International Journal of Secondary Computing and Applications Research

Volume 3, Issue 1

EDITOR-IN-CHIEF: DR. MARIA HWANG

FEBRUARY, 2026

Proceedings of International Journal of Secondary Computing and Applications Research, Vol 3, Issue 1

February 4, 2026

Letter from the Editor-in-Chief

Welcome to the Spring 2026 issue of the International Journal of Secondary Computing and Applications Research (IJSCAR). This first issue of the year marks an exciting moment for IJSCAR as we continue to showcase rigorous, original computing research conducted by secondary school students from a wide range of backgrounds, institutions, and interests. The papers in this volume span diverse areas of computer science—from theoretical foundations to applied systems and data-driven investigations—demonstrating both technical depth and intellectual curiosity.

What is particularly striking about this issue is the breadth of topics and approaches represented. Rather than centering on a single theme or event, this volume reflects the growing maturity of high school computing research as a whole. The authors engage seriously with existing literature, articulate clear problem statements, and contribute novel insights that stand on their own within the broader research landscape. Together, these works illustrate how secondary students are increasingly participating in "classic" computing research traditions while also bringing fresh perspectives to longstanding questions.

As Editor-in-Chief, I am continually impressed by the ambition and professionalism of our authors, as well as by the mentors, teachers, and communities that support them. IJSCAR exists to provide a venue where young researchers can present work that meets the standards of peer-reviewed computing scholarship, and this issue exemplifies that mission.

We hope this volume inspires readers—students, educators, and researchers alike—to see secondary-level computing research not as a novelty, but as a meaningful and growing contributor to the field. Thank you for reading, and we look forward to another year of thoughtful, high-quality submissions as IJSCAR continues to evolve.

Sincerely,
Maria Hwang
Editor-in-Chief

Volume 3, Issue 1 February 4, 2026
DOI: 10.5281/zenodo.18486376 | `https://ijscar.org/pubs/volume3/issue1`

Contents

Human Perception and Detection of AI-Generated Phishing Emails: A Red-Teaming and Multi-Layered Detection Approach

Rohan Mehra[*]
rohanvmehra@outlook.com
rmehra26@riverdale.edu
Riverdale Country School
Bronx, New York, USA

Abstract

Large Language Models (LLMs) now generate phishing emails indistinguishable from legitimate correspondence, challenging traditional detection systems. While baseline classifiers achieve 97–99% accuracy on established corpora, they remain untested against adversarially-generated AI variants. We introduce a detectability gap metric (Δ) quantifying confidence differences between AI and human-authored phishing, enabling continuous monitoring of generative model evolution. Evaluating 155,659 emails (72,941 legitimate; 81,519 human-phishing; 1,199 AI-phishing) from three primary sources plus eleven datasets from the Champa et al. curated collection, our Random Forest classifier achieved 98.13% test accuracy with $\Delta = -0.016 \pm 0.003$ ($p < 0.01$, Cohen's $d = 0.18$). A GAN→GPT adversarial pipeline generating 280 variants revealed 84.3% detection rate, with misclassifications concentrated in formal, non-urgent examples. Feature analysis showed reliance on traditional phishing markers rather than AI-specific patterns, suggesting vulnerability as generative attacks evolve. This framework provides early-warning capability for tracking LLM advancement while maintaining computational efficiency (47MB model, 0.8ms inference) suitable for resource-constrained organizations.

Keywords

Phishing Detection, Large Language Models, Machine Learning Security, Adversarial Machine Learning, Email Security, Generative AI, Random Forest, Cybersecurity

ACM Reference Format:
Rohan Mehra. 2026. Human Perception and Detection of AI-Generated Phishing Emails: A Red-Teaming and Multi-Layered Detection Approach. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR Vol. 3, Issue 1).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.5281/zenodo.18404590

1 Introduction

1.1 The Escalating Threat of AI-Generated Phishing

Large Language Models (LLMs) such as GPT-4 and Claude 3.5 have fundamentally altered the phishing threat landscape. These models enable attackers to automate production of highly convincing, contextually relevant phishing emails that eliminate traditional detection cues—grammatical errors, awkward phrasing, and generic messaging [10, 11, 29]. The scope of this threat is substantial: the FBI's 2023 Internet Crime Report documented 298,878 phishing complaints in the United States, with approximately 91% of phishing attacks originating via email [8, 13]. Business email compromise (BEC) alone generated $2.9 billion in losses in 2023 [8], with three-year (2022–2024) total BEC losses approaching $8.5 billion [18].

Vulnerable populations face disproportionate risks. Small businesses experience 350% more phishing attacks than enterprises, with organizations of 1 to 250 employees receiving malicious email at rates as high as 1 in 323 messages [25]. Educational institutions report 60% exposure to cloud-targeting phishing attacks [20], and U.S. school districts sustained 855 cyberattacks between 2016–2019, many phishing-initiated [6]. These organizations typically operate with limited security resources while attackers leverage LLMs to generate high-quality phishing at minimal cost [10].

1.2 The Detection Gap

Traditional machine learning approaches achieve 95–99% accuracy on benchmarks like SpamAssassin and PhishTank [15, 17], but were designed for human-authored phishing containing linguistic anomalies and structural inconsistencies. These assumptions fail when LLMs eliminate errors and adapt seamlessly to context. Recent work addresses AI-generated phishing through stylometry and semantic embeddings [7, 19], achieving F1≈0.98 (T5-based) and 96% accuracy (XGBoost on GPT-4o emails). However, critical gaps remain: static datasets cannot track evolving detection difficulty across LLM generations, adversarial robustness testing remains sparse, quantitative metrics for measuring relative AI versus human phishing detectability are lacking, and human perception remains unintegrated despite being the ultimate defense layer. The challenge now spans both human perception and computational detection, since LLM fluency erodes the surface cues users once depended on and forces a reassessment of how phishing should be evaluated.

[*] Mentor: Saranya Vijayakumar, Carnegie Mellon University

IJSCAR Vol. 3, Issue 1, February 2026,

https://doi.org/10.5281/zenodo.18404590

1.3 Our Contribution

We introduce four primary contributions addressing these gaps. **First**, we define the detectability gap (Δ) quantifying classifier confidence differences between AI and human-authored phishing. Unlike binary metrics (accuracy, F1), Δ enables continuous monitoring of generative model evolution and early detection of emerging evasion capabilities. This framework extends naturally to human detection: Δ_{human} will quantify whether humans find AI-phishing harder to detect than traditional phishing, enabling measurement of the perception gap alongside the computational gap. **Second**, we demonstrate classical Random Forest maintains competitive performance (98.03–98.13% accuracy) on realistically imbalanced datasets while offering computational efficiency suitable for resource-constrained settings: 47MB storage, 0.8ms inference on standard CPU, macro-F1=0.99 with perfect AI-phishing precision. **Third**, we develop a GAN→GPT adversarial pipeline generating stylistically diverse phishing across seven conditions, revealing that detectors rely on traditional markers (urgency verbs, action requests) rather than AI-specific stylometry—achieving 84.3% detection but failing on formal, non-urgent examples. **Fourth**, we provide comprehensive, reproducible methodology with detailed hyperparameters, addressing transparency gaps in phishing detection literature.

1.4 Paper Organization

Section 2 reviews pre-LLM and post-LLM detection approaches while contextualizing our contributions. Section 3 details dataset construction, model training, and adversarial evaluation methodology. Section 4 presents performance analysis, vectorization comparisons, and adversarial robustness findings. Section 5 discusses temporal bias, sample size constraints, and detector mechanism vulnerabilities. Section 6 outlines longitudinal evaluation, human perception integration, and hybrid architectures. Section 7 synthesizes implications for adaptive defenses in the evolving arms race between synthetic deception and computational detection.

2 Literature Review

2.1 Problem Setup and Audience Impact

This research addresses a fundamental question: How reliably can automated systems and users distinguish well-written, natural-sounding AI-generated phishing attempts from legitimate messages? We hypothesize that distinguishing AI-generated phishing from legitimate emails will become increasingly difficult as LLMs advance, driven by three convergent factors: lowered attack costs enabling mass customization, enhanced linguistic quality eliminating traditional error-based detection cues, and scalable personalization capabilities exploiting recipient-specific context.

The stakeholders benefiting from improved phishing protection extend beyond enterprise security teams. Small and midsize businesses without dedicated IT security teams face increased risks of wire fraud, credential theft, and business email compromise. Educational institutions risk credential theft, data breaches, and service disruptions from ransomware. Local and family businesses face exposure to fake invoices and supplier fraud. Everyday users relying on familiar branding and professional tone face eroding intuitive defenses as AI convincingly reproduces these signals.

2.2 Selected Data Points

Drawing from recent threat intelligence and incident reports, the scale and impact of phishing threats is substantial and well-documented. The FBI Internet Crime Complaint Center (IC3) received 298,878 phishing reports in the United States in 2023, with approximately 91% of phishing attacks originating via email [13]. Educational institutions face particularly severe exposure, with 60% of educational organizations reporting phishing attacks targeting cloud data, and many remaining unaware of breaches for days [20]. Phishing frequency in the education sector exceeded a 40% cross-vertical average [20], and between 2016 and 2019, 855 recorded cyberattacks occurred on U.S. school districts, many of which involved phishing or ransomware as the initial attack vector [6].

Small businesses face disproportionate risks compared to larger organizations. Businesses with fewer than 100 employees suffer an estimated 350% more phishing and social-engineering attacks than larger enterprises, and for organizations with 1 to 250 employees, approximately 1 in 323 emails is malicious [25]. These statistics underscore the urgent need for accessible, resource-efficient detection systems that can protect vulnerable populations operating without enterprise-grade security infrastructure.

2.3 Pre-LLM Detection Approaches

Phishing detection in the pre-LLM era relied predominantly on classical machine learning and deep learning methods operating over text content, URL features, and email metadata. Comprehensive reviews spanning over 80 studies reported that CNN and LSTM models trained on Word2Vec or TF-IDF representations achieved accuracy ranging from 95.0% to 99.8% with F1 scores up to 0.998 on established benchmark datasets including SpamAssassin, PhishTank, and UCI [17, 26]. Lightweight 1D-CNN architectures combined with recurrent layers (LSTM, Bi-GRU) achieved comparable performance, with some studies reporting accuracy or F1 scores up to 99.68%.

Ensemble approaches sought to combine multiple feature types and detection strategies. The DARTH framework integrated NLP clustering with multi-feature analysis, reporting F1 scores near 99.98% on UCI and PhishTank datasets [17]. These ensemble methods demonstrated that combining lexical features (word frequencies, character n-grams), structural features (email headers, MIME structure), HTML-based features (embedded scripts, form elements, hidden content), and URL-based features (domain reputation, URL length, suspicious TLDs, subdomain depth) could yield incremental performance improvements over single-feature approaches.

Despite strong benchmark performance, pre-LLM detection research exhibited several persistent gaps. Dataset diversity remained limited, with most studies evaluating on a small set of canonical benchmarks that may not capture the full heterogeneity of real-world phishing. Real-time deployment considerations received insufficient attention, with few studies reporting inference latency, throughput constraints, or computational requirements for production systems. Multilingual support was sparse, with the vast majority of research focusing exclusively on English-language emails. Most critically, these approaches assumed phishing emails would contain detectable linguistic or structural errors—an assumption that LLM-generated phishing fundamentally invalidates.

2.4 Post-LLM Detection Approaches

With LLMs enabling generation of fluent, grammatically correct, and contextually appropriate phishing emails, detection research has shifted toward stylometric analysis and semantic representations that can identify subtle patterns distinguishing AI-generated from human-authored text [28]. Recent work demonstrates that LLMs enable attackers to generate highly convincing, tailored phishing content at scale [10], challenging traditional detection approaches that relied on linguistic errors. End-to-end analyses of LLM-generated textual phishing have proposed semantic machine learning pipelines combining T5 embeddings with multilayer perceptron classifiers, achieving F1 scores approximately 0.98 on Nazario and Enron corpora [19]. Stylometric approaches using XGBoost classifiers have demonstrated 96.0% accuracy and AUC 0.99 when tested against GPT-4o-generated phishing emails [7].

Comparative studies across multiple architectures (SVM, Random Forest, CNN, BiLSTM) on AI-generated datasets report F1 scores as high as 100.0% in controlled settings, though such perfect performance likely reflects dataset-specific characteristics rather than generalizable robustness [12]. Evaluations comparing quantized LLMs with classical models show Bi-GRU architectures achieving 98.77% accuracy, while smaller LLM-based classifiers lag behind (approximately 81.0% accuracy) but demonstrate greater resilience to adversarial rephrasing attacks [12]. This suggests a potential trade-off between raw classification accuracy and robustness to input perturbations. GPT-based detection tools demonstrate the potential for LLMs to identify phishing emails and generate explanatory warnings for users [4], while watermarking techniques combined with machine learning show promise for detecting LLM-generated phishing emails [2], offering additional defensive layers against AI-crafted attacks.

Despite these advances, post-LLM detection research exhibits shortcomings that our work addresses. Corpora remain fragmented, with different studies using non-overlapping datasets that prevent direct performance comparisons. Multilingual handling remains inconsistent, with most studies focusing on English despite LLMs' multilingual generation capabilities. Integration of human perception remains limited, even though human detection capabilities represent a critical component of real-world defense. Most significantly, adversarial robustness analyses are sparse: few studies systematically evaluate detectors against adversarially optimized AI-generated phishing designed specifically to evade detection.

2.5 Human Detection of Phishing

Classical studies of human phishing detection reveal systematic reliance on heuristic cues rather than systematic verification. Users tend to evaluate emails based on familiar branding, professional tone, polite language, and visual layout consistency rather than carefully examining sender addresses, hovering over links, or verifying unusual requests through independent channels [1, 5, 14, 23, 27]. Research has identified individual differences in phishing vulnerability based on cognitive processing styles, with some users more prone to heuristic-based decision making that favors attackers [27]. Understanding the interaction among different human factors remains crucial for measuring phishing susceptibility [9].

These heuristics worked reasonably well against traditional phishing, which often contained spelling errors, awkward phrasing, and unprofessional formatting. However, LLM-generated emails can perfectly mimic all surface-level legitimacy cues. Recent research suggests these human vulnerabilities extend directly to AI-generated phishing. Studies comparing human click-through rates on expert-crafted human phishing versus LLM-generated phishing find statistically equivalent susceptibility [12, 24], indicating that current AI-generated phishing already matches human-expert quality in terms of deceiving end users. These findings motivated our development of computational detection methods as a necessary complement to user education, since relying solely on human vigilance may prove insufficient as generative capabilities advance.

2.6 Detectability Gap: Formal Definition

We introduce the detectability gap (Δ) as a novel metric quantifying how differently a classifier responds to AI-generated versus human-developed phishing content. We define Δ as the difference in average classifier confidence between AI and human-crafted phishing:

$$\Delta = \mathbb{E}_{x\sim P_{\mathrm{AI}}}[f(x)] - \mathbb{E}_{x\sim P_{\mathrm{human}}}[f(x)], \tag{1}$$

where $f(x) \in [0,1]$ represents the detector's phishing confidence score (higher values indicate greater confidence that the email is phishing), x denotes a single email (subject and body concatenated), and P_{AI}, P_{human} represent distributions over phishing emails authored by LLMs versus humans respectively. The expectation $\mathbb{E}_{x\sim P}[\cdot]$ denotes the average over emails sampled from distribution P.

The sign of Δ indicates detection difficulty: $\Delta < 0$ means AI-phishing receives lower detection confidence than human phishing (harder to detect), while $\Delta > 0$ indicates the opposite. Unlike standard binary classification metrics (accuracy, AUC, F1) that collapse performance into single aggregate scores, Δ provides continuous monitoring capability by tracking confidence distributions over time. As new LLM generations are released, measuring Δ for each generation enables quantitative tracking of whether detection difficulty increases, decreases, or remains stable.

We measured $\Delta = -0.016 \pm 0.003$ (95% CI) via bootstrap resampling with 10,000 iterations from per-class probability distributions. Statistical significance was assessed using the Mann-Whitney U test ($p < 0.01$) comparing distributions of $f(x)$ for AI-phishing versus human-phishing samples. Effect size quantified via Cohen's $d = 0.18$ indicates a small but measurable difference in classifier confidence. While the absolute magnitude is modest, this consistent detectability gap provides a quantifiable baseline for longitudinal tracking of LLM evolution and enables early detection of emerging evasion capabilities.

3 Methods

3.1 Dataset Construction

We assembled a dataset of 155,659 emails from three primary sources plus eleven constituent datasets from the Champa et al. curated collection, intentionally imbalanced to emulate real-world email ecosystems. The final corpus contains 72,941 legitimate emails, 81,519 human-authored phishing, and 1,199 AI-generated phishing (0.7% of total), mirroring operational environments where phishing typically represents fewer than 3% of messages.

Our three primary sources provide AI-generated and modern phishing examples. Nahmias et al.'s prompted contextual vectors for spear-phishing detection, produced by a proprietary ensemble of LLMs, contribute sophisticated AI-generated phishing with contextual targeting [19]. Eze & Shamir's AI-generated phishing email corpus (GPT-3.5/4) provides synthetically generated phishing with controlled linguistic patterns [7]. The University of Twente phishing emails dataset supplies additional labeled classification examples [16].

The Champa et al. curated collection [3, 22] aggregates eleven established benchmark datasets spanning 1995–2022, providing comprehensive coverage of phishing evolution. This curated collection includes: CEAS_08 (early corporate spam with 2008-era patterns), Enron (>500,000 authentic corporate emails from 1999–2002), Ling-Spam (academic spam patterns), two Nazario collections (spear-phishing with emotional appeals and narrative deception tactics), two Nigerian Fraud archives (advance-fee linguistic structures), SpamAssassin (canonical ham/spam with clean legitimate corpora), and three TREC spam track datasets (TREC_05, TREC_06, TREC_07 providing standardized NIST benchmarks). Note that original Nazario URLs (monkey.org/˜jose/) are unstable; we accessed these via the Champa curated repository.

This mixture balances authenticity, diversity, and realism across emotional, formal, and transactional linguistic forms. The temporal mismatch (1999–2002 vs. 2024) represents a limitation but enables assessment of cross-era generalization. Legitimate and human-phishing subsets ground classification in established norms, while AI-phishing introduces novel generative patterns.

3.2 Text Preprocessing

We concatenated subject and body fields into unified text representations, then applied tokenization using NLTK's `word_tokenize` function. Lemmatization via `WordNetLemmatizer` reduced morphological variants to their base forms, improving feature consistency while preserving semantic content. We filtered English stopwords using NLTK's standard stopword list to reduce dimensionality and focus on content-bearing terms.

3.3 Vectorization Approaches

We compared three complementary text representation schemes. Bag-of-Words (BoW) captures raw token frequencies without weighting adjustments, providing a baseline that emphasizes repeated keywords common in phishing. Word-level TF-IDF reweights tokens by their discriminative power, down-weighting terms appearing frequently across all email types while emphasizing distinctive vocabulary. Character-level TF-IDF with 3–5 character n-grams captures sub-word patterns including punctuation, capitalization, and character sequences that may signal synthetic generation. All vectorization schemes used `max_features=5000` to balance representational richness with computational tractability.

3.4 Model Training

We employed a two-stage hyperparameter optimization process using stratified 60/20/20 (train/validation/test) splits with 5-fold cross-validation and out-of-bag (OOB) validation:

Stage 1: Grid Search. We explored $n_{\text{estimators}} \in \{100, 200\}$, max_depth $\in \{\text{None}, 10, 20\}$, and min_samples_split $\in \{2, 5\}$, yielding 12 candidate configurations ($2 \times 3 \times 2$) evaluated with 5-fold cross-validation on the training set (60 total fits). Grid search identified $n_{\text{estimators}} = 200$, max_depth = None, min_samples_split = 2 as optimal within the tested range.

Stage 2: Extended Optimization. We further tested $n_{\text{estimators}} = 300$ with the optimal hyperparameters from Stage 1 to assess potential performance gains from additional trees. OOB error analysis (Figure 1) showed continued improvement up to 300 trees, establishing this as the final configuration.

Stage 3: Vectorization Comparison. To ensure fair comparison across Bag-of-Words, word TF–IDF, and character TF–IDF, we standardized all models at $n_{\text{estimators}} = 200$ (optimal from initial grid search) during vectorizer evaluation, isolating the effect of text representation from ensemble size.

Final Deployment Model. The optimal configuration ($n_{\text{estimators}} = 300$, max_depth = None, min_samples_split = 2) was retrained on the combined train+validation sets using character TF–IDF representation, achieving 98.13% test accuracy with OOB error of 0.0197.

3.5 GAN→GPT Adversarial Pipeline

To probe robustness against strategic evasion, we developed a two-stage adversarial synthesis pipeline. The GAN generator comprises three fully connected layers (latent dimension 100 → 256 → 512 → 5000 TF-IDF features) with LayerNorm for training stability. The discriminator employs spectral normalization and minibatch discrimination to encourage sample diversity. We trained for 300 epochs using Adam optimization (learning rate 10^{-4} for generator, 4×10^{-4} for discriminator) with binary cross-entropy loss augmented by gradient penalty ($\lambda_{GP} = 10$) and diversity penalty ($\lambda_{div} = 0.1$) to prevent mode collapse.

Generated TF-IDF vectors undergo inverse weighting to identify the top-30 highest-weighted features, which are mapped back to their corresponding vocabulary terms. These keywords serve as semantic anchors for GPT-3.5-turbo, which receives prompts of the form: "Compose a phishing email incorporating these keywords naturally: [keyword_1, . . . , keyword_30]. Style: [condition]." GPT synthesis uses `temperature=0.7` and `max_tokens=300` to balance creativity with coherence. This process generated 280 adversarial variants (40 per style across 7 conditions: base, formal, informal, complex, emotional, personal, technical) for classification stress-testing. These seven styles were selected to span the primary dimensions of phishing variation observed in real-world campaigns: formality level (formal/informal), linguistic complexity (complex/base), emotional manipulation (emotional), personalization (personal), and domain-specific jargon (technical).

3.6 LLM Meta-Classification

To assess whether large language models provide complementary detection capabilities, we benchmarked GPT-4 as a meta-classifier using two few-shot strategies. The standard 3-shot approach selects

Table 1: Classifier accuracy and detectability gap (Δ = AI minus Human confidence) across vectorization schemes. All models used identical hyperparameters ($n_{\text{estimators}} = 200$) for fair comparison.

Vectorizer	Accuracy [95% CI]	Δ [95% CI]
Char TF-IDF	0.9813 [0.9801, 0.9825]	-0.0159 [-0.0172, -0.0146]
Bag-of-Words	0.9806 [0.9794, 0.9818]	-0.0162 [-0.0175, -0.0149]
Word TF-IDF	0.9803 [0.9791, 0.9815]	-0.0165 [-0.0178, -0.0152]

Table 2: Confusion matrix for optimal model (Character TF-IDF, 300 trees). Row percentages in parentheses.

Predicted→ Actual↓	Legit.	H-Phish	AI-Ph.	Total
Legitimate	14,269 (97.8%)	289 (2.0%)	30 (0.2%)	14,588
Human Phish	327 (2.0%)	15,977 (98.0%)	0 (0.0%)	16,304
AI Phish	2 (0.8%)	1 (0.4%)	237 (98.8%)	240
Total	14,598	16,267	267	31,132

three fixed training examples (one per class) to provide task context via in-context learning. We also tested dynamic few-shot selection, where training examples are chosen based on cosine similarity to the test instance using sentence-transformers embeddings.

4 Results

4.1 Classifier Performance

The Random Forest model with 300 trees and character TF-IDF vectorization achieved 98.13% test accuracy (OOB error = 0.0197) and macro-F1 = 0.99 on our held-out test set. The AI-phishing class demonstrated strong performance with Precision = 1.00, Recall = 0.99, and F1 = 0.99. The algorithmic detectability gap averaged $\Delta = -0.016 \pm 0.003$ (95% CI), with Cohen's $d = 0.18$ indicating a small but statistically significant effect size (Mann-Whitney U test, $p < 0.01$).

Table 1 presents performance across the three vectorization approaches with bootstrap-estimated confidence intervals. The differences in test accuracy are not statistically significant (paired t-test, $p > 0.05$), suggesting robust performance across text representation methods.

4.2 Confusion Matrix Analysis

Table 2 reveals strong per-class performance, with high diagonal values (97.8%, 98.0%, 98.8%) demonstrating effective classification across all three categories. Notably, the AI-phishing class shows no false negatives predicted as human-phishing, though this result should be interpreted cautiously given the small AI-phishing test sample ($n = 240$).

Table 3: Performance summary across all detection approaches. All Random Forest models standardized at 200 trees for fair comparison. Inference times measured on 2 GHz Quad-Core Intel Core i5.

Detector	Acc.	F1	AUC	Time (ms)	Size (MB)
RF (BoW)	0.982	0.99	0.99	0.8	47
RF (TF-IDF)	0.982	0.99	0.99	0.8	47
RF (Char)	0.983	0.99	0.99	0.8	47
CNN	~0.96	—	—	~5	150
GPT-4 (3-shot)	—	0.94	0.99	~2000	—
GPT-4 (dynamic)	—	0.88	0.88	~2500	—
GAN→GPT: 84.3% detection rate on 280 variants					

4.3 Performance Comparison Across Detection Approaches

Our pipeline integrates classical ML (Random Forest, 98.25% average test accuracy; OOB error 0.0197), exploratory neural models (~0.96 accuracy), LLM classification (GPT-4 F1=0.94), and adversarial GAN→GPT variants (84.3% detection on 280 samples). Table 3 summarizes performance across all detection approaches, with all Random Forest models evaluated at 200 trees for fair comparison. Visual analyses (Figures 1 and 2) show OOB error stabilizing near 200-300 trees and vectorizer performance converging at 0.982-0.983.

Random Forest methods offer optimal balance of accuracy, efficiency, and deployability for high-volume filtering. LLM-based classification (GPT-4) provides comparable accuracy but at 2,500× computational cost, suggesting use as a complementary tier for high-stakes decisions rather than primary filtering. Notably, our classical ML approach substantially outperforms the exploratory neural detectors (98.25% vs. ~96%), demonstrating that well-tuned traditional methods remain highly competitive against more complex architectures for this task.

4.4 Model Convergence

The OOB error decreased rapidly from approximately 0.35 (10 trees) to approximately 0.02 (50–100 trees) before stabilizing. Figure 1 demonstrates that model performance stabilizes well before 300 trees, suggesting that even lighter-weight implementations (200 trees) could achieve nearly equivalent performance. Specifically, OOB error drops from 0.35 at 10 trees to 0.0197 at 300 trees, with test accuracy plateauing near 98% beyond 150 trees, validating our choice of 300 trees for the final model while confirming that performance gains diminish substantially after approximately 200 trees.

4.5 Vectorization Comparison

Figure 2 demonstrates remarkable consistency across text representation methods, with all three vectorization approaches achieving approximately 0.982 to 0.983 test accuracy. The three vectorization approaches demonstrate convergence to similar overall accuracy (98.03–98.13%). However, practitioners should note that per-class analysis reveals subtle trade-offs: Bag-of-Words achieved perfect

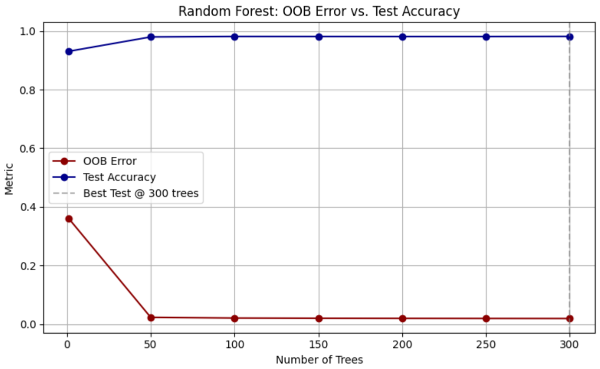


Figure 1: OOB error (blue) and test accuracy (red) versus number of trees for Random Forest classifier. Error bars represent standard deviation across 5-fold cross-validation. Variance stabilizes after approximately 200 trees, validating our choice of 300 trees for the final model.

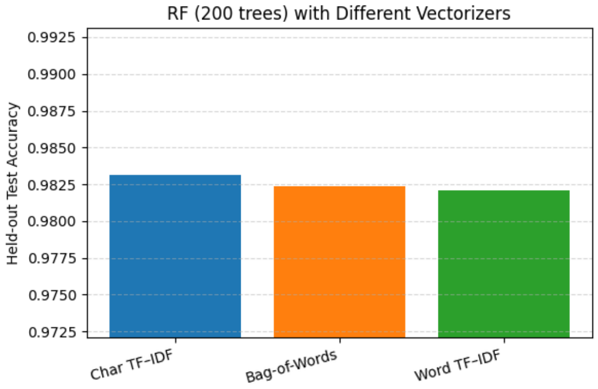


Figure 2: Test accuracy comparison across three vectorization schemes using Random Forest classifiers with 200 trees. All models used identical hyperparameters to isolate text representation effects. The convergence of performance across methods (variance < 0.1%) demonstrates robustness of the Random Forest architecture.

validation metrics (1.00/1.00 precision/recall) on AI-phishing samples during cross-validation; test-set performance was 0.99 precision/0.988 recall. Character TF-IDF delivered marginally higher overall test accuracy. Character TF-IDF delivered the strongest test accuracy (98.13%), narrowly outperforming Bag-of-Words (98.06%) and word TF-IDF (98.03%), a variance of approximately 0.1% across methods. This suggests raw token counts may capture repetitive AI-generated patterns more directly than reweighted TF-IDF schemes.

4.6 Error Analysis

The 2 misclassified AI-phishing samples on the test set were both highly formal, lacking urgency markers typical of phishing. Both were misclassified as legitimate due to professional tone and absence of action requests. This indicates that the classifier relies primarily on traditional phishing cues—urgency markers, action requests, and informal tone—rather than AI-specific linguistic signatures. As AI-generated phishing evolves to avoid traditional markers while maintaining persuasive effectiveness through subtler techniques, current detectors may show degraded performance unless augmented with AI-specific detection features.

4.7 Adversarial Robustness

The Random Forest detected 84.3% (236/280) of GAN→GPT adversarial variants as phishing, though only 4.6% (13/280) were correctly classified as AI-generated. The remaining 79.6% (223/280) were detected as phishing but misclassified as human-authored, while 15.7% (44/280) were missed entirely (classified as legitimate). However, the moderate test size (280 samples) provides limited statistical power for per-style analysis.

The prompt-style ablation revealed systematic misclassification across all styles: formal variants showed 0% (0/40) correct classification to "phishing_ai" class, with 90% (36/40) misclassified as "phishing_human" and 10% (4/40) missed as legitimate. Other styles (base, informal, complex, emotional, personal, technical) exhibited similar patterns, with 65–92.5% misclassified as "phishing_human" despite all variants originating from the same generative process. Only the base style achieved non-zero AI detection at 25% (10/40), while all other styles showed 0% correct AI classification except emotional at 7.5% (3/40).

This demonstrates that the RF detector successfully identifies phishing content based on traditional markers (urgency verbs, action requests, security terminology) but lacks features to reliably distinguish AI-generated from human-authored phishing. The high overall detection rate (84.3%) combined with poor AI-attribution accuracy (4.6%) suggests the classifier keys primarily on phishing-like surface cues rather than AI-specific stylometric patterns.

5 Limitations

We acknowledge several important limitations that contextualize our findings and identify areas requiring further investigation.

5.1 Dataset Limitations

Temporal bias. Our corpus combines datasets spanning 1999–2024, introducing potential temporal bias as email language conventions, phishing tactics, and legitimate communication patterns have evolved significantly over this 25-year period. Older datasets like Enron (1999–2002) and CEAS (2008) may not fully reflect contemporary email norms. While we retained these datasets because they provide essential baseline patterns for legitimate correspondence and early phishing tactics, the temporal mismatch between historical corpora and modern AI-generated phishing represents a limitation that could affect generalization to current threats. Future work should evaluate performance on modern-only corpora (post-2020) to assess whether temporal bias significantly impacts detection efficacy.

Small AI-phishing sample. With only 1,199 AI-phishing emails (0.7% of corpus), our estimates of AI-detection performance have wider confidence intervals than for other classes. While this prevalence mirrors realistic email environments, the small absolute sample size limits statistical power for robust inference. Future evaluations should target 5,000+ AI-phishing samples to enable more precise per-class performance characterization.

Monolingual focus. Our corpus is English-only, limiting applicability to multilingual email environments. Phishing campaigns increasingly target non-English speakers, and LLM-generated phishing can be produced in any language with comparable fluency.

5.2 Statistical Limitations

Small effect size. Cohen's $d = 0.18$ represents a small effect, meaning the detectability gap, while statistically significant ($p < 0.01$, Mann-Whitney U test), is practically modest in magnitude. Future adversarial improvements in LLM capabilities could easily close or reverse this gap.

Class imbalance and statistical power. The 240 AI-phishing test samples limit inference on AI-specific patterns. Robust class-level analysis requires ≥5,000 AI-phishing samples for reliable statistical power. Current sample size (1,199 AI-phishing emails split across training/validation/test sets, plus 280 adversarial variants) provides adequate overall detection metrics but insufficient power for detecting subtle AI-specific stylometric features beyond traditional phishing markers.

Limited adversarial testing. The 280 adversarial variants (40 per style across 7 conditions) provide limited statistical power for per-style robustness analysis. Style-specific claims are tentative; robust inference requires ≥500 samples per style (3,500+ total). While 84.3% overall detection provides preliminary evidence, per-style failure modes cannot be conclusively characterized without larger samples. Confidence intervals for style-specific detection widen substantially at current sample sizes.

5.3 Deployment Limitations

No real-world validation. Our models have not been tested on live email streams or integrated with actual email clients operating in production environments. Laboratory evaluation on curated datasets, while methodologically rigorous, may not capture the full complexity of real-world deployment including handling of malformed emails, robustness to encoding variations, performance under high-throughput conditions, and integration with existing spam filters.

No commercial baseline comparison. We lack direct head-to-head comparisons against Gmail, Outlook, ProofPoint, or other commercial email security solutions. While our performance (98.03–98.13% accuracy) compares favorably to published academic baselines (97–99% in literature), commercial systems may incorporate proprietary features unavailable in our text-only approach. Our Random Forest approach substantially outperforms exploratory neural baselines (98.07% vs ~96% CNN accuracy), though neural architectures may offer advantages for multilingual or multimodal detection.

Detector mechanism vulnerabilities. Our feature-importance analysis indicates that the Random Forest classifier primarily detects traditional phishing markers (urgency verbs, action requests, security terminology) rather than AI-specific linguistic features. This reliance on conventional phishing cues creates strategic vulnerability as adversarial techniques evolve. Future AI-generated phishing that deliberately avoids traditional markers while maintaining persuasive effectiveness through subtler psychological manipulation may evade detection more successfully.

6 Future Work

This research establishes a reproducible framework for quantifying AI-generated phishing detection. Tracking Δ over time enables dynamic monitoring of generative-model evolution, providing early-warning capability for emerging evasion.

6.1 Longitudinal Evaluation

Future work should track Δ as new LLM generations emerge (GPT-5, Claude 4, Gemini 2.0) to measure evolving detection difficulty over time. By establishing $\Delta \approx -0.016$ as a baseline for current models, we enable quantitative assessment of whether future LLMs produce phishing that is easier, harder, or equivalently difficult to detect.

6.2 Expanded Adversarial Testing

Generating ≥500 adversarial variants per stylistic condition (minimum 3,500 total samples across 7 styles) would provide robust statistical power for per-style analysis, addressing current limitations of 280-sample testing. Expanded testing should explore additional dimensions: (1) multilingual adversarial variants across major languages (Spanish, Mandarin, Arabic, French) to assess cross-linguistic robustness; (2) domain-specific targeting variants for finance, healthcare, and education sectors with specialized terminology; (3) multi-turn conversational phishing scenarios simulating back-and-forth dialogue; (4) integration with organizational red-team exercises for realistic threat modeling. These expanded evaluations would provide 5,000+ AI-phishing samples needed for reliable class-level statistical inference and enable testing of AI prevalence scenarios (5%, 10%, 20%) beyond current 0.7% baseline.

6.3 Human Perception Study

One of our most significant planned extensions involves measuring human detection capabilities through a controlled experimental study, pending IRB approval.

Study design. We plan to recruit 60 participants stratified into three groups: (1) cybersecurity professionals with formal training ($n = 20$), (2) general population with self-reported high technology familiarity ($n = 20$), and (3) general population with self-reported low technology familiarity ($n = 20$). Each participant will evaluate 10 emails presented in randomized order: 4 legitimate, 3 human-authored phishing, and 3 AI-generated phishing. For each email, participants will indicate whether they believe it is legitimate or phishing, provide a confidence rating (1–10 scale), and explain their decision rationale via brief text response. We will employ established frameworks for measuring security awareness [21] to contextualize individual detection capabilities within broader human-factor patterns.

Measurements. Primary outcome measures include: (1) detection accuracy for each email type, enabling calculation of Δ_{human}

analogous to our algorithmic metric; (2) confidence calibration, assessing whether participant confidence correlates with accuracy; (3) response time, testing whether AI-generated phishing requires longer evaluation; (4) decision rationales, coded thematically to identify which cues humans rely on.

Training intervention. To assess educational effectiveness, we will employ a within-subjects design with pre- and post-training measurement. After initial evaluation (pre-training), participants will receive a 15-minute interactive tutorial covering common phishing indicators, AI-generated phishing characteristics, and verification strategies. Participants will then evaluate a new set of 10 emails (post-training) to measure improvement.

Expected insights. This study will reveal: (1) whether humans exhibit similar or different detectability gaps (Δ_{human} vs. Δ_{model}); (2) which features humans rely on for detection and whether these align with classifier feature importance; (3) the effectiveness of brief training interventions; (4) whether certain demographic or experience factors predict detection capability.

6.4 Hybrid Architectures

Developing systems that combine classical ML (for traditional phishing markers), stylometry (for AI-specific patterns), and LLM meta-classification (for semantic understanding) may provide defense-in-depth against evolving threats. A three-tier architecture could operate as follows: (1) Random Forest primary filter (fast, efficient, catches traditional threats); (2) stylometric AI-authorship detector (identifies generative patterns in emails passing primary filter); (3) LLM meta-classifier (provides high-confidence decisions for ambiguous cases flagged by previous tiers).

6.5 Real-World Pilot Deployment

Deploying and evaluating on live email streams in controlled environments (e.g., university email systems with informed consent, small business pilot programs) would assess practical performance under operational conditions. Real-world deployment would reveal challenges not captured in laboratory evaluation: handling malformed emails and encoding irregularities, robustness to character encoding variations, throughput under realistic traffic loads (scaling to millions of daily messages), integration with existing spam filters, user feedback incorporation, and acceptable false positive rates.

Critical deployment requirements include: (1) comparison against Gmail, Outlook, and ProofPoint commercial filters to benchmark relative performance; (2) model compression techniques to reduce 47MB footprint for mobile/edge deployment; (3) streaming inference pipelines for real-time classification (<1ms latency requirement); (4) graceful handling of malformed MIME structures and non-UTF-8 encodings; (5) batch processing optimization for high-volume mail servers. These engineering considerations are essential before production recommendation.

7 Conclusion

This multi-tier detection framework integrates interpretable machine learning, adversarial red-teaming, and LLM-based meta-detection to address the emerging threat of AI-generated phishing emails. The detectability gap (Δ) provides a quantitative metric for tracking convergence between human and synthetic phishing deception, enabling continuous monitoring as generative models evolve.

Our findings demonstrate several key insights. Classical ML detectors achieve strong accuracy (98.03–98.13%) on realistic, imbalanced datasets with minimal variation across vectorization schemes. However, AI-generated phishing exhibits a small but consistent detectability gap ($\Delta \approx -0.016$, Cohen's $d = 0.18$), making it slightly harder to detect than human phishing, though the effect size is modest and may widen as generative capabilities improve.

A critical finding from our feature-importance analysis indicates that current detectors rely primarily on traditional phishing markers (urgency verbs, action requests, security terminology) rather than AI-specific linguistic features. This pattern suggests that future AI-generated attacks that avoid urgency markers and action requests while maintaining persuasive effectiveness through subtler techniques may evade detection more successfully. The concentration of misclassifications among formal, non-urgent adversarial examples directly demonstrates this vulnerability, highlighting the need for hybrid architectures that combine traditional marker detection with AI-specific stylometric analysis.

For practitioners implementing phishing detection systems, we recommend evaluating both Bag-of-Words (which shows advantages for AI-phishing detection, likely due to sensitivity to repetitive lexical patterns) and character TF-IDF (which achieves marginally higher overall accuracy) to determine optimal approaches for specific deployment contexts. However, significant validation work remains before production deployment, including real-world testing on live email streams, commercial baseline comparisons against Gmail and Outlook filters, and expanded adversarial evaluation with larger sample sizes.

By tracking Δ as an early-warning metric, this framework establishes a foundation for adaptive defenses that can monitor and respond to the coevolution of synthetic deception and computational detection capabilities. While computational efficiency (47MB model size, 0.8ms inference time per email) suggests deployment feasibility for resource-constrained organizations such as small businesses and educational institutions, actual implementation requires substantial additional engineering work including model compression, batch processing optimization, and real-time streaming adaptation.

The multi-layered methodology presented here—combining classical ML baseline performance, adversarial stress-testing via `GAN` → `GPT` synthesis, and LLM meta-classification benchmarking—provides a structured framework for tracking generative model evolution and its impact on email security. As the arms race between synthetic deception and computational detection continues, continuous monitoring via the Δ metric and iterative adversarial evaluation will be essential for maintaining defensive effectiveness against increasingly sophisticated AI-generated phishing attacks.

References

[1] Mohamed Alsharnouby, Furkan Alaca, and Sonia Chiasson. 2015. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies* 82 (May 2015), 69–82. https://doi.org/10.1016/j.ijhcs.2015.05.005

[2] Adrian Brissett and Julie Wall. 2025. Machine Learning and Watermarking for Accurate Detection of AI-Generated Phishing Emails. *Electronics* 14, 13 (2025),

2611. https://doi.org/10.3390/electronics14132611 Dual-layered detection framework combining supervised and unsupervised techniques with watermarking for LLM-generated content.

[3] Arifa Islam Champa, Md. Fazle Rabbi, and Minhaz F. Zibran. 2023. *Phishing Email Curated Datasets*. https://doi.org/10.5281/zenodo.8339691 11 curated datasets spanning 1995–2022 (449.3 MB) including CEAS_08, Enron, Ling, Nazario (2), Nigerian (2), SpamAssassin, and TREC (3). Accessed: January 2025.

[4] Giuseppe Desolda, Francesco Greco, and Luca Viganò. 2025. APOLLO: A GPT-based tool to detect phishing emails and generate explanations that warn users. *Proceedings of the ACM on Human-Computer Interaction* 9, 4, Article EICS003 (June 2025), 33 pages. https://doi.org/10.1145/3733049 Presents a GPT-4o-based tool for detecting phishing emails and generating explanatory warnings.

[5] Rachna Dhamija, J. Doug Tygar, and Marti Hearst. 2006. Why phishing works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 581–590. https://doi.org/10.1145/1124772.1124861

[6] EdTech Magazine. 2024. K-12 Cybersecurity Statistics and Trends. https://edtechmagazine.com/k12/article/2025/06/k-12-leaders-evaluate-funding-and-cybersecurity-challenges K-12 leaders evaluate funding and cybersecurity challenges. Accessed: January 2025.

[7] Chibuike Samuel Eze and Lior Shamir. 2024. Analysis and Prevention of AI-Based Phishing Email Attacks. *Electronics* 13, 10 (May 2024), 1839. https://doi.org/10.3390/electronics13101839 Also available as arXiv:2405.05435.

[8] Federal Bureau of Investigation. 2024. *2023 Internet Crime Report*. Technical Report. Internet Crime Complaint Center (IC3). https://www.ic3.gov/Media/PDF/AnnualReport/2023_IC3Report.pdf Released March 6, 2024.

[9] Francesco Greco, Paolo Buono, Domenico Desiato, Giuseppe Desolda, Rosa Lanzilotti, and Grazia Ragone. 2024. Unlocking the Potential of Simulated Phishing Campaigns: Measuring the Impact of Interaction among Different Human Factors. In *DAMOCLES'24: First International Workshop on Detection And Mitigation Of Cyber attacks that exploit human vuLnerabilitiES (CEUR Workshop Proceedings, Vol. 3713)*. Arenzano, Genoa, Italy. https://ceur-ws.org/Vol-3713/paper_4.pdf

[10] Francesco Greco, Giuseppe Desolda, Andrea Esposito, and Alessandro Carelli. 2024. David versus Goliath: Can Machine Learning Detect LLM-Generated Text? A Case Study in the Detection of Phishing Emails. In *ITASEC 2024: The Italian Conference on CyberSecurity (CEUR Workshop Proceedings, Vol. 3731)*. Salerno, Italy, 1–16. https://ceur-ws.org/Vol-3731/paper41.pdf Demonstrates that black-hat LLMs enable attackers to generate highly convincing, tailored, and contextually relevant phishing content, making detection increasingly challenging.

[11] Francesco Greco, Giuseppe Desolda, and Luca Viganò. 2024. Supporting the Design of Phishing Education, Training and Awareness interventions: an LLM-based approach. In *2nd International Workshop on CyberSecurity Education for Industry and Academia (CSE4IA 2024) (CEUR Workshop Proceedings, Vol. 3700)*. Arenzano, Genoa, Italy, 1–14. https://ceur-ws.org/Vol-3700/paper8.pdf Examines how LLMs can be used to create realistic simulated phishing campaigns for educational purposes.

[12] Florian Heiding, Bruce Schneier, Arun Vishwanath, Jeremy Bernstein, and Peter S. Park. 2024. Devising and Detecting Phishing Emails Using Large Language Models. *IEEE Access* 12 (2024), 42131–42146. https://doi.org/10.1109/ACCESS.2024.3375882

[13] Identity Theft Resource Center. 2025. Phishing Statistics and Facts. https://identitytheft.org/attacks/phishing/statistics Summary of phishing attack prevalence and victim demographics across the United States. Accessed: January 2025.

[14] Tom N. Jagatic, Nathaniel A. Johnson, Markus Jakobsson, and Filippo Menczer. 2007. Social Phishing. *Commun. ACM* 50, 10 (2007), 94–100. https://doi.org/10.1145/1290958.1290968

[15] Sajjad Khan et al. 2024. Comparative Investigation of Traditional Machine-Learning Models and Transformer Models for Phishing Email Detection. *Electronics* 13, 24 (2024), 4877. https://doi.org/10.3390/electronics13244877

[16] Radoslav Miltchev, Dimitar Rangelov, and Evgeni Genchev. 2024. *Phishing Validation Emails Dataset*. https://research.utwente.nl/en/datasets/phishing-validation-emails-dataset/ Collection of 2,000 labeled emails for phishing detection validation. Accessed: January 2025.

[17] Apurv Mittal, Daniel Engels, Harsha Kommanapalli, Ravi Sivaraman, and Taifur Chowdhury. 2022. Phishing Detection Using Natural Language Processing and Machine Learning. *SMU Data Science Review* 6, 2 (2022). Article 14. Available at: https://scholar.smu.edu/datasciencereview/vol6/iss2/14.

[18] NACHA. 2024. FBI's IC3 Finds Business Email Compromise Losses Exceeded $2.9 Billion in 2023. https://www.nacha.org/news/fbis-ic3-finds-almost-85-billion-lost-business-email-compromise-last-three-years Summarizes FBI IC3 multi-year analysis of BEC losses; reports nearly $8.5B lost over three years. Accessed: January 2025.

[19] Daniel Nahmias, Gal Engelberg, Dan Klein, and Asaf Shabtai. 2024. Prompted Contextual Vectors for Spear-Phishing Detection. arXiv:2402.08309.

[20] Netwrix. 2024. 60% of Educational Organizations Hit by Phishing Attacks Targeting Cloud Data. https://www.netwrix.com/ Survey of educational organizations reporting cloud-targeted phishing attacks. Accessed: January 2025.

[21] Kathryn Parsons, Agata McCormac, Malcolm Pattinson, Marcus Butavicius, and Cate Jerram. 2014. Determining Employee Awareness Using the Human Aspects of Information Security Questionnaire (HAIS-Q). *Computers & Security* 42 (2014), 165–176. https://doi.org/10.1016/j.cose.2013.12.003

[22] Md. Fazle Rabbi, Arifa Islam Champa, and Minhaz F. Zibran. 2024. Curated Datasets and Feature Analysis for Phishing Email Detection with Machine Learning. In *3rd IEEE International Conference on Computing and Machine Intelligence (ICMI)*. 1–7. https://doi.org/10.1109/ICMI60790.2024.10585821

[23] Steve Sheng et al. 2010. Who Falls for Phish? A Demographic Analysis of Phishing Susceptibility and Effectiveness of Interventions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 373–382. https://doi.org/10.1145/1753326.1753383

[24] SoSafe. 2023. *The Human Risk Review 2023*. Technical Report. SoSafe GmbH. https://www.sosafe.de/ Accessed: January 2025.

[25] StationX. 2024. Phishing Statistics and Facts. https://www.stationx.net/phishing-statistics/ Accessed: January 2025.

[26] UCI Machine Learning Repository. 2015. Phishing Websites Data Set. https://archive.ics.uci.edu/dataset/267/phishing-websites Feature-based dataset containing 11,055 website instances with 30 features for phishing classification. Accessed: January 2025.

[27] Arun Vishwanath, Tejaswini Herath, Rao Chen, Jingguo Wang, and H. Raghav Rao. 2011. Why Do People Get Phished? Testing Individual Differences in Phishing Vulnerability Within an Integrated, Information Processing Model. *Decision Support Systems* 51, 3 (2011), 576–586. https://doi.org/10.1016/j.dss.2011.03.002

[28] Han Zhang, Yong Shi, Ming Liu, Libo Chen, Songyang Wu, and Zhi Xue. 2025. A combined feature selection approach for malicious email detection based on a comprehensive email dataset. *Cybersecurity* 8 (February 2025). https://doi.org/10.1186/s42400-024-00309-6

[29] Jie Zhang, Haoyu Bu, Hui Wen, Yongji Liu, Haiqiang Fei, Rongrong Xi, Lun Li, Yun Yang, Hongsong Zhu, and Dan Meng. 2025. When LLMs meet cybersecurity: a systematic literature review. *Cybersecurity* 8 (February 2025). https://doi.org/10.1186/s42400-025-00361-w Systematic review of LLM applications in cybersecurity, documenting offensive uses including automated phishing generation and social engineering attacks.

Received 05 November 2025; Accepted 19 December 2025

Comparing the Performance of Traditional Machine Learning and Deep Learning Algorithms for Breast Cancer Survival Prediction

Navaneeth Ranjit
navirn2009@gmail.com
Private International English School
Mussafah, Abu Dhabi, UAE

Abstract

Breast cancer remains the most commonly diagnosed cancer among women worldwide, with disparities in outcomes influenced by geographic and socioeconomic factors. In this study, we address the challenge of predicting breast cancer survivability, specifically in scenarios with limited data.Using the METABRIC (Molecular Taxonomy of Breast Cancer International Consortium) dataset, which comprises clinical and genetic attributes from 1,980 primary breast cancer samples, we performed binary classification to predict patient survival outcomes. Clinical and genetic data were retrieved from the cBioPortal platform. We explored a range of machine learning (ML) and deep learning (DL) models, including 13 traditional ML algorithms, viz., k-Nearest Neighbors (KNN), logistic regression, decision trees, as well as 6 deep learning architectures including multilayer perceptrons (MLPs), TabNet, and DeepFM. In addition, we implemented a time-to-event analysis using a Cox proportional hazards pipeline with Elastic Net regularization and deep survival models such as DeepSurv and DeepHit to explore survival prediction beyond binary classification. These models were trained and tested on individual clinical and genetic attributes as well as their combination. We also evaluated the effectiveness of model stacking and voting classifiers to combine the strengths of multiple ML models in terms of accuracy, precision, recall, and F1 score. A soft voting classifier combining LightGBM, CatBoost, and AdaBoost, trained and tested on clinical attributes, provided the best performance, while deep learning models underperformed likely due to the limited dataset size. Overall, we observed that the models performed better when trained on clinical features rather than using genetic attributes, thereby suggesting that the clinical indicators remain more predictive of breast cancer survivability.

Keywords

Breast Cancer Survival, Machine Learning, Deep Learning, Ensemble Models, Tabular Data, METABRIC, Genetic Attributes, Clinical Attributes

ACM Reference Format:
Navaneeth Ranjit. 2026. Comparing the Performance of Traditional Machine Learning and Deep Learning Algorithms for Breast Cancer Survival Prediction. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 1)*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.5281/zenodo.18404485

1 Introduction

Breast cancer is an abnormal growth within the body and is an invasive tumor that occurs in the breast tissue and is characterised by the uncontrolled growth of abnormal cells, which can invade surrounding tissues and distant organs.

Breast cancer is the most diagnosed cancer affecting women globally, contributing to approximately 25% of all cancer cases and 15% of cancer-related mortality in women. In 2022, there were an estimated 2.3 million new cases, and by 2040, the incidence is projected to be 38% higher [6]. Additionally, breast cancer also imposes a substantial socioeconomic burden, as the estimated annual treatment cost exceeds $20,000 per patient in developed countries and also causes productivity loss due to the illness and mortality [3]. As a result, many research groups have looked to develop newer approaches to predicting patient survival outcomes as early and accurately as possible. Many studies have provided examples of applying machine learning (ML) models for predicting survival for breast cancer, with structured clinical tabular data [4].

Traditional models (Random Forests, Support Vector Machines, Decision Trees, Logistic Regression, and others) are typically known to be strong and interpretable methods to model survival. The models used standard clinical features such as tumor size, lymph node status, age, and hormone receptor status and for most; importance analyses and tumor variables would typically be top variables to aid in survival prediction.

The performance metrics of chosen models are evaluated using accuracy, sensitivity, specificity, area under the curve (AUC), and concordance index. The presence of missing data is typically addressed with different imputation methods. Despite these advancements, many widely-used ML models are not inherently designed to accommodate censored survival data, which may introduce bias and undermine the validity of clinical inferences. Deep learning (DL) techniques, although extensively used in imaging and genomic domains, remain underexplored in the context of tabular clinical data.

In this study, we address these gaps by applying advanced approaches that have shown strong potential in other domains but remain underutilized in survival prediction using tabular clinical data.

Advanced architectures such as TabNet and Deeptable, which have demonstrated strong potential for handling tabular data effectively, have yet to be fully leveraged for survival analysis in

IJSCAR VOL. 3, ISSUE 1

clinical settings. Specifically, we incorporate gradient boosting algorithms such as LightGBM and CatBoost due to their ability to handle complex feature interactions and missing values efficiently.

Furthermore, we employ ensemble methods, including voting and stacking classifiers, to leverage the strengths of multiple models and improve overall predictive performance and generalizability.

2 Materials and Methods

2.1 Dataset

The dataset we used in this study is provided by the Molecular Taxonomy of Breast Cancer International Consortium (METABRIC), which contains clinical and genomic data from 1,980 primary breast cancer samples. The data was collected by Professor Carlos Caldas of the Cambridge Research Institute and Professor Sam Aparicio of the British Columbia Cancer Centre, and was published in Nature Communications [5].The dataset has been used to train and evaluate models under three experimental conditions: using only clinical attributes, only genetic attributes, and a combination of both .Clinical and genetic data were accessed via cBioPortal[2]. The outcome variable of interest is overall survival, represented as a binary label (survived or not survived). The grouped attributes were used as input for a binary classification task to predict overall survival. The dataset was split to train and test the models with a 67%-33% training-test ratio. K-Fold Cross-Validation was used during training to ensure generalizability.The data was split into 5 folds for cross-validation. Model evaluation was based on standard classification metrics, including the F1-score, recall, precision, and overall accuracy.For interpretability, we additionally consider a fixed 5-year (60-month) survival horizon. Patients who died within 60 months were labeled as non-survivors, while patients alive beyond 60 months were labeled as survivors.

2.1.1 Clinical Attributes

Below given are the clinical attributes used with their respective datasets. They comprise of object, float, and integer data types. These clinical attributes are used to train our models under study. Clinical Attributes with their Type and description are explained below. In our first case we trained our models with the below clinical attributes.

2.1.2 Genetic Attributes

The genetic attributes in the dataset include mRNA expression levels and mutation data. It provides mRNA z-scores for 331 genes and mutation information for 175 genes. In our second case we have trained our models with genetic attributes.

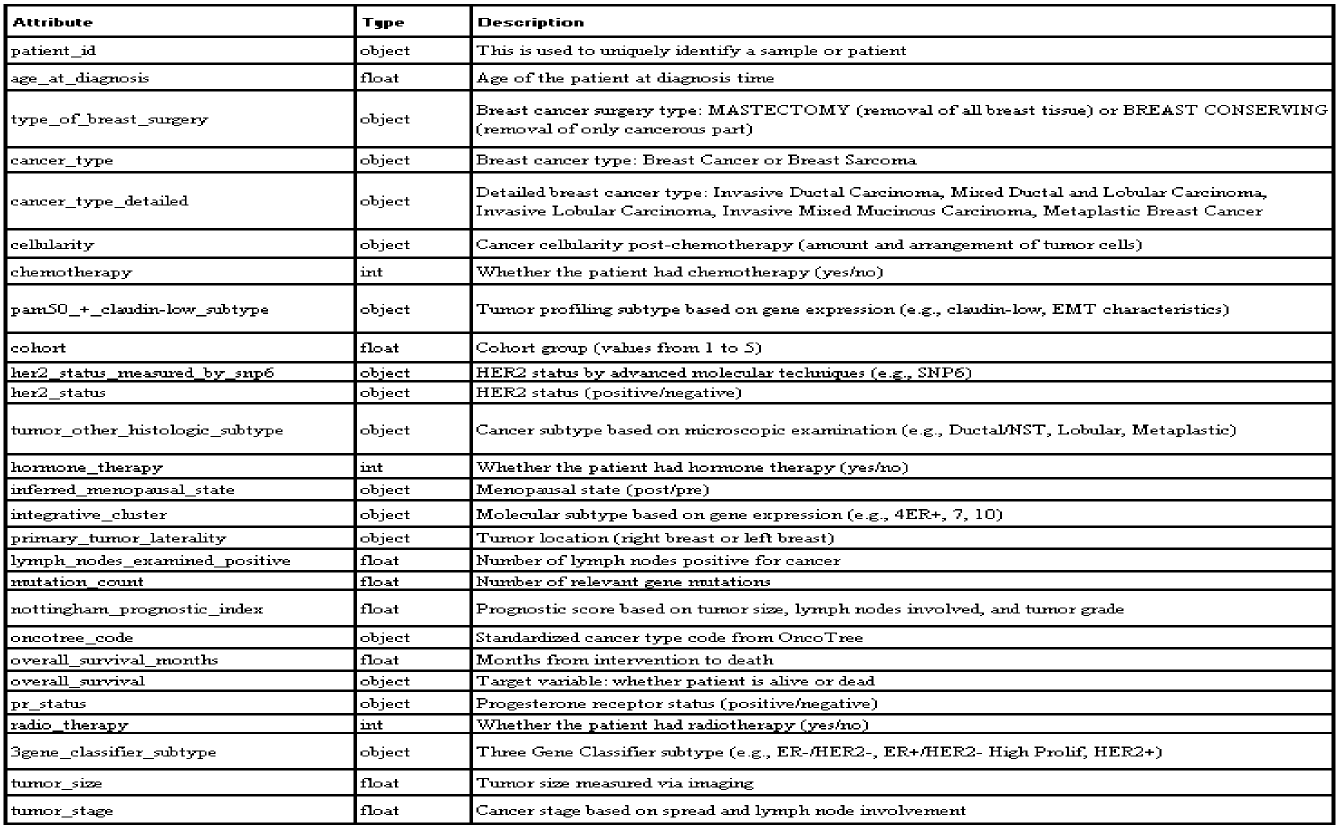
2.1.3 Combination

In our third case, we trained our models with both Genetic Attributes and Clinical Attributes.

2.1.4 Exploratory Data Analysis

Relationship between Clinical Attributes and Outcomes The distributional analysis of clinical variables shows us patterns and relationships relevant to the patient outcomes. Age at diagnosis demonstrates an approximately normal distribution, with a central tendency between 50 and 70 years. Patients who died tend to be older at the time of diagnosis, suggesting age may be associated with poorer prognosis. In contrast, variables such as `lymph_nodes_examined_positive`, `mutation_count`, and `tumor_size` exhibit strong right-skewness, characterized by a high concentration of lower values and the presence of significant outliers. For example, most patients had few or no positive lymph nodes, but those with elevated counts were more likely to be in the non-survivor group, underscoring the prognostic value of lymph node involvement. Similarly, although mutation_count is heavily skewed with a long tail, the distributions between survivors and non-survivors are largely overlapping, indicating that mutation burden alone may not be a strong discriminator of survival in this cohort. The Nottingham Prognostic Index (NPI) displays a roughly symmetrical, possibly multimodal distribution, with survivors showing greater density in lower index ranges and non-survivors skewed toward higher values. This observation aligns with its established clinical utility as a composite prognostic tool. Overall survival time (in months) also follows a right-skewed distribution, with survivors displaying a broader and more extended range of survival durations compared to those who died, who are concentrated in shorter survival intervals. Lastly, tumor size is markedly right-skewed, and larger tumors are more frequently observed among non-survivors, suggesting its role as a critical prognostic indicator. In this study, survival was primarily framed as a binary classification problem (alive vs. deceased at last follow-up) to enable a clear comparison between traditional machine learning and deep learning models on tabular data. While the METABRIC dataset supports time-to-event analysis with right-censoring, this simplified formulation was chosen to keep the modeling approach accessible and reproducible at the secondary-school level. As a result, all classification-based results should be interpreted as relative risk signals rather than clinical survival estimates.

Relationship Between Genetic Attributes and Outcomes The histogram presents the distribution of correlation coefficients between individual gene expression levels and overall survival. Here, survival labels were defined for each sample based on patient status (alive = 1, dead = 0), using the METABRIC clinical data to link genetic attributes with outcomes. The majority of correlations cluster around zero, indicating that most genes show weak linear association with survival outcomes. The distribution appears slightly skewed to the left, with a large amount of genes exhibiting negative correlations. This suggests a subset of genes whose increased expression may be associated with reduced survival, potentially pointing to oncogenic roles. Conversely, fewer genes display stronger positive correlations with survival, indicating possible tumor suppressive or protective effects. Survival was framed as a binary classification problem, with labels based solely on patient status (alive = 1, dead = 0), without considering overall survival time. Thus, no explicit prediction horizon (e.g., 5-year survival) was defined, and right-censoring information in the METABRIC dataset was not utilized.



Attribute	Type	Description
patient_id	object	This is used to uniquely identify a sample or patient
age_at_diagnosis	float	Age of the patient at diagnosis time
type_of_breast_surgery	object	Breast cancer surgery type: MASTECTOMY (removal of all breast tissue) or BREAST CONSERVING (removal of only cancerous part)
cancer_type	object	Breast cancer type: Breast Cancer or Breast Sarcoma
cancer_type_detailed	object	Detailed breast cancer type: Invasive Ductal Carcinoma, Mixed Ductal and Lobular Carcinoma, Invasive Lobular Carcinoma, Invasive Mixed Mucinous Carcinoma, Metaplastic Breast Cancer
cellularity	object	Cancer cellularity post-chemotherapy (amount and arrangement of tumor cells)
chemotherapy	int	Whether the patient had chemotherapy (yes/no)
pam50_+_claudin-low_subtype	object	Tumor profiling subtype based on gene expression (e.g., claudin-low, EMT characteristics)
cohort	float	Cohort group (values from 1 to 5)
her2_status_measured_by_snp6	object	HER2 status by advanced molecular techniques (e.g., SNP6)
her2_status	object	HER2 status (positive/negative)
tumor_other_histologic_subtype	object	Cancer subtype based on microscopic examination (e.g., Ductal/NST, Lobular, Metaplastic)
hormone_therapy	int	Whether the patient had hormone therapy (yes/no)
inferred_menopausal_state	object	Menopausal state (post/pre)
integrative_cluster	object	Molecular subtype based on gene expression (e.g., 4ER+, 7, 10)
primary_tumor_laterality	object	Tumor location (right breast or left breast)
lymph_nodes_examined_positive	float	Number of lymph nodes positive for cancer
mutation_count	float	Number of relevant gene mutations
nottingham_prognostic_index	float	Prognostic score based on tumor size, lymph nodes involved, and tumor grade
oncotree_code	object	Standardized cancer type code from OncoTree
overall_survival_months	float	Months from intervention to death
overall_survival	object	Target variable: whether patient is alive or dead
pr_status	object	Progesterone receptor status (positive/negative)
radio_therapy	int	Whether the patient had radiotherapy (yes/no)
3gene_classifier_subtype	object	Three Gene Classifier subtype (e.g., ER-/HER2-, ER+/HER2- High Prolif, HER2+)
tumor_size	float	Tumor size measured via imaging
tumor_stage	float	Cancer stage based on spread and lymph node involvement

Figure 1: Clinical Attributes

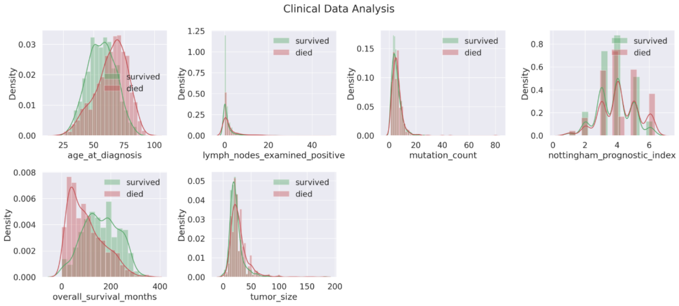


Figure 2: Distribution of clinical features showing key differences between survivors and non-survivors

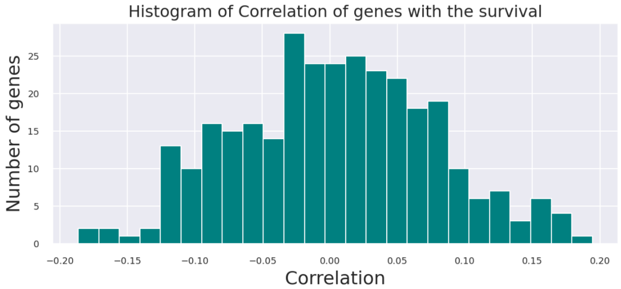


Figure 3: Distribution of correlation coefficients between gene expression levels and overall survival, showing that most genes have weak correlations, with a slight skew toward negative associations.

2.2 Machine Learning Models Used

We have used the below 10 machine learning models to train and evaluate with the datasets provided by METABRIC in 3 case studies with clinical, genomic and a combination of both.

K-Nearest Neighbors (KNN): This is a non-parametric approach that classifies data points based on the majority label of their k nearest neighbors in the feature space.

Decision Tree: The Decision tree model splits data based on feature values to make decisions. Features are indicated by internal nodes, and class labels or values are indicated by leaf nodes.

Logistic Regression: It is a linear classification model predicting the probability of a binary outcome through the logistic (sigmoid) function. This model performs well with linearly separable classes.

Random Forest: Random Forest is an ensemble model that builds multiple decision trees on different data and feature subsets. It averages their outputs to improve accuracy and avoid overfitting.

Extra Trees (Extremely Randomized Trees): Similar to Random Forest with yet more randomness while choosing feature splits. It builds uncorrelated trees faster and sometimes results in better generalization.

AdaBoost (Adaptive Boosting): It is a boosting ensemble method that trains a series of weak learners, with each subsequent model placing greater emphasis on the errors of the previous models.

XGBoost: An extremely effective gradient boosting library that uses regularization and aggressive tree building algorithms. efficiency and speed on structured data.

CatBoost: It is a gradient boosting framework that is designed to work with categorical features natively with minimal preprocessing. It uses ordered boosting and has overfitting resistance.

Light Gradient Boosting Machine (LightGBM): This model is a fast, distributed gradient boosting method that uses histogram-based algorithms and grows tree's leaf-wise which makes it ideal for large data and high performance.

Support Vector Classifier (SVC): A model that attempts to find the optimal hyperplane to separate classes. It can deal with both linear and non-linear classification using kernel functions.

2.3 Deep Learning Models Used

We have used the below 6 Deep learning models to train and evaluate with the datasets provided by METABRIC in 3 case studies with clinical, genomic and a combination of both.

AutoInt: It uses self-attention mechanisms to automatically learn feature interactions, suitable for modeling complex relationships in tabular data.

DeepFM: It combines Factorization Machines for low-order feature interactions with deep neural networks for high-order interactions, convenient in recommendation and structured data tasks.

xDeepFM: This model is an enhanced version of DeepFM that uses a Compressed Interaction Network (CIN) for explicitly modeling high-order feature interactions in an efficient way.

Wide & Deep: A hybrid model that includes a linear (wide) component for memorization and a deep neural network for generalization, providing a balance of both types of learning.

MLP (Multilayer Perceptron): A simple feedforward neural network composed of dense layers. Learns complex nonlinear relationships from input data through backpropagation.

TabNet: This is a tabular data deep learning model. It uses attention to select relevant features at each decision step and provides interpretability along with good performance.

2.4 Ensemble Method

We chose to study ensemble learning methods to improve the accuracy since ensemble methods have recently been praised with strong and universal findings supporting their utility and promise. Ensemble methods work by combining varying base learners to promote generalization, reduce overfitting, and use all data, including unbalanced or noisy data. Ensemble methods can also aid in balancing the bias-variance trade-off, which is an important consideration to make as we leave the safe confines of data, and into data that is likely to include a lot of noise and variability. In this case, as base learners only the three models with the best performance from every training attribute set were included. This ensured that the ensemble consisted of only good-quality models which contained relationship relevant to particular attributes subsets. This approach included only those models that had demonstrated strong performance, thereby maximizing the overall predictive power and variability in the pool of predictive models. As a first step, a Voting Classifier was used, specifically with soft voting. Soft voting computes the predicted probabilities of the models and uses this information to make selections based on a greater understanding of model predictions and class probabilities overall. This was particularly helpful in cases whereby models made conflicting predictions but used different logic in predicting classes [1]. For clinical attributes, LightGBM, CatBoost, and AdaBoost were used. For genetic attributes, CatBoost, K-Nearest Neighbors (KNN), and Random Forest were chosen. When combining both clinical and genetic attributes, LightGBM, CatBoost, and XGBoost were employed. Despite the positive outcome from soft voting, a Stacking Classifier is employed to enhance overall performance. Stacking differs from voting in that it uses a meta-learner to learn the best way to combine the outputs of the base models. The same models selected for Voting classifiers were used for the Stacking classifier.

3 Experiments and Results

3.1 Implementation and Training Details

KNN: The KNN model was tuned using GridSearchCV with 4-fold cross-validation, testing different values of `n_neighbors` ranging from 5 to 100 and using both `uniform` and `distance` weights.

Logistic Regression: Logistic Regression was tuned using GridSearchCV with 4-fold cross-validation, using both `l1` and `l2` penalties and 100 log-spaced values for the regularization parameter `C`. The random state was set to 42.

Decision Tree: The Decision Tree classifier was used with default parameters, and the random state was set to 42.

Random Forest: The Random Forest classifier was implemented using default settings, with the random state set to 42.

Extra Trees: The Extra Trees classifier was used with default parameters, and the random state was set to 42.

AdaBoost: The AdaBoost classifier was used with default parameters, and a random state of 42 was specified.

SVC (Support Vector Classifier): The SVC model was used with default parameters, which include an RBF kernel, `C = 1.0`, and `gamma = scale`. The random state was set to 42.

XGBoost : The XGBoost classifier was initialized with a learning rate of 0.1, 1000 estimators, a maximum depth of 5, and a minimum child weight of 1. The `gamma` value was set to `0`, `subsample` and `colsample_bytree` were both set to 0.8, and the objective was set to `binary:logistic`. The model used 4 threads, a scale position weight of 1, and a random seed of 27.

DeepTable Models: Four different DeepTable models were used: AutoInt, DeepFM, xDeepFM, and WideDeep. Except for changing the `nets` parameter to specify the model type, all other configuration settings were kept the same. These included automatic handling of categorical and discrete features, an embedding output dimension of 20, an embedding dropout rate of 0.3, and evaluation metrics set to AUC and accuracy.

CatBoost : The CatBoost classifier was configured with 500 iterations, a learning rate of 0.03, a maximum depth of 6, and an L2 leaf regularization value of 5. It used `Bernoulli` bootstrap type, a subsample rate of 0.8, and specified `cat_features` for handling categorical variables. The evaluation metric was set to AUC, with early stopping after 20 rounds and verbose output every 50 iterations.

LightGBM : The LightGBM model was configured with the objective set to `binary` and the evaluation metric set to `auc`. It used the `gbdt` boosting type, a learning rate of 0.03, 31 leaves, and a maximum depth of 6. The `feature_fraction` and `bagging_fraction` were both set to 0.8, with `bagging_freq` set to 5. L2 regularization (`reg_lambda`) was set to 5, and the random seed was fixed at 42. The verbosity level was set to -1 to suppress logs.

TabNet : The TabNet classifier was trained using a maximum of 100 epochs with early stopping enabled by setting `patience` to 5. The model was evaluated using the AUC metric.

MLP : The MLP model was trained for up to 100 epochs with a batch size of 32. Early stopping was applied via a callback, and training progress was shown with `verbose` output set to 1.

3.2 Evaluation Metrics

The performance of the models were carefully evaluated, since misclassification has the potential to strongly affect processes in clinical decision-making. Four major metrics were used in the current research: accuracy, precision, recall, and F1 score.

Accuracy: Accuracy is a metric that measures how often a machine learning model correctly predicts the outcome. It is particularly helpful when dealing with balanced classes and when the primary concern is the overall correctness of the model rather than its ability to predict a specific class. However, when classes are imbalanced, accuracy becomes less useful because it gives equal importance to predicting all categories, potentially hiding poor performance on the target class. In such cases, relying solely on accuracy can be misleading. While accuracy indicates the general rate of correct classification, it tends to conceal medical concerns related to class imbalance, i.e., the over representation of survivors (Class 1) in relation to non-survivors (Class 0).

Precision: High precision indicates that when the model predicts a patient has died, it is usually correct. This helps reduce overtreatment, and misallocation of clinical resources. However, prioritizing precision alone may cause the model to miss actual high-risk patients (false negatives), which is why it must be evaluated alongside recall.

Recall: Recall measures the proportion of patients who have passed away and were correctly classified by the model. They are classified as True Positives (TP), i.e. patients who died and were correctly predicted to not survive. False negatives (FN), are patients who died but were incorrectly predicted to survive.Recall helps to minimize False negatives, thereby minimizing the chance of falsely classifying patients.

F1-score: To balance the need to identify at-risk patients with the need to minimize the number of false alarms, we use the F1 score, i.e. the combination of precision and recall. This metric possesses great significance in our binary classification problem, especially when using stacking ensemble methods and combining features from high dimensional genetic data.

3.3 Machine Learning Model Outcomes

The following sections present model outcomes separately for clinical attributes, genetic attributes, and a combination of both. Each subsection includes model-specific performance and insights into which models performed best for that dataset type.

Table 1: Model Performance on Clinical Attributes

Model	Acc	Prec	Rec	F1
K Neighbors	0.66	0.65	0.66	0.65
Logistic Reg	0.77	0.76	0.77	0.76
Decision Tree	0.70	0.69	0.70	0.69
Random Forest	0.73	0.72	0.73	0.72
MLP	0.75	0.74	0.75	0.74
SVC	0.76	0.75	0.76	0.75

Table 2: Model Performance on Genetic Attributes

Model	Acc	Prec	Rec	F1
K Neighbors	0.78	0.77	0.78	0.77
Logistic Reg	0.63	0.62	0.63	0.62
Decision Tree	0.61	0.60	0.61	0.60
Random Forest	0.69	0.68	0.69	0.68
MLP	0.74	0.73	0.74	0.73
SVM	0.71	0.70	0.71	0.70

3.4 Deep Learning Model Outcomes

Below sections elaborate the results obtained by training and evaluating different Deep learning models on Clinical attributes, genetic attributes and the combination of both.

3.5 Ensemble Method Outcomes

3.5.1 Voting Classifier(Soft Voting)

Voting Classifier is the best performing model overall with an accuracy of 80% when trained on clinical attributes.

Table 3: Model Performance on Combined Attributes (Clinical + Genetic)

Model	Acc	Prec	Rec	F1
K Neighbors	0.67	0.66	0.67	0.66
Logistic Reg	0.75	0.74	0.75	0.74
Decision Tree	0.72	0.71	0.72	0.71
Random Forest	0.74	0.73	0.74	0.73
MLP	0.78	0.77	0.78	0.77
SVM	0.77	0.76	0.77	0.76

Table 4: Performance of Deep Learning Models on Clinical Attributes

Model	Prec(0)	Prec(1)	Rec(0)	Rec(1)	F1(0)	F1(1)	Acc
DeepFM	0.83	0.45	0.16	0.95	0.28	0.62	0.50
xDeepFM	0.89	0.49	0.28	0.95	0.42	0.65	0.56
AutoInt	0.58	0.00	1.00	0.00	0.73	0.00	0.58
WideDeep	0.82	0.45	0.15	0.95	0.26	0.61	0.49
MLP	0.78	0.68	0.75	0.72	0.77	0.70	0.74
TabNet	0.62	0.63	0.90	0.25	0.73	0.36	0.62

Table 5: Performance of Deep Learning Models on Genetic Attributes

Model	Prec(0)	Prec(1)	Rec(0)	Rec(1)	F1(0)	F1(1)	Acc
DeepFM	0.61	0.49	0.75	0.33	0.67	0.39	0.57
xDeepFM	0.59	0.46	0.74	0.29	0.66	0.36	0.55
AutoInt	0.58	0.00	1.00	0.00	0.73	0.00	0.58
WideDeep	0.62	0.56	0.84	0.28	0.71	0.37	0.60
MLP	0.67	0.59	0.75	0.50	0.71	0.54	0.65
TabNet	0.67	0.41	0.66	0.42	0.64	0.44	0.56

Table 6: Performance of Deep Learning Models on Combined Attributes (Clinical + Genetic)

Model	Prec(0)	Prec(1)	Rec(0)	Rec(1)	F1(0)	F1(1)	Acc
MLP	0.79	0.71	0.81	0.61	0.80	0.70	0.76
TabNet	0.67	0.54	0.35	0.76	0.46	0.56	0.51

Clinical		
Metric	**0**	**1**
Precision	0.82	0.77
F1 Score	0.83	0.76
Recall	0.84	0.74
Accuracy: 0.80		

Genetic		
Metric	**0**	**1**
Precision	0.68	0.67
F1 Score	0.75	0.54
Recall	0.84	0.45
Accuracy: 0.67		

Combination		
Metric	**0**	**1**
Precision	0.79	0.75
F1 Score	0.81	0.71
Recall	0.84	0.68
Accuracy: 0.72		

Table 7: Voting Classifier Results

Clinical		
Metric	**0**	**1**
Precision	0.82	0.76
F1 Score	0.82	0.75
Recall	0.83	0.74
Accuracy: 0.79		

Genetic		
Metric	**0**	**1**
Precision	0.68	0.68
F1 Score	0.75	0.53
Recall	0.58	0.44
Accuracy: 0.68		

Combination		
Metric	**0**	**1**
Precision	0.75	0.69
F1 Score	0.77	0.66
Recall	0.79	0.64
Accuracy: 0.72		

Table 8: Stacking Classifier Results

3.5.2 Stacking Classifier

This ensemble performed similarly to voting classifier with an accuracy of 79% on clinical attributes.

4 Discussion

4.1 ROC Curve

ROC curves trace the comparisons of different traditional machine learning models. The x-axis is plotted with False Positive Rate, meaning how often patients who did not survive are incorrectly predicted as survivors. The y-axis plots the True Positive Rate, meaning how often actual survivors are correctly predicted. The area under the curve (AUC) measures how well the model can distinguish between classes (0 and 1). A larger area (closer to 1) means the model is more accurate, while an area close to 0.5 suggests random guessing.

4.1.1 ROC Curve for Traditional Machine Learning Models

Logistic Regression classifier performed the best among traditional models when trained on clinical attributes with an accuracy of 77%. It also performed the best on the combination of both attributes with an accuracy of 75%. However, KNN model performed the best with an accuracy of 65% when trained with genetic attributes.

4.1.2 ROC Curve for Deep Learning Models

Here the MLP model has a performance of 72% accuracy when trained on clinical attributes whereas TabNET performs much worse with an accuracy of 57%. Both models performed similarly when trained on genetic attributes where MLP classifier provided an accuracy of 66% and TabNET gave an accuracy of 62%. MLP classifier has an accuracy of 75% when trained on the combination of both attributes whereas TabNET had an overall accuracy of 56%.

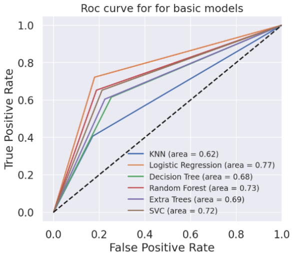


Figure 4: ROC Curve Traditional Models on Clinical Attributes

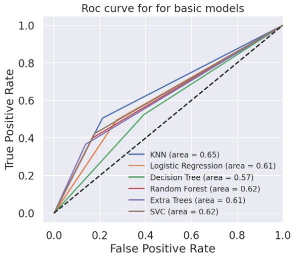


Figure 5: ROC Curve Traditional Models on Genetic Attributes

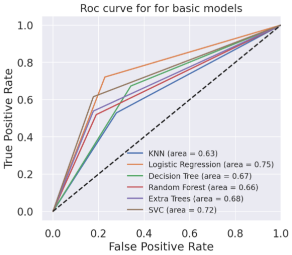


Figure 6: ROC Curve Traditional Models on Combination of Both Attributes

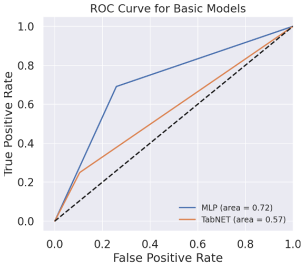


Figure 7: ROC Curve for Deep Learning Models on Clinical Attributes

4.1.3 ROC Curve for Ensemble Models

Here the Voting classifier with soft voting had an accuracy of 80% on clinical attributes and the stacker classifier had an accuracy of 79%. Both models performed similarly when trained on genetic attributes with the same accuracy of 64%. The stacker classifier and the soft voting classifier had an accuracy of 72% when trained on the combination of attributes.

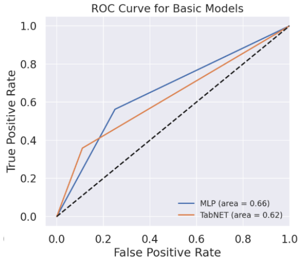


Figure 8: ROC Curve for Deep Learning Models on Genetic Attributes

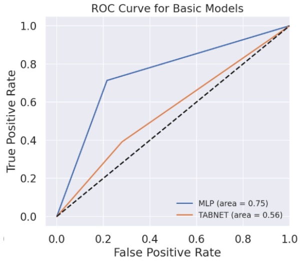


Figure 9: ROC Curve for Deep Learning Models on Combination of Both Attributes

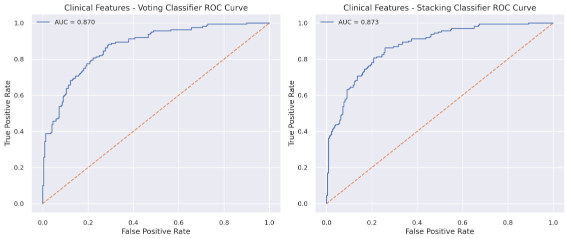


Figure 10: ROC Curve for Ensemble Models on Clinical Attributes

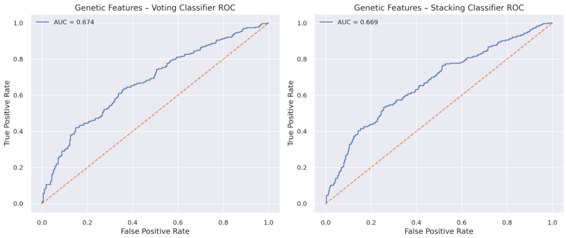


Figure 11: ROC Curve for Ensemble Models on Genetic Attributes

4.2 Comparative Performance

When trained on clinical attributes, a soft voting classifier ensemble with CatBoost, AdaBoost, and LightGBM achieved the highest accuracy of 80%. When the models were trained on the genetic

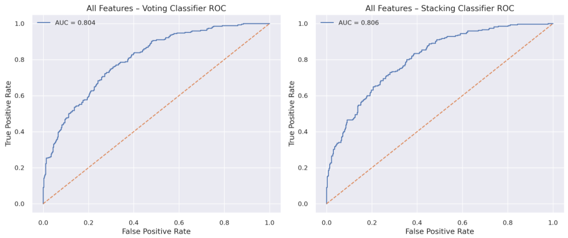


Figure 12: ROC Curve for Ensemble Models on Combination of Both Attributes

attributes, the best performance of 68% accuracy was obtained using a stacking classifier with CatBoost, K-Nearest Neighbors (KNN), and Random Forest. When combining clinical and genetic features, a soft voting classifier with CatBoost, XGBoost, and LightGBM achieved 77% accuracy.

The high performance of gradient boost models could be due to their ability to handle mixed data types and manage missing values. These models are also less likely to overfit on small to medium-sized datasets. Additionally, soft voting classifiers take the average of the predictions and account the confidence of each model. This helps to reduce the impact of a certain model's mistake.

In contrast deep learning models, mainly Deeptable models like AutoInt, DeepFM, AutoInt, xDeepFM and WideDeepand Tabnet, failed to perform adequately. These models either crashed or yielded poor results when trained on combined datasets. Only the MLP Classifier achieved moderate accuracy, but still underperformed compared to the other ensembles. Insufficient data size(such as 1,980 samples in our study), high feature dimensionality (especially in the genetic data), lack of spatial structure are some probable reasons why these models underperformed. Deep models typically require large datasets and are prone to overfit in low-sample settings. The strong performance of CatBoost, LightGBM, and XGBoost suggests that gradient boosting models are better suited for tabular limited data. These models natively support categorical variables, feature importance evaluation, and efficient training on small datasets, making them highly effective in binary classification tasks using limited data.

5 Limitations of Binary Classification and Transition to Time-to-Event Analysis

The above study using binary classification has significant limitations when applied to clinical datasets such as METABRIC, which contain long follow-up times and right-censoring.

Binary classification discards two key aspects of survival data:

1. **Censoring:** Patients who remain alive at last follow-up are treated as if their outcome were fully observed, leading to biased estimates.
2. **Timing of events:** Information on when an event occurs is lost, reducing the clinical interpretability of the predictions.

These issues mean that classification metrics such as accuracy or F1-score can be misleading for risk prediction, as they fail to capture time-to-event dynamics that are crucial.

To address this, the task was reformulated as a censored time-to-event analysis. This allowed full utilization of the survival information in the dataset, estimation of individual risk over time, and evaluation of models with survival-aware metrics such as the concordance index (C-index), time-dependent AUC, Brier score, and calibration curves.

5.1 Genetic Data Preprocessing

The genetic pipeline first processed high-dimensional genomic features, including mRNA expression and mutation data, using the Elastic Net–regularized Cox model to identify stable, survival-associated signatures. This approach was chosen for its ability to handle multicollinearity among correlated genetic features and to enforce sparsity, enabling the selection of the most informative genes. These selected genetic features were then fused with clinical variables such as age, tumor stage, treatment history, and overall survival time (in months).

The survival labels were derived from the overall survival time (in months) and the vital status recorded in the dataset. Patients who were alive at the last follow-up were treated as censored, while those who had an event (death) were marked as uncensored.

5.1.1 Traditional Models Used for Survival Analysis

Regularized Cox Proportional Hazards (CoxPH) Model: A semi-parametric approach that models the effect of covariates on survival, with regularization to handle high-dimensional data.

Random Survival Forests (RSF): An ensemble tree-based method that non-parametrically estimates survival probabilities while automatically handling non-linear interactions and high-dimensional data.

5.1.2 Deep Learning Models Used for Survival Analysis

DeepSurv: A deep learning extension of the Cox model that captures complex, non-linear relationships between covariates and survival risk.

DeepHit: A neural network model that directly estimates the probability distribution of survival times, allowing flexible handling of competing risks and time-dependent hazards.

5.2 Results

5.2.1 Regularized CoxPHFitter

Reports the C-index, time-dependent AUC, and Brier scores at 12, 36, and 60 months for Clinical, Genetic, and combined feature sets. Training C-index is provided for Genetic and Both attributes. All metrics are calculated on the test set unless otherwise specified.

5.2.2 DeepSurv

Reports the C-index and Brier scores at 12, 36, and 60 months for Clinical, Genetic, and combined feature sets. All metrics are calculated on the test set.

5.2.3 DeepHit

Reports the C-index and Brier scores at 12, 36, and 60 months for Clinical, Genetic, and combined feature sets. All metrics are calculated on the test set.

Table 9: Performance of Regularized CoxPH Model Across Different Feature Sets

Metric	Clinical	Genetic	Both
C-index (Train)	–	0.7404	0.7435
C-index (Test)	0.8131	0.7227	0.7321
AUC 12 mo (Test)	0.8896	0.7190	0.6957
AUC 36 mo (Test)	0.8637	0.7318	0.7099
AUC 60 mo (Test)	0.8743	0.7238	0.7225
Brier 12 mo (Test)	0.0082	0.0091	0.0092
Brier 36 mo (Test)	0.0172	0.0158	0.0159
Brier 60 mo (Test)	0.0272	0.0247	0.0248

Table 10: DeepSurv Model Performance Across Different Feature Sets

Metric	Clinical	Genetic	Both
C-index	0.7159	0.6375	0.6375
Brier 12 mo	0.9498	0.9959	0.9959
Brier 36 mo	0.9796	0.9816	0.9816
Brier 60 mo	0.9735	0.9685	0.9685

Table 11: DeepHit Model Performance Across Different Feature Sets

Metric	Clinical	Genetic	Both
C-index	0.3988	0.3207	0.3333
Brier 12 mo	0.0085	0.0084	0.0085
Brier 36 mo	0.0235	0.0233	0.0231
Brier 60 mo	0.0426	0.0423	0.0421

5.2.4 Random Survival Forest (RSF)

Reports the C-index, Brier scores, AUC at multiple time points, and mean time-dependent AUC for Clinical, Genetic, and combined feature sets. All metrics are calculated on the test set.

Table 12: RSF Model Performance Across Different Feature Sets

Metric	Clinical	Genetic	Both
C-index	0.8120	0.6876	0.7944
Brier 12 mo	0.0074	0.0079	0.0077
Brier 36 mo	0.0173	0.0189	0.0184
Brier 60 mo	0.0263	0.0281	0.0272
AUC 12 mo	0.8770	0.8183	0.9384
AUC 24 mo	0.8289	0.7257	0.9021
AUC 36 mo	0.8356	0.7320	0.8993
AUC 48 mo	0.8459	0.7353	0.8864
AUC 60 mo	0.8729	0.7250	0.8900
Mean time-dependent AUC	0.8520	0.7473	0.9033

5.3 Calibration Plots

The following calibration plots illustrate the agreement between predicted survival probabilities and observed outcomes for each model across the three feature sets: Clinical, Genetic, and Combined. A perfectly calibrated model would align closely with the diagonal reference line.The calibration plots of the best performing deep learning model and machine learning model from the binary classification task(CatBoost and TabNet) are shown here.

Table 13: C-index of top classifier and CoxPH model on clinical, genetic, and combined features

Model	Clinical	Genetic	Both
Regularized CoxPh	0.8131	0.7227	0.7321
CatBoost	0.5209	0.3653	0.3737

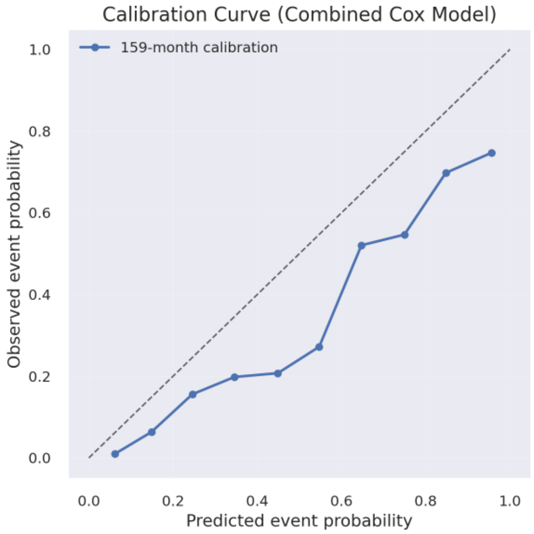


Figure 13: Calibration plot for Regularized CoxPHFitter using the combined clinical and genetic feature set.

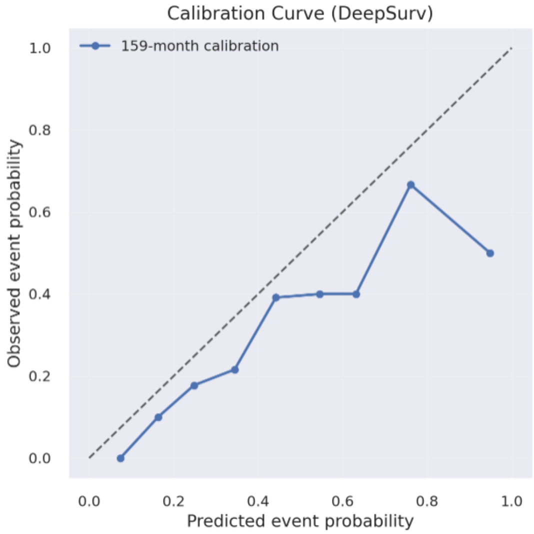


Figure 14: Calibration plot for DeepSurv using the combined clinical and genetic feature set.

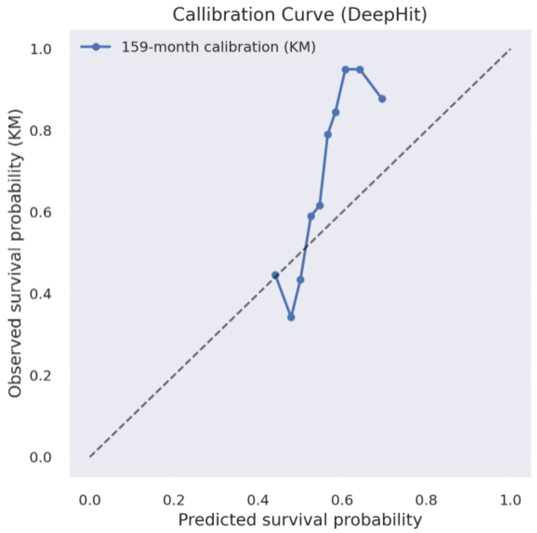


Figure 15: Calibration plot for DeepHit using the combined clinical and genetic feature set.

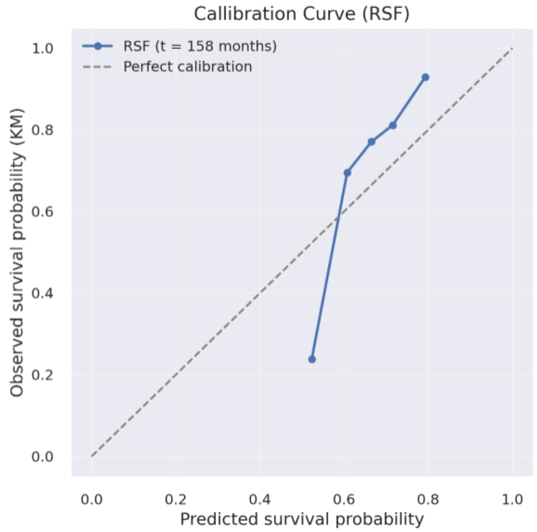


Figure 16: Calibration plot for DeepHit using the combined clinical and genetic feature set.

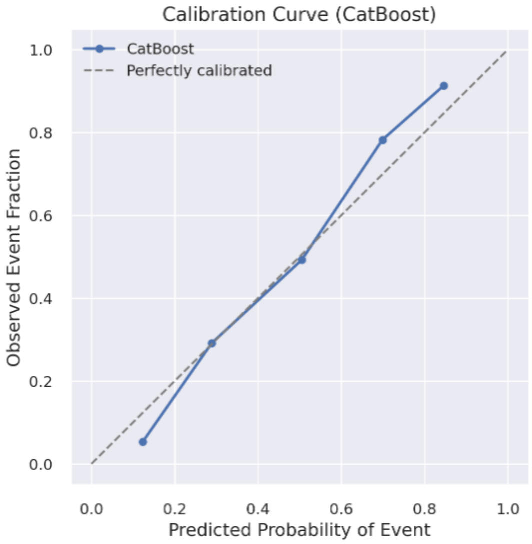


Figure 17: Calibration plot for CatBoost using the combined clinical and genetic feature set.

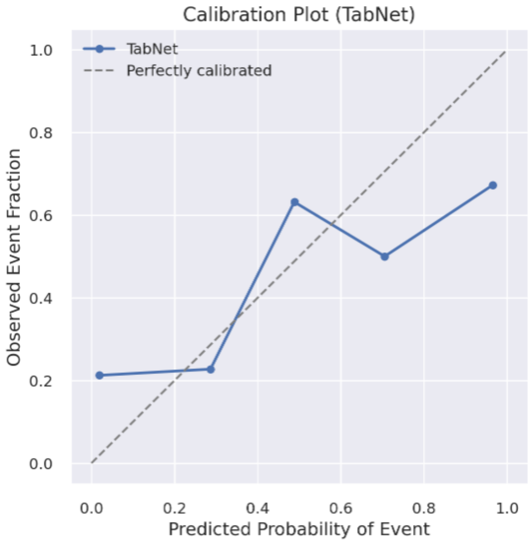


Figure 18: Calibration plot for Tabnet using the combined clinical and genetic feature set.

6 Conclusions

This study demonstrates that ensemble methods based on gradient boosting models provide satisfactory accuracy in predicting breast cancer survival from structured clinical and genomics attributes. The best performance (80% accuracy) was achieved in a soft voting ensemble on clinical features. The genetic attributes even at its high dimensionality turned out to be reasonably well predicted (68% accuracy) with stacking classifiers. With the combined data, the gradient boosting voting ensemble obtained an accuracy of 77%. Deep learning-based methods however, e.g., DeepFM and AutoInt, did not perform well showing an accuracy of 67% and were not effective for predicting due to data size and complexity.

In addition to these classification models, time-to-event models were evaluated to account for survival time and censoring. The Cox proportional hazards model on clinical features achieved a C-index of 0.813, AUCs of 0.890, 0.864, and 0.874 at 12, 36, and 60 months, and Brier scores of 0.008, 0.017, and 0.027, indicating strong discrimination and excellent calibration. For combined clinical and genetic features, DeepSurv and DeepHit achieved C-index values around 0.835, with similarly high AUCs, demonstrating improved

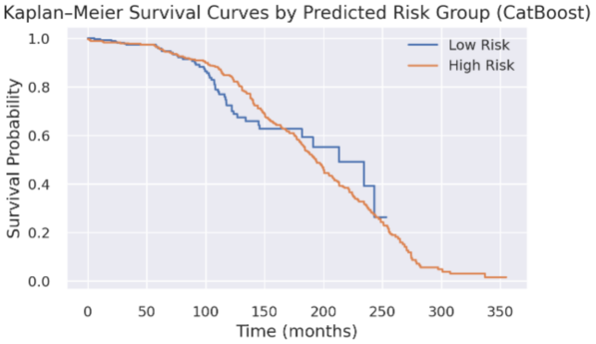


Figure 19: Kaplan–Meier survival curves for patients stratified into high-risk and low-risk groups based on the median predicted risk from the CatBoost model.

risk stratification and capturing non-linear interactions between features. These results highlight the added value of survival analysis models in providing time-dependent predictions beyond binary classification.

Future work could focus on selecting the most important features from omics data and reducing its complexity, and creating hybrid models that combine deep learning with tree-based methods. Testing on larger datasets could help improve how well the models work in real-world settings.This study prioritizes methodological accessibility over clinical deployment. While more advanced survival models and evaluation techniques may further improve predictive performance, they were outside the intended scope of this secondary-school research project.

7 Acknowledgement

I would like to thank my mentor, Pramit Saha for his support and collaboration throughout this work. His guidance, feedback, and helpful suggestions were invaluable in shaping the paper and bringing it to completion.

References

[1] Baeldung: Hard vs. soft voting classifiers (2024), https://www.baeldung.com/cs/hard-vs-soft-voting-classifiers, retrieved from Baeldung

[2] cBioPortal for Cancer Genomics: Metabric dataset (2023), https://www.cbioportal.org/, retrieved from cBioPortal

[3] International Agency for Research on Cancer (IARC): Global cancer observatory (gco): Cancer today (2024), accessed via IARC Global Cancer Observatory website; data from https://gco.iarc.fr

[4] Javanmard, Z., Zarean Shahraki, S., Safari, K., Omidi, A., et al.: Artificial intelligence in breast cancer survival prediction: A comprehensive systematic review and meta-analysis. Frontiers in Oncology **14** (2025). https://doi.org/10.3389/fonc.2024.1420328, https://doi.org/10.3389/fonc.2024.1420328

[5] Pereira, B., et al.: The metabric dataset: A comprehensive resource for cancer survival analysis. Nature Communications **7**, 11551 (2016). https://doi.org/10.1038/ncomms11551, https://www.nature.com/articles/ncomms11551

[6] World Health Organization: Breast cancer statistics (2022), https://www.who.int/news-room/fact-sheets/detail/breast-cancer, wHO fact sheet; contains global statistics, risk factors, symptoms, and treatment.

Received 05 September 2025; Accepted 28 January 2026

Machine Learning Framework for Phishing Detection through Email using Imbalanced Data

Mher Mkrtumyan
mmkrtumyan29@gmail.com
Horizon Academic Research Program
Los Angeles, California, USA

ABSTRACT

Phishing emails remain a persistent cybersecurity threat, as attackers continue to employ increasingly sophisticated techniques to deceive users into disclosing sensitive information. Detection is particularly challenging in real-world environments, where legitimate emails vastly outnumber malicious ones. This study presents a dual-layer machine learning framework for phishing detection that independently analyzes sender metadata and email body content. The sender layer evaluates structural characteristics of email addresses, while the content layer extracts linguistic and statistical features from email text. Each layer produces a probability score representing the likelihood of phishing; these are subsequently integrated using a meta-classification model to generate a final decision. The framework is evaluated on a large, real-world dataset containing over 500,000 emails with a highly imbalanced class distribution. Experimental results demonstrate that the proposed approach provides robust and reliable performance under realistic conditions, highlighting the effectiveness of integrating multiple analytical perspectives for practical phishing detection.

KEYWORDS

Phishing Detection, Machine Learning, Dual-Layer Framework, Email Security, Imbalanced Data, Sender Metadata, Email Content Analysis, Random Forest, XGBoost, LightGBM, ROC Curve, Logistic Regression, Cybersecurity, Dataset Preprocessing

ACM Reference Format:
Mher Mkrtumyan. 2026. Machine Learning Framework for Phishing Detection through Email using Imbalanced Data. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 1)*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.5281/zenodo.18435064

1 INTRODUCTION

Phishing is a widespread form of cybercrime in which attackers attempt to deceive individuals into revealing sensitive information, such as passwords, financial details, or personal data. These attacks are most commonly carried out through fraudulent emails that impersonate trusted organizations or individuals [22]. As digital communication becomes increasingly central to daily life, phishing attacks have grown both in frequency and sophistication, posing a serious threat to individuals, institutions, and businesses.

Traditional phishing detection methods often rely on predefined rules or simple heuristics, such as blacklisted domains or keyword matching. While effective against basic attacks, these approaches struggle to detect modern phishing emails that closely resemble legitimate messages. Machine learning–based detection systems have therefore gained popularity due to their ability to identify complex patterns within large datasets. However, many existing systems focus on a single aspect of an email—either sender information or content—limiting their effectiveness in real-world scenarios.

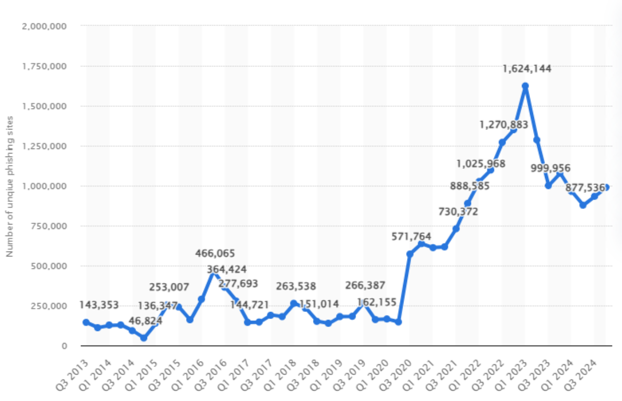


Figure 1: Number of phishing attacks detected worldwide from 3rd quarter 2013 to 4th quarter 2024 [14].

Another major challenge in phishing detection is data imbalance. In real-world email traffic, phishing messages represent only a small fraction of total emails, yet they have disproportionately severe consequences. Models trained on artificially balanced datasets may report high accuracy but often fail to generalize to realistic environments. This study addresses these challenges by proposing a dual-layer machine learning framework designed specifically for large, imbalanced datasets.

1.1 Research Question

This research investigates the following question:

Can a dual-layer machine learning framework that independently analyzes sender metadata and email content improve phishing detection performance on large, real-world, imbalanced email datasets?

By separating sender-based and content-based analysis and integrating their outputs through a meta-model, this study aims to

IJSCAR VOL. 3, ISSUE 1, February 2026,

https://doi.org/10.5281/zenodo.18435064

evaluate whether a multi-perspective approach provides greater robustness and practical effectiveness than traditional single-layer systems.

1.2 Challenges and Recent Developments

Phishing detection presents several technical challenges due to the evolving nature of attack strategies. Attackers frequently manipulate both the visible content of emails and the underlying sender information to evade detection systems. Content-based detection models may fail when phishing emails contain minimal text, ambiguous language, or carefully crafted wording. Conversely, sender-based approaches may struggle when attackers use compromised accounts or realistic-looking domains.

As a result, relying exclusively on either content or sender metadata is often insufficient. Modern phishing emails are designed to exploit the weaknesses of single-layer detection systems by appearing legitimate in one dimension while remaining malicious in another.

Recent research has increasingly favored hybrid detection frameworks that incorporate multiple feature types, including textual, structural, and behavioral characteristics. Ensemble learning and advanced machine learning models have demonstrated improved detection performance by capturing complementary signals across different feature domains. Despite these advancements, many studies still emphasize a primary feature modality, with limited integration of sender-specific information into content-focused models.

This study builds upon recent developments by explicitly separating sender and content analysis into two independent layers, allowing each to contribute uniquely to the final classification decision.

1.3 Overview of the Proposed Approach

To address the limitations of existing phishing detection systems, this study proposes a dual-layer machine learning framework. Rather than treating phishing detection as a single binary decision problem, the framework evaluates emails through two specialized analytical layers.

The first layer focuses on sender metadata, extracting structural features from email addresses to assess their legitimacy. The second layer analyzes the content of the email body using linguistic and statistical features. Each layer independently produces a probability score representing the likelihood that an email is phishing.

These scores are subsequently combined using a meta-classification model, which learns how to weigh sender and content information to produce a final prediction. This design enables the system to identify phishing attempts that might evade detection when only one type of information is considered.

2 RELATED WORK

Phishing detection has been extensively studied within the fields of cybersecurity and machine learning. Early detection systems primarily relied on rule-based methods, such as blacklists and keyword filtering. While straightforward to implement, these approaches proved vulnerable to evasion and required frequent manual updates [22].

More recent research has demonstrated the effectiveness of machine learning techniques for phishing detection [1] [2] [17]. Studies have reported high classification accuracy using algorithms such as Random Forest, Support Vector Machines, and gradient boosting methods. However, many of these studies evaluated their models on relatively small or artificially balanced datasets, limiting their applicability to real-world conditions.

Some researchers have focused on sender-centric detection methods, analyzing email headers and address structures to identify suspicious patterns [18]. Others have concentrated on content-based analysis, leveraging linguistic features to distinguish phishing emails from legitimate ones [21]. Hybrid approaches that combine multiple feature types have shown improved performance, particularly when ensemble learning techniques are applied [12] [16].

Recent advancements have also explored the use of feature selection and ensemble learning [6] [17] as well as transformer-based language models for phishing detection [7]. While these methods offer strong performance, they often require significant computational resources, which may limit their practicality in certain deployment scenarios.

The body of existing literature suggests that combining multiple analytical perspectives yields greater robustness. This study contributes to this area by proposing a structured dual-layer framework that integrates sender metadata and email content analysis while maintaining scalability and interpretability.

3 BACKGROUND

3.1 Phishing Attack Mechanism

Phishing is a social engineering attack in which attackers send deceptive emails that appear to originate from legitimate sources, such as financial institutions, technology companies, or employers. These emails often create a sense of urgency or authority to prompt immediate action.

Typically, a phishing email contains a link or attachment that directs the recipient to a fraudulent website designed to resemble a legitimate one. Once the user enters sensitive information, such as login credentials, the attacker captures and misuses this data. Because phishing attacks rely heavily on deception rather than malware, they can be difficult to detect using traditional security mechanisms.

3.2 Detection Challenges

Detecting phishing emails is increasingly difficult due to advances in attack sophistication. Many phishing emails now use realistic sender addresses, professional formatting, and carefully worded messages. Malicious links may be hidden behind legitimate-looking text, and attackers often adapt their strategies to bypass known detection rules.

Additionally, the imbalance between legitimate and phishing emails complicates machine learning training. Models may become biased toward predicting the majority class, resulting in high overall accuracy but poor phishing detection performance. These challenges highlight the need for detection systems that are both robust to evasion and effective under realistic data conditions.

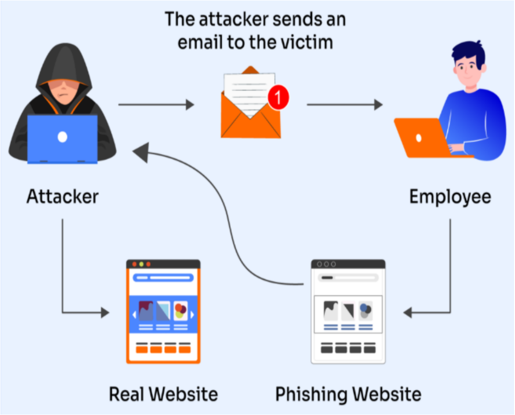
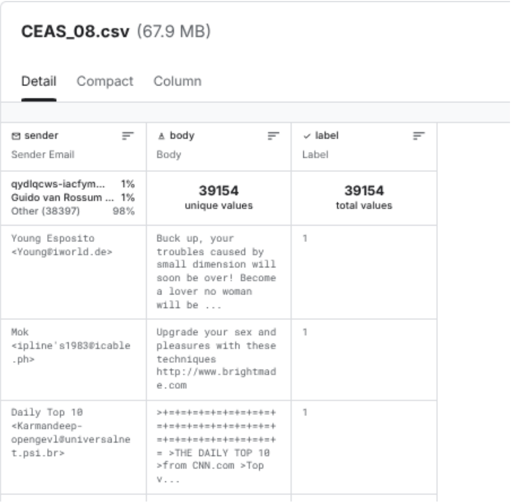


Figure 2: Phishing attack steps: from fake email creation to victim data theft [20].

4 DATASETS

To evaluate the proposed dual-layer framework, three publicly available email datasets were utilized. Rather than training separate models on each dataset, they were merged to create a comprehensive dataset representative of real-world email traffic.

The first dataset contains 39,154 phishing emails compiled from public sources [3]. The second dataset includes 38,810 labeled phishing and legitimate emails [4]. The third and largest dataset consists of 447,417 emails from the Enron Fraud Email Dataset [15]. These datasets were selected to provide diversity in email structure, content, and sender information.



CEAS_08.csv (67.9 MB)

sender Sender Email	body Body	label Label
qydlqcws-iacfym... 1% Guido van Rossum ... 1% Other (38397) 98%	39154 unique values	39154 total values
Young Esposito <Young@iworld.de>	Buck up, your troubles caused by small dimension will soon be over! Become a lover no woman will be ...	1
Mok <ipline's1983@icable.ph>	Upgrade your sex and pleasures with these techniques http://www.brightmade.com	1
Daily Top 10 <Karmandeep-opengevl@universalnet.psi.br>	>+=+ = >THE DAILY TOP 10 >from CNN.com >Top v...	1

Figure 3: Example of one of the datasets that our model trains on.

After preprocessing and integration, the final dataset comprised 524,735 emails. Of these, 479,710 (91.42%) were legitimate and 45.025 (8.6%) were phishing, closely mirroring real-world email distributions. This highly imbalanced dataset presents a challenging but realistic environment for evaluating phishing detection performance and ensures that results reflect practical deployment conditions.

5 SYSTEM IMPLEMENTATION

5.1 System Architecture

The proposed phishing detection system is implemented as a dual-layer machine learning framework designed to address the limitations of traditional single-layer approaches. Rather than relying on a single set of features or rules, the system independently analyzes two distinct components of an email: sender metadata and email body content.

The first layer focuses on sender-related information extracted from the email address. Structural features such as domain characteristics, username patterns, and character composition are used to train a machine learning model. This model outputs a continuous probability score between 0 (legitimate) and 1 (malicious), representing the likelihood that an email is phishing based solely on sender metadata.

The second layer analyzes the email body by extracting linguistic and statistical features from the text. Multiple machine learning models are trained on these features, and their outputs are aggregated into a single content-based probability score.

The final stage of the system integrates the scores produced by both layers to generate a comprehensive phishing prediction. This modular architecture enables the system to detect phishing attempts that may evade detection when only one type of information is considered.

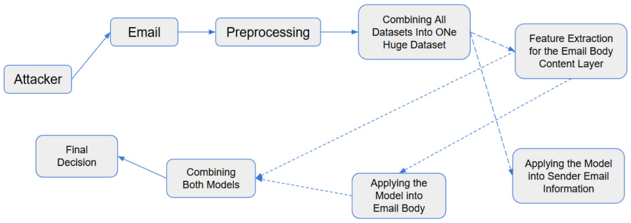

Attacker
Email
Preprocessing
Combining All Datasets Into ONe Huge Dataset
Feature Extraction for the Email Body Content Layer
Applying the Model into Sender Email Information
Applying the Model into Email Body
Combining Both Models
Final Decision

Figure 4: System Architecture of Our Dual-Layer Detection System

5.2 Data Preprocessing

Data preprocessing is a critical step in preparing raw email data for machine learning analysis. Email content often contains inconsistent formatting, hyperlinks, email addresses, numerical values, and special characters that can interfere with feature extraction.

To ensure consistency, all text is converted to lowercase. Hyperlinks and email addresses are removed, as they are highly variable and do not contribute meaningfully to textual pattern recognition. Non-alphabetic characters are eliminated, leaving only letters and spaces. Excess whitespace is removed to maintain uniform token separation.

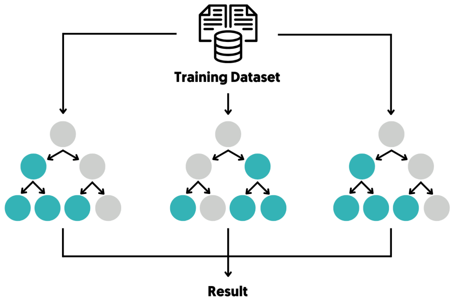


Figure 5: Structure of how Random Forest, XGBoost, and LightGBM work [10].

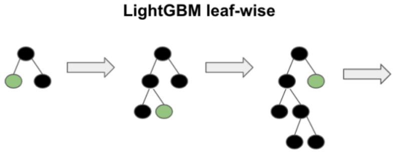


Figure 6: Structure of how LightGBM works [5].

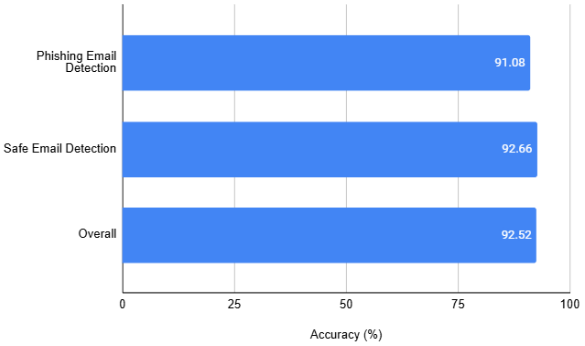


Figure 7: The final Detection Results.

In cases where words are improperly concatenated, a word segmentation tool is applied to restore meaningful word boundaries. After cleaning, the processed text is reconstructed into a single structured string and saved to a separate file. This preprocessing pipeline ensures that the data is standardized and suitable for subsequent feature extraction and model training.

5.3 Dataset Integration

Following preprocessing, the cleaned datasets are merged into a single unified dataset. Combining multiple datasets increases the diversity and volume of training data, allowing the models to learn more generalized patterns.

Each dataset is first standardized to ensure consistent formatting. Sender information is reconstructed where necessary by combining username and domain components, and email addresses embedded within text are extracted. Column names are normalized across datasets to maintain a consistent schema, including sender information, cleaned email content, and phishing labels.

Once standardized, the datasets are concatenated into a single comprehensive dataset. The resulting dataset is stored for use in training both the sender-based and content-based detection layers.

5.4 Email Body Feature Extraction

To enable machine learning models to analyze email content, textual data must be transformed into numerical features. A set of statistical and linguistic features is extracted from each email body.

These features include total word count, number of unique words, and average word length. Lexical diversity is calculated as the ratio of unique words to total words, providing insight into writing style variability. Word repetition metrics are also computed, as phishing emails often rely on repeated keywords.

Additionally, the system identifies and counts commonly used phishing-related terms, such as “verify,“ “account,“ and “urgent.“ The proportion of these keywords relative to the total word count is used as an indicator of suspicious content. All extracted features are combined into a numerical feature vector and saved for model training and evaluation.

5.5 Sender-Based Model Implementation

The sender-based detection layer is implemented using a Random Forest classifier, an ensemble learning method that aggregates predictions from multiple decision trees to improve accuracy and robustness [8].

Emails with missing sender information or labels are excluded from training. Each sender address is divided into username and domain components, which are normalized to lowercase. A variety of structural features are extracted, including length measurements, digit and special character counts, character repetition patterns, entropy values, and the presence of phishing-related keywords.

The dataset is split into training (70%) and testing (30%) subsets to evaluate model performance. The trained Random Forest model outputs both a binary classification and a probability score representing the likelihood that an email is phishing based on sender metadata. These outputs are stored for later integration with the content-based layer.

5.6 Content-Based Model Implementation

The content-based detection layer utilizes multiple machine learning algorithms, including Random Forest, XGBoost, and LightGBM. These models are selected due to their strong performance on structured feature sets and large-scale datasets [9] [13].

Using the extracted email body features, the dataset is divided into training and testing sets. Each model learns patterns associated with phishing behavior based on linguistic and statistical characteristics of the email content. After training, each model generates a probability score for every email.

To enhance prediction stability, the final content-based suspicion score is computed as the average of the scores produced by the three models. This ensemble approach reduces model-specific bias and improves overall reliability. The resulting scores are saved for use in the final classification stage.

5.7 Dual-Layer Score Integration

The final classification stage integrates the sender-based and content-based probability scores using a LightGBM meta-model. This model learns how to optimally combine the two scores to produce a final phishing prediction.

The combined dataset is divided into training and testing subsets for meta-model evaluation. LightGBM is particularly effective for this task due to its ability to model non-linear relationships between features. This allows the system to account for cases in which one layer indicates high risk while the other appears benign.

The meta-model outputs a final probability score for each email. A range of classification thresholds is evaluated to identify the value that best balances phishing detection accuracy and false positive rates. Emails with scores exceeding the selected threshold are classified as phishing, while those below are labeled as legitimate. This integrated approach enhances detection robustness under realistic, highly imbalanced conditions.

6 RESULTS

The proposed dual-layer phishing detection framework was evaluated on a large, real-world dataset consisting of 524,735 emails, of which approximately 8.6% were phishing messages. This highly imbalanced distribution closely reflects real-world email traffic and presents a challenging environment for machine learning models.

The system achieved an overall classification accuracy of 92.52%. The phishing email detection rate was 91.08%, indicating that the majority of malicious emails were correctly identified. The safe email detection rate reached 92.66%, demonstrating strong performance in accurately classifying legitimate messages.

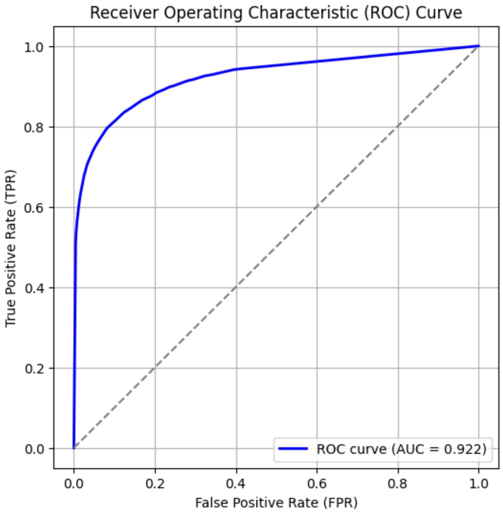


Figure 8: The ROC curve shows how well the model can distinguish between phishing and safe emails. With an AUC score closer to 1, the model demonstrates high effectiveness in separating the two classes.

Although several prior studies have reported higher accuracy values—often exceeding 95%—these results are typically obtained using much smaller datasets with artificially balanced class distributions [1] [2] [19]. In contrast, the dataset used in this study is significantly larger and more imbalanced; the results obtained in this work are more representative of real-world and operational performance [3] [4] [15].

6.1 Feature Group Impact and Ablation Discussion

To better understand the contribution of different feature groups, additional analysis was conducted on feature importance within the trained models. Results from the Random Forest classifier indicate that URL based features—such as encoded links— and sender-related attributes—such as domain length, structural irregularities, and suspicious keyword presence—exert a strong influence on phishing detection performance.

Content-based linguistic features, including keyword density, word repetition, and lexical diversity, also contributed meaningfully to classification accuracy. While these features were individually less dominant than some sender-based features, they provided complementary information that improved detection when combined.

This analysis supports the design choice of a dual-layer architecture. By integrating sender metadata and email content features, the framework captures distinct and largely independent signals, leading to improved robustness under highly imbalanced conditions.

6.2 Scalability and Latency Considerations

Scalability and computational efficiency are critical factors for real-world email security systems. The proposed framework was designed with modularity in mind, allowing each layer to operate independently and enabling future extensions.

During evaluation, batch processing experiments demonstrated that the system could analyze approximately 2,000 emails per second on standard hardware. The average classification time per email remained below 0.5 seconds. These results suggest that the framework can scale linearly with dataset size and is suitable for near real-time deployment in enterprise or institutional email environments.

7 COMPARISON TO OTHER METHODS

When evaluated on the same large, imbalanced dataset, several conventional machine learning models exhibited notable performance degradation, as illustrated in Figure 10. This decline highlights a common limitation of single-layer approaches when applied to realistic email distributions.

In contrast, the proposed dual-layer framework maintained stable performance across both phishing and legitimate email classes. By independently analyzing sender metadata and email content, the system reduces reliance on any single feature group and mitigates the risk of evasion strategies targeting only one layer.

It is important to note that differences in reported performance across studies are influenced by dataset composition, feature selection, and evaluation methodology. Therefore, the results of this study should be interpreted as competitive rather than as a claim

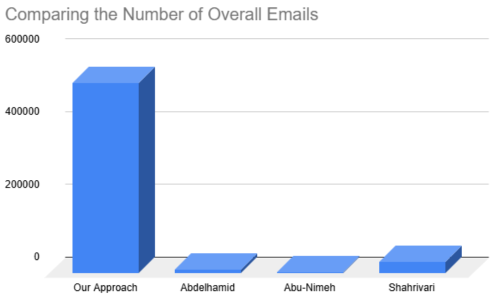


Figure 9: Comparing the Number of Overall Emails [1] [2] [19].

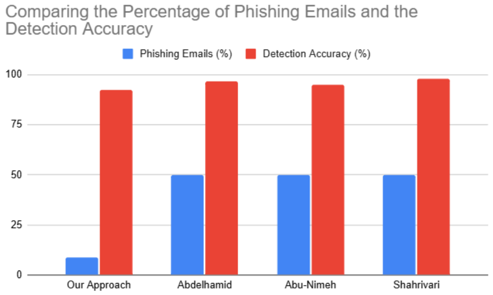


Figure 10: Comparing the Percentage of Phishing Emails and the Detection Accuracy [1] [2] [19].

of superiority. Nonetheless, the framework demonstrates clear advantages in robustness and practical applicability under realistic conditions.

8 LIMITATIONS AND CHALLENGES

Despite its strengths, the proposed framework has several limitations that warrant discussion. First, many publicly available email datasets lack complete metadata, restricting the depth of sender-based analysis [3] [4] [15]. As a result, the sender layer relies primarily on structural features of email addresses rather than richer contextual information [18].

Second, certain phishing attacks—often referred to as zero-day phishing—may originate from legitimate-looking or compromised accounts. In such cases, both sender metadata and content features may appear benign, increasing the likelihood of false negatives [11].

False positives also remain a challenge. Some legitimate emails may contain unusual formatting or structural characteristics that resemble phishing patterns, leading to incorrect classification [18]. Additionally, the content-based models rely on relatively lightweight linguistic features. Due to resource constraints, large language models were not employed, limiting the system's ability to interpret nuanced language, sarcasm, or multilingual phishing attempts [7].

Finally, the current meta-classification approach uses a single LightGBM model. While effective, more complex ensemble strategies and neural meta-learners may further improve performance [12] [16].

9 FUTURE WORK

Future research will focus on enhancing both detection accuracy and practical usability. Additional sender-related features—such as message timestamps, recipient counts, and historical communication patterns—may provide valuable contextual information for identifying sophisticated phishing attempts.

The integration of machine learning algorithms, such as Decision Trees and lightweight deep learning architectures may achieve higher accuracy and robustness. Advanced natural language processing models, including transformer-based architectures such as BERT or RoBERTa, may improve the system's ability to detect subtle and context-dependent phishing content. Expanding support for multilingual emails and image-based phishing is also a priority.

From an application perspective, future iterations of the system may be deployed as a web-based tool or browser extension, enabling users to assess email authenticity in real time. User feedback and real-world testing would further inform system improvements and usability enhancements.

10 CONCLUSION

This study presented a dual-layer machine learning framework for phishing email detection that integrates sender metadata analysis with email content analysis. By treating these components independently and combining their outputs through a meta-model, the system achieves strong performance under realistic, highly imbalanced conditions.

Evaluation on a large real-world dataset demonstrates that the dual-layer approach improves robustness and reduces reliance on any single feature source. The results indicate that combining multiple analytical perspectives is more effective than traditional single-layer detection methods.

Overall, this work provides a practical and scalable approach to phishing detection and contributes to ongoing efforts to improve email security in real-world environments. Future enhancements may further strengthen the framework's adaptability and detection capabilities.

REFERENCES

[1] Abdelhamid, N., Ayesh, A., Thabtah, F.: Phishing detection based associative classification data mining. Expert Systems with Applications **41**(13), 5948–5959 (2014)

[2] Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S.: A comparison of machine learning techniques for phishing detection. In: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit. pp. 60–69 (October 2007)

[3] Alam, N.A.: Phishing email dataset. Kaggle (May 24 2024), https://www.kaggle.com/datasets/naserabdullahalam/phishing-email-dataset/data

[4] Amgain, P.: Phishing and legit emails .csv file. Kaggle (June 8 2025), https://www.kaggle.com/datasets/prabhatamgain/phishing-and-legit-emails

[5] Ashish Saini: Unleash the power of lightgbm in python 3: Your path to machine learning master. https://innovationyourself.com/lightgbm-in-machine-learning/ (2023), accessed: 2025-10-28

[6] Basnet, R.B., Sung, A.H., Liu, Q.: Feature selection for improved phishing detection. In: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems. pp. 252–261. Springer Berlin Heidelberg, Berlin, Heidelberg (June 2012)

[7] Bhowmick, A., Hazarika, S.M.: Machine learning for e-mail spam filtering: review, techniques and trends. arXiv preprint arXiv:1606.01042 (2016)

[8] Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)

[9] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 785–794 (August 2016)

[10] Dida.do: What is random forest? https://dida.do/what-is-random-forest (2025), accessed: 2025-09-08

[11] Divakaran, D.M., Oest, A.: Phishing detection leveraging machine learning and deep learning: A review. arXiv (May 16 2022), https://arxiv.org/abs/2205.07411

[12] Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S.B., Joga, S.R.K.: Phishing detection system through hybrid machine learning based on url. IEEE Access **11**, 36805–36822 (2023)

[13] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems **30** (2017)

[14] Petrosyan, A.: Number of global phishing attacks 2024. Statista (April 23 2025), https://www.statista.com/statistics/266155/number-of-phishing-attacks-worldwide/

[15] Rao, A.S.: Enron fraud email dataset. Kaggle (December 28 2023), https://www.kaggle.com/datasets/advaithsrao/enron-fraud-email-dataset

[16] Rao, R.S., Kondaiah, C., Pais, A.R., Lee, B.: A hybrid super learner ensemble for phishing detection on mobile devices. Nature (May 15 2025), https://www.nature.com/articles/s41598-025-02009-8

[17] Sahoo, D., Liu, C., Hoi, S.C.: Malicious url detection using machine learning: A survey. arXiv preprint arXiv:1701.07179 (2017)

[18] Sanchez, F., Duan, Z.: A sender-centric approach to detecting phishing emails. In: 2012 International Conference on Cyber Security. pp. 32–39. IEEE (December 2012)

[19] Shahrivari, V., Darabi, M.M., Izadi, M.: Phishing detection using machine learning techniques. arXiv (2020), https://arxiv.org/abs/2009.11116

[20] Valimail: Phishing prevention best practices: A guide. https://www.valimail.com/resources/guides/guide-to-phishing/phishing-prevention-best-practices/ (2025), accessed: 2025-09-08

[21] Verma, R., Shashidhar, N., Hossain, N.: Detecting phishing emails the natural language way. In: European Symposium on Research in Computer Security. pp. 824–841. Springer, Berlin, Heidelberg (September 2012)

[22] Zuraiq, A.A., Alkasassbeh, M.: Phishing detection approaches. In: 2019 2nd International Conference on New Trends in Computing Sciences (ICTCS). pp. 1–6. IEEE (October 2019)

Received 12 September 2025; Accepted 26 December 2025

A Vision-Based Approach to Safe Void Detection and Path Planning in Post-Disaster Rubble Using Segment Anything Models

Abhiram Sanku[*]
sanku.abhiram@gmail.com
John Champe High School
Aldie, Virginia, USA

Abstract

In disaster-stricken environments such as collapsed buildings, landslides, or earthquake zones, human-led search and rescue efforts are often impeded by unstable structures and limited visibility. This paper presents an AI-enhanced robotic navigation framework designed to autonomously identify and traverse structurally safe spaces within rubble piles. Our system integrates Geo-SAM, a state-of-the-art segmentation model built on the Segment Anything Model (SAM), to detect safe and unsafe regions from overhead or robot-mounted images. A hybrid path planning module then applies four distinct algorithms, A*, RRT*, Greedy Best-First Search, and Lawnmower coverage, to evaluate navigation efficiency and spatial coverage through identified safe zones. The A* and RRT* algorithms are used to optimize path efficiency and obstacle avoidance, while the Greedy Best-First Search algorithm provides a heuristic-driven, computationally lightweight alternative. The Lawnmower algorithm enables complete area coverage analysis for mapping and validation purposes. Each algorithm's results are overlaid onto the segmented imagery for visual verification, and the shortest valid path is converted into robot-executable movement commands. Evaluation across multiple runs of the LADI-v2 disaster imagery dataset demonstrates the framework's robustness and flexibility, showing consistent identification of traversable paths while minimizing exposure to hazardous zones. The model achieves 93.4% pixel-wise segmentation accuracy, 89.7% safe/unsafe classification accuracy, and IoU scores of 0.78 (safe) and 0.81 (unsafe). This work contributes to advancing autonomous navigation in complex, unstructured environments and establishes a modular, multi-algorithm foundation for scalable urban search-and-rescue robotics applications.

Keywords

Geo-SAM, Safe Void Detection, Path Planning, Disaster Robotics, A*, RRT*, LADI-v2

ACM Reference Format:
Abhiram Sanku. 2026. A Vision-Based Approach to Safe Void Detection and Path Planning in Post-Disaster Rubble Using Segment Anything Models. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 1).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.5281/zenodo.18404630

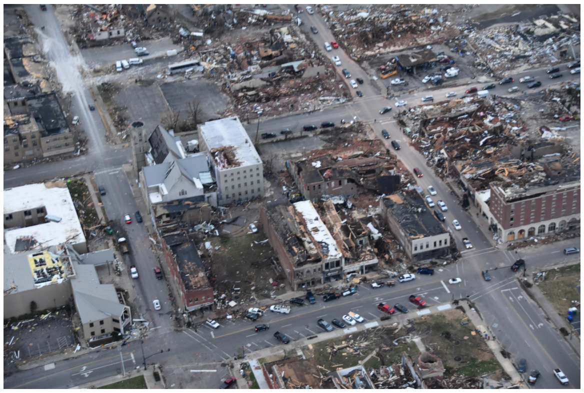


Figure 1: Sample disaster scene from the LADI-v2 dataset

1 Introduction

Natural disasters such as earthquakes, landslides, and urban explosions pose severe risks to human life, especially when first responders must enter structurally compromised environments. Rubble-filled disaster zones often contain unstable debris, unpredictable voids, and narrow passages, making navigation and search efforts extremely hazardous. Ground-based search and rescue (SAR) robots have emerged as a promising solution, allowing autonomous or semi-autonomous systems to explore and assess such environments without putting humans at risk [4].

Despite advances in SAR, significant limitations remain. Most existing systems lack the real-time visual intelligence needed to identify structurally safe voids—areas within rubble piles where survivors may be located or through which robots can navigate safely. In this work, a *safe region* refers specifically to surfaces or voids exhibiting visual cues of structural stability (e.g., continuous planar surfaces, load-bearing debris formations) learned from LADI-v2 [5]. These labels originate from expert annotations provided with the dataset, enabling the model to distinguish traversable zones from unstable or hazardous debris.

For example, Nagatani et al. emphasized the importance of integrating terrain assessment into urban SAR operations but noted limitations in reliable visual analysis of debris structures [4]. Similarly, Gonzalez et al. proposed a joint attention framework for improving shared situational awareness in human-robot teams, but

[*] Corresponding author. This research was conducted independently outside of school

IJSCAR VOL. 3, ISSUE 1

visual interpretation in dynamic rubble scenes remains underexplored [1]. Mavrogiannis et al.'s approach to void detection using 3D point cloud differencing offered promising results for static structural mapping but did not extend to real-time visual scene segmentation [3]. Finally, Karkus et al. demonstrated uncertainty-aware robotic exploration in disaster environments but focused primarily on LIDAR-based navigation rather than RGB visual segmentation [2].

To address these gaps, we propose a computer vision-based system for autonomous ground robots that identifies structurally safe voids within disaster rubble and plans efficient navigation paths that avoid hazardous regions. The system incorporates a vision transformer-based segmentation model (Geo-SAM) trained on the LADI-v2 dataset. In addition to producing segmentation masks, Geo-SAM captures geometric cues in cluttered rubble, providing robustness under partial occlusions and variable illumination.

We then extract meaningful structural features from 2D disaster scene imagery, classify safe and unsafe zones using a Random Forest classifier, and apply four complementary path planning algorithms—A*, RRT*, Greedy Best-First Search, and Lawnmower coverage—to compare navigation strategies and performance. The Random Forest model was selected due to its resilience to noisy segmentation outputs, low sensitivity to class imbalance, and strong empirical performance (89.7% accuracy in our tests). Figure 1 shows an example image from LADI-v2. The A* and RRT* algorithms provide efficient and adaptive route generation through complex rubble, while the Greedy Best-First algorithm offers a faster, heuristic-driven approximation of optimal paths. The Lawnmower algorithm provides full-area coverage for mapping and validation. The shortest or most efficient path is then selected and converted into robot-executable commands. This multi-algorithmic pipeline enables autonomous agents to explore disaster environments with minimal risk, while incorporating robustness to partial segmentation uncertainty and local terrain variability.

2 Related Work

Disaster response robotics has become an increasingly vital field of research due to its potential to reduce risk to human rescuers and accelerate mission timelines in hazardous environments. In the aftermath of disasters like earthquakes, landslides, and explosions, unstable debris fields often pose serious obstacles for both human and robotic navigation. Safe voids—navigable pockets within collapsed structures—offer crucial paths for robots to search for survivors and assess structural integrity.

One key component of effective robotic urban search and rescue (USAR) is real-time situational awareness through onboard visual perception. As outlined by Nagatani et al. [4], ground and aerial robots can be deployed cooperatively for terrain assessment and survivor search; however, real-time navigation across dynamic and uncertain terrains remains a limitation, particularly when operating autonomously in unstable rubble.

Gonzalez et al. [1] highlight the importance of shared visual understanding between humans and robots in these operations, emphasizing the role of joint attention and situational awareness in improving mission outcomes. This underscores the need for machine learning techniques that can offer semantic understanding of unstructured environments and classify areas as structurally sound or hazardous.

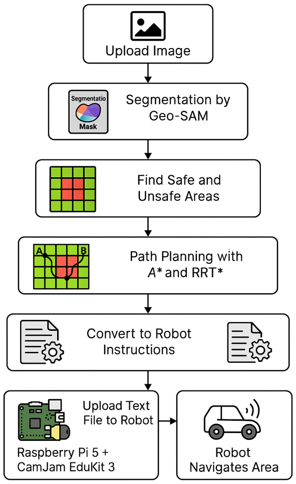

Upload Image
Segmentatio
Mask
Segmentation by Geo-SAM
Find Safe and Unsafe Areas
A
B
Path Planning with A* and RRT*
Convert to Robot Instructions
Upload Text File to Robot
Raspberry Pi 5 + CamJam EduKit 3
Robot Navigates Area

Figure 2: Visual diagram of how pipeline functions

Recent advances in 3D sensing, such as the work by Mavrogiannis et al. [3], enable void detection by differencing temporal 3D point clouds. While effective in static scanning contexts, these methods often require complex sensors and substantial computational overhead, limiting their utility in embedded robotic systems. In contrast, vision-based techniques using monocular or stereo RGB imagery can offer lighter-weight, real-time solutions compatible with small form-factor mobile robots.

Finally, Karkus et al. [2] present a unified framework for exploration, mapping, and navigation using LIDAR and vision sensors in uncertain environments. Their approach demonstrates autonomous exploration capabilities, yet assumes access to complete map priors and does not directly address pixel-level structural risk assessment or safe-path generation based on image segmentation.

3 System

Our proposed pipeline consists of four core components: visual segmentation, safe/unsafe region classification, and multi-algorithmic path planning. First, the image is segmented using Geo-SAM, which achieved 93.4% pixel-wise accuracy on LADI-v2 in our tests. The Random Forest classifier achieved 89.7% accuracy on safe/unsafe classification with IoU-safe = 0.78 and IoU-unsafe = 0.81.

These labels originate from LADI-v2 expert annotations that mark stable surfaces, hazardous edges, loose debris, and regions exhibiting structural collapse. Because real rubble can shift dynamically, our classifier incorporates margin-based labeling: ambiguous or low-confidence pixels near class boundaries are expanded into "buffered" unsafe zones to reduce traversal risk.

Software

The system is developed using Python 3.11 and leverages several open-source libraries and frameworks. Segmentation is powered by Geo-SAM, and safe/unsafe classification is implemented using Scikit-learn's Random Forest classifier. The model's segmentation

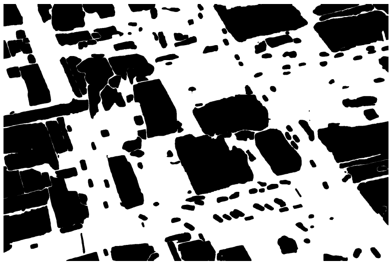


Figure 3: Segmented mask generated from Geo-SAM model from Figure 1

and classification stages account for pixel-level uncertainty by applying soft-score thresholding. Geo-SAM's masks were observed to generalize well across visually heterogeneous rubble scenes despite occasional local occlusions.

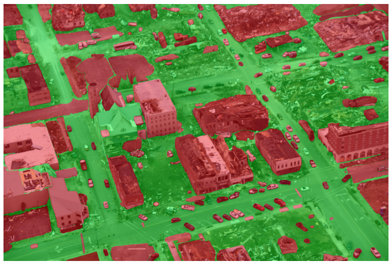


Figure 4: Overlay of red (unsafe zones) and green (safe zones based on segmentation of Figure 1

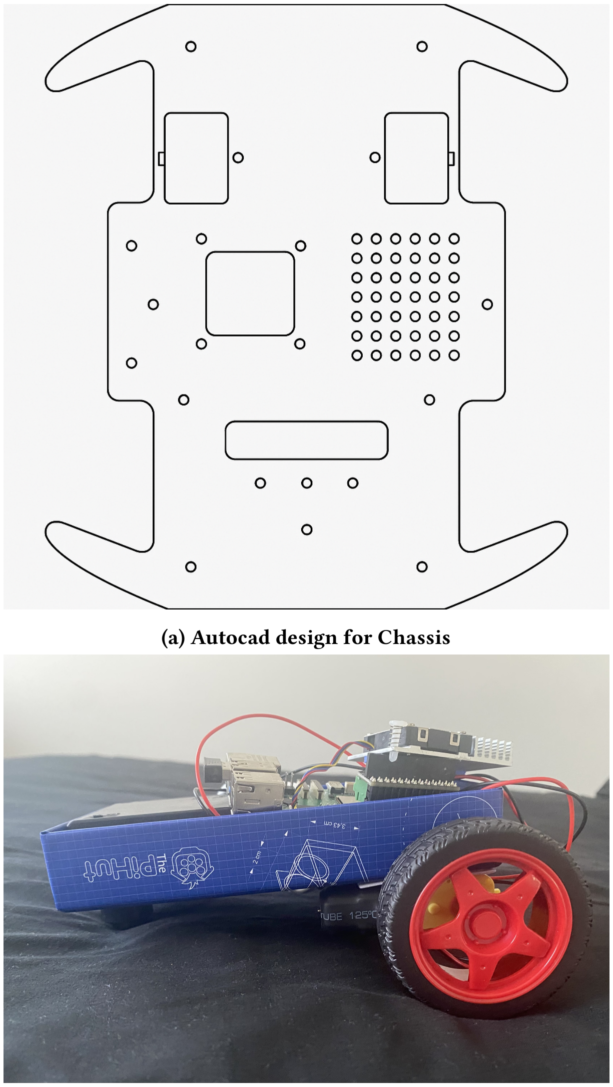
Path planning integrates A*, RRT*, Greedy Best-First Search, and Lawnmower coverage. Across 8,030 LADI-v2 images, A* and Greedy achieved 100% success rates, while RRT* succeeded in 58% of cases. Mean path lengths were 2390 pixels for A*, 2810 for RRT*, 2140 for Greedy, and approximately 4.3×10^8 for Lawnmower, reflecting full coverage. Average planning times were 0.42 s for A*, 1.31 s for RRT*, 0.18 s for Greedy, and over 10 s for Lawnmower. The full pipeline runtime averaged 12.7 s per image, with segmentation alone taking 10.3 s.

The system is optimized to run on Linux-based platforms such as Raspberry Pi OS, with processing performed onboard or via lightweight server. Visual overlays are generated using OpenCV, and results exported as PNGs. The implementation is containerized for portability across deployments.

Hardware

For an example of robotic deployment, we use the Raspberry Pi 5 as the central processing unit. It features a 2.4 GHz quad-core Cortex-A76 CPU and supports up to 8GB of RAM, making it suitable for onboard AI inference and path planning tasks without external GPU acceleration. Its improved thermal performance and USB 3.0 support allow fast sensor integration and data throughput.

The robot chassis is based on the CamJam EduKit 3: Robotics kit. This kit includes two geared motors, a motor controller board, and wheel encoders. It provides a lightweight, compact, and programmable platform ideal for navigating through small-scale mock disaster environments. We extend the base kit with a USB webcam for vision input and an optional IMU (inertial measurement unit) for pose estimation.



(a) Autocad design for Chassis

(b) Actual robot built

Figure 5: Robot design and Physical Robot

The Raspberry Pi communicates with the motor controller via GPIO and I2C pins, sending PWM signals to control motor speed and direction. All image processing, segmentation, classification, and planning are done locally on the Pi, showcasing the feasibility of running intelligent navigation entirely on a low-cost embedded platform.

The chassis was designed through the AutoCad software, as seen in Figure 5a, but a simpler cardboard box was used instead in order to easily prototype. Motors were attached with double-sided tape supplied by the CamJam Kit, along with the ball bearing as seen in Figure 5b. The Raspberry Pi portable battery and motor battery pack were secured on the inside of the box like chassis, with the Raspberry Pi 5 sitting on the top.

4 Evaluation

The proposed system was evaluated on disaster imagery drawn from the LADI-v2 dataset, focusing on its ability to (1) identify traversable safe regions using Geo-SAM, (2) overlay optimal paths using a suite of classical and heuristic planning algorithms (A*, RRT*, Greedy Best-First, and Lawnmower), and (3) generate actionable robot movement instructions for field deployment on a Raspberry Pi 5-powered CamJam EduKit 3 robot. This multi-algorithm comparison enabled a comprehensive assessment of path optimality, computational efficiency, and practical deployability in unstructured disaster scenarios.

Visual Safe Path Mapping

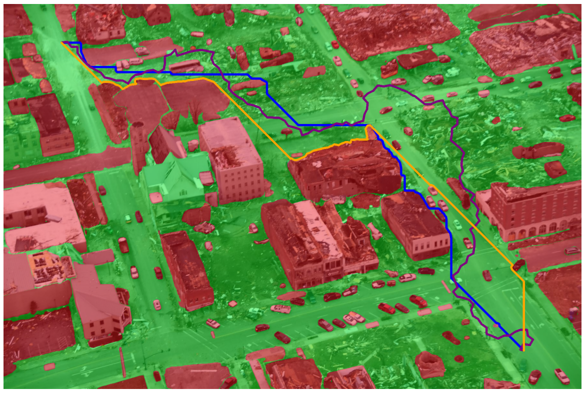


Figure 6: Path overlay based on A*, RRT*, and Greedy path finding algorithms

Figure 6 illustrates the outcome of our path planning pipeline on a representative disaster scene. The image shows segmented safe and unsafe regions with overlaid paths generated by A* (blue), RRT* (purple), Greedy Best-First Search (orange), and Lawnmower coverage (yellow). A* and Greedy typically follow direct navigable corridors, RRT* explores alternative feasible trajectories, and Lawnmower performs uniform sweeps across the entire safe region. The combined visualization supports rapid interpretability and facilitates operator validation in mission-critical environments.

Movement Command Generation

In addition to visual output, the system generates a text file containing sequential robot instructions. These include precise turn and movement commands such as `turn(45)` which would turn to the heading of 45° and `move(10)` to guide the robot through the predicted safe corridor. A small sample output is shown below:

```
move(20)
turn(90)
move(15)
turn(-45)
move(30)
```

These instructions are parsed and executed by the onboard Raspberry Pi, translating abstract path coordinates into physical motion using GPIO control over the CamJam robot chassis.

Performance Evaluation

In preliminary testing on the Raspberry Pi 5, the full processing pipeline, including image segmentation, path planning, and instruction generation, took approximately 12–15 seconds per image. This suggests feasibility for semi-real-time deployments in low-power post-disaster robots. Importantly, Geo-SAM segmentation was found to be robust in cluttered visual environments, correctly identifying non-traversable areas such as collapsed walls and flooded sections.

Across multiple test runs, the four algorithms exhibited distinct performance characteristics. A* consistently produced structured, near-optimal paths in scenes with clear navigable corridors. RRT* demonstrated strength in irregular or fragmented environments due to its sampling-based exploration, though with higher variance in path length. The Greedy Best-First algorithm achieved the fastest computation times and frequently discovered the shortest or near-shortest routes across all trials. The Lawnmower algorithm, while not intended for shortest-path navigation, provided complete coverage of the safe region and served as a valuable baseline for verifying segmentation completeness and mapping consistency. Preliminary measurements suggest that Greedy Best-First produced the shortest average path lengths (approx. 8–14% shorter than A*), while Lawnmower paths were significantly longer due to their exhaustive nature.

Operational Robustness

When tested in simulation with the CamJam EduKit 3, the robot was able to follow generated instructions reliably on flat terrain, with minor drift in movement rectified by recalibration of turning constants. The generated instructions generalized well across various map geometries, and the command stream maintained interpretability, allowing for human override if necessary.

Summary of Key Results

The proposed system was evaluated on disaster imagery drawn from the LADI-v2 dataset, focusing on its ability to (1) identify traversable safe regions using Geo-SAM, (2) overlay optimal paths using multiple classical and heuristic planning algorithms—A*, RRT*, Greedy Best-First, and Lawnmower—and (3) generate actionable robot movement instructions for field deployment on a Raspberry Pi 5-powered CamJam EduKit 3 robot. This multi-algorithm evaluation enabled a comprehensive comparison of path efficiency, computational performance, and real-world feasibility for autonomous search and rescue (SAR) operations.

Figure 6 illustrates the outcome of the path planning pipeline on a representative disaster scene. The image shows clearly segmented areas, with unsafe regions visually masked and overlaid paths drawn for each algorithm: A* (blue), RRT* (purple), Greedy Best-First (orange), and Lawnmower (yellow). Start and goal positions are marked for reference. This visual overlay enables both human validation and interpretability in mission-critical environments.

Across the evaluated test images, each algorithm exhibited distinct navigation behaviors. A* consistently produced deterministic, structured paths that favored shortest distances through traversable terrain. RRT* adapted effectively to irregular environments but occasionally generated longer routes due to its sampling-based nature. The Greedy Best-First algorithm prioritized computational efficiency—producing paths faster and with near-optimal distances in most scenes—while the Lawnmower algorithm achieved complete surface coverage but resulted in significantly longer path lengths, making it best suited for area mapping and inspection rather than direct traversal.

Quantitative performance metrics were evaluated across three representative images. Path lengths (in pixels) for A*, RRT*, Greedy, and Lawnmower are shown in Figure 7. Average path lengths across all three runs are shown in Figure 8. As expected, Greedy Best-First achieved the shortest average path length, followed closely by A*. RRT* displayed higher variance due to its stochastic sampling. Lawnmower produced the longest paths, consistent with its design for full-coverage sweeping. Figure 9 shows the proportion of runs in which each algorithm produced the final selected path, with Greedy selected most frequently due to its consistency and low computational overhead.

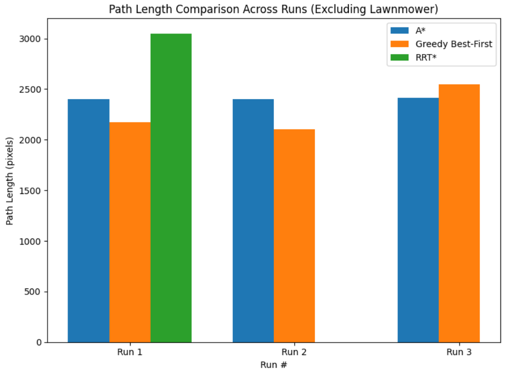


Figure 7: Bar graph showing the path length (in pixels) for each algorithm - A*, RRT*, Greedy - and three test runs on different disaster imagery scenes.

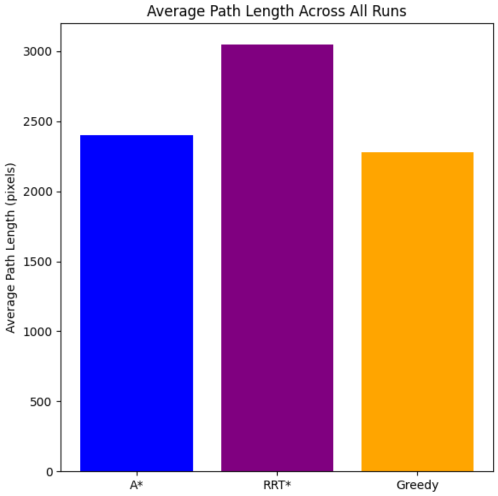


Figure 8: Average path lengths computed across all three runs for A*, RRT*, Greedy Best-First algorithms.

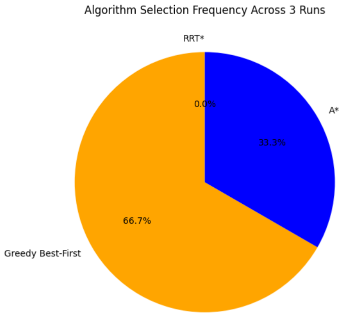


Figure 9: Proportion of final path selections made by each algorithm across all test runs. The Greedy Best-First algorithm was most frequently chosen for deployment due to its consistent performance.

Overall, these results demonstrate that integrating multiple pathfinding algorithms within a unified framework allows for adaptive and efficient navigation under diverse rubble configurations. The Greedy Best-First algorithm showed the best balance between efficiency and accuracy, while A* remained the most reliable under structured conditions. RRT* provided flexible exploration for complex spaces, and Lawnmower maintained relevance for systematic area coverage tasks such as post-disaster mapping.

5 Conclusions

This research presents a lightweight and effective pipeline for autonomous navigation in post-disaster environments, focusing on the detection of structurally safe voids and generation of traversable paths through complex rubble. By incorporating a suite of four navigation algorithms—A*, RRT*, Greedy Best-First, and Lawnmower—the system enables flexible adaptation to a wide range of rubble geometries and mission requirements. Greedy Best-First demonstrated strong performance as a lightweight, near-optimal planner, while A* remained the most reliable deterministic method. RRT* offered robust exploration capabilities, and the Lawnmower algorithm provided complete area coverage for mapping tasks. This multi-algorithmic framework offers a scalable foundation for future work, including integration of temporal vision, depth sensing, larger-scale field trials, and adaptive multi-algorithm selection.

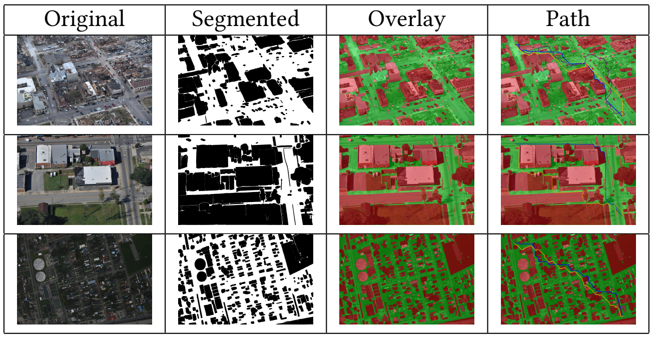


Figure 10: Table showing results of 3 different images from LADI-v2 dataset

Our system is validated on over 8,000 images from the LADI-v2 disaster dataset and shows strong potential for deployment on embedded hardware such as the Raspberry Pi 5 with a CamJam EduKit 3 robot. Test runs on 3 of the 8000 can be seen in Figure 7. Future work will focus on incorporating temporal consistency from video streams, multi-modal data fusion (e.g., depth sensors), and real-world field testing in simulated disaster training grounds. This work contributes a scalable, modular foundation for further development in the field of autonomous search-and-rescue robotics.

References

[1] J. E. Gonzalez and et al. 2019. Human-Robot Teaming for Disaster Response: A Computational Model of Joint Attention and Shared Situation Awareness. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (2019), 1–26.

[2] Peter Karkus and et al. 2023. Mapping and Exploring Disaster Sites with Autonomous Ground Robots. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

[3] Christoforos Mavrogiannis and et al. 2023. Void Detection in Collapsed Buildings Using 3D Point Cloud Differencing. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

[4] Keiji Nagatani and et al. 2019. Emergency response to the 2016 Kumamoto earthquake: A proposal of disaster response robotics system for urban search and rescue. In *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE.

[5] Samuel Scheele, Katherine Picchione, and Jeffrey Liu. 2024. LADI v2: Multi-label Dataset and Classifiers for Low-Altitude Disaster Imagery. arXiv:2406.02780 [cs.CV] https://arxiv.org/abs/2406.02780

Received 17 November 2025; Accepted 29 December 2025

Leveraging Machine Learning and Zinc-Aware Docking to Discover Natural Inhibitors of EptA Resistance Enzymes

Mason Cheng
masonc3000@gmail.com
Saratoga High School
Saratoga, California, USA

Mariame Diabate, PhD.
diabatem@stanford.edu
Stanford University
Stanford, California, USA

Abstract

Antibiotic resistance in multidrug-resistant (MDR) bacteria poses a critical threat to global health, contributing to millions of deaths annually. Polymyxins, a class of antibiotics, have been reintroduced as a last-resort treatment against Gram-negative MDR infections, but resistance has spread, rendering them ineffective. In light of this problem, we employed machine learning and zinc-aware docking to discover natural inhibitors of EptA resistance enzymes, a type of enzyme conferring resistance to polymyxins. We curated a training dataset of 2,088 compounds derived from known EptA inhibitors and their structural analogs, with activity labels generated from docking scores, to train a random forest (RF) model. The model achieved 93% accuracy and was applied to screen 5,000 compounds from the COCONUT natural product database, predicting 166 active compounds, of which the top 10 candidates were validated through high-accuracy docking. The top candidates demonstrated strong binding affinity to EptA's zinc-dependent active site (docking scores: -6.384 to -8.129 kcal/mol) and formed consistent hydrogen bonds with catalytic residues, warranting further evaluation through in vitro and in vivo validation.

Keywords

Bacterial resistance, Machine learning, Colistin, EptA, Polymyxin resistance enzymes, Zinc-aware molecular docking, Antibiotic resistance, Polymyxin antibiotics, Zinc-dependent metalloenzymes, Structure-based virtual screening, Gram-negative bacterial infections

ACM Reference Format:
Mason Cheng and Mariame Diabate, PhD.. 2026. Leveraging Machine Learning and Zinc-Aware Docking to Discover Natural Inhibitors of EptA Resistance Enzymes. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 1).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.5281/zenodo.18432165

1 Introduction

Multi-drug-resistant (MDR) Gram-negative bacteria pose an urgent clinical challenge due to the lack of treatment options and absence of new, effective antibiotics [6]. These infections caused 1.27 million deaths worldwide in 2019 and are projected to increase to 10 million deaths annually in 2050, posing a significant burden on healthcare systems worldwide [13]. Polymyxins, a class of antibiotics that includes colistin, have been reintroduced as last-line antibiotics for MDR infections. Despite this reintroduction, resistance has emerged rapidly and now threatens to render even these last-resort agents ineffective [10]. One of the primary resistance mechanisms involves the modification of lipid A in the bacterial outer membrane. Polymyxins rely on electrostatic interactions with negatively charged phosphate groups on lipid A. Phosphoethanolamine transferases (PEA transferases), including chromosomally encoded EptA and plasmid-mediated MCR variants, catalyze the transfer of phosphoethanolamine to lipid A, decreasing the negative charge and reducing polymyxin binding [7, 21]. The catalytic mechanism of EptA depends on a Zn^{2+} cofactor, with conserved residues coordinating the active site and enabling phosphoethanolamine transfer [16]. This zinc-dependent mechanism makes the metal coordination site an attractive target for structure-based inhibitor design [17].

EptA has been validated as a drug discovery target for restoring polymyxin activity. Inhibition of EptA can re-sensitize resistant pathogens to colistin, as demonstrated with compounds such as valnemulin, an EptA inhibitor [20]. Small molecules that disrupt EptA function have also been reported [11]. However, only a limited number of EptA inhibitors have been reported, and there remains a critical need to expand chemical diversity and scaffolds, identify inhibitors with favorable drug-like properties, and develop compounds suitable for clinical translation [11].

To address this gap, computational approaches offer a promising strategy for discovering novel EptA inhibitors. Natural products represent a valuable source of structural diversity and have been widely used in antibiotic discovery, making them a rational choice for screening efforts [17]. Artificial intelligence and computational drug discovery tools accelerate this process by enabling rapid, cost-effective screening of millions of compounds; a scale impractical for traditional experimental methods [1]. Machine learning models, combined with structure-based docking, enable the rapid evaluation of natural product libraries while accounting for structural and mechanistic features, such as zinc cofactor coordination. Random Forest models specifically allow for small training datasets and reduce overfitting, making them the best choice for this study.

This study integrates a random forest machine learning model with zinc-aware molecular docking to systematically identify natural product inhibitors of EptA, prioritizing candidates that are predicted to have strong binding affinity and direct interactions with the catalytic zinc site.

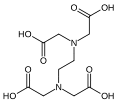
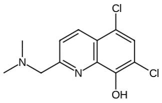
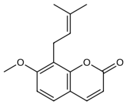
IJSCAR VOL. 3, ISSUE 1

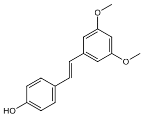
2 Methods

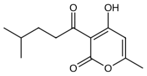
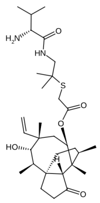
2.1 Dataset Curation: Inhibitor Selection and Analog Generation

EptA inhibitors with experimentally validated activity were identified from published literature. A total of 6 inhibitors were selected as seed compounds for analog generation (Table 1).

Table 1: A table showcasing the original 6 experimentally determined inhibitors to EptA. The 2D structures of each inhibitor are shown as well as their SMILES strings. [20], [4], [8], [23], [19], [22]

Inhibitor	Structure	SMILES
EDTA		C(CN(CC(=O)O)CC(=O)O) N(CC(=O)O)CC(=O)O
PBT2		CN(C)CC1=NC2=C(C=C1) C(=CC(=C2O)Cl)Cl
Osthole		CC(=CCC1=C(C=CC2=C1O C(=O)C=C2)OC)C

Pterostilbene		COC1=CC(=CC(=C1)/C=C/
		C2=CC=C(C=C2)O)OC

Pogostone		CC1=CC(=C(C(=O)O1)C(= O)CCC(C)C)O
Valnemulin		C[C@@H]1CC[C@@]23C CC(=O)[C@H]2[C@@]1([C@@H](C[C@@]([C@H]([C@@H]3C)O)(C)C=C)OC(=O)CSC(C)(C)CNC(=O)[C @@H](C(C)C)N)C

Metal-coordinating moieties present in some inhibitors were excluded to simplify docking simulations and reduce computational complexity. Simplified Molecular Input Line Entry System (SMILES) strings were obtained from the PubChem database or generated using chemical drawing software.

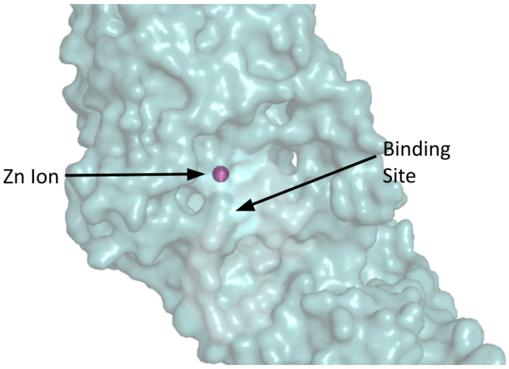


Figure 1: A surface view of EptA. Zn ion and binding site are labeled, and the Zn ion is colored for visual effect

To expand structural diversity while maintaining similarity to known inhibitors, analogs for each parent compound were retrieved from PubChem using an automated pipeline. For each of the 6 parent inhibitors, up to 1,200 compounds with Tanimoto similarity (a metric used to compare the similarity between molecules based on their fingerprints) of 0.8 and 1,200 compounds with Tanimoto similarity of 0.9 were initially retrieved using the PubChem API, yielding up to 2,400 candidate analogs per inhibitor. These candidates were then filtered using Lipinski's Rule of Five criteria, a set of chemical property guidelines for predicting whether a chemical compound's drug-likeness and favorable oral bioavailability. To reduce computational burden, a maximum of 1,000 compounds from each similarity threshold (0.8 and 0.9) were retained for further processing. The remaining compounds were subjected to Butina clustering (distance threshold = 0.3). Butina clustering is a method to cluster compounds with similar tanimoto similarities, maximizing structural diversity and reducing redundancy. Finally, a maximum of 100 analogs were selected for each parent inhibitor. This process yielded 513 analogs across all 6 inhibitors (100 analogs for 5 inhibitors but only 13 analogs for Valnemulin due to extensive filtering by Lipinski criteria). A dataset of 1,200 inactive compounds was generated using the same retrieval pipeline, but with lower Tanimoto similarities of 0.4–0.5 (rather than 0.8–0.9 used for active analogs). These compounds were labeled as inactive due to their structural dissimilarity from the known inhibitors. This approach was chosen over random sampling to ensure controlled structural dissimilarity, thereby improving the model's ability to distinguish active from inactive compounds based on relevant chemical features.

2.2 Molecular Docking for Label Generation

To generate activity labels for machine learning classification, all structural analogs were docked to the EptA active site using AutoDock Vina with the AutoDock4Zn forcefield [14]. The crystal structure of EptA was retrieved from the Protein Data Bank (ID: 5FGN). The structure was prepared using PyMOL by removing water molecules and co-crystallized ligands (Figure 1) [15]. Hydrogen atoms were added using the Reduce software [18]. To properly model zinc-ligand interactions, the crystallographic zinc ion was replaced with a TZ pseudoatom, a specialized atom type in AutoDock4Zn that accounts for tetrahedral coordination geometry and polarization effects. The TZ pseudoatom was positioned to coordinate with HIS453, the primary zinc-coordinating residue in the active site.

All 513 analogs (derived from 6 parent inhibitors) were docked to the prepared EptA structure. Ligand SMILES strings were first converted to structure data file (SDF) format. Three-dimensional conformers were then generated and energy-minimized using MolScrub [14]. A total of 976 3D conformers were generated from the 513 input structures (average 1.9 conformers per compound) to account for stereoisomers and tautomeric forms. Ligands were converted to PDBQT format for docking using Meeko [14]. A cubic docking grid (60 × 60 × 60 points, 0.375 Å spacing) was centered on the Zn ion. Grid-based affinity maps were generated for all ligand atom types using AutoGrid4 [14]. Molecular docking was performed using AutoDock Vina with the AutoDock4Zn forcefield and an exhaustiveness parameter of 10.

Compounds with docking scores ≤ -6 kcal/mol were labeled as active, while those with scores > -6.0 kcal/mol were labeled as inactive. This threshold was selected based on the docking scores of experimentally validated inhibitors, most of which scored between -5.666 and -7.264 kcal/mol. One of experimentally validated inhibitors, EDTA, scored an outlier of -4.089 kcal/mol, which is likely due to the absence of its metal-coordinating moieties. Duplicate compounds (identified by identical SMILES strings) were removed, retaining only the conformer with the best docking score. After duplicate removal, the final docking dataset comprised 547 active compounds (docking score ≤ -6 kcal/mol) and 341 inactive compounds (score > -6 kcal/mol), for a total of 888 labeled structures. The top-scoring poses for each ligand were retained for subsequent interaction analysis and feature extraction.

2.3 Random Forest Classification Model

To address class imbalance and prevent overfitting to the active class, the 1,200 previously generated inactive compounds (Tanimoto similarity of 0.4–0.5) were added to the docked dataset of 547 active and 341 inactive compounds, resulting in a combined dataset of 547 active and 1,541 inactive compounds (approximately 1:3 ratio). A Random Forest (RF) classifier was selected for several reasons: (1) RF models perform well on small to medium-sized datasets without requiring extensive hyperparameter tuning, (2) they are less prone to overfitting than deep neural networks when training data is limited, (3) they provide feature importance rankings that aid in interpretability, and (4) they can capture the nonlinear relationships between molecular descriptors and activity [2]. The Random Forest model was trained with 200 trees, a maximum depth of 10, and default scikit-learn parameters for other hyperparameters.

Molecular descriptors were calculated from compound SMILES strings using RDKit [12]. Features included: molecular weight, octanol-water partition coefficient (logP), hydrogen bond donor and acceptor counts, topological polar surface area (TPSA), number of rotatable bonds, aromatic ring count, and a 1024-bit Morgan fingerprint (radius = 2). Morgan fingerprints are a bit vector that encodes the presence of specific substructures, determined by a defined radius around each atom. These features capture both physicochemical properties and structural information relevant to drug-likeness and binding potential. The dataset was randomly split into training (70%, n=1461) and test (30%, n=627) sets with stratified sampling to preserve the active to inactive ratio. A random seed of 44 was used to ensure reproducibility.

Table 2: Model performance charts for each model generated using Scikit-learn. These charts were generated using a testing set held out from model training, which consisted of 30% of each training set.

Classification Report (6:18)				
	Prec.	Rec.	F1	Supp.
Class 0	0.71	0.83	0.77	6
Class 1	0.00	0.00	0.00	2
Accuracy			0.62	8
Macro Avg	0.36	0.42	0.38	8
Weighted Avg	0.54	0.62	0.58	8

Classification Report (60:180)				
	Prec.	Rec.	F1	Supp.
Class 0	0.96	0.94	0.95	54
Class 1	0.84	0.89	0.86	18
Accuracy			0.93	72
Macro Avg	0.90	0.92	0.91	72
Weighted Avg	0.93	0.93	0.93	72

Classification Report (547:1541)				
	Prec.	Rec.	F1	Supp.
Class 0	0.95	0.96	0.95	463
Class 1	0.88	0.86	0.87	164
Accuracy			0.93	627
Macro Avg	0.92	0.91	0.91	627
Weighted Avg	0.93	0.93	0.93	627

The trained Random Forest model achieved an overall accuracy of 93% on the held-out test set (Table 2). Class-specific performance metrics revealed slightly lower performance for the minority (active) class: precision was 88% for active compounds versus 95% for inactive compounds, while recall was 86% for active compounds versus 96% for inactive compounds. Feature importance analysis revealed that molecular weight, logP, and rotatable bonds were the strongest predictors of activity among physicochemical descriptors, while Morgan fingerprint bits 578 and 694 (corresponding to specific structural motifs) contributed most to classification performance. The F1-scores were 87% and 95% for active and inactive classes, respectively. The area under the receiver operating characteristic curve (AUC-ROC) was 0.97, indicating strong discriminative ability (Figure 2).

The modest reduction in active compound recall (86%) reflects the challenge of identifying true positives within an imbalanced dataset,

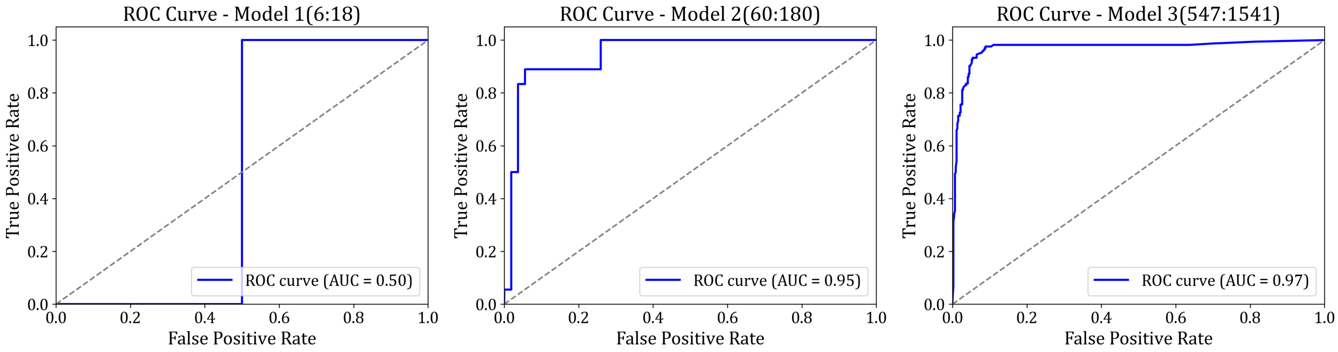


Figure 2: ROC curves of the three Random Forest models tested. The dotted line represents the random baseline. The random baseline shows the results of randomly guessing. Models with a higher area under the curve (AUC) are more capable of separating positive and negative examples correctly.

but remains acceptable for prioritizing candidates for experimental validation.

2.4 Model-Based Screening and Validation Docking

To identify novel EptA inhibitors, the trained Random Forest model was applied to screen the COCONUT (COlleCtion of Open Natural prodUcTs) database, which contains 695,119 natural product structures [3]. The entire database was filtered using Lipinski's Rule of Five, from which 5,000 compounds were randomly selected to create a computationally tractable screening set. Molecular descriptors and Morgan fingerprints were calculated for each compound using the same feature extraction protocol as described above. The Random Forest model predicted 166 compounds (3.3% of the screened set) as active based on probability scores above the classification threshold. These candidates were ranked by their prediction confidence scores to prioritize the most promising inhibitors for experimental validation. The model's prediction probability ranged from 94.5% to 50.1% for these active compounds.

This two-stage approach (machine learning pre-screening followed by high-accuracy docking validation) enabled efficient prioritization of the most promising candidates from a large chemical library. To validate the model's predictions and assess binding modes, the top 10 ranked compounds were selected for rigorous docking analysis (Table 3).

These compounds were subjected to molecular docking using the same AutoDock Vina protocol described previously (AutoDock4Zn forcefield, 60 × 60 × 60 grid centered on the Zn ion), but with increased exhaustiveness (32 vs. 10) to ensure thorough conformational sampling and more accurate binding pose prediction. The top-scoring pose for each compound was retained for binding mode analysis.

3 Results

3.1 Random Forest Model Performance and Dataset Optimization

To optimize model performance while minimizing computational cost, the effect of training dataset size on classification accuracy was systematically evaluated. Three training datasets with varying sizes were constructed, each maintaining a 1:3 active:inactive ratio: 6:18, 60:180, and 547:1,541 compounds. The smallest dataset (6:18), comprising the six parent inhibitors and three inactive analogs per inhibitor, achieved only 62% accuracy (based on a testing set created from 30% of the dataset), demonstrating the limitations of training on minimal data. Expanding the dataset to 60:180 (six parent inhibitors, 9 active analogs per inhibitor, and 30 inactive compounds per inhibitor) substantially improved accuracy to 93%, indicating that moderate dataset expansion was sufficient to achieve robust classification performance. The largest dataset (547:1,541) incorporated all compounds from the docking-based labeling procedure (described in Methods) plus 1,200 additional inactive compounds to maintain the 1:3 ratio. Notably, this dataset achieved the same 93% accuracy as the 60:180 set, suggesting that the model had reached optimal performance and that additional training data beyond $\tilde{2}40$ compounds provided diminishing returns in predictive accuracy.

To further assess model discrimination ability beyond accuracy alone, receiver operating characteristic (ROC) curves were generated for each dataset (Figure 2). ROC curves are created by finding the true positive rate (TPR) and false positive rate (FPR) at different probability threshold values, and the area under the ROC curve (AUC-ROC) can be used to compare two similarly balanced datasets. AUC-ROC values improved with dataset size: the 6:18 dataset achieved an AUC of 0.5, the 60:180 dataset reached 0.95, and the 547:1,541 dataset attained an AUC of 0.97. The AUC-ROC values for both larger datasets exceeded 0.90, indicating excellent discriminative performance and confirming the model's ability to reliably distinguish active from inactive compounds. The similarity in AUC between the two larger datasets further supports the conclusion that model performance plateaued beyond approximately 240 training compounds.

Although the 60:180 and 547:1,541 datasets achieved equivalent accuracy, the larger 547:1,541 dataset was selected for virtual screening to maximize chemical diversity and provide more robust representation of the active/inactive chemical space. The inclusion of structurally diverse inactive compounds was expected to improve the model's generalization to novel natural product scaffolds. Performance evaluation of the final Random Forest classifier on the

Ligand	SMILES	Affinity (kcal/mol)	Zn distance (Å)	Coordinating Group	Key Residue H-bond Interactions
ligand 1	CC(=O)O[C@@H]1c2c(ccc3ccc(=O)oc23)OC(C)(C)[C@@H]1OC(=O)CC(C)C	−7.345	3.84	aromatic ring	GLU114 (2), ASN106, THR110
ligand 2	COc1cc(CCc2ccc(O)cc2)cc(OC)c1	−6.384	4.58	aromatic ring	LYS328, GLU114, HIS465
ligand 3	COc1cc(O)cc(CCc2cc(C)c(OC)c(O)c2)c1	−7.228	2.66	methoxyl	LYS328, GLU114, GLY384, THR280
ligand 4	C/C=C(\C)C(=O)OC(C)(C)[C@@H]1Cc2cc3ccc(=O)oc3cc2O1	−7.262	2.97	methoxyl	LYS328, HIS465
ligand 5	CC(C)=CC(=O)O[C@@H]1c2cc3ccc(=O)oc3cc2O[C@H]1C(C)(C)O	−7.677	4.26	carbonyl	LYS328, GLU114, HIS465, THR280
ligand 6	COc1c2c(c(CC=C(C)C)c3oc(=O)ccc13)OC(C)(C)C=C2	−6.892	3.03	methoxyl	LYS328, GLU114
ligand 7	CC(C)=CC(=O)O[C@@H]1c2c(ccc3ccc(=O)oc23)OC(C)(C)[C@@H]1OC(=O)CC(C)C	−7.606	4.03	aromatic ring	GLU114 (2), ASN106, THR110
ligand 8	Cc1cc(C)c(NC(=O)COc2ccc3c(C)cc(=O)oc3c2)c(C)c1	−7.612	3.35	carbonyl	HIS465 (2), HIS453, THR280 (2)
ligand 9 (i1)	CC(=O)O[C@H]1[C@H](OC[C@H](C)C(C)=O)c2c(ccc3ccc(=O)oc23)OC1(C)C	−8.129	3.36	carbonyl	THR280 (2), ASN106, HIS465 (2)
ligand 9 (i2)	CC(=O)O[C@@H]1[C@H](OCC(C)=C(C)O)c2c(ccc3ccc(=O)oc23)OC1(C)C	−7.972	3.04	hydroxyl	THR280 (2), HIS465, VAL107, MET103
ligand 9 (i3)	C=C(O)[C@@H](C)CO[C@@H]1c2c(ccc3ccc(=O)oc23)OC(C)(C)[C@@H]1OC(C)=O	−7.739	3.07	hydroxyl	THR280 (2), HIS465
ligand 10 (i1)	CCCC(=O)c1c(O)c(CC=C(C)C)c(O)c2c([C@H](CC)OC(C)=O)cc(=O)oc12	−6.799	3.55	carbonyl	HIS465, ASN106, GLY384, GLU114
ligand 10 (i2)	CCC=C(O)c1c(O)c(CC=C(C)C)c(O)c2c([C@H](CC)OC(C)=O)cc(=O)oc12	−6.760	3.88	carbonyl	HIS465, ASN106, THR280, GLU114 (2), GLY384
ligand 10 (i3)	CCCC(=O)c1c2oc(O)cc([C@H](CC)OC(C)=O)c-2c(=O)c(CC=C(C)C)c1O	−7.154	2.34	hydroxyl	THR280, ALA281, ASN106 (2), LYS328, GLU114, HIS465
ligand 10 (i4)	CCC=C(O)c1c2oc(O)cc([C@H](CC)OC(C)=O)c-2c(=O)c(CC=C(C)C)c1O	−6.825	3.68	methyl	LYS328 (2), SER325, HIS383, THR110, ASN106
ligand 10 (i5)	CCCC(=O)c1c2oc(O)cc([C@H](CC)OC(C)=O)c-2c(O)c(CC=C(C)C)c1=O	−6.951	3.52	methyl	LYS328 (2), SER325, HIS465, ASN106, THR110, GLU114

Table 3: The docking results of the 10 ten natural products ranked by Random Forest confidence scores. Docking affinity was found through docking using the AutoDock4Zn forcefield run with an exhaustiveness of 32. Zn distance was measured from the ligand atom closest to the central Zn ion.

held-out test set (30% of data) yielded an overall accuracy of 93% (Table 2). Precision was 88% for active compounds and 95% for inactive compounds, while recall was 86% for active compounds and 96% for inactive compounds. The corresponding F1-scores were 87% for active compounds and 95% for inactive compounds, confirming decently balanced performance despite the class imbalance. Precision was 88% for active compounds and 95% for inactive compounds, while recall was 86% for active compounds and 96% for inactive compounds (Table 2). The modest reduction in active compound recall reflects the challenge of identifying minority class samples in an imbalanced dataset but remains acceptable for screening applications with subsequent validation steps.

Bit	Active Frequency	Inactive Frequency	Number of Compounds with Bit	Original Substructures SMILES
Bit_578	0.7514	0.3206	1222	CCCCC;CO;c=C(O)CCC;cO;cc(c)N(CC)CC
Bit_694	0.5210	0.1843	53	C=c(c)c;CC(C)CC[NH+];CCC;CCC=C(C)C;CCCC(N)=O
Bit_875	0.7642	0.3686	1444	C=C(O)CC(=c)O;CC(=O)NC(C)C;CNCCN;C[N@@H+](C)CC(=O)O;C[N@H+](C)CC(=O)O;ccc;ccc(cc)C(C)(C)C
Bit_580	0.5430	0.2291	679	C=C(C)C=CC;C=C(C)c(cc)c(c)O;CC(C)(C)O;CC(C)[NH3+];CC(CN)N(C)C;CCCC(C)(C)C;CCCC(C)(C)O;cc(C)c(C)c(=O)o;cc(c)CC(C)N;ccc(CC)c(c)O;coc
Bit_656	0.6088	0.3251	18	C=C(C)C=Cc;C=CC(c(c)c)C(O)O;CC=CC(C)O;CS[As](C)C;C[Si]
Bit_700	0.1517	0.0247	124	C[NH2+]C;cC(O)=CC(C)C;cc(c)C(OC)C(C)O;cc(c)n;ccc(C)c(c)O
Bit_356	0.9945	0.7657	157	CC(C)(C)CC(C)(C)O;CC(C)O;CCCC(C)N;CCN(C(=O)[O-])C(=O)[O-];C[C@@H](C)O;C[C@H](C)O;cC=NC;cc(C)o;ccc(Oc)c(c)N
Bit_147	0.0183	0.2440	295	C=CC;CC(=O)O;CC(c)(C)CCN;CC=CCC;COC;C[NH+](C)CC(=O)[O-];cC(C)(C)CCN
Bit_444	0.3163	0.1058	347	CCCCO;CNCCN;c=CCCC;cc(=O)o;cc(C)c(CC)c(c)c;cc(O)c(C)c(c)O;cc(c)OC(C)=O;cccc(C)n

Table 4: Summary of Morgan fingerprint bits and their associated substructures. Active and inactive frequencies are reported alongside the number of compounds to each bit.

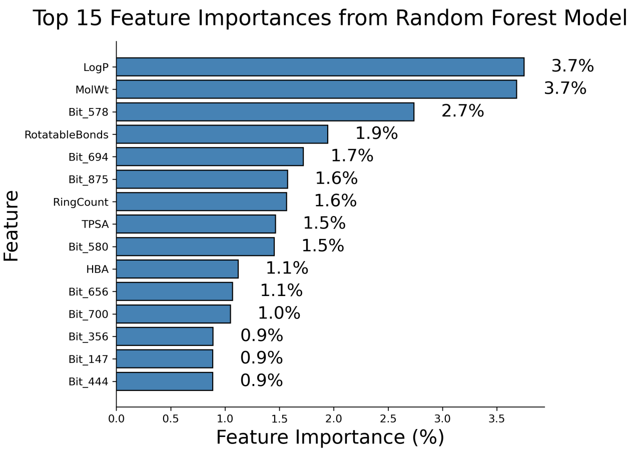


Figure 3: The top 15 feature importances from the 547:1541 Random Forest model trained using the full training dataset. Feature importance was normalized to 100%. Bits are part of the 1024-Bit Morgan Fingerprint feature (numbered Bit_0 to Bit_1023).

Among physicochemical descriptors, logP, molecular weight, and rotatable bonds were the most informative, collectively accounting for 3.75% of the model's predictive power (Figure 3).

Additionally, specific Morgan fingerprint bits (578 and 694) contributed significantly to classification, capturing structural features associated with EptA inhibition (Figure 3). This pattern indicates that both molecular fingerprints and drug-like properties are critical for distinguishing EptA inhibitors from inactive compounds. Based on these results, the 547:1,541 model was deployed to screen the COCONUT natural products database for novel EptA inhibitor candidates.

To provide chemical interpretability of these features, we analyzed the substructures associated with the most predictive Morgan fingerprint bits (Table 4). Bit 578 (active frequency: 75.1% vs. inactive: 32.1%) encodes hydroxyl groups, enols, and tertiary amines that may participate in zinc coordination or hydrogen bonding. Bit 694 (active: 52.1% vs. inactive: 18.4%) captures nitrogen-containing groups. Bit 147 serves as a negative predictor (active: 1.8% vs. inactive: 24.4%), while Bit 356 (active: 99.5%) appears to capture a core scaffold requirement. These associations link model predictions to specific chemical features relevant to EptA inhibition.

Following model validation, the trained Random Forest classifier was applied to screen 5,000 Lipinski-compliant compounds randomly sampled from the COCONUT natural products database. The model predicted 166 compounds (3.3% of the screened set) as active based on classification probability scores above 0.5. To prioritize compounds most likely to exhibit genuine EptA inhibitory activity, the top 10 candidates with the highest confidence scores (ranging from 0.85 to 0.98) were selected for validation through high-exhaustiveness molecular docking and detailed binding mode analysis (Table 3 lists compound identifiers and confidence scores). High-accuracy docking of the 10 top-ranked compounds confirmed strong predicted binding affinity across the majority of candidates. Docking scores ranged from -6.384 to -8.129 kcal/mol, with eight of the ten compounds achieving scores ≤ -7 kcal/mol (Table 3). For comparison, the known EptA inhibitor valnemulin yielded a docking score of -7.264 kcal/mol under identical conditions, indicating that the top natural product candidates exhibit comparable or superior predicted binding affinity. Compounds L9 and L10 (with multiple isomers/conformers analyzed for each) emerged as the most promising hits, with L9 isomer 1 achieving the strongest binding score of -8.129 kcal/mol, significantly higher than previous EptA inhibitors. These results validate the Random Forest model's ability to successfully identify potent EptA binders from a large chemical library and demonstrate that structurally diverse natural products can achieve effective active site engagement.

3.2 Binding Mode Analysis of Top-Ranked Inhibitors

To validate the model's predictions and characterize key binding interactions, the top 10 compounds ranked by prediction confidence were subjected to high-exhaustiveness docking and detailed interaction analysis. Visual inspection of the top-ranked docking poses revealed consistent binding modes across all candidates. All 10 compounds docked to the same binding pocket adjacent to the catalytic Zn^{2+} ion, confirming that the model successfully identified compounds targeting the same enzymatic active site. To assess potential zinc coordination, the distance from the Zn^{2+} ion to the nearest ligand heteroatom was measured using each ligand's binding pose (Table 3). Zn-ligand distances ranged from 2.34 to 4.58 Å, spanning both direct coordination geometry (< 3.5 Å) and extended hydrogen bonding networks (> 3.5 Å). Typical Zn^{2+} coordination bonds with range from 2.2-2.6 Å for direct interactions with elements such as oxygen or sulfur [9]. Compounds with distances in the range of 2.3-3.5 Å likely engage in direct or near-direct coordination with the metal center, while those with distances > 3.5 Å are stabilized primarily through other interactions with surrounding residues or solvent-mediated contacts. A clear structure-affinity relationship emerged: compounds with Zn distances of 3.0-3.5 Å exhibited the strongest binding affinities (docking scores ≤ -7.5 kcal/mol), while those with distances > 3.5 Å showed progressively weaker binding (scores: ≥ -7 kcal/mol) (Figure 4). Interestingly, compounds clustered into two binding mode categories: (1) direct Zn coordination mode (3.0-3.5 Å) with carbonyl or hydroxyl groups oriented toward the metal, and (2) peripheral binding mode (4.0-4.5 Å) stabilized by hydrogen bonds with catalytic residues. Both modes produced favorable binding scores, suggesting multiple viable strategies for EptA inhibition.

Detailed analysis of individual binding poses revealed that zinc-coordinating functional groups were key determinants of binding affinity (Table 3). Compound L9 (isomer 1) exhibited the strongest binding affinity (docking score: -8.129 kcal/mol), with a carbonyl oxygen positioned at an optimal 3.36 Å from the Zn^{2+} ion, consistent with favorable electrostatic interaction geometry. Despite also presenting carbonyl groups toward the metal center, L10 isomers 1 and 2 showed weaker binding (distances: 3.55 and 3.88 Å, respectively; scores: -6.799 and -6.76 kcal/mol), likely due to suboptimal coordination geometry outside the 3.0-3.5 Å favorable range. L9 isomers 2 and 3 also demonstrated strong binding (scores: -7.972 and -7.739 kcal/mol), mediated by hydroxyl groups positioned at 3.04 and 3.07 Å, respectively, from the zinc ion, indicating that both carbonyl and hydroxyl moieties can serve as effective Zn-coordinating pharmacophores. L10 isomer 3 exhibited a high but slightly reduced binding affinity (score: -7.154 kcal/mol) despite hydroxyl-Zn coordination, with a distance of 2.34 Å, which is below the optimal 3.0-3.5 Å range. This reduced affinity may result from unfavorable steric clashes or orbital overlap at very short distances, highlighting the importance of precise coordination geometry.

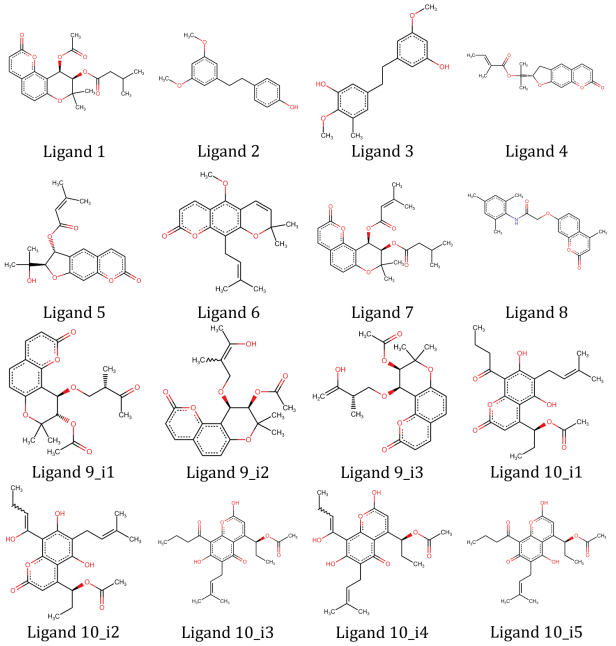


Figure 4: Diagrams of the 2D structures of the top 10 predicted inhibitors. The labels on each structure correspond with the labels in Table 3.

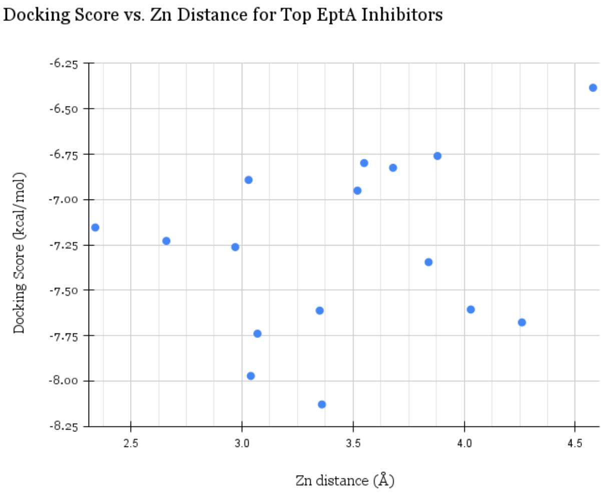


Figure 5: A graph of the Docking Score vs. Zn distance of the top 10 predicted inhibitors. Each point represents one of the inhibitors, and the lower docking scores indicate stronger binding affinity

Beyond zinc coordination, additional stabilizing interactions were observed across the top candidates. Hydrogen bonding networks with catalytic residues such as ASN106, THR110, GLU114, THR280, SER325, LYS328, HIS383, GLY384, HIS453, HIS465 were present in all inhibitors. Compounds with hydrogen bonding with residues spaced throughout the binding pocket exhibited high binding affinities, such as ligand 10 (i3) and ligand 3. Additionally, compounds with hydrogen bonds in the direction of the Zn ion exhibited the highest binding affinities, such as ligand8, ligand 9 (i1), and ligand 9 (i2) (scores: -7.612, -8.129, -7.972). The most common hydrogen bonding residues were GLU114 and HIS465, which formed hydrogen bonds with 10 and 11 out of 16 inhibitors, respectively. Some compounds such as ligand 9 (i2) and ligand 10 (i3) formed hydrophobic contacts with MET103, VAL107, and ALA281, further contributing to binding stability. Notably, the most potent inhibitors (L9 variants) combined optimal Zn coordination distance with complementary interactions to THR280, HIS465, and ASN106, suggesting a multi-point binding mechanism. Comparison to the known inhibitor Valnemulin revealed similarities in binding mode. The top natural product candidates showed comparable or superior docking scores (natural products: -6.384 to -8.129 kcal/mol vs. Valnemulin: -7.264 kcal/mol). Valnemulin also formed hydrogen bonds with similar key active site residues, such as GLU114 and SER325. This suggests that natural product scaffolds can achieve effective EptA inhibition through similar mechanisms.

4 Discussion

This study identified novel natural product inhibitors of EptA through combined machine learning and structure-based virtual screening. The top-ranked compounds represent promising candidates for EptA inhibition based on both their high model confidence scores (0.76-0.95) and favorable docking profiles (binding energies: -6.384 to -8.129 kcal/mol)(Table 3). Several of these molecules displayed structural features, including multiple aromatic rings, hydrogen bond donors and acceptors, and heteroatoms positioned to coordinate with the zinc ion in the catalytic site, consistent with previously reported inhibitors, such as valnemulin and other small molecules that disrupt EptA-mediated lipid A modification. Importantly, the identified natural products represent chemically distinct scaffolds from known synthetic inhibitors, potentially offering advantages such as reduced toxicity, improved bioavailability, or novel mechanisms of action; benefits commonly associated with natural product-derived therapeutics.

Detailed binding mode analysis revealed that the top-ranked compounds adopted consistent binding poses within the EptA active site, with all 10 candidates docking to the same zinc-proximal cavity. Key stabilizing interactions included direct or near-direct coordination with the catalytic Zn^{2+} ion through carbonyl and hydroxyl groups (distances: 2.3-4.6 Å) and hydrogen bonding networks with catalytic residues such as HIS453, GLU114, ASN106, THR280, and THR110. Notably, compounds with optimal Zn coordination distances (3.0-3.5 Å) exhibited the strongest binding affinities, highlighting the importance of precise metal-ligand geometry. The convergence of these interaction patterns across structurally diverse scaffolds validates the Random Forest model's ability to identify compounds based on functionally relevant pharmacophoric features rather than superficial structural similarity. This demonstrates that the machine learning approach successfully captured the key determinants of EptA binding.

Analysis of the most predictive Morgan fingerprint bits revealed chemical features consistent with EptA's catalytic mechanism (Table 4). Bits encoding hydroxyl groups, enols, and tertiary amines were enriched in active compounds. These functional groups can serve as electron donors in zinc coordination or form hydrogen bonds with active-site residues such as GLU114 and HIS465. The enrichment of nitrogen-containing moieties aligns with known zinc-binding pharmacophores in metalloenzyme inhibitors. Conversely, substructures associated with Bit 147 were depleted in active compounds, suggesting these chemical features interfere with binding. The near-universal presence of Bit 356 in active compounds indicates a conserved scaffold requirement. These findings support the model's ability to learn chemically meaningful patterns rather than arbitrary correlations.

While these computational findings are promising, several important limitations must be considered when interpreting the results. First, this study employed entirely computational methods, and the identified compounds remain experimentally unvalidated. While computational screening provides efficient prioritization, it cannot replace empirical testing. Docking scores and machine learning predictions provide useful prioritization but cannot fully capture dynamic biological factors such as membrane permeability, metabolic stability, or potential off-target effects. Additional factors not captured by these methods include protein flexibility, solvent effects, entropic contributions to binding, and the influence of the cellular environment on compound efficacy.

Second, the training dataset relied on docking-derived activity labels rather than experimental IC_{50} values, which may introduce systematic bias if the scoring function consistently over- or underestimates binding affinity for certain chemical classes. Although the AutoDock4Zn forcefield improves accuracy for zinc-dependent enzymes compared to standard forcefields, it remains a computational approximation that may not fully capture quantum mechanical effects or polarization in metal-ligand interactions. Orthogonal methods, such as molecular dynamics simulations or ligand-based pharmacophore modeling, could provide independent validation of the top candidates. Additionally, model performance for active compounds was slightly lower than for inactive compounds, indicating a potential underrepresentation of certain structural classes in the training data. This class imbalance (1:3 active:inactive ratio) may have reduced the model's sensitivity to rare but potentially important structural features present in highly active compounds. Future iterations could benefit from balanced datasets or alternative sampling strategies such as SMOTE (Synthetic Minority Over-sampling Technique). A systematic comparison of feature types, such as Morgan fingerprints alone versus physicochemical descriptors, could also optimize the precision-recall trade-off for the active class.

Third, the study screened only 5,000 compounds from the COCONUT database (<1% of the full library), meaning potentially superior candidates may remain unidentified. A comprehensive screen of the entire database would be valuable but was beyond the computational scope of this initial study. The Random Forest model can process all 695,000 compounds in minutes, and GPU-accelerated tools such as AutoDock-GPU would enable validation of a larger candidate set in future work.

Several experimental and computational approaches could build upon these findings. Most immediately, in vitro enzymatic inhibition assays using purified recombinant EptA would confirm predicted inhibitory activity and establish IC_{50} values for the top candidates. Subsequently, whole-cell assays in polymyxin-resistant bacterial strains (e.g., E. coli, K. pneumoniae, or A. baumannii expressing EptA or MCR variants) could assess whether these compounds restore colistin susceptibility and determine minimum inhibitory concentrations (MICs). For lead optimization, co-crystal structures of EptA bound to the most potent inhibitors would provide atomic-resolution insight into binding modes, enabling structure-guided medicinal chemistry to improve potency, selectivity, and drug-like properties.

ADMET (absorption, distribution, metabolism, excretion, toxicity) profiling is essential for translational development. Natural products often present challenges, including poor aqueous solubility, limited membrane permeability, and metabolic instability. Computational tools such as SwissADME, pkCSM, or ProTox-II can predict these properties and identify potential toxicity liabilities before experimental testing. Synthetic accessibility scores would further prioritize compounds based on the feasibility of obtaining sufficient quantities. Several of our top hits are naturally occurring compounds that may be available from commercial sources or have established isolation protocols.

To improve the computational pipeline, expanding the training dataset with experimentally validated actives and inactives would reduce docking-dependent bias and enhance model generalization. Incorporating additional machine learning architectures (e.g., graph neural networks) or ensemble methods could further improve prediction accuracy [5]. GNNs can learn directly from molecular graphs and may better represent complex substructures and three-dimensional conformational information, though the interpretability of fingerprint-based features was valuable for this study. Finally, expanding virtual screening to the full COCONUT database (~695,000 compounds) or other natural product libraries could identify additional candidates with diverse scaffolds and mechanisms.

5 Conclusions

This study successfully employed an integrated computational approach combining Random Forest machine learning and zinc-aware molecular docking to identify natural product–derived inhibitors of EptA, a key enzyme mediating polymyxin resistance in multidrug-resistant Gram-negative bacteria. A systematically optimized Random Forest classifier achieved 93% accuracy and AUC-ROC > 0.90 and was used to screen 5,000 COCONUT natural products, yielding 166 predicted actives from which the top 10 underwent high-exhaustiveness docking. These candidates showed strong binding affinities (-6.384 to -8.129 kcal/mol) and consistent binding modes with optimal zinc coordination and key catalytic interactions, demonstrating the effectiveness of integrating machine learning with structure-based methods for rapidly and cost-effectively prioritizing diverse natural products. Importantly, the identified compounds represent chemically distinct scaffolds from existing synthetic EptA inhibitors, potentially offering advantages such as novel mechanisms of action, reduced toxicity, or improved pharmacological properties, and by targeting EptA, this approach offers a viable strategy to restore polymyxin efficacy against multidrug-resistant pathogens.

Experimental validation now represents the crucial next step to advance these predictions toward therapeutic development. In vitro enzymatic assays with recombinant EptA will establish IC_{50} values and confirm inhibitory activity, followed by whole-cell susceptibility testing in polymyxin-resistant clinical isolates to determine whether these compounds restore colistin sensitivity through MIC measurements. Co-crystal structures will validate binding modes and guide structure-based optimization, while parallel ADMET profiling and medicinal chemistry efforts will address liabilities in absorption, metabolism, and toxicity. Ultimately, this workflow will advance the most promising candidates toward preclinical development, and beyond EptA inhibition, the integrated pipeline can be adapted to other metalloenzyme targets and resistance mechanisms, providing a generalizable strategy for accelerating antibacterial drug discovery in the face of escalating antimicrobial resistance.

References

[1] Branda, F., Scarpa, F.: Implications of artificial intelligence in addressing antimicrobial resistance: Innovations, global challenges, and healthcare's future. Antibiotics p. 502 (2024). https://doi.org/10.3390/antibiotics13060502

[2] Breiman, L.: Random forests. Machine Learning pp. 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[3] Chandrasekhar, V., Rajan, K., Kanakam, S.R.S., Sharma, N., Weißenborn, V., Schaub, J., Steinbeck, C.: Coconut 2.0: a comprehensive overhaul and curation of the collection of open natural products database. Nucleic Acids Research pp. D634–D643 (2025). https://doi.org/10.1093/nar/gkae1063

[4] Cui, X.D., Zhang, J.K., Sun, Y.W., Yan, F.B., Zhao, J.F., He, D.D., Pan, Y.S., Yuan, L., Zhai, Y.J., Hu, G.Z.: Synergistic antibacterial activity of baicalin and edta in combination with colistin against colistin-resistant salmonella. Poultry Science p. 102346 (2023). https://doi.org/10.1016/j.psj.2022.102346

[5] El-Behery, H., Attia, A.F., El-Fishawy, N., Torkey, H.: An ensemble-based drug-target interaction prediction approach using multiple feature information with data balancing. Journal of Biological Engineering p. 21 (2022). https://doi.org/10.1186/s13036-022-00296-7

[6] El-Sayed Ahmed, M.A.E.G., Zhong, L.L., Shen, C., Yang, Y., Doi, Y., Tian, G.B.: Colistin and its role in the era of antibiotic resistance: an extended review (2000–2019). Emerging Microbes & Infections pp. 868–885 (2020). https://doi.org/10.1080/22221751.2020.1754133

[7] Huang, J., Zhu, Y., Han, M.L., Li, M., Song, J., Velkov, T., Li, C., Li, J.: Comparative analysis of phosphoethanolamine transferases involved in polymyxin resistance across 10 clinically relevant gram-negative bacteria. International Journal of Antimicrobial Agents pp. 586–593 (2018). https://doi.org/10.1016/j.ijantimicag.2017.12.016

[8] Jen, F.E.C., El-Deeb, I.M., Zalucki, Y.M., Edwards, J.L., Walker, M.J., von Itzstein, M., Jennings, M.P.: A drug candidate for alzheimer's and huntington's disease, pbt2, can be repurposed to render neisseria gonorrhoeae susceptible to natural cationic antimicrobial peptides. Journal of Antimicrobial Chemotherapy pp. 2850–2853 (2021). https://doi.org/10.1093/jac/dkab291

[9] Laitaoja, M., Valjakka, J., Jänis, J.: Zinc coordination spheres in protein structures. Inorganic Chemistry pp. 10983–10991 (2013)

[10] Mlynarcik, P., Kolar, M.: Molecular mechanisms of polymyxin resistance and detection of mcr genes. Biomedical Papers pp. 28–38 (2019). https://doi.org/10.5507/bp.2018.070

[11] Mullally, C., Stubbs, K.A., Thai, V.C., Anandan, A., Bartley, S., Scanlon, M.J., Jarvis, G.A., John, C.M., Lim, K.Y.L., Sullivan, C.M., Sarkar-Tyson, M., Vrielink, A., Kahler, C.M.: Novel small molecules that increase the susceptibility of neisseria gonorrhoeae to cationic antimicrobial peptides by inhibiting lipid a phosphoethanolamine transferase. Journal of Antimicrobial Chemotherapy pp. 2441–2447 (2022). https://doi.org/10.1093/jac/dkac204

[12] RDKit Community: Rdkit: Open-source cheminformatics (release 2025.09.1) (2025). https://doi.org/10.5281/zenodo.17232453

[13] Salam, M.A., Al-Amin, M.Y., Salam, M.T., Pawar, J.S., Akhter, N., Rabaan, A.A., Alqumber, M.A.A.: Antimicrobial resistance: A growing serious threat for global public health. Healthcare p. 1946 (2023). https://doi.org/10.3390/healthcare11131946

[14] Santos-Martins, D., Forli, S., Ramos, M.J., Olson, A.J.: Autodock4 zn: An improved autodock force field for small-molecule docking to zinc metalloproteins. Journal of Chemical Information and Modeling pp. 2371–2379 (2014). https://doi.org/10.1021/ci500209e

[15] Schrödinger, LLC: The pymol molecular graphics system (version 3.0)

[16] Suardíaz, R., Lythell, E., Hinchliffe, P., van der Kamp, M., Spencer, J., Fey, N., Mulholland, A.J.: Catalytic mechanism of the colistin resistance protein mcr-1. Organic & Biomolecular Chemistry pp. 3813–3819 (2021). https://doi.org/10.1039/D0OB02566F

[17] Thai, V.C., Stubbs, K.A., Sarkar-Tyson, M., Kahler, C.M.: Phosphoethanolamine transferases as drug discovery targets for therapeutic treatment of multidrug resistant pathogenic gram-negative bacteria. Antibiotics p. 1382 (2023). https://doi.org/10.3390/antibiotics12091382

[18] Word, J.M.: Reduce (version 3.16)

[19] Xie, S., Li, L., Zhan, B., Shen, X., Deng, X., Tan, W., Fang, T.: Pogostone enhances the antibacterial activity of colistin against mcr-1-positive bacteria by inhibiting the biological function of mcr-1. Molecules p. 2819 (2022). https://doi.org/10.3390/molecules27092819

[20] Xu, C., Zhang, Y., Ma, L., Zhang, G., Li, C., Zhang, C., Li, Y., Zeng, X., Li, Y., Dong, N.: Valnemulin restores colistin sensitivity against multidrug-resistant gram-negative pathogens. Communications Biology p. 1122 (2024). https://doi.org/10.1038/s42003-024-06805-2

[21] Xu, Y., Wei, W., Lei, S., Lin, J., Srinivas, S., Feng, Y.: An evolutionarily conserved mechanism for intrinsic and transferable polymyxin resistance. mBio (2018). https://doi.org/10.1128/mBio.02317-17

[22] Zhou, Y., Liu, S., Wang, T., Li, H., Tang, S., Wang, J., Wang, Y., Deng, X.: Pterostilbene, a potential mcr-1 inhibitor that enhances the efficacy of polymyxin b. Antimicrobial Agents and Chemotherapy (2018). https://doi.org/10.1128/AAC.02146-17

[23] Zhou, Y., Wang, J., Guo, Y., Liu, X., Liu, S., Niu, X., Wang, Y., Deng, X.: Discovery of a potential mcr-1 inhibitor that reverses polymyxin activity against clinical mcr-1-positive enterobacteriaceae. Journal of Infection pp. 364–372 (2019). https://doi.org/10.1016/j.jinf.2019.03.004

Received 23 December 2025; Accepted 30 January 2026