four LLMs for comparative evaluation: GPT-4o, GPT-4 Mini, Gemini 2.5 Pro, and LLaMA3.1-8B, which include two large-parameter models and two small-parameter models. Consistent prompt requests for evaluation results were sent to the four LLMs simultaneously.

Statistical analysis: In the final analysis stage, this study focuses on 3 dimensions: recognition accuracy, interpretation clarity, and semantic cue consistency.

In this coordination mechanism, Layer-1 addresses high-speed, low-cost detection, and Layer-2 provides interpretability, semantic robustness, and human-readable explanations.

IV. Results and Discussion

To validate the effectiveness of the proposed framework, this section presents the experimental results in three key parts: the performance of the machine learning models, the outcomes of the feature engineering process, and the analysis of the large language model's semantic verification.

A. Result of Machine Learning Model

To validate the effectiveness of the first layer of the proposed framework, this study conducts comparative experiments on five machine learning classifiers: LR, SVM, RF, GBDT, and XGBoost in a dataset divided into training, validation, and testing sets in a ratio of 70:10:20.

The results of the validation set, shown in Table III, indicate that the integrated learning model achieves a better balance among detection accuracy, recall, and robustness. Among them, XGBoost achieved the best performance on key metrics (Recall=94.12%, F1-Score=96.00%, Accuracy=97.16%), while Random Forest and GBDT also maintained high recall and AUC values (98.94% and 98.77%, respectively). These results demonstrate that the integrated model has a significant advantage in capturing smishing signals.

TABLE III. Performance Verification on Validation Set

Model	Precision	Recall	F1-Score	Accuracy	AUC
LR	98.86%	85.29%	91.58%	94.33%	97.83%
SVM	95.24%	78.43%	86.02%	90.78%	95.35%
RF	95.00%	93.14%	94.06%	95.74%	98.94%
GBDT	96.00%	94.12%	95.05%	96.45%	98.77%
XGBoost	97.96%	94.12%	96.00%	97.16%	98.76%

The results in Table IV further confirm the stability of the integrated model, with XGBoost maintaining the highest recall values, as with the RF model. Unfortunately, the precision of the RF model is also lower than that of XGBoost. Although the difference between GBDT and XGBoost is small, XGBoost still outperforms GBDT across most metrics. Comparing the validation and test sets, both indicate that XGBoost achieves better, more balanced results.

TABLE IV. Performance Verification on Test Set

Model	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	AUC (%)
LR	97.60	80.69	88.35	92.35	97.56
SVM	92.99	72.28	81.34	88.08	95.21
RF	92.96	91.58	92.27	94.48	98.14
GBDT	95.83	91.09	93.40	95.37	97.97
XGBoost	95.36	91.58	93.43	95.37	98.43

To assess the applicability of the five models to mobile devices, reasoning efficiency was measured. The results in Table V show that the overall inference performance is significantly lightweight. The single-sample inference latency of all models is in the millisecond range, and the maximum memory consumption during inference is only about 45 KB. However, XGBoost shows high efficiency in the inference phase, with an average latency of only 0.005 ms and a memory consumption of 8.56 KB, the lowest among all models.

TABLE V. Model Inference Time and Memory Usage

Model	Avg time (ms)	Max time (ms)	Avg mem (KB)	Max mem (KB)
Logistic Regression	0.002	0.003	45.30	45.44
SVM	0.057	0.060	45.29	45.43
Random Forest	0.047	0.048	45.66	45.79
GBDT	0.006	0.007	45.54	45.67
XGBoost	0.005	0.008	8.56	8.65

In Table VI, based on previous studies, we found that the higher the number of features, the better the model's performance tends to be. However, this study achieved relatively balanced performance using 22 features. The results of [30] used nearly twice as many features as this study, yet the precision was similar. Similarly, [22] used nearly 4 times as many features as this study. In addition, comparing the studies of [30] and [31], this study maintains robust precision, recall, and F1-score on a much larger sample size of 2811, suggesting that it achieves a more generalized and practical feature system with fewer features.

TABLE VI. COMPARATIVE RESULTS OF SMISHING DETECTION STUDIES

Study	Features	Samples	Precision	Recall	F1-Score	Accuracy
[21]	43	63	96.0%	96.0 %	96.0 %	96.0%
[22]	87	2,288	97.0%	97.0 %	97.0 %	96.8%
[31]	12	620	92.2%	92.2 %	92.2 %	92.2%
This study	22	2,811	95.4%	91.6 %	93.4 %	95.4%

In summary, XGBoost shows the best balancing ability in the first-layer framework of this study. First, XGBoost achieves the most balanced results across all recognition metrics and maintains the highest recall and F1 score. Second, in the inference phase, XGBoost's average latency is only 0.005 ms, and its memory consumption is about 8.56 KB, making it better than other models and especially suitable for deployment on resource-constrained mobile devices. Finally, XGBoost achieves stable, competitive results in this study with only 22 features, unlike previous studies that relied on much larger feature sets to achieve high performance. Based on these three lines of evidence, XGBoost achieves an optimal balance among performance, efficiency, and deployability and should be selected as a layer-1 model.

B. Feature Engineering with CTI

Smishing attacks are "semantically deceptive", "irregularly formatted", and "highly variant in content". Content features alone are not enough, so this study introduces CTI to enhance the traditional features. Therefore, this study introduces CTI to enhance the traditional features. To analyze the impact of CTI-guided features on mods, the XGBoost Gain on Feature Importance and a correlation heatmap were used.

The XGBoost gains the top twenty features in Fig. 5. This further demonstrates that the IOC metrics have_virus_detected and have_virus_malicious are the two most contributing to the model, followed by TLS issuer. Also, in the case of social engineering features, has_login_term, has_action_required and has__verify_term features also occupy an important position. This shows that IOC features are more valuable than social engineering features.

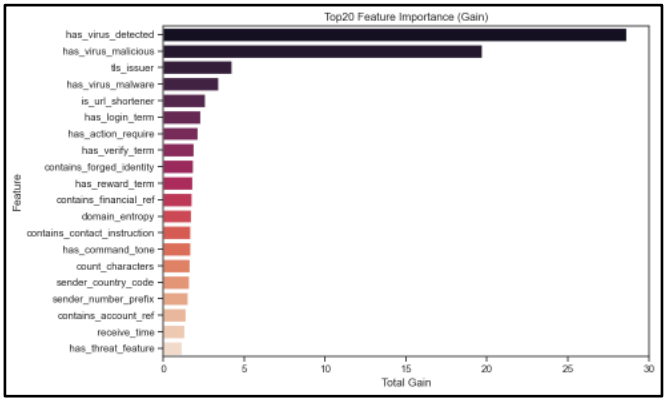

Fig. 5. Top-20 feature importance by XGBoost gain.

A strong association between has_virus_malware and has_virus_detected is observed in the correlation heatmap (Fig. 6). The correlation between social engineering features is also strong, whereas the correlation between IOC features and the middle of social engineering features is weak. These features confirm the independence of the 22 features.

After the import of CTI, the importance of IOC features among the top twenty is significantly increased. It indicates that when attack clues are predicted in an attack sample, it is more likely to be a smishing attack, which is more distinguishable than traditional content features. In addition, the correlation between IoC features is high, and the correlation between IoC features and social engineering features is weak, while there is also internal aggregation between social engineering features. This study suggests that CTI features do not duplicate traditional features' information but introduce new dimensions.

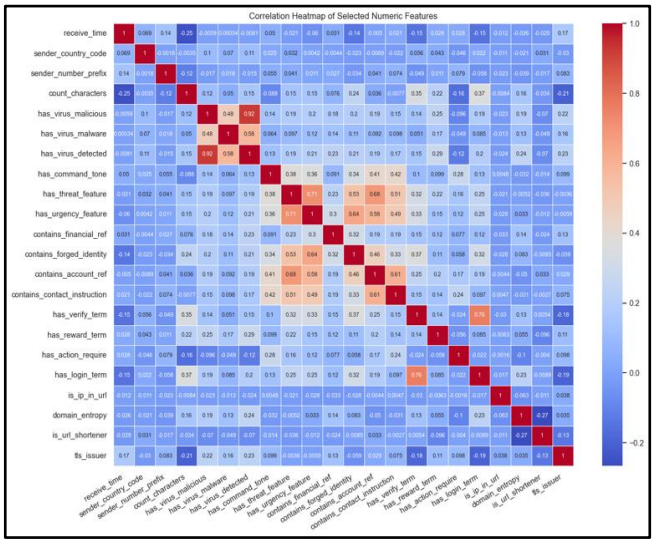

Fig. 6. Correlation heatmap of constructed features.

C. Result of LLMs

In the LLM cross-validation and semantic analysis phase, a total of 562 samples were selected for evaluation. The study first obtained the Layer-1 XGBoost classifier's predictions and compared with those of four LLMs. A representative boundary case was further examined to assess the model's semantic interpretation capabilities.

Since interpretation clarity and semantic cue consistency lack universally accepted objective measurements, this study treats them as proxy metrics. Interpretation clarity is defined as whether the LLM-generated explanation contains all required semantic components specified in the prompt and is consistent with the final decision, and it is measured by the proportion of samples satisfying these criteria. Semantic cue consistency is defined as the degree of overlap in identified smishing-related cues across different LLMs when analyzing the same message.

From Table VII, it is observed that the higher-parameter LLMs demonstrate higher performance in smishing detection. GPT-4o and Gemini-2.5 Pro both reduced the number of false negatives (FN) to 1, significantly outperforming XGBoost, which produced 14 FN on the same evaluation set. Among the models, Gemini-2.5 Pro achieved the most balanced overall performance.

TABLE VII. CROSS-VERIFICATION PERFORMANCE COMPARISON

Model	TP	FP	FN	TN
XGBoost	188	6	14	354
Llama3.1-8B	199	355	3	3
GPT-4omini	200	346	2	12
GPT-4o	201	351	1	7
Gemini-2.5pro	201	350	1	8

In addition to their predictive accuracy, an important advantage of LLMs is their ability to generate contextually relevant, human-readable explanations. To illustrate this capability, Fig. 7 presents a representative boundary case for the XGBoost classifier. The message contains social engineering-related keywords such as "refund" and "reply", which frequently appear in smishing messages, leading the feature-based model to classify it as phishing in the absence of explicit threat indicators. In contrast, the LLM identified the message as benign by recognizing that it originated from a legitimate "temu.com" source and followed the structure of a standard customer service notification. This example demonstrates the LLM's ability to capture higher-level semantic and organizational contexts that are difficult to model using handcrafted features alone.

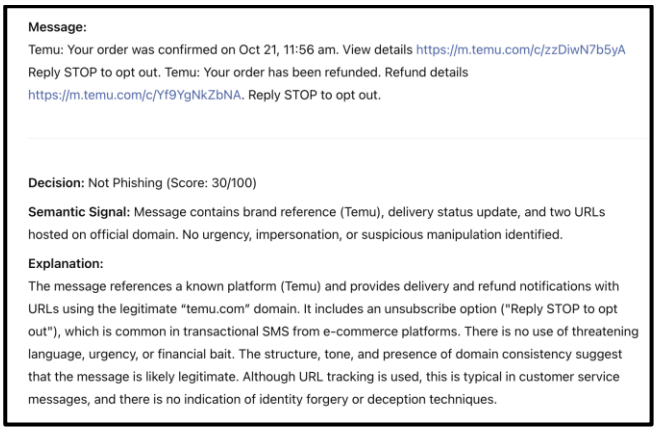

Fig. 7. LLM analysis sample for smishing detection.

D. Limitations

This study has several limitations. First, the dataset is primarily derived from SmishTank and may not fully capture the diversity of smishing campaigns across regions and languages, potentially affecting generalizability. Second, the effectiveness of CTI-guided features depends on the availability and quality of external threat intelligence, and incomplete or outdated CTI may reduce the contribution of IOC-related features. Third, although the LLM layer enhances semantic verification and interpretability, it involves additional computational overhead and is subject to known robustness issues; however, its impact is limited by its role as a secondary verifier rather than the primary detector.

From a security perspective, this design limits the impact of adversarial manipulation, as potential prompt injection or output manipulation at the LLM layer cannot directly bypass the CTI-guided machine learning decision in the first layer.

In addition, an explicit ablation study isolating CTI-driven features from traditional content- and context-based features is not conducted, as several CTI indicators are intrinsically integrated into the feature space. Nevertheless, feature importance and correlation analyses indicate that CTI features provide complementary and non-redundant information.

These limitations define the scope within which the experimental results should be interpreted.

V. CONCLUSION AND FUTURE WORK

Traditional smishing detection approaches face challenges balancing computational efficiency, expressive feature representation, and boundary-case identification. To address these issues, this study proposes a lightweight dual-layer detection framework that integrates a CTI-guided machine learning classifier with an LLM-based semantic verification module.

Experimental results show that XGBoost provides efficient and accurate first-layer detection, while the LLM layer complements it by improving recall and interpretability in ambiguous cases. This collaborative design achieves a practical balance between detection performance, transparency, and deployability in resource-constrained environments.

Future work will focus on expanding the dataset to cover multiple languages and regions, integrating more dynamic CTI sources, and conducting systematic ablation and cross-dataset evaluations to further quantify feature contributions and generalization. In addition, optimizing the cost and robustness of the LLM layer remains an important direction for large-scale deployment.

REFERENCES

[1] Hoxhunt, "Phishing Trends Report (Updated for 2025)." Accessed: Mar. 17, 2025. [Online]. Available: https://hoxhunt.com/guide/phishing-trends-report

[2] APWG, "APWG Trends Reports," Anti-Phishing Working Group. Accessed: Apr. 12, 2025. [Online]. Available: https://apwg.org/trendsreports/

[3] G. Ho et al., "Understanding the Efficacy of Phishing Training in Practice," 2025.

[4] C. Nobles, "Stress, Burnout, and Security Fatigue in Cybersecurity: A Human Factors Problem," HOLISTICA – Journal of Business and Public Administration, vol. 13, no. 1, pp. 49–72, July 2022, doi: 10.2478/hjbpa-2022-0003.

[5] C. S. Eze and L. Shamir, "Analysis and Prevention of AI-Based Phishing Email Attacks," Electronics, vol. 13, no. 10, p. 1839, May 2024, doi: 10.3390/electronics13101839.

[6] E. Lim, G. Tan, T. K. Hock, and T. Lee, "Turing in a Box: Applying Artificial Intelligence as a Service to Targeted Phishing and Defending against AI-generated Attacks," GovTech Singapore, White Paper, 2021. [Online]. Available: https://i.blackhat.com/USA21/Wednesday-Handouts/US-21-Lim-Turing-in-a-Box-wp.pdf

[7] S. S. Roy, P. Thota, K. V. Naragam, and S. Nilizadeh, "From Chatbots to Phishbots?: Phishing Scam Generation in Commercial Large Language Models," in 2024 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA: IEEE, May 2024, pp. 36–54. doi: 10.1109/sp54263.2024.00182.

[8] A. Barredo Arrieta et al., "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," Information Fusion, vol. 58, pp. 82–115, June 2020, doi: 10.1016/j.inffus.2019.12.012.

[9] M. Alsaedi, F. Ghaleb, F. Saeed, J. Ahmad, and M. Alasli, "Cyber Threat Intelligence-Based Malicious URL Detection Model Using Ensemble Learning," Sensors, vol. 22, no. 9, p. 3373, Apr. 2022, doi: 10.3390/s22093373.

[10] N. Sun et al., "Cyber Threat Intelligence Mining for Proactive Cybersecurity Defense: A Survey and New Perspectives," IEEE Commun. Surv. Tutorials, vol. 25, no. 3, pp. 1748–1774, 2023, doi: 10.1109/COMST.2023.3273282.

[11] X. Zhang, J. Zhang, B. Rekabdar, Y. Zhou, P. Wang, and K. Liu, "Dynamic and Adaptive Feature Generation with LLM," June 04, 2024, arXiv: arXiv:2406.03505. doi: 10.48550/arXiv.2406.03505.

[12] A. Shaikh, M. Shaikh, Student, Department of Information Technology, University of Mumbai, Mumbai (Maharashtra), India, Srivaramangai R., and Head of the Department of Information Technology, University of Mumbai, Mumbai (Maharashtra), India, "Smishing Detection: Combating SMS Phishing Attacks by Utilising Machine-Learning Algorithms," IJITEE, vol. 14, no. 5, pp. 28–33, Apr. 2025, doi: 10.35940/ijitee.D1068.14050425.

[13] P. Y. Leonov, "The Main Social Engineering Techniques Aimed at Hacking Information Systems," 2021.

[14] J. Osamor, M. Ashawa, A. Shahrabi, A. Philip, and C. Iwendi, "The Evolution of Phishing and Future Directions: A Review," iccws, vol. 20, no. 1, pp. 361–368, Mar. 2025, doi: 10.34190/iccws.20.1.3366.

[15] "Snapshot." Accessed: Aug. 24, 2025. [Online]. Available: https://hoxhunt.com/blog/ai-powered-phishing-vs-humans

[16] C. S. Johnson, M. L. Badger, D. A. Waltermire, J. Snyder, and C. Skorupka, "Guide to Cyber Threat Information Sharing," National Institute of Standards and Technology, NIST SP 800-150, Oct. 2016. doi: 10.6028/NIST.SP.800-150.

[17] M. Phythian, Understanding the Intelligence Cycle, 1st ed. Routledge, 2013. doi: 10.4324/9780203558478.

[18] I. Alzahrani, S. Lee, and K. Kim, "Practical Cyber Threat and OSINT Analysis based on Implementation of CTI Sharing Platform," May 06, 2024, Computer Science and Mathematics. doi: 10.20944/preprints202405.0277.v1.

[19] M. T. Alam, D. Bhusal, Y. Park, and N. Rastogi, "Looking Beyond IoCs: Automatically Extracting Attack Patterns from External CTI," in Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defences, Hong Kong, China: ACM, Oct. 2023, pp. 92–108. doi: 10.1145/3607199.3607208.

[20] S. Fujii, N. Kawaguchi, T. Shigemoto, and T. Yamauchi, "Extracting and Analyzing Cybersecurity Named Entity and its Relationship with Noncontextual IOCs from Unstructured Text of CTI Sources," Journal of Information Processing, vol. 31, no. 0, pp. 578–590, 2023, doi: 10.2197/ipsjjip.31.578.

[21] C. Opara, P. Modesti, and L. Golightly, "Evaluating spam filters and stylometric detection of AI-generated phishing emails," Expert Systems with Applications, vol. 276, p. 127044, June 2025, doi: 10.1016/j.eswa.2025.127044.

[22] S. S. Shafin, "An Explainable Feature Selection Framework for Web Phishing Detection with Machine Learning," Data Science and Management, p. S2666764924000419, Aug. 2024, doi: 10.1016/j.dsm.2024.08.004.

[23] S. R. Abdul Samad et al., "Analysis of the Performance Impact of Fine-Tuned Machine Learning Model for Phishing URL Detection," Electronics, vol. 12, no. 7, p. 1642, Mar. 2023, doi: 10.3390/electronics12071642.

[24] Y. B. Prasad and V. Dondeti, "PDSMV3-DCRNN: A novel ensemble deep learning framework for enhancing phishing detection and URL extraction," Computers & Security, vol. 148, p. 104123, Jan. 2025, doi: 10.1016/j.cose.2024.104123.

[25] S. M., N. K., S. K. Ravva, R. R., P. Balaji, and R. K. T., "Enhanced Phishing URL Detection Using a Novel GRU-CNN Hybrid Approach," JMC, pp. 089–101, Jan. 2025, doi: 10.53759/7669/jmc202505007.

[26] E. Irshad and A. Basit Siddiqui, "Cyber threat attribution using unstructured reports in cyber threat intelligence," Egyptian Informatics Journal, vol. 24, no. 1, pp. 43–59, Mar. 2023, doi: 10.1016/j.eij.2022.11.001.

[27] F. Rashid, N. Ranaweera, B. Doyle, and S. Seneviratne, "LLMs are one-shot URL classifiers and explainers," Computer Networks, vol. 258, p. 111004, Feb. 2025, doi: 10.1016/j.comnet.2024.111004.

[28] F. Trad and A. Chehab, "Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models," MAKE, vol. 6, no. 1, pp. 367–384, Feb. 2024, doi: 10.3390/make6010018.

[29] Y. Wang, W. Zhu, H. Xu, Z. Qin, K. Ren, and W. Ma, "A Large-Scale Pretrained Deep Model for Phishing URL Detection," in ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece: IEEE, June 2023, pp. 1–5. doi: 10.1109/ICASSP49357.2023.10095719.

[30] C. Opara, P. Modesti, and L. Golightly, "Evaluating spam filters and stylometric detection of AI-generated phishing emails," Expert Systems with Applications, vol. 276, p. 127044, June 2025, doi: 10.1016/j.eswa.2025.127044.

[31] S. Demir and E. K. Sahin, "An investigation of feature selection methods for soil liquefaction prediction based on tree-based ensemble algorithms using AdaBoost, gradient boosting, and XGBoost," Neural Comput & Applic, vol. 35, no. 4, pp. 3173–3190, Feb. 2023, doi: 10.1007/s00521-022-07856-4.