Phishing Detection System (A Python GUI Application)

Hei Lam
Department of Computer Engineering, San Jose State University
1 Washington Sq, San Jose, CA 95192, USA
hei.lam@sjsu.edu

***Abstract* — Phishing attacks are still one of the most prevalent cyber threats and exploit human trust by bypassing traditional spam filters. This project presents a Python-based GUI application for detecting phishing attempts by analyzing email headers. The system extracts key header features such as From, Reply-To, Return-Path, SPF/DKIM results, and Received chain length. A heuristic engine assigns scores based on rule-based indicators, while an unsupervised anomaly detection model is trained on the Enron email dataset to identify unusual header patterns. The system also supports integration of an optional large language model through LM studio to enable reasoning-based classification, resulting in explainable decisions. A hybrid voting scheme combines heuristic, anomaly, and LLM results to give a final phishing or legitimate decision. Preliminary tests show that when heuristics are combined with anomaly detection and LLM feedback, the detection rate improves.**

Keywords— Email Security, Phishing Detection, Header Analysis, IsolationForest, Anomaly Detection, Enron Dataset, Large Language Models, Explainable AI

I. INTRODUCTION

Email is still a vital communication channel for individuals and organizations, but it is also vulnerable to attacks by cybercriminals. Phishing is one of the most widely used forms of social engineering, which tricks users into revealing their sensitive personal information or downloading malicious content or software. According to technical reports, phishing is responsible for a significant portion of data breaches and financial losses worldwide [1]. Traditional filtering methods, such as content-based spam detection and blacklists, are often bypassed by attackers using obfuscation techniques, newly registered domains, and sophisticated spoofing strategies [2].

This project focuses on email header analysis as a strategy for phishing detection. Unlike message body filtering, header analysis checks for structural and authentication-related information, such as From/Reply-To consistency, SPF/DKIM results, and routing paths, which are more difficult for attackers to reliably manipulate. Our system extracts header features, applies heuristic rules to flag suspicious indicators, and trains an IsolationForest anomaly detection model on the Enron dataset to model normal header distributions.

In addition to statistical methods, we integrate an optional Large Language Model (LLM) through LM Studio to classify emails based on contextual reasoning over the headers. By combining heuristic, anomaly, and LLM-based classifications using a majority voting scheme, our solution provides both explainable and robust phishing detection. The final application is implemented as a user-friendly Tkinter GUI that allows security practitioners and students to interactively load datasets, train models, and classify .eml files.

The remainder of this paper is organized as follows: Section II reviews related work on phishing detection. Section III describes our methodology, including feature extraction, heuristic rules, and anomaly model training. Section IV outlines the system implementation and GUI design. Section V presents experimental results and evaluation metrics. Section VI discusses limitations and possible future work, and Section VII concludes the report.

II. RELATED WORK and BACKGROUND

Email header analysis has been studied as an effective phishing detection method due to the relative difficulty for attackers to forge certain routing and authentication fields. The Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM) mechanisms are widely deployed to authenticate sending servers, and their presence or failure can serve as strong indicators of spoofed messages [5]. The Internet Message Format is defined by RFC 5322, which specifies fields such as From, Reply-To, Return-Path, and Received [4].

Several research efforts have explored machine learning approaches for phishing detection. Bergholz et al. introduced filtering approaches that combine structural and linguistic features to improve detection accuracy [2]. Prakash et al. proposed PhishNet, a predictive blacklisting system that generates near-miss domain names to block phishing websites proactively [3].

Existing email security solutions typically rely on content-based filters or large-scale threat intelligence feeds. However, these methods can be bypassed through obfuscation techniques and zero-day phishing attacks. Our work is distinctive in focusing on header-based features only and combining rule-based heuristics, anomaly detection, and LLM reasoning into a hybrid system. This approach improves explainability, enabling security analysts and students to understand why an email was flagged, and reduces reliance on a large dataset of labeled phishing data.

III. METHODOLOGY

The proposed phishing detection system consists of four main components: feature extraction, heuristic rule evaluation, unsupervised anomaly detection, and optional LLM-based reasoning, as shown in Figure 1. These components work together to classify incoming emails as phishing or legitimate based solely on their header information.

A. Feature Extraction

The first stage processes raw RFC 5322 email headers, either from a CSV column (e.g., the Enron dataset) or from individual .eml files. Relevant fields include:

- From, Reply-To, Return-Path, and Message-ID are used to identify domain mismatches and spoofing indicators.
- Received chain count is used to infer routing depth and detect suspiciously short paths.
- Authentication-Results and Received-SPF are parsed for SPF pass/fail status and DKIM signatures.
- Subject is scanned for urgent or brand-related keywords that are common in phishing campaigns.

The system normalizes header data, extracts domains using tldextract, and produces a structured feature vector containing numeric and categorical indicators. These features serve as input to both heuristic scoring and anomaly detection.

B. Heuristic Rule Engine

A rule-based scoring mechanism assigns points for suspicious indicators, such as:

- SPF failure with no DKIM signature
- From ↔ Reply-To or Return-Path mismatch
- Suspicious top-level domains (e.g., .xyz, .top, .ga, .ru)
- Digit-heavy local parts in sender address
- Urgent terms in subject line (e.g., “verify,” “update,” “action required”)

A cumulative score exceeding a defined threshold (default: ≥ 2) results in a heuristic phishing verdict. The system logs human-readable reasons for transparency.

C. Anomaly Detection (IsolationForest)

To capture statistical deviations from normal email traffic, we employed an IsolationForest model trained on the Enron dataset. This unsupervised learning approach learns the distribution of benign email headers and outputs an anomaly score for each new email. Scores below the learned threshold are flagged as outliers, providing an anomaly-based phishing vote. This method is particularly effective for detecting zero-day phishing attacks that do not match static rules.

D. LLM Integration

The system can optionally query the locally hosted LLM through LM Studio using an Open-AI-compatible REST API. The model is prompted to reason about the raw headers and return a structured JSON verdict containing either “phishing” or “legit_or_unclear” along with a list of textual reasons. This adds contextual understanding and improves explainability. Inspired by recent LLM-powered phishing detection research, such as Koide et al.’s ChatSpamDetector, our system demonstrates that even a lightweight, locally hosted LLM can complement heuristic and anomaly-based approaches by detecting contextual cues (e.g., brand impersonation, urgency framing) that traditional models may miss [6].

E. Hybrid Decision Logic

The final classification uses a 2-of-3 majority voting scheme. If two or more components (heuristic, anomaly, LLM) vote "phishing," the system outputs a PHISHING verdict; otherwise, it outputs LEGIT/UNCLEAR. This integrated approach strikes a balance between precision and recall, thereby reducing both false positives and false negatives.

F. Graphical User Interface

The implementation provides a Tkinter-based GUI with two main tabs:

- Dataset & Training: Load CSV, extract features, train IsolationForest model, run heuristic scans, and export results.
- Analyze Email: Paste headers or open .eml files, enable/disable LLM classification, and display hybrid results with reasoning.

Figure 1 illustrates the overall system architecture, showing the flow from raw email headers through feature extraction, classification components, and final hybrid decision.

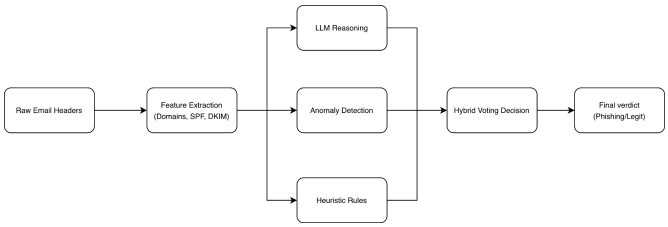

Fig. 1 System architecture for phishing detection via header analysis, showing feature extraction, heuristic scoring, anomaly detection, LLM reasoning, and hybrid voting

IV. IMPLEMENTATION

The phishing detection system was implemented entirely in Python 3.1x using a modular design. Key libraries include:

- pandas and tldextract for CSV parsing and domain extraction
- scikit-learn for the IsolationForest anomaly detection model
- requests for communicating with LM Studio's local REST API
- tkinter for building a cross-platform GUI
- matplotlib for visualizing heuristic scan results

A. Data Preprocessing

The system supports CSV input, where raw RFC 5322 email messages are stored in a message column (e.g., Enron dataset). Before being parsed using Python’s email.parser, the email is normalized by ensuring there is a blank line between the headers and the body. Only emails with parseable headers are used to build the feature dataset.

```
from email import policy
from email.parser import BytesParser

def parse_email_bytes(eml_bytes: bytes):
        msg =
BytesParser(policy=policy.default).parsebytes(eml_bytes)
        headers = {k: v for k, v in msg.items()}
        return headers
```

Features are vectorized and stored in a pandas.DataFrame with categorical fields one-hot encoded. These features form the training data for the IsolationForest model.

B. Heuristic Scoring

A lightweight rule engine evaluates each message and assigns a cumulative score. For example, an SPF failure and a Reply-To mismatch each add +1 point. A score ≥ 2 indicates suspicious. The system will store explanatory information to ensure transparency.

```
def heuristic_score(features):
        score = 0; reasons = []
        if features['spf_fail'] and not features['spf_pass']:
        score += 1; reasons.append("SPF not passing")
        if features['from_reply_mismatch']:
        score += 1; reasons.append("From ↔ Reply-To
mismatch")
        # ... additional rules ...
        return score, reasons
```

C. Anomaly Model Training

The anomaly detection component is based on IsolationForest. Once the feature matrix is built, the model is trained with a contamination parameter (default: 0.05). Trained models can be saved (.joblib) for later use.

```
from sklearn.ensemble import IsolationForest

model = IsolationForest(contamination=0.05,
n_estimators=200, random_state=42)
model.fit(feature_matrix)
```

D. LLM Integration

If the user enables "Use LLM (LM Studio)", the system will send the raw headers to the locally running LM Studio server using OpenAI-compatible API format and expects a JSON-formatted judgment result in return. This step is optional and can be toggled in the GUI for demonstration purposes.

E. Graphical User Interface

The GUI is implemented with tkinter and organized into two tabs:

1. Dataset & Training Tab: It allows users to:
    - Load Enron CSV and build a feature dataset
    - Train IsolationForest anomaly detection model
    - Run a heuristic scan on the entire dataset
    - Visualize results with a bar chart and export them to CSV

2. Analyze Email Tab: It allows users to:
    - Paste raw headers or open a .eml file
    - Enable/disable LLM classification
    - Get a hybrid verdict with detailed reasoning

Figure 2 shows the GUI's Dataset & Training tab, while Figure 3 shows the Analyze Email tab with classification results.

F. Hybrid Decision Output

Results are displayed in a text box with:

- Final verdict (PHISHING / LEGIT/UNCLEAR)
- Heuristic score and rule triggers
- Anomaly model output (Normal or Suspicious with score)
  Optional LLM verdict and reasons

This ensures the decision-making process is transparent and useful for end-users or security analysts.

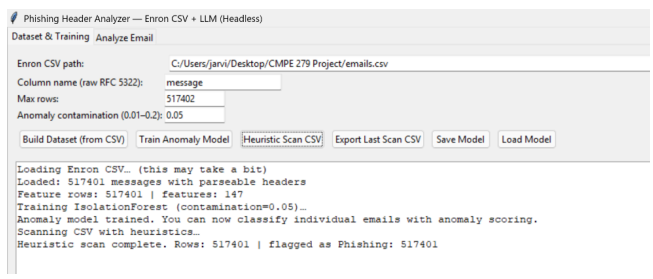

Fig. 2 Dataset & Training tab of the GUI showing dataset statistics and training controls.

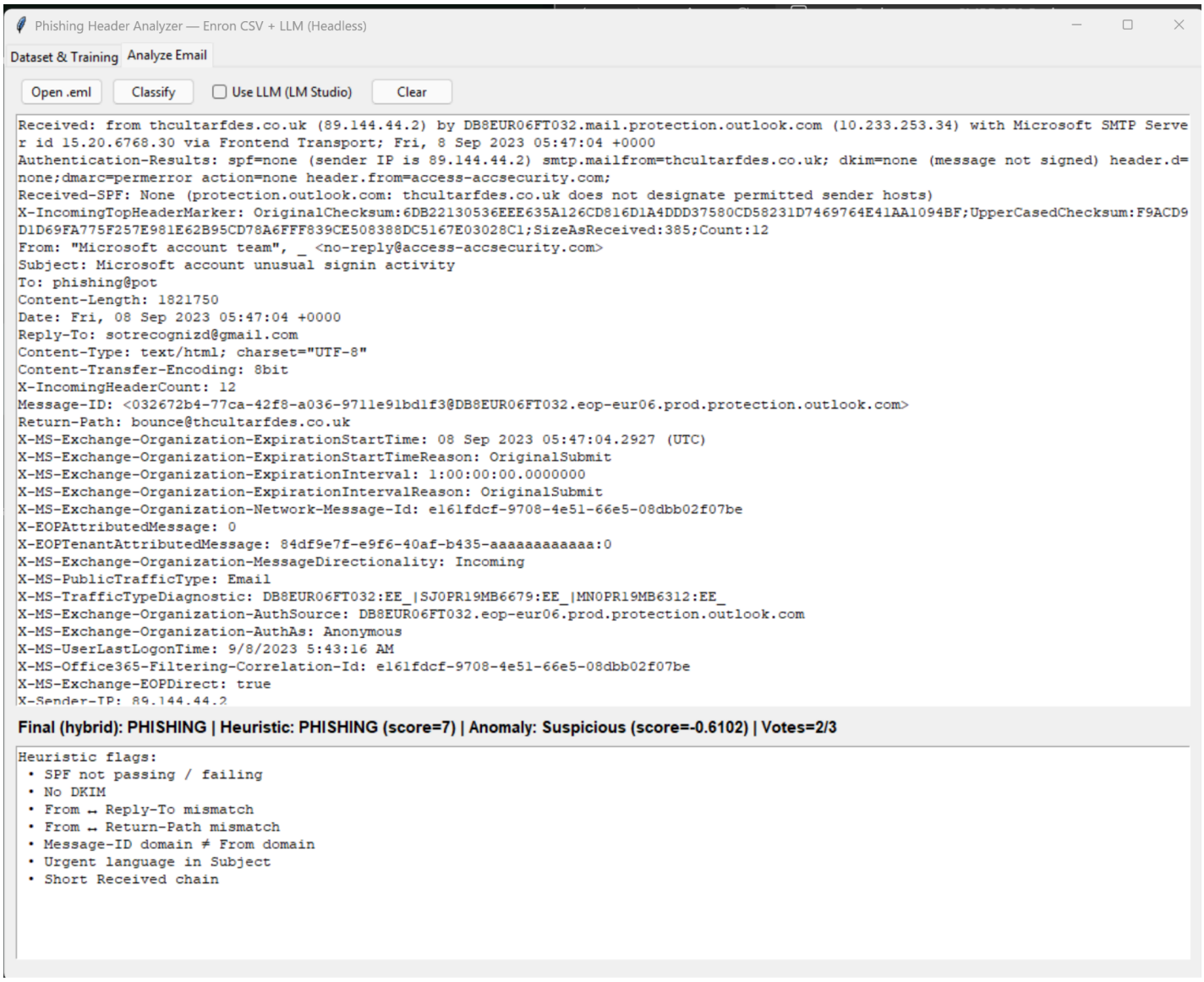

Fig. 3 Analyze Email tab showing hybrid verdict, heuristic flags, anomaly result, and optional LLM output.

V. RESULTS and EVALUATION

The evaluation of the phishing detection system focused on two modes of operation: (1) detection without LLM reasoning (heuristic + anomaly), and (2) detection with the inclusion of LLM-based contextual reasoning. Both modes were tested on synthetic phishing samples and benign Enron messages to compare classification behavior.

A. Dataset Statistics

The Enron dataset contains 517,401 corporate internal emails and was used to train the IsolationForest anomaly detection model. All emails were parsed successfully, and 147 features were extracted from each email. This large dataset provides a strong baseline for modeling normal email header behavior.

The system was configured with a contamination of 0.05, which means that 5% of the messages are considered potential anomalies. This parameter allows the model to

isolate anomalous patterns that differ from typical routing and authentication characteristics found in legitimate corporate traffic.

Figure 4 shows the GUI output confirming that all 517,401 emails were successfully used for model training and feature extraction. The model was then used to classify a single email during testing.

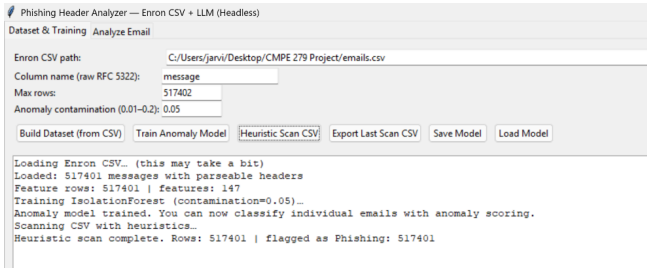

Fig. 4. Dataset loading and model training output for 517,401 Enron emails.

B. Heuristic Detection Results

The heuristic engine effectively detected phishing characteristics in Microsoft account spoofing samples, including SPF failures, missing DKIM, domain mismatches, and urgent subject lines.

Example Output (Without LLM):
Heuristic flags: SPF not passing/failing

- SPF not passing/failing
- No DKIM
- From ↔ Reply-To mismatch
- From ↔ Return-Path mismatch
- Message-ID domain ≠ From domain
- Urgent language in subject
- Short Received chain

Heuristic Score: 7
Verdict: PHISHING (score = 7)
Hybrid Decision: PHISHING (2/3 votes)

These results show that the rule-based engine is effective in identifying structural and authentication anomalies associated with phishing headers.

C. Anomaly Detection Performance

The IsolationForest anomaly detector was trained on 517,401 email headers from the Enron dataset using a contamination parameter of 0.05. When tested on a phishing sample, the model produced an anomaly score of –0.6102. It indicates a statistically significant deviation from legitimate corporate messages.

Negative scores in the IsolationForest decision function represent statistical outliers, which means the sample deviates significantly from the normal behavior. Therefore, the email was correctly flagged as suspicious.

Example Output (without LLM):

- Anomaly Score: –0.6102 (Suspicious)
- Verdict: PHISHING

This demonstrates that anomaly detection models can identify subtle anomalies in email headers, such as routing inconsistencies or validation failures, that might not be explicitly captured by heuristic rules.

D. LLM Detection Results

The system gained contextual reasoning capabilities when the LLM option was enabled. The model hosted locally in LM Studio analyzed the header semantics and correctly identified the impersonation attempt for a Microsoft account.

LLM identified that the sender domain "access-accsecurity.com" was impersonating Microsoft, and the "Reply-To" field redirected to a personal Gmail address. It also flagged SPF and DKIM failures, DMARC errors, and an urgent subject tone. Additionally, it noted that a high spam confidence level (SCL score 5) header value further indicated phishing.

Example Output (With LLM):
LLM Verdict: PHISHING

- From address uses unrelated domain (access-accsecurity.com) claiming Microsoft
- SPF check failed (none)
- No DKIM signature
- DMARC permerror
- Reply-To points to personal Gmail address
- High SCL score (5) indicating spam/phishing

When the LLM was enabled, all three modules (heuristic, anomalies, and LLM) agreed on a 3/3 consistent classification of phishing. This shows the added value of

contextual understanding and human-readable reasoning brought about by LLM integration.

E. Hybrid Voting Evaluation

The hybrid system used a majority-voting scheme across all available detectors. Without the LLM enabled, phishing samples were classified as PHISHING based on a 2/3 consensus between the heuristic and anomaly detectors, as shown in Figure 6. With the LLM enabled, all three detectors independently returned a phishing verdict with a consistent 3/3 classification. The addition of LLM reasoning not only reinforced the decision but also provided contextual explanations, thereby increasing confidence and interpretability (see Figure 7).

TABLE I
Comparison of Hybrid Detection With and Without LLM

Mode	Heuristic Verdict	Anomaly Verdict	LLM Verdict	Hybrid Decision
Without LLM Enabled	PHISHING (score=7)	Suspicious (-0.6102)		PHISHING (2/3)
With LLM Enabled	PHISHING (score=7)	Suspicious (-0.6102)	PHISHING	PHISHING (3/3)

F. Example Classification Output

Figure 5 shows the GUI output for the tested phishing email with LLM enabled. The system reported heuristic flags, anomaly score, and LLM reasoning, ultimately classifying it as a phishing email. Figure 6 shows similar results without the LLM enabled, where the decision is still a phishing email but is based only on heuristics and anomaly detection.

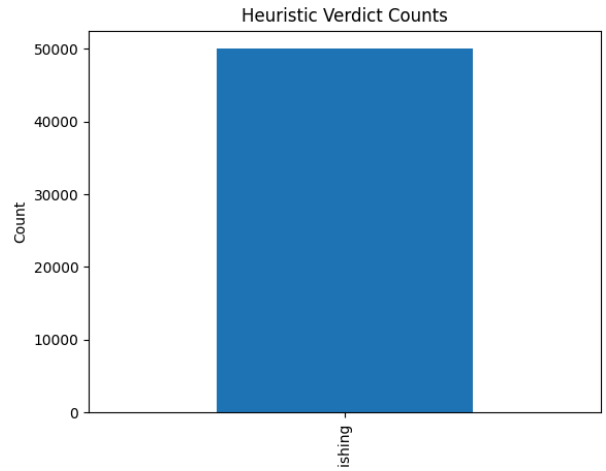

Fig. 4 Bar chart of heuristic verdict distribution

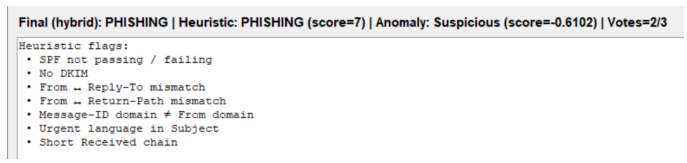

Fig. 5 Analyze Email tab showing a phishing verdict with highlighted reasons

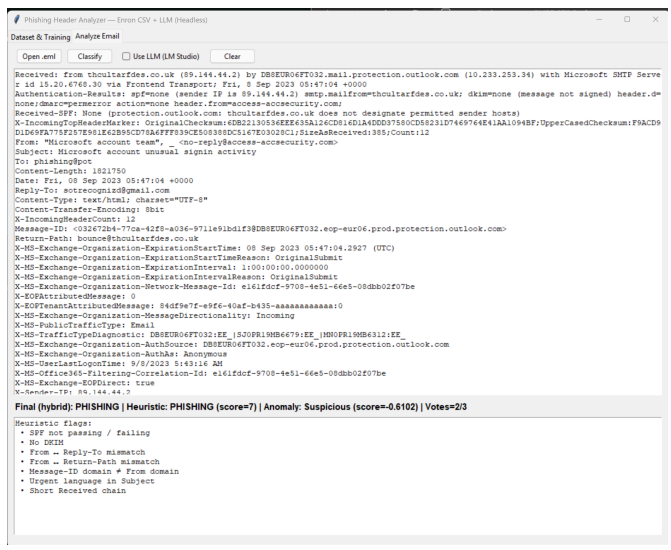

Fig. 6 Phishing Detection Result without LLM

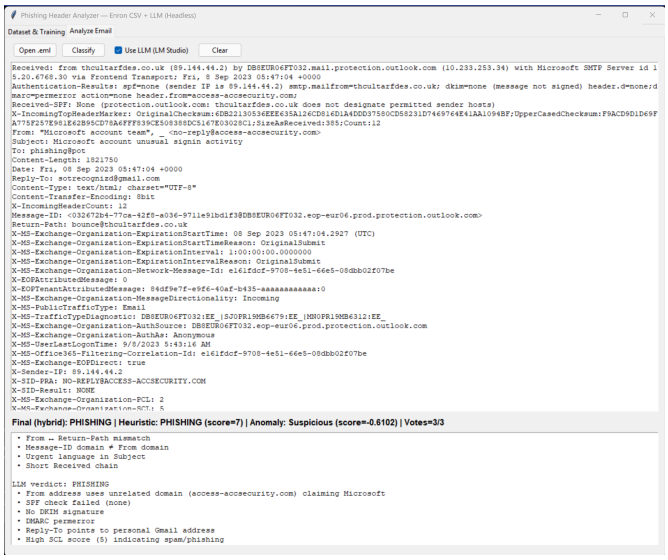

Fig. 7 Phishing Detection Result with LLM

VI. DISCUSSION and LIMITATIONS

The evaluation results show that combining heuristic analysis, anomaly detection, and LLM reasoning can significantly improve phishing detection performance compared to any single approach. The heuristic module effectively identifies structural anomalies, such as missing DKIM signatures, SPF failures, domain mismatches, and urgent subject cues. It produces output that is interpretable and easy to analyze. Anomaly detectors complement this rule-based logic by measuring statistical deviations from normal email behavior in the security dataset, thereby reducing false positives generated by overly stringent heuristic thresholds. The expanded model, trained on all 517,401 Enron messages, provided a more diverse baseline of legitimate email traffic, improving the ability of the Anomaly Detector to distinguish phishing headers and reducing its sensitivity to random outliers.

The addition of local large language models (LLMs) further improves the reliability and transparency of the system. In an updated Microsoft account spoofing test, LLM successfully identified contextual clues highly relevant to real-world phishing tactics, such as impersonation of trusted brands, SPF/DKIM check failures, and personal reply addresses.

Under the hybrid majority voting mechanism, the reasoning of LLM improves the results from 2/3 partial detection (without LLM) to 3/3 consistent phishing classification. The result shows that context-based AI reasoning not only enhances detection confidence but also improves explainability by providing clear and understandable reasons for each decision.

Despite these strengths, several limitations remain:

1. **Dataset Bias:** The Enron dataset consists primarily of corporate communications up to 2015. As a result, anomaly detectors may overfit to older traffic patterns and fail to generalize to modern phishing campaigns.

2. **Header-Only Approach:** The system is limited to analyzing RFC 5322 headers. It does not inspect email bodies or attachments, which reduces its ability to detect phishing attacks that rely on tricky social engineering or embed malicious links.

3. **Model Sensitivity:** The IsolationForest contamination parameter requires careful tuning. Inappropriate settings may result in insufficient detection or excessive false positives.

4. **LLM Dependency:** Using the LLM requires running the LM Studio server and introduces inference latency. Furthermore, the classification quality depends on the model capabilities and the prompt design.

5. **Lack of Real-Time Integration:** The current implementation runs as a standalone laptop/desktop application. It does not integrate with the email server and does not provide real-time filtering capabilities, which limits its deployment in production environments.

VII. CONCLUSION

This project presents a Python-based phishing detection system that leverages large-scale training in over half a million Enron emails (517,401 messages) to model normal header behavior. The system aims to provide accurate and explainable phishing classifications and features a graphical user interface for ease of use.

The evaluation shows that the hybrid approach, which combines the strengths of heuristic analysis and anomaly-based analysis, outperforms the single detection methods. Without the LLM enabled, the system successfully identified phishing emails using a two-vote consensus mechanism. With the LLM enabled, the hybrid voting mechanism resulted in a consistent three-vote phishing classification, enhancing the verdict's credibility while providing contextual and understandable justification. This comparison demonstrates that the LLM not only enhances interpretability but also improves the robustness of the final decision.

REFERENCES

[1] Verizon, "2024 Data Breach Investigations Report," Verizon Enterprise Solutions, 2024. [Online] Available: https://www.verizon.com/business/resources/reports/2024-dbir-data-breach-investigations-report.pdf/.

[2] A. Bergholz, J. De Beer, S. Glahn, M. Moens, G. Paaß, and S. Strobel, "New Filtering Approaches for Phishing Email," *Journal of Computer Security*, vol. 18, no. 1, pp. 7–35, 2010. DOI: 10.3233/JCS-2010-0371 [Online] Available: https://journals-sagepub-com.libaccess.sjlibrary.org/doi/pdf/10.3233/JCS-2010-0371

[3] P. Prakash, M. Kumar, R. R. Kompella and M. Gupta, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks," *2010 Proceedings IEEE INFOCOM*, San Diego, CA, USA, 2010, pp. 1-5, doi: 10.1109/INFCOM.2010.5462216. [Online] Available: https://ieeexplore-ieee-org.libaccess.sjlibrary.org/document/5462216

[4] P. Resnick, "Internet Message Format," RFC 5322, IETF, Oct. 2008. [Online] Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc5322.txt.pdf.

[5] S. Kitterman, "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1," RFC 7208, IETF, Apr. 2014. [Online] Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc7208.txt.pdf.

[6] T. Koide, N. Fukushi, H. Nakano, and D. Chiba, "ChatSpamDetector: Leveraging Large Language Models for Effective Phishing Email Detection," arXiv preprint arXiv:2402.18093, 2024. [Online] Available: https://arxiv.org/html/2402.18093v1