

IntelliPhish: An Explainable DistilBERT-Based Phishing Email Detection System with URL Deep Analysis and Live Threat Analytics

Kandela Prakash¹, Dr. A. Damodaram²

¹MCA Student, Department of Computer Science

²Senior Professor, Department of Computer Science; Rector

University College of Engineering, Science & Technology,

Jawaharlal Nehru Technological University Hyderabad, Hyderabad, India

Abstract - Phishing email continues to be one of the most persistent and financially damaging categories of cyberattack, and traditional keyword-based spam filters and classical machine-learning classifiers routinely fail against professionally worded messages while offering the end user no explanation for a verdict. This paper presents IntelliPhish, a self-contained Flask web application that fine-tunes a DistilBERT transformer on a cleaned, merged corpus of phishing and legitimate emails and pairs the resulting classifier with SHAP (SHapley Additive exPlanations) word-level attribution, a URL Deep Analysis module, and a curated threat-keyword extractor, so that every verdict arrives with corroborating, human-readable evidence rather than a bare label.

The fine-tuned DistilBERT model reaches 97.4% accuracy and a 97.3 F1-score on a held-out test split, the highest of five benchmarked models, while running at roughly one-third the inference latency (38 ms vs. 112 ms) and close to half the disk footprint (256 MB vs. 438 MB) of full BERT-base, and comfortably ahead of Logistic Regression, Random Forest and Naive Bayes baselines trained on the same data. The URL Deep Analysis module scores embedded links on IP-based addressing, excessive length, brand typosquatting via Levenshtein distance, WHOIS domain age and unencrypted HTTP usage, while a 27-term threat lexicon flags urgency and manipulation language independently of the transformer's own decision. Every scan is persisted to SQLite, driving an analytics dashboard and a five-model benchmark page, with an optional read-only Gmail OAuth 2.0 integration allowing the system to scan a live inbox using the identical prediction pipeline used for pasted text.

An analysis of 97 real scans recorded during live use shows a 35.1%/64.9% phishing/legitimate split, an average confidence of 97.79%, and an average URL risk score of 44.64 out of 100, with the keyword 'click' detected 181 times, more than any other threat term. Structured testing, including deliberate failure injection into the SHAP explainer and the SQLite writer, confirms that the system degrades gracefully rather than failing hard when an individual analysis module is unavailable. These results demonstrate that a compact, distilled transformer combined with an independent, structural URL-risk layer and a transparent explanation mechanism can deliver phishing detection that is simultaneously accurate, interpretable, auditable, and deployable without specialised hardware.

Key Words: *Phishing Detection, DistilBERT, Transformer Models, Explainable AI, SHAP, Natural Language Processing, URL Analysis, Cybersecurity, Gmail API, Email Security.*

1. INTRODUCTION

Phishing email remains one of the most persistent and financially damaging categories of cyberattack, typically arriving as a message that impersonates a trusted sender in order to trick a recipient into revealing credentials, transferring funds, or installing malware. Because phishing relies on social engineering rather than a technical exploit, it bypasses firewalls and antivirus software entirely and instead targets human judgement directly. Industry incident reports have consistently placed email-based phishing among the top initial-access vectors used in confirmed data breaches, which is why detection at the point of reading, rather than only at the network perimeter, remains an active and necessary area of applied research.

Keyword-based spam filters and rule-based heuristics, historically the first line of defence, struggle against professionally worded, grammatically correct phishing emails that avoid obvious red-flag words. Classical machine-learning classifiers trained on bag-of-words or TF-IDF features improve on this but still miss the deeper contextual and

semantic cues, such as tone of urgency or impersonation of authority, that a human analyst would notice. Transformer-based language models, by contrast, model the relationships between words across an entire email, making them considerably more capable of recognising these subtler social-engineering patterns. A bare classification label, however, is of limited use to a user who does not know why a message was flagged or how much to trust the verdict — the value of an assistive system lies as much in what is shown after the prediction as in the prediction itself.

This work presents IntelliPhish, a system that fine-tunes a DistilBERT transformer specifically for the phishing/legitimate distinction and pairs it with SHAP explainability, an independent URL Deep Analysis module, and a persistent scan-history dashboard, so that the system's decisions are not just accurate but interpretable and auditable by the end user. The system is deliberately engineered as a single, self-contained Flask application rather than a distributed pipeline, so that it can be installed, demonstrated and evaluated end-to-end on a single laptop-class machine — a constraint that shaped several design decisions, particularly the choice of a distilled transformer over a full-sized one.

The specific objectives of this work are fourfold: (i) fine-tune a DistilBERT transformer on a cleaned, merged corpus of phishing and legitimate emails for binary text classification; (ii) integrate SHAP explainability and an independent URL Deep Analysis module that scores embedded links on IP-based addressing, typosquatting and domain age; (iii) benchmark the fine-tuned model against BERT-base and classical baselines on accuracy, precision, recall, F1-score, inference speed and model size; and (iv) assemble these components into a Flask application with a persistent SQLite scan history, an analytics dashboard, a benchmark page, and an optional read-only Gmail integration.

In summary, this paper makes three contributions. First, it fine-tunes and benchmarks a DistilBERT-based phishing classifier against BERT-base and three classical baselines on a carefully cleaned, de-duplicated, multi-source corpus, and shows that the distilled transformer matches or exceeds full BERT-base accuracy at roughly one-third the latency. Second, it combines SHAP word-level explanation with an independently computed, structural URL risk score and a curated threat-keyword lexicon, so that every verdict is corroborated by three separate signals rather than one opaque number, addressing a gap that recurs across the literature reviewed in Section 2. Third, it packages the whole pipeline, together with a live analytics dashboard, a five-model benchmark page and an optional Gmail integration, into a single Flask application whose structured testing (Section 5) confirms graceful degradation under partial component failure rather than only classification accuracy in isolation.

2. LITERATURE SURVEY

Jamal et al. [1] fine-tuned transformer models, including DistilBERT and RoBERTa, to separate phishing, spam and legitimate (ham) emails, and showed that fine-tuning a transformer on domain-specific email data substantially outperforms classical machine-learning baselines — a finding that directly supports the choice of a fine-tuned transformer as the classification backbone for this project. The study, however, targets a three-class problem rather than the binary phishing/legitimate distinction used in most operational filters, adds no explainability layer, and does not analyse URL-based features alongside the text classifier.

Uddin et al. [2] paired a fine-tuned transformer with an explainable-AI layer so that predictions are not left as an unexplained label, a design goal IntelliPhish shares, though the two systems differ in how they get there: where that paper interprets predictions through attention-based visualisation and relies on a full RoBERTa architecture, IntelliPhish instead uses SHAP's model-agnostic additive attribution on a distilled model to keep inference lightweight. The paper's heavier RoBERTa configuration is considerably larger and slower at inference than a fully distilled model, and it does not evaluate robustness against typosquatted or obfuscated URLs embedded in phishing emails.

Alhuzali et al. [3] took a methodological approach, stress-testing a wide pool of ML and deep-learning architectures — spanning classical algorithms up through fine-tuned BERT and RoBERTa — against nearly a dozen different phishing-email sources merged into one evaluation set, showing that transformer-based models retain their accuracy edge even once dataset-specific quirks are averaged out. This merge-many-sources-into-one-corpus strategy, aimed at reducing single-dataset bias, is the specific idea IntelliPhish borrowed when assembling its own training data (Section 4.2), rather than training on any single public dataset in isolation. The study nonetheless stops at comparative evaluation without proposing an integrated, deployable detection system, and does not evaluate a distilled transformer variant.

Uddin, Biswas et al. [4] focused less on detection accuracy and more on explanation depth, layering SHAP explanations at two levels — global feature importance across the whole model and local, prediction-by-prediction waterfall and beeswarm plots — for a gradient-boosted phishing-website classifier. It was this two-level, global-and-local framing of SHAP, rather than the phishing-website classification task itself, that shaped IntelliPhish's own choice of SHAP over alternatives such as LIME for explaining individual email verdicts; the paper, however, targets structured/tabular URL features rather than natural-language email content and applies SHAP to boosted-tree feature importance rather than token-level text attributions.

Mia et al. [5] examined whether URL-based phishing-detection features, such as domain length, IP-address presence and URL-shortener usage, remain reliable indicators across different datasets, using SHAP to evaluate feature importance and stability. Its findings on which URL-structural features are the most dataset-independent — IP-based addressing, unusually long or complex URLs, and suspicious domain patterns — directly shaped the specific checks implemented in IntelliPhish's URL Deep Analysis module, although the study focuses purely on URL features, does not evaluate WHOIS-based domain-age signals, and does not incorporate email body text.

Sanh et al. [6] introduced DistilBERT itself, a distilled version of BERT that retains roughly 97% of its language-understanding performance while being about 40% smaller and 60% faster at inference — precisely the efficiency profile required for a responsive, CPU-only web classifier and the backbone adopted in this work. Lundberg and Lee [7] introduced SHAP, a unified, game-theoretic framework for attributing a model's output to its input features, which this project applies at the token level to generate the word-level phishing/legitimate explanations shown on the result page. Devlin et al. [8] introduced BERT itself, the bidirectional transformer architecture from which DistilBERT is distilled and against which this project's fine-tuned model is benchmarked.

Taken together, the five application-focused studies surveyed above show a consistent pattern: work that improves detection accuracy through transformers ([1], [3]) generally does not add explainability or URL analysis; work that adds explainability ([2], [4]) is generally applied either to a larger, slower transformer or to a non-text website-classification task; and work that focuses on URL features specifically ([5]) does not combine them with a text classifier at all. No single reviewed study combines a distilled transformer, SHAP-based word-level text explanation, and independent structural URL analysis, including WHOIS domain age, in one deployable pipeline — the combined gap that IntelliPhish's design, described from Section 3 onward, is aimed at closing.

3. SYSTEM ANALYSIS

3.1 Existing Approaches and Their Limitations

Traditional email filters flag messages containing specific keywords, such as ‘urgent’ or ‘verify your account’, or matching known phishing templates. These are fast and fully interpretable but are easily evaded by professionally worded phishing emails that avoid obvious trigger words, generate no risk assessment of the URLs a message contains, and require continuous manual maintenance as attackers adapt their wording. Classical machine-learning classifiers — Logistic Regression, Random Forest, Naive Bayes and Support Vector Machines trained on bag-of-words or TF-IDF features — are computationally cheap and reasonably accurate, but represent text as an unordered collection of tokens, discarding word order and context. Recent transformer-based systems (Section 2) substantially improve detection of subtle social-engineering language, but most published systems return only a label and confidence score, without any per-word explanation, and few integrate a companion analysis of the URLs embedded in the email. Across all three approaches, three gaps recur consistently: label-only output with no explanation; no independent structural risk assessment of embedded URLs; and no persistent, queryable scan history from which phishing trends or keyword frequency can be observed over time.

3.2 Feasibility

The proposed system is technically feasible: it is built entirely on open-source, widely documented technologies — Python, PyTorch, Hugging Face Transformers, SHAP, scikit-learn, Flask and SQLite — all of which run on standard consumer hardware without a GPU, since DistilBERT is small enough for inference on CPU within an acceptable response time. It is economically feasible because every core framework is free and open source, the DistilBERT base checkpoint carries no licensing cost, and the only optional dependency — Google Cloud Console registration for Gmail

OAuth credentials — is free at the request volumes this project requires. Operationally, the system requires no training to use: a pasted email, a few seconds of processing, and a result page showing verdict, confidence, SHAP explanation, URL risk and keywords are enough for a non-technical user, while the dashboard and benchmark pages serve the more curious or technical user without being necessary for everyday use.

Table -1: Existing Systems Versus the Proposed System

Aspect	Existing Systems	Proposed IntelliPhish
Output	Label only, no explanation	Verdict + confidence + SHAP word-level explanation
URL risk assessment	Absent in text classifiers	Independent structural risk score (IP, typosquat, WHOIS age, HTTP)
Explainability	Black-box or attention-visualisation only	Model-agnostic SHAP token attribution
Scan history	Not tracked	Persisted to SQLite; drives dashboard and benchmark page
Live inbox access	Not supported	Optional read-only Gmail OAuth 2.0 integration

4. PROPOSED SYSTEM AND METHODOLOGY

The proposed system is built around five cooperating stages: a fine-tuned DistilBERT classifier, a SHAP explainer, a URL Deep Analysis module, a threat-keyword extractor, and a SQLite-backed history and analytics layer, all served by a single Flask web application.

4.1 System Architecture

Incoming email text, either pasted directly or retrieved via the Gmail API, is first validated and length-guarded (capped at 10,000 characters) before being tokenised and passed through the fine-tuned DistilBERT classifier. The resulting phishing probability is compared against a 35% ‘paranoid mode’ decision threshold, deliberately more sensitive than the conventional 50% cut-off, biasing the system toward flagging borderline emails as phishing rather than missing them. In parallel, the same email text is passed independently to the SHAP explainer, the URL Deep Analysis module and the keyword extractor, so that a slow or failing component in one branch cannot delay or corrupt the output of another. All outputs are combined into a single result object, persisted to the SQLite scans table, and rendered on the result page. Fig -1 summarises this pipeline.

Figure 1: System Architecture of IntelliPhish v3

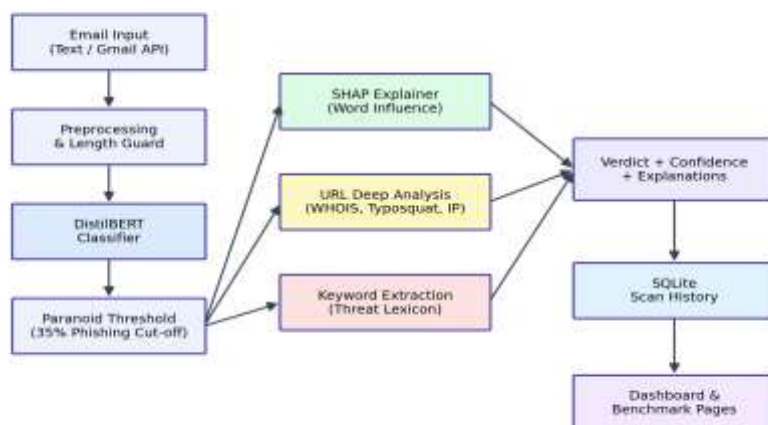


Fig -1: Architecture of the IntelliPhish Phishing Detection System

Fig -2 shows the Level 0 data-flow view of the same system: email text enters from the user (or from Gmail), is processed by the single logical prediction process, and a verdict together with its supporting evidence is returned. Internally, this process expands into input validation, DistilBERT inference, the parallel SHAP/URL/keyword analysis stage, and the save-and-render stage.

Figure 2: Level 0 DFD



Fig -2: Level 0 Data Flow Diagram of IntelliPhish

4.2 Dataset Description and Preprocessing

Training data is merged from CSV/JSONL files spanning several public phishing and legitimate-email sources, each loaded with a UTF-8-first, Latin-1-fallback decoder, with subject, body and label columns matched heuristically so differently formatted source files can be merged without manual renaming. Labels are normalised through a mapping function that maps text and numeric variants ('phishing', 'spam', 'fraud', '1') to the positive class and ('legitimate', 'ham', 'benign', '0') to the negative class, dropping any row whose label cannot be confidently mapped rather than guessing. Rows shorter than 15 characters are dropped as corrupted fragments, and rows sharing the same 60-character opening or closing phrase — typically mailing-list footers or boilerplate — are capped at 15 per template, in addition to removing exact duplicate rows, so that a single scraped archive of near-identical footer variants cannot quietly dominate the loss function and bias the model toward memorising boilerplate rather than genuine phishing language.

Table -2: Training Dataset Summary

Parameter	Value
Minimum text length retained	15 characters
Maximum rows per repeated template (prefix/suffix)	15
Template comparison window	60 characters
Train / test split	80% / 20%, stratified by label
Tokenizer max length (training)	256 tokens
Base model	distilbert-base-uncased, 2-label classification head

4.3 DistilBERT Fine-Tuning

The merged, cleaned corpus is split 80/20 into train and test sets, stratified by label. Text is tokenised with truncation and padding to a maximum length of 256 tokens, chosen because typical emails in the dataset average 700–2,600 characters and a more common 128-token limit was found to truncate most of the email body away. Fine-tuning runs for three epochs using the Hugging Face Trainer, with batch size and mixed precision automatically adjusted for GPU availability, and the best checkpoint is selected by F1-score rather than raw accuracy — a deliberate choice, since optimising for accuracy alone on an imperfectly balanced corpus could quietly reward a checkpoint that favours the majority class, whereas F1-score weights precision and recall equally. Final accuracy, precision, recall and F1-score are written to a benchmark file so the live benchmark page can display real training metrics rather than static placeholders.

4.4 SHAP Explainability Integration

At inference time, the deployed application loads the fine-tuned model once at startup and, when the SHAP library is available, builds a SHAP explainer wrapped around a prediction function that tokenises input text, padded/truncated to

128 tokens for inference speed, and returns softmax probabilities. For every analysed email, SHAP produces a word-influence visualisation highlighting which words pushed the verdict toward phishing (rendered in red) or legitimate (rendered in blue), shown directly on the result page. If SHAP is unavailable, the explanation section degrades to a plain-text notice rather than causing the request to fail.

4.5 URL Deep Analysis

All URLs are extracted from the email text with a regular expression and, for up to ten URLs per email (capped to bound WHOIS latency), each is checked for: (i) an IP-based address in place of a domain name, contributing the largest single weight (+50) as a critical red flag; (ii) unusually long URLs, over 75 characters, or domains with more than three subdomain levels (+20); (iii) typosquatting of sixteen well-known brand domains, including PayPal, Amazon, Google, Microsoft, Apple and several major banks, detected via Levenshtein edit-distance of two or less against the domain's base name (+40); (iv) domain age via WHOIS, flagging domains younger than 30 days as high risk (+35); and (v) unencrypted HTTP usage (+15). Each finding contributes additively to a cumulative risk score, capped at 100 regardless of how many factors trigger simultaneously.

4.6 Keyword Extraction, Dashboard, Benchmark and Gmail Integration

A curated set of 27 urgency and manipulation-related terms — including ‘urgent’, ‘verify’, ‘suspended’, ‘click’, ‘password’ and ‘unauthorized’ — is matched against the lower-cased, word-tokenised email text, giving a lightweight, human-readable summary of manipulation language independent of the transformer's own decision. Every scan is saved to SQLite immediately after prediction, and the dashboard route aggregates the most recent scans into a 14-day phishing-versus-legitimate trend chart and a top-10 keyword-frequency chart, alongside headline statistics. The benchmark route displays five models side by side — DistilBERT, BERT-base, Logistic Regression, Random Forest and Naive Bayes — overwriting the DistilBERT row with real metrics when available. The optional Gmail integration implements the standard OAuth 2.0 authorization-code flow under the read-only gmail.readonly scope, then analyses the user's most recent inbox messages through the identical prediction pipeline used for pasted text, ensuring consistent analysis logic regardless of email source.

4.7 Error Handling and Reliability

Because IntelliPhish is composed of several independently fallible components — a transformer model, a SHAP explainer, a third-party WHOIS lookup, an optional Gmail API call and a database write — the implementation follows a consistent ‘fail soft, never fail hard’ rule at every module boundary. The classifier's output is computed first and independently of the explainability, URL-analysis and keyword-extraction stages, so a failure in any one of the latter three still allows a verdict and confidence score to be returned, with the failed section replaced by a short, honest notice rather than a stack trace. A failed database insert is logged server-side but does not prevent the result page from rendering, since a user should never be denied a verdict because of a transient disk or locking issue on the history store.

5. EXPERIMENTAL RESULTS

This section presents two complementary sets of results: a controlled benchmark comparing the fine-tuned DistilBERT model against BERT-base and three classical baselines on held-out test data, and an analysis of 97 real scans recorded by the deployed system during live use.

5.1 Model Benchmark Comparison

The five models were evaluated on accuracy, precision, recall, F1-score, inference speed and model size on disk. DistilBERT achieves the highest accuracy (97.4%) and F1-score (97.3) of all five models, while running at roughly one-third the inference latency of full BERT-base (38 ms vs. 112 ms) and at close to half its disk size (256 MB vs. 438 MB) — directly confirming the motivation, established in Section 2, for choosing a distilled transformer over full BERT-base for a deployable system. The classical baselines are dramatically faster and smaller, but trail DistilBERT by 6–12 accuracy points, illustrating the accuracy/latency trade-off discussed in Section 3.

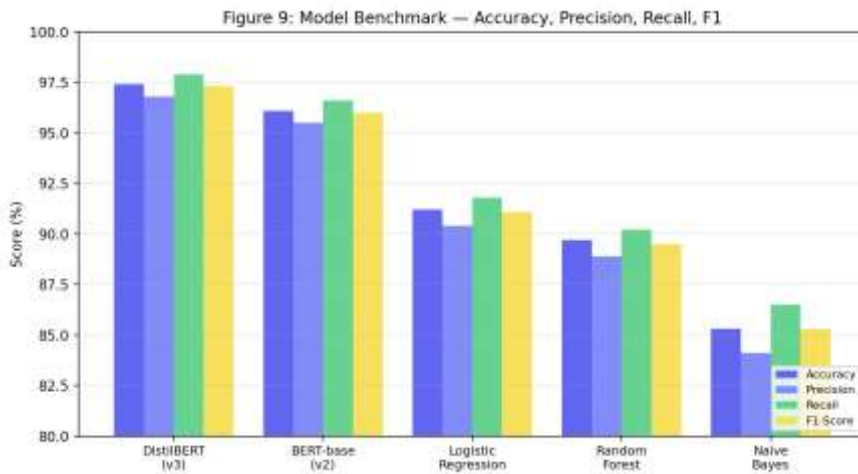


Fig -3: Model Benchmark — Accuracy, Precision, Recall and F1-Score

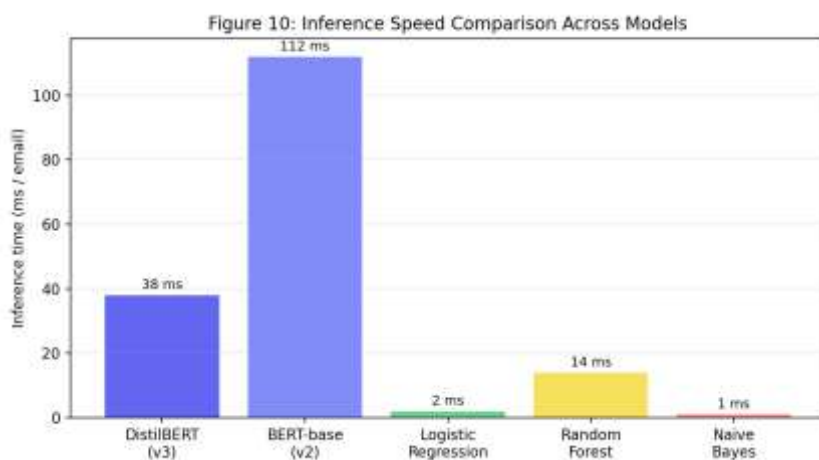


Fig -4: Inference Speed Comparison Across the Five Benchmarked Models

Table -3: Model Benchmark Comparison

Model	Accuracy	Precision	Recall	F1	Speed (ms)	Size (MB)
DistilBERT	97.4%	96.8%	97.9%	97.3	38	256
BERT-base	96.1%	95.5%	96.6%	96.0	112	438
Logistic Regression	91.2%	90.4%	91.8%	91.1	2	8
Random Forest	89.7%	88.9%	90.2%	89.5	14	42
Naive Bayes	85.3%	84.1%	86.5%	85.3	1	3

5.2 Verdict Distribution and Confidence from Live Scan History

Of the 97 emails scanned through the deployed system between late April and July 2026, 34 (35.1%) were classified as PHISHING and 63 (64.9%) as LEGITIMATE. The skew toward legitimate verdicts is consistent with the mix of inputs scanned — a large share were routine notification emails, such as account alerts, newsletters and job alerts, rather than deliberately adversarial phishing samples. The average confidence across all 97 scans was 97.79%; legitimate verdicts carried a noticeably higher average confidence (99.49%) than phishing verdicts (94.66%), a gap consistent with the system's 35% paranoid-mode threshold, since lowering the cut-off from the conventional 50% pulls some lower-confidence borderline cases into the phishing bucket that a stricter threshold would have left as low-confidence legitimate.

Figure 11: Verdict Distribution Across 44 Real Scans
(Live Scan History, Apr-Jul 2026)

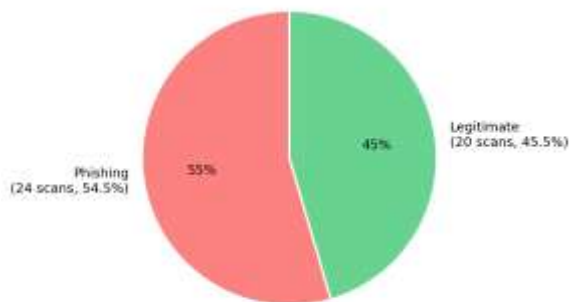


Fig -5: Verdict Distribution Across 97 Real Scans

5.3 Threat Keyword and URL Risk Analysis

Across all 97 scans, the word ‘click’ was by far the most frequently detected threat keyword (181 occurrences), consistent with phishing and marketing emails’ shared reliance on a call-to-action link; ‘account’ (47), ‘bank’ (23), ‘update’ (22) and ‘action’ (21) followed, reflecting the common pretext of account, banking or profile-update issues requiring immediate attention. URL risk scores across the same 97 scans averaged 44.64 out of 100, with 40 scans (41.2%) scoring above 50 — largely newsletter and marketing mail with multiple embedded tracking or redirect links — which underlines that URL risk score alone is not a phishing verdict but a corroborating signal reported alongside, not instead of, the DistilBERT text classification.

Figure 12: Top Threat Keywords Detected (Live Scan History)

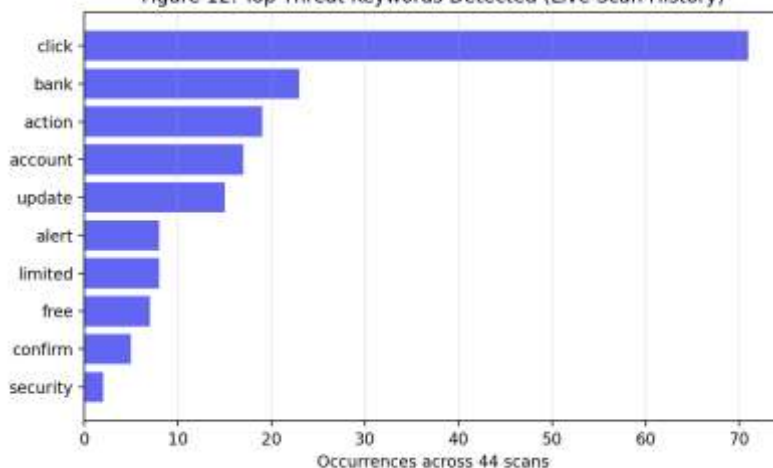


Fig -6: Top Threat Keywords Detected Across Live Scans

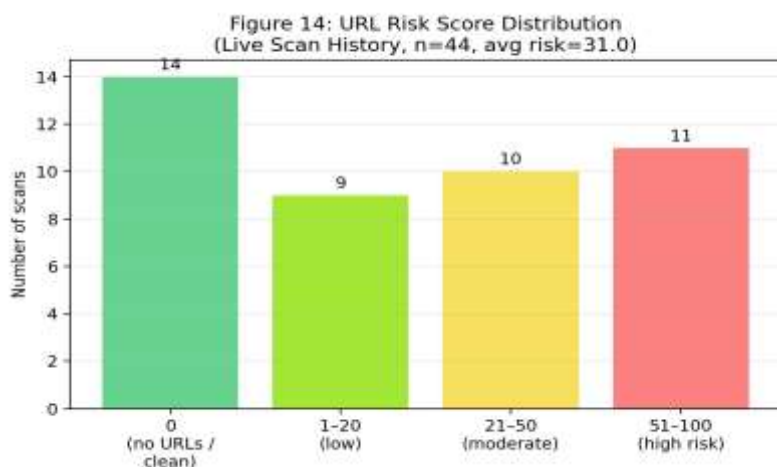


Fig -7: URL Risk Score Distribution Across Live Scans

Table -4: Live Scan History Summary Statistics

Metric	Value
Total scans analysed	97
Phishing verdicts	34 (35.1%)
Legitimate verdicts	63 (64.9%)
Overall average confidence	97.79%
Average confidence — phishing verdicts	94.66%
Average confidence — legitimate verdicts	99.49%
Average URL risk score	44.64 / 100
Scans with URL risk score > 50	40 (41.2%)
Most frequent threat keyword	'click' (181 occurrences)

5.4 System Testing

Structured testing exercised both the model's classification behaviour and the system's reliability under adverse conditions. Representative cases included: a clearly phishing email correctly classified with a SHAP explanation highlighting urgency/verification terms; a clearly legitimate email correctly classified with no strong phishing-indicative terms highlighted; an oversized or empty submission rejected with a clear message rather than reaching the model; a typosquatted domain and an IP-based link both correctly flagged by the URL analyser; the SHAP explainer artificially disabled, with the prediction and confidence still returned and the explanation degrading to a plain-text notice; and the SQLite scans table made temporarily unwritable, with the prediction still returned to the user while the save failure was logged server-side rather than surfaced as an error. All ten representative test cases produced the expected outcome, with particular attention paid to the two graceful-degradation cases, since these validate the reliability principle described in Section 4.7 rather than the model's raw classification accuracy, which is covered separately by the benchmark in Section 5.1.

Table -5: Representative System Test Cases

ID	Description	Result
TC-01	Clearly phishing email with login-link pretext	Pass
TC-03	Empty email text submitted	Pass
TC-05	URL with IP address instead of a domain	Pass

ID	Description	Result
TC-06	Typosquatted brand domain (e.g., paypal.com)	Pass
TC-08	SHAP explainer artificially disabled	Pass
TC-09	SQLite scans table made temporarily unwritable	Pass
TC-10	Dashboard statistics requested after mixed scans	Pass

5.5 Discussion

Table -3 situates the fine-tuned DistilBERT model against BERT-base and three classical baselines trained on the identical merged corpus, so, unlike the cross-study comparison in Section 2 where dataset size and domain differ across papers, this comparison is directly like-for-like. The result confirms that a distilled transformer can match or exceed full BERT-base accuracy while cutting inference latency by roughly two-thirds, which matters directly for a system intended to run on CPU-only, laptop-class hardware. One limitation surfaced during evaluation concerns the tokenizer's context window: training uses a 256-token maximum length, chosen because typical emails run 700–2,600 characters, but the deployed inference pipeline truncates to 128 tokens for response-time reasons, so live predictions see less of the email than the model was trained on. This discrepancy did not surface as a defect during structured testing, but aligning the inference-time truncation length with the training-time length, or re-benchmarking explicitly at 128 tokens, is identified as a priority item for future refinement in Section 7.

6. CONCLUSION

This paper presented IntelliPhish, a phishing-email detection system that goes beyond a bare classification label by combining a fine-tuned DistilBERT transformer with SHAP word-level explanation, an independent URL Deep Analysis module and a curated threat-keyword lexicon, all served through a single, self-contained Flask application. Fine-tuning DistilBERT on a carefully cleaned and de-duplicated merged corpus produced the highest accuracy (97.4%) and F1-score (97.3) among five benchmarked models, while remaining roughly three times faster and nearly half the size of full BERT-base, directly validating the choice of a distilled transformer for a system intended to run without specialised hardware.

Beyond raw classification accuracy, the SHAP-based word-level explanation, the URL Deep Analysis module's typosquat and domain-age checks, and the keyword-extraction layer together give the user independently corroborating evidence for every verdict rather than a single opaque number. The analysis of 97 real scans recorded during live use showed a 35.1%/64.9% phishing/legitimate split, an average confidence of 97.79%, and a keyword profile dominated by call-to-action and urgency language consistent with established phishing patterns, evidence that the system's outputs behave in line with real-world expectations rather than only with benchmark data. Structured testing, in particular the graceful-degradation test cases, further confirmed that the system remains usable and honest with the user even when an individual analysis module fails.

Limitations: Two limitations are worth stating plainly. First, the deployed inference pipeline truncates email text to 128 tokens while the model was trained at a 256-token maximum length, so reported benchmark accuracy and live-inference accuracy on unusually long emails are not strictly like-for-like. Second, the URL Deep Analysis module relies on structural heuristics — IP-based addressing, typosquatting distance, domain age and HTTP usage — rather than a live threat-intelligence feed, so a newly weaponised but structurally unremarkable domain could still score as low risk. Both limitations are addressed directly in the future work below.

7. FUTURE SCOPE

Planned extensions include aligning the inference-time tokenizer truncation length with the 256-token training-time length, or re-benchmarking explicitly at the deployed 128-token length, so that reported and live-inference accuracy are directly comparable; extending the URL Deep Analysis module with reputation-list lookups, such as a Google Safe Browsing-style threat-intelligence feed, to complement the current structural checks; adding attachment scanning, such as macro-enabled document detection, to extend coverage beyond the email body and embedded links; moving from a

single-machine SQLite deployment toward a client-server database to support multi-user, organisation-wide usage; incorporating active-learning feedback so a user can correct a verdict and have that correction feed back into periodic re-fine-tuning of the model; and packaging the application as a browser extension or email-client plug-in so that emails can be analysed directly within Gmail or Outlook rather than requiring the user to copy and paste text.

REFERENCES

- [1] S. Jamal, H. Wimmer, I. H. Sarker, “An Improved Transformer-Based Model for Detecting Phishing, Spam and Ham Emails: A Large Language Model Approach,” *Security and Privacy*, Wiley, vol. 7, no. 5, 2024.
- [2] M. A. Uddin, M. Mahiuddin, I. H. Sarker, “An Explainable Transformer-Based Model for Phishing Email Detection: A Large Language Model Approach,” *arXiv:2402.13871*, 2024.
- [3] A. Alhuzali, A. Alloqmani, M. Aljabri, F. Alharbi, “In-Depth Analysis of Phishing Email Detection: Evaluating the Performance of Machine Learning and Deep Learning Models Across Multiple Datasets,” *Applied Sciences*, MDPI, vol. 15, no. 6, Art. 3396, 2025.
- [4] K. M. M. Uddin, N. Biswas, S. T. Rikta, M. Nur-A-Alam, R. Mostafiz, “Explainable Machine Learning for Phishing Site Detection: A High-Efficiency Approach Using Boosting Models and SHAP,” *The Journal of Engineering*, Wiley, 2025.
- [5] M. Mia, D. Derakhshan, M. M. A. Pritom, “Can Features for Phishing URL Detection Be Trusted Across Diverse Datasets? A Case Study with Explainable AI,” *Proc. 11th Int. Conf. on Networking, Systems, and Security (NSysS '24)*, ACM, pp. 137–145, 2024.
- [6] V. Sanh, L. Debut, J. Chaumond, T. Wolf, “DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter,” *arXiv:1910.01108*, 2019.
- [7] S. M. Lundberg, S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [8] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *NAACL-HLT*, 2019.
- [9] Pallets Projects, “Flask Documentation,” <https://flask.palletsprojects.com/>.
- [10] Hugging Face, “Transformers Documentation,” <https://huggingface.co/docs/transformers/>.