

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO THỰC TẬP CƠ SỞ**

**Đề tài: Phân loại URL độc hại**

**dựa trên mô hình học máy**

|  |
| --- |
| **Sinh viên thực hiện:** Phạm Văn Tiến  **Mã sinh viên:** B21DCCN708  **Giảng viên hướng dẫn:** Phạm Văn Cường |

**Hà Nội 2024**

**MỤC LỤC**

[I. Giới thiệu 3](#_Toc165537224)

[II. Đôi nét về URL 3](#_Toc165537225)

[III. Chuẩn bị dữ liệu 4](#_Toc165537226)

[IV. Xây dựng mô hình học máy 7](#_Toc165537227)

[1. Logistic Regression 8](#_Toc165537228)

[2. Decision Tree 9](#_Toc165537229)

[3. Random Forest 9](#_Toc165537230)

[4. Gradient Boosting 10](#_Toc165537231)

[5. XGBoost 10](#_Toc165537232)

[V. Kết quả và đánh giá 11](#_Toc165537233)

[VI. Kết luận 11](#_Toc165537234)

[VII. Tài liệu tham khảo 11](#_Toc165537235)

# Giới thiệu

Trong thời đại công nghệ số phát triển mạnh mẽ, internet đã trở thành một phần không thể thiếu trong cuộc sống của con người. Tuy nhiên, bên cạnh những tiện ích mà internet mang lại thì cũng có không ít những mối đe dọa về an toàn thông tin. Một trong những mối đe dọa phổ biến nhất là các trang web độc hại, có thể gây ra các thiệt hại như chứa các mã độc có thể tự động tải xuống, giả mạo đánh cắp thông tin cá nhân, lừa đảo tài chính,...

Vì vậy, việc nghiên cứu để phát hiện sớm có các kỹ thuật ngăn chặn URL độc hại trước khi người dùng truy vấn là biện pháp hiệu quả trong giải quyết vấn đề xâm nhập và phát tán các phần mềm độc hại thông qua môi trường web. Nhiệm vụ này được coi là một trong những hướng đi thu hút sự quan tâm của rất nhiều nhà nghiên cứu trong lĩnh vực bảo mật và an toàn thông tin.

Hiện nay hướng tìm hiểu, phân tích dựa vào các mô hình học máy để phân loại các URL độc hại là một phương pháp rất hiệu quả cho ngành công nghệ thông tin nói chung và an toàn thông tin nói riêng. Trong đề tài này, em sẽ sử dụng một số thuật toán phân loại được sử dụng trong học máy để đánh giá URL. Sau đó sử dụng mô hình đạt hiệu quả tốt nhất thiết kế một tiện ích mở rộng (extension) tích hợp trực tiếp vào trình duyệt web để giúp người dùng người dùng phòng tránh các trang web độc hại.

# Đôi nét về URL

Định vị tài nguyên thống nhất (Uniform Resource Locator – URL) được sử dụng để tham chiếu tới tài nguyên trên Internet, mang lại khả năng siêu liên kết cho các trang web. Các tài nguyên khác nhau được tham chiếu bằng các địa chỉ mạng (hay gọi cách khác là đường dẫn liên kết mạng) khác nhau.

Ảnh có chứa văn bản, ảnh chụp màn hình, hàng, Phông chữ

Mô tả được tạo tự độngCấu trúc cơ bản của một liên kết URL gồm 3 phần chính: giao thức (protocol), tên miền (host name) và đường dẫn (path).

Rất nhiều lập luận cho rằng URL là một địa chỉ web nhưng thực tế nó không hoàn toàn đơn giản như vậy. Một địa chỉ web là URL nhưng tất cả URL không phải chỉ là địa chỉ web. Một số dịch vụ có thể truy cập trên Internet như FTP hoặc thậm chí MAILTO cũng được sử dụng dưới dạng URL. Phần Protocol của URL biểu thị giao thức mà ứng dụng và máy chủ giao tiếp. Phần Host name của URL có nhiệm vụ ánh xạ tới một địa chỉ IP của một tài nguyên trên Internet. Host name bao gồm các thành phần như tên miền phụ (Subdomain name), tên miền (Domain name) và tên miền cao cấp (Top-level domain). Ngoài ra, trong phần Host name có thể chứa thông tin người dùng (ví dụ //username:password@www.example.com) hoặc chứa thông tin về cổng kết nối dịch vụ (ví dụ //www.example.com:8080).

Nếu như phần Host name của URL đưa trình duyệt (hoặc các ứng dụng khác) đến đúng máy chủ trên mạng, thì phần Path sẽ giúp truy vấn đến đúng thư mục hoặc tệp tin trên máy chủ đó.

# Chuẩn bị dữ liệu

Tập dữ liệu (dataset) thu thập từ các kho dữ liệu lớn hay được các hãng bảo mật nổi tiếng trên thế giới cung cấp như Kaggle, UCI Machine Learning Repository, OpenPhish, PhishTank được tổng hợp lại và dán nhãn 1 cho các URL độc hại và 0 cho URL an toàn. Sau khi xử lý để loại bỏ các giá trị trùng lặp và giảm sự mất cân bằng giữa 2 lớp, ta thu được một danh sách 288158 URL gồm 144079 URL an toàn và 144079 URL độc hại.

Ảnh có chứa văn bản, ảnh chụp màn hình, Sơ đồ, biểu đồ

Mô tả được tạo tự động

**Trích chọn đặc trưng (Feature Extraction)**

Những tính chất sau được lựa chọn là đặc trưng tiêu biểu để phân định URL độc hại hay an toàn, trong báo cáo này mới chỉ chọn các đặc trưng từ vựng của URL:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Đặc trưng | Kiểu dữ liệu | Mô tả |
| 1 | IpAddress |  |  |
| 2 | UrlLength |  |  |
| 3 | HostnameLength |  |  |
| 4 | PathLength |  |  |
| 5 | QueryLength |  |  |
| 6 | UseShortService |  |  |
| 7 | SusTlds |  |  |
| 8 | NumSensitiveWords |  |  |
| 9 | NumNumericChars |  |  |
| 10 | NumDots |  |  |
| 11 | NumDash |  |  |
| 12 | NumDashInHostname |  |  |
| 13 | NumUnderscore |  |  |
| 14 | NumPercent |  |  |
| 15 | NumAmpersand |  |  |
| 16 | NumHash |  |  |
| 17 | NumQueryComponents |  |  |
| 18 | AtSymbol |  |  |
| 19 | TildeSymbol |  |  |
| 20 | DoubleSlashInPath |  |  |
| 21 | SubDomainLevel |  |  |
| 22 | PathLevel |  |  |
| 23 | DomainInSubdomains |  |  |
| 24 | DomainInPaths |  |  |
| 25 | EntropyDomainName |  |  |
| 26 | rank\_host |  |  |

* haveIP(): Phương thức này trả về 1 nếu URL có sử dụng địa chỉ IP (kiểm tra cả IPv4 hoặc IPv6), không có trả về 0. Nếu địa chỉ IP được sử dụng (VD như “http://137.127.5.147/fake.html”) thay thế cho tên miền trong URL, người dùng có thể chắc chắn rằng ai đó đang cố lấy cắp thông tin cá nhân quan trọng của họ. Đôi khi, địa chỉ IP còn được chuyển đổi thành dạng mã thập lục phân.
* lenURL(): Phương thức này trả về độ dài URL. Các URL độc hại thường dài hơn so với các URL bình thường vì chúng thường chứa nhiều thông tin hơn để che giấu hoạt động độc hại. Ví dụ, chúng có thể chứa các script độc hại hoặc các tham số dẫn đến các trang web độc hại khác.Trong tập dữ liệu này, độ dài trung bình của 1 URL độc hại là 63.36 ký tự và an toàn là 49.74 ký tự.
* lenHostname(): Phương thức này trả về độ dài của tên máy chủ trong URL. Tên máy chủ dài có thể là dấu hiệu của một URL độc hại vì nó có thể được sử dụng để mạo danh một trang web hợp lệ. Ví dụ, một tên máy chủ dài có thể chứa tên của một trang web hợp lệ nhưng thêm vào đó là các từ khóa hoặc tham số độc hại.
* tinyURL(): Phương thức này trả về 1 nếu URL này sử dụng dịch vụ rút gọn URL, ngược lại trả về 0. Rút gọn URL là một phương pháp trên “World Wide Web”, trong đó URL có thể được làm cho có độ dài nhỏ hơn đáng kể mà vẫn dẫn đến trang web được yêu cầu. Điều này được thực hiện bằng cách “Chuyển hướng HTTP” trên một tên miền ngắn, liên kết đến trang web có URL dài. Ví dụ: URL “http://portal.hud.ac.uk/” có thể được rút ngắn thành “bit.ly/19DXSk4”. Với các công cụ phổ biến hỗ trợ rút ngắn độ dài URL, kẻ tấn công có thể che giấu những đặc trưng dễ nhận biết trên URL đối với người dùng và có thể liên kết đó là độc hại.
* httpDomain(): Phương thức này trả về 1 nếu phát hiện có “http” hoặc “https” trong phần tên miền, ngược lại trả về 0. Kẻ tấn công có thể thêm “https” hoặc “http” vào phần tên miền của URL để lừa người dùng.
* abnormal\_url(): Phương thức này trả về 1 nếu không tồn tại hostname trong URL, ngược lại trả về 0.
* suspicious\_tlds(): Phương thức này trả về 1 nếu tên miền cấp cao nhất (TLD) của URL có nằm trong danh sách các TLD đáng ngờ, ngược lại trả về 0.
* digit\_count(): Phương thức này trả về số lượng chữ số trong URL. Các URL độc hại thường chứa nhiều chữ số hơn so với các URL bình thường.
* letter\_count(): Phương thức này trả về số lượng chữ cái trong URL. Các URL độc hại thường chứa nhiều chữ cái hơn so với các URL bình thường.
* special\_chars\_count(): Phương thức này trả về số lượng ký tự đặc biệt trong URL. Các URL độc hại thường chứa nhiều ký tự đặc biệt hơn so với các URL bình thường.
* haveAtSign(): Phương thức này trả về 1 nếu URL có chứa ‘@’, không có trả về 0. Việc sử dụng ký hiệu ‘@’ trong URL sẽ khiến trình duyệt bỏ qua mọi thứ trước ký hiệu này và địa chỉ thực thường theo sau ký hiệu ‘@’. Các URL độc hại thường chứa ký tự '@' để mạo danh một URL an toàn.
* haveDash(): Phương thức này trả về 1 nếu ký tự ‘-‘ xuất hiện trong tên miền của URL, ngược lại trả về 0. Đối với các URL an toàn việc xuất hiện ký tự ‘-‘ tương đối ít, những kẻ đáng ngờ có xu hướng thêm tiền tố hoặc hậu tố được phân tách bằng ‘-‘ vào tên miền để người dùng cảm thấy rằng họ đang truy cập một trang web hợp pháp. Ví dụ: “http://www.Confirm–paypal.com/”.
* redirection(): Phương thức này trả về 1 nếu “//” nằm ở bất kỳ đâu trong URL ngoại trừ sau giao thức, ngược lại trả về 0. Sự tồn tại của “//” trong liên kết URL có nghĩa là người dùng sẽ được chuyển hướng đến một trang web khác (VD như “http://www.legitimate.com//http://www.phishing.com”). Trong 1 URL chuẩn, “//” xuất hiện ngay sau giao thức của URL là “http:” hoặc “https:”. Vì vậy, nếu URL bắt đầu bằng “http” thì “//” sẽ xuất hiện ở vị trí thứ 6 (tính từ 0) và bắt đầu bằng “https” thì sẽ xuất hiện ở vị trí thứ 7. Nếu “//” xuất hiện ở 1 vị trí khác thì có thể nghi ngờ rằng URL này là độc hại vì nó đang chuyển hướng đến 1 trang web khác.
* Ảnh có chứa văn bản, ảnh chụp màn hình

  Mô tả được tạo tự độngsubDomains(): Phương thức này trả về 1 nếu URL chứa nhiều hơn 1 tên miền phụ, ngược lại trả về 0. Giả sử ta có liên kết sau: “http://www.hud.ac.uk/students/”. Một tên miền có thể bao gồm các tên miền cấp cao nhất theo mã quốc gia (ccTLD), trong ví dụ là “uk”. Phần “ac” là viết tắt của “academic”, phần “ac.uk” được gọi là tên miền cấp hai (SLD) và “hud” là tên thật của miền. Để tạo quy tắc trích xuất đặc trưng này, trước tiên phải bỏ qua (www.) khỏi URL, trên thực tế, bản thân nó là một tên miền phụ. Sau đó phải xóa (ccTLD) nếu nó tồn tại. Cuối cùng, đếm các dấu chấm còn lại.

Mối tương quan giữa các đặc trưng trong bộ dữ liệu này

# Xây dựng mô hình học máy

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự độngTrong quá trình áp dụng mô hình học máy, báo cáo này áp dụng tỷ lệ 8 : 2 nhằm phân chia tập dữ liệu huấn luyện và tập kiểm tra. Trong đó, X\_train và X\_test là tập hợp mẫu dữ liệu lần lượt có trong tập huấn luyện và tập kiểm tra. Kế tiếp, y\_train và y\_test là các nhãn dán tương ứng cho các mẫu dữ liệu trong 2 tập trên, trong đó y\_train sẽ được sử dụng để huấn luyện mô hình, và sau đó y\_test sẽ được dùng để so sánh kết quả phân loại, từ đó đánh giá hiệu suất phân loại của mô hình.

## Logistic Regression

Logistic Regression là một thuật toán thường được sử dụng trong phân loại nhị phân. Dù tên gọi có chứa từ "Regression", nhưng thực tế nó được sử dụng cho các bài toán phân loại. Logistic Regression là một thuật toán phân loại trong Machine Learning dựa trên nguyên tắc của Hồi quy Tuyến tính (Linear Regression). Tuy nhiên, thay vì dự đoán giá trị liên tục như Hồi quy Tuyến tính, Logistic Regression dự đoán xác suất của một sự kiện nhất định. Điều này làm cho nó rất hữu ích cho các bài toán phân loại nhị phân, nơi mục tiêu là dự đoán một trong hai lớp có thể có.

Logistic Regression hoạt động bằng cách sử dụng hàm logistic (hoặc hàm sigmoid) để biến đổi đầu ra của một mô hình hồi quy tuyến tính thành một giá trị xác suất từ 0 đến 1.

Cụ thể, mô hình hồi quy tuyến tính là hàm tuyến tính các đặc trưng đầu vào, với công thức:

Để giới hạn đầu ra trong khoảng [0, 1], hồi quy logistic sử dụng mô hình sau:

Với là hàm logistic hay hàm sigmod.

Mục tiêu của việc huấn luyện mô hình Logistic Regression là tìm các tham số mô hình sao cho hàm mất mát (Loss function) trên tập dữ liệu huấn luyện là nhỏ nhất. Hàm mất mát là một hàm số được sử dụng để đo lường mức độ lỗi mà mô hình của chúng ta tạo ra khi dự đoán các kết quả từ dữ liệu đầu vào. Trong bài toán Hồi quy Logistic, chúng ta sử dụng hàm mất mát Cross-Entropy (còn gọi là Log Loss) để đánh giá hiệu năng của mô hình.

Hàm Cross-Entropy đo lường khoảng cách giữa hai phân phối xác suất của giá trị thực tế ứng với đầu vào thứ i (yi) và xác suất dự đoán thuộc lớp 1 của mô hình cho đầu vào thứ I (pi). Khi mô hình dự đoán chính xác, tức là nếu yi = 1 thì pi càng gần 1 và nếu yi = 0 thì pi càng gần 0, sau đó hàm mất mát sẽ tiến gần về 0.

Trong quá trình huấn luyện, chúng ta tìm cách cập nhật bộ tham số sao cho giá trị hàm mất mát Cross-Entropy đạt giá trị nhỏ nhất, dẫn đến một mô hình dự đoán tốt nhất. Để tìm giá trị tối ưu cho bộ tham số, chúng ta có thể sử dụng kỹ thuật Gradient Descent.

Trong báo cáo này sử dụng lớp LogisticRegression được xây dựng sẵn trong thư viện sklearn.linear\_model với max\_iter=1000.

## Decision Tree

Thuật toán Decision Tree, hay Cây quyết định, là một phương pháp học máy phổ biến được sử dụng trong cả bài toán phân loại và hồi quy. Cấu trúc của Decision Tree gồm có nút gốc, nút quyết định, và nút lá. Mỗi nút quyết định trong cây tương ứng với một thuộc tính hay đặc trưng của dữ liệu, và mỗi nhánh đi xuống từ nút đó tương ứng với một giá trị hoặc một khoảng giá trị của thuộc tính đó. Nút lá của cây chứa nhãn hoặc giá trị dự đoán cho dữ liệu.

Quá trình học của Decision Tree bao gồm việc chọn thuộc tính để chia dữ liệu tại mỗi nút. Các thuật toán như ID3, C4.5, và CART được sử dụng để quyết định thuộc tính nào sẽ được chọn dựa trên các tiêu chí như Entropy, Information Gain, Gini Index, và Reduction in Variance. Mỗi tiêu chí này đều cung cấp một cách để đo lường chất lượng của một phép chia, giúp thuật toán quyết định cách chia dữ liệu tại mỗi nút.

Trong báo cáo này sử dụng lớp DecisionTreeClassifier được xây dựng sẵn trong thư viện sklearn.tree với criterion (hàm đo lường chất lượng phép chia) sử dụng entropy.

## Random Forest

Random Forest là một trong những phương pháp học máy tập hợp (ensemble learning) phổ biến nhất, thuộc nhóm phương pháp đóng gói (Bagging method).

Phương pháp tập hợp liên quan đến việc sử dụng nhiều thuật toán học (leaners) để nâng cao hiệu suất của bất kỳ mô hình nào trong số đó một cách riêng lẻ. Các phương pháp này có thể được mô tả như là kỹ thuật sử dụng một nhóm những mô hình học yếu với nhau (những mô hình học mà trong đó, chúng chỉ đạt được kết quả phân loại tốt hơn giá trị trung bình một chút so với việc sử dụng một mô hình học ngẫu nhiên), để tạo ra một mô hình tổng hợp, mạnh mẽ hơn. Trong trường hợp này, mô hình Rừng ngẫu nhiên là sự kết hợp của rất nhiều mô hình học Cây quyết định (Decision Trees) để áp dụng vào các bài toán phân loại hoặc hồi quy.

Random Forest hoạt động bằng cách tạo ra một tập hợp các cây quyết định trong quá trình huấn luyện, sau đó đưa ra dự đoán dựa trên kết quả đầu ra của mỗi cây quyết định. Các kết quả đầu ra này sẽ được xếp hạng, và kết quả cao nhất sẽ được chọn làm kết quả phân loại đầu ra cuối cùng. Việc lựa chọn kết quả dự đoán cuối cùng tuân theo nguyên tắc đa số. Do đó, kết quả phân loại được lựa chọn bởi số lượng lớn các cây quyết định sẽ trở thành kết quả đầu ra cuối cùng của mô hình rừng ngẫu nhiên. Mỗi cây quyết định trong "rừng" được tạo ra từ một tập con ngẫu nhiên của dữ liệu huấn luyện, và chỉ sử dụng một tập con ngẫu nhiên của các đặc trưng để tạo ra các điểm phân chia. Điều này giúp giảm thiểu hiện tượng quá khớp (overfitting), một vấn đề thường gặp khi sử dụng cây quyết định đơn lẻ.

Trong báo cáo này sử dụng lớp RandomForestClassifier được xây dựng sẵn trong thư viện sklearn.ensemble với criterion (hàm đo lường chất lượng phép chia) sử dụng entropy.

## Gradient Boosting

Gradient Boosting là một thuật toán học máy mạnh mẽ dựa trên kỹ thuật boosting. Boosting là một phương pháp kết hợp nhiều mô hình yếu để tạo ra một mô hình mạnh. Khác với phương pháp đóng gói (bagging), boosting xây dựng một lượng lớn các mô hình. Mỗi mô hình sau sẽ học cách giảm thiểu lỗi của mô hình trước, tạo thành một chuỗi các mô hình mà mô hình sau sẽ tốt hơn mô hình trước bởi việc chú ý tới và cố gắng khắc phục những ví dụ bị phân loại sai hơn (cụ thể ở đây là trọng số của những dữ liệu dự đoán đúng sẽ giảm trọng số, còn trọng số của những dữ liệu dự đoán sai sẽ được tăng thêm). Chúng ta sẽ lấy kết quả của mô hình cuối cùng trong chuỗi mô hình này làm kết quả trả về.

Mô hình Gradient Boosting tiến hành việc đánh lại trọng số cho các điểm dữ liệu hiện tại, nhằm giúp các mô hình mới có thể tập trung hơn vào các mẫu dữ liệu đang bị học sai, từ đó làm giảm giá trị của hàm mất mát, quá trình này sử dụng phương pháp tối ưu hóa Gradient Descent.

## XGBoost

XGBoost (eXtreme Gradient Boosting) là phiên bản cải tiến của Gradient Boosting. Ưu điểm của nó được chứng minh trên nhiều khía cạnh:

* XGBoost thực hiện tinh toán song song nên tốc độ xử lý có thể tăng gấp 10 lần so với GBM. Ngoài ra, XGboost còn hỗ trợ tính toán trên Hadoop.
* XGBoost áp dụng cơ chế Regularization nên hạn chế đáng kể hiện tượng Overfitting
* XGboost cho phép người dùng sử dụng hàm tối ưu và chỉ tiêu đánh giá của riêng họ, không hạn chế ở những hàm cung cấp sẵn.
* XGBoost bao gồm cơ chế tự động xử lý missing value bên trong nó. Vì thế, có thể bỏ qua bước này khi chuẩn bị dữ liệu cho XGBoost.
* Tự động cắt tỉa cây (auto pruning): tự động bỏ qua những leaves, nodes không mang giá trị tích cực trong quá trình mở rộng tree.

# Kết quả và đánh giá

## Các chỉ số đánh giá

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, số

Mô tả được tạo tự độngĐể đánh giá các mô hình được đề xuất, sử dụng ma trận nhầm lẫn và các chỉ số đánh giá phổ biến như Accuracy, Precision, Recall và F1 score:

Trong bài toán này thì nhãn URL độc hại là 1 và URL an toàn là 0:

* TP (True Positive): Số lần URL độc hại được dự đoán đúng.
* TN (True Negative): Số lần URL an toàn được dự đoán đúng.
* FP (False Positive): Số lần URL an toàn bị dự đoán sai thành URL độc hại.
* FN (False Negative): Số lần URL độc hại bị dự đoán sai thành URL an toàn.

Dựa trên các thông số trên, những chỉ số đánh giá sau có thể được tính:

* Độ chính xác (Accuracy): Tỷ lệ số lần dự đoán chính xác trong tất cả các mẫu thử nghiệm: Accuracy = (TP + TN) / (TP + TN + FP + FN) = (TP + TN) / N
* Precision: Tỷ lệ số lần dự đoán đúng URL độc hại với tất cả URL được dự đoán là độc hại bởi mô hình: Precision = TP / (TP + FP)
* Recall: Tỷ lệ số lần dự đoán đúng URL độc hại với tất cả URL độc hại trong mẫu thử nghiệm: Recall = TP / (TP + FN)
* F1 score: Tổng hợp 2 chỉ số Precision và Recall, một mô hình có chỉ số F-score cao chỉ khi cả 2 chỉ số Precision và Recall đều cao. Sử dụng F1-score cũng là một thước đo đáng tin cậy về hiệu năng của mô hình trong các bài toán phân loại, đặc biệt khi dữ liệu về một lớp lớn hơn gấp nhiều lần so với dữ liệu về lớp còn lại: F1 score = (2 \* Precision \* Recall) / (Precision + Recall)

## Kết quả

# Kết luận

Trong báo cáo này, em đã trình bày một phương pháp phát hiện URL độc hại đó là sử dụng machine learning. Kết quả được rút ra từ Bảng V và VI đã cho thấy tính hiệu quả của các đặc trưng được trích xuất. Kết quả của báo cáo này đã được sử dụng để xây dựng một tiện ích đơn giản trên trình duyệt nhằm phát hiện các URL độc hại.

Tuy nhiên, em cũng nhận ra rằng việc chỉ sử dụng các đặc trưng kí tự từ URL có thể không đủ để đối mặt với tất cả các tình huống trong thực tế. Có rất nhiều các đặc trưng khác có thể được rút ra từ URL hoặc dữ liệu trang web, bao gồm như các đặc trưng được trích rút từ code HTML và JavaScript của trang web, hay tuổi tên miền, số lượng truy cập và nhiều hơn nữa. Những đặc trưng này có thể cung cấp thêm nhiều thông tin quan trọng giúp cải thiện tính đúng đắn của mô hình học máy. Vì việc trích xuất các đặc trưng đó mất rất nhiều thời gian, nên trong báo cáo này em vẫn chưa thực hiện được, đó là một thiếu xót lớn.

Trong tương lai, em dự định sẽ mở rộng mô hình của mình để bao gồm thêm các đặc trưng này và sử dụng các tập dữ liệu lớn hơn. Em tin rằng việc này sẽ giúp mô hình của mình trở nên mạnh mẽ hơn và có khả năng đối mặt với nhiều tình huống phức tạp hơn trong thực tế.

# Tài liệu tham khảo

1. Phạm Thị Lan Anh (2023), *Tổng Quan về công Nghệ Blockchain và ứng dụng: Khoa Công nghệ thông tin VNUA*. Available at: <https://fita.vnua.edu.vn/tong-quan-ve-cong-nghe-blockchain-va-ung-dung/>
2. [Xây dựng cơ sở dữ liệu huấn luyện phục vụ phát hiện URL độc hại (mic.gov.vn)](https://frt.mic.gov.vn/mic_2020/Pages/TinTuc/144326/Xay-dung-co-so-du-lieu-huan-luyen-phuc-vu-phat-hien-URL-doc-hai.html)
3. [PhiUSIIL Phishing URL (Website) - UCI Machine Learning Repository](https://archive.ics.uci.edu/dataset/967/phiusiil+phishing+url+dataset)
4. [Malicious And Benign URLs (kaggle.com)](https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls?resource=download)
5. [OpenPhish - Phishing Intelligence](https://openphish.com/)
6. [PhishStorm - phishing / legitimate URL dataset — Aalto University's research portal](https://research.aalto.fi/en/datasets/phishstorm-phishing-legitimate-url-dataset)
7. [shreyagopal/Phishing-Website-Detection-by-Machine-Learning-Techniques (github.com)](https://github.com/shreyagopal/Phishing-Website-Detection-by-Machine-Learning-Techniques/tree/master)
8. [[PDF] An assessment of features related to phishing websites using an automated technique | Semantic Scholar](https://www.semanticscholar.org/paper/An-assessment-of-features-related-to-phishing-using-Mohammad-Thabtah/0c0ff58063f4e078714ea74f112bc709ba9fed06)
9. [Detection of malicious URLs using machine learning | Wireless Networks (springer.com)](https://link.springer.com/article/10.1007/s11276-024-03700-w#Fig1)
10. [Bài 6: Logistic Regression (Hồi quy Logistic) - Trí tuệ nhân tạo (trituenhantao.io)](https://trituenhantao.io/machine-learning-co-ban/bai-6-logistic-regression-hoi-quy-logistic/)
11. [Gradient Boosting - Tất tần tật về thuật toán mạnh mẽ nhất trong Machine Learning (viblo.asia)](https://viblo.asia/p/gradient-boosting-tat-tan-tat-ve-thuat-toan-manh-me-nhat-trong-machine-learning-YWOZrN7vZQ0)
12. [(PDF) Malicious URL Detection based on Machine Learning (researchgate.net)](https://www.researchgate.net/publication/339023050_Malicious_URL_Detection_based_on_Machine_Learning)
13. [Entropy | Free Full-Text | Malicious URL Detection Based on Associative Classification (mdpi.com)](https://www.mdpi.com/1099-4300/23/2/182)