TrustME: A Lexical Approach to Phishing Detection

Jeremiah Johnson, Ronit Motwani, Jundong Zhang
Group Name: Phish and Chips
joh18813@umn.edu, motwa011@umn.edu, zhan7569@umn.edu
Mentor: DK and Robert

Abstract

In this project, our aim is to present an approach to the widespread problem of phishing. We proposed a novel method for analyzing the commonality and readability of words to determine the probability that the message is a phishing attempt on a given text without compromising privacy. Our approach creates a foundation for further research and improvement.

Keywords: Cybersecurity, Natural Language Processing (NLP), Phishing Detection, Lexical Analysis

1 Introduction

The widespread use of online communication platforms, such as emails and text messages, has created opportunities for malicious actors to distribute deceptive messages on a massive scale. Traditional phishing detection models rely heavily on techniques such as keyword detection, hyperlink verification, and machine learning classifiers such as RoBERTa (Liu, 2019). However, these approaches often overlook deeper lexical patterns inherent in text-based phishing messages. Inspired by the theory that phishing emails are crafted using simpler language and commonly used words to appeal to a broader audience, our research explores a novel approach based on text readability and word commonality analysis. By examining how easily a message can be read and how frequently its words occur in standard language use, we aim to create a more comprehensive phishing detection system that addresses gaps in existing methods.

1.1 Research Problem and Objective

Phishing messages are intentionally crafted to deceive users by blending into legitimate communication. Existing models excel at detecting obvious signs like suspicious links and known phishing terms, but they might struggle when such explicit features are absent. Our project introduces a new perspective by hypothesizing that fishing messages tend to favor more common words and easier-to-read sentences to maximize their potential reach. We developed a detection framework that combines two critical features: word commonality based on Zipf scores and readability scores calculated through established text readability evaluation formulas. By integrating these metrics into a unified scoring system, our goal is to detect phishing messages that use this unique analysis vector.

1.2 Current Method and Limitations

Current phishing detection research has focused extensively on machine learning models and structural analysis. Researchers such as Uddin and Sarker (2024) explored transformer-based models, while Çolhak et al. (2024) demonstrated that combining AI-driven text and HTML structure analysis improves detection. However, these models often rely solely on surface-level features, such as explicit phishing keywords and malicious link detection. This leaves a critical gap in detecting messages designed to appear genuine through subtle text manipulation. Our research proposes a more nuanced approach by incorporating lexical patterns, emphasizing readability and word commonality as previously unexplored dimensions in phishing detection.

1.3 Impact and Relevance

Our work has significant implications for advancing phishing detection systems by offering an innovative lexicographical analysis layer. Cybersecurity professionals, developers of secure communication systems, and email service providers could integrate our proposed approach into their existing frameworks. The potential to detect phishing messages based on readability and linguistic patterns adds a new dimension to message screening processes. By focusing on how messages are constructed rather than solely on what they contain, our research highlights the untapped potential of text-based analysis in strengthening online security and mitigating the risks posed by phishing attacks while protecting message confidentiality.

2 Approach

2.1 Procedures

An overview of our procedures is shown in the Figure 1.

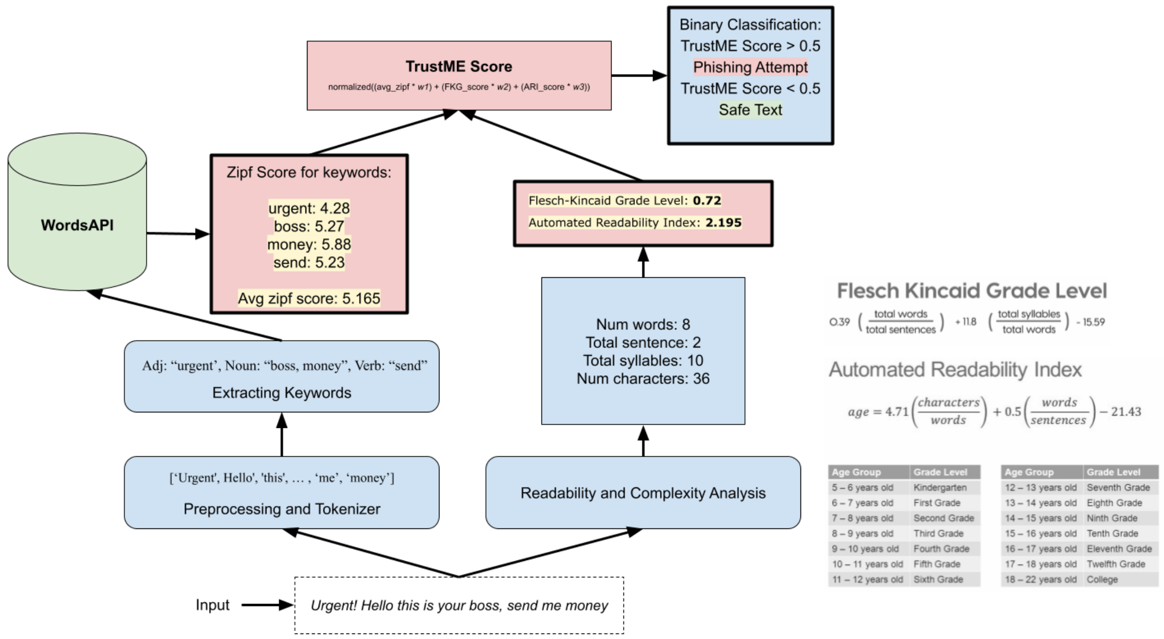
2.1.1 Initial Idea and Re-Approach

Our initial approach aimed to develop a contextual understanding of phishing messages by leveraging word relationship data from external APIs. The idea was to extract contextual associations between words, enabling a deeper semantic analysis of message content. However, this approach faced significant limitations due to insufficient relational data provided by available APIs such as ConceptNet and DBpedia. This restricted our ability to build a comprehensive knowledge-base model. Recognizing these constraints, we revisited the core intent behind phishing messages – to deceive as many users as possible through easily digestible and widely understood language (van der Laan, 2021). This reflection led us to formulate a new theory: phishing messages are likely crafted using simpler sentence structures and more commonly used words to maximize their reach. Shifting our focus we designed the detection system that combines readability scoring, which evaluates how easily a message can be understood, with word commonality analysis based on frequency metrics found in WordsAPI. This re-approach enabled us to explore phishing detection through a unique lexical lens, grounded in the fundamental principles of language accessibility and deceptive communication tactics.

2.1.2 Commonality

Our initial hypothesis was that since phishing emails cast a wide net to attract the largest number of people possible, these types of messages would select more common words that a large group users could understand (van der Laan, 2021). To produce a commonality score, we extracted keywords from the message using Python's spaCy Library, then used an API to WordsAPI, which is an online knowledge base that holds information about words, including but not limited to their definition, examples, and frequency of use. We retrieved the Zipf score of the words, which is a logarithmic score for how often that word appears. A score of one would mean the word appears once every hundred million words, whereas a score of two would be once every ten million words, and so on.

We averaged the Zipf scores of the extracted keywords to produce an average score for the sentence, assuming that if our hypothesis was true, we would


Binary Classification:
TrustME Score > 0.5
Phishing Attempt
TrustME Score < 0.5
Safe Text
TrustME Score
normalized((avg_zipf * w1) + (FKG_score * w2) + (ARI_score * w3))
WordsAPI
Zipf Score for keywords:
urgent: 4.28
boss: 5.27
money: 5.88
send: 5.23
Avg zipf score: 5.165
Flesch-Kincaid Grade Level: 0.72
Automated Readability Index: 2.195
Num words: 8
Total sentence: 2
Total syllables: 10
Num characters: 36
Adj: "urgent', Noun: "boss, money", Verb: "send"
Extracting Keywords
['Urgent', Hello', 'this', … , 'me', 'money']
Preprocessing and Tokenizer
Readability and Complexity Analysis
Input
Urgent! Hello this is your boss, send me money

Flesch Kincaid Grade Level

$$0.39\left(\frac{\text{total words}}{\text{total sentences}}\right) + 11.8\left(\frac{\text{total syllables}}{\text{total words}}\right) - 15.59$$

Automated Readability Index

$$age = 4.71\left(\frac{characters}{words}\right) + 0.5\left(\frac{words}{sentences}\right) - 21.43$$

Age Group	Grade Level
5 – 6 years old	Kindergarten
6 – 7 years old	First Grade
7 – 8 years old	Second Grade
8 – 9 years old	Third Grade
9 – 10 years old	Fourth Grade
10 – 11 years old	Fifth Grade
11 – 12 years old	Sixth Grade

Age Group	Grade Level
12 – 13 years old	Seventh Grade
13 – 14 years old	Eighth Grade
14 – 15 years old	Ninth Grade
15 – 16 years old	Tenth Grade
16 – 17 years old	Eleventh Grade
17 – 18 years old	Twelfth Grade
18 – 22 years old	College

Figure 1: Procedure

see the phishing messages have a higher average Zipf score than the nonphishing messages.

2.1.3 Readability

We think that for a phishing attempt to be successful, it needs to be simple and straightforward enough to catch the receiver's attention at the very beginning. Therefore, we hypothesize that the readability is negatively correlated with the probability of a text being a phishing attempt.

To evaluate the readability of a given text, we measured the complexity of the text by the following 5 indexes:

1. Total words of the given text;
2. Average syllables per word of the given text;
3. Average characters per word used in the given text;
4. Automated Readability Index (ARI) (Smith and Senter, 1967);
5. Flesch-Kincaid grade (FKG) (Solnyshkina et al., 2017).

The ARI and FKG index are similar indicators of text complexity. The numbers indicate the school level education that is required to understand the text. For instance, an index of 11 suggests a high school junior year level student's readability level.

2.1.4 Commonality and Readability Scores Integration

We conducted an exploratory evaluation by systematically testing various weight configurations to understand their impact on our model's performance. Specifically, we iteratively adjusted the weights with a 0.05 steps at each iteration for the Commonality, ARI, and FKG scores (e.g., weight for commonality $= 0.05$, weight for ARI $= 0.05$, and weight for FKG $= 1 - 0.05 - 0.05 = 0.9$). However, as we did not find any significant changes in the overall performance of the model with different weights. Therefore, we decided to implement a most explainable model by evenly assign weights to commonality and readability scores.

In the final model, we assigned a weight of 50% to the commonality score, 25% to the ARI metric, and 25% to the FKG metric. These weights were selected based on their ability to maximize the alignment of the model's predictions with the desired outcomes. Using these weights, we defined a composite **TrustME** score, calculated as follows, to predict the trustworthiness of a given text:

$$\begin{aligned} TrustME =& 0.5 \times Commonality \\ &+ 0.25 \times ARI + 0.25 \times FKG \end{aligned}$$

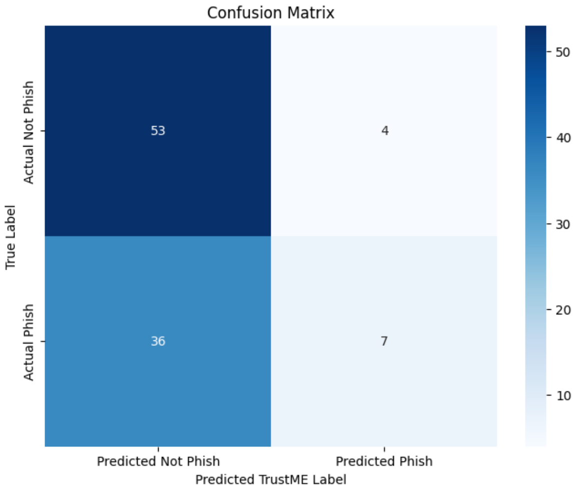


Figure 2: TrustME Confusion Matrix at 0.5 Threshold

2.1.5 Potential Integration of RoBERTa Model

Our initial idea was to use a RoBERTa model (Liu, 2019) to do an initial sweep through the data to have an initial classification on the data, then go back through the classified phishing data and do a content based analysis to focus on the false positives that the RoBERTa made more often than false negatives. However, we thought that this would be relying too much on an existing model and would become an accessory function for existing models rather than a novel approach on how to detect phishing messages, so we focused on developing our own model that could tread a new path for the detection of phishing messages without needing to take in the actual content of a message that could potentially be leaked and pose a security risk.

2.2 Challenges

2.2.1 Commonality

We ran into limitations with our API, as we were only allotted twenty-five thousand API requests per day on our current paid plan. This forced up to both limit the sample size of the messages we could use, and also limited the amount of keywords we could request per message in order to fit the limit of our current WordsAPI plan. Running our model multiple times while tweaking the code also contributed heavily to our limit as well. Additionally, the parsing of the message and extraction of the keywords made it difficult for us to run a robust sample size without our application timing out and losing progress.

2.2.2 Readability

Considering that calculating readability of a given text is relatively straightforward and simple with the python package *nltk*, we did not anticipate to encounter any kind of challenges and difficulties during the implementation and calculation procedure itself. And we indeed got the readability scores smoothly.

2.3 Novelty of approach

2.3.1 Commonality

Our method introduces a novel approach to detecting phishing messages by leveraging Zipf's Law to analyze the linguistic simplicity of phishing messages. By extracting keywords and retrieving their Zipf scores, we produce an efficient, simplistic, and understandable metric for a message, rather than computationally heavy methods like sentiment analysis or semantic embedding. By using this linguistic generalization, it could reveal potentially unknown patterns about phishing messages that can be explored further

2.3.2 Readability

Current language models, such as BERT (Devlin, 2018), RoBERTa (Liu, 2019), and Llama (Touvron et al., 2023), focus more on catching phishing attempts via semantic understanding. However, we believe that there are other ways that phishing texts share. Specifically, from a social engineering perspective, those fraudulent attempts are filtering at the same time as phishing. As they intend to keep the text as simple as possible to target those with lower awareness of fraudulent attempts. Therefore, besides the semantic meaning of the text itself, we would like to utilize the readability scores of the text to see whether we can predict whether a given text is a phishing attempt or not.

3 Experiment Results and Error Analysis

3.1 Experiment Results: Commonality and Readability Scores Integration

The way we measured success was fairly simple: whether our lexical analysis, combining commonality and readability scores together, correctly determined whether the message was phishing or not against the ground truth label of the message. We wanted to answer whether the phishing messages used more common language than non-phishing messages and whether we can detect it.

However, contrary to our prediction, our model was unable to reliably create a distinction in the word choices and readability of the phishing messages versus the non-phishing messages. When tested, we ran with a threshold of 0.5 since the score was normalized on a scale of zero to one. However, we leaned far more towards classifying messages as non-phishing. When we lowered the threshold down to around 0.33, we began to see the classification be evenly split up between phishing and non-phishing, but our accuracy decreased due to this as well, since non-phishing messages began to be classified as phishing.

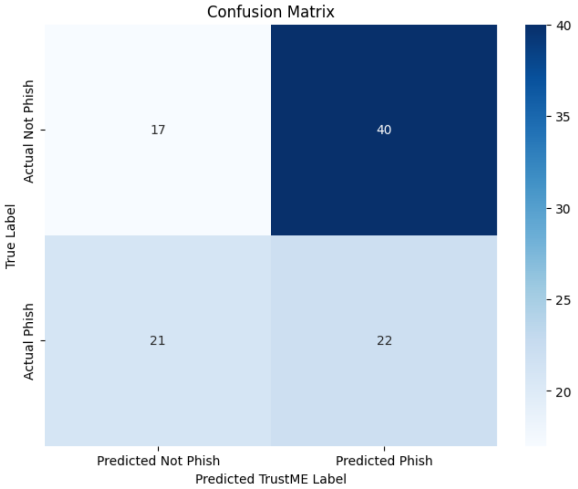


Figure 3: TrustME Confusion Matrix at 0.33 Threshold

3.2 Error Analysis

3.2.1 Commonality Error Analysis

Initially, we ran the Zipf score with each keyword extracted from the message. With the longer messages, we potentially ended up with dozens of keywords we were sending through the API. We found that the more keywords we used for the average Zipf scores, the more the averages converged for all messages and we were unable to have a true distinction between any samples. To address this problem, we began only using the top n longest keywords in an attempt to get the most unique keywords for each message, and therefore to be able to distinguish the average Zipf scores more clearly. We had initially started with thirty keywords, eventually reducing to twenty, ten, and then five keywords, where we finally started to see a bit of divergence within the average Zipf scores per message, even if not within phishing/non-phishing messages. Additionally, while running the API, if there were words that were misspelled or not present within the WordsAPI database, those words would not return a Zipf score, which would count as a zero. This would skew the score inversely to our hypothesis, resulting in a non-phishing classification even though misspelling are a core identifier of phishing attempts. Unfortunately this was a factor we could not account for with the use of the WordsAPI database.

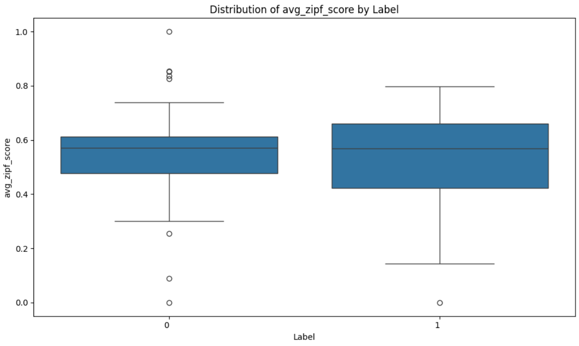


Figure 4: Normalized Zipf Score Distribution

3.2.2 Other Factors

There are potentially some minor issues with our dataset. First, the dataset we used did not have niche business topics that we imagined would have the lower Zipf scores since they would be used, since the data did not explicitly define where it was gathered from.

Second, the dataset we used in this study does not have the highly specific technical language that we were looking for to determine the phishing attempt. For instance, if one text mentions that "please use **xxx** tool to send the money since your account is locked" then this specific tool name would be a red flag.

3.3 Potential Integration of RoBERTa

We first tried to fine-tune a RoBERTa model for phishing text detection. Specifically, we fine-tuned the RoBERTa model with the Phishing Dataset (ealvaradob, 2024). This is a dataset containing URL, SMS messages, Email messages, and HTML code. However, since our study focuses on identifying phishing texts, we only fine-tuned the RoBERTa model with SMS and Email messages.

The training process is shown in Table 1. The entire finetuning took over three epochs. As shown in the table, the training loss decreased significantly from epoch 1 to epoch 3, showing effective learning of the model during training. However, the validation loss drops in epoch 2 but increases slightly in epoch 3. This might indicate slight overfitting, as the model performs well on the training data but struggles slightly more on unseen validation data in epoch 3. As for the accuracy, it shows an upward trend, indicating improved classification performance across epochs. In addition, the precision remained high across epochs, indicating that the model performed consistently well. Lastly, the recall rate improved significantly from Epoch 1 to Epoch 2 and decreased slightly in Epoch 3. Therefore, we decided to stop the fine-tuning process at step 3 and use the model from then on.

Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall
1	0.129	0.112	0.973	0.994	0.935
2	0.040	0.053	0.989	0.982	0.990
3	0.011	0.066	0.990	0.991	0.983

Table 1: Process of Fine-tuning RoBERTa Model

To have a better understanding of the performance of your fine-tuned RoBERTa model, we asked ChatGPT to generate a new dataset with both phishing and nonphishing texts. And the performance is shown in Figure 5.

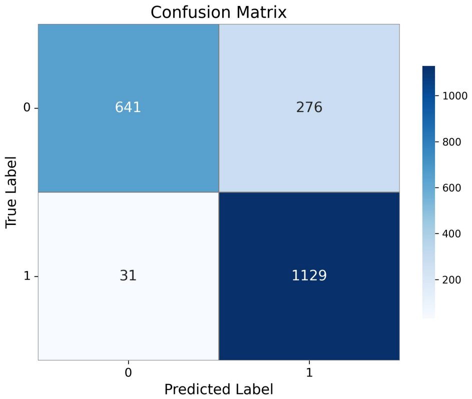


Figure 5: Confusion Matrix of Fine-tuned RoBERTa Model with GPT Generated Dataset

As shown in Figure 5, the performance of the fine-tuned RoBERTa model generally works well. However, we have slightly more false positive cases, where some nonphishing texts were identified as phishing texts.

To improve the performance, we considered combining the confidence scores generated by the fine-tuned RoBERTa model together with the lexical scores generated from the previous steps. To determine the weights of different weights, we applied a logistic regression. Specifically, we assigned weights as follows:

- Coefficient: -0.567881
- RoBERTa Confidence Score: 2.993977
- Average Syllables Per Word: 0.504654
- ARI: 0.178134
- Total Words: 0.104803
- Commonality Score: 0.072311
- Adjusted Score: -0.072311
- Average Zipf Score: -0.166475
- Average Characters Per Word: -0.219793
- FKG: -0.239904

After adding all the scores based on their weights, we applied the initial score with the following formula to make a prediction with a threshold at 0.999:

$$P(Phishing) = \frac{1}{1 + e^{-InitialScore}}$$

However, the performance was not as expected. Among the 100 examples we used, only 52 out of 100 were successfully identified by this method.

4 Discussions and Conclusion

4.1 Replicability

Our research framework is designed with replicability in mind, leveraging widely available tools, publicly accessible datasets and APIs. The model's implementation relies on standard machine learning libraries, including Hugging Face's Transformers for RoBERTa fine-tuning and common readability scoring algorithms and word frequency retrieval. Our methodology is clearly defined, with step-by-step processes for data preparation, feature extraction, and score aggregation. Future researchers can reproduce our results by following these procedures, adjusting parameters, or applying the model to new datasets and gain better scores.

4.2 Datasets

Our research utilized a dataset consisting of general messages, covering a wide range of topics typically encountered in everyday online communication. While this dataset provided valuable insights into common linguistic patterns in phishing and benign messages, it limited our ability to explore more specialized phishing tactics aimed at specific industries or professional contexts. A more targeted dataset focusing on business-related messages, such as corporate emails or financial correspondence, could enhance the applicability of our model by exposing it to niche vocabulary and context-specific phishing strategies. Future research could benefit from developing or sourcing such domain-specific datasets, enabling a deeper understanding of how lexical features manifest in professionally oriented phishing attempts. This adaptation could inspire new lines of research in business email compromise (BEC) detection and tailored phishing prevention systems.

4.3 Ethics

While our model processes textual data by extracting keywords and calculating scores based on word commonality and readability formulas, it inherently carries a potential privacy risk. Analyzing message content could, in theory, expose sensitive information if messages were stored or transmitted insecurely. However, our approach minimizes this risk by focusing solely on lexical features rather than the message's actual content. The model processes the text locally, extracts relevant scores, and discards the original message, ensuring that sensitive information is neither retained nor shared. Since we operate on abstracted numerical representations rather than raw text, the possibility of data leakage is significantly reduced. To further mitigate privacy concerns, future implementations could enhance security by applying encryption protocols, processing messages entirely on-device, or anonymizing data before analysis. This ensures that our research remains both technically sound and ethically responsible.

4.4 Conclusion

While our model offers a novel approach to phishing detection through lexical analysis, it faces certain limitations due to the evolving nature of phishing messages. Historically, phishing emails were often poorly constructed, featuring grammatical errors, awkward phrasing, and low lexical sophistication—traits our model was designed to detect through readability and word commonality scoring. However, the rise of advanced LLMs capable of generating highly coherent and convincing phishing messages challenges the effectiveness of our approach. As these models become more accessible, phishing attempts may increasingly exhibit professional writing quality, making lexical analysis alone insufficient. To address this, future research could integrate AI-generated text checkers into our TrustME scoring system, adding an adaptive layer that assesses the likelihood of a message being machine-generated. This would strengthen the detection process by balancing lexical evaluation with modern AI-driven checks, ensuring that the scoring system remains robust against more sophisticated phishing tactics.

References

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

ealvaradob. 2024. ealvaradob/phishing-dataset · Datasets at Hugging Face.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Edgar A Smith and RJ Senter. 1967. *Automated readability index*, volume 66. Aerospace Medical Research Laboratories, Aerospace Medical Division, Air Force Systems Command.

Marina Solnyshkina, Radif Zamaletdinov, Ludmila Gorodetskaya, and Azat Gabitov. 2017. Evaluating text complexity and flesch-kincaid grade level. *Journal of social studies education research*, 8(3):238–248.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Mohammad Amaz Uddin and Iqbal H. Sarker. 2024. An explainable transformer-based model for phishing email detection: A large language model approach.

Jacob Jan van der Laan. 2021. The semantics of persuasion: a case study using phishing emails.

Furkan Çolhak, Mert Ecevit, Hasan Dağ, and Reiner Creutzburg. 2024. Securereg: Combining nlp and mlp for enhanced detection of malicious domain name registrations. pages 1–6.