**Development of an AI-Driven Phishing Detection System Using Advanced NLP and ML Techniques**

CSI 4900 – Honours Project

2024

University of Ottawa

Professor: Mohamad Hoda

Name – Student number

Date

# Table of Contents

[1. Introduction 3](#_Toc183546214)

[2. Methodology 4](#_Toc183546215)

[c. Explainable AI (XAI) 4](#_Toc183546216)

[3. Related Studies 6](#_Toc183546217)

[3. Model Testing and Comparison 7](#_Toc183546218)

[5. Application Development 8](#_Toc183546219)

[6. Conclusion 9](#_Toc183546220)

[7. References 10](#_Toc183546221)

# 1. Introduction

# 2. Methodology

## c. Explainable AI (XAI)

Explainable AI or XAI “is a set of processes and methods that allows human users to comprehend and trust the results and output created by machine learning algorithms” (IBM Corp, n.d.). Models in artificial intelligence are often considered a black box, where information is given as input, and a prediction is made after some time. This lack of explainability, interpretability, and reproducibility poses a challenge for us. As humans, when we explain a phenomenon or a concept, we usually try our best to accompany our conclusion with explanations to help others understand how we came to a given decision. However, with AI models, is hard to understand how they come up with their results. To avoid this problem, models need to be explainable in the sense that the underlying workings of a machine-learning model are described in human language. It must also be interpretable. This means it must show the capacity to recognize how inputs and outputs relate to one another and estimate how the inputs will react when they change. Models must be reproducible as well by indicating their capacity to consistently replicate a model’s output using the same inputs (Cloudera Inc, 2024).

Knowing how impactful XAI can be, it should come as no surprise that it could be a powerful tool to use for phishing detection. Many of the currently available products marketed to detect phishing emails and phishing links simply take in an input, and give an output of phishing or not phishing. We, therefore, believed that it would be important to incorporate XAI into our product to promote a better understanding of why any email or link is labeled the way it is. Explainability in phishing detection is beneficial for many other reasons. Namely, when a system explains its choices, users are more inclined to trust it. The ability of a phishing detection model to explain why an email is deemed suspicious increases the trust in the technology. It can also assist in detecting biases and mistakes in the model’s prediction, which enables constant improvements to the detection methods, and it promotes AI transparency as laws regarding data protection become stricter (Restack, 2024).

Of the many different XAI tools, SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-Agnostic Explanations) are some of the most popular and most useful for phishing detection. At its core, SHAP is a technique for providing context for machine learning models by giving each feature a value that corresponds to a specific prediction. It evaluates every potential feature subset to determine a feature’s contribution by comparing predictions made with and without that feature. It also ensures that every prediction is covered, and it will favor features that increase the model’s accuracy and relevance, which aligns with how the model behaves at a global and local level (Bak, 2024). To better understand, let’s imagine we want to understand why a friend chose a specific restaurant. SHAP will look at all past dining decisions and calculate how much each factor such as cuisine, location, and price contributed to the choice.

LIME, on the other hand, provides local explanations that help users understand why a model makes specific predictions. It works by randomly modifying features of the input data and observing how the model's predictions change. By comparing the modified predictions with the original, LIME determines the relative importance of each feature to the final decision. These modifications are weighted by their similarity to the original input to ensure that the explanation is relevant to that specific instance (Bak, 2024). Going back to the example of understanding why a friend chose a specific restaurant, LIME will only look at the current decision and explore how modifying factors such as lowering the price, or changing the location would affect the choice.

Another technique to enhance the explainability of models is the use of case-based reasoning (CBR). This technique aims to evaluate new instances by comparing them with previously known cases stored in a database. Instead of relying on a defined model such as Logistic Regression, CBR works on the idea that similar problems should have similar solutions. This is particularly useful in link analysis, where we can compute the cosine similarity of a link to a database of phishing and benign links. This solution contributes to explainable predictions by showing the most similar cases and their labels.

## d. File Analysis

Embedded macros have come to be as a major threat, especially in Microsoft Office files. These files use Visual Basic for Applications (VBA) scripts, which are designed to automate repetitive tasks in spreadsheets, presentations, and documents. However, because of their flexibility, fraudsters can easily take advantage of them. By inserting dangerous code inside the macro of an apparently innocent document, macro malware exploits this functionality. These files, which are often distributed via phishing emails, frequently pose as shipment notifications, bills, or legal papers in order to trick naïve users. In order to avoid Microsoft Office's default option, which prevents macros for security reasons, the virus usually adds warnings or false warnings after the document is launched, pushing the user to activate macros. Despite improvements in security, macro malware is still a major threat, they are a clear red flag that shouldn’t be ignored.

The measure of unpredictability in a file, known as file entropy, is another important tool for detecting potentially harmful files. It calculates the level of data unpredictability or compression in a file, generating a number between 0 and 8. Data that can be easily compressed and is predictable or repeated tends to have a lower entropy level. Higher numbers, on the other hand indicate unpredictability and are frequently linked to compressed, packed, or encrypted files. To hide their code, for instance, malware developers usually use packing or encryption techniques, which greatly increases entropy values and raises red flags suspicions (IBM Corp, 2024). According to a study of approximately 500,000 malicious and benign files, harmful documents are more likely to have levels above 7.2, while legitimate files generally fall within the 4.8 to 7.2 range (Pracsec, 2019). This shows that file entropy is a powerful predictor of potentially suspicious activity.

Another important concept to consider for file analysis is the presence of embedded JavaScript in PDFs which poses significant security risks. JavaScript can be nested into these types of files to change the text of pages, run code, or communicate with the user’s computer. Even though it can be used in good faith, its use can also present serious security problems. Malicious actors can take use of this capacity to insert malicious scripts into particular parts of the PDF structure, frequently using flaws in PDF readers to execute unauthorized operations when the file is accessed. The Catalogue Object, which is a PDF document’s root structure is a common area for JavaScript embedding. Attackers can set up the document to run embedded scripts automatically when it is opened by using OpenAction entries (O'Neill, 2024). These scripts can compromise a system's integrity by downloading malware or modifying system files.

A crucial technique in cybersecurity and digital forensics is hash analysis, and it compares the cryptographic hash values of files to a database of known harmful hashes to confirm its status. A unique digital fingerprint created by techniques such as MD5, SHA-1, or SHA-256 is called a file hash. Even small changes to a file can cause these hashing algorithms significantly change the fixed length character string of the hash. Because of this characteristic, hash values provide a reliable way of identifying and verifying the integrity of files.

# 3. Related Studies

# 3. Model Testing and Comparison

# 5. Application Development

# 6. Conclusion

# 7.

# References

Bak, U. (2024). *Techniques for Explainable AI: LIME and SHAP*. Retrieved from UnnatBak: https://www.unnatbak.com/blog/techniques-for-explainable-ai-lime-and-shap

Cloudera Inc. (2024). *Model explainability, interpretability, and reproducibility*. Retrieved from Cloudera Documentation: https://docs.cloudera.com/machine-learning/cloud/models/topics/ml-model-explainability-interpretability-reproducibility.html

IBM Corp. (2024). *Analyzing files for embedded content and malicious activity*. Retrieved from IBM: https://www.ibm.com/docs/en/qsip/7.5?topic=content-analyzing-files-embedded-malicious-activityhttps://www.ibm.com/docs/en/qsip/7.5?topic=content-analyzing-files-embedded-malicious-activity

IBM Corp. (n.d.). *What is explainable AI?* Retrieved from IBM: https://www.ibm.com/topics/explainable-ai

O'Neill, B. (2024). *Understanding Embedded JavaScript in PDF Documents*. Retrieved from Cloudmersive: https://cloudmersive.com/article/Understanding-Embedded-JavaScript-in-PDF-Documents

Pracsec. (2019). *Threat Hunting with File Entropy*. Retrieved from https://practicalsecurityanalytics.com/file-entropy/

Restack. (2024). *Explainable AI for Phishing Detection*. Retrieved from Restack: https://www.restack.io/p/explainable-ai-answer-phishing-detection-cat-ai