Malicious URL Detection

CS549 Fall

Maksym Yelisyeyev, Nghi Tran, Wasay Zaman

---

1. Abstract

This project focuses on developing machine learning models that identify malicious URLs derived from raw URL strings from the provided Kaggle dataset. This dataset will undergo preprocessing methods such as feature extraction and handling missing values. The models used in the project will be SVM, XGBoost, and Logistic Regression, and compared using metrics like Accuracy, Precision, Recall, and F1-score. The goal is to identify the best-performing model for distinguishing malicious URLs. On a larger scale, this is essential for the future of cybersecurity and enables early prevention of phishing, malware, and other bad actors who use URLs as a method of their malicious activity.

2. Introduction

Malicious websites are constantly forcing users to double-check their URLs and live with the fear of their information being stolen. Detecting such URLs has become a more prevalent problem because it's complicated to do so in a timely manner and with high accuracy, with scammers frequently masking malicious URLs as benign websites. Traditional blacklist and content-based methods of malicious URL detection are struggling to keep up with newly generated URLs and require heavy computation. Nowadays, machine learning opens a new approach to this problem by identifying statistical and structural patterns within a URL, distinguishing benign links from malicious ones.

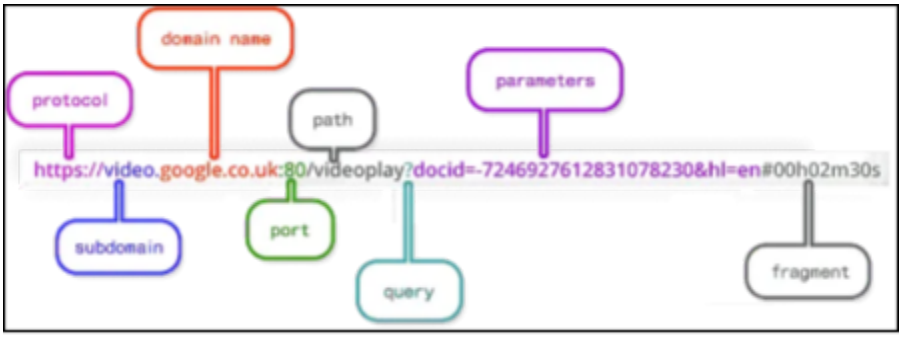

Figure 1 - Complex URL Structure

A URL (Uniform Resource Locator) consists of several components - a protocol, subdomain, domain name, etc. (Figure 1). Each of these components can give us clues about the content of the website. For instance, malicious

URLs tend to be longer, incorporate more digits, and often use unusual domain structures. Such patterns can be turned into meaningful data through feature extraction, transforming each URL into numerical values that capture properties like length, entropy, protocol type, and many more. By focusing solely on these values our project avoids the need to fetch website data and analyze website content, giving us more speed, making the prediction process safer and computationally efficient.

For training and evaluating our models, we will use a publicly available Kaggle Malicious URLs Dataset. It's a reliable and widely used dataset, with a lot of documentation and usability, which provides a balanced mix of benign and malicious URLs, allowing us to comparatively assess performance across multiple learning algorithms. Data from this set will be preprocessed, removing inconsistencies, handling missing values, and standardizing feature scales before modeling.

The goal of this project is to develop, train, and compare multiple machine learning models that use lexical and structural features to identify malicious URLs. SVM, Logistic Regression, and XGBoost will be implemented and evaluated to determine which model offers the best trade-off between metrics like accuracy, precision, recall, and computational efficiency. Through this study, the project will demonstrate that well-implemented lexical features, when used alongside learning algorithms, can effectively detect malicious URLs without relying on external content analysis.

3. Methodology

3.1 Data Preprocessing

Two main data preprocessing methods will be used:

Feature Extraction/Encoding

Since ML Models can't directly interpret raw URLs, we are going to transform URLs into a numerical format with their lexical and structural characteristics. Lexical features can tell us different patterns within the data, such as length, number of digits, or entropy, that we will use to differentiate malicious URLs from the valid ones. As an example, malicious URLs can be longer in length, contain more digits, or have higher entropy.

We will extract a set of features from a URL without accessing the webpage content, ensuring that in case we're handling a malicious URL, the process remains safe. Features planned for extraction include:

Length-based features: total URL length, host length, path length, query length.

Count-based features: number of subdomains, number of digits, number of "special" symbols (e.g., -, _, @, ?, %, =, ...)

Ratio features: digit-to-length, symbol-to-length, and uppercase-to-length.

Entropy: Shannon entropy to measure the randomness of characters.

Categorical features: protocol type (HTTP vs. HTTPS) and top-level domain (e.g., ".com", ".net", ".xyz") converted into numerical form through encoding.

All these features will be normalized and standardized so that all values will fall into comparable ranges.

Handling Missing Values

Missing data points for specific variables in a dataset could affect accuracy & reliability. They can reduce sample size, create biases in results, which leads to difficulties in analysis. Common missing values are represented as blank cells, specific values "Na", "NaN", "Unknown"..., codes or flags,... Some strategies to approach this issue are:

Removing rows with missing values

Imputation methods: filling the blanks with estimated values. This could be the mean, median, mode, the nearest data from the same columns, …

Interpolation methods: estimate missing values based on the values of surrounding data points

Using these strategies, we would be able to improve data quality, enhance model performance, preserve data integrity, and reduce biases

3.2 Model Development

We will use Supervised Learning across all models. This is because we have a dataset that lists specifically whether the data is labeled as malicious or benign, and supervised learning can learn from this given data. It's more effective than the model learning without any prior context/labels.

Logistic Regression:

Logistic Regression will be one of the main supervised learning frameworks used to recognize malicious URLs. It is a statistical classifier that estimates the probability that a given instance belongs to a particular class, in our case whether a URL is malicious or benign, based on a logistic (sigmoid) function. Logistic Regression is appealing for binary classification tasks which is clearly appropriate for the current task.

This model was selected because it is easy to interpret, represents a simple computation, and will provide a strong baseline for us to compare with more complex models. The coefficients learned by the model will help us understand which features of the URL are most important in determining malicious behavior by looking at lexical or structural features of the URL (length of the URL, number of digits in the URL, or special characters). Logistic Regression also has good performance when working with linearly separable data and can be regularized to limit overfitting, resulting in better modeling across multiple datasets.

Support Vector Machines:

SVMs are good classification models for high-dimensional datasets, like those derived from lexical URL features. Since feature extraction will produce many numerical features (like length, counts, entropy, etc.), SVM can efficiently find a separating boundary between malicious and benign features, even when they overlap or are not linearly separable.

The SVM model will be trained on the extracted feature vectors, experimenting with linear and non-linear kernels. Hyperparameters like the regularization parameter C and kernel coefficient γ will be adjusted with cross-validation to balance margins and tolerance for misclassification. SVM's decision boundary will be able to tell us the most prominent feature patterns separating malicious URLs from benign ones.

XGBoost:

XGBoost, or "Extreme Gradient Boosting" builds trees and learns from the errors of previous trees built. This in turn improves, or "boosts" the performance. In essence, it starts with a rough estimate and refines its guesses with each tree iteration.

XGBoost works great with tabular data like the one in this dataset, moreover, the data has a lot of complex micro-features that can be detected. XGBoost also regularizes the data and prevents overfitting

- **3.3 Evaluation Metrics**

Accuracy

Accuracy shows how often the machine learning model makes a correct prediction, in thighs case, how often it correctly identifies a URL as being faulty or valid. This is done by calculating the total number of correct predictions and dividing them by the total number of samples. The total correct predictions consist of True Positives and True Negatives, and the total sample is the same plus False Positives and False Negatives. This metric gives a quick sense of how well the model performs, but it can be misleading if one class (like benign URLs) greatly outnumbers the other. In that case, a model might appear accurate just by predicting the majority class more often.

Precision

Precision in classification means the ratio that measures the reliability of a positive classification. It answers the question: "When the model predicts 'faulty,' how often is it correct?" A high-precision model can be trusted on its positive predictions, with very few false alarms. This is crucial when the cost of acting on a false positive is high.

Recall

Recall measures how well the model finds all the positive cases. It answers the question: "Of all the actual faulty URLs, what proportion did the model successfully find?" A model with high recall misses very few true positives. This is important in cases where failing to detect a positive instance is going to have serious consequences.

F-1 Score

The F1 score is one metric, calculated as the combination of Precision and Recall. It is the mean of two factors; hence, it gives a balanced measure of a model's performance. The F1 score is most useful when you need to find a balance between minimizing false positives (high Precision) and minimizing false negatives (high Recall), especially when you must deal with imbalanced datasets.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

---

4. Task Division

Maksym will work on feature extraction, SVM model development, document formatting, and writing abstract and introduction for the final paper. Nghi will work on the model's handling of missing values, logistic regression model development, and will also help on accuracy and precision. Wasay will work on feature extraction, XGBoost model development, Recall and F1 Score evaluation metrics.