The Phishermen

Hila Lewin
Abdullah Garra
Lilach Daniel

ABSTRACT

This report presents a solution for detecting phishing attempts in emails. The solution is a Chrome extension that utilizes machine learning technology and the Gmail API to provide real-time phishing detection for users. By integrating seamlessly with users' email experience, the extension empowers them to make informed decisions about the emails they engage with. The report details the development process, including the creation of a dataset of legitimate and phishing emails, the selection and optimization of machine learning models, and the implementation of the extension. The report also discusses the limitations and future work of the project, presenting a potential approach for future work; ML-based email content analysis that aims to detect techniques of influence commonly used in phishing emails. Overall, the presented Chrome extension project offers a promising solution to the problem of phishing attacks, with potential broader consequences that are substantial and multifaceted.

1 INTRODUCTION

In the rapidly evolving landscape of the digital age, where communication and transactions are increasingly conducted online, the threat of cyberattacks is substantial large. Among these threats, phishing has emerged as a particularly common form of cybercrime.

Cybercriminals adeptly exploit human psychology, utilizing deceptive tactics to manipulate users into disclosing sensitive information and compromising security and privacy. Of the various methods employed, phishing emails stand out as a prevalent and dangerous vector for these attacks, capable of infiltrating both organizational networks and personal accounts.

Phishing primarily relies on two human vulnerabilities: lack of knowledge and lack of attention. Fortunately, usable security and privacy tools have the potential to address both of these issues effectively. They act as a powerful shield against phishing attacks by minimizing the need for extensive cybersecurity knowledge and by alerting users to suspicious behavior.

In the context of reducing the need for knowledge, usable security tools provide user-friendly interfaces that are intuitive and require minimal technical expertise or limited cybersecurity knowledge.

Furthermore, these tools play a crucial role in bringing suspicious activities to the user's attention. Through real-time monitoring and intelligent algorithms, they can detect unusual or potentially harmful behaviors, such as suspicious emails. When such threats are identified, these tools promptly alert the user, enabling them to take immediate action to protect their personal information and digital assets.

In light of that, we present a solution in the form of a Chrome extension. Our extension targets email phishing attempts by providing users with valuable insights into the emails they receive. We will examine the email on various layers: content, links, spelling and grammar, urgency sense and sender information. The user will be warned if any of the above is detected in the email. By harnessing the power of machine learning technology and leveraging the capabilities of the Gmail API, our extension seamlessly integrates with users' email experience, delivering real-time phishing detection as they browse Gmail.

Our hypothesis is that users equipped with our extension will exhibit heightened resilience against falling victim to phishing attacks. Additionally, we believe that users who employ the extension with configurations that avoid excessive notifications will demonstrate diminished susceptibility to habituation. This, in turn, is projected to lead to more effective and judicious use of the extension, resulting in reduced susceptibility to phishing traps. In contrast, we expect that the group without the extension will not experience these benefits.

2 RELATED WORK

One aspect of security-related behavior is mindfulness. Mindfulness involves adopting specific practices, such as visiting trusted websites or practicing safe email habits, to guard against common cyber threats like phishing attacks[7]. Therefore, developing tools, such as browser security extensions, that enhance users' mindfulness can help mitigate phishing attacks. Recent research has shown that a large portion of users have recognized the practicality and usefulness of browsers security extensions[10]. This research also highlighted some key recommendations, including the following: only use HTTPS, be suspicious of unusual email if grammar in an email is not correct, double check email addresses and be suspicious of links.

2.1 Phishing Strategies

Previous studies have consistently highlighted the susceptibility of users to phishing threats due to errors in spelling and grammar within these malicious messages[12]. Therefore, a program that systematically scans email content for linguistic deficiencies, will be able to detect telltale signs of potential email-based cyber threats, allowing it to provide an additional layer of protection against these attacks.

The concept of urgency in email has been recognized as a pivotal dimension influencing user susceptibility to phishing attacks[5]. Furthermore, time pressure has the potential to impact motivation-related factors, potentially compromising the commitment to cybersecurity practices. By incorporating urgency analysis, a tool will have an enhanced ability to discern potential phishing threats.

It's also important to note the significance of the email sender's identity . Recent research indicates that the perceived sender of an email plays a pivotal role in predicting susceptibility to phishing attempts[9]. It was discovered that users will cooperate more with perceived familiar sources. Hence, an extension that alerts the user

in case the email source is unknown, can empower the user to identify a possible malicious source.

Furthermore, building upon previous statements, numerous studies have highlighted the Source Credibility Theory, which suggests that individuals tend to trust and depend on information that comes from sources they consider credible. By searching for any previous communication between the user and the domain of the email source, the extension will establish the genuine credibility of the sender.

It is widely recognized that attackers frequently employ malicious links within phishing emails, a tactic that renders these emails malicious in nature. In a prior study, it was discovered that a minimal proportion of individuals habitually verified URLs by hovering over email hyperlinks[10]. However, even if people uncover the real URL, determining its security status, remains a challenging task, one that even security experts find difficult to accomplish accurately without supplementary information[2]. It appears that a website's URL can reveal information about the website's nature. Earlier research, utilizing publicly accessible URL features, has successfully identified phishing URLs[3]. Given the research in this field, it is evident that a tool that is able to uncover the real URLs in the email and evaluate their security for the user in real time, is incredibly powerful.

2.2 User Study and Usability Measures

We will rely on a previous study [10], which provides measures for assessing the quality of security and privacy advice, to evaluate our Chrome extension in our user study.

Comprehensibility is metric that measures how easily users can understand the security and privacy advice presented.

Another metric presented in the article is **Actionability**. Actionability assesses whether the recommended security behaviors are specific, feasible, and relevant to users. One aspect of actionability is disruptiveness. Disruptiveness evaluates how much the recommended behaviors would require users to change their existing habits or routines.

Additionally, the study introduces the measure of **Compliance**, estimating the extent to which users report following the recommended security behaviors outlined in the advice.

3 METHODOLOGY/DESIGN

When developing the extension, we aimed to provide real-time phishing attempts detection each time an email message is read.

In the following section, we will go into details on our implementation and design choices.

3.1 Design Overview

The extension is activated every time the user browses Gmail. It comprises both client-side code written in JavaScript and server-side code written in Python. The client-side interacts with Gmail API to retrieve information required for the analysis, and displays a pop-up with the results of the analysis. The server-side handles requests by running ML models and other logic required to perform the analysis.

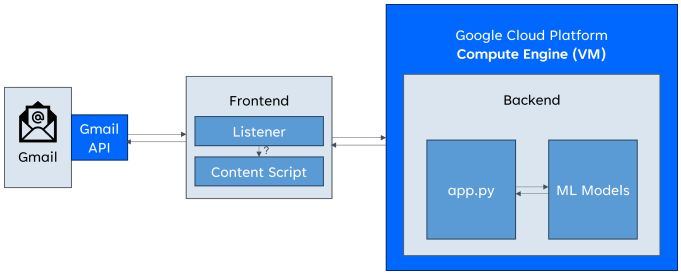

Figure 1: Design Overview

3.2 Machine Learning

3.2.1 *Analyzing Email Content.*

Constructing the Dataset. Our dataset consists of benign and phishing emails. When constructing the dataset, we had two main criteria in mind: (1) Large enough dataset, as it enhances the trained model's ability to capture complex patterns, leading to improved performance on real-world tasks. (2) A dataset that accurately represents an average inbox content, considering the emails' content and the ratio between benign and phishing emails.

To meet both of criteria, we needed a very large dataset, given that most of the emails are benign. The source we worked with is Enron Corpus[8], a well-known, large dataset that consists of emails generated by Enron Corporation's employees, utilizing approximately 60,000 emails from this corpus.

As for the phishing emails collection, we have managed to find a few sources online. Most of the datasets we found did not manage to fulfill criterion (2): they consisted of short emails that were poorly diverse. Not surprisingly enough, when we tried training different types of models on these datasets, it resulted in overfitted models. The only dataset we found that fits our needs[11] contains 326 emails, half of which are phishing. With ChatGPT's assistance, we successfully generated an additional 62 phishing emails, while conscientiously addressing potential overfitting issues. These were further refined and manually tailored to align with our requirements. The newly crafted emails exhibit a higher level of sophistication and better represent real-world phishing emails.

Optimized metrics. Accuracy, the commonly used metric, is the proportion of correctly predicted instances over the total number of instances in a dataset. However, it can be misleading when the classes in the dataset have unequal representation, as it does in our case.

The F1 score, on the other hand, is a harmonic mean of precision and recall.

$$F1 = \frac{2\,(\text{precision} \times \text{recall})}{\text{precision} + \text{recall}}$$

Precision measures the proportion of true positives among the predicted positives, while recall measures the proportion of true positives among the actual positives. The F1 score considers both false positives and false negatives, making it a better metric when class distribution is imbalanced, like in our case, as it balances the trade-off between precision and recall.

When optimizing the F1 score, we are seeking the threshold or decision point that maximizes this metric. This threshold determines how the model classifies instances based on their predicted probabilities. Although optimizing the F1 score provided substantially better results than optimizing accuracy, detecting phishing Emails is a Recall-Oriented problem since the focus in our case is to ensure that very few actual positive cases are missed. However, neglecting the precision would result in a high false positive rate which can significantly decrease the usability of our chrome extension.

Recall the F_β-score formula:

$$\frac{(1+\beta^2)\cdot \text{precision}\cdot \text{recall}}{(\beta^2)\cdot \text{precision} + \text{recall}}$$

Where β is the parameter that controls the trade-off between precision and recall:

- $\beta = 1$ is the F1-score.
- $\beta > 1$ emphasizes more on the recall.
- $0 < \beta < 1$ emphasizes more on the precision.

Since we have a Recall oriented problem, we will be focusing on F_β with $\beta > 1$.

In our pursuit of identifying the most effective model, we will conduct a series of experiments, optimizing each model using the following metrics:

- Accuracy
- F1-score
- F_β-score with $\beta = 2$
- F_β-score with $\beta = 1.2$

Vectorization. Vectorization is a fundamental technique in natural language processing (NLP) that transforms textual data into numerical vectors, enabling machine learning algorithms to process and analyze text. Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF) are two common vectorization methods. BOW creates vectors by representing each document as a frequency count of its constituent words, effectively capturing the presence of words in the text. On the other hand, TF-IDF assigns numerical weights to words based on their occurrence in a specific document relative to their frequency across the entire dataset, thus highlighting words with unique contextual importance.

In our analysis, we'll assess various model and metric combinations with both BOW and TF-IDF vectorization methods. By comparing results systematically, we aim to identify the best-performing approach for our phishing email detection system.

Model Selection. We systematically employed a rigorous approach to select an appropriate machine learning model for analyzing email content. Our methodology involved considering widely recognized and straightforward models. With a dataset containing more than 60,000 emails, 225 labeled as phishing and the rest as legitimate, we conducted 100 iterations. Each iteration involved 3600 randomly sampled legitimate emails. This approach introduced an element of randomness, ensuring that the performances were not accidental. Simultaneously, we maintained consistency with the 225 phishing emails throughout the iterations. We split the sampled data into 80% training and 20% testing, hence around 770 emails were in the testing part, while maintaining an identical distributions of labels in both sets. In each iteration, the models were freshly initialized. The models we evaluated were:

(1) Random Forest
(2) Logistic Regression
(3) Support Vector Machine (SVM)
(4) Naive Bayes Classifier
(5) KNN

Our evaluation process delved into assessing the performance of each model using various metrics, including accuracy, recall, precision, and F1-score. For each combination of model, vectorization method, we conducted four evaluations sessions. These sessions were tailored to optimize different metrics: the first session aimed for accuracy, the second focused on the F1-score, and the third and fourth aimed to optimize the F_β score for β values of 2 and 1.2 respectively. We also took into account the number of False Negatives and False Positives to provide a comprehensive perspective. In summarizing the results, we calculated the mean score for each metric across the 100 iterations. The same approach was taken for assessing False Negatives and False Positives, providing an overview of each model's performance. Our primary objective is to minimize the occurrence of false negatives while simultaneously maintaining a low count of false positives. The classifier we have selected after the evaluation is Support Vector Machine (SVM). SVM from scikit-learn uses a radial basis function (RBF) kernel by default, enabling it to capture non-linear relationships between input features and the target variable.
The final model was optimized by $F_{\beta=2}$ and was trained on ~10,000 legitimate emails along with 225 phishing emails.

3.2.2 Analyzing Links. During our research, we encountered a study aimed at the identification of malicious links, through the utilization of 30 extracted features. To assess feature importance, the study employed a permutation on full model[4], revealing that a subset of 14 features had an importance of more than zero.

The study also compared 9 models by their accuracy, F1 score, recall and precision. Gradient boosting classifier got the best results overall[4]. Motivated by these findings, we adopted the Gradient Boosting classifier and incorporated the features from this research into our analysis.

To streamline email analysis, we trained the classifier with different subsets of features. We started with the 14 important features and gradually trimmed them until results were less ideal. For each subset, we conducted 100 experiments on the same dataset, containing 4,893 phishing URLs and 6,157 benign URLs. The dataset from the study was partitioned into an 80% training set and a 20% testing set, resulting in the evaluation of around 2210 data points. We also computed the average accuracy and the average confusion matrix. Out of the 100 experiments, the model that had the highest accuracy and lowest false negative was selected.

We ultimately opted for one model comprising all 14 significant features and another consisting of only the top 5 significant features. The model chosen for the analysis is determined by user preferences.

3.2.3 Analyzing Grammar and Spelling. Misspelled emails or emails with bad grammar might indicate phishing attempts, according to multiple sources. For that reason, we integrated a grammar

and spelling detection ML model that will allow our extension to display an insight in case poor grammar or spelling is detected. The machine learning model is the most common Hugging Face grammar correction model[13]. The model runs on the content of each email we analyze. The output of the model is a corrected version of the email content. The set of the words in the output is compared to the set of the words in the original email by calculating Jaccard Similarity Index. Jaccard Similarity Index is a measure of the similarity between two sets of data, that ranges from 0 to 1.
The final response provided to the user depends on their preferences.

3.2.4 Urgency Detection. It was found that phishing emails often incorporate a heightened sense of urgency. This prompted us to analyze the degree of urgency present in each email.

To accomplish this, we utilized a zero-shot language model (bart-large-mnli by Hugging Face[6]) queried with the label "urgent". By employing this technique, we can effectively assess the urgency level and incorporate it into our analysis logic.
The final response provided to the user depends on their preferences.

3.3 Server-side Logic and User Preferences

This logic is designed to conduct a comprehensive analysis, contributing to a thorough evaluation of potential threats.

Our server-side logic processes sender information to provide insights that aid users in making decisions when interacting with emails. We assess whether the sender is a first-time contact and whether the user has received emails from the sender's domain previously. If the sender is indeed a first-time contact or if their domain is unfamiliar, we promptly notify the user.

The machine learning models incorporated into the email phishing analysis will be selected based on the user's preferences. Users will have the ability to configure which of the four ML features (content, grammar and spelling, urgency, links) are to be included in the evaluation process, as well as specifying the desired sensitivity or complexity for each analysis, if chosen.

The rationale behind allowing users to configure their preferences is to enhance the usability of our solution. While our models offer higher coverage, they can also potentially slow down the analysis process, or present inaccuracies to the users (unlike the non ML-based features). Thus, by considering the user's preferences, the responsibility for managing this trade-off lies with the user.

3.4 Cache

The implementation of cache within our Chrome extension, has been an important aspect of optimizing user experience and efficiency. We adopted a cautious approach, ensuring that we didn't overly depend on the browser's memory to avoid any inaccuracies in the results.

Our extension needs a token to use the user's Gmail API. After retrieving the token for the first time, we store the token in-memory. By storing it, we eliminate the need for repeated requests to the user for their token, streamlining the adjustment process.

Additionally, we harnessed the cache to enhance our sender analysis capabilities. Prior to querying the API for information about the sender, we first check the browser's stored data. Through this approach we aim to minimize unnecessary API calls and reducing response times.

We also store the user's preferences in the browser's cache, enabling us to retain them for the long term.

3.5 Client-side

3.5.1 User Interface. Our goal is to develop a user-centered tool aimed at enhancing user security and privacy by providing useful information that users may overlook. Therefore, a significant aspect of our project is its usability.

Our extension includes two popups. The first popup enables users to configure their preferences, while the second one displays warnings when the analysis, based on their selected preferences, detects potential issues.

Custom Preferences Selection. In the preferences popup, users can click on a question mark to learn more about each feature and its significance across various options.

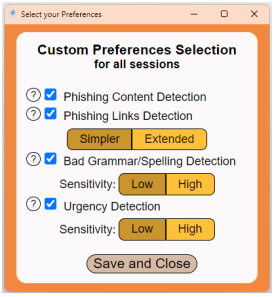

Figure 2: Custom preferences selection pop-up

To encourage users to utilize our available features, we incorporated distinct indicators for cases where they selected at least one feature, as well as for cases where they did not select any features.

Phishing Warnings. In the warnings popup, a warning icon is provided instead of a question mark, which users can click on to receive specific instructions based on the identified findings.

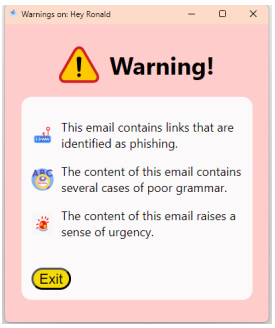

Figure 3: Phishing warning pop-up

One design decision was to display warnings as pop-ups. Most users tend to ignore security warnings[1], and by appearing as a separate window, pop-ups can draw visual focus and stand out

from the rest of the content, ensuring that the message catches the user's eye. Although pop-ups can be attention grabbing, they are also designed to be non-intrusive. Once the user acknowledges the information in the pop-up, it can be dismissed, allowing the user to continue their email interaction without significant interruption.

Another crucial design choice revolved around the frequency of these pop-up notifications. The primary consideration is to strike a balance between effectively alerting users to potential phishing threats and ensuring a smooth user experience. Our main concern was that a constant barrage of pop-ups for harmless emails could lead to user frustration and hinder the overall adoption of the extension.

To achieve this, the decision was made to trigger pop-up notifications selectively, rather than displaying them for every email opened. We have decided to display a pop-up only when the message is in the inbox and is unread, and at least one of the conditions is met: (1) This is the first time the user has received an email from this sender or this email domain, or (2) the content is detected as potential phishing attempt, or (3) a suspicious link was found, or (4) if bad spelling/grammar is detected, or (5) if the mail is found to convey urgency . Please be aware that certain conditions might remain unchecked according to the user's preferences. In such instances, one of the remaining conditions must be met for the pop-up to be displayed.

A few small additions have been introduced to enhance comprehensibility and usability. One of these additions is the inclusion of a title in the pop-up warning, displaying the name of the email. Moreover, we integrated an icon directly into the page to provide real-time feedback on the scanning process. When scanning is complete, it dynamically changes to either a green icon for clear messages or a red icon for suspected phishing attempts, offering users information about the email's security status in a convenient manner.

3.5.2 Content Script Injection. When a user opens an email, our extension's listener springs into action, executing the injection of a content script. This script connects directly to the Gmail API, where it promptly retrieves vital information for a comprehensive analysis. The collected data is then swiftly sent to our server for in-depth analysis. Once receiving a response from the server, a pop-up is seamlessly triggered, instantly displaying the real-time results of the analysis to the user.

3.6 Infrastructure

Our project's infrastructure is built on the foundation of Flask and Google Cloud Platform's Compute Engine.

3.6.1 Flask. Leveraging Flask, a lightweight and flexible Python web framework, we have designed a responsive and efficient backend system. Flask facilitates seamless integration with our frontend components, and ensures smooth communication and data exchange, enabling real-time interactions and quick responses to user actions when they open emails.

3.6.2 GCP Compute Engine. Google Cloud Platform's Compute Engine plays a crucial role in our infrastructure by providing the essential computing resources needed to execute our analysis logic. By leveraging the Compute Engine's offering, we can process incoming emails swiftly and seamlessly through the existing ML models. This enables stable backend performance, allowing our system to deliver real-time responses to potential phishing threats.

3.7 User Study

In the attempt to test our chrome extension's usability, we are suggesting a user study.

3.7.1 Study Design. The study will consist of two sequential stages. The first stage will be an experimental one where some participants will have our extension on their computers, while the second stage will be a survey in which these participants will evaluate the usability of our product based on the metrics defined in 2.2.

3.7.2 Participants. Participants in this study will consist of individuals randomly selected from the general population who use Gmail as their primary email server. Additionally, these participants will have already completed a preliminary survey providing insights into their backgrounds which will be analyzed in a way that defines a diverse group division.

3.7.3 Stage One: Experimental Phase. The initial stage encompasses an experiment focused primarily on evaluating users' compliance to the tool within the context of their private accounts.

Participants will be categorized into two primary groups: those equipped with the extension and those without. Within the extension group, sub-groups will be established, each of equal size. These sub-groups will play a crucial role in addressing the question: whether a predefined optimal configuration for the extension exists or whether allowing users to customize their own configurations proves to be more advantageous.

Configuration Sub-Groups. One sub-group will have the freedom to customize their extension reflecting real-world scenarios where users often seek a personalized balance between security and convenience. This approach acknowledges the significance of user agency in defining their security experiences. However, there exists a potential risk that some users might not configure their extensions optimally, leading to the possibility of missing crucial notifications and perform actions that harm their security and privacy. Other sub-groups will be assigned specific predefined settings, intended to cover various usage scenarios. This will allow us to test various possible combinations of configuration settings with the aim of identifying the most secure and usable combination or combinations of features. (A sub-group division suggestion was attached to the study protocol.)

Scenario Design and Email Simulation. Phishing scenarios, tailored to replicate real-world phishing attempts, along with non-phishing emails, will be sent to the participants equipped with the extension. The scenarios will encompass a diverse range of topics and domains commonly encountered in everyday online interactions. These topics will include post deliveries, university-related matters, studying abroad, and more. The experiment will incorporate emails with and without links, as well as attachments, to realistically simulate the types of emails users might encounter.

Since the extension is configured to notify users when detecting specific indicators, including urgency, bad grammar, phishing content, and first-time senders, we will specifically manipulate the email content and sender information to trigger distinct notifications from our extension during the creation of phishing and legitimate emails.

3.7.4 Stage Two: Survey Phase. In Stage 2, we will focus on evaluating the usability of our extension, by collecting feedback and data from participants who have actively used the extension during Stage 1.

3.7.5 Data Collection.

Stage 1. : Throughout the experiments, all participant behavioral activities, such as link clicking rate, file downloads, email opening rate and reply rate will be methodically logged. Additionally, we will record user-specific configurations each time the extension sends a notification. This data collection will provide valuable insights into user interactions and engagement levels under varying extension settings, providing a comprehensive understanding of the extension's impact in different usage scenarios. The rationale behind collecting email openings is as follows: Email openings serve as the sole indicator that our extension has issued a notification. We will only consider interactions with emails that have been opened, as this indicates user engagement with our extension. Emails that remain unopened will be excluded from the analysis, as they do not demonstrate user interaction with our provided content.

Stage 2. We will gather survey responses from participants who have used the extension during Stage 1. Each metric is associated with one or more questions, and we will calculate scores for compliance, comprehensibility, actionability, and disruptiveness based on participants' Likert scale answers. It's worth noting that while disruptiveness is a component of the extension's actionability, its distinctive significance led us to allocate a dedicated set of questions for this aspect in the study protocol. The sections where participants are asked to provide explanations will provide us with valuable insights into the reasons behind their rating choices. It's important to note that, for simplicity purposes, some questions within each metric have varying Likert scale ratings, where choosing a high rating (e.g., 5) represents the highest score, while in others, it signifies the lowest score. We will consider this variation in ratings when calculating the overall score of each metric, recognizing that the highest and lowest values may differ based on the question's statement.

3.7.6 Analysis and Measures.

Compliance. We define compliance with our extension as the degree to which users adhere to the recommendations associated with each feature. We will evaluate compliance using data from both stages. The first stage will provide us with the following sub-metrics:

Link Clicking Rate: Since our extension detects phishing emails containing harmful links, we expect users to be more cautious when encountering such emails. Monitoring link clicking rates can help evaluate the effectiveness of our extension in encouraging users to avoid potentially harmful links. The rate is defined to be*:

$$\frac{\#\text{ of times a user clicked on a link inside an email with warning}}{\#\text{ of opened emails with warning}}$$

File Downloading Rate: Similar to link clicking rates, the file downloading rate reflects users' caution when it comes to downloading files from potentially suspicious sources. High downloading rates for files attached to phishing emails could indicate that users are not following the extension's advice, while lower rates might suggest compliance which means our extension really helps. The rate is defined to be*:

$$\frac{\#\text{ of times a user downloaded on a file inside an email with warning}}{\#\text{ of opened emails with warning}}$$

Reply Rate: Phishing emails often aim to trick users into responding in a cooperative manner. This metric is essential in assessing how much users engage with phishing emails, even when notified by your extension. It provides insights into whether users are still falling into the common phishing trap of responding to malicious messages. The rate is defined to be*:

$$\frac{\#\text{ of times a user respond to an email with warning cooperatively}}{\#\text{ of opened emails with warning}}$$

* Our primary goal is to assess compliance with the extension. Therefore, the rate definition may be adjusted depending on the selection of the full email set.

We will record all mentioned sub-metrics for the controlled group and compare their overall scores to assess the effectiveness of our extension. These rates are calculated with respect to the opened emails with warnings since we can't make assumptions on the ones they never opened. The scores of the three sub-metrics mentioned will be averaged, and this average will determine each participant's compliance score. The average of all participants' compliance score will represent the compliance score of stage one. The compliance score for the second stage will be calculated as the participants' average compliance score. The overall compliance score will be the average of this metric in both stages. Both stages are crucial for evaluating the overall compliance score. Furthermore, examining each participant's individual compliance throughout both stages will provide insights into the habituation phenomenon. Participants who claimed compliance with the extension but achieved low scores in stage 1 are likely to have developed habituation.

Comprehensibility and Actionability. Individual scores for these metrics will be calculated by averaging their Likert scale test responses. The overall score for each metric will be the average of these individual scores.

4 RESULTS

4.1 Content Model

In this section, we present the outcomes of our extensive evaluation of five distinct machine learning models applied for the analysis of email content. We focus on the top three performing models, namely the Support Vector Machine (SVM), Logistic Regression, and Standard Random Forest. We selected a Support Vector Machine (SVM) model optimized by $F_{\beta=2}$, due to its ability to minimize false negatives while also keeping false positives at a low level.

4.1.1 Logistic Regression

Vectorizer	BOW			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.98	0.98	0.98	0.98
Recall	0.79	0.84	0.88	0.85
F1-Score	0.84	0.86	0.84	0.86
Precision	1	1	1	1
FN/665	0.91	0.88	0.81	0.87
FP	3	5	9	6
Vectorizer	TF-IDF			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.96	0.99	0.98	0.99
Recall	0.42	0.88	0.91	0.89
F1-Score	0.59	0.91	0.86	0.91
Precision	1	0.94	0.81	0.93
FN/665	26	5	3.2	4.87
FP	0	2.3	8	2.88

4.1.2 Standard Random Forest

Vectorizer	BOW			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.97	0.99	0.98	0.99
Recall	0.54	0.88	0.92	0.89
F1-Score	0.7	0.92	0.9	0.92
Precision	1	0.96	0.88	0.95
FN/665	19	4.83	3.56	4.6
FP	0	1.46	5.31	1.88
Vectorizer	TF-IDF			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.97	0.99	0.98	0.99
Recall	0.48	0.86	0.93	0.89
F1-Score	0.65	0.91	0.87	0.91
Precision	0.997	0.95	0.83	0.93
FN/665	21.7	5.55	2.9	4.4
FP	0.05	2	8.2	2.9

4.1.3 SVM

Vectorizer	BOW			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.95	0.97	0.96	0.98
Recall	0.1	0.81	0.91	0.84
F1-Score	0.2	0.8	0.76	0.8
Precision	1	0.8	0.65	0.77
FN/665	37.7	7.9	3.5	6.67
FP	0	8.4	20.4	10.7
Vectorizer	TF-IDF			
Optimized Metric	Accuracy	F1	$F_{\beta=2}$	$F_{\beta=1.2}$
Accuracy	0.97	0.99	0.988	0.99
Recall	0.63	0.9	0.94	0.91
F1-Score	0.77	0.92	0.9	0.92
Precision	0.99	0.95	0.86	0.93
FN/665	16	4.5	2.7	4.0
FP	0.2	2.1	6.5	2.8

4.2 Links Model

In this section, we present the outcomes of our careful evaluation of two models: the more extended version, that incorporates the 14 important features, and the simpler model, that includes only the top 5 important features.

4.2.1 Extended Phishing Links Model Averages

Accuracy	FP	FN	TP	TN
0.94	47	68	901	1171

4.2.2 Simple Phishing Links Model Averages

Accuracy	FP	FN	TP	TN
0.92	87	77	891	1131

5 DISCUSSION

The presented Chrome extension project offers a promising solution to the persistent problem of phishing attacks, particularly focusing on email phishing attempts. The potential broader consequences of this project are substantial and multifaceted.

First and foremost, the project's emphasis on empowering users to recognize and avoid phishing emails, can have a significant positive impact on user security and privacy. This can lead to a reduction in successful phishing attempts, which are often the precursor to data breaches, identity theft, and financial losses.

However, it's important to acknowledge the potential limitations of the current work. While the extension's approach of analyzing email contents and metadata is promising, it might still encounter challenges in accurately identifying sophisticated phishing attempts. Cyber attackers continuously refine their tactics, making it an ongoing challenge to keep up with evolving methods. Moreover, false positives could inadvertently cause users to mistrust legitimate emails, potentially leading to communication breakdowns. Furthermore, the extension's dependency on machine learning models introduces its own set of considerations. For example, while we explored various parameter combinations for the Random Forest model, we found that the default configuration in scikit-learn demonstrated the best results out of the different random forest configurations. However, it is worth noting that further, more profound research may unveil an even more optimized combination. This underscores the potential for continued refinement and enhancement in the pursuit of heightened accuracy and effectiveness in email content analysis.

Further more, the effectiveness of these models heavily relies on the quality and diversity of training data, suggesting that there may be room for improvement in our datasets.

Another potential limitation is that we employed existing machine learning models and customized them for our specific needs, potentially diverging from their original intended use. This approach, while innovative, carries the subtle risk of introducing unforeseen biases or inaccuracies into our system.

Another approach that could be considered is "influence techniques" analysis. Influence techniques refer to specific methods used by phishers to persuade individuals to comply with their requests. Identifying influence techniques used in phishing emails could be an innovative approach for phishing email detection. This can help individuals and organizations to become more aware of the tactics used by cybercriminals to elicit compliance from users.

During our project, we gave our primary focus on a specific article[14] that investigated influence techniques used in phishing attacks and their relative effectiveness. This study highlighted potential content-based characteristics within emails that could serve as indicators of their legitimacy or illegitimacy (consistency, authority, scarcity and more).

One significant challenge we encountered was the absence of a labeled dataset that specifies the influence techniques used in each record. This limitation constrained our efforts to train or fine-tune a model capable of effectively identifying these distinctive traits.
Leveraging zero-shot classification large language models (LLMs), probabilities were assigned to each email, indicating their association with specific characteristics. Yet, it became evident that the LLMs utilized in this endeavor fell short of providing the desired level of accuracy in pinpointing characteristic associations.
Moreover, we tried several methods to describe each characteristic using labels in order to find the label that allows the model to identify the characteristic in the best way possible. However, each label we found was not sufficiently effective when the input to the model was a slightly longer email.

In pursuit of more accurate characterizations, an exploration into leveraging ChatGPT's online API was undertaken. Unfortunately, the employed API did not exhibit substantial proficiency in detecting techniques of influence. Enhancing its performance would necessitate engaging in extended interactions to correct the model, a process that would entail longer conversations. It's evident that applying this approach for every email analysis is neither practical nor sustainable. This observation further emphasizes the need for a targeted model training to fulfill this specific purpose.

Further more, as part of our future work, we plan to publish the extension and pilot our user study.
One methodological limitation of this study lies in its controlled experimental design during Stage One. While the experiment aims to simulate real-world scenarios and user interactions, it takes place within a controlled environment. Participants are aware of the study's purpose and the presence of the extension, which may influence their behavior and decision-making compared to their natural, everyday email usage. This controlled setting may not fully capture the complexities and nuances of user behavior in their typical email interactions, potentially leading to findings that do not fully align with real-world experiences.
To address this limitation, future research could consider conducting long-term, unobtrusive observations of users in their natural email environments to gain a more comprehensive understanding of how the extension impacts their day-to-day email practices. Additionally, assessing user perceptions and behaviors in both controlled and real-world settings may provide a more holistic view of the extension's efficacy and usability.

6 CONCLUSION

"The Phishermen" project represents a valuable step forward in enhancing user security and privacy in the face of persistent phishing threats. The extension's well-rounded approach to analyzing email in multiple aspects provides users with the ability to better identify potential phishing attempts. By incorporating user preferences and selective pop-up notifications, the project strikes a balance between alerting users to threats and ensuring a smooth email interaction experience.

The project's infrastructure, built on Flask and Google Cloud Platform's Compute Engine, provides a robust solution for processing incoming emails and delivering real-time responses to potential threats. The previous work combined with the results presented in this article demonstrate the effectiveness of various machine learning models in identifying phishing emails, providing a promising foundation for future refinements. Conducting the user study procedure in the future will paint the full picture by evaluating not only the efficacy of the extension, but also its usability aspects (actionability, comprehensibility, compliance).

7 CONTRIBUTIONS

Throughout this project, our team maintained a balanced distribution of responsibilities while ensuring we all actively engaged in every aspect. Each of us also took the lead in managing specific areas:
Lilach oversaw research and the technical report.
Abed led the user study protocol and the development of the content ML model.
Hila took charge of the coding, managing both frontend and backend development.

This approach allowed us to collectively contribute to every facet of the project, ensuring a well-rounded solution.

REFERENCES

[1] Devdatta Akhawe and Adrienne Porter Felt. 2013. Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness. (2013).
[2] Kholoud Althobaiti, Nicole Meng, and Kami Vaniea. 2021. I Don't Need an Expert! Making URL Phishing Features Human Comprehensible. (2021).
[3] Ram B. Basnet, Andrew H. Sung, and Quingzhong Liu. 2014. Learning to Detect Phishing URLs. (2014).
[4] Vaibhav Bichave. 2022. Phishing-URL-Detection GitHub Repository. (2022). https://github.com/VaibhavBichave/Phishing-URL-Detection/
[5] Marcus Butavicius, Ronnie Taib, and Simon J. Han. 2022. Why People Keep Falling for Phishing Scams: The Effects of Time Pressure and Deception Cues on the Detection of Phishing Emails. (2022).
[6] Facebook. n.d.. BART Large (MNLI) Model. (n.d.). https://huggingface.co/facebook/bart-large-mnli
[7] Iulia Ion, Rob Reeder, Google, and Sunny Consolv. 2015. "...no one can hack my mind": Comparing Expert and Non-Expert Security Practices. (2015).
[8] Kaggle. 2016. Enron Email Dataset. (2016). https://www.kaggle.com/datasets/wcukierski/enron-email-dataset
[9] Gregory D. Moody, Dennis F. Galletta, and Brian Kimball Dunn. 2011. Which Phish Get Caught? An Exploratory Study of Individuals' Susceptibility to Phishing. (2011).
[10] Elissa M. Redmiles, Noel Warford, Amritha Jayanti, Aravind Koneru, Sean Kross, Miraida Morales, Rock Stevens, and Michelle L. Mazurek. 2020. A Comprehensive Quality Evaluation of Security and Privacy Advice on the Web. (2020).
[11] M. S. Sadat. 2022. Phishing Email Dataset. (2022). https://github.com/sadat1971/Phishing_Email/tree/main/Data
[12] Gunikhan Sonowal. 2021. *Phishing and Communication Channels: A Guide to Identifying and Mitigating Phishing Attacks.*
[13] Vennify. n.d.. T5 Base Grammar Correction Model. (n.d.). https://huggingface.co/vennify/t5-base-grammar-correction?text=grammar
[14] Ryan T. Wright, Matthew L. Jensen, Jason Bennett Thatcher, Michael Dinger, and Kent Marett. 2014. Influence Techniques in Phishing Attacks: An Examination of Vulnerability and Resistance. (2014).