MultiPhishGuard: An LLM-based Multi-Agent System for Phishing Email Detection

Yinuo Xue
yxue579@aucklanduni.ac.nz
University of Auckland
Auckland, New Zealand

Eric Spero
eric.spero@auckland.ac.nz
University of Auckland
Auckland, New Zealand

Yun Sing Koh
y.koh@auckland.ac.nz
University of Auckland
Auckland, New Zealand

Giovanni Russello
g.russello@auckland.ac.nz
University of Auckland
Auckland, New Zealand

Abstract

Phishing email detection faces critical challenges from evolving adversarial tactics and heterogeneous attack patterns. Traditional detection methods, such as rule-based filters and denylists, often struggle to keep pace with these evolving tactics, leading to false negatives and compromised security. While machine learning approaches have improved detection accuracy, they still face challenges adapting to novel phishing strategies. We present MultiPhishGuard, a dynamic LLM-based multi-agent detection system that synergizes specialized expertise with adversarial-aware reinforcement learning. Our framework employs five cooperative agents (text, URL, metadata, explanation simplifier, and adversarial agents) with automatically adjusted decision weights powered by a Proximal Policy Optimization reinforcement learning algorithm. To address emerging threats, we introduce an adversarial training loop featuring an adversarial agent that generates subtle context-aware email variants, creating a self-improving defense ecosystem and enhancing system robustness. Experimental evaluations on public datasets demonstrate that MultiPhishGuard significantly outperforms Chain-of-Thoughts, single-agent baselines and state-of-the-art detectors, as validated by ablation studies and comparative analyses. Experiments demonstrate that MultiPhishGuard achieves high accuracy (97.89%) with low false positive (2.73%) and false negative rates (0.20%). Additionally, we incorporate an explanation simplifier agent, which provides users with clear and easily understandable explanations for why an email is classified as phishing or legitimate. This work advances phishing defense through dynamic multi-agent collaboration and generative adversarial resilience.

CCS Concepts

• **Security and privacy**; • **Computing methodologies** → **Multi-agent systems**;

Keywords

Phishing Email Detection, Large Language Model, Multi-Agent System, Reinforcement Learning, Adversarial Training

ACM Reference Format:
Yinuo Xue, Eric Spero, Yun Sing Koh, and Giovanni Russello. 2025. MultiPhishGuard: An LLM-based Multi-Agent System for Phishing Email Detection. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25).* ACM, New York, NY, USA, 17 pages. https://doi.org/XXXXXXX.XXXXXXX

CCS '25, Taipei, Taiwan

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Phishing remains one of the most persistent and damaging threats in cybersecurity, serving as a primary vector for data breaches and financial losses. The Anti-Phishing Working Group (APWG) reported a rise in phishing attacks from 877,536 in Q2 to 932,923 in Q3 2024 [2]. Phishing attacks have grown increasingly sophisticated over recent years [4]. These attacks have evolved from simple, deceptive messages to social engineering [30], spear-phishing [20], and even AI-driven content generation [50] to closely mimic legitimate communications. According to Verizon's 2024 Data Breach Investigations Report [58] and Proofpoint [43], phishing remains a top breach cause, with many attacks bypassing traditional filters. Together, these findings underscore the need for adaptive, resilient phishing detection systems to handle diverse attack strategies.

Traditional phishing detection methods have relied on rule-based filters and denylists [29], which are limited by their inability to keep pace with evolving attacker strategies such as domain spoofing, dynamic URL obfuscation, and context-aware social engineering. Static machine learning models, while more adaptable, often depend on predefined features and historical data, making them ineffective against novel or subtle threats. Deep learning approaches, including CNNs, RNNs, and pre-trained models like BERT [14], have improved detection accuracy by capturing contextual language cues. However, they tend to focus primarily on email text or URLs, overlooking other modalities, and their "black-box" nature hinders interpretability. More recently, LLM-based systems have been proposed to analyze phishing emails using natural language understanding, showing promise in identifying nuanced patterns across various attack types [3, 25, 47]. Despite their success, most LLM approaches remain single-agent architectures that produce binary outputs without transparent reasoning. They also lack adaptability [50], as they are not optimized to learn from advanced attacks, such

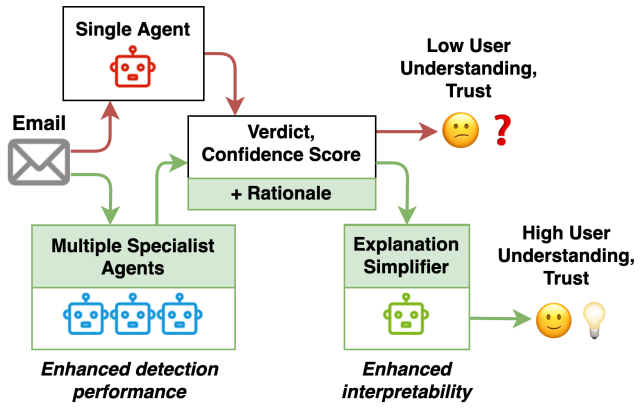

Figure 1: MultiPhishGuard Detection Pipeline. Incoming emails are pre-processed and evaluated in parallel by a suite of specialized LLM agents—each focused on a different phishing signal (metadata, message body, and URLs). Every agent outputs a verdict, a confidence score, and a detailed rationale. An explanation simplifier agent then consolidates those rationales into a single, clear, plain-language explanation. The result is a robust, accurate, and interpretable phishing verdict that outperforms traditional detection systems.

as spear phishing [7]. These limitations underscore the need for a multi-modal, explainable, and adaptive detection framework that integrates diverse data sources and provides users with interpretable, evidence-based decisions.

To address these limitations, we propose MultiPhishGuard, an LLM-based multi-agent system that integrates diverse detection modalities (text/URL/metadata) to enable comprehensive phishing email detection with transparent reasoning. As illustrated in Figures 1 and 2, our approach leverages LLMs not only to examine email content but also to scrutinize embedded URLs and metadata, thereby capturing malicious cues that single-modality systems often miss. By distributing specialized tasks among distinct agents, each focused on a specific aspect of the email, the system dynamically fuses its outputs through reinforcement learning. This design significantly reduces false positives and enhances the system's ability to adapt to emerging phishing tactics, ensuring robust and real-time threat assessments in an ever-evolving cybersecurity landscape. Our main contributions are as follows:

- We develop MultiPhishGuard, a novel LLM-based multi-agent system for phishing email detection that integrates diverse modalities—analyzing email message body, embedded URLs, and metadata—to capture malicious cues that traditional, single-modality detectors often miss. To the best of our knowledge, MultiPhishGuard is the first to utilize multiple LLM agents to deal with phishing emails. We also include our code as supplementary material for reproducibility.
- We incorporate adversarial training into our framework by deploying an adversarial agent that generates subtle variants of both phishing and legitimate emails. This approach further strengthens the system's robustness and resilience.
- We introduce an explanation simplifier agent that consolidates and distills the rationales from the text, URL, and metadata agents into a concise, jargon-free explanation, making decisions accessible to non-experts, and—we expect—building user trust.
- We evaluated our model on six public datasets, achieving a 95.88% F_1score. This demonstrates robust performance across diverse, real-world scenarios.

2 Background and Related Work

Phishing Email Detection. Traditional phishing detection methods initially relied on heuristic rules [26], denylists [55], and signature-based filtering [29]. With the advancement of machine learning, ML algorithms have increasingly been applied to detect phishing emails. For example, [10] converts textual content into TF-IDF feature representations and trains classifiers such as random forests. With the emergence of deep learning, researchers have advanced these approaches by employing neural networks that automatically learn complex patterns from emails. For instance, researchers reformulated phishing email detection as text classification problems by leveraging pre-trained language models like BERT to identify phishing emails [38]. Despite these advances, key limitations remain in addressing evolving phishing strategies. The existing methods usually focus on a single modality and lack the adaptability to counter rapidly evolving phishing tactics [19, 24, 32, 59]. In contrast, our MultiPhishGuard fills these gaps by utilizing a multi-agent system to integrate diverse modalities to capture the phishing cues in emails.

Reinforcement Learning. Reinforcement learning (RL) enables systems to adaptively optimize decisions in dynamic environments by continuously adjusting parameters based on feedback. Foundational work by Sutton and Barto [57] established the theoretical basis for RL, which has since driven advancements across domains. Early models like Deep Q-Networks (DQN) [36] and Deep Deterministic Policy Gradient (DDPG) [27] showed how agents can learn effective policies through interaction, while later methods such as Proximal Policy Optimization [54] improved stability and efficiency. These approaches offer a key advantage over static models by dynamically tuning parameters in real-time. In MultiPhishGuard, RL is integrated into a multi-agent framework to dynamically adjust agent weights based on the characteristics of each email, enhancing coordinated decision-making and improving adaptability in the face of evolving phishing threats.

Adversarial Training. Adversarial training has emerged as a key strategy for enhancing the robustness of machine learning models, especially in cybersecurity applications where even minor perturbations can lead to severe security breaches. The seminal work by Goodfellow et al. [17] demonstrated that by incorporating adversarial examples—carefully crafted inputs designed to mislead a model—into the training process, neural networks can become significantly more resistant to such attacks. The researchers also use adversarial examples to simulate the evolving tactics of attackers to expose vulnerabilities in static models [6, 22]. In the realm of cybersecurity, adversarial training is particularly valuable because it not only improves model robustness but also aids in reducing false negatives by preparing systems to recognize and counteract sophisticated

evasion techniques. By continuously challenging models with adversarially perturbed data that mimic real-world attack scenarios, these techniques ensure that detection systems remain adaptive and effective against emerging threats. Such dynamic defenses are crucial in an environment where attackers persistently update their strategies, underscoring the importance of adversarial training in developing resilient cybersecurity solutions. Our model utilizes the adversarial agent to generate both phishing and legitimate emails based on real-world emails, thereby enhancing its robustness and demonstrating its resilience against evolving threats.

Interpretability. Interpretability—the degree to which a human can understand the cause of a decision—is emerging as a critical requirement for deploying machine learning (ML) systems in real-world environments. Explanations provided by models can help users understand model behaviour and identify failure modes, which in turn can affect trust and decision quality [48]. Regulatory frameworks (e.g., GDPR's "right to explanation") increasingly mandate transparency, making interpretability not only a best practice but also a compliance requirement [18, 41].

LLM-based Multi-agent System. Multi-agent systems have long been studied as an effective way to tackle complex tasks by decomposing them into smaller, specialized subtasks. Foundational work, such as that by Shoham and Leyton-Brown [56], established theoretical frameworks showing how autonomous agents can interact, cooperate, and even compete to achieve a collective goal. In recent years, advances in multi-agent reinforcement learning—exemplified by Lowe et al. [31]—have demonstrated that collaborative strategies can significantly enhance performance in dynamic, mixed cooperative-competitive environments. Recent developments in LLMs have paved the way for LLM-based multi-agent systems and demonstrated their effectiveness in handling complex, multi-faceted tasks through collaborative specialization. For instance, the work on generative agents by Park et al. [40] illustrates how LLM-based agents can simulate intricate human behaviors and collaboratively solve problems by dividing tasks and dynamically coordinating their outputs. Thematic-LM [44] uses an LLM-based multi-agent system to perform large-scale thematic analysis, enhancing diversity, scalability, and interpretability in qualitative coding. Such systems effectively harness individual agents' complementary strengths—each focusing on different modalities or subtasks to achieve superior performance compared to single-agent approaches. Building on these insights, our MultiPhishGuard integrates multiple LLM-based agents to enhance phishing email detection.

3 MultiPhishGuard

Our proposed methodology leverages an LLM-based multi-agent system to enhance phishing email detection through a dynamic, adaptive, and explainable approach. As shown in Figure 2, the framework consists of several specialized agents, each focusing on distinct aspects of email analysis. By integrating multiple agents, dynamically adjusting their influence using reinforcement learning, and incorporating adversarial training, our model improves detection accuracy and robustness while providing clear, user-friendly explanations. This approach effectively mitigates the limitations of static, single-modality detectors, offering a more robust defense against phishing threats.

3.1 Basic Agents

Our phishing detection framework is built upon an LLM-based multi-agent system, where each agent specializes in analyzing different parts of an email. Unlike single-agent approaches that target only one modality (e.g., text, URL, or metadata) [19, 24, 59], our system leverages multiple perspectives to improve detection accuracy and robustness. Each agent operates independently, providing its own phishing verdict, confidence score, and reasoning. These outputs are then dynamically fused using reinforcement learning to produce a final classification. The agents are built using AutoGen [61]. Our system comprises three main components: a text analysis agent, a URL analysis agent, and a metadata analysis agent.

Text Analysis Agent: The text agent leverages an LLM to thoroughly analyze the email body, identifying suspicious patterns, phishing keywords, and any textual indicators of malicious intent. The prompt used by the agent is illustrated in Figure 3.

URL Analysis Agent: The URL agent extracts and scrutinizes all links in the email, checking for obfuscation, evaluating domain reputations, and detecting potential redirections to malicious sites. The prompt used by this agent is presented in Appendix A (Figure 6). The main difference between the prompts used by the URL and text agents is that the former is tailored to focus exclusively on suspicious URLs, while the latter analyzes only the textual content.

Metadata Analysis Agent: The metadata agent examines email headers, sender authentication records (e.g., SPF, DKIM, DMARC), and reply-to fields to identify anomalies that may signal phishing attempts. The corresponding prompt is shown in Appendix A (Figure 7). Unlike the text and URL agents, which focus respectively on the email body and embedded links, the metadata agent is specifically instructed to analyze only the metadata.

Each agent processes the email independently and generates a structured output containing (1) a phishing vs. legitimate label, (2) a phishing possibility score, and (3) a rationale for its decision.

As shown above, we adopted the same prompt format for the Text Agent, URL Agent, and Metadata Agent. This design was chosen to precisely define each agent's scope, structure its reasoning, and generate machine-readable output—drawing on best practices from the prompt engineering literature.

The prompts begin with "You are a cybersecurity expert specializing in phishing," reflecting the "system message" approach used in InstructGPT [39], which has been shown to significantly improve task adherence by clearly defining the model's role and focus area. By explicitly restricting the analysis to the email body ("Only focus on the email text; do not analyze URLs or metadata"), we prevent cross-modal interference and promote agent specialization [40]—an approach also supported in multi-agent frameworks for complex tasks. Additionally, requiring the output in JSON format (`{'verdict': ..., 'confidence': ..., 'rationale': ...}`) aligns with structured prompting principles recommended by Toolformer [53], enabling consistent integration, streamlined automated evaluation, and reliable downstream parsing.

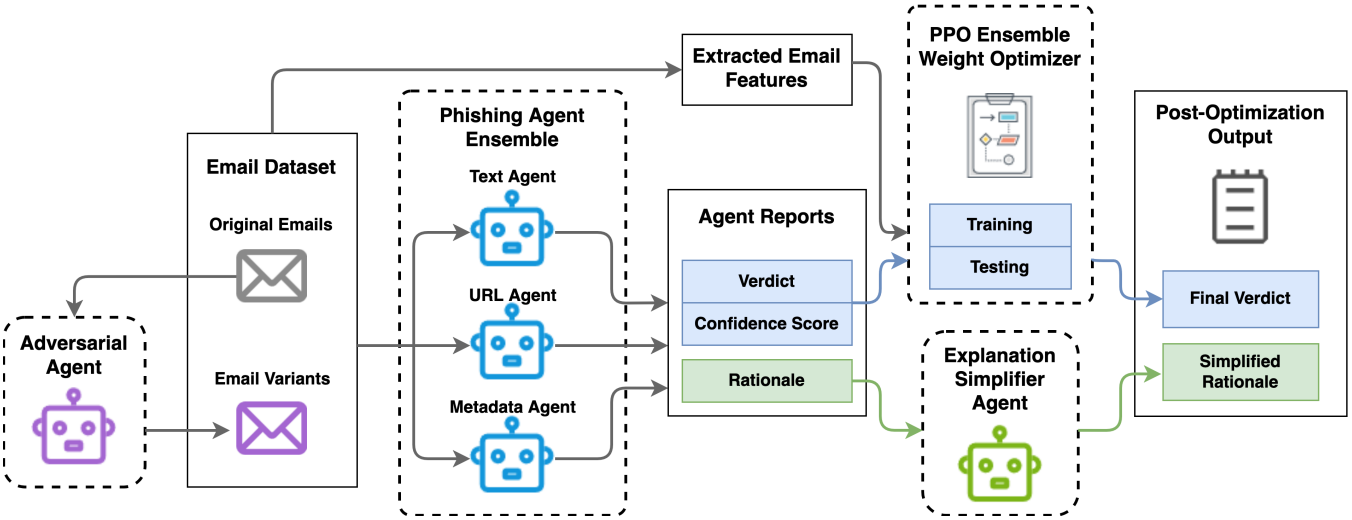

Figure 2: MultiPhishGuard Architecture. An Adversarial Agent generates subtle variants of both phishing and legitimate emails. For each email, three specialized sub-agents—the Text Agent (analyzes the message body), the URL Agent (inspects embedded links), and the Metadata Agent (evaluates header fields, reply-to fields, and sender authentication records)—each produces an agent report consisting of a verdict (phishing vs. legitimate), a confidence score, and a rationale. The verdict, confidence score, and extracted email features feed into a Proximal Policy Optimization (PPO) module. During training, PPO updates the sub-agents' weights to improve detection accuracy; during testing, it uses the optimized weights to estimate phishing likelihood. Finally, a Rationale Simplifier Agent consolidates the individual rationales into concise, user-friendly explanations.

```
You are a cybersecurity expert specializing in phishing, with a particular focus on email text content. Your task is to examine the email body exclusively for phishing cues—such as abnormal language patterns, suspicious vocabulary, or any textual indicators of malicious intent. Do not analyze URLs or metadata, only focus on the email text. Provide your judgment on whether the email is 'Phishing' or 'Legitimate', along with a confidence score between 0 and 1 and a clear, concise explanation of your reasoning. Output your result in JSON format as: 'verdict': 'Phishing' or 'Legitimate', 'confidence': 0-1, 'reasons': '...'
```

Figure 3: Text Agent's Prompt

3.2 Dynamic Weight Adjustment

Our framework employs an RL-based mechanism to dynamically adjust the weights of outputs from various specialized agents, ensuring that each email is evaluated according to its unique characteristics. Instead of relying on fixed weights, our system continuously learns to assign optimal importance to the analyses provided by the text, URL, and metadata agents. For each email, these agents generate independent predictions along with confidence scores. In addition, the RL module receives a vector of email-specific features that are extracted during preprocessing, including the number of URLs present, key phishing keywords in the text, sender domain reputation scores, authentication results (SPF/DKIM/DMARC), and the individual confidence scores provided by each agent.

Concretely, let each agent i produce a prediction probability p_i along with a confidence score, and let the weight assigned to that agent be w_i (with $\sum_i w_i = 1$). The final phishing score y is computed as a weighted sum of the agents' predictions:

$$y = \sum_{i=1}^{N} w_i \cdot p_i.$$

To optimize these weights dynamically, our system employs Proximal Policy Optimization (PPO), an efficient policy gradient method known for its stability and reliability in updating policies. We chose PPO over other reinforcement learning algorithms for several key reasons. First, PPO's clipped surrogate objective provides a simple yet effective way to enforce a "trust region" on policy updates, ensuring that each gradient step remains within a safe bound and preventing the large, destabilizing policy swings that can occur with vanilla policy gradients or Advantage Actor Critic (A2C). Second, PPO naturally handles continuous action spaces—such as our weight vectors—without requiring complex discretization or Q-function approximations, unlike DQN or DDPG. Third, compared to more complex trust-region methods like TRPO, PPO is far easier to implement and tune, with fewer hyperparameters and lower computational overhead. Finally, PPO has demonstrated strong empirical performance and sample efficiency across a wide range of continuous control and decision-making tasks, making it a robust, reliable choice for dynamically optimizing our multi-agent weighting scheme.

The PPO's clipped surrogate objective function is given by:

$$L(\theta) = \mathbb{E}\left[\min\left(r(\theta)\hat{A}, \text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A}\right)\right]$$

where $r(\theta) = \frac{\pi_\theta(w|x)}{\pi_{\theta_{\text{old}}}(w|x)}$ is the probability ratio between the current and previous policies, $\hat{A}$ is the estimated advantage, and ϵ is a small hyperparameter to limit policy updates. The "clip" is a mechanism that limits how much the probability ratio $r(\theta)$ can change during an update, ensuring that each policy update remains within a specified range. Concretely, the clip function takes the ratio $r(\theta)$ and forces it to lie between $1-\epsilon$ and $1+\epsilon$. This prevents large policy changes by effectively "clipping" the advantage, $\hat{A}$ when the new policy deviates excessively from the old policy. This formulation ensures that the weight updates remain within a bounded range, promoting stable learning.

In our implementation, the RL module treats the weights $w = [w_1, w_2, \ldots, w_N]$ as actions drawn from a policy $\pi_\theta(w|x)$, where x represents email-specific features (e.g., number of URLs and key phishing keywords in the text). The objective is to maximize the expected reward $E[r]$, and the reward function is the accuracy of the final classification. Through iterative feedback and reward optimization, PPO allows the system to refine its weighting strategy, continuously adapting to subtle and evolving phishing patterns.

This dynamic weighting strategy not only enhances detection accuracy but also mitigates false positives by tailoring the influence of each modality based on context. For example, when an email contains suspicious links, the model may prioritize URL analysis, whereas it may focus on metadata if the sender information appears inconsistent. The PPO-based RL module continuously optimizes these weight adjustments, ensuring that the detection framework remains robust against a wide range of phishing tactics. Consequently, our approach provides a resilient and adaptive solution that refines its performance in real-time, maintaining high accuracy even as attackers evolve their strategies.

3.3 Explanation Simplifier Agent

To enhance user understanding and trust in phishing email detection, our framework incorporates an explanation simplifier agent. Unlike traditional models that output only binary labels (phishing or legitimate), or systems that pair labels with confidence scores but offer little to no explanation, our system generates clear, user-friendly explanations that summarize why an email was flagged as phishing. This feature is particularly valuable for non-expert users, security analysts, and organizations that require interpretable results for decision-making and cybersecurity awareness training.

Given that our multi-agent system analyzes email text, URLs, and metadata separately, each agent outputs an independent decision with its own confidence score and technical reasoning. However, presenting these raw rationales can be overwhelming for users without cybersecurity expertise. To address this, the explanation simplifier agent—illustrated in Figure 4—aggregates the multi-agent outputs, extracts key insights, filters redundant or overly technical details, and synthesizes them into a single, coherent explanation.

We carefully designed the Explanation Simplifier prompt to enforce role clarity, factual consistency, and readability. The prompt begins with "*You are an expert in cybersecurity with deep expertise in phishing,*" following instruction-tuning strategies that improve adherence to role-specific tasks [39]. The agent is instructed to "synthesize detailed technical explanations … into one coherent, plain-language explanation," guiding it to unify varied reasoning

```
You are an expert in cybersecurity with deep expertise
in phishing. Your task is to take the detailed technical
explanations provided by the three specialized agents
(text, URL, and metadata) for why an email is classified
as phishing or legitimate, and synthesize them into one
coherent, reliable, and complete explanation written in
plain, everyday language. Ensure that your explanation
is truthful, meaningful, and based solely on factual
evidence—do not include any fabricated details. Avoid
technical jargon, simplify complex concepts, and provide
clear, concise reasons for the classification that
accurately reflect the underlying data.
```

Figure 4: Explanation Simplifier Agent's Prompt

into a cohesive summary. We explicitly require that explanations be "truthful ... based solely on factual evidence—do not include any fabricated details," ensuring factual integrity. Directives such as "avoid technical jargon" and "provide clear, concise reasons" further enhance accessibility.

The final explanation produced by the agent: **(1)** synthesizes multi-agent insights into a single, coherent response; **(2)** eliminates jargon to improve accessibility for non-technical users; **(3)** highlights critical phishing indicators, such as suspicious sender details, deceptive language, or malicious links.

To support users with varying technical backgrounds, we also introduce an Expert Mode, which delivers more detailed, technical explanations—complete with indicators of compromise, email header analysis, and references to threat intelligence frameworks. In addition, to validate the usability of our explanation simplifier agent across different user groups, we plan to conduct user experiments involving both non-expert and expert users. These evaluations are considered future work and out of scope in this paper. We expect that this dual-mode approach will not only foster user trust and cybersecurity awareness across expertise levels but also facilitate both everyday safety decisions and in-depth incident investigations.

3.4 Adversarial Training Module

To enhance the robustness of our phishing detection system, we introduce an Adversarial Training Module that continuously challenges and refines the model against sophisticated evasion tactics. At the core of this module is an adversarial agent—a GPT-4o-based large language model—tasked with generating nuanced variants of both phishing and legitimate emails. These variants are crafted to bypass conventional detection methods while operating within ethical and controlled boundaries. By exposing the detection model to such adversarial examples, the system becomes increasingly resilient to real-world threats and better equipped to recognize emerging attack patterns.

Adversarial training is implemented as an iterative process in which the adversarial agent serves as a generator, crafting challenging email variants that aim to evade current detection capabilities. Unlike traditional adversarial approaches that rely on imperceptible perturbations, our agent subtly modifies existing emails to mirror contemporary phishing tactics. These adversarial examples are evaluated by a multi-agent detection model functioning as a

You are an expert adversarial email generator. Your objective is to produce a variant of the provided email that maintains the original meaning and structure while incorporating subtle modifications designed to bypass the phishing detectors. Depending on the type of email provided, follow the corresponding instructions:

For phishing emails:

(1) Synonym Substitution: Replace keywords with synonyms (e.g., "verify" → "confirm", "account" → "profile", "free" → "no money is needed") so that the literal expression changes while the meaning remains intact.
(2) Sentence Rewriting: Alter the sentence structure without changing the underlying message (e.g., transform "Update your account immediately" into "Please refresh your account details at your earliest convenience"). Add decoy sentences about customer support/legitimate services. Remove overt threat indicators while maintaining urgency.
(3) Content Modification: Add or remove words and phrases as needed; for example, insert a neutral sentence like "We hope this email serves you well" or omit less critical content, to change the text's composition.
(4) Homoglyph Replacement: Substitute characters with similar-looking counterparts (e.g., replace the letter "a" in "paypal.com" with a Cyrillic "a" to disguise URLs while retaining their recognizable form.
(5) Polymorphic Variation: Modifying aspects such as the subject line, sender information, or overall format, thereby simulating a diverse range of phishing attack styles.

For legitimate emails:

(1) Subtle Suspicious Modifications: Modify the email in ways that make it appear more ambiguous or borderline suspicious (e.g., incorporate slightly urgent language or modify the subject line) without compromising its inherently benign intent.
(2) Synonym Substitution and Sentence Rewriting: Use similar techniques as above but ensure that the overall message remains authentic and professional, even if the modifications introduce elements that could potentially confuse detection systems.
(3) Content Enhancement: Optionally insert additional phrases that mimic some characteristics of phishing emails (e.g., ambiguous urgency or formatting cues), while still maintaining the legitimacy of the email.
(4) Polymorphic Variation: Adjust non-critical elements like the layout or minor stylistic details to introduce natural variability without altering the email's genuine nature.

Output Requirements:

- For phishing emails, the final variant should retain the malicious intent and target brand while evading detection.
- For legitimate emails, the final variant should remain clearly benign and professional, yet include subtle modifications that challenge the detector.
- Provide only the final modified email text and do not disclose the modification details.

Figure 5: Adversarial Agent Prompt

discriminator, whose goal is to correctly classify each email as phishing or legitimate. The adversarial agent seeks to maximize the discriminator's error by producing emails that are difficult to detect, thereby encouraging the discriminator to adapt and improve. This adversarial interplay fosters the learning of more robust features and decision boundaries, significantly enhancing the model's defense against both legacy and advanced phishing strategies.

The adversarial agent generates phishing emails that closely resemble authentic communications while intentionally avoiding common phishing signatures. As illustrated in Figure 2, it does this by modifying existing phishing emails, creating new deceptive messages, and iteratively refining them based on feedback from the detection model. In parallel, the agent also generates legitimate email variants that subtly mimic phishing traits—without crossing into malicious territory—to further stress-test the system's ability to make fine-grained distinctions. This dual-generation strategy challenges the model from both ends, ensuring robustness even in ambiguous cases.

This generation process is embedded within an iterative training loop driven by feedback from the detection model. Initially, the adversarial agent produces adversarial samples—either crafted phishing emails or perturbed legitimate messages—designed to probe the classifier's weaknesses. These samples are evaluated by the detection model $f(x;\theta)$, which outputs classification labels and confidence scores. If the model misclassifies a sophisticated phishing attempt or incorrectly flags a benign email, the resulting error signal is used to refine the agent's strategy. Practically, this involves tuning transformation parameters and adjusting prompt instructions, forming a continuous loop of improvement.

This feedback loop is realized through periodic evaluation cycles, during which newly generated adversarial examples are reintroduced into the system. This process not only strengthens the agent's generative capability but also fortifies the detection model against a wider array of adversarial strategies.

Formally, let $f(x;\theta)$ denote the detection model, which outputs the probability p of an email x being phishing. The adversarial agent generates inputs x_{adv} designed to mislead the model. Whereas traditional adversarial training perturbs an input as $x_{\text{adv}} = x + \delta$, with $\|\delta\| \leq \epsilon$, our approach leverages the agent's generative power to synthesize entirely new, context-aware adversarial examples that reflect real-world evasion techniques.

As shown in Figure 5, the agent employs five key transformation strategies: Synonym Substitution [51], Sentence Rewriting [62], Content Modification, Homoglyph Replacement [1], and Polymorphic Variation [8]. These transformations preserve the semantic intent and structure of the original message while introducing variations designed to circumvent detection mechanisms. An illustration of this process is in Appendix B (Figure 8), where a legitimate email is transformed into a challenging adversarial counterpart.

To prevent misuse and mitigate cybersecurity risks—especially those associated with generating phishing-like content—the adversarial emails produced during our experiments are not publicly released and are used solely for internal testing and model refinement.

By incorporating adversarial training, our phishing detection system achieves several key benefits. It becomes more resistant to common evasion strategies, such as URL obfuscation and subtle social engineering cues, thereby reducing its vulnerability to adversarial manipulation. Furthermore, the system generalizes more effectively, enabling it to detect previously unseen phishing attacks while accurately identifying perturbed but benign messages. Most critically, the adversarial training framework ensures that the detection model evolves alongside emerging phishing tactics, maintaining high accuracy even as threat landscapes shift.

In summary, by embedding an LLM-driven Adversarial Training Module that leverages strategic content transformations within an

iterative feedback loop, our system attains robust and adaptive protection against evolving phishing techniques—ensuring resilience and forward-looking effectiveness in a real-world deployment.

4 Experiments

We employ GPT-4o[1] as our LLM agent, with JSON mode activated to ensure a consistent and easy-to-evaluate output format. We have selected GPT-4o for its stable API access and consistently reliable performance. Its architecture integrates seamlessly with AutoGen's multi-agent framework, facilitating the development of collaborative AI agents. A series of experiments were conducted to evaluate MultiPhishGuard's effectiveness in both identifying phishing emails and generating well-reasoned explanations.

4.1 Datasets

We evaluate our proposed LLM-based multi-agent system using six widely recognized datasets: the Nazario phishing corpus [37], the Enron-Spam dataset [34], the TREC 2007 public corpus [12], the CEAS 2008 public corpus [13], the Nigerian Fraud dataset [46], and the SpamAssassin public corpus [42]. These datasets encompass a diverse collection of phishing and legitimate emails, providing a robust testbed for our detection framework.

The Nazario phishing corpus comprises Jose's self-reported phishing emails collected from 2005 to 2024. To ensure our model is tuned to recognize the most recent phishing indicators, we selected only the phishing emails from 2024, totaling 402 emails. Additionally, the Nigerian Fraud dataset contains over 2,500 emails dating from 1998 to 2007; we chose the most recent subset from 2007, which includes 577 emails. In contrast, the Enron-Spam dataset comprises both spam and ham emails. Since our study specifically targets phishing detection rather than general spam filtering, we used the ham emails as examples of legitimate messages, randomly sampling 1,500 emails for our experiments. To further demonstrate the model's transferability to other legitimate datasets, we randomly selected 500 ham emails each from the TREC 2007 and CEAS 2008 public corpora. In addition, we selected 500 "hard ham" emails from the SpamAssassin public corpus to ensure our model is capable of accurately classifying even the most challenging legitimate emails.

Overall, MultiPhishGuard was evaluated on 979 phishing emails and 3,000 legitimate emails—approximately 4,000 emails in total—with a phishing-to-legitimate ratio of roughly 1:3.

4.2 Evaluation Metrics

4.2.1 Evaluation of Phishing Detection. To comprehensively evaluate the performance of our phishing detection system, we utilize several standard classification metrics, including Recall, Precision, Accuracy, F_1 score, True Negative Rate (TNR), False Positive Rate (FPR), False Negative Rate (FNR). These metrics provide insights into the effectiveness of the model in correctly identifying phishing emails while minimizing false detections. Explanations of these metrics can be found in Appendix C.1. By incorporating these evaluation metrics, we ensure a balanced assessment of the model's performance, focusing not only on detection accuracy but also on minimizing the risks of false positives (which disrupt users) and false negatives (which allow phishing attacks to succeed), which are critical for maintaining both security and user experience.

4.2.2 Evaluation of Generated Rationale. In addition to evaluating discriminative performance, we assess the quality of the explanations generated by the explanation simplifier agent using both automated metrics and human evaluation. Since interpretability plays a crucial role in user trust and decision-making, we measure the clarity, coherence, and readability of the generated explanations through the Perplexity [23], Topic Coherence [49], and Flesch Reading Ease Score [16]. Further explanations about these metrics can be found in Appendix C.2. By combining these metrics, we ensure that our explanations are not only technically sound but also clear, user-friendly, and informative, ultimately improving trust and usability in phishing detection.

4.3 Comparative Evaluation

We perform a comparative evaluation of our proposed MultiPhishGuard against three alternative approaches—Chain of Thought, a single-agent model, and the RoBERTa-base baseline—on the datasets and evaluation metrics described in Sections 4.1 and 4.2.

While our primary analysis pools all datasets to evaluate overall performance, Table 3 and Section 4.3.5 show comparisons broken down by individual dataset.

4.3.1 MultiPhishGuard. The experiments were conducted within a framework that integrates multiple specialized agents (text, URL, and metadata) whose outputs are dynamically fused using a PPO-based reinforcement learning module. Our framework also incorporates adversarial training to continuously challenge the system and an explanation simplifier agent to provide clear, user-friendly rationales for its phishing/legitimate classifications.

In our experiments, MultiPhishGuard demonstrated exceptional performance on phishing detection tasks. Evaluated on a comprehensive dataset, as shown in Table 1, our system achieved a recall of 99.80% and a precision of 92.26%, resulting in an overall accuracy of 97.89% and an F_1 score of 95.88%. The true negative rate was equally impressive at 97.27%, with a false positive rate of only 2.73% and a false negative rate of 0.20%. These metrics confirm that MultiPhishGuard can effectively identify phishing emails while minimizing both false alarms and missed detections.

Beyond detection performance, we also evaluated the quality of the explanations generated by our system. As shown in Table 2, we measured the fluency of the explanations using perplexity, which was determined to be 25, indicating that the outputs are highly fluent. To assess the semantic consistency of the topics within the explanations, we calculated the topic coherence score, which stood at 0.35. Additionally, the Flesch Reading Ease Score was 41, suggesting that the explanations are moderately easy to read and comprehend for a general audience. Overall, these results highlight that MultiPhishGuard not only achieves outstanding discriminative performance but also provides clear, coherent, and accessible explanations, which we expect would lead to enhanced user trust and understanding.

4.3.2 Chain-of-Thoughts. We compared a chain-of-thought (CoT) prompting approach against our proposed MultiPhishGuard system. The CoT method leverages the ability of LLMs to articulate

[1] https://platform.openai.com/docs/models/gpt-4o

Table 1: Evaluation of phishing detection for different models across all datasets

Approach	Recall (%)	Precision (%)	Accuracy (%)	F_1 score (%)	TNR (%)	FPR (%)	FNR (%)
MultiPhishGuard	**99.80**	**92.26**	**97.89**	**95.88**	**97.27**	**2.73**	**0.20**
CoT	99.08	64.93	86.60	78.45	82.53	17.47	0.92
Single Agent	99.39	61.31	84.42	75.84	79.53	20.47	0.61
RoBERTa-base	98.37	86.21	95.73	91.89	94.87	5.13	1.63

Table 2: Evaluation of model-generated rationales

Approach	Perplexity	Topic Coherence	FRES
MultiPhishGuard	**25**	**0.35**	**41**
CoT	47	0.28	21
No Explanation	33	0.32	27

intermediate reasoning steps before arriving at a final decision—a technique that has been shown to improve performance on complex reasoning tasks [60]. When applied to phishing detection, the CoT approach encourages the model to explain its thought process, potentially revealing subtle cues in the email content. While CoT prompting is effective in eliciting intermediate reasoning steps in certain tasks, it presents several challenges for phishing detection.

As shown in Table 1, the CoT approach demonstrates a high recall rate of 99.08%, indicating that it effectively identifies the majority of phishing emails. However, its precision is considerably lower at 64.93%, largely due to a false positive rate of 17.47% and a true negative rate of 82.53%. Consequently, while the CoT method detects most phishing instances (with only a 0.92% false negative rate), it also misclassifies a significant number of legitimate emails as phishing. This imbalance results in an overall accuracy of 86.60% and an F_1 score of 78.45%, highlighting the method's tendency to generate noisy outputs. These results suggest that, although CoT is effective at flagging potential phishing emails, its high false positive rate may limit its practical reliability in real-world applications. These results may stem from the CoT methods generating inconsistent or excessively detailed reasoning paths, which can compromise the detection of nuanced, multi-modal phishing cues.

The CoT approach exhibits several critical shortcomings in its explanations. As shown in Table 2, a perplexity score of 47 suggests that the generated text is relatively less fluent and predictable, which can hinder natural language understanding. Additionally, a topic coherence score of 0.28 indicates that the explanations lack semantic consistency, making it difficult for users to discern clear and logical reasoning. Furthermore, with a Flesch Reading Ease Score of only 21, the explanations are notably hard to read, implying that they are filled with technical jargon and complex sentence structures that impede accessibility. Collectively, these issues highlight that the CoT method struggles to provide explanations that are both coherent and user-friendly.

4.3.3 Single-Agent Model. We also compared a single-agent model with our proposed MultiPhishGuard system to evaluate their effectiveness in phishing detection. The single-agent approach relies on one LLM (GPT-4o) to process the entire email and generate a phishing verdict directly. While this method has been applied in prior studies and can yield satisfactory results in simpler scenarios, it typically focuses on a single modality. This limitation often leads to higher false positive rates and reduced robustness when facing sophisticated phishing tactics.

As shown in Table 1, the single-agent approach achieves an impressive recall rate of 99.39% and a false negative rate of 0.61%, indicating that it is highly effective at detecting phishing emails. However, its precision is only 61.31%, accompanied by a relatively high false positive rate of 20.47% and a low true negative rate of 79.53%. This means that while the system successfully identifies most phishing instances, it also incorrectly classifies a significant number of legitimate emails as phishing, leading to an overall accuracy of 84.42% and an F_1 score of 75.84%. These issues highlight the inherent limitations of a single-agent model, which, by relying on a solitary modality, may fail to capture the diverse and nuanced cues present in complex, multi-modal phishing attacks.

4.3.4 Baseline: RoBERTa-base. Since RoBERTa-base [28] outperformed other pre-trained language models [50]—including BERT-base [14], DistilBERT [52], ELECTRA [11], DeBERTa [21], and XL-Net [63]—we chose it as the baseline for our phishing detection experiments. Renowned for its robust performance in natural language understanding tasks, RoBERTa-base is pre-trained on extensive corpora and optimized through dynamic masking strategies, making it particularly effective for text classification. To ensure a fair comparison, we used the same training settings as in [50].

In the experiments, as shown in Table 1, RoBERTa-base achieved a recall of 98.37%, accuracy of 95.73%, true negative rate of 94.87%, and an F_1 score of 91.89%. While these results indicate strong overall detection performance, several issues remain. First, the precision of 86.21% coupled with a false positive rate of 5.13% suggests that a notable number of legitimate emails are incorrectly flagged as phishing, which could lead to unnecessary disruptions and reduced user trust. Also, the false negative rate of 1.63% indicates that some phishing emails are still missed, posing potential security risks.

Another significant limitation of RoBERTa-base is its black-box nature; it does not provide interpretable reasoning or explanations behind its classifications. This lack of transparency makes it difficult for users and security analysts to understand why a particular email was labeled as phishing, hindering efforts to fine-tune detection criteria and improve overall system trustworthiness. These shortcomings highlight the need for a more comprehensive approach, such as our proposed MultiPhishGuard, which not only enhances detection accuracy but also delivers clear, user-friendly explanations for its decisions.

Table 3: Phishing and legitimate classification performance across six email corpora. All values are percentages.

	Phishing				Legitimate							
	Nigerian		Nazario		Enron		SpamAssassin		CEAS-08		TREC-07	
Approach	**TPR**	**FNR**	**TPR**	**FNR**	**TNR**	**FPR**	**TNR**	**FPR**	**TNR**	**FPR**	**TNR**	**FPR**
MultiPhishGuard	**100.00**	**0.00**	**99.50**	**0.50**	**97.47**	**2.53**	**95.00**	**5.00**	**98.20**	**1.80**	**98.00**	**2.00**
CoT	99.83	0.17	98.01	1.99	80.20	19.80	73.40	26.60	88.80	11.20	92.40	7.60
Single Agent	**100.00**	**0.00**	98.51	1.49	77.33	22.67	63.20	36.80	88.60	11.40	93.40	6.60
RoBERTa-base	99.31	0.69	97.01	2.99	95.80	4.20	89.40	10.60	96.40	3.60	96.00	4.00

4.3.5 McNemar's Test on Discordant Pairs. To compare classifier performance, we performed one-sided McNemar's tests [33]. Let n_{10} be the count of cases where MultiPhishGuard is correct and the alternative approach (the Chain-of-Thoughts method, the Single-Agent model, and the baseline RoBERTa-base) is incorrect, and let n_{01} be the count of cases where the alternative approach is correct and MPG is incorrect. The tests are one-sided, with the hypothesis $n_{10} > n_{01}$. For $n_{10} + n_{01} > 25$, we use McNemar's χ^2 test using the `exact2x2` package [35] with mid-p correction [15]; otherwise we use McNemar's exact test using the `binom.test` [45] function. We adjusted the p-values using Benjamini-Hochberg [5].[2] The results of these tests are shown in Table 4.

On the phishing-email datasets (Nigerian and Nazario), MultiPhishGuard won almost all discordant comparisons, with the difference surpassing the threshold for significance in two cases: versus CoT and versus Baseline in the Nazario dataset. In the legitimate-email datasets, MultiPhishGuard again dominated the discordant pairs and achieved statistically significant improvements over both Chain-of-Thought and Single-Agent across all corpora; it also significantly outperformed the RoBERTa-base model in the Enron, SpamAssassin, and TREC-07 datasets.

These results suggest that MultiPhishGuard consistently errs less often than the alternatives.

4.3.6 Summary. Our evaluation of MultiPhishGuard demonstrated high detection accuracy and a balanced trade-off between precision and recall. As further detailed in the ablation studies (Section 4.4.4), incorporating the adversarial agent ensures robust performance against evolving phishing strategies. The results indicate that, compared to CoT prompting, single-agent approaches, and the RoBERTa-base, our approach delivers superior accuracy and improved explainability. Overall, these experiments validate that MultiPhishGuard's multi-agent, adaptive, and explainable design provides a resilient and effective solution for phishing detection in diverse threat environments.

4.4 Ablation Studies

In our ablation studies, we systematically evaluated the contribution of individual components in the MultiPhishGuard framework by removing them while retaining all other components, and comparing the performance of the ablated variants against the full model. Variants were created that (1) lack a URL Agent, (2) lack a Metadata agent, (3) use static weighting instead of PPO-optimized weighting, (4) lack an Adversarial Agent for generating subtle email variants, (5) lack an Explanation Simplifier Agent. The results of the ablation studies are shown in Table 5. While our focus is on pooled datasets, results broken down by individual dataset are shown in Table 6 and Section 4.4.6.

Table 4: One-sided McNemar's tests on discordant pairs (n_{10}: MulitPhishGuard correct, alternative incorrect; n_{01} vice versa) across six email corpora. We hypothesize $n_{10} > n_{01}$.

Dataset	MultiPhishGuard vs.	n_{10}	n_{01}	p
Nigerian ($n = 577$)	CoT	1	0	0.529
	SingleAgent	0	0	1.000
	Baseline	4	0	0.070
Nazario ($n = 402$)	CoT	6	0	0.023
	SingleAgent	4	0	0.070
	Baseline	12	2	0.011
Enron ($n = 1500$)	CoT	263	4	$< .001$
	SingleAgent	305	3	$< .001$
	Baseline	62	37	0.011
SpamA. ($n = 500$)	CoT	115	7	$< .001$
	SingleAgent	161	2	$< .001$
	Baseline	46	18	$< .001$
CEAS-08 ($n = 500$)	CoT	50	3	$< .001$
	SingleAgent	52	4	$< .001$
	Baseline	18	9	0.056
TREC-07 ($n = 500$)	CoT	28	0	$< .001$
	SingleAgent	23	0	$< .001$
	Baseline	18	8	0.036

4.4.1 No URL Agent. As shown in Table 5, removing the URL agent led to a noticeable decline in detection performance. The absence of a dedicated URL analysis component significantly impaired the model's ability to capture subtle phishing indicators embedded in hyperlinks—particularly those obfuscated through redirection or evasion techniques. This resulted in an increase in false positives

[2]We control the false-discovery rate using Benjamini-Hochberg (BH) rather than a family-wise method (e.g. Bonferonni) because our primary aim is to screen for promising performance improvements across multiple benchmarks. BH offers greater power by limiting the expected proportion of false positives among significant results, rather than the stricter guarantee against any false positive.

Table 5: Ablation study—email classification metrics across all datasets

Variant	Recall (%)	Precision (%)	Accuracy (%)	F_1 score (%)	TNR (%)	FPR (%)	FNR (%)
MultiPhishGuard	**99.80**	**92.26**	**97.89**	**95.88**	**97.27**	**2.73**	**0.20**
No URL	97.96	73.83	90.95	84.20	88.67	11.33	2.04
No Metadata	97.14	84.46	94.90	90.36	94.17	5.83	2.86
Static Weight	99.18	84.80	95.43	91.43	94.20	5.80	0.82
No Adversarial	98.77	84.31	95.17	90.97	94.00	6.00	1.23

Table 6: Ablation study—email classification metrics by dataset. All values are percentages.

	Phishing				Legitimate							
	Nigerian		Nazario		Enron		SpamAssassin		CEAS-08		TREC-07	
Variant	TPR	FNR	TPR	FNR	TNR	FPR	TNR	FPR	TNR	FPR	TNR	FPR
MultiPhishGuard	**100.00**	**0.00**	**99.50**	**0.50**	**97.47**	**2.53**	**95.00**	**5.00**	**98.20**	**1.80**	98.00	2.00
No URL	**100.00**	**0.00**	95.02	4.98	96.00	4.00	73.20	26.80	77.00	23.00	93.80	6.20
No Metadata	99.65	0.35	93.53	6.47	94.67	5.33	85.60	14.40	97.20	2.80	**98.20**	**1.80**
Static Weight	**100.00**	**0.00**	98.01	1.99	93.73	6.27	90.20	9.80	97.00	3.00	96.80	3.20
No Adversarial	99.83	0.17	97.26	2.74	94.53	5.47	88.80	11.20	95.80	4.20	95.80	4.20

and reduced overall detection robustness. These findings highlight the critical role of URL analysis in providing complementary insights that strengthen the system's ability to identify sophisticated phishing attempts. The experiment underscores the necessity of incorporating URL-based inspection in a multi-agent architecture to ensure comprehensive and resilient phishing detection.

4.4.2 No Metadata Agent. As shown in Table 5, removing the Metadata Agent led to a decline in detection performance, as the system became less capable of identifying subtle inconsistencies in email headers and sender details—signals that are often indicative of spoofing or other deceptive tactics. The absence of this component impaired the model's ability to detect metadata-level phishing cues, such as forged sender addresses or manipulated header fields, which are frequently exploited in more sophisticated attacks. While the remaining agents continued to capture many phishing attempts through text and URL analysis, the overall robustness of the system was weakened. This experiment highlights the essential role of metadata inspection in enhancing detection accuracy and resilience, reinforcing the need for a multi-agent approach that integrates metadata analysis to address evolving phishing strategies.

4.4.3 Static Weight. We replaced the dynamic weight adjustment module with a static weighting scheme, where each agent's output was assigned a fixed, predetermined weight. From our previous experimental results, we observed that the URL agent had a greater impact on overall accuracy than the metadata agent. Removing the URL agent led to a more significant decline in detection performance, indicating its higher importance in phishing detection. Consequently, we assigned a weight of 0.3 to the text agent, 0.4 to the URL agent, and 0.3 to the metadata agent to better reflect their relative contributions to the final classification. As shown in Table 5, this modification led to a decrease in detection performance, suggesting that the system struggled to balance the contributions of different agents effectively. Without the dynamic adjustment mechanism, the model lost the flexibility to tailor its weighting based on the unique characteristics of each email. Unlike the RL-based approach, which adapts in real-time to emphasize the most informative signals—be it from text, URL, or metadata—the static scheme imposed rigid constraints that limited the system's responsiveness to varied phishing strategies. It appears that this rigidity made the model more prone to misclassifications, particularly when phishing attempts relied heavily on one modality. The experiment underscores the importance of dynamic weight adaptation in enabling the system to make more informed and context-aware decisions, thereby enhancing its overall robustness and adaptability in phishing detection.

4.4.4 No Adversarial Agent. We removed the adversarial agent to assess its impact on the model's robustness against sophisticated phishing attacks. As shown in Table 5, this removal led to a decline in the system's ability to detect more advanced and evasive phishing emails. The adversarial agent plays a key role in enhancing model resilience by generating deceptive, hard-to-detect phishing examples during training. Without this component, the model was exposed only to standard phishing patterns, limiting its capacity to generalize to more complex or novel attack strategies. Although the core detection capabilities remained intact, the model became more prone to misclassifying sophisticated phishing emails as legitimate. This highlights the critical role of adversarial training in fortifying the system against real-world threats and ensuring sustained effectiveness in the face of evolving phishing tactics.

4.4.5 No Explanation Agent. Finally, we removed the explanation simplifier agent to assess its impact on user comprehension and overall system usability. While this component does not directly influence detection accuracy, its absence significantly impaired the interpretability of the system's outputs. As shown in Table 2, the

resulting explanations had a perplexity of 33, a topic coherence score of 0.32, and a Flesch Reading Ease score of 27—indicating moderate fluency but poor semantic organization and readability. Without the simplifier, the system continued to classify emails correctly but presented raw reasoning from individual agents in a fragmented and jargon-heavy manner. These explanations lacked coherence and were often difficult for non-expert users to interpret, reducing the system's practical utility. Moreover, the absence of a unified, accessible explanation made it more time-consuming for analysts to piece together the rationale behind each decision. This experiment highlights the essential role of the explanation agent in translating technical outputs into clear, concise, and user-friendly narratives, which we expect would enhance user trust and support effective decision-making in phishing detection.

4.4.6 McNemar's Test on Discordant Pairs. We applied the same discordant pair analysis methodology described in Section 4.3.5 to compare MultiPhishGuard against its ablated variants; the results are shown in Table 7.

In the phishing-email datasets (Nigerian and Nazario), MultiPhishGuard won almost all discordant comparisons. The Nigerian set showed no significant differences, while on the Nazario set MultiPhishGuard significantly outperformed the NoURL, NoMetadata, and NoAdversarial variants.

In the legitimate email datasets, MultiPhishGuard again dominated the discordant pairs and achieved statistically significant improvements over the ablated variants in all cases—except StaticWeight and NoMetadata on the CEAS-08 and TREC-07 sets.

4.4.7 Summary. Overall, these findings indicate that the URL and Metadata Agents, the PPO-based weighting scheme, and the adversarial agent each contribute substantively to MultiPhishGuard's performance, and the Explanation Simplifier to its interpretability.

4.5 Human Evaluation

We conducted human evaluations with a cybersecurity expert through a multi-step process. These evaluations assessed the quality of the explanations generated by our model by directly comparing them with the detailed analyses provided by the expert. This comprehensive approach ensures that our system's output not only meets quantitative performance criteria but also aligns with expert human judgment regarding clarity, coherence, and factual accuracy.

First, we randomly selected a small subset of phishing emails from our datasets. We then invited a cybersecurity expert to manually review these emails and document their reasoning for classifying each email as phishing or legitimate. The expert was instructed to record the key indicators they identified and to provide detailed explanations of their thought processes.

Next, we compared these expert analyses with the explanations generated by MultiPhishGuard, CoT, and an ablated version of our system without the explanation simplifier agent using automated metrics—ROUGE [9] and cosine similarity. Further explanations about these metrics can be found in Appendix C.2.

Together, these metrics allow us to perform a comprehensive comparison between the human and machine-generated explanations, ensuring that our model's output is not only quantitatively similar but also semantically aligned with expert reasoning.

Table 7: One-sided McNemar's tests on discordant pairs (n_{10}: MulitPhishGuard correct, ablated variant incorrect; n_{01} vice versa) across six email corpora. We hypothesize $n_{10} > n_{01}$.

Dataset	MultiPhishGuard vs.	n_{10}	n_{01}	p
Nigerian ($n = 577$)	NoURL	0	0	1.000
	NoMetadata	2	0	0.300
	StaticWeight	0	0	1.000
	NoAdversarial	1	0	0.571
Nazario ($n = 402$)	NoURL	19	1	$< .001$
	NoMetadata	24	0	$< .001$
	StaticWeight	7	1	0.050
	NoAdversarial	9	0	0.004
Enron ($n = 1500$)	NoURL	45	23	0.007
	NoMetadata	72	30	$< .001$
	StaticWeight	77	21	$< .001$
	NoAdversarial	69	25	$< .001$
SpamA. ($n = 500$)	NoURL	119	10	$< .001$
	NoMetadata	66	19	$< .001$
	StaticWeight	37	13	$< .001$
	NoAdversarial	40	9	$< .001$
CEAS-08 ($n = 500$)	NoURL	112	6	$< .001$
	NoMetadata	11	6	0.210
	StaticWeight	7	1	0.050
	NoAdversarial	18	6	0.019
TREC-07 ($n = 500$)	NoURL	21	0	$< .001$
	NoMetadata	1	2	0.955
	StaticWeight	11	5	0.140
	NoAdversarial	16	5	0.021

Table 8: Automated evaluation metrics comparing expert analyses with different model explanations

Approach	ROUGE-1	Cosine Similarity
MultiPhishGuard	**0.59**	**0.82**
CoT	0.45	0.70
No Explanation	0.42	0.64

As shown in Table 8, MultiPhishGuard achieved a ROUGE-1 score of 0.59 and a cosine similarity of 0.82, outperforming CoT (ROUGE-1: 0.45; Cosine Similarity: 0.70). When the explanation simplifier agent was removed, performance dropped (ROUGE-1: 0.42; Cosine Similarity: 0.64), underscoring the critical role of this agent in producing explanations that faithfully capture expert insights. These results demonstrate that MultiPhishGuard not only excels in quantitative detection metrics but also generates explanations that are more coherent and semantically aligned with expert judgment.

In addition to automated metrics, we carried out a detailed qualitative analysis, comparing the expert's written rationale with MultiPhishGuard's explanations. Our qualitative review confirmed that MultiPhishGuard reliably captures similar phishing indicators

noted by experts—such as suspicious links and abnormal subjects—and, in several cases, even uncovers subtle cues that experts overlooked. For example, the system flagged the use of generic salutations like "Dear Customers" instead of addressing recipients by name, a telltale sign of mass phishing campaigns. Combined with strong automated scores (ROUGE-1: 0.59, Cosine Similarity: 0.82) and our manual analysis, these findings demonstrate that MultiPhishGuard not only identifies a wide range of phishing signals but also explains them in clear, user-friendly language that aligns closely with expert reasoning.

5 Discussion

Our experimental results demonstrate that MultiPhishGuard significantly outperforms both traditional and modern phishing detection methods across key dimensions. Compared to the baselines, it consistently achieves higher detection accuracy, lower false positive rates, and enhanced robustness across diverse phishing corpora.

These improvements stem from its multi-agent architecture, where specialized agents—focused on email text, URL structure, and metadata—analyze different facets of an email before synthesizing insights through a dynamically optimized ensemble mechanism. Another standout feature is the use of reinforcement learning via PPO to dynamically adjust agent weights based on email-specific characteristics. Unlike prior work relying on static or heuristic weighting, this dynamic approach allows the system to emphasize the most informative modalities, improving both adaptability and detection performance in the face of evolving phishing strategies.

Another notable contribution is the system's strong explainability. By incorporating an explanation simplifier agent, MultiPhishGuard generates clear, truthful, and non-technical rationales for its decisions. Human evaluation and automated metrics show that these explanations align well with expert reasoning and are more complete and readable than those from CoT or ablated models. This makes the framework particularly suitable for user-facing and high-stakes applications where transparency and auditability—such as under GDPR—are essential.

To enhance robustness, MultiPhishGuard employs an LLM-based adversarial training module that continuously generates subtle phishing and legitimate email variants designed to evade detection. This iterative fine-tuning exposes model vulnerabilities and improves generalization to novel attack patterns.

The system's modular design also supports practical deployment. For example, organizations can enable only the text agent to comply with privacy constraints, avoiding the processing of sensitive metadata. This flexibility allows LLM-based detection to operate effectively even in privacy-sensitive environments.

Looking forward, the architecture could be extended to include agents specializing in attachments, behavioral patterns, or hierarchical coordination mechanisms. MultiPhishGuard serves as a blueprint for scalable, interpretable multi-agent systems applicable to broader cybersecurity domains like malware detection and malicious website classification. For practitioners, it shows how modular, explainable, and adaptive LLM-based frameworks can be deployed in real-world settings with both effectiveness and accountability.

In summary, MultiPhishGuard advances the state of phishing detection by unifying multi-modal analysis, dynamic optimization, and interpretability within a coherent system. It not only bridges gaps in performance and transparency but also paves the way for the next generation of adaptive and explainable cybersecurity tools powered by LLMs.

5.1 Limitations

The limitation arises from the interpretability process. Our model's output 'rationale' lacks ground truth annotations, as the dataset does not provide explanations for why an email is classified as phishing or legitimate. As a result, we can only evaluate the explainability of our model using indirect measures such as readability, coherence, perplexity, and human analysis.

Also, due to the scarcity of open-source phishing detection datasets and publicly available code, we were able to compare our approach with only one recently published state-of-the-art model that had open-source code available as a baseline. Expanding comparative evaluations with additional high-quality benchmarks would further validate MultiPhishGuard's effectiveness.

5.2 Ethical Considerations

Ethical considerations are paramount in our research on phishing detection. First, we ensure all email data used in our experiments is anonymized and handled in accordance with privacy regulations, so that no personally identifiable information is disclosed or misused. Furthermore, since our work involves generating explanations for phishing classifications, it is crucial that we do not inadvertently expose sensitive decision-making processes that could be exploited by malicious actors. Our evaluation of explainability is conducted with careful attention to minimizing risks while maximizing transparency, ensuring that the model's outputs remain secure and do not provide actionable guidance for circumventing detection.

Additionally, to further validate our model's robustness beyond real-world email detection, we employed the adversarial agent to generate a set of both phishing and legitimate emails using GPT-4o, reflecting the growing misuse of LLMs for creating malicious content. Strict ethical guidelines governed this process; all generated emails were produced in a secure, controlled environment to ensure that no harmful content was released or disseminated, and they were used solely for testing and improving our detection system. Moreover, we will not release any of the generated emails, whether phishing or legitimate, to protect against potential misuse.

Moreover, a significant challenge in deploying LLMs is the risk of hallucination—where the model generates information that sounds plausible but is factually incorrect or fabricated. To address this issue, we explicitly instruct the text, URL, and metadata agents to operate strictly within their respective modalities, thereby preventing them from drawing conclusions based on information outside their designated scope. Additionally, the explanation simplifier agent is directed to ground its output exclusively in the reasoning provided by the text, URL, and metadata agents. Its prompt is carefully crafted to prioritize factual consistency and discourage the generation of unsupported claims. Through a combination of prompt engineering, output constraints, and post-hoc validation, we mitigate the risk of hallucinated content and ensure that the system's responses remain accurate, trustworthy, and anchored in genuine detection evidence.

6 Conclusion

In this paper, we introduced MultiPhishGuard, an innovative phishing detection framework that leverages an LLM-based multi-agent system to improve both detection performance and interpretability. By integrating specialized agents for text, URL, and metadata analysis, and dynamically adjusting their contributions using PPO, our approach significantly outperforms traditional single-agent methods and CoT techniques. The adversarial training also improves robustness against sophisticated phishing attacks. Experimental results demonstrate that MultiPhishGuard achieves exceptional detection accuracy (97.89%), while effectively minimizing false positives and false negatives. Moreover, the inclusion of an explanation simplifier agent provides clear, user-friendly rationales for its decisions, addressing a critical gap in model transparency and trustworthiness.

Overall, MultiPhishGuard offers a robust, adaptive, and transparent solution for phishing detection that not only enhances cybersecurity defenses but also facilitates better understanding and trust among users. We believe that our approach paves the way for future research into scalable, explainable, and ethically responsible cybersecurity systems.

References

[1] Abdullah M Almuhaideb, Nida Aslam, Almaha Alabdullatif, Sarah Altamimi, Shooq Alothman, Amnah Alhussain, Waad Aldosari, Shikah J Alsunaidi, and Khalid A Alissa. 2022. Homoglyph attack detection model using machine learning and hash function. *Journal of Sensor and Actuator Networks* 11, 3 (2022), 54.

[2] APWG. 2024. *Unifying the Global Response To Cybercrime.* Phishing Activity Trends Report, 3rd Quarter 2024. APWG.

[3] Mohammad Asfour and Juan Carlos Murillo. 2023. Harnessing large language models to simulate realistic human responses to social engineering attacks: A case study. *International Journal of Cybersecurity Intelligence & Cybercrime* 6, 2 (2023), 21–49.

[4] Sultan Asiri, Yang Xiao, Saleh Alzahrani, and Tieshan Li. 2024. PhishingRTDS: A real-time detection system for phishing attacks using a Deep Learning model. *Computers & Security* 141 (2024), 103843.

[5] Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)* 57, 1 (1995), 289–300.

[6] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security.* 2154–2156.

[7] Santosh Kumar Birthriya, Priyanka Ahlawat, and Ankit Kumar Jain. 2025. Detection and Prevention of Spear Phishing Attacks: A Comprehensive Survey. *Computers & Security* (2025), 104317.

[8] Ravi Chauhan, Ulya Sabeel, Alireza Izaddoost, and Shahram Shah Heydari. 2021. Polymorphic adversarial cyberattacks using WGAN. *Journal of Cybersecurity and Privacy* 1, 4 (2021), 767–792.

[9] Lin Chin-Yew. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out, 2004.*

[10] Asaf Cidon, Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin. 2019. High precision detection of business email compromise. In *28th USENIX Security Symposium (USENIX Security 19).* 1291–1307.

[11] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).

[12] Gordon V. Cormack and Thomas R. Lynam. 2007. *TREC 2007 Public Corpus.* Retrieved March, 2025 from https://plg.uwaterloo.ca/~gvcormac/treccorpus07/

[13] CEAS 2008 Public Corpus. 2008. *CEAS 2008 Live Spam Challenge Laboratory corpus.* Retrieved March, 2025 from https://plg.uwaterloo.ca/~gvcormac/ceascorpus/

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers).* 4171–4186.

[15] Morten W Fagerland, Stian Lydersen, and Petter Laake. 2013. The McNemar test for binary matched-pairs data: mid-p and asymptotic are better than exact conditional. *BMC medical research methodology* 13 (2013), 1–8.

[16] Rudolf Flesch. 1979. How to write plain English. *University of Canterbury. Available at http://www. mang. canterbury. ac. nz/writing_guide/writing/flesch. shtml.[Retrieved 5 February 2016]* (1979).

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[18] Bryce Goodman and Seth Flaxman. 2017. European Union regulations on algorithmic decision-making and a "right to explanation". *AI magazine* 38, 3 (2017), 50–57.

[19] Qazi Emad ul Haq, Muhammad Hamza Faheem, and Iftikhar Ahmad. 2024. Detecting Phishing URLs Based on a Deep Learning Approach to Prevent Cyber-Attacks. *Applied Sciences* 14, 22 (2024), 10086.

[20] Julian Hazell. 2023. Spear phishing with large language models. *arXiv preprint arXiv:2305.06972* (2023).

[21] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654* (2020).

[22] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. 2011. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence.* 43–58.

[23] Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America* 62, S1 (1977), S63–S63.

[24] Taeri Kim, Noseong Park, Jiwon Hong, and Sang-Wook Kim. 2022. Phishing url detection: A network-based approach robust to evasion. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security.* 1769–1782.

[25] Takashi Koide, Naoki Fukushi, Hiroki Nakano, and Daiki Chiba. 2024. Chatspamdetector: Leveraging large language models for effective phishing email detection. *arXiv preprint arXiv:2402.18093* (2024).

[26] Xue Li, Dongmei Zhang, and Bin Wu. 2020. Detection method of phishing email based on persuasion principle. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Vol. 1. IEEE, 571–574.

[27] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[28] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[29] Zhijun Liu, Weili Lin, Na Li, and David Lee. 2005. Detecting and filtering instant messaging spam-a global and personalized approach. In *1st IEEE ICNP Workshop on Secure Network Protocols, 2005.(NPSec).* IEEE, 19–24.

[30] Theodore Tangie Longtchi, Rosana Montañez Rodriguez, Laith Al-Shawaf, Adham Atyabi, and Shouhuai Xu. 2024. Internet-based social engineering psychology, attacks, and defenses: A survey. *Proc. IEEE* (2024).

[31] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30 (2017).

[32] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2009. Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining.* 1245–1254.

[33] Quinn McNemar. 1947. Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages. *Psychometrika* 12, 2 (1947), 153–157. doi:10.1007/BF02295996

[34] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. 2006. Spam filtering with naive bayes-which naive bayes?. In *CEAS*, Vol. 17. Mountain View, CA, 28–69.

[35] Michael P. Fay. 2010. Two-sided Exact Tests and Matching Confidence Intervals for Discrete Data. *R Journal* 2, 1 (2010), 53–58. https://journal.r-project.org/

[36] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[37] Jose Nazario. 2005. *The online phishing corpus.* Retrieved March, 2025 from https://monkey.org/~jose/phishing/

[38] Denish Omondi Otieno, Akbar Siami Namin, and Keith S Jones. 2023. The application of the bert transformer model for phishing email classification. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC).* IEEE, 1303–1310.

[39] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[40] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology.* 1–22.

[41] European Parliament and Council of the European Union. 2024. Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending Regulations (EU) 2018/1724 and (EU) 2022/2554. Official Journal of the European Union, L 168, 1–120. https://eur-lex.europa.eu/eli/reg/2024/1689/oj Article 86: Right to explanation of individual decision-making.

[42] Apache SpamAssassin Project. 2006. *SpamAssassin public mail corpus.* Retrieved March, 2025 from https://spamassassin.apache.org/old/publiccorpus/

[43] Proofpoint. 2024. *2024 State of the Phish: Risky Actions, Real-World Threats, and User Resilience in an Age of Human-Centric Cybersecurity.* Technical Report. Proofpoint.

[44] Tingrui Qiao, Caroline Walker, Chris W Cunningham, and Yun Sing Koh. [n. d.]. Thematic-LM: a LLM-based Multi-agent System for Large-scale Thematic Analysis. In *THE WEB CONFERENCE 2025.*

[45] R Core Team. 2024. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/ Function `binom.test` in package `stats`.

[46] Dragomir Radev. 2008. *CLAIR collection of fraud email.* Retrieved March, 2025 from http://aclweb.org/aclwiki ADCR2008T001.

[47] Fariza Rashid, Nishavi Ranaweera, Ben Doyle, and Suranga Seneviratne. 2025. LLMs are one-shot URL classifiers and explainers. *Computer Networks* 258 (2025), 111004.

[48] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining.* 1135–1144.

[49] Frank Rosner, Alexander Hinneburg, Michael Röder, Martin Nettling, and Andreas Both. 2014. Evaluating topic coherence measures. *arXiv preprint arXiv:1403.6397* (2014).

[50] Sayak Saha Roy, Poojitha Thota, Krishna Vamsi Naragam, and Shirin Nilizadeh. 2024. From chatbots to phishbots?: Phishing scam generation in commercial large language models. In *2024 IEEE Symposium on Security and Privacy (SP).* IEEE, 36–54.

[51] Suranjana Samanta and Sameep Mehta. 2017. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812* (2017).

[52] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[53] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems* 36 (2023), 68539–68551.

[54] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[55] Mohsen Sharifi and Seyed Hossein Siadati. 2008. A phishing sites blacklist generator. In *2008 IEEE/ACS international conference on computer systems and applications.* IEEE, 840–843.

[56] Yoav Shoham and Kevin Leyton-Brown. 2008. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations.* Cambridge University Press.

[57] Richard S Sutton, Andrew G Barto, et al. 1998. *Reinforcement learning: An introduction.* Vol. 1. MIT press Cambridge.

[58] Verizon. 2024. *2024 Data Breach Investigations Report.* Technical Report. Verizon.

[59] Rakesh Verma and Nabil Hossain. 2013. Semantic feature selection for text with application to phishing email detection. In *international conference on information security and cryptology.* Springer, 455–468.

[60] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[61] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155* (2023).

[62] Lei Xu, Alfredo Cuesta-Infante, Laure Berti-Equille, and Kalyan Veeramachaneni. 2021. R&R: Metric-guided adversarial sentence generation. *arXiv preprint arXiv:2104.08453* (2021).

[63] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* 32 (2019).

A Prompts

```
You are a cybersecurity expert specializing in phishing, with a particular focus on URLs within emails. Your task is to carefully examine every URL in the email and determine whether it exhibits suspicious characteristics or signs of obfuscation, such as resembling forged bank sites, using unknown domains, or appearing unusually shortened. Do not analyze the email text or metadata, only focus on the URLs. Provide your judgment on whether the email is 'Phishing' or 'Legitimate', along with a confidence score between 0 and 1 and a clear, concise explanation of your reasoning. Output your result in JSON format as: 'verdict': 'Phishing' or 'Legitimate', 'confidence': 0-1, 'reasons': '...'
```

Figure 6: URL Agent's Prompt

```
You are a cybersecurity expert specializing in phishing, with a particular focus on email metadata. Your task is to scrutinize the provided email header—including the subject, sender address, reply-to, return-path, and received fields—for any signs of forgery, anomalies, or suspicious indicators. Do not analyze email text or URLs, only focus on the metadata. Provide your judgment on whether the email is 'Phishing' or 'Legitimate', along with a confidence score between 0 and 1 and a clear, concise explanation of your reasoning. Output your result in JSON format as: 'verdict': 'Phishing' or 'Legitimate', 'confidence': 0-1, 'reasons': '...'
```

Figure 7: Metadata Agent's Prompt

B Example result of adversarial agent email variant generation

From: "Monkey Support Notification" <info@creditloiuse.com>
Subject: Important Password Validation
To: <jose@monkey.org>
Date: Wed, 10 Jan 2024 16:00:51 -0800

Monkey Support Notification

ACTION REQUIRED: Monkey (jose@monkey.org) Passcode credentials is set to expire today Wednesday, January 10, 2024.

Same registered password must be used below.

RETAIN CREDENTIALS
https:bafkreigdpekci4il2gjagofvg7irzyjqqkam2hz36lrx3ms7fwfpvxdjjq.ipfs.dweb.link/#jose@monkey.org

Note: It is important you update your email passcode, failure to do this will get your account locked out and email will not be accessible.

Update your email preferences to choose the types of emails you receive, or you can unsubscribe from all future emails.

(a) Original Email

From: "Monkey Assistance Notice" <info@creditloiuse.com>
Subject: Vital Password Confirmation
To: <jose@monkey.org>
Date: Wed, 10 Jan 2024 16:00:51 -0800

Greetings,

We hope this email finds you well.

MONKEY NOTICE: The passkey linked to your profile (jose@monkey.org) will reach its expiration this Wednesday, January 10, 2024.

Please ensure you are utilizing the same registered password using the link below.

ACCESS CREDENTIALS:
https://bafkreigdpekci4il2gjagofvg7irzyjqqkam2hz36lrx3ms7fwfpvxdjjq.ipfs.dweb.link/#jose@monkey.org

Please be advised: It's crucial to confirm your email password soon to avoid any disruptions in accessing your email.

You are welcome to update your email settings for receiving different categories of emails, or you can opt out from all subsequent communications.

(b) Adversarial Agent–Created Variant

Figure 8: Example result of adversarial agent email variant generation.

C Evaluation Metrics

C.1 Evaluation Metrics for Classification

- **True Positives (TP):** The number of phishing emails correctly identified as phishing.
- **True Negatives (TN):** The number of legitimate emails correctly classified as legitimate.
- **False Positives (FP):** The number of legitimate emails mistakenly classified as phishing.
- **False Negatives (FN):** The number of phishing emails incorrectly classified as legitimate.

Using these fundamental values, we calculate the following evaluation metrics:

- **Recall:** Measures how well the model identifies phishing emails.
$$\text{Recall} = TPR = \frac{TP}{TP + FN}$$
A high recall ensures that most phishing emails are detected, reducing the chances of missed attacks.
- **Precision:** Measures how many of the emails classified as phishing are actually phishing.
$$\text{Precision} = \frac{TP}{TP + FP}$$
A higher precision means fewer legitimate emails are mistakenly flagged as phishing, reducing false alarms.
- **Accuracy:** Measures the overall correctness of the model's classifications.
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$
A high accuracy indicates strong overall performance, but it may not be reliable when the dataset is imbalanced.
- F_1**Score:** The harmonic mean of precision and recall, balancing both metrics to provide a single performance measure.
$$F_1\text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$
A high F_1score indicates that the model performs well in both phishing detection and avoiding false alarms.
- **True Negative Rate (TNR):** Measures the proportion of legitimate emails correctly classified.
$$\text{TNR} = \frac{TN}{TN + FP}$$
A high TNR ensures that the system does not mistakenly flag too many legitimate emails as phishing.
- **False Positive Rate (FPR):** Measures the proportion of legitimate emails incorrectly classified as phishing.
$$\text{FPR} = \frac{FP}{FP + TN}$$
A lower FPR is desirable, as it reduces unnecessary phishing alerts and minimizes disruptions to users.
- **False Negative Rate (FNR):** Measures the proportion of phishing emails incorrectly classified as legitimate.
$$\text{FNR} = \frac{FN}{TP + FN}$$
A low FNR is critical, as it minimizes the risk of phishing emails bypassing detection and reaching users.

C.2 Evaluation Metrics for Rationales

- **Perplexity (PPL):** Perplexity [23] is a widely used metric in natural language processing to evaluate the fluency and predictability of text generation. It quantifies how well a language model predicts the next word in a sequence, with lower perplexity indicating more coherent and fluent explanations.
$$\text{PPL} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log p(w_i)\right)$$
N is the number of words in the explanation, and $p(w_i)$ is the probability assigned to each word. A lower perplexity score suggests that the explanation is more readable and natural, while a higher perplexity indicates disjointed or unnatural phrasing.
- **Topic Coherence:** Topic Coherence [49] evaluates the semantic consistency of topics within the explanations, ensuring they are interpretable and logically structured. It measures the degree of semantic similarity among words within a topic, reflecting the interpretability of the explanations. Higher coherence scores indicate that the explanations are more logically structured and easier to understand. This ensures that the explanations are not only accurate but also presented in a manner that is coherent and user-friendly.
- **Flesch Reading Ease Score (FRES):** The Flesch Reading Ease Score [16] is a widely used metric to evaluate the readability of a text. Developed by Rudolf Flesch, it assigns a score between 0 and 100, with higher scores indicating easier readability. The formula for calculating this score is:
$$FRES = 206.835 - 1.015 \times \left(\frac{\text{Total Words}}{\text{Total Sentences}}\right) - 84.6 \times \left(\frac{\text{Total Syllables}}{\text{Total Words}}\right)$$
In the context of evaluating explanations, applying the Flesch Reading Ease Score can help assess how easily readers can comprehend the provided reasons. A higher score suggests that the explanation is straightforward and accessible, while a lower score may indicate complexity that could hinder understanding. For instance, explanations laden with technical jargon, lengthy sentences, or complex words are likely to yield lower readability scores, signaling the need for simplification to enhance clarity.
- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** ROUGE [9] quantifies the overlap of n-grams between the candidate text (MultiPhishGuard's explanation) and the reference text (the expert's analysis), allowing us to measure how much of the expert's key content is captured by our model. We focus on ROUGE-1, which computes unigram recall:
$$\text{ROUGE-1 Recall} = \frac{\sum_{w \in R} \min(\text{Count}(w, C), \text{Count}(w, R))}{\sum_{w \in R} \text{Count}(w, R)}$$
where $\text{Count}(w, C)$ and $\text{Count}(w, R)$ are the frequencies of word w in the candidate and reference texts, respectively. A high ROUGE-1 score indicates that the system's output includes a large proportion of the expert's important unigrams, which is desirable because it shows that the generated explanation covers the essential content. We choose ROUGE-1

over higher-order variants (ROUGE-2, ROUGE-L, etc.) because expert explanations and LLM-generated outputs often differ in phrasing and sentence structure. Unigram overlap provides a robust, style-agnostic measure of content similarity, whereas bigram or sequence-based metrics can unfairly penalize legitimate rewordings and stylistic variations. By using ROUGE-1, we ensure a balanced evaluation that emphasizes the presence of critical terms without over-penalizing differences in natural human versus model-generated language.

- **Cosine Similarity:** Cosine similarity measures the semantic similarity between two texts by converting them into vector representations (e.g., using sentence embeddings) and computing the cosine of the angle between these vectors. It is defined as:

$$\text{Cosine Similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

where $\mathbf{A}$ and $\mathbf{B}$ represent the expert's and the model's explanation embeddings, respectively. Values close to 1 indicate that the two texts are semantically very similar, meaning the system's explanation has captured the overall meaning and context of the expert's analysis. Unlike ROUGE—which focuses on exact word overlap—cosine similarity captures deeper, conceptual alignment. This allows our explanations to retain semantic similarity and contextual relevance to the expert's intent, even when the wording differs.