Source-Aware Detection Of AI-Generated Phishing Emails Using Linguistic, Structural, And Contextual Feature Fusion

V. Saranya[1], Parthasarathi S[2], Faizal Mohamed P.[3], Shalini C.[4], Sharanshanth S.[5]
[1]*Assistant Professor of Computer Science Department, Sri Venkateshwaraa College of Engineering and Technology Ariyur, Puducherry*
[2,3,4,5]*Student, Department of Computer Science and Engineering, Sri Venkateshwaraa College of Engineering and Technology, Ariyur, Puducherry, India*

***Abstract*—Attackers can now use generative AI to write phishing emails that read smoothly, address the victim by name, and differ from one another in wording. That undermines detectors built around clumsy grammar, known bad URLs, or repeated templates. A second problem sits on the research side: many detectors report near-perfect scores on randomly split data, which often reflects the same campaign appearing in both training and test sets rather than genuine learning. In this paper we build a phishing-email detector that combines linguistic, structural, URL, and header features, and we test it three ways: on a random split, on sources the model has never seen, and, where timestamps allow, on later time periods. We compare rule-based, text-only, context-only, and fused models, and we also look at calibration, sensitivity to paraphrasing and formatting changes, and a small subgroup of phishing with documented AI-assisted origin. On [dataset names], the fused model reached [F1 = X, PR-AUC = Y] on the random split but [F1 = X′, PR-AUC = Y′] on held-out sources, a difference of [Δ] from the best text-only baseline (95% CI [a, b]). Paraphrasing shifted recall by [Δ]. We never treat AI authorship as something the model can infer from style; it appears only as a documented subgroup label. [Replace every bracket with measured values, then delete this sentence.]**

***Index Terms*—Phishing detection, generative AI, email security, natural language processing, feature fusion, explainable AI, distribution shift, calibration**

I. INTRODUCTION

Email is still one of the most common routes for social engineering. A phishing message tries to get the reader to hand over a password, send money, open a harmful attachment, or click through to a fake site. Whether it works depends partly on the delivery infrastructure, but just as much on the wording, who the message pretends to be, and when it arrives.

Large language models (LLMs) make it cheap to produce grammatical, well-targeted, and endlessly varied messages. Early studies of LLM-assisted spear phishing found that such emails can be as convincing as those written by people [6], [7]. Detectors that quietly lean on spelling mistakes, odd phrasing, or recycled templates may therefore get worse over time. Still, polished writing proves neither bad intent nor machine authorship: honest senders increasingly use AI writing tools, and human-written phishing can be perfectly fluent. Tools that try to tell machine text from human text are also easy to fool with paraphrasing [8]. For that reason, we aim to estimate *phishing risk*, and we make no claim about who or what wrote a message.

Evaluation is the second concern. Public phishing and legitimate-email corpora were collected in different years, by different methods, from different mail clients. A classifier can tell them apart from encodings, headers, boilerplate, or date formats without learning anything about deception. A random train/test split makes this worse, because near-duplicate emails from one campaign can land on both sides and inflate the score.

This paper tackles both problems. Our contributions are:

1 A reproducible pipeline that fuses text, structure, URL, and permitted header features, with explicit controls against data leakage.

2 A controlled comparison of text-only, context-only, and fused models under random, source-held-out, and temporal evaluation.
3 A robustness protocol based on paraphrase and formatting changes, plus calibration reporting to guide threshold choice.
4 A separate, carefully limited analysis of phishing whose AI-assisted origin is documented.
5 A protocol for checking whether explanations are stable and for spotting shortcut features during error analysis.

Research questions.
RQ1: Does fusing feature families beat text-only features? RQ2: How well do the models handle sources they have not seen, and later time periods? RQ3: How easily do paraphrasing and formatting edits throw them off? RQ4: Which features and email types cause the false positives and false negatives?

Hypotheses.
We state these before running any experiment and will report honestly whether each one is supported.

- H1: Fusion beats the best text-only model on the random split, but the gain shrinks on held-out sources if shortcut cues contribute to it.
- H2: Header-derived features give the largest gain on the random split and the largest loss on held-out sources, because they tend to be corpus-specific.
- H3: Models built on n-grams lose more recall under paraphrasing than contextual encoders do.
- H4: Calibration with Platt scaling or isotonic regression lowers ECE without materially changing ranking metrics such as PR-AUC.
- H5: Recall on documented AI-assisted phishing is not lower than on other phishing, provided the detector relies on intent cues rather than on writing quality.

The paper continues with related work (Section II), the method (Section III), the experimental setup (Section IV), results (Section V), a discussion (Section VI), and a conclusion (Section VIII). Limitations and ethics are covered in Section VII.

II. RELATED WORK

Classical and machine-learning approaches.
The first defenses were blacklists and hand-written rules, and empirical work showed that blacklists trail behind newly launched campaigns [1]. Later studies applied data mining and supervised learning to engineered features drawn from URLs, headers, and message content [2]. These methods are easy to interpret, but they depend on careful feature design and on indicators that are already known.

Text representations.
Word embeddings [3], attention-based architectures [4], and pretrained bidirectional encoders [5], [9] let a detector capture what a message means and how it sounds. The catch is that, without control over where the training data comes from, such models readily pick up dataset-specific shortcuts.

LLM-generated phishing.
Recent work looks at how LLMs can write and personalize phishing content, and at how well people and existing defenses cope with it [6], [7]. This motivates robust, source-aware evaluation. Few studies, however, report source-held-out generalization, perturbation robustness, calibration, and feature-family ablations together in one controlled setup.

AI-generated-text detection.
Origin detectors try to judge whether a piece of text came from a machine, but their reliability depends on the generator, the prompt, the decoding settings, and any later editing, and paraphrasing can defeat them [8]. We therefore do not use origin detection as a label or as a claim. Documented provenance is used only to define a subgroup for analysis.

Explainability and calibration.
Model-agnostic attribution methods such as LIME [10] and SHAP [11] are common in security tooling, and modern neural networks are often poorly calibrated [12], which matters whenever alerts are triaged by a score threshold. Our evaluation uses both.

Email-specific detection.
Fette et al. showed early on that a small set of structural and URL features could separate phishing from legitimate mail with a standard classifier [13]. Verma et al. later argued that natural-language cues, such as the way a message frames urgency or asks for action, carry signal that structural features miss [14].

Most of this line of work was evaluated on a handful of public corpora, notably the Enron corpus for legitimate mail [15] and Nazario-style collections of phishing, which have been combined into curated benchmark sets [16]. Because those sources differ in era and collection method, high scores on them do not by themselves show that a model has learned phishing behavior.

Evaluation pitfalls in security machine learning.
A growing body of work documents how security classifiers look better in the lab than in practice. Arp et al. catalogue recurring mistakes, including spatial and temporal bias and unrealistic base rates [17]. Pendlebury et al. show that ignoring time in the train/test split inflates malware-detection results [18], and Kapoor and Narayanan describe how data leakage undermines reproducibility across machine-learning research more broadly [19]. Geirhos et al. explain why models take shortcuts when a spurious feature predicts the label in the training data [20]. We apply these lessons directly: grouped splits, held-out sources, a temporal split, and an explicit shortcut check.

Robustness of text classifiers.
Small, meaning-preserving edits can change the prediction of strong text classifiers [23], and behavioral test suites such as CheckList offer a systematic way to probe for such failures [24]. Origin detectors are fragile in the same way: paraphrasing can defeat them [21], and they may be biased against non-native writers [22]. These findings justify our perturbation protocol and our decision not to build the detector on a human-versus-machine signal.

Positioning.
[Insert 1–2 paragraphs after the literature search that state precisely what prior work did and did not evaluate. Do not claim novelty until this search is complete.]

III. PROPOSED METHOD

A. Problem Formulation

Let an email be $e = (s, b, u, h)$ with subject s, body b, URL set u, and permitted header/metadata fields h. The goal is a function $f(e) \to p \quad [0,1]$, the estimated probability that e is phishing, with decision $\hat{y} = 1[p \geq \tau]$. The threshold τ is selected on validation data only. Phishing is the positive class.

Threat model.
We assume an attacker who can use an LLM to write and rewrite phishing text, who can paraphrase a message until it passes a filter, and who may send from a freshly registered domain or a compromised legitimate account. We assume the attacker cannot forge authentication results for a domain they do not control, and that they do not know the detector's parameters (a black-box setting). The defender sees the message content and only those header fields the data policy permits.

Out of scope are malware analysis of attachments, image-only emails, and attacks that need access to the mail server. These boundaries matter for interpreting results: a detector that does well here is not thereby safe against an adaptive attacker with query access.

B. System Overview

The pipeline has six stages: collecting data with provenance records; preprocessing, deduplication, and redaction; feature extraction; model training with tuning on validation data only; evaluation under leakage controls; and explanation with error analysis. We intend it as decision support for analysts, not as a system that blocks mail on its own.

C. Feature Families

Family	Examples	Cautions
Linguistic	word/char n-grams, sentence length, lexical diversity, punctuation, urgency and persuasion cue counts	No single cue proves phishing or AI authorship
Semantic	TF–IDF, document embeddings, fine-tuned transformer representation	Fit only on training data
Structure	subject–body relation, HTML ratio, link count, length	Differs across mail clients and corpora
URL/domain	URL count, anchor-text/URL mismatch, IP-literal and shortener indicators	Record collection time; reputation changes
Header (optional)	SPF/DKIM/DMARC results, reply-to mismatch	May be absent, forged, or corpus-specific; ablate to test shortcuts

D. Models

Baselines: majority class; keyword/rule filter; TF–IDF + logistic regression (LR); TF–IDF + linear SVM; gradient-boosted trees on numeric context features; and (resources permitting) a fine-tuned pretrained transformer encoder [9]. For TF–IDF features, term *t* in document *d* is weighted as

$$w(t,d) = \mathrm{tf}(t,d) \cdot \log\left(\frac{N}{1+\mathrm{df}(t)}\right),$$

with df and *N* computed on the training partition only. LR minimizes the regularized binary cross-entropy

$$L = -\frac{1}{M}\sum_{m=1}^{M} [y_m \log p_m + (1-y_m)\log(1-p_m)] + \lambda\|w\|_2^2,$$

with $p_m = \sigma(w^{\mathrm{T}}x_m + b)$.

E. Feature Fusion

Two fusion strategies are compared. Early fusion: concatenate standardized feature blocks, $x = [x_{text}; x_{struct}; x_{url}; x_{hdr}]$*, and train one classifier. Late fusion: train one calibrated model per family and combine scores,*

$$p = \sigma\left(\sum_k \alpha_k \cdot \mathrm{logit}(p_k) + \beta\right),$$

with α and β fitted on validation data. A block-ablation protocol removes one family at a time to measure its marginal contribution.

F. Calibration and Thresholding

Scores are calibrated on validation data with Platt scaling or isotonic regression. Calibration is reported with the Brier score and expected calibration error,

$$\mathrm{ECE} = \sum_b \frac{|B_b|}{M} \cdot |\mathrm{acc}(B_b) - \mathrm{conf}(B_b)|,$$

over equal-width probability bins *B*. The threshold τ is chosen on validation data to meet a stated b false-positive-rate target, or to maximize F1, and then frozen.

G. Explanation and Shortcut Analysis

For linear models we inspect coefficients; for tree and transformer models we use SHAP [11] or LIME [10]. Because explanations are approximate, we measure them rather than only display them:

- Stability: mean Jaccard overlap of the top-*k* attributed features across random seeds and across paraphrased copies of the same email.
- Counterfactual sensitivity: fraction of predictions that flip when the top-*k* attributed tokens are masked, versus *k* randomly chosen tokens (a permutation baseline).
- Shortcut flag: header- or format-only features whose ablation causes a larger source-held-out drop than content features.

H. Algorithm

Algorithm 1: Source-aware training and evaluation

1. Input: labelled corpus D with source *g*(*e*) and optional timestamp *t*(*e*).
2. Normalize and redact; remove exact duplicates; cluster near-duplicates (e.g., MinHash/Jaccard ≥ 0.8) and keep each cluster within one partition.
3. Build splits: (a) random grouped by cluster; (b) leave-one-source-out; (c) train-on-earlier / test-on-later if timestamps exist.
4. Fit all vectorizers and scalers on the training partition only.
5. Train each model; tune hyperparameters by validation cross-validation grouped by source.
6. Calibrate scores; choose τ on validation data; freeze.
7. Evaluate once on the untouched test partitions; compute metrics with bootstrap confidence intervals.
8. Run perturbation tests (Section IV-F) and explanation-stability tests (Section III-G).
9. Save splits, seeds, configuration, and predictions for audit.

IV. EXPERIMENTAL SETUP

A. Datasets and Provenance

Candidate public sources include established phishing corpora, legitimate-email corpora (e.g., Enron-derived, SpamAssassin ham), and curated combined phishing datasets. [Select final datasets, then verify each one's licence, terms, label definitions, collection period, and privacy status before use. List them in Table I.]

For every sample we retain an internal ID, label, source, date (if known), and label evidence.

Table I. Datasets (to be completed from the real data used).

Dataset	Period	Phishing	Legitimate	Licence	Role
[name]	[yrs]	[n]	[n]	[lic.]	[train/test]

AI-assisted subgroup. Where the evaluation includes AI-assisted phishing, it must be either (a) generated by the authors with a documented model, prompt template, and settings, under institutional approval and without releasing operational content, or (b) taken from a source with credible provenance labels. Authorship is never inferred from style.

B. Label Policy

Phishing: a message intended to deceive recipients into harmful actions (credential disclosure, payment, malicious link or attachment). Legitimate: messages from genuine senders, including transactional and marketing mail, per a written annotation policy. State how spam, scams without links, business-email compromise, and security simulations are handled, and how ambiguous items were adjudicated.

C. Preprocessing

Decode and normalize encodings while preserving security-relevant characters; separate subject, body, URLs, and metadata; remove exact duplicates and cluster near-duplicates before splitting; redact personal identifiers before feature generation; do not strip suspicious strings that carry signal.

D. Splits and Leakage Controls

We report three regimes: random (cluster-grouped), leave-one-source-out, and temporal. Test labels are never used for preprocessing, feature selection, thresholds, or tuning. Class counts per split are reported in Table II.

Table II. Split statistics (to be completed).

Split	Train (P/L)	Validation (P/L)	Test (P/L)
Random	[ ]	[ ]	[ ]
Source-held-out	[ ]	[ ]	[ ]
Temporal	[ ]	[ ]	[ ]

E. Experiments

ID	Purpose
E1	Text-only baseline
E2	Context-only (structure, URL, header)
E3	Fused model (early and late fusion)
E4	Source-held-out generalization
E5	Perturbation robustness
E6	Documented AI-provenance subgroup
E7	Feature-family ablation and shortcut check

Additional experiments. Four further analyses complete the study.

- E8, label-noise sensitivity: flip 5%, 10%, and 20% of training labels at random and measure how quickly each model degrades.
- E9, operating-point analysis: report recall at fixed false-positive rates of 0.1% and 1%, since an analyst team usually works to a false-alarm budget rather than to a default threshold.
- E10, slice analysis: report metrics by message length, presence of HTML, and (if the data allow) language, to find the slices where the model is weakest.
- E11, computational cost: report training time, inference latency per email, and memory for every model, so readers can judge the cost of the extra accuracy.

F. Perturbation Protocol

Semantics-preserving edits applied to test phishing emails only: (i) paraphrase by an LLM or back-translation, checked for preserved intent; (ii) formatting changes (HTML/plain-text conversion, whitespace, signature variation); (iii) typo and punctuation noise; (iv) URL-obfuscation-neutral changes. We report the change in recall at the frozen threshold τ and in PR-AUC. The attacks are intended for defensive evaluation; adversarial examples are not released.

G. Metrics

Precision = TP/(TP+FP); recall = TP/(TP+FN); F1 = 2PR/(P+R); FPR = FP/(FP+TN); PR-AUC; Brier score; ECE. Report a confusion matrix with class support. 95% confidence intervals use stratified bootstrap (1,000 resamples) resampled at the cluster level; model comparisons on the same test set use McNemar's test or paired bootstrap. Accuracy alone is not used because of class imbalance.

Statistical analysis plan.
We fix the primary metric and comparison in advance: PR-AUC on the source-held-out split, fused model versus the best text-only model.

Because accuracy-style metrics are misleading under class imbalance, PR-AUC is preferred over ROC-AUC for imbalanced test sets [25]. We train every model with at least five random seeds and report the mean and standard deviation.

Paired comparisons on the same test set use McNemar's test or a paired cluster bootstrap [26], and when several models are compared at once we correct p-values with the Holm–Bonferroni method. Alongside p-values we report effect sizes with confidence intervals, because a statistically significant difference can still be too small to matter operationally. A subgroup with fewer than 30 positive examples is described but not tested.

H. Implementation and Reproducibility

Table III. Hardware and software configuration (to be completed).

Item	Specification
Hardware	[CPU/GPU/RAM]
Python / libraries	[versions: scikit-learn, PyTorch, transformers, shap]
Models	[checkpoint names and hyperparameters]
Seeds	[list; results averaged over ≥ 5 seeds]
Data/Code	[dataset versions/hashes; repository link]

V. RESULTS

Populate this section only with measured results. Do not insert estimated or illustrative values.

A. Main Comparison

Table IV. Random-split test performance (mean ± 95% CI).

Model	Precision	Recall	F1	FPR	PR-AUC	ECE
Rule/keyword	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
TF–IDF + LR	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
TF–IDF + SVM	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
Context-only	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
Fused (early)	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
Fused (late)	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
Transformer (optional)	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]

B. Generalization (RQ2)
Table V. Random vs. source-held-out vs. temporal F1 and PR-AUC. [Same models; one row per model, one column per regime. Discuss the size of the drop and which model degrades least.]

C. Robustness (RQ3)
Table VI. Change in recall and PR-AUC under each perturbation type. [Report per model and perturbation.]

D. AI-Provenance Subgroup (E6)
[Report recalls on AI-assisted vs. other phishing at the frozen threshold, with subgroup sizes and CIs. State clearly if the subgroup is too small for conclusions.]

E. Ablation and Explanation Analysis (RQ4)
[Report per-family ablation results, top contributing features, explanation stability (Jaccard), counterfactual flip rates versus the random-mask baseline, and any shortcut features found.]

F. Error Analysis
[Categorize false positives (e.g., account notices, invoices, newsletters, internal alerts) and false negatives (e.g., targeted impersonation, short messages, URL-centric messages) with counts and anonymized, non-operational examples.]

Narrative template.
On the [split], the fused model achieved [metrics] versus [baseline metrics]; the difference was [Δ] (95%

CI [a, b]). On source-held-out testing, performance [changed by Δ]. Common false positives were [categories]; false negatives included [patterns].

VI. DISCUSSION

Read the results against the four research questions. If scores are high on the random split but fall sharply on held-out sources, the classifier has probably learned corpus artifacts, and the ablation should show which features are responsible. If fusion helps only under random splits, say plainly that its benefit does not survive source control. Then consider what this means in practice: how many false alarms an analyst would face at the chosen threshold, and whether the calibrated scores are good enough to prioritize review. A detector that flags polished language will also misjudge legitimate AI-assisted business email, and explanations help with diagnosis but cannot show why an attacker wrote a message or whether a model produced it.

VII. LIMITATIONS, ETHICS, AND RESPONSIBLE USE

- Dataset bias: public corpora over-represent certain languages, regions, mail clients, and eras, so results may not carry over.
- Label noise: both phishing labels and AI-provenance labels can be incomplete or wrong.
- Distribution shift: attackers adapt, and past performance does not guarantee future performance.
- Privacy: collect as little as possible, redact identifiers, follow dataset terms, and get institutional approval where required.
- Dual use: generated phishing and perturbed emails are used only for defensive evaluation, and live links, credentials, and operational templates are not released.
- Automation risk: use the system for triage, with human review and a way to appeal false positives.

Threats to validity.

- Internal validity: tuning choices, preprocessing, or leaked near-duplicates could favor one model. We guard against this with grouped splits, frozen test sets, and a shared preprocessing pipeline for all models.
- External validity: results on public corpora may not transfer to a real organization's mail flow, where the base rate of phishing is far lower. We therefore report recall at fixed low false-positive rates (E9) and discuss base-rate effects.
- Construct validity: a label of "phishing" or "AI-assisted" is only as good as its annotation procedure. We report how labels were obtained and avoid treating style as evidence of origin.
- Conclusion validity: small subgroups give wide confidence intervals. We report intervals everywhere and avoid claims about subgroups below the minimum size.

VIII. CONCLUSION AND FUTURE WORK

This paper studies phishing-email detection with linguistic, structural, URL, and header features under leakage-controlled, source-aware testing. [Summarize the measured findings in two or three sentences, and state whether fusion improved detection and generalization.]

Future work includes multilingual and cross-domain evaluation, adaptive attackers, privacy-preserving deployment, and learning from analyst feedback.

DATA AND CODE AVAILABILITY

[State where code, split IDs, and configuration are available, and which datasets are public vs. restricted.]

ETHICS STATEMENT AND CONFLICT OF INTEREST

[Institutional approval/exemption statement.] The authors declare [no] conflict of interest.

APPENDIX A. FEATURE DEFINITIONS

Feature	Family	Definition
Urgency-cue rate	Linguistic	Count of urgency phrases (e.g., deadline, "within 24 hours") per 100 words, from a fixed lexicon
Credential-request rate	Linguistic	Count of phrases asking to verify, confirm, or update account details, per 100 words

Mean sentence length	Linguistic	Average words per sentence
Type–token ratio	Linguistic	Unique words divided by total words (length-normalized)
Punctuation rate	Linguistic	Exclamation and question marks per 100 characters
Subject–body overlap	Structure	Cosine similarity of TF–IDF vectors of subject and body
HTML-to-text ratio	Structure	Length of HTML markup divided by visible text length
Link count	URL	Number of distinct URLs in the body
Anchor–URL mismatch	URL	1 if visible link text shows a different domain from the target
IP-literal or shortener URL	URL	1 if any URL uses a raw IP address or a known shortening service
Reply-To mismatch	Header	1 if the Reply-To domain differs from the from domain
SPF/DKIM/DMARC result	Header	Categorical authentication outcome, when present

APPENDIX B. HYPERPARAMETER SEARCH SPACE

Model	Parameter	Values searched
TF–IDF	Word n-gram range	(1,1), (1,2), (1,3)
TF–IDF	Character n-gram range	(3,5) or none
TF–IDF	Minimum document frequency	2, 5
TF–IDF	Maximum features	50,000; 200,000
Logistic regression	Regularization C (L2)	0.01, 0.1, 1, 10, 100

Model	Parameter	Values searched
Linear SVM	Regularization C	0.01, 0.1, 1, 10, 100
Gradient-boosted trees [27]	Max depth; learning rate; trees	3, 5, 7; 0.05, 0.1; 200, 500
Transformer [9]	Learning rate; epochs	2e-5, 3e-5, 5e-5; 2, 3, 4
Transformer	Batch size; max length	16, 32; 256, 512 tokens

Selection uses grouped cross-validation over the training sources, never the test partition. Near-duplicate clusters are formed with MinHash similarity [28].

APPENDIX C. ABLATION MATRIX

Config	Text	Structure	URL	Header
A1	Yes	No	No	No
A2	No	Yes	Yes	Yes
A3	Yes	Yes	No	No
A4	Yes	No	Yes	No
A5	Yes	No	No	Yes
A6	Yes	Yes	Yes	No
A7	Yes	Yes	Yes	Yes
A8	No	No	No	Yes

A8 is a deliberate diagnostic: if a header-only model scores well on the random split but collapses on held-out sources, header features are acting as a corpus shortcut.

APPENDIX D. PRE-SUBMISSION CHECKLIST

- Dataset licences, terms, and label definitions verified
- Near-duplicate clustering done before any split
- All vectorizers and scalers fit on training data only
- Threshold and calibration frozen before the test run
- Every table value traced to a logged run
- Confidence intervals and seeds reported
- Related-work search completed and novelty claim checked
- References verified in a scholarly database
- Ethics approval or exemption statement added
- Target venue template and page limit applied

REFERENCES

[1] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang, “An empirical analysis of phishing blacklists,” in *Proc. 6th Conf. Email Anti-Spam (CEAS)*, 2009.

[2] N. Abdelhamid, A. Ayesh, and F. Thabtah, “Phishing detection based associative classification data mining,” *Expert Syst. Appl.*, vol. 41, no. 13, 2014.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[4] A. Vaswani *et al.*, “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017.

[5] J. Devlin, M.-W. Chang, K. Lee, and K.

Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.: Human Lang. Technol. (NAACL-HLT)*, 2019.

[6] J. Hazell, "Spear phishing with large language models," arXiv preprint arXiv:2305.06972, 2023.

[7] F. Heiding, B. Schneier, A. Vishwanath, J. Bernstein, and P. Park, "Devising and detecting phishing: Large language models vs. smaller human models," arXiv preprint arXiv:2308.12287, 2023.

[8] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi, "Can AI-generated text be reliably detected?" arXiv preprint arXiv:2303.11156, 2023.

[9] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," arXiv preprint arXiv:1907.11692, 2019.

[10] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2016.

[11] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017.

[12] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in Proc. Int. Conf. Mach. Learn. (ICML), 2017.

[13] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in Proc. 16th Int. Conf. World Wide Web (WWW), 2007.

[14] R. Verma, N. Shashidhar, and N. Hossain, "Detecting phishing emails the natural language way," in Proc. Eur. Symp. Res. Comput. Security (ESORICS), 2012.

[15] B. Klimt and Y. Yang, "The Enron corpus: A new dataset for email classification research," in Proc. Eur. Conf. Mach. Learn. (ECML), 2004.

[16] A. I. Champa, M. F. Rabbi, and M. F. Zibran, "Curated datasets and feature analysis for phishing email detection with machine learning," in Proc. IEEE Int. Conf. Computing and Machine Intelligence (ICMI), 2024.

[17] D. Arp et al., "Dos and don'ts of machine learning in computer security," in Proc. USENIX Security Symp., 2022.

[18] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "TESSERACT: Eliminating experimental bias in malware classification across space and time," in Proc. USENIX Security Symp., 2019.

[19] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," Patterns, vol. 4, no. 9, 2023.

[20] R. Geirhos et al., "Shortcut learning in deep neural networks," Nat. Mach. Intell., vol. 2, pp. 665–673, 2020.

[21] K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer, "Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2023.

[22] W. Liang, M. Yuksekgonul, Y. Mao, E. Wu, and J. Zou, "GPT detectors are biased against non-native English writers," Patterns, vol. 4, no. 7, 2023.

[23] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is BERT really robust? A strong baseline for natural language attack on text classification and entailment," in Proc. AAAI Conf. Artif. Intell., 2020.

[24] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of NLP models with CheckList," in Proc. Annu. Meeting Assoc. Comput. Linguist. (ACL), 2020.

[25] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," PLoS ONE, vol. 10, no. 3, 2015.

[26] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," Neural Comput., vol. 10, no. 7, pp. 1895–1923, 1998.

[27] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2016.

[28] A. Z. Broder, "On the resemblance and containment of documents," in Proc. Compression and Complexity of Sequences, 1997.