PiMRef: Deducing Ever-evolving Spear-phishing Emails with Knowledge Base Invariants

Ruofan Liu
liu.ruofan16@u.nus.edu
National University of Singapore
Singapore, Singapore

Yun Lin*
lin_yun@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Yuxin Wang
w_yuxin@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Xiwen Teoh
xiwen.teoh@u.nus.edu
National University of Singapore
Singapore, Singapore

Zhenkai Liang
liangzk@comp.nus.edu.sg
National University of Singapore
Singapore, Singapore

Gongshen Liu
lgshen@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Haojin Zhu
zhu-hj@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Jin Song Dong
dcsdjs@nus.edu.sg
National University of Singapore
Singapore, Singapore

Abstract

Phishing email is a critical step in the cybercrime kill chain due to the high reachability of victims' email accounts and the low cost of launching phishing campaigns. The proportion of AI-generated phishing emails peaked at 82.6% in 2025, and showed a higher click-through rate than manually written ones. This ever-evolving nature of phishing emails makes traditional rule-based and feature-engineering-based phishing email detectors fight an uphill battle in the cat-and-mouse game of defense and attack.

In this work, we show that, large language models (LLMs) can be effectively exploited to generate profile-grounded spear-phishing, compromising major paradigms of phishing email detectors. To defend against such LLM-based spear-phishing attacks, we propose PiMRef, the first reference-based solution to detect ever-evolving phishing emails using knowledge-based invariants, targeting the identity-impersonation attacks that characterize spear-phishing. Our rationale lies in the fact that convincing phishing emails often include "disprovable claims", which contradict certain real-world facts. Therefore, we reduce the problem of phishing email detection to an identity fact-checking problem on the sender's identity within the email context, enabling defenses against evolving phishing threats with high accuracy and explainability. Technically, given an email, PiMRef (i) discovers the claimed identity of the sender, (ii) verifies the sender's email domain against a dynamically expandable knowledge base, and (iii) infers call-to-action instructions that encourage next-step engagement. The detected contradictory identity facts serve as both alarms and explanations.

Compared to existing baselines such as D-Fence, HelpHed, and ChatSpamDetector, *PiMRef* reduces the false-positive rate to 0.81% while maintaining a recall of 90.7%-93.1% on conventional phishing benchmarks such as Nazario and PhishPot. On *SpearMail*, our newly constructed benchmark of 14,672 LLM-generated spear-phishing emails targeting 681 public profiles, *PiMRef* reaches a recall of 86.4% without incurring additional false positives. Furthermore, on 12,289 real-world emails collected from university accounts, spam feeds, and honeypots, *PiMRef* achieves a recall of 80.5%–87.1% on wild phishing emails, with a median runtime of 0.05 seconds, outperforming state-of-the-art phishing email detectors. our code is publicly available at https://github.com/code-philia/PhishEmail.

CCS Concepts

• **Security and privacy** → **Spam and phishing**; *Malware and its mitigation*; • **Computing methodologies** → Information extraction.

Keywords

phishing detection, spear-phishing, email security, large language models

ACM Reference Format:
Ruofan Liu, Yun Lin, Yuxin Wang, Xiwen Teoh, Zhenkai Liang, Gongshen Liu, Haojin Zhu, and Jin Song Dong. 2026. PiMRef: Deducing Ever-evolving Spear-phishing Emails with Knowledge Base Invariants. In *Proceedings of the 2026 ACM SIGSAC Conference on Computer and Communications Security (CCS '26), November 15–19, 2026, The Hague, Netherlands.* ACM, New York, NY, USA, 20 pages. https://doi.org/10.1145/3830454.3832577

1 Introduction

Large Language Models are a double-edged sword. While they help users draft emails and summarize documents, the same capability also makes phishing attacks easier to launch and harder to recognize. Attackers can now generate high-fidelity, personalized forgeries in seconds with little effort. Recent evidence shows that LLMs are already being used to produce spear-phishing emails at scale. An industry report finds that 82.6% of phishing emails contain AI-generated content [64], and a controlled human-subject study

*Corresponding author.

CCS '26, November 15–19, 2026, The Hague, Netherlands

ACM ISBN 979-8-4007-2871-6/2026/11
https://doi.org/10.1145/3830454.3832577

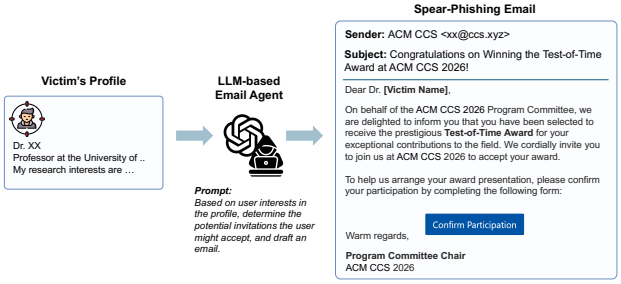


Figure 1: Given the victim's profile, an attacker can construct a prompt to generate a spear-phishing email without bypassing LLM safeguards. We show that such LLM-generated spear-phishing emails can evade the practically deployable detectors, spanning three major design paradigms (feature-engineering-based, LLM-based, and rule-based) (see Section 6.1).

reports that fully AI-automated spear-phishing achieves a 54% click-through rate, 4.5× higher than generic campaigns [47]. To combat such phishing campaigns, existing research on phishing email detection has mainly advanced along two lines: *infrastructure-based authentication* and *content-based detection*.

Infrastructure-based defenses, including SPF [63], DKIM [33], and DMARC [66], can verify whether an email is sent from an IP address authorized by the claimed sending domain. However, they do not inspect whether the email content itself is deceptive. As a result, an attacker can simply send a phishing email from an attacker-controlled but properly authenticated domain, thereby passing all infrastructure-level checks.

Content-based defenses, such as RSpamd [102], SpamAssassin [104], and Trend Micro [106], together with machine-learning-based detectors [6, 23, 24, 30, 35, 39, 41, 44, 45, 49, 50, 60, 61, 71, 72, 80, 99, 100, 103], inspect the email body, header, URL, sender reputation, or other handcrafted and learned features to classify whether an email is phishing. These approaches are effective when phishing emails contain recognizable features, such as suspicious wording, abnormal headers, or other statistical patterns that differ from legitimate emails. However, with LLMs, attackers can easily produce emails with content which is (1) **evasive**, i.e., not exhibit the linguistic features used by traditional content-based detectors and (2) **targeting**, i.e., more catering to victims' unique interest to improve the attack success rate.

As illustrated in Figure 1, given a public victim profile, an attacker can infer the victim's interests and generate an invitation email tailored to the victim by LLMs, then replace the link as a fake website. Our experiments confirm that the evaluated detectors struggle under this new threat model (Table 2). Importantly, this attack does not require explicit jailbreak prompts. From the LLM's perspective, the attacker is merely asking it to draft a benign, personalized invitation email, making the generation process difficult to block using jailbreak-oriented safety filters.

In this work, we present *PiMRef* (**Phi**shing **M**ail Detection by **Ref**erence), a reference-based phishing email detector. Different from traditional approaches, *PiMRef* reformulates phishing detection as a fact-checking problem in the email context. The key rationale is that LLM-generated phishing emails can still make claims that are falsified by external facts. The contradictions between the claims and the facts provide a more stable signal than historical phishing features. Therefore, rather than *inductively* learning what phishing emails used to look like, *PiMRef deductively* detects phishing emails by verifying the consistency between their claims and relevant real-world references. However, a naive fact-checking solution is impractical as an email may contain many factual claims, and verifying all of them would be computationally expensive. To make reference-based detection practical, *PiMRef* focuses on the most inexpensive and effective type of claim, i.e., the *sender identity claim*. This claim is highly practical because phishing emails commonly rely on impersonating a trusted sender to establish credibility, which can be efficiently verified against a knowledge base that maps legitimate identities to their official email domains.

To this end, we design *PiMRef* to (1) extract identity claims from an email via a named-entity-recognition model over the email header and body, and (2) verify the claimed identity through typo-robust embedding matching against a self-evolving identity knowledge base maintained by an LLM-based web search agent. As a result, for the phishing email in Figure 1, *PiMRef* generates the following explanation: "*This email is flagged as phishing because it claims to be from ACM CCS but was sent from a non-official address as xx@ccs.xyz*".

We evaluate the performance of *PiMRef* on both conventional phishing benchmarks (i.e., Nazario and PhishPot) and *SpearMail*, a benchmark consisting of 14,672 phishing emails generated from 681 public profiles by LLM. We compare *PiMRef* against academic baselines (i.e., D-Fence [71], HelpHed [23], and ChatSpamDetector [65]) and commercial baselines (i.e., RSpamd [102], SpamAssassin [104]). Our results show that *PiMRef* reduces the false-positive rate to 0.81% while maintaining a recall of 90.7%-93.1% on conventional phishing benchmarks such as Nazario and PhishPot. And it significantly boosts recall to 86.4% on the *SpearMail* dataset. Additionally, we conducted an open-world study, analyzing 12,289 real-world emails collected over two years. The study revealed that *PiMRef* achieves a precision of 90.29% and a recall of 80.5%-100% on real-world emails, while maintaining an efficient runtime of 0.05 seconds, significantly outperforming the baseline solutions. In summary, our contributions are as follows:

- **Methodology**: To the best of our knowledge, we introduce the first reference-based phishing email detecting methodology to deductively detect and explain phishing emails, which is robust to problems of rule degradation and distribution shifts.
- **Benchmark**: We show that LLM-empowered spear-phishing emails are realistic threats to commercial and academic anti-phishing solutions, spanning three major design paradigms. To this end, we construct the *SpearMail* dataset, containing 14,672 spear-phishing emails targeting 681 public profiles, which we use to reveal the vulnerabilities of existing defenses.
- **Tool**: We deliver the *PiMRef* tool, along with its Outlook plugin version, which can be practically deployed to scan large volumes of incoming emails. The code [1] and video [12] are available online.

[1] https://github.com/code-philia/PhishEmail

- **Evaluation**: We conduct extensive evaluations on both closed-world and open-world experiments, on 12,289 emails over two years, showing that *PiMRef* can significantly improve the precision and recall over baselines.

2 Related Work

Phishing Email Detection. Employees remain highly susceptible to phishing despite security training [26, 67–69], making automatic defenses crucial. Infrastructure-level protocols (SPF, DKIM, DMARC) [33, 63, 66] prevent domain-level spoofing but miss targeted or compromised-account attacks [18, 97]. Content-based detection instead inspects the email itself, spanning anomaly detection and hand-crafted rules [35, 39, 70, 100], feature-engineered classifiers over body, URLs, and sender reputation [23, 30, 46, 49, 50, 71], detectors built on psychological and persuasive cues [36, 96, 98, 107, 108], and industrial spam filters [102, 104]. Recent work adds hybrid detectors that combine learned classifiers with LLMs [40], LLM-based judges that reason over the email content [65], and Heiding et al. [48], who evaluate LLMs on both spear-phishing generation and malicious-intent judging. All fall into the same inductive paradigm, learning or eliciting what phishing typically looks like. Such ML-based detectors can be evaded by adversarial perturbations [16, 17], and LLMs escalate the threat further. They can be repurposed to generate phishing content at scale [95], drive web-enabled agents that autonomously harvest personal data for targeted lures [62], and produce emails that match human experts in click-through rate [47] and rival human-crafted attacks in large-scale organizational deployments [19]. Even reporting pipelines can be undermined through email-tracking-based evasion [28], so robust content-based detection at the inbox remains a necessary line of defense.

Reference-based Phishing Detection (RBPD). RBPD checks a message against a trusted reference set rather than inspecting it in isolation. In the URL and website setting, RBPD systems maintain logo-domain pairs and flag a page when the displayed brand does not match the actual domain [7–9, 73, 75–78], including lightweight client-side variants [94]. This does not transfer to email, since around 35% of phishing emails in PhishPot [13] contain no URL, and sender identity is expressed through names, addresses, signatures, and free-form text rather than logos, requiring text encoders and efficient matching rather than computer vision. *PiMRef* is, to our knowledge, the first reference-based phishing detector for email, addressing ambiguous identity descriptions, description-domain mapping, and runtime throughput with two encoder-based language models.

3 Threat Model

Phishing emails in this work are emails whose sender *deceives* the recipient into believing they are someone else, with the intent of compelling the recipient to take a specific action such as replying, clicking an unsafe link, or calling a phone number. We do not consider general spam such as advertisements or promotions. Our primary target is *spear-phishing*, where the impersonated identity is personalized to the victim to gain trust. The same verification applies to generic phishing emails that carry an identity claim, and our evaluation covers both (Section 6.1).

Content-based impersonation. We focus on attackers who use attacker-controlled accounts or domains to imitate a sender identity in the email content, without forging or compromising the legitimate domain. This is a deliberate scope choice, as domain spoofing is an authentication problem best mitigated by strict enforcement of SPF/DKIM/DMARC [33, 63, 66]. An attacker who spoofs the exact official domain of the claimed identity produces a consistent identity-domain pair and raises no alarm in *PiMRef*, but such an email fails SPF/DKIM alignment at the receiving server and is rejected or quarantined under a DMARC enforcement policy before content analysis applies. *PiMRef* therefore assumes sender authentication as a deployed lower layer and detects the complementary class of attacks that pass authentication legitimately. Business Email Compromise likewise involves compromised or domain-consistent accounts and requires signals beyond content-only analysis, such as sender-history anomalies and workflow controls [30, 49, 105]. In our observations on PhishPot [4], an open phishing-email repository, **90%** of phishing emails fall into this content-based setting and only **10%** forge the sender address. Content-based impersonation is also operationally cheaper, requiring only email generation rather than compromising any infrastructure.

Non-triviality. We consider a phishing email non-trivial when the attacker both instructs the victim to take a specific action and claims an identity, in the header or the content, to establish trust. Spear-phishing emails usually satisfy this. An email claiming to be UPS [109] and saying "*Track your parcel here to ensure delivery*" is non-trivial, whereas "*Hi, are you available?*" is trivial because it specifies no identity. Attackers may further use web crawlers and LLMs to craft the title, sender identity, and body according to their malicious intent.

4 Approach

Overview. Figure 2 presents the workflow of *PiMRef*, which takes an email as input, and outputs a phishing alert along with its counterfactual explanation if the email is phishing. *PiMRef* consists of three modules, i.e., claimed identity recognition, knowledge-grounded identity verification, and a call-to-action instruction recognition module.

- **Claimed Identity Recognition** (Section 4.2): This module parses the email subject, sender name, and email body as input to infer the claimed sender identities ID_{rec}.
- **Knowledge-Grounded Identity Verification** (Section 4.3): This module maps the predicted identities ID_{rec} to their legitimate official email domains $\mathcal{D}_{rec}$, using an identity-domain knowledge base.
- **Instruction Recognition** (Section 4.5): This module outputs the set of phrases of call-to-action instructions *inst*, if any.

Then, we verify the consistency between the expected official email domains $\mathcal{D}_{rec}$ and the domain of the sender's email address in the target email. Specifically, a phishing alert is raised if (i) the actual email domain $d \notin \mathcal{D}_{rec}$ (i.e., d is not a legitimate domain of any claimed identity) and (ii) $inst \neq \emptyset$ (i.e., the email contains instructions for next-step engagement).

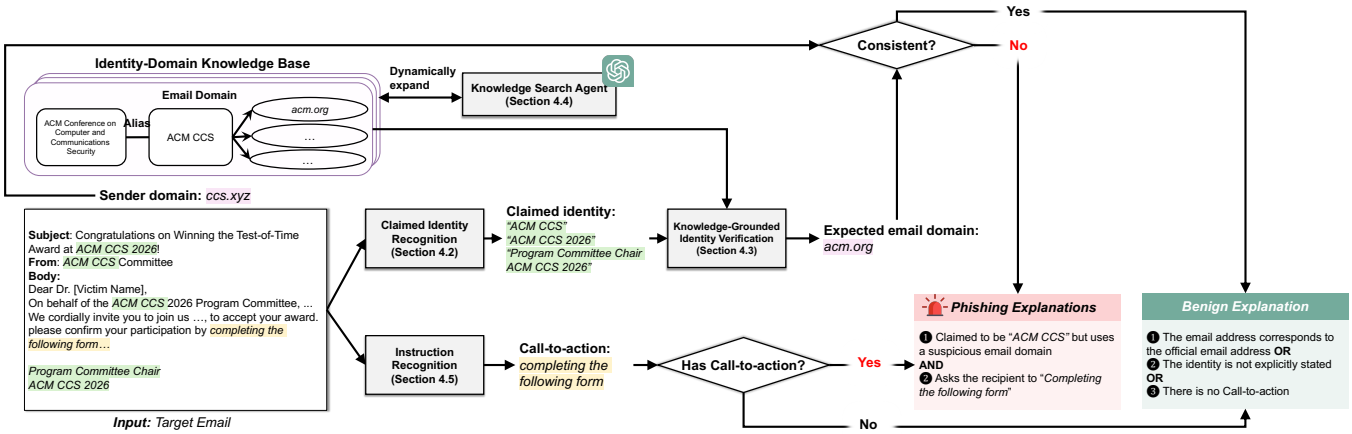


Figure 2: Overview of *PiMRef*. The *Claimed Identity Recognition* module first extracts the phrases claiming the identity. The *Knowledge-Grounded Identity Verification* module then converts the identity-claiming phrases into their expected email domains (e.g., *acm.org*), based on a predefined *Identity-Domain Knowledge Base*. Finally, the *Instruction Recognition* module extracts the phrases of call-to-action instruction in the email. *PiMRef* reports the phishing alert if the actual email domain is different from all of the expected email domains, and the email has call-to-action instructions.

4.1 Email Pre-processing

We first render the email body (including HTML content and common attachments) into PDF format, on which we run OCR [3] to recognize the textual content. This design makes our pipeline effectively immune to common text-salting attacks [101] (e.g., zero-width characters or non-printable characters) that target tokenization at the raw text level. Then we tokenize each preprocessed email $m = \{name, subject, body\}$ into sequences of subword tokens using a standard tokenizer [34]. Formally, $name = \langle t_{i_1}, t_{i_2}, \dots \rangle$ denotes the sender name, $subject = \langle t_{j_1}, t_{j_2}, \dots \rangle$ denotes the email subject, and $body = \langle t_{k_1}, t_{k_2}, \dots \rangle$ denotes the message body.

4.2 Claimed Identity Recognition

In this step, we infer all token spans in m which indicate the claimed identities of the sender. For example, in Figure 1, we report terms such as *"ACM CCS 2026"*.

Naive Solution and its Practical Challenge. Prompt engineering a state-of-the-art LLM [86] is common practice for general NLP problems, but it scales poorly here. Autoregressive decoding makes per-email inference slow, and paying per API call for every email is costly, so the naive solution is impractical for an organization parsing tens of thousands of emails a day.

Our Solution. We propose an efficient and lightweight solution to recognize the identity of the sender. We adopt an encoder-based solution which is smaller in size (e.g., 340M parameters), but can process all the tokens in an email in parallel, empirically incurring the average runtime overhead of only 0.02 seconds (see Section 6.1 for more details). Specifically, we reframe the identity recognition task as a Named Entity Recognition (NER) problem [82].

Design of Model and Dataset. Overall, our NER model takes a sequence of tokens as input, and assigns each token with one of the following classes, i.e., *BE* (i.e., the beginning of entity), *IE* (i.e., the inside of an entity), and *O* (i.e., the outside), following the practice

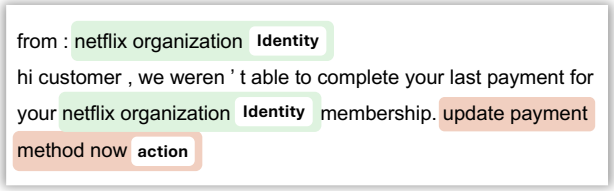


Figure 3: Example of NER training samples.

of training any NER models. In this work, we prepare a dataset of 2,086 emails with labeled entities (from the Nazario and Enron corpora, see Section 6.1) as shown in Figure 3, with the state-of-the-art labeling tool, Label Studio [1]. Then, we use the focal loss [74] in Equation 1 to train our model. Here, i indexes each token in the email, and p_{y_i} denotes the predicted probability of the ground-truth class for token i. Focal loss is well-suited for class imbalance tasks, where negative tokens (O class) outnumber positive tokens (I or B classes).

$$\mathcal{L}_{\text{Focal}} = -\frac{1}{N} \sum_{i=t_1}^{i=t_N} (1 - p_{y_i})^{\gamma} \log(p_{y_i}) \tag{1}$$

Finally, we augment the training dataset by mutating the phrases of identities (with character-level perturbation) to improve model robustness.

4.3 Knowledge-Grounded Identity Verification

Problem Statement. We maintain a knowledge base $KB = \{kb_1, kb_2, \dots\}$, where each entry $kb_i = \langle id, d \rangle$ maps an identity $id \in ID$ to one of its legitimate email domains $d \in \mathcal{D}$, so an identity may appear in multiple entries, and ID and $\mathcal{D}$ denote all known identities and all their legitimate domains. Given the identity-claiming

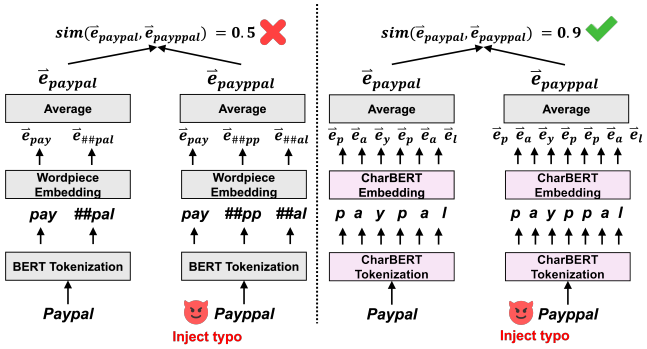


Figure 4: Comparison between BERT (LHS) and CharacterBERT (RHS) when encountering the typo injection attack.

phrases $ID_{rec} = \{id_1, id_2, \dots\}$ extracted in Section 4.2, our task is to retrieve $\mathcal{D}_{rec} = \{\, d \mid \langle id, d\rangle \in KB,\ id \text{ matches some } id_j \in ID_{rec} \,\} \subseteq \mathcal{D}$, the set of legitimate domains associated with ID_{rec}.

Technical Challenge. Two difficulties arise. First, an attacker may fake an *external* identity, an organization outside the recipient's own, or an *internal* one, and internal claims often name no specific identity at all (e.g., "*Dear colleague, ...*") yet still require legitimate domains for validation. Second, exact matching is too restrictive because identity names carry intentional or unintentional typos [24], while edit distance captures string overlap but not semantics, as "payppall" is a typo of "paypal" whereas the comparably distant "payday" is a different word.

Our Solution. We learn an embedding model $f(\cdot) : ID \to \mathbb{R}^k$ from identity phrases in natural language to a k-dimensional space, projecting semantically similar phrases closer than dissimilar ones. We adopt CharacterBERT [22, 111], which tokenizes at the character level rather than the token level, so unaffected characters remain intact under typos. Figure 4 illustrates this, where BERT's tokenization of "PayPal" is significantly altered by typos while CharacterBERT preserves the tokenization of the unaffected characters and yields more robust embeddings. We pre-train on the dataset of [111] with the loss below, whose first term is a retrieval loss enforcing that the query q stays close to its true neighbor set p^+ within the candidate set $\mathcal{P}$, and whose second term is an auxiliary KL divergence requiring a typo-ed q' to yield the same prediction as q.

$$\mathcal{L} = \underbrace{-\log \frac{e^{\left(f(q)^\top f(p^+)\right)}}{\sum_{p\in\mathcal{P}} e^{\left(f(q)^\top f(p)\right)}}}_{\mathcal{L}_{\text{Retrieval}}} + \underbrace{D_{KL}\left(f(q')^\top f(\mathcal{P}) \| f(q)^\top f(\mathcal{P})\right)}_{\mathcal{L}_{\text{KL}}} \tag{2}$$

This resolves both challenges together. Identity phrases such as *PayPal Inc* or *Colleague* map to knowledge base identities such as *Paypal* or the special identity *Internal* that we introduce for internal claims. For an external identity we verify that the **official email domain** is consistent with the **sender's email domain**; for an internal one we verify that the **sender's** and the **recipient's** email domains agree, confirming they belong to the same organization. The character-level architecture in turn keeps the similarity score robust to intentional or unintentional perturbation of identity phrases.

Subdomain Handling. To accommodate legitimate subdomain usage, *PiMRef* canonicalizes both the sender domain and the KB domains to their registrable domain, i.e., effective top-level domain plus one label (eTLD+1), before comparison, so a sender domain such as `cs.cmu.edu` is consistent with the KB entry `cmu.edu`. This avoids falsely flagging legitimate emails sent from departmental, regional, or service-specific subdomains of an official organization domain.

4.4 Knowledge Base Evolution

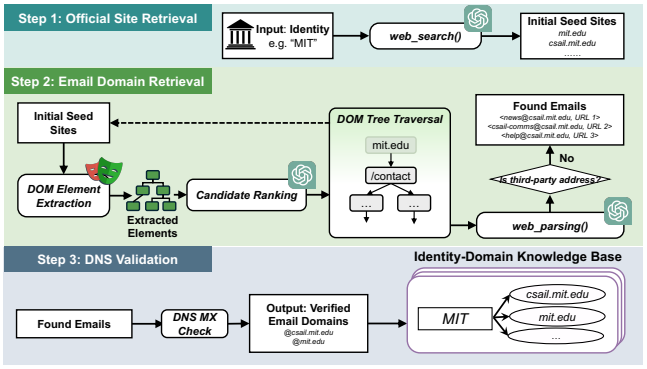


Figure 5: Knowledge Search Agent Implementation.

We develop an email domain search agent to evolve the identity-domain knowledge base. The input is an organization identity (e.g., "Citibank"), and the output is a list of official contact email domains. The agent workflow is shown in Figure 5. Throughout the workflow, the agent is powered by OpenAI o4-mini [88], which performs both the official-site retrieval in Step 1 and the page-traversal decisions in Step 2.

- **Step 1: Official Site Retrieval.** Given an identity id, we call an LLM with its built-in *web_search()* tool to retrieve a set of candidate official sites' domains $S_{id} = \{s_1, s_2, \dots\}$. In cases where id is ambiguous or corresponds to multiple entities (e.g., "Citibank" is associated with `citi.com`, `citibank.com`, and `citigroup.com`), we explicitly ask LLM to retain all candidate domains for downstream processing.
- **Step 2: Email Domain Retrieval.** For each candidate site $s \in S_{id}$, the agent performs page-level traversal to extract all listed email addresses $E_d = \{e_1, e_2, \dots\}$. We perform BFS from the homepage to extract interactive elements at each step and applying the LLM to choose which to expand. At every visited page, the agent invokes *web_parsing*(), which applies a regular expression to the page's HTML to extract email addresses. Traversal terminates upon reaching a predefined depth limit or when the LLM judges further expansion unproductive.
- **Step 3: Domain-Level DNS Validation.** We probe whether the retrieved emails E_d have valid DNS MX records. This validation

confirms that the recipient domain is operational. Operational email domains are added to the knowledge base.

Initial Knowledge Base Construction. We curate an initial identity-domain knowledge base by running our knowledge search agent on the 5,600 phishing targets reported by DynaPhish [78] and KnowPhish [73]. For each input identity, the agent retrieves an average of 3.75 candidate official sites (Step 1), traverses each site to a maximum depth of 3 (Step 2), and applies DNS-level domain validation (Step 3), with an average end-to-end runtime of 7 minutes per identity. This process yields a KB of 3,650 resolved identities mapped to 4,536 email domains. Identities without sufficient supporting evidence are left unresolved rather than forcefully mapped.

Knowledge Freshness. Knowledge base entries can become outdated as organizations migrate or retire their email domains. The construction process above can be rerun periodically offline, so entries can be re-validated and updated on a regular schedule.

4.5 Instruction Recognition

Definition. We define call-to-actions (CTAs) as phrases that prompt users to exit the current interface and perform a subsequent action, such as visiting an embedded link or QR code, replying to the email, or downloading attachments.

Our Solution. To detect such actions, the solution is similar to claimed identity recognition: we employ a single NER model to label both claimed-identity spans and action-instruction spans in one pass. However, action instructions exhibit greater lexical variety than identity phrases. For instance, "*click here*" may also appear as "*follow this link*" or "*visit this page*". Raw email datasets may not include all these variations, and manually annotating additional data is labor-intensive.

To increase diversity, we adopt a structured way to augment the annotated CTAs. Specifically, each training example has a 50% chance of being paraphrased. The paraphrase is from sentence structures and persuasion strategies aspects. We randomly select one of seven sentence structures: Imperative, Declarative-Permission, Fronted-Purpose, Conditional, Coordination, Relative-Clause, or Topic-Fronting [20, 53]. Then we apply one of six persuasion strategies [29]: Reciprocity, Consistency, Social Proof, Authority, Liking, or Scarcity. For example, the original CTA is "Click this link to verify your information". If "Conditional" and "Scarcity" are chosen, the paraphrased CTA could be: "If you do not verify your information within the next 24 hours, you may lose access to your account, so please click this link now to secure it". The augmentation is conducted automatically using GPT-4o [87] with the prompt shown in Appendix D.

5 *SpearMail* Benchmark

To evaluate phishing detectors against the emerging threat of LLM-assisted spear-phishing, we construct *SpearMail*, the largest profile-grounded spear-phishing benchmark generated against real-world researcher profiles. *SpearMail* uniquely combines profile-grounded customization with high per-victim attack diversity (30 emails per profile). Starting from 681 publicly available researcher profiles, *SpearMail* synthesizes 14,672 personalized lures that impersonate senders from real, recognizable organizations and entice each target into performing attacker-specified actions. Our generation pipeline is designed with three guiding principles: (1) *customization*, (2) *diversity* and (3) *authenticity*.

Profile Collection. We collect 681 researcher profiles from ORCID [90], spanning PhD students, research fellows, and professors across 144 disciplines. We choose researchers as the victim population for two reasons. First, researcher profiles provide a controlled and verifiable source of public victim information: their affiliations, research interests, and professional activities are often explicitly documented in structured public records. This allows us to objectively evaluate whether the generated emails are grounded in the victim's actual profile, rather than being generic phishing templates. Second, although our benchmark uses researchers, the generation pipeline is not specific to academia. It only requires public profile attributes, such as a person's role, interests, organization, and activities, which are also commonly available in other domains, such as corporate biographies, LinkedIn profiles, conference speaker pages, government directories, and professional webpages. Therefore, researcher profiles serve as a measurable and ethically manageable proxy for studying profile-grounded spear-phishing, while the underlying attack mechanism generalizes to other populations.

Email Generation. Directly prompting an LLM to produce a personalized phishing email from a raw profile tends to yield generic text that weakly references the victim. We therefore decompose generation into three reasoning steps, *interest_inference*, *activity_inference*, and *email_generation*. This forces the model to commit to a concrete recipient interest before writing. To suppress hallucinated organizations, we add a fact-checking step that verifies each generated organization exists and, when victim geolocation is available, that it is plausible for the victim's region. The full prompt templates are in Appendix B.

- **Step 1 (Interest Inference):** Given a parameter m, we infer m potential interests from p, based on the prompt *interest_inference()* in Figure 9 in Appendix B.
- **Step 2 (Activity Inference):** Given a parameter n, for each interest i, we infer n relevant activities along with their corresponding organizations to attract the user, based on the prompt *activity_inference()* in Figure 9 in Appendix B.
- **Step 3 (Email Generation):** Given the user p, a potential interest i, and a relevant activity a, we generate an email to invite p with a pseudo-link in the name of a, based on the prompt *email_generation()* in Figure 9 in Appendix B.

We set $m = 6$ interests per profile and $n = 5$ activities per interest, yielding 30 candidate emails per victim and 20,430 generated emails in total. The fact-checking step then discards 5,758 emails whose impersonated organizations are hallucinated, i.e., non-existent, leaving the final 14,672 emails impersonating 5,680 unique organizations. Consider a user profile described as a "PhD student with research interests in computational mathematics". Given this profile, one potential activity is an "internship program at Google Research". The resulting spear-phishing email examples are present in Appendix Figure 13.

Ethics Consideration. *SpearMail* is constructed entirely from publicly available ORCID profiles under ORCID's Public API Terms of Service, which permit non-commercial research use [90]. No generated email was delivered to any profiled researcher, no phishing infrastructure or live URLs were deployed, and our generation

pipeline uses only benign prompts to commercial LLM APIs within the providers' standard terms of service. **To mitigate dual-use risk, the *SpearMail* corpus will not be publicly disseminated, but will be made available to vetted researchers upon request under a data-use agreement.** A detailed ethical statement is provided in Section 8.

Benchmark Diversity. We measure benchmark diversity by applying Latent Dirichlet Allocation (LDA) topic clustering [21] to the 14,672 emails, with the number of clusters selected via the elbow method on perplexity scores. This yields 500 distinct activity clusters and 150 interest clusters spanning multiple disciplines, including urban studies, healthcare, materials science, and energy.

Benchmark Customization. To verify that *SpearMail* constitutes *spear*-phishing rather than generic phishing, we measure the profile-grounding rate of each email against the recipient's original ORCID profile. Among the 14,672 emails, 98% reference the victim's research area and 54% mention at least one specific publication, confirming that *SpearMail* emails are structurally grounded in victim profiles rather than relying on generic phishing templates.

6 Experiments

We carry out comprehensive experiments to answer the following research questions:

- **RQ1: Closed-World Experiment:** How effective can *PiMRef* detect phishing emails on different phishing benchmarks, compared to the state-of-the-art approaches?
- **RQ2: Open-World Experiment:** How does *PiMRef* perform on real-world emails in comparison to both academic baselines and industry-standard anti-spam filters?
- **RQ3: Adversarial Robustness Evaluation:** How resilient is *PiMRef*, a neural model based solution, against various adversarial attacks?
- **RQ4: Ablation Study:** How can each feature of *PiMRef* contribute to the performance of *PiMRef*?
- **RQ5: User Study:** Can *PiMRef* explanations help users better identify phishing emails?

Model Hyperparameter Setup We train the NER model using the bert-large-uncased backbone released by Google [34]. The model is fine-tuned for 7 epochs with a learning rate of 2e-5 and a batch size of 8. For the identity matching model, we directly use the same pre-training pipeline in [111]. The CharacterBERT model has been pre-trained on English Wikipedia and OpenWebText [79] and has been specifically designed to be resistant to typo-squatting attacks. We set the identity-matching threshold to 0.83 (Table 8), selected on the in-distribution training set (Section 6.1.2) as the best F_β trade-off, where we use $\beta = 0.5$ to favor precision and compute $F_\beta = \frac{(1+\beta^2)\,\text{Precision}\cdot\text{Recall}}{\beta^2\,\text{Precision}+\text{Recall}}$. All experiments were conducted on an Ubuntu 20.04 system using four NVIDIA RTX 4090 GPUs.

6.1 RQ1: Closed-World Experiment

6.1.1 ***Baselines.*** We select baselines by three criteria: coverage of the major design paradigms for phishing detection, representativeness measured by adoption in follow-up work and citations, and availability of a runnable implementation for faithful reproduction. This yields three groups, described in detail in Appendix F.

Feature-engineering detectors aggregate handcrafted content features, for which we use HelpHed [23] and D-Fence [71] under the configurations recommended in the original papers, including both HelpHed ensembling variants. The **LLM-based detector** ChatSpamDetector [65] performs prompt-driven reasoning over cues such as impersonation and suspicious actions or links, and we use GPT-4 following the original setup. The **production anti-spam filters** RSpamd [102] and SpamAssassin [104] are widely deployed rule- and reputation-based systems that approximate real-world deployments, run under their default configurations.

6.1.2 ***Setup.*** **Training and In-Distribution Test.** Following D-Fence and HelpHed [23, 71], we train all learning-based methods on the same corpora, 4,558 phishing emails from the Nazario 2005 corpus [84] and 10,000 benign emails subsampled from Enron [32]. For NER training we manually annotate a random subset of 2,086 emails from these corpora with claimed identities and CTAs, since labeling all is prohibitive, and split it into 1,701 for training and 385 for testing. Two annotators are involved, the first performing the initial labeling and the second independently reviewing and correcting it in Label Studio [1], with 83% agreement and disagreements resolved by the reviewer. All methods, including *PiMRef*, are trained on this conventional corpus and never on *SpearMail*, ensuring a fair comparison.

Conventional Benchmarks (Out-of-Distribution Test). To evaluate generalization to a different distribution, we use a separate test set from newer and independent sources, 2,584 phishing emails from the Nazario corpus (2015–2023) and 2,949 benign emails from CSDMC-Ham [85], yielding 2,053 phishing and 2,103 benign emails after deduplication. We also include results on the benign splits of TREC-05/06/07 [31], CEAS-08 [27], Enron [32], SpamAssassin [15], and Ling-Spam [11].

Real-World and LLM-Generated Phishing. We further evaluate on *PhishPot* [4], an open repository of real-world phishing emails collected between Sep 2023 and Nov 2024, from which 4,300 collected emails yield 794 unique phishing emails after deduplication, and on four LLM-generated benchmarks, *Prompted Contextual Vector* [83], *E-PhishGen* [91], *SpearBot* [93], and *SpearMail* (Section 5). These four contain profile-grounded spear phishing, whereas Nazario and PhishPot represent generic phishing with identity claims, so together they cover both classes in our scope (Section 3).

Metrics. We report the False Positive Rate (FPR) on the CSDMC dataset, recall on the phishing datasets, and the median runtime as a measure of operational cost.

6.1.3 ***Results of Phishing Detection.*** Table 1 and Table 2 show that *PiMRef* achieves a consistent advantage over the baselines in balancing false positives, false negatives, and runtime overhead. The reported median runtime of 0.05 seconds covers per-email identity recognition and knowledge base matching. It excludes knowledge base expansion (Section 4.4), which is triggered only on a cold miss, i.e., the first encounter of an unresolved identity, runs offline and asynchronously, and is a one-time cost per identity of about 7 minutes amortized across all subsequent emails claiming that identity. Feature-engineering solutions struggle to balance false positives and negatives, largely due to distribution shift. ChatSpamDetector performs well on PhishPot but is over-aggressive on benign emails,

Table 1: Performance on closed-world and real-world phishing benchmarks. Each cell shows the raw counts as well as FPR or Recall. Lower FPR is better; higher recall is better. We highlight the top-3 solutions in green and the worst in red.

Solution	FP Counts (FPR on CSDMC)	TP Counts (Recall on Nazario [84])	TP Counts (Recall on PhishPot [4])	Runtime (s)
D-Fence [71]	2043 (97.15%)	1697 (82.66%)	788 (99.24%)	0.08
HelpHed (Soft Voting) [23]	0 (0.00%)	1100 (53.58%)	213 (26.83%)	**0.03**
HelpHed (Stacking) [23]	308 (14.65%)	1878 (91.48%)	685 (86.27%)	**0.03**
ChatSpamDetector (GPT-4) [65]	208 (9.89%)	752 (36.63%)	794 (100.00%)	5.66
SpamAssassin [104]	17 (0.81%)	51 (2.48%)	64 (8.06%)	1.32
RSpamd [102]	83 (3.95%)	1038 (50.56%)	371 (46.73%)	2.03
PiMRef (Ours)	25 (1.19%)	1872 (91.18%)	739 (93.07%)	0.04
Total	**2103** (100.00%)	**2053** (100.00%)	**794** (100.00%)	–

Table 2: Recall on LLM-generated phishing benchmarks. Each cell shows the raw counts as well as Recall. Higher recall is better. We highlight the top-2 solutions in green and the worst in red. *Note: E-PhishGen has 8,308 emails, but only 2,529 emails are imitating real organizations.

Solution	TP Counts (Recall on *SpearMail*)	TP Counts (Recall on ContextualVector [83])	TP Counts (Recall on E-PhishGen [91])	TP Counts (Recall on SpearBot [93])
D-Fence [71]	0 (0.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)
HelpHed (Soft Voting) [23]	0 (0.00%)	52 (15.57%)	1155 (45.67%)	1 (0.10%)
HelpHed (Stacking) [23]	14132 (96.32%)	254 (76.05%)	1450 (57.33%)	416 (41.60%)
ChatSpamDetector (GPT-4) [65]	11150 (76.00%)	334 (100.00%)	2259 (89.32%)	142 (14.20%)
SpamAssassin [104]	22 (0.15%)	0 (0.00%)	8 (0.32%)	0 (0.00%)
RSpamd [102]	42 (0.29%)	0 (0.00%)	0 (0.00%)	1 (0.10%)
PiMRef (Ours)	12678 (86.41%)	301 (90.12%)	2113 (83.55%)	841 (84.10%)
Total	– / **14672**	– / **334**	– / **2529***	– / **1000**

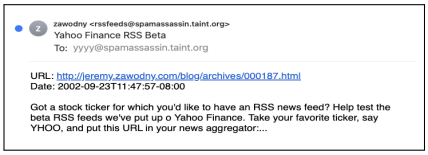
and it retains non-negligible false negatives on LLM-generated phishing because it makes ungrounded decisions on whether an email is *impersonated*. RSpamd and SpamAssassin rarely produce false alerts but miss the majority of phishing emails, even on conventional datasets.

Why do D-Fence and HelpHed overfit? Both D-Fence and HelpHed can overfit because their decisions are dominated by a few header- or format-level features that are not stable across deployments. We visualize the top-3 important features of them in Figure 11. We observe that D-Fence heavily relies on the count of "Received" headers. D-Fence operates under the assumption that benign emails typically traverse multiple layers of email servers. However, legitimate emails can vary widely in their routing paths, and malicious actors could easily manipulate the "Received" headers to mimic benign patterns. HelpHed can be biased toward Content-Transfer-Encoding. While encoding type can offer some insights into the nature of the email content, it is not a definitive indicator of phishing.

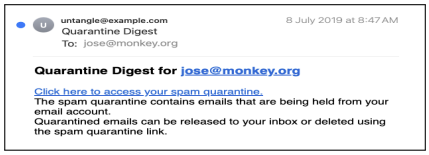
Why does ChatSpamDetector have false positives and negatives? ChatSpamDetector makes mistakes when it is forced to make *ungrounded* decisions on whether an email is impersonated. Specifically, it detects phishing emails using a *single* information source. For instance, when the phishing email is imitating the "Annual Conference on Human-Robot Interaction (HRI 2025)", if a sender address is registered as *ieeehri@humanrobot.com*, GPT might mistakenly recognize it as consistent with the identity. In contrast, the official address should be from *humanrobotinteraction.org*, which can be well captured by the reference-based design of *PiMRef*. More examples of false negatives and false positives can be found on our anonymous website [13].

Why do RSpamd and SpamAssassin miss phishing emails? We examine the rules that most often trigger each solution. SpamAssassin fires mainly on "MIME HTML ONLY", "TO MALFORMED", and "DKIM SIGNED", while RSpamd fires on "DATE IN PAST", "RDNS NONE", and "MANY INVISIBLE PARTS". These rules key on transport and formatting artifacts rather than the email's claims, so they are interpretable but trivially circumvented by an attacker who sends well-formed, DKIM-signed mail.

***PiMRef* False Negatives. (1) Missing or ambiguous sender identities.** As illustrated in Figure 6b, some phishing emails avoid claiming a clear organizational identity. In such cases, *PiMRef* does not recognize any claimed identity, and therefore cannot perform the downstream inconsistency check. This reflects an ASR (Attack-Success-Rate) and BR (Bypass-Rate) trade-off: vague identities may evade detection yet reduce user trust [59]. As a mitigation, when *PiMRef* fails to extract a confident sender identity, we plan to extend identity inference from richer contextual cues in future work. Moreover, instead of treating such cases as benign, we can explicitly flag "missing or highly ambiguous identity" as a soft warning signal to downstream filters or user interfaces. **(2) Non-salient call-to-action phrases.** As shown in Figure 6c, some phishing emails use CTAs such as "Surprise in your inbox" that stand out visually but lack an explicit imperative verb, and thus are not recognized as clear instructions. To mitigate this, we can incorporate visual saliency detection [25, 55] into our pipeline, combined with



(a) FP example on CSDMC dataset



(b) FN example on Nazario dataset

(c) FN example on PhishPot dataset

Figure 6: Failure examples of *PiMRef*

cognitive and advertising theories on how visually prominent elements drive behavior [29, 92]. Such extensions would help *PiMRef* better capture indirect yet visually highlighted CTAs.

***PiMRef* False Positives** Upon investigation, we find that part of *PiMRef*'s false alarms on the CSDMC benign dataset stem from label noise: the dataset itself contains some suspicious emails. Figure 6a is an example *PiMRef* flags as Yahoo Finance phishing, promoting a beta Yahoo Finance RSS feed with a test URL but sent from `rssfeeds@spamassassin.taint.org`, which does not belong to the Yahoo domain. We provide a systematic analysis of *PiMRef*'s false positives on a large public benign corpus in Section 6.1.4.

6.1.4 ***Understanding False Positives on Benign Emails.*** To understand when *PiMRef* raises false alarms, we run it on the benign portion of the seven-dataset email collection [58], which aggregates 108,689 benign emails from TREC-05/06/07 [31], CEAS-08 [27], Enron [32], SpamAssassin [15], and Ling-Spam [11]. Of these, 90,155 carry a sender address and form our evaluation set, on which *PiMRef* flags 250 emails, a false positive rate of 0.28%. These emails exhibit *genuine identity-domain inconsistencies*, typically because the email is delivered through a third-party Email Service Platform, sent from a sibling or parent domain of the claimed organization, or sent on behalf of an organization from a personal address (Figure 7). This reveals a limitation of *PiMRef*, since our knowledge base is *domain-level* whereas a specific sender address is sometimes legitimately authorized to represent an identity, e.g., `wsj.service@dowjones.com` sends newsletters on behalf of the Wall Street Journal. A natural remedy is an address-level whitelist layered on the domain-level knowledge base, following the user-managed allowlists in production email systems [42, 81], binding an exact sender address e to a claimed identity id and exempting only that pair $\langle e, id \rangle$. Such a whitelist is cheap in practice, as the false positives are concentrated on a few high-volume senders. Whitelisting the **Top-10** most frequent senders already eliminates 216 of the 250 false positives (86.4%) while introducing no false negatives, since an exemption is scoped to the exact pair and the same address claiming a *different* identity is still flagged. We leave a full design to future work, where the open questions concern how entries are bootstrapped and revoked when a delegated address changes ownership or is compromised, and how the whitelist interacts with authentication signals such as SPF and DKIM alignment so that an exemption is never granted to a spoofable address.

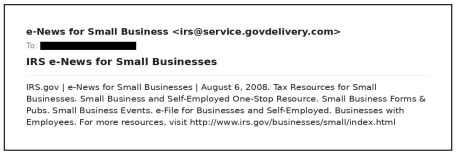
6.1.5 ***Does PiMRef Overfit the SpearMail Template?*** Since *SpearMail* is produced by a fixed LLM generation pipeline, its emails may share a common template, and a detector could appear effective by recognizing the template rather than the underlying phishing signal. We examine this from both directions. On the phishing side, we sample 500 *SpearMail* emails and instruct GPT-4o to perturb their discourse structure, reordering the functional segments (greeting, pretext, call-to-action, sign-off) and relocating the embedded link, while preserving the claimed identity, the recipient-specific details, the URL string, and the phishing intent. On the benign side, we sample 500 CSDMC-Ham emails and rewrite them into *SpearMail*'s presentation style while strictly preserving their communicative intent. Both prompts are in Appendix C. A template-dependent detector should produce false negatives and false positives respectively. Table 3 shows that *PiMRef* retains a recall of 81.0% under discourse perturbation and reports 0.0% false positives on the rewritten benign emails, indicating that its decisions rest on the counterfactual identity-domain inconsistency rather than on the generation template, whereas the baselines either fail to detect the perturbed phishing emails or become unstable on the template-styled benign ones.

Table 3: Template-invariance evaluation. Recall on 500 discourse-perturbed SpearMail emails and FPR on 500 benign CSDMC-Ham emails rewritten into SpearMail's surface style.

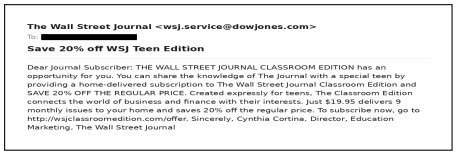
Solution	Recall (500 Perturbed *SpearMail*)	FPR (500 Rewritten CSDMC-Ham)
D-Fence	0.00%	0.00%
HelpHed (Soft Voting)	0.00%	1.40%
HelpHed (Stacking)	1.60%	28.20%
ChatSpamDetector (GPT-4)	47.60%	0.60%
SpamAssassin	0.00%	0.00%
RSpamd	0.00%	2.20%
PiMRef (Ours)	**81.00%**	**0.00%**

6.1.6 ***Results of Knowledge Evolution.*** To evaluate the agent's output quality, we randomly sample 100 identities and manually collect their official contact emails from organization contact pages, LinkedIn, and verified social media as ground truth. The agent achieves 97% precision and 93% recall on this subset, confirming that the construction strategy yields suitable performance. False positives mainly arise from anti-crawling obfuscation, where official pages hide contact emails in non-standard forms or image-rendered addresses that can be mistakenly converted into valid-looking domains. False negatives mainly arise from the limited navigation capability of the LLM-based search agent, as some official addresses sit several clicks from the homepage, under dynamically rendered pages, inside PDFs, or across department-level subpaths. Both can be mitigated in future work with more engineering effort.

(a) A Netflix investor-relations alert delivered by the third-party service `shareholder.com`.



(b) A genuine IRS newsletter distributed through the third-party service `govdelivery.com`, used by many government agencies, rather than an `irs.gov` domain.



(c) A genuine Wall Street Journal newsletter sent from its parent company's domain `dowjones.com` rather than an official WSJ domain.

Figure 7: False-positive cases from *PiMRef* on the public benign corpus: they exhibit genuine identity-domain inconsistencies, where a legitimate brand delivers mail through a third-party service or parent-company domain rather than its own official domain. Recipient fields are redacted for privacy.

6.2 RQ2: Open-World Experiment

In this study, we investigate the performance of *PiMRef* and baselines for phishing emails occurring in the wild. We present the dataset summary we used in this experiment in Table 4.

Table 4: Open-world datasets summary.

Datasets	Total	# Wild Phishing Emails	# Simulated Phishing Emails
Researcher Email Dataset	9,369	6 (reported)	87
University Spam Feeds	2,850	1,868	0
Honeypot Phishing	70	70	0

University Spam Feed dataset. The University provides access to a subscription-based feed available to researchers, which logs emails received within the university organization that are flagged by its anti-spam filter (RSpamd [102]). This dataset spans 2024-11 to 2026-04 and comprises 2,850 spam emails, of which we label 1,868 as phishing. The labeling follows our threat-model definition (Section 3), i.e., an email is phishing if it both impersonates an identity and carries a call-to-action, and is applied by the authors via manual inspection, independent of *PiMRef*'s output. This openly accessible dataset is included as part of our evaluation.

Honeypot Phishing dataset. In addition, we registered a honeypot email account, which was actively distributed by submitting it to phishing websites. Each day, the email address was submitted to 100 newly identified phishing sites listed on OpenPhish [89] using an automated Selenium-based form filler [78]. This honeypotting activity was conducted over the course of one year, attracting 70 phishing emails. This activity involved only the passive collection of unsolicited phishing emails. **No personally identifiable information was collected**, and all procedures adhered to applicable ethical and institutional guidelines.

Real-World Researcher Email Evaluation. To assess deployment feasibility, we conducted a small-scale on-device field evaluation with three consented volunteers at the same institution (IRB No. 20250859I). *PiMRef* ran locally on participants' machines; no raw email content or identifiers were collected or transmitted. Participants provided feedback only for emails flagged as suspicious by *PiMRef* or a baseline. To supplement the rarity of naturally occurring phishing emails, each participant additionally received 30 synthetic spear-phishing emails generated via the same procedure as SpearMail. 3 of the 90 were dropped for referencing non-existent organizations, yielding 87 in total. None of the three participants was involved in the design of *PiMRef*. Participants consented in advance to receive simulated phishing emails during the study period, but were not told their timing, sender, or content. None reported acting on any synthetic email (e.g., clicking its link or replying). For internal-identity matching (Section 4.3), the recipient domain is taken directly as the volunteer's own mailbox domain, without additional inferences.

6.2.1 ***Results.*** We evaluate the performance of each solution using precision and recall. As shown in Table 5, we observe that D-Fence, ChatSpamDetector, and HelpHed tend to report false alarms. On the researcher's email dataset, *PiMRef* is less likely to report false alarms. Our approach achieves recalls of 80.5% and 87.1% on the spam feed and honeypot datasets, respectively. More wild phishing examples are shown in Appendix A.

6.3 RQ3: Adversarial Robustness

6.3.1 ***Attack Setup.*** Once deployed, *PiMRef*'s model weights are confidential, thus, we evaluate the robustness of *PiMRef* under practical, black-box text perturbations widely used in prior work [24, 37, 38, 43, 51, 57]. We consider three attack surfaces: (1) **email processing** via text salting/invisible-character injection [101], (2) **identity and CTA extraction (the NER model)** via token and character perturbations and paraphrasing (BAE [38], DeepWordBug [37], GPT paraphrasing, ConcatSent [43], TextFooler [57]) and (3) **identity matching** via typo-level perturbations on brand names (DeepWordBug [37]). Surfaces (1–2) are evaluated on the 385 test emails in Section 6.1.2, and surface (3) on the 4,536 knowledge-base brand variants (Section 4.4).

Attack Surface 1: Email Processing. We apply the text salting attack [101], where emails are manipulated by inserting invisible characters at random positions. After injection, we assess whether the rendered text contains these "salted" characters.

Attack Surface 2: Identity NER. For attacks on the named entity recognition (NER) model, we differentiate between identity and action classes, as action phrases tend to be longer and semantically richer. For the *identity class*, we apply:

- **BAE** [38]: Masks the token immediately preceding the target entity and uses a pre-trained BERT model to generate top-k candidate insertions.

Table 5: Experimental results on open-world datasets.

Solution	Researcher Email Precision	Researcher Email Recall (Simulated)	University Spam Feed Dataset Recall (Wild)	Honeypot Phishing Dataset Recall (Wild)	Runtime (s)
D-Fence [71]	1.39%	**100.00%**	79.49%	**100.00%**	0.11
HelpHed (Soft Voting) [23]	2.57%	17.24%	44.22%	32.86%	0.08
HelpHed (Stacking) [23]	0.42%	20.69%	62.52%	90.00%	0.07
ChatSpamDetector (GPT-4) [65]	6.85%	93.10%	**98.07%**	95.71%	3.75
University Anti-Spam Filter	–	3.45%	–	–	–
PiMRef	**90.29%**	**100.00%**	80.51%	87.14%	0.05

Table 6: Adversarial attacks on the three components of *PiMRef*. Panel (a) reports attack success rate, where 0% means the salting characters still appear in the rendered text seen by *PiMRef*. Panels (b) and (c) report recognition and matching rates as percentages.

(a) Email processing. Attack	Attack Success Rate
Zero-width characters [101]	0%
Hidden CSS elements [101]	0%
White text on white background [101]	0%

(b) NER model. Attack (target field)	Clean	After Attack
BAE [38] (identity)	89%	87% (↓2.25%)
DeepWordBug [37]–Delete (identity)	91%	92% (↑1.10%)
DeepWordBug [37]–Replace (identity)	94%	94% (–)
DeepWordBug [37]–Switch (identity)	90%	91% (↑1.11%)
DeepWordBug [37]–Repeat (identity)	92%	92% (–)
GPT Paraphrase (action)	90%	88% (↓2.22%)
ConcatSent [43] (action)	90%	89% (↓1.11%)
TextFooler [57] (action)	89%	89% (–)

(c) Identity matching model. Attack	BERT	CharBERT
No Attack	100%	100%
DeepWordBug [37]–Delete	22% (↓78%)	73% (↓27%)
DeepWordBug [37]–Replace	22% (↓78%)	75% (↓25%)
DeepWordBug [37]–Switch	15% (↓85%)	85% (↓15%)
DeepWordBug [37]–Repeat	20% (↓80%)	92% (↓8%)

- **DeepWordBug** [37]: Introduces character-level perturbations by replacing, deleting, switching, or repeating characters within an entity.

For the *action class*, we apply:

- **GPT Paraphrasing**: Rewrites the action phrase using a paraphrasing model.
- **ConcatSent** [43]: Merges phrases with preceding sentences to disrupt original sentence boundaries.
- **TextFooler** [57]: Replaces key verbs with contextually appropriate synonyms.

We evaluate the entity recognition rate to determine whether the model can still accurately identify and classify entities under adversarial perturbations.

Attack Surface 3: Identity matching. To assess robustness against attacks on the identity matching model, we evaluate the matching accuracy between original and perturbed brand names using the *DeepWordBug* [37] attack.

6.3.2 ***Results.*** Table 6 summarizes the three attack surfaces. Email rendering is unaffected by text salting (panel a). For the NER model (panel b), the clean recognition rate varies across attack methods because it is computed only over cases where the attack is feasible, e.g., BAE is not performed when it cannot find a meaningful token for insertion. The model is generally robust to token insertion, typo insertion, and sentence paraphrasing. For identity matching between original and typo-inserted brand names (panel c), CharacterBERT substantially outperforms BERT.

Table 7: Ablation study on model design. We report false positive rate (FPR), recall, and median runtime per email.

Modules			Performance		
CTA Detector	Internal	Decoder-based ID Recog	FPR	Recall	Runtime
	✓		13.79%	92.26%	0.04s (-)
✓			0.95%	64.90%	0.04s (-)
✓	✓	LLaMA2-7B	0.33%	67.12%	1.92s (↑)
✓	✓	Mistral-7B	1.00%	58.55%	2.58s (↑)
✓	✓		1.19%	91.18%	0.04s (-)

6.4 RQ4: Ablation Study

Setup. We evaluate three design alternatives, reporting recall on the Nazario phishing test set and false positive rate (FPR) on the CSDMC-Ham benign set (Section 6.1). **Op1** drops the call-to-action requirement, so an email is reported as phishing whenever sender identity inconsistency is detected. **Op2** disables *PiMRef*'s capability on internal identities, restricting it to brand impersonation. **Op3** replaces the NER-based identity recognition model (Section 4.2) with a decoder-based text generation model [54] that is instruction-tuned [110] to produce the claimed identity and call-to-action phrases directly from the plain-text email body, under the instruction "First, recognize the sender's claimed identity. Second, identify the call-to-action phrases". We use LLaMA2 [10] and Mistral [56] at 7B parameters, both fine-tuned with LoRA [52] under causal language modeling loss until convergence.

Results. Table 7 reports the results, with the full configuration in the last row. Call-to-action recognition is essential for precision, since removing it improves recall by 1% but inflates FPR by 12% on the benign dataset. Internal impersonation is essential for recall, since disabling it drops recall by roughly 30%, indicating that a substantial fraction of phishing attempts in our evaluation impersonate entities within the recipient's own organization and would be missed by checks limited to external brands. The encoder-based NER model also outperforms the decoder-based alternatives on both accuracy and latency, which reduce recall by 20% and raise median per-sample latency to 1.92 s and 2.58 s respectively, making them impractical.

6.5 RQ5: User Study

We conducted a small-scale user study, approved by our Institutional Review Board, on whether *PiMRef*'s explanations improve users' ability to identify phishing emails.

Participants. We recruited 20 volunteers via an institutional mailing list (authors excluded), from senior undergraduates to postdoctoral researchers (age 22 to 29, 25% female), all with CS-related backgrounds and no formal security training beyond a 5-minute pre-study tutorial based on Google Jigsaw's phishing quiz [5]. Prior phishing exposure (90% control, 100% treatment) and self-rated identification confidence (4.2/5 control, 3.9/5 treatment) were comparable across groups. The recruitment form, consent form, and full questionnaire are in Appendix E.

Email Materials. The 10 test emails were real emails contributed by volunteers from their own inboxes, fully de-identified and presented as simulated emails within the survey platform. The 5 phishing emails impersonate conference or journal invitations, and the 5 benign emails are genuine account-security notifications, chosen as hard negative controls since their urgent tone resembles phishing. The balanced design sets chance performance at 50%.

Questionnaire and Study Design. Part 1 collects two covariates, prior phishing exposure and self-rated confidence on a behaviorally anchored five-point scale, used to verify that the randomized groups are comparable. Demographics are collected at recruitment as coarse, non-identifying attributes only. Part 2 is the primary task, a judgment plus a five-point confidence rating per email that separates correctness from certainty. Part 3 collects overall difficulty, perceived mental demand, and the primary judgment cues, and the treatment group additionally rates the helpfulness of *PiMRef*'s explanations on a five-point Likert scale. Completion time is recorded automatically by the platform rather than self-reported. We follow a randomized A/B design where Group A (control, n=10) saw raw emails and Group B (treatment, n=10) saw the same emails augmented with *PiMRef*'s explanations highlighting claimed identities and call-to-action statements. Judgments are three-way (phishing, benign, or unsure), with unsure scored as incorrect. We report **Accuracy**, the overall correct classification rate, and **Completion Time**, the average time per judgment.

Results. Table 9 (in Appendix) shows that Group B's accuracy is statistically significantly higher than Group A's ($t(16.32) = -5.37$, $p < .001$), while Group A is slower by about 13.5 seconds per question. Group B rated our explanations at 4.7 out of 5 on average, indicating that they are perceived as very helpful for decision making.

7 Discussion

Deployment Scenarios. *PiMRef* supports both server- and client-side deployment. On the server side, it integrates into an organization's centralized Mail Transfer Agent as a phishing scanner, enabling comprehensive filtering across the organization, and enterprises can supply a customized fine-grained knowledge base to further improve accuracy. On the client side, it runs as an Outlook plugin that complements existing detectors with personalized in-interface alerts highlighting the detected identity-domain inconsistency, improving users' phishing awareness. A video demo is available at [12].

Other Disprovable Claims. Counterfactual identity is a foundational step toward the broader goal of disprovable claim detection, and we focus on identity for its prevalence in phishing emails. Other disprovable claims exist, such as delivery notifications for nonexistent packages, billing requests for services the user never subscribed to, and role-based claims asserting privileged access or executive identity. Verifying these requires cross-referencing personal calendars, billing history, or organizational context, so extending this line of work hinges on obtaining such user-sensitive data under appropriate permissions while preserving privacy.

Limitations. *PiMRef* can be evaded when the sender identity is highly ambiguous or absent, e.g., abbreviating *Kaiser Permanente* as *KP*, but such evasion sharply reduces the email's plausibility, creating a fundamental tradeoff for the attacker. Its knowledge base is domain-level and shared across users, so it cannot capture address-level exceptions where a specific sender address is legitimately authorized to represent an identity. Such exceptions are inherently per-user and cannot be enumerated globally, and we envision a user-side whitelisting mechanism that incrementally refines the knowledge base to the address level. The open-world evaluation is further bounded by privacy constraints. The field study covers only three volunteers under our IRB scope, and the university spam feed contains only emails already flagged by RSpamd, so the measured recall reflects that filter's positives rather than the full mail stream. We plan a larger-scale deployment study in future work. More broadly, as AIGC techniques mature, detecting misinformation from a single information source becomes increasingly difficult. We therefore advocate an email writing practice that enables cross-validation across multiple sources, where organizational users send official emails from enterprise rather than personal accounts and explicitly claim their identity, building verifiable trust between senders and recipients.

8 Conclusion

PiMRef is the first reference-based solution for detecting spear-phishing emails. By analyzing disprovable claims and call-to-action phrases within the email body, *PiMRef* sets a new state-of-the-art in accuracy, explainability, and efficiency. It leverages Named Entity Recognition (NER) and word embedding models to extract the sender identities and precisely cross-validate them against a comprehensive brand-email knowledge base. Our evaluation demonstrates that *PiMRef* consistently outperforms both academic baselines and industry-standard anti-spam filters in both closed-world and open-world scenarios, highlighting its practicality and effectiveness for real-world deployment.

Ethical Considerations

We analyze three components, the construction of SpearMail, the open-world evaluation, and the user study, under the stakeholder-based framework prescribed by the ethics guidelines. The stakeholders are the researchers whose public ORCID profiles were ingested, the open-world recipients, the study participants, and society at large.

Beneficence. Our contribution is a defense against attacks already observed in the wild. We judged the benefit of a reproducible evaluation of such defenses to outweigh the marginal uplift SpearMail

offers an adversary, since its generation recipe requires no capability beyond public LLM APIs and we restrict its dissemination. No delivered email contained a working credential-harvesting payload, malware, or a live link to adversary-controlled infrastructure, and all embedded URLs are fake.

Respect for persons. SpearMail is synthesized from biographies, education histories, and publication lists that researchers released publicly through ORCID, whose terms permit third-party reuse for research. Since they did not anticipate this use, instances are stored under internal identifiers and no generated email is attributed to a real individual. Both the open-world evaluation and the user study received full IRB approval (Protocol No. 20250859I). The three recipients in Section 6.2 consented in advance to a red-team exercise but were blind to its timing, sender, and content; each was debriefed within 24 hours and could withdraw their data. Participants gave informed consent, were told that some emails might be phishing without being told which, and were debriefed after the session; all emails shown to them were sanitized of personally identifiable information, institutional names traceable to real individuals, and content that could cause distress.

Dual use. Collection complies with the ORCID Terms of Use and the public-data provisions of the EU GDPR. We performed no credential harvesting, unauthorized access, or circumvention of provider security controls, and generation used only benign prompts within providers' terms, requiring no jailbreak. We will release the PiMRef detector and the knowledge base construction code, whose reproduction is necessary to verify our claims, but **not** the raw SpearMail corpus; the generation prompts will be released only in redacted form, sufficient to reproduce the methodology but not to run a pipeline at scale.

Open Science

All model checkpoints and inference scripts are available at https://github.com/code-philia/PhishEmail.

Acknowledgements

This research is supported in part by the National Natural Science Foundation of China (62572300), the Minister of Education, Singapore (MOE-T2EP20124-0017, MOET32020-0004), the National Research Foundation, Singapore and the Cyber Security Agency under its National Cybersecurity R&D Programme (NCRP25-P04-TAICeN), DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-GC-2023-008-1B), and Cyber Security Agency of Singapore under its National Cybersecurity R&D Programme and CyberSG R&D Cyber Research Programme Office, and partially by HUAWEI's Al Hundred Schools Program using the Ascend AI technology stack. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore, Cyber Security Agency of Singapore as well as CyberSG R&D Programme Office, Singapore.

References

[1] Label studio. https://labelstud.io/guide/.
[2] Lunar bank. https://www.lunar.app/.
[3] Paddlepaddle paddleocr. https://github.com/PaddlePaddle/PaddleOCR/tree/main.
[4] Phishing pot github repository. https://github.com/rf-peixoto/phishing_pot.
[5] Phishing quiz: Jigsaw - google. https://phishingquiz.withgoogle.com/.
[6] Isredza Rahmi A. Hamid and Jemal Abawajy. Hybrid feature selection for phishing email detection. In *Proc. ICA3PP*, 2011.
[7] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. Whitenet: Phishing website detection by visual whitelists. *arXiv:1909.00300*, 2019.
[8] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. Visualphishnet: Zero-day phishing website detection by visual similarity. In *Proc. ACM CCS*, 2020.
[9] Sadia Afroz and Rachel Greenstadt. Phishzoo: Detecting phishing websites by looking at them. In *Proc. IEEE ICSC*, 2011.
[10] Meta AI. Llama 2: Open-source language model, 2023.
[11] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras, and Constantine D. Spyropoulos. An evaluation of naive bayesian anti-spam filtering. In *Proc. ECML*, 2000.
[12] Anonymous. Anonymous website for pimref: Homepage. https://sites.google.com/view/pimref/home, 2025.
[13] Anonymous. Anonymous website for pimref: Supplementary examples. https://sites.google.com/view/pimref/supplementary-examples, 2025.
[14] Anonymous. Anonymous website for pimref: Wild study. https://sites.google.com/view/pimref/our-failure-cases-in-the-wild, 2025.
[15] Apache SpamAssassin Project. SpamAssassin public mail corpus. https://spamassassin.apache.org/old/publiccorpus/, 2005. Accessed: 2026-08-08.
[16] Giovanni Apruzzese, Mauro Conti, and Ying Yuan. Spacephish: The evasion-space of adversarial attacks against phishing website detectors using machine learning. In *Proc. ACSAC*, 2022.
[17] Giovanni Apruzzese and VS Subrahmanian. Mitigating adversarial gray-box attacks against phishing detectors. *IEEE TDSC*, 20(5), 2022.
[18] Md Ishtiaq Ashiq, Weitong Li, Tobias Fiebig, and Taejoong Chung. You've got report: Measurement and security implications of {DMARC} reporting. In *Proc. USENIX Security*, 2023.
[19] Mazal Bethany, Athanasios Galiopoulos, Emet Bethany, Mohammad Bahrami Karkevandi, Nicole Beebe, Nishant Vishwamitra, and Peyman Najafirad. Lateral phishing with large language models: A large organization comparative study. *IEEE Access*, 2025.
[20] Douglas Biber, Stig Johansson, Geoffrey Leech, Susan Conrad, and Edward Finegan. *Longman Grammar of Spoken and Written English*. 1999.
[21] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *JMLR*, 3(Jan), 2003.
[22] Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Junichi Tsujii. Characterbert: Reconciling elmo and bert for word-level open-vocabulary representations from characters. *arXiv:2010.10392*, 2020.
[23] Panagiotis Bountakas and Christos Xenakis. Helphed: Hybrid ensemble learning phishing email detection. *J. Netw. Comput. Appl.*, 210, 2023.
[24] Jan Brabec, Filip Šrajer, Radek Starosta, Tomáš Sixta, Marc Dupont, Miloš Lenoch, Jiří Menšík, Florian Becker, Jakub Boros, Tomáš Pop, et al. A modular and adaptive system for business email compromise detection. *arXiv:2308.10776*, 2023.
[25] Neil DB Bruce and John K Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of vision*, 9(3), 2009.
[26] Marco Casagrande, Mauro Conti, Monica Fedeli, Eleonora Losiouk, et al. Alpha phi-shing fraternity: Phishing assessment in a higher education institution. *JOURNAL OF CYBERSECURITY EDUCATION, RESEARCH & PRACTICE*, 2023.
[27] CEAS. CEAS 2008 live spam challenge laboratory corpus. Fifth Conference on Email and Anti-Spam (CEAS), 2008.
[28] Anish Chand, Nick Nikiforakis, and Phani Vadrevu. Doubly dangerous: Evading phishing reporting systems by leveraging email tracking techniques. In *Proc. USENIX Security*, 2025.
[29] Robert B. Cialdini. Influence: The psychology of persuasion. 2007.
[30] Asaf Cidon, Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin. High precision detection of business email compromise. In *Proc. USENIX Security*, 2019.
[31] Gordon V. Cormack and Thomas R. Lynam. TREC 2005 spam track overview. In *Proc. TREC*, 2005.
[32] Enron Corp and William W. Cohen. Enron email dataset. Software, E-Resource, 2015.
[33] Dave Crocker, Tony Hansen, and Murray Kucherawy. DomainKeys Identified Mail (DKIM) Signatures. RFC 6376, Internet Engineering Task Force, 2011.
[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. NAACL-HLT*, 2019.
[35] Sevtap Duman, Kubra Kalkan-Cakmakci, Manuel Egele, William Robertson, and Engin Kirda. Emailprofiler: Spearphishing filtering with header and stylometric features of emails. In *Proc. IEEE COMPSAC*, 2016.

[36] Keelan Evans, Alsharif Abuadbba, Tingmin Wu, Kristen Moore, Mohiuddin Ahmed, Ganna Pogrebna, Surya Nepal, and Mike Johnstone. Raider: Reinforcement-aided spear phishing detector. In *Proc. NSS*, 2022.

[37] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *Proc. IEEE SPW*, 2018.

[38] Siddhant Garg and Goutham Ramakrishnan. Bae: Bert-based adversarial examples for text classification. *arXiv:2004.01970*, 2020.

[39] Hugo Gascon, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck. Reading between the lines: content-agnostic detection of spear-phishing emails. In *Proc. RAID*, 2018.

[40] Alexandru Ghiuruțan and Ciprian Pavel Oprișa. Hybrid spear-phishing email detection with llm and machine learning. In *Proc. CSNet*, 2025.

[41] Argha Ghosh and A Senthilrajan. Comparison of machine learning techniques for spam detection. *Multimedia Tools and Applications*, 82(19), 2023.

[42] Google. Add custom spam filters to Gmail. https://knowledge.workspace.google.com/admin/gmail/advanced/add-custom-spam-filters-to-gmail, 2026. Accessed: 2026-08-06.

[43] Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Qinzhuo Wu, Jiacheng Ye, et al. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing. *arXiv:2103.11441*, 2021.

[44] Lukáš Halgaš, Ioannis Agrafiotis, and Jason RC Nurse. Catching the phish: Detecting phishing attacks using recurrent neural networks (rnns). In *Proc. WISA*, 2020.

[45] NB Harikrishnan, R Vinayakumar, and KP Soman. A machine learning approach towards phishing email detection. In *Proc. IWSPA-AP*, 2018.

[46] Daojing He, Xin Lv, Xueqian Xu, Sammy Chan, and Kim-Kwang Raymond Choo. Double-layer detection of internal threat in enterprise systems based on deep learning. *IEEE TIFS*, 2025.

[47] Fred Heiding, Simon Lermen, Andrew Kao, Bruce Schneier, and Arun Vishwanath. Evaluating large language models' capability to launch fully automated spear phishing campaigns. In *Proc. ICML Workshop*, 2025.

[48] Fred Heiding, Simon Lermen, Andrew Kao, Claudio Mayrink Verdun, Bruce Schneier, and Arun Vishwanath. Evaluating large language models' ability to automate spear phishing. *Expert Systems with Applications*, 2026.

[49] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M Voelker, and David Wagner. Detecting and characterizing lateral phishing at scale. In *Proc. USENIX Security*, 2019.

[50] Grant Ho, Aashish Sharma, Mobin Javed, Vern Paxson, and David Wagner. Detecting credential spearphishing in enterprise settings. In *Proc. USENIX Security*, 2017.

[51] Tobias Holgers, David E Watson, and Steven D Gribble. Cutting through the confusion: A measurement study of homograph attacks. In *Proc. USENIX ATC*, 2006.

[52] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv:2106.09685*, 2021.

[53] Rodney Huddleston and Geoffrey K. Pullum. *The Cambridge Grammar of the English Language*. 2002.

[54] Hugging Face. Causal language modeling. https://huggingface.co/docs/transformers/tasks/language_modeling, 2026. Accessed: 2026-08-08.

[55] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE TPAMI*, 20(11), 2002.

[56] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv:2310.06825*, 2023.

[57] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proc. AAAI*, 2020.

[58] Sofien Kaabi. Seven phishing email datasets. https://huggingface.co/datasets/SofienK-s/seven-phishing-email-datasets, 2025. Aggregates TREC-05/06/07, CEAS-08, Enron, SpamAssassin, and Ling-Spam. Accessed: 2026-08-08.

[59] Kalam Khadka, Abu Barkat Ullah, Wanli Ma, Elisa Martinez Marroquin, and Yibeltal Alem. A survey on the principles of persuasion as a social engineering strategy in phishing. In *Proc. IEEE TrustCom*, 2023.

[60] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. Mitigation of spear phishing attacks: A content-based authorship identification framework. In *Proc. ICITST*, 2011.

[61] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. Enhancing phishing e-mail classifiers: A lexical url analysis approach. *International Journal for Information Security Research (IJISR)*, 2(1/2), 2012.

[62] Hanna Kim, Minkyoo Song, Seung Ho Na, Seungwon Shin, and Kimin Lee. When llms go online: The emerging threat of web-enabled llms. *arXiv:2410.14569*, 2024.

[63] Scott Kitterman. Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1. RFC 7208, Internet Engineering Task Force, 2014.

[64] KnowBe4. New knowbe4 report reveals a spike in ransomware payloads and ai-powered polymorphic phishing campaigns. https://www.knowbe4.com/press/new-knowbe4-report-reveals-a-spike-in-ransomware-payloads-and-ai-powered-polymorphic-phishing-campaigns, 2025.

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

[66] M. Kucherawy, E. Zwicky, et al. Domain-based message authentication, reporting & conformance (dmarc). https://tools.ietf.org/html/rfc7489, 2015.

[67] Daniele Lain, Tarek Jost, Sinisa Matetic, Kari Kostiainen, and Srdjan Capkun. Content, nudges and incentives: A study on the effectiveness and perception of embedded phishing training. In *Proc. ACM CCS*, 2024.

[68] Daniele Lain, Kari Kostiainen, and Srdjan Čapkun. Phishing in organizations: Findings from a large-scale and long-term study. In *Proc. IEEE S&P*, 2022.

[69] Daniele Lain, Yoshimichi Nakatsuka, Kari Kostiainen, Gene Tsudik, and Srdjan Capkun. Url inspection tasks: Helping users detect phishing links in emails. *arXiv:2502.20234*, 2025.

[70] Carlos Laorden, Xabier Ugarte-Pedrero, Igor Santos, Borja Sanz, Javier Nieves, and Pablo G Bringas. Study on the effectiveness of anomaly detection for spam filtering. *Information Sciences*, 277, 2014.

[71] Jehyun Lee, Farren Tang, Pingxiao Ye, Fahim Abbasi, Phil Hay, and Dinil Mon Divakaran. D-fence: A flexible, efficient, and comprehensive phishing email detection system. In *Proc. IEEE EuroS&P*, 2021.

[72] Younghoo Lee, Joshua Saxe, Richard Harang, and Sophos AI. Catbert: Context-aware tiny bert for detecting targeted social engineering emails. *arXiv:2010.03484*, 2021.

[73] Yuexin Li, Chengyu Huang, Shumin Deng, Mei Lin Lock, Tri Cao, Nay Oo, Bryan Hooi, and Hoon Wei Lim. Knowphish: Large language models meet multimodal knowledge graphs for enhancing reference-based phishing detection. *arXiv:2403.02253*, 2024.

[74] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE TPAMI*, 42(2), 2018.

[75] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *Proc. USENIX Security*, 2021.

[76] Ruofan Liu, Yun Lin, Xiwen Teoh, Gongshen Liu, Zhiyong Huang, and Jin Song Dong. Less defined knowledge and more true alarms: Reference-based phishing detection without a pre-defined reference list. In *Proc. USENIX Security*, 2024.

[77] Ruofan Liu, Yun Lin, Xianglin Yang, Siang Hwee Ng, Dinil Mon Divakaran, and Jin Song Dong. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In *Proc. USENIX Security*, 2022.

[78] Ruofan Liu, Yun Lin, Yifan Zhang, Penn Han Lee, and Jin Song Dong. Knowledge expansion and counterfactual interaction for {Reference-Based} phishing detection. In *Proc. USENIX Security*, 2023.

[79] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 364, 2019.

[80] Liping Ma, Bahadorrezda Ofoghi, Paul Watters, and Simon Brown. Detecting phishing emails using hybrid features. In *Proc. UIC/ATC*, 2009.

[81] Microsoft. Add recipients to the safe senders list in Outlook. https://support.microsoft.com/en-us/outlook/mail/add-recipients-to-the-safe-senders-list-in-outlook, 2026. Accessed: 2026-08-06.

[82] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1), 2007.

[83] Daniel Nahmias, Gal Engelberg, Dan Klein, and Asaf Shabtai. Prompted contextual vectors for spear-phishing detection. *arXiv:2402.08309*, 2024.

[84] J. Nazario. The online phishing corpus. http://monkey.org/~jose/wiki/doku.php, 2005.

[85] Conference on Soft Computing and Data Mining. Csdmc2010 spam corpus. Dataset, 2010.

[86] OpenAI. Chatgpt (gpt-4). https://openai.com/chatgpt, 2023.

[87] OpenAI. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/, 2024.

[88] OpenAI. Introducing OpenAI o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/, 2025. Accessed: 2026-08-06.

[89] OpenPhish. Openphish: Phishing threat intelligence. https://openphish.com/, 2024.

[90] ORCID. ORCID: Connecting Research and Researchers.

[91] Luca Pajola, Eugenio Caripoti, Stefan Banzer, Simeone Pizzi, Mauro Conti, and Giovanni Apruzzese. E-phishgen: Unlocking novel research in phishing email detection. *arXiv:2509.01791*, 2025.

[92] Rik Pieters, Edward Rosbergen, and Michel Wedel. Visual attention to repeated print advertising: A test of scanpath theory. *Journal of marketing research*, 36(4), 1999.

[93] Qinglin Qi, Yun Luo, Yijia Xu, Wenbo Guo, and Yong Fang. Spearbot: Leveraging large language models in a generative-critique framework for spear-phishing email generation. *Information Fusion*, 122, 2025.
[94] Sayak Saha Roy and Shirin Nilizadeh. Phishlang: A real-time, fully client-side phishing detection framework using mobilebert. *arXiv:2408.05667*, 2024.
[95] Sayak Saha Roy, Poojitha Thota, Krishna Vamsi Naragam, and Shirin Nilizadeh. From chatbots to phishbots?: Phishing scam generation in commercial large language models. In *Proc. IEEE S&P*, 2024.
[96] Anastasia Sergeeva, Björn Rohles, Verena Distler, and Vincent Koenig. "we need a big revolution in email advertising": Users' perception of persuasion in permission-based advertising emails. In *Proc. ACM CHI*, 2023.
[97] Kaiwen Shen, Chuhan Wang, Minglei Guo, Xiaofeng Zheng, Chaoyi Lu, Baojun Liu, Yuxuan Zhao, Shuang Hao, Haixin Duan, Qingfeng Pan, et al. Weak links in authentication chains: A large-scale analysis of email sender spoofing attacks. In *Proc. USENIX Security*, 2021.
[98] Mariya Shmalko, Alsharif Abuadbba, Raj Gaire, Tingmin Wu, Hye-Young Paik, and Surya Nepal. Profiler: Distributed model to detect phishing. In *Proc. IEEE ICDCS*, 2022.
[99] C Emilin Shyni, S Sarju, and S Swamynathan. A multi-classifier based prediction model for phishing emails detection using topic modelling, named entity recognition and image processing. *Circuits and Systems*, 7(9), 2016.
[100] Gianluca Stringhini and Olivier Thonnard. That ain't you: detecting spearphishing emails before they are sent. *arXiv:1410.6629*, 2014.
[101] CISCO Talos. Seasoning email threats with hidden text salting. https://blog.talosintelligence.com/seasoning-email-threats-with-hidden-text-salting/.
[102] Rspamd Team. Rspamd: Rapid spam filtering system. https://rspamd.com, 2024.
[103] Tanmay Thakur and Rakesh Verma. Catching classical and hijack-based phishing attacks. In *Proc. ICISS*, 2014.
[104] The Apache Software Foundation. *SpamAssassin*, 2024.
[105] Travelers Insurance. How to protect your company from business email attacks. https://www.travelers.com/resources/business-topics/cyber-security/inside-an-email-hack, 2022. Recommends multifactor authentication for email and dual-authorization policies for financial transactions as key controls against BEC.
[106] Trend Micro. Trend micro cybersecurity solutions. https://www.trendmicro.com, 2024.
[107] Rohit Valecha, Pranali Mandaokar, and H Raghav Rao. Phishing email detection using persuasion cues. *IEEE TDSC*, 19(2), 2021.
[108] Amber Van Der Heijden and Luca Allodi. Cognitive triaging of phishing attacks. In *Proc. USENIX Security*, 2019.
[109] Wikipedia. United parcel service. https://en.wikipedia.org/wiki/United_Parcel_Service.
[110] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. Instruction tuning for large language models: A survey. *arXiv:2308.10792*, 2023.
[111] Shengyao Zhuang and Guido Zuccon. Characterbert and self-teaching for improving the robustness of dense retrievers on queries with typos. In *Proc. ACM SIGIR*, 2022.

Table 8: Threshold selection for sender identity matching.

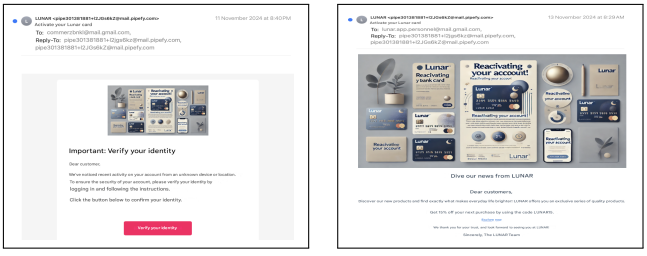
Threshold	Precision	Recall	$F_{0.5}$
0.78	0.98	0.92	0.97
0.80	0.98	0.91	0.97
0.81	0.99	0.90	0.97
0.82	0.99	0.90	0.97
0.83	**0.99**	**0.90**	**0.97**
0.84	0.99	0.89	0.97
0.85	0.99	0.89	0.97
0.86	0.99	0.89	0.97
0.87	0.99	0.87	0.96

A Wild phishing emails caught by *PiMRef*

Through analysis of honeypot emails, we observe several key patterns characterizing the current phishing email landscape. First, phishing emails commonly disguise themselves as originating from **internal roles within an organization**. We find that 30% of these emails impersonate HR personnel, sending income or tax statements, or the organization's mail delivery system, issuing automated notifications. Second, phishing emails are typically designed for **one-time engagement**, phishers rarely invest additional effort in responding to victims. To investigate this behavior, we attempted to reply to emails received by our honeypot address, posing as cautious recipients seeking further clarification from the sender. Our findings indicate that 56% of these replies received no response from the phishers, while in the remaining 44%, the mail domains used by the phishers were configured to disable any reply-to actions. Third, phishers are increasingly leveraging not only large language models (LLMs) to refine their generated emails but also **vision-language models (VLMs)**. Figure 8 presents an example where the phishing attacker created an image displaying a credit card purportedly from Lunar Bank, likely by a vision-language model (VLM). The two emails impersonate *lunar.app* [2], a digital banking app offering personal finance management, with the urgency-inducing message, "Reactivating your account". The information is embedded within the image rather than the email's text body. In the first email, the image has the urgency-inducing message. In the second attempt, the phisher enlarged the image and softened the tone of the message. Those emails indicate the growth of AIGC exploitation. More qualitative examples can be found on our anonymous website [14].

Table 9: User study result.

Idx	Group A		Group B	
	Acc	Time (s)	Acc	Time (s)
1	60%	40.5	100%	28.2
2	60%	60.2	90%	44.1
3	60%	38.9	100%	38.6
4	80%	22.5	90%	38.8
5	70%	58.0	80%	25.4
6	60%	88.8	90%	46.9
7	90%	68.3	90%	32.1
8	80%	46.3	100%	27.9
9	60%	26.9	80%	35.3
10	70%	51.9	100%	49.8
Avg.	**69%**	**50.23**	**92%**	**36.71**



(a) Initial attempt. **(b) Second attempt.**

Figure 8: A phishing email whose lure content is embedded in a VLM-generated image impersonating Lunar Bank, showing the attacker's two attempts.

B Prompts for Section 5 *SpearMail* Generation

We present the prompt pipeline for *SpearMail* generation in Figure 9. Starting from a victim profile, it produces $m \times n$ candidate spear-phishing lures. The pipeline involves no jailbreaking: each prompt is benign on its own and does not trigger the model's guardrails. The LLM benchmark is generated using GPT-4o, which was selected as the most accurate and cost-efficient option available at the time of submission.

Step 1 — INTEREST_INFERENCE $(p, m) \rightarrow m$ interests

Given the list of information about an individual: `{profile}`. Please analyze the information, and give me m `{interests}` they might have, along with where you obtained this interest from.

Step 2 — ACTIVITY_INFERENCE (interest, n) $\rightarrow n$ (org, activity)

Given the individual's `{interest}`, help me generate n unique `{(organization, activity)}` pairs that are related to this interest, and would be something that this individual would participate in. The organization must be real.

Step 3 — EMAIL_GENERATION (p, interest, activity) $\rightarrow$ email

Generate an invitation email to the following recipient based on the details below. The recipient's identity is as follows: `{recipient name}`. Write them an email about this activity: `{activity}`, which is related to the interest that he has: `{interest}`. Your identity as the email sender is as follows: `{organization}`.
The content of the email must be linked to this sender's identity, as well as the interest of the recipient. Make sure a link to `{a fake link}` is included in the body of the message, before the signature. Please provide your response in the following format:

```
{
    "subject": <email_subject>,
    "body": <html><email_body></html>
}
```

Figure 9: Prompts for invitation email generation. `{var}` denotes a runtime-substituted variable; underlined variables are intermediate outputs reused downstream.

C Prompt for Section 6.1.5 Template Invariance Evaluation

We present the prompt used for rewriting the benign emails following *SpearMail* template format, as well as the prompt to perturb the structure of *SpearMail* emails. For the benign email rewriting prompt, we deliberately keep it close to the email generation prompt used in Figure 9.

Benign Email Rewriting Prompt

Generate an invitation email to the following recipient based on the details below. The recipient's identity is as follows: `{recipient name}`. Write them an email about this activity: `{original email subject}`, with the rough context behind the email: `{original email body}`. Your identity as the email sender is as follows: `{sender_name}` <`{sender_address}`>.
The content of the email must be linked to this sender's identity. Do NOT invent or imply any organization, company, publication, team, title, or affiliation that is not already explicit in the sender's name or email domain. Please provide your response in the following JSON format:

```
{{
    "subject": "<email_subject>",
    "body": "<html><email_body></html>"
}}
```

Phishing Email Restructuring Prompt

You are rewriting an email to vary its **discourse structure** while keeping the exact same intent, target, and factual content. Apply **ALL** of the following transformations:

(1) Reorder the functional segments (greeting, self-introduction, pre-text/event, call-to-action, link, urgency, sign-off).
(2) Relocate the link to a different position than in the original (e.g. early in the body, or mid-sentence).

Preserve exactly:

- The sender's identity and any claimed organization.
- The recipient-specific details.
- The pseudo-link URL itself.

Output ONLY the rewritten email body as plain text. No subject line, no explanations, no markdown.
Original email body:
`{body}`

D Prompt for Section 4.5 Call-to-Action Augmentation

To boost the diversity of call-to-actions in our training set, we randomly paraphrase the sentence structures and persuasion strategies aspects via LLM prompting (GPT-4o). The prompt is presented in Figure 10.

E Design of the Questionnaire in Section 6.5 User Study

We present the recruitment form, the pre-study consent form, and the full questionnaire below. We also include three out of the 10 email examples selected in the user study questionnaire in Figure 12.

Recruitment and Eligibility for the User Study

Recruitment channel. Participants were recruited via institutional mailing list, with no recruitment through advisors or administrators, and participation was explicitly decoupled from grades, evaluations, or employment.
Study format. A one-time online session of about 15–20 minutes on a survey platform; 20 participants randomly assigned to a control

CTA Augmentation Prompt

Task. Rewrite **ONE** CTA phrase. **Steps:** (1) infer the pattern from `Imperative`, `Decl.-Permission`, `Fronted-Purpose`, `Conditional`, `Coordination`, `Relative-Clause`, `Topic-Fronting`; (2) pick a **different** pattern; (3) pick one Cialdini principle from `Reciprocity`, `Consistency`, `Social Proof`, `Authority`, `Liking`, `Scarcity`; (4) paraphrase: preserve intent; clear CTA; neutral tone; structurally distinct. **Constraints:** `original` = input verbatim; `paraphrased` structurally different.
Output (JSON only):

```
{
"pattern":"...",
"principle":"...",
"phrases":[
    {"original":"...",  "paraphrased":"..."},
...
]
}
```

Example input: `Click the link to start your trial.` **Example output:**

```
{
"pattern":"Conditional",
"principle":"Consistency",
"phrases":[
    {"original":
    "Click the link to start your trial.",
    "paraphrased":
    "You can start your trial via this link."}
]
}
```

Input CTA: `{phrase_list}`

Figure 10: Prompt for Call-to-Action augmentation.

group (n=10, email content only) and a treatment group (n=10, email content with the tool's alert).
Inclusion criteria. (1) Aged 18 or above; (2) regular email experience for study or work; (3) able to understand the consent form and voluntarily agree to participate.
Exclusion criteria. (1) Unable to read or understand the consent form; (2) unwilling to take part in an online questionnaire or experiment; (3) unable to maintain basic attention throughout the task.
Compensation. No monetary payment.
Consent. Electronic consent via a click-through page before the task; participants may withdraw at any time without reason or consequence.

Informed Consent for the User Study

Dear participant,

Thank you for taking part in this study on "phishing email detection and assistance tools". This questionnaire aims to understand how you judge emails while reading them, and how phishing detection tools influence your ability to identify suspicious emails.

- The questionnaire is expected to take about 15-20 minutes to complete.
- There are no right or wrong answers; please respond according to your true thoughts.
- We do not collect your name, student ID, account passwords, or other sensitive information. All data will be used for academic research only and will be kept strictly confidential.
- You may withdraw from the questionnaire at any time.

If you have read and agree to the above information, please click the button below to start.

I have read and agree to participate

Questionnaire for the User Study

Part 1: Background

(1) Have you ever encountered suspected or confirmed phishing emails in real life?
☐ I may have, but I am not sure ☐ Yes, confirmed 1–2 times
☐ Yes, confirmed multiple times

(2) How confident are you in spotting a phishing email?
☐ Not at all confident (I rely entirely on my spam filter and cannot identify them on my own.)
☐ Slightly confident (I can spot very obvious signs, but I am often unsure about others.)
☐ Somewhat confident (I can identify most common phishing emails, but sophisticated ones might still trick me.)
☐ Very confident (I am confident in my ability to identify nearly all phishing attempts.)
☐ Extremely confident (I am certain I can identify all phishing emails, including highly targeted spear-phishing attacks.)

Part 2: Email Judgment Task (instructions: "This section presents a series of simulated emails. Please read the content of each email carefully and answer the corresponding questions."; repeated for each of the 10 emails)

(1) Do you think this is a phishing email?
☐ Yes ☐ No ☐ Unsure

(2) How confident are you in the judgment you just made?
1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ (1 = not confident at all, 5 = extremely confident)

Part 3: Subjective Experience and Evaluation

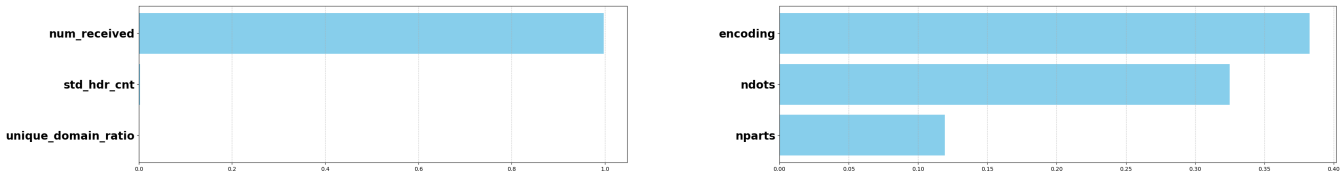
(1) How difficult did you find judging these emails overall? 1–5 (very easy to very difficult)

(2) During the task, how mentally demanding or stressful did you feel it was? 1–5 (very relaxed to very stressful)

(3) What are your primary factors for judging whether an email is a phishing attempt? (select all that apply)
☐ Sender's address ☐ Email greeting ☐ Urgency or threats
☐ Request for sensitive information ☐ Grammar and spelling
☐ Unprofessional design ☐ Gut feeling

F Detailed Descriptions on Baselines in Closed-World Experiment

In the closed-world experiment, we consider two representative feature-engineering-based baselines (D-Fence [71] and HelpHed [23]) based on email content and one LLM-based baseline ChatSpamDetector [65]. In addition, we also include two open-source anti-spam filtering solutions, SpamAssassin [104] and RSpamd [102].



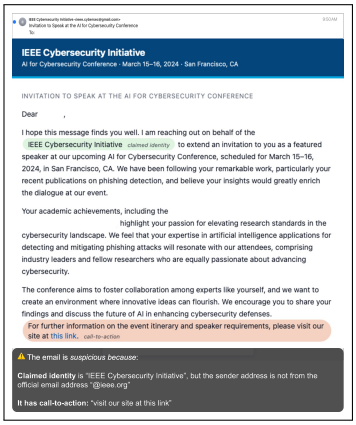
(a) Top-3 Important Features for D-Fence **(b) Top-3 Important Features for HelpHed**

Figure 11: Visualization of feature importance for D-Fence and HelpHed

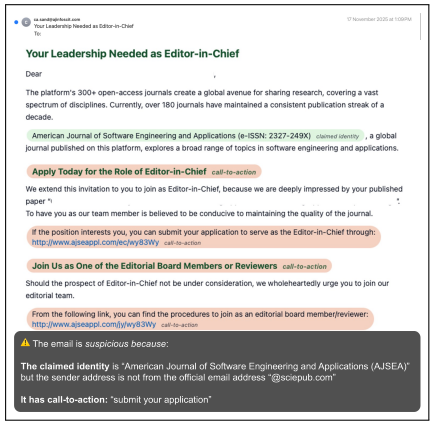
• **Feature-engineering-based baselines: HelpHed [23] and D-Fence [71].** Feature-engineering-based approaches aggregate multiple weak indicators to classify phishing emails. D-Fence is designed based on URL-based, structure-based, and text-based features, which are combined through a meta-classifier to determine the final classification. Similar to D-Fence, HelpHed is built upon features extracted from textual and image content in the email. HelpHed offers two options to ensemble these feature sets: (1) stacking-based ensemble fuses two learners using a multilayer perceptron (MLP) layer, and (2) voting-based ensemble takes the maximum confidence score among the two learners. We consider both in the study.

• **LLM-based baseline: ChatSpamDetector [65].** It is built upon ChatGPT via chain-of-thought prompting. The prompt instructs the LLM to identify indicators such as brand impersonation, signs of spoofing, and suspicious hyperlinks, ultimately delivering a final verdict based on its intermediate reasoning. We use GPT-4, which is the best version provided in the work.

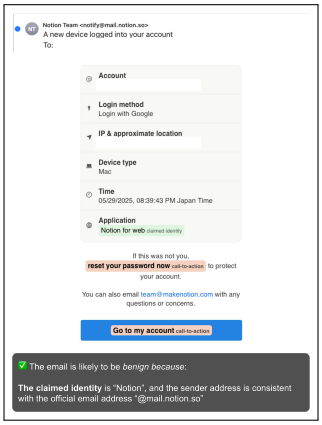
• **Rule-based and URL-Blacklist-based baselines: RSpamd [102] and SpamAssassin [104].** Modern commercial anti-spam filters are rule-based, with a number of rules to match a suspiciousness score to an email. Those rules (typically 200-500+ rules) encompass various aspects of email analysis, including header analysis, URL blacklist checking, IP blacklist checking, Bayesian filtering, and more. We use their default settings in the experiment.



(a) Phishing: personal Gmail account claiming "IEEE Cybersecurity Initiative".

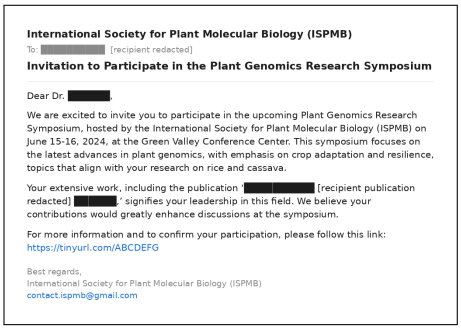


(b) Phishing: look-alike domain *scichemj.org* claiming the journal "AJSEA" (official: *sciencepg.com*).



(c) Benign: login alert from the subdomain `mail.notion.so`, consistent with "Notion" under eTLD+1 canonicalization.

Figure 12: Three of the ten emails used in the user study, shown in the treatment-group (Group B) view with *PiMRef*'s annotations (claimed identity, call-to-action) and the verdict explanation. The control group (Group A) saw the same emails without annotations. The full set of ten emails is available on our anonymous website [12].

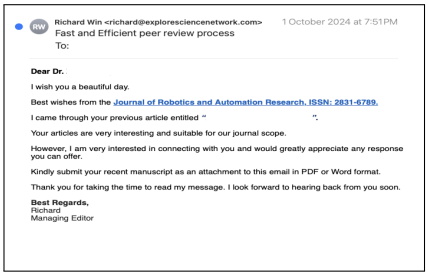


(a) Personal Gmail account claiming the ISPMB, a plant science society.

(b) Look-alike domain *fraunhofer-institute.com* claiming the Fraunhofer Institute (official: *fraunhofer.de*).

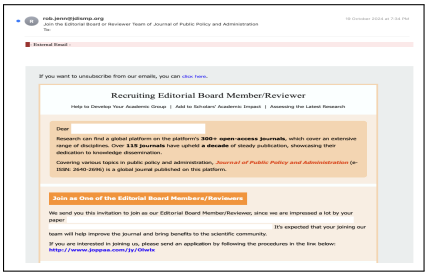
(c) Typo top-level domain *sme.orb* claiming the Society of Manufacturing Engineers (official: *sme.org*).

Figure 13: Three *SpearMail* samples generated against real researcher profiles, each impersonating a recognizable organization and luring the recipient toward an attacker-controlled link. Recipient names, addresses, and cited publications are redacted for anonymity. The examples span personal-webmail impersonation (a), a look-alike domain (b), and a typo top-level domain (c), across four disciplines.



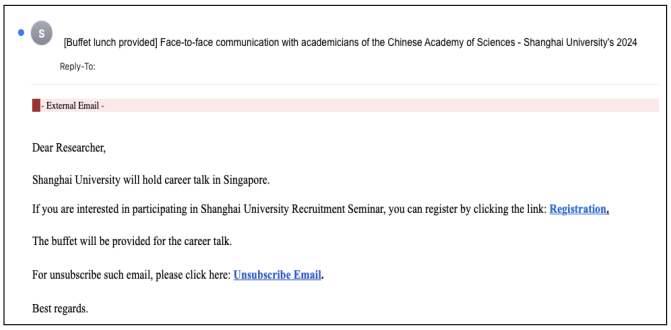
(a) Impersonating *Journal of Robotics and Automation Research*, whose official site is at *opastpublishers.com*, while the email sender is from *@explorescienceneetwork.com*, and the embedded link points to *jengineeringtech.com*.

(b) Impersonating *American Journal of Software Engineering and Applications*, whose official site is at *sciencepg.com*, while the email sender is from *@amjonc.org*, and the embedded link points to *amjsea.org*.

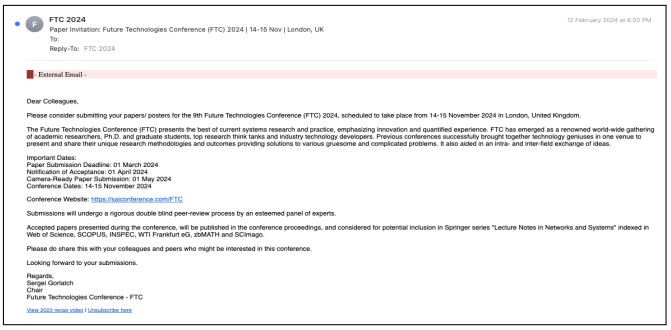


(c) Impersonating *Journal of Public Policy and Administration*, whose official site is at *sciencepg.com*, while the email sender is from *@jdismp.org*, and the embedded link points to *joppaa.com*.

Figure 14: Wild phishing email examples detected by *PiMRef*. In each case, the email claims a legitimate journal identity, but its submission link points to a domain different from the journal's official site (hosted on a different IP), contradicting the claimed identity.

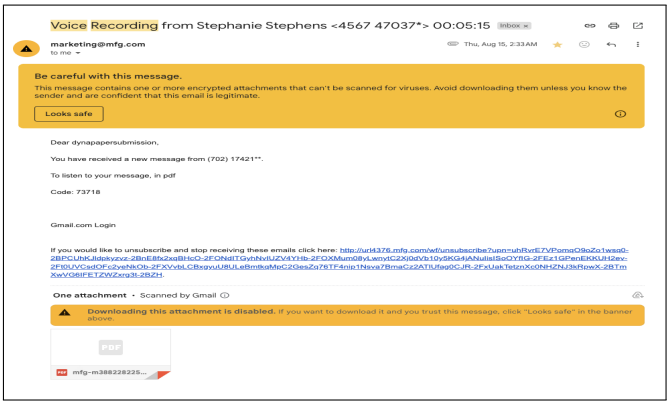


(a) FP1 Career talk invitation claiming Shanghai University, sent from a personal 163.com account rather than the university's official domain.



(b) FP2 Notification claiming the Future Technologies Conference, sent from a personal Gmail account rather than the organizing committee's official domain.

(c) FN1 Impersonates a co-worker and delivers a malicious attachment. The email contains no link or explicit call-to-action phrase, so the instruction recognition module does not fire.



(d) FN2 Claims an ambiguous sender identity that cannot be resolved against the knowledge base, so no identity-domain contradiction can be established; the payload is again a malicious attachment.

Figure 15: Failure cases in the open-world experiment.