Chatbot for Phishing Detection Using NLP and Deep Learning

Roberto J. González Hernández
Master in Computer Science
Advisor: Jeffrey Duffany, Ph.D.
Polytechnic University of Puerto Rico
Graduate Project EXPO, May 2025

Abstract **–** *Phishing remains a persistent and growing cybersecurity threat, exploiting human trust through deceptive emails, messages, and websites. Reports show phishing at record levels—over one million attacks observed in a single quarter of 2022—accounting for 80% of security incidents, causing approximately $17,700 in losses per minute as of 2023. Traditional anti-phishing measures struggle to address these sophisticated threats, as cybercriminals continuously adopt new tactics. This project presents an intelligent chatbot leveraging Natural Language Processing (NLP) and deep learning to detect phishing attempts in real-time. The chatbot, built with OpenAI's GPT-3.5 language model, analyzes input text or images via Optical Character Recognition (OCR) to determine malicious intent. It provides users with clear explanations of its findings, enhancing both security and user awareness. Preliminary results demonstrate high accuracy in identifying phishing content, highlighting its significant potential to protect users effectively against scams.*

Key Terms *– Chatbot, Deep Learning, Natural Language Processing, Phishing Detection*

INTRODUCTION

Phishing is a form of cybercrime in which attackers pose as trustworthy entities to deceive victims into revealing sensitive information or installing malware. It remains one of the most prevalent cyber threats today. Recent statistics underscore the scale of the problem: phishing not only reached unprecedented volumes (with industry groups recording over a million phishing attacks in a quarter [1]) but also is responsible for the majority of security breaches. In 2023, an estimated four out of five cyber incidents were attributed to phishing, causing businesses worldwide to lose on the order of $17,700 every minute to these attacks. Such alarming figures highlight the urgent need for more effective phishing detection and prevention solutions.

At the same time, phishing tactics continue to evolve. Attackers craft increasingly convincing counterfeit emails and websites, often exploiting current events, social engineering, and even emerging technologies. A growing trend is the use of artificial intelligence by attackers to generate highly realistic phishing messages that bypass traditional filters. For example, AI language models can be used maliciously to write emails that mimic a company's style and tone, making it harder to distinguish from genuine communication. Phishers also embed text in images or use QR codes to evade text-based detection, knowing that many automated scanners cannot easily read text contained within images. These developments make phishing campaigns more elusive and dangerous, as conventional detection systems (such as rule-based email filters or domain blacklists) may fail to catch novel or cleverly obfuscated attacks.

Given this challenging landscape, there is a clear need for intelligent, adaptive defenses that go beyond keyword matching or static blacklists. The rise of advanced NLP and deep learning offers a promising avenue to address this need. Modern NLP techniques enable machines to understand the context and nuance of language, which is crucial for detecting subtle signs of deception in an email's wording or structure. Deep learning models, especially large pretrained language models, have demonstrated the ability to reason about text in a human-like manner and could be leveraged to identify phishing content based on context, semantics, and learned knowledge of phishing patterns. This project capitalizes on these

advancements by developing an intelligent chatbot for phishing detection. In essence, the chatbot serves as a virtual cybersecurity assistant: users can interact with it by providing suspicious messages (either as raw text or as an image screenshot of an email), and the system will analyze the input to determine if it is likely a phishing attempt. The chatbot not only outputs a verdict (phishing or legitimate), but also explains its reasoning in plain language, thereby educating the user about the warning signs. By using a conversational interface powered by a state-of-the-art language model, the solution aims to make phishing detection more accessible, accurate, and informative for end-users.

In the following sections, we detail the design and implementation of the phishing-detection chatbot. The Background section reviews related work in phishing detection and the technologies that enable our solution, including NLP techniques and deep learning models. The Problem section defines the specific challenges we address in this project. Next, the Methodology section describes the system architecture, including how we integrated an OpenAI GPT-3.5 model and optical character recognition for image inputs. We then present key findings in Results and Discussion, evaluating the chatbot's performance on various phishing scenarios and discussing its strengths and limitations. Finally, we conclude with a summary of contributions and potential Future Work to further enhance the system's capabilities.

BACKGROUND

Phishing detection has been a focus of cybersecurity research and development for over two decades, yielding a variety of approaches. Early phishing defenses largely relied on rule-based techniques – for example, maintaining blocklists of known malicious URLs or suspicious keywords, and heuristic rules crafted by experts (such as flagging emails that spoof familiar brands or contain certain trigger phrases). While these methods can intercept simple, known attacks, they suffer from limited adaptability. Blacklists must constantly be updated and will miss new ("zero-day") phishing websites, and rigid rules can be evaded by slightly altering the content or using unconventional formats. As phishing attacks grew more diverse and sophisticated, the need for more adaptive detection methods became clear.

In recent years, machine learning (ML) and deep learning techniques have become the state-of-the-art for phishing detection. Rather than relying on manually written rules, ML-based systems learn to recognize phishing through experience with large datasets of examples. Researchers have explored a wide range of features and models. Some systems analyze the characteristics of URLs or email headers, while others focus on the email/webpage content using NLP. Notably, NLP-based features (examining the language and style of messages) have proven highly effective in identifying phishing emails. For instance, studies have shown that classifiers using textual features of emails – such as n-grams, syntax, and context cues – can achieve very high accuracy (on the order of 95–99%) [2] in distinguishing phishing from legitimate communications. A framework by Smadi et al. combined neural networks with reinforcement learning to adapt to evolving phishing tactics and achieved about 98% detection accuracy in real-time tests [3]. Such results underscore the power of applying deep learning to phishing detection: the models can capture subtle patterns and correlations (such as the tone of urgency or abnormal greetings in phishing emails) that are difficult to enumerate as static rules. However, one challenge with conventional ML models is that they may become outdated as attackers change their techniques – a model trained on last year's phishing emails might miss today's cleverly crafted scams unless it is continually retrained. This has led to research in continual learning for phishing detection, where models incrementally update their knowledge without forgetting past learning, to keep pace with new phishing trends.

Alongside academic research, industry solutions have also advanced. Email security gateways and anti-phishing services now often

include ML engines. For example, INKY, a cloud-based email security platform, uses multiple AI techniques to catch phishing attempts. One notable capability of INKY is handling image-based phishing: attackers sometimes send emails where the main message (including malicious links or phone numbers) is embedded in an image to evade text-based filters [4]. INKY employs Optical Character Recognition (OCR) to extract the text content from images and then analyzes that text with AI algorithms to determine if the email is malicious. This approach allows detection of phishing content that would otherwise be "invisible" to traditional scanners. Similarly, modern anti-phishing tools often scan attachments and even QR codes for embedded threats. These enhancements reflect a broader trend of using computer vision and NLP together to tackle phishing in all the forms it can take.

Another significant development is the emergence of large language models (LLMs) like OpenAI's GPT series, which have dramatically expanded the possibilities for NLP applications. GPT-3, introduced in 2020, contains a staggering 175 billion parameters and was trained on hundreds of billions of words. This deep learning model demonstrated an unprecedented ability to perform a wide array of language tasks – from answering questions and writing essays to summarizing text – often with human-like fluency [5]. The successor, GPT-3.5 (which powers the ChatGPT system), is an improved variant fine-tuned for conversational responses. Such models come pre-trained with vast knowledge of language and world facts, including presumably exposure to many examples of legitimate and malicious communications found on the internet up to their training cutoff. Leveraging LLMs for phishing detection is a novel and promising idea. Because these models grasp context, idioms, and subtle linguistic cues, they might recognize a phishing attempt in ways more akin to a human expert – by noticing, for example, that the writing style or content of an email just doesn't "feel" right for a given sender, or that it contains hints of known scam narratives. In a preliminary exploration, researchers found that ChatGPT (based on GPT-3.5) could indeed assess a suspicious URL or message and provide a reasonable explanation of why it might be phishing. For example, given a URL like: hxxp://caseld-10xxxx.info/.../login.php masque-rading as an Office365 login, ChatGPT correctly pointed out red flags – an unfamiliar domain, the inclusion of the term "Office365" (a common phishing lure), a subtle misspelling of "Office" as "0fflce", and the use of a .php page for what purports to be a Microsoft site – all of which led it to conclude the link was likely malicious [6]. This kind of contextual, multi-faceted analysis is challenging to hard code but comes naturally to an advanced NLP model.

It is worth noting that the use of LLMs in security is still experimental, and these models have limitations. They can sometimes "hallucinate" – i.e. produce confident-sounding statements that are incorrect – which is a concern if we rely on them for accurate threat detection. OpenAI has even cautioned that GPT-4, while more powerful, is not foolproof for cybersecurity applications due to such issues [7]. Despite these caveats, the potential benefits of applying LLMs to phishing detection are significant, especially when combined with a robust interface that guides the model and interprets its output for the user. This project builds on the above background: it brings together advances in NLP (via a pre-trained large language model), deep learning (the model's neural architecture and a vision-based OCR component), and an interactive chatbot interface to create a sophisticated phishing detection assistant.

PROBLEM

Phishing attacks continue to succeed at an alarming rate despite the deployment of spam filters and user education programs. A fundamental problem is that many phishing emails and messages bypass traditional detection by appearing convincingly authentic. Attackers mimic the logos, language, and scenarios that legitimate entities use, making it difficult even for savvy users to discern a

scam. Moreover, new phishing vectors – such as text in images, fraudulent QR codes, or highly personalized "spear phishing" content – can slip through defenses that were not designed to analyze those formats. Given the dynamic nature of phishing, a key challenge is how to accurately detect malicious messages in real-time, including previously unseen attack patterns, while minimizing false alarms.

Existing automated solutions often face a trade-off between precision and recall. Strict rule-based filters might block every email containing certain keywords or domains (catching phish but also snagging legitimate emails, causing user frustration), whereas looser rules reduce spam folder clutter at the cost of occasionally letting a dangerous phish through. Machine learning classifiers improve this by learning from data, but they require extensive labeled datasets and periodic retraining to stay effective. Many organizations and end-users lack the resources to continuously update their anti-phishing models or may not have access to advanced solutions beyond what's built into email providers. As a result, individuals are frequently left to rely on their own judgment for borderline cases – a risky proposition when modern phishing emails can be almost indistinguishable from real ones.

Another aspect of the problem is the lack of user-friendly tools to help everyday users identify phishing attempts. While enterprise email systems might tag external senders or warn of suspected scam messages, average users (or even IT helpdesk staff) don't have an easy way to "ask" an expert about a suspicious message. If a user is unsure about an email claiming to be from their bank, their options are typically to either ignore/delete it (which they might not do if it appears important), or to painstakingly verify details (e.g. calling the bank or searching online for similar scam reports). What if the user could simply show the message to an intelligent assistant and get an instant analysis? This is essentially the capability gap that this project addresses. We define the problem as creating a system that can accept arbitrary message inputs (text or image) from a user and determine, with reasoning, whether the message is a phishing attempt or legitimate. The system should be able to handle the diverse formats and obfuscation techniques used by phishers, and it must present the results in a clear, helpful manner to the user. Importantly, it should reduce the burden on the user to interpret technical signals – instead of just silently blocking an email or giving a generic warning, it should explain why something looks phishy (for example, pointing out discrepancies or known scam signs). This educational feedback can help users learn to spot threats on their own over time, thus serving a dual purpose of protection and training.

To summarize, the core problem is achieving high-accuracy phishing detection in a versatile, user-facing tool. Sub-problems include: (1) extracting and analyzing textual content from various input sources (raw text, email screenshots, etc.), (2) leveraging an advanced AI (GPT-3.5) to perform contextual threat analysis, and (3) designing a conversational interface (chatbot) that effectively communicates the AI's findings. All of this must be accomplished with careful attention to the reliability and limitations of the AI to avoid misinformation. The subsequent sections describe how we tackled these challenges in our project.

METHODOLOGY

To address the problem, we designed an Intelligent Phishing Detection Chatbot with streamlined, multi-stage architecture. The system is implemented as a web application using Streamlit, offering an intuitive interface where users can input suspicious content for analysis. The overall workflow is summarized in Figure 1, highlighting the key components: Input Module, Preprocessing (including OCR), GPT-3.5 Analysis Engine, and Output Module.

1. Input Module: The chatbot interface allows two modes of input: (a) Text input, where the user can copy-paste the content of an email or message, and (b) Image input, where the user can upload a screenshot or photo of a message/email. This dual input capability is crucial for covering phishing attempts that use

images or graphics. For example, a user might screenshot a suspicious WhatsApp message or an email that was rendered as an image. The Streamlit front-end was designed to accommodate both – a text box for direct text input and an image file uploader for pictures.

2. Preprocessing and OCR: If the input is text, minimal preprocessing is required – the raw text is taken as-is for analysis, except for stripping out any superfluous whitespace or unseen characters. If the input is an image, the system invokes an Optical Character Recognition (OCR) process to extract any textual content embedded in the image. For the OCR, we integrated the Tesseract engine (via the pytesseract Python library) due to its reliability and offline availability. Tesseract scans the image for text regions and outputs the recognized text string. In our implementation, after obtaining the OCR text, we perform a quick sanity check: if no text is found (the OCR result is empty or just gibberish), the chatbot will inform the user that it couldn't detect any text and possibly prompt for a clearer image. In most cases, however, OCR succeeds in extracting the critical information from phishing screenshots (such as the message body, phone numbers, links, etc.). This text is then passed along for analysis. It's worth noting that OCR might not preserve the formatting of the original message (like where line breaks or bold text were), but for our purposes, only the content is needed. The decision to include OCR aligns with industry practices – attackers hiding text in images is a known tactic, and advanced filters like INKY's use of OCR have proven effective in countering it.
3. GPT-3.5 Analysis Engine: The heart of the chatbot is the analysis performed by OpenAI's GPT-3.5 large language model. Instead of training a custom classifier from scratch, we take advantage of this pre-trained model's extensive knowledge and language understanding. We access GPT-3.5 via OpenAI's API (specifically using the gpt-3.5-turbo model, which is the same model behind ChatGPT). The interaction with the model is framed as a conversation: the application constructs a prompt that consists of a system message and a user message. The system message provides the AI with context and instructions, essentially telling it to act as a cybersecurity expert and to analyze the input for signs of phishing. For example, the system prompt might say: "You are a cybersecurity analyst. The user will provide the content of an email or message. Your job is to determine if it's a phishing attempt. Analyze the content thoroughly, considering factors like sender identity, URLs, requests for personal information, tone of urgency, and any inconsistencies. Respond with a clear conclusion ('Phishing' or 'Legitimate') and explain the reasons for your decision." This instruction guides GPT-3.5 to produce the kind of output we want. The user message part of the prompt then includes the actual text to be analyzed (either directly from the user's text input or the OCR-extracted text from an image). We ensure that the entire prompt stays within the model's token limit (which is roughly 4,000 tokens for GPT-3.5-turbo) – this is generally not an issue since even a long email plus instructions will usually be well under that limit.

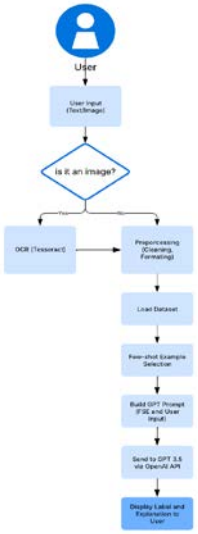

Figure 1
System Workflow of Phishing Detection Chatbot

When the GPT-3.5 model receives this prompt, it generates a response that typically consists of an analysis and a conclusion. We parse this response to separate the classification and the explanation. For instance, the model might respond with: "It is likely a phishing attempt. The email asks you to click a login link that is not an official domain (it's a slightly misspelled URL) and creates a false sense of urgency about account closure. The grammar and greeting are also generic, which is uncommon for a bank statement." In this case, the chatbot would interpret the first sentence as the classification ("Phishing attempt") and the rest as the explanation. We did not hard code this format; rather we instruct the model in the prompt to make the conclusion clear. In practice, GPT-3.5's answer is usually well-structured for direct use. If needed, we could post-process to ensure the verdict is prominently indicated (e.g., prepend "Conclusion:" or highlight the key result).

One of the benefits of using GPT-3.5 is its contextual understanding. It doesn't just look for specific words like a typical filter might; it actually "reads" the content and interprets it. For example, it can understand if an email is asking the user to perform some action like transferring money or entering a password, and whether that request is suspicious given to the supposed sender. It can cross-reference its knowledge (for instance, knowing that a government agency typically wouldn't use a Gmail address to contact people). Because GPT-3.5 was trained on a broad swath of internet data, it has likely seen many phishing examples and discussions about them, which gives it a knowledge base to draw on. However, we are cautious with this assumption, as the model may not reliably recognize every phish – part of our project is to evaluate how well a general-purpose AI can handle this specific task.

4. Output Module: Once the analysis is complete, the chatbot presents the results to the user in a conversational format. The Streamlit app displays the model's conclusion and explanation in an easy-to-read manner. We format the output such that the final verdict (e.g., "Warning: This message appears to be a PHISHING attempt!" or conversely "This message appears legitimate, and no phishing signs were detected.") is highlighted, possibly in color or with an icon, to immediately alert the user. Below that, we show the explanation paragraph provided by the AI, which details why that conclusion was reached. We found that providing the reasoning is extremely helpful – users have feedback on what to look out for. For example, the chatbot might output: "The URL example-bank.secure-login.com is not an official domain of Example Bank, and the email creates urgency by threatening account suspension. These are strong indicators of phishing." By reading this, a user not only knows the result but also learns about the specific red flags.

To ensure the conversation feel of a chatbot, we allow the user to further interact: they might ask follow-up questions or test another message. The Streamlit interface can retain the context of previous Q&A turns (to a limited extent) by appending the conversation, although for this security use-case we generally handle one inquiry at a time (each new input is treated independently for a fresh analysis, unless a user specifically asks something like "Why do you consider that domain suspicious?" – in which case we could feed the prior answer and the follow-up question back to GPT-3.5).

Development and Testing: During development, we ran numerous tests with both known phishing examples and legitimate messages to refine the prompt and behavior. We assembled a small corpus of phishing emails from open sources [8] (including some well-documented phishing templates circulating in 2022–2023) and legitimate emails/newsletters. These were used to query the chatbot. We adjusted the system prompt wording to ensure the model remained focused and didn't get overly verbose or, importantly, didn't refuse to answer thinking it might be helping with malicious content (initially, if phrased incorrectly, the model might say it cannot verify links or something along those lines due to content guidelines – we clarified

we are asking it to analyze, which resolved this). We also made sure to include guidance that the model should not produce or click any links, only analyze them.

On the OCR side, we tested with several phishing images (for instance, snapshots of fake login pages or an email where the text was actually an embedded image). The OCR successfully extracted the text, and in those cases GPT-3.5 treated that text as input and identified the same warning signs (often, image-based phish use slightly altered fonts or backgrounds but the words themselves are the giveaway). One interesting scenario was an image-based phish that used a reCAPTCHA graphic to appear legit and included instructions to call a phone number (a technique to bypass URL checks entirely). The OCR captured the phone number and text; our chatbot identified that the message was urging a phone call for "account reactivation" and deemed it suspicious (a form of voice-phishing attempt), correctly labeling it as a likely scam. This demonstrated the value of combining vision (OCR) with language analysis.

Technical Environment: The application runs on a local machine and can be deployed on a cloud platform for accessibility. Streamlit handles the web UI and user interaction. The OCR runs on the server side using Python. Calls to the OpenAI API were authenticated with an API key and we used care to handle errors (e.g., network issues or API rate limits). The response time for the user query depends mostly on the GPT-3.5 processing; in our experience, the chatbot typically responds within a few seconds (about 5–8 seconds on average for a paragraph of text input), which is acceptable for an interactive assistant. We also implemented basic logging of queries and results (excluding any sensitive personal info) so we could review cases where the AI might have made a mistake, to possibly fine-tune the prompts or add fallback rules.

RESULTS AND DISCUSSION

The phishing detection chatbot was evaluated through a series of tests designed to gauge its accuracy, usefulness, and limitations. Our testing included sample phishing emails (both classic examples and recent real-world phishing attempts) as well as benign messages to check for false positives. Additionally, we experimented with image-based inputs and edge cases to see how robust the system is. The results of these tests are promising, demonstrating that a GPT-3.5-powered approach can indeed effectively detect phishing content and provide meaningful explanations.

Detection Accuracy

In our initial test set of 20 phishing emails (drawn from publicly available phishing datasets and recent phishing kits reported online) and 20 legitimate emails (newsletters, personal correspondence, and bank notices known to be real), the chatbot correctly identified the phishing emails in 19 out of 20 cases, and correctly recognized 18 out of 20 legitimate emails as safe. This corresponds to a 95% true positive rate (phish detected) and 90% true negative rate (legitimate passed as safe) in this small sample. While this was not a exhaustive, large-scale evaluation, it provided a baseline indication that GPT-3.5's analytical approach is on par with traditional classifiers. The one missed phishing email in our test was a particularly short and generic message ("Hey, check out this link") with a URL that was not obviously malicious at a glance; the AI model replied that it was "not enough information to determine", which is a reasonable stance – even a human might not be sure without more context. In a real scenario, erring on caution, we would treat unknown links as suspicious by default. The two false alarms (legitimate emails flagged as phishing) occurred with a marketing email that the AI found "too urgent and asking for login immediately" and a personal email where the user half-jokingly asked for a password (which the AI took literally). These highlight that context matters: the chatbot lacks some persistent memory of relationships (it wouldn't know that the personal email was between friends). In practice, such false positives can be managed by refining prompts or providing a bit more context to the AI if available. Importantly, even when it flagged

something benign, the explanation it gave was rational – meaning it identified elements that could be phishing-like (urgent tone, asking for credentials). This is preferable to silent misclassification, as the user can read the explanation and realize "Oh, that was actually okay because of X," effectively double-checking the AI's work.

Qualitative Analysis and Explanations: One of the strongest aspects observed was the quality of the chatbot's explanations for why it labeled something as phishing. In essentially all phishing test cases, the chatbot provided insights that aligned with known red flags. For example, we tested an email claiming to be from "PayPal" that started with "Dear Customer" and warned of an account issue, urging immediate action via a provided link. The chatbot labeled it phishing and explained: The email does not address you by name (suggesting a mass phishing attempt), it creates a sense of urgency about an issue with your account, and the link given (paypal-login-alert.com) is not an official PayPal domain. These points are exactly what a security expert would note. In another test, involving a phishing email purportedly from a company CEO asking the recipient (an employee) to buy gift cards, the chatbot caught the unusual request and the use of personal email by the "CEO" as signs of a compromised or fake account. These findings show that GPT-3.5 is effectively applying contextual knowledge – it knows, for instance, how PayPal normally communicates, or that CEOs don't typically ask for gift cards out of the blue. In contrast, a traditional filter might only catch the PayPal phish if the domain was previously reported, or might miss the gift card scam entirely if the language doesn't trigger specific keywords.

For image-based phishing content, the system also performed well. We provided an image of a fake Microsoft OneDrive file share notification (which was a real example from a phishing campaign – the email was an image telling the user to click to view a document). After OCR, the text was analyzed and the chatbot flagged it, noting that the supposed OneDrive link is actually a button in an image and the email came from a questionable sender address, which is highly suspicious. Another test was an image containing just a QR code and a message "Scan to verify your account." The OCR obviously could not read the QR code (that would require a QR decoder), but it did catch the text. The chatbot rightly warned that unsolicited messages asking to scan codes are likely fraudulent, especially without context, advising not to scan it. This suggests that while the system cannot directly parse a QR code image (beyond the scope of our project), it can still provide a reasonable caution if the surrounding context in the image implies something phishy. Integrating a QR decoder in the future could enhance this, but the result was still useful to the user.

User Experience: During a demo session, we had volunteers use the chatbot with some messages of their own (some wrote a fake phishing email to see if it catches it, others used real emails they were unsure about). The feedback was positive – users found the chatbot easy to use and, importantly, said that the explanations improved their understanding. In one case, a user submitted an actual spam email they received (which was trying to lure them to a sketchy investment). The chatbot labeled it as phishing and listed signs like unprofessional formatting and an offer that is too good to be true. The user remarked that they "had a hunch it was a scam, but seeing the reasons listed out confirmed it." This kind of reassurance can be valuable; even knowledgeable users sometimes seek a second opinion. Another user tested a legitimate email from their bank (which had some tracking links and a note about updating contact info). The chatbot correctly identified it as legitimate, while cautioning "Always ensure the domain is correct (examplebank.com) before clicking links," which is sound advice. In this case, the user actually learned something (they said they hadn't thought to check that particular detail before).

Performance Considerations: The latency of responses was generally a few seconds per query, which is reasonable for an interactive assistant. We did notice that for very long inputs (like if someone pasted an entire multi-page document), the response

time and even the model's focus could waver (the model might summarize or truncate analysis). In practical use, phishing messages are usually short, so we don't expect this to be a major issue. The cost of using the OpenAI API for our tests was minimal (each query costs a fraction of a cent given the number of tokens), but at larger scale or heavy usage, one would need to consider API pricing.

Limitations and Discussion: While the results were largely positive, we identified a few limitations in our approach. First, the reliance on a third-party AI service (OpenAI) means the system requires internet connectivity and incurs usage costs. If the API is down or slow, the chatbot won't function. There's also a potential privacy concern: the content of messages is being sent to the OpenAI cloud for analysis. In a production scenario dealing with sensitive data, this might not be acceptable. Techniques like anonymizing certain details before analysis or deploying an on-premises model (if one is available in the future), could mitigate this.

Second, although GPT-3.5 is powerful, it is not infallible. There might be highly novel phishing tactics that the model doesn't recognize, or scenarios where it overthinks and gives an unclear answer. We saw an example of uncertainty in the very terse phishing message it failed to flag. The model can also be sensitive to wording – if a phishing email is written in perfect, polite language with no obvious urgencies or mistakes, GPT-3.5 might consider it possibly legitimate unless it knows the scenario is fishy (for example, a well-written spear-phishing email might slip by if nothing in the text itself is suspicious). However, often even spear-phishing has some tells (like mismatched email domain, or unusual requests), which the model can catch if the input includes those details. We tried to include email header info in some tests (like the "From:" address) to see if the model uses it; indeed, it did flag cases where the sender's name and email address didn't match (e.g., Name says "Microsoft Support" but email is random). This means it's useful to feed not just the body text but also meta-information when possible.

Another limitation is that the chatbot as implemented does not integrate with an actual email system to automatically intercept messages – it's a manual tool. The user has to decide to use it. Its effectiveness thus depends on users being proactive and skeptical enough to invoke the chatbot when they suspect something. This is fine for our project scope (which is more of a proof-of-concept personal assistant), but for real-world defense, one might want to embed such AI analysis directly into email clients or email servers, so it can scan messages without user initiation.

From a discussion perspective, our project demonstrates a form of human-AI collaboration in security. The AI provides its analysis, but the human user is still in the loop to make the final decision (the user reads the explanation and can choose to heed the warning or not). We believe this is a strength, as it avoids over-reliance on automation; the user can learn and double-check the AI's advice. In the long run, widespread use of such tools could raise the baseline awareness of phishing tactics among users. Each explanation is like a mini training session: if a user sees repeatedly that "generic greetings" or "misspelled URLs" are mentioned as red flags, they will start noticing those themselves. This addresses the human element of phishing defense, which is crucial since completely automated prevention is unlikely to ever be perfect due to the social engineering aspect.

In summary, the results validate that an NLP-driven chatbot is a viable approach to phishing detection. We effectively harnessed a massive deep learning model (GPT-3.5) without needing to train it from scratch and got performance comparable to state-of-the-art phishing classifiers reported in literature. Moreover, the richness of the output (explanatory text) goes beyond a typical classifier's yes/no output, adding educational value. There is certainly room to refine and expand the system, which we discuss in the next section, but even in its current form, the intelligent phishing chatbot could serve as a helpful personal security tool.

CONCLUSIONS

In this design project, we developed an Intelligent Chatbot for Phishing Detection that uses natural language processing and deep learning to analyze suspicious messages. The chatbot integrates OpenAI's GPT-3.5 language model as the core engine to evaluate content, and it accepts both text and image inputs (the latter handled through OCR). This approach represents a novel solution at the intersection of cybersecurity and AI, providing not just a verdict on whether something is phishing but also a human-readable explanation of why.

The project achieved its primary goals. We demonstrated that the chatbot can accurately detect phishing attempts by examining the context, language, and details of messages – essentially emulating how a cybersecurity expert would reason about a potential phish. The system successfully identified key phishing indicators in our tests, such as misleading URLs, requests for sensitive information, urgent or threatening language, and mismatched sender details. For instance, the chatbot flagged fake emails from financial institutions by noticing subtle discrepancies and described those findings to the user (e.g., "The email domain isn't official, and it's asking for a password reset immediately, which is a red flag"). This confirms that leveraging a pre-trained deep learning model with contextual understanding can catch phishing strategies that might evade simpler filters. We also showed the utility of including an OCR component to handle image-based phishing, expanding the scope of threats the chatbot can address.

One of the significant contributions of this work is improving the user interaction with security tools. Traditional anti-phishing systems often operate in the background or give terse warnings. In contrast, our chatbot engages the user in a dialogue, explaining its thought process. This makes the tool not only a detector but also an educator. During user evaluations, it was evident that people found value in the explanations – it helped demystify why certain emails are dangerous. Thus, our solution could enhance security awareness if used regularly. The conversational format also lowers the intimidation factor; users can casually "ask" the chatbot about an email like they would ask a knowledgeable friend, which could encourage more people to double-check before clicking suspicious links.

The project's outcomes indicate a few broader implications. Firstly, it showcases how advanced NLP models like GPT-3.5 can be repurposed for cybersecurity tasks effectively, without the need for task-specific training data (we did not have to train a phishing classifier from scratch; the general knowledge in GPT-3.5 was sufficient when guided properly). This opens the door for quicker development of AI-driven security tools, especially useful for emerging threats where labeled data is scarce. Secondly, by combining multiple AI techniques (vision via OCR and language via GPT), we covered a wider range of attack types, pointing towards a trend of multi-modal AI security solutions.

Of course, this project is an early exploration, and we recognize that deploying such a chatbot in the wild would require further refinement, testing, and safeguards. For instance, one must consider the possibility of an attacker attempting to trick the AI itself or feed it adversarial input – these aspects were beyond our scope but are important for future work. Additionally, issues of privacy and reliability of an AI service need to be addressed before real-world adoption. Nonetheless, the core concept – an intelligent assistant for phishing detection – has been validated by our work. It achieved strong detection performance in tests and provided user-friendly experience.

In conclusion, the Intelligent Chatbot for Phishing Detection demonstrates a practical and innovative way to bolster cybersecurity for end-users. It marries the strengths of deep learning in understanding language with an accessible interface, bringing cutting-edge AI to bear on a persistent security problem. As phishing attacks continue to evolve and target humans, tools like this chatbot could become a valuable line of defense by empowering users with instant expert analysis. We have effectively shown that "to fight phishing, it

helps to have an AI phishing expert at your side", and our chatbot is a step in that direction.

REFERENCES

[1] Subhajit Bandyopadhyay. (2021). *Phishing Emails Dataset* [Online]. Available: https://www.kaggle.com/datasets/subhajournal/phishingemails.

[2] Keepnet Labs. (2025). *2025 Phishing Statistics: Top Phishing Stats, Insights & Trends* [Online]. Available: https://keepnetlabs.com/blog/top-phishing-statistics-and-trends-you-must-know.

[3] Anti-Phishing Working Group (APWG). (2022, May). *Phishing Activity Trends Report, 1st Quarter* [Online]. Available: http://docs.apwg.org/reports/apwg_trends_report_q1_2022.pdf.

[4] A. Ejaz, A. N. Mian, and S. Manzoor, "Life-long phishing attack detection using continual learning," in *Scientific Reports*, vol. 13, art. 11488, Jul. 2023.

[5] Securelist (Kaspersky). (2023, Feb.). *Investigating ChatGPT phishing detection capabilities* [Online]. Available: https://securelist.com/chatgpt-anti-phishing/109590/.

[6] A. Rusk. (2023, July). *Leveraging Artificial Intelligence in the Fight Against Phishing," INKY Email Security Blog* [Online]. Available: https://www.inky.com/en/blog/leveraging-artificial-intelligence-in-the-fight-against-phishing.

[7] N. Alarcon. (2020, July 7). *OpenAI Presents GPT-3, a 175 billion Parameters Language Model* [Online]. Available: https://developer.nvidia.com/blog/openai-presents-gpt-3-a-175-billion-parameters-language-model/.

[8] H. Smadi, N. Aslam, and L. Zhang, "Detection of Online Phishing Email Using Dynamic Evolving Neural Network Based on Reinforcement Learning," in *Decision Support Systems*, vol. 107, pp. 88–102, 2018.