Phát triển hệ thống AI để phát hiện và ngăn chặn URL độc hại

Dương Hớn Minh

Khoa Dược, Trường Đại học Nguyễn Tất Thành, Thành phố Hồ Chí Minh, Việt Nam
dhminh@ntt.edu.vn

Tóm tắt

URL độc hại là một trong những hình thức tấn công mạng phổ biến nhất, gây ra nhiều hậu quả nghiêm trọng về tài chính và an toàn thông tin. Các phương pháp truyền thống như blacklist và heuristic gặp nhiều hạn chế về khả năng mở rộng và không thể phát hiện các mối đe dọa mới. Nghiên cứu này đề xuất pipeline phát hiện URL độc hại dựa trên học máy, kết hợp tiền xử lý dữ liệu, trích xuất 9 đặc trưng bổ sung từ chuỗi URL và kỹ thuật SMOTE để xử lý mất cân bằng lớp. Ba mô hình Random Forest, XGBoost và CatBoost được huấn luyện và đánh giá trên tập dữ liệu 181 916 mẫu. Kết quả cho thấy cả ba mô hình đều đạt Accuracy trên 98,6 % và AUC-ROC trên 99,1 %, vượt trội so với các phương pháp cơ sở. Random Forest đạt độ ổn định cao nhất, XGBoost cân bằng tốt giữa hiệu suất và tốc độ huấn luyện. SMOTE giúp duy trì Recall cao mà không làm suy giảm đáng kể các chỉ số khác.

Nhận 06/04/2026
Được duyệt 02/06/2026
Công bố 28/07/2026

Từ khóa

URL độc hại; học máy; Random Forest; XGBoost; CatBoost; SMOTE.

1 Giới thiệu

Uniform Resource Locator (URL) là chuỗi ký tự dùng để định danh và định vị tài nguyên trên internet. Do xuất hiện trong hầu hết các hoạt động trực tuyến như email, SMS và mạng xã hội, URL trở thành mục tiêu phổ biến của các cuộc tấn công phishing nhằm đánh cắp thông tin cá nhân hoặc phát tán mã độc, gây thiệt hại về tài chính, dữ liệu và uy tín tổ chức [1-3].

Các hệ thống phát hiện truyền thống chủ yếu dựa trên phương pháp Rule-based, tiêu biểu là blacklist, hoạt động bằng cách đối chiếu URL với danh sách các địa chỉ độc hại đã biết. Tuy nhiên, hiệu quả của phương pháp này phụ thuộc vào tốc độ cập nhật dữ liệu và không thể phát hiện các URL phishing mới (zero-day phishing) [4, 5]. Để khắc phục hạn chế này, các phương pháp heuristic khai thác các đặc trưng của URL như độ dài, tên miền và ký tự đặc biệt nhằm mở rộng khả năng phát hiện. Mặc dù vậy, heuristic vẫn dựa trên các quy tắc được xây dựng thủ công, khó thích nghi trước sự thay đổi liên tục của các kỹ thuật tấn công [6].

Sự phát triển của học máy cho phép mô hình tự động học quy luật từ dữ liệu thay vì phụ thuộc vào các quy tắc cố định. Các nghiên cứu đã kết hợp đặc trưng URL với nội dung HTML [7], xây dựng tập dữ liệu lớn kết hợp học tăng dần và chỉ số tương đồng [8], đồng thời áp dụng SMOTE để xử lý mất cân bằng dữ liệu và cải thiện hiệu quả phân loại [9]. Song song đó, các mô hình học sâu như CNN, LSTM và Transformer tiếp tục nâng cao khả năng học biểu diễn trực tiếp từ chuỗi URL, trong đó TransURL và urlBERT đạt hiệu suất AUC trên 99 % trên nhiều bộ dữ liệu chuẩn [10-12]. Tuy nhiên, các nghiên cứu tổng quan cho thấy những mô hình này vẫn gặp thách thức về chi phí huấn luyện, điều chỉnh tham số và tính ổn định khi triển khai thực tế [13].

https://doi.org/10.55401/mtt0vw30

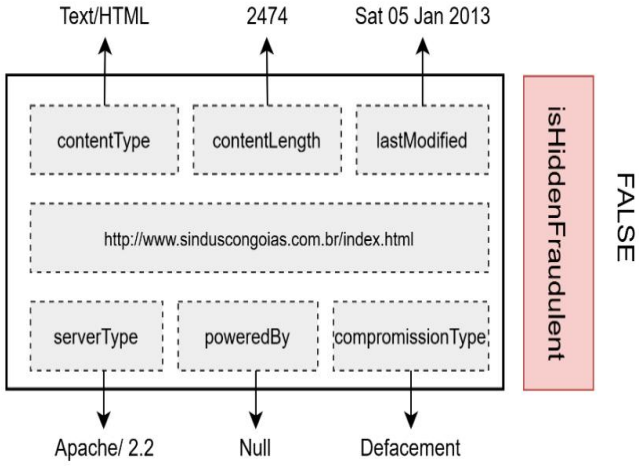
Bên cạnh đó, tình trạng mất cân bằng dữ liệu vẫn là vấn đề phổ biến trong bài toán phát hiện phishing do số lượng URL hợp lệ thường lớn hơn đáng kể so với URL độc hại. Kỹ thuật SMOTE được chứng minh có khả năng cải thiện hiệu quả của các mô hình học máy bằng cách tạo thêm các mẫu tổng hợp cho lớp thiểu số mà không yêu cầu giả định phức tạp về phân phối dữ liệu [14]. Đồng thời, nhiều nghiên cứu cũng chỉ ra rằng tiền xử lý và trích xuất đặc trưng vẫn chưa được khai thác đầy đủ, mặc dù đây là những yếu tố ảnh hưởng trực tiếp đến chất lượng phân loại [15].

Xuất phát từ những khoảng trống trên, nghiên cứu này đề xuất quy trình phát hiện URL phishing gồm làm sạch dữ liệu, trích xuất chín đặc trưng URL, mã hóa biến chuỗi và cân bằng dữ liệu bằng SMOTE trước khi huấn luyện các mô hình Random Forest, XGBoost và CatBoost. Hiệu quả của từng mô hình được đánh giá thông qua Accuracy, Precision, Recall, F1-score và AUC-ROC, đồng thời phân tích tác động của SMOTE đối với từng họ thuật toán nhằm xác định mô hình phù hợp cho bài toán phát hiện URL phishing.

2 Phương pháp nghiên cứu

2.1 Dữ liệu

2.1.1 Tổng quan



Hình 1 Một mẫu dữ liệu bao gồm 7 đặc trưng và nhãn phân loại

Dữ liệu được sử dụng trong nghiên cứu lấy từ: https://machinelearning.inginf.units.it/data-and-tools/hidden-fraudulent-urls-dataset. Tập dữ liệu bao gồm 185 180 URL với 6 đặc trưng được trích xuất kèm 1 URL và 1 nhãn phân loại. Tỷ lệ URL không độc hại chiếm 95,3 %, gây mất cân bằng nghiêm trọng. Hình 1 mô tả một ví dụ mẫu trong bộ dữ liệu đang xem xét.

Mặc dù pipeline đạt hiệu suất cao trên tập dữ liệu Hidden Fraudulent URLs, một thách thức thực tế là hiện tượng trôi dạt dữ liệu (Data Drift), khi các kỹ thuật tấn công phishing liên tục thay đổi cấu trúc URL theo thời gian. Để nâng cao tính thực tiễn, nghiên cứu đề xuất cơ chế giám sát liên tục: theo dõi sự sụt giảm đột ngột của độ tin cậy dự đoán (prediction confidence score) hoặc sự trôi dạt đặc trưng (Feature Drift Monitoring).

Cơ chế giám sát này có thể triển khai dựa trên hai lớp kiểm soát sau:

1. Giám sát trôi dạt đặc trưng (Feature Drift Monitoring): hệ thống sẽ theo dõi phân phối thống kê của các đặc trưng đầu vào theo cửa sổ thời gian trượt (sliding window). Bằng cách áp dụng các kiểm định thống kê như chỉ số ổn định quần thể (Population Stability Index – PSI) hoặc kiểm định Kolmogorov-Smirnov, hệ thống có thể phát hiện sự thay đổi đột ngột trong phân phối dữ liệu so với ban đầu được thiết lập trong giai đoạn huấn luyện.

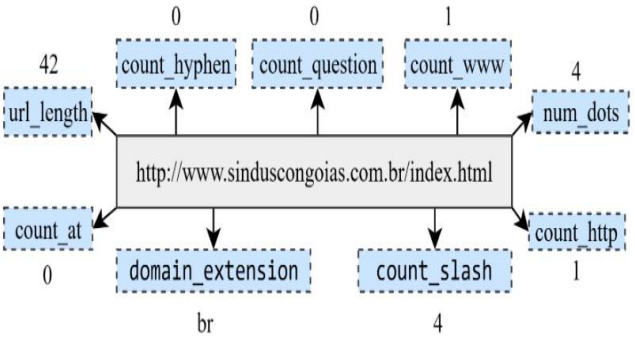
2. Giám sát độ tin cậy dự đoán (Prediction Confidence Monitoring): bên cạnh việc theo dõi đặc trưng, hệ thống sẽ giám sát điểm tin cậy trung bình (average confidence score) của các dự đoán. Nếu tỷ lệ các mẫu URL được phân loại với độ tin cậy thấp (ví dụ: probability score < 0,70) vượt quá một ngưỡng cảnh báo θ định trước, cho thấy mô hình đang gặp khó khăn trong việc phân loại các mẫu mới.

Khi này để đảm bảo hệ thống hoạt động tốt trở lại, khuyến cáo đưa ra sẽ gồm hai bước: (i) chuyển các mẫu URL có độ tin cậy thấp sang hàng đợi ưu tiên để chuyên gia bảo mật (SOC Analyst) xem xét và gán nhãn thủ công; (ii) kích hoạt quy trình huấn luyện lại (Retraining Pipeline) định kỳ, bổ sung các mẫu mới đã được gán nhãn vào tập dữ liệu để cập nhật trọng số mô hình, đảm bảo hệ thống luôn thích nghi với các mối đe dọa mới.

2.1.2 Tiền xử lý

Trong giai đoạn tiền xử lý dữ liệu, nghiên cứu tiến hành loại bỏ các thuộc tính không cần thiết nhằm tối ưu chất lượng dữ liệu đầu vào. Cụ thể, cột *lastModified*

được loại bỏ do không đóng góp ý nghĩa đáng kể cho mục tiêu phân loại URL độc hại. Đối với dữ liệu thiếu, các giá trị NaN trong cột *poweredBy* được thay thế bằng giá trị 0, trong khi các dòng chứa dữ liệu thiếu ở hai cột *serverType* và *contentType* bị loại bỏ khỏi tập dữ liệu. Sau quá trình xử lý, tập dữ liệu còn lại 181 916 mẫu. Ngoài ra, nghiên cứu cũng thực hiện trích xuất bổ sung 9 đặc trưng mới từ chuỗi URL nhằm tăng khả năng biểu diễn thông tin và hỗ trợ nâng cao hiệu quả phân loại của mô hình học máy, được trực quan trong Hình 2.



Hình 2 Các đặc trưng bổ sung trích xuất từ URL

Bảng 1 Tổng hợp các đặc trưng

Tên đặc trưng	Mô tả	Ý nghĩa bảo mật	Loại đặc trưng
url	URL gốc	Thông tin đầu vào ban đầu	Gốc
compromissionType	Loại tấn công	Phân loại kiểu xâm nhập	Gốc
isHiddenFraudulent	Nhãn mục tiêu	Xác định URL độc hại	Gốc
contentLength	Độ dài nội dung HTTP	Phát hiện bất thường kích thước	Gốc
serverType	Loại máy chủ	Nhận diện môi trường hosting	Gốc
poweredBy	Công nghệ server	Xác định nền tảng backend	Gốc
contentType	Kiểu nội dung	Phân biệt loại tài nguyên	Gốc
url_length	Độ dài URL	URL giả mạo thường dài	Bổ sung
num_dots	Số dấu “.”	Nhiều subdomain gây nhầm lẫn	Bổ sung
count_http	Số lần “http/https”	Tạo cảm giác hợp pháp	Bổ sung
count_www	Số lần “www”	Đánh lừa người dùng	Bổ sung
count_slash	Số dấu “/”	Cấu trúc URL bất thường	Bổ sung
count_hyphen	Số dấu “-”	Giả mạo tên miền	Bổ sung
count_at	Số ký tự “@”	Đánh lạc hướng trình duyệt	Bổ sung
count_question	Số ký tự “?”	Lạm dụng tham số URL	Bổ sung
domain_extension	Phần mở rộng tên miền	Một số TLD thường dùng cho URL độc hại	Bổ sung

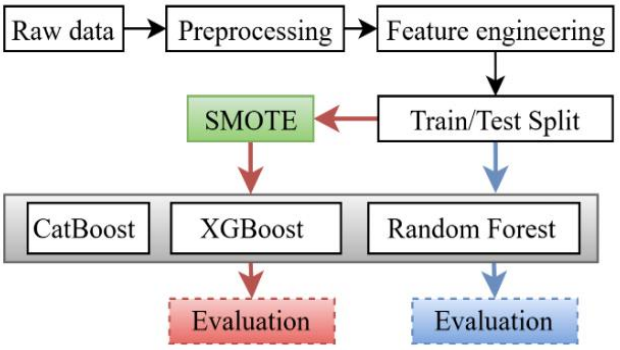
Đối với bước mã hóa dữ liệu, thuộc tính *domain_extension* được mã hóa bằng phương pháp frequency encoding. Các biến phân loại gồm *compromissionType*, *serverType*, *poweredBy* và *contentType* được mã hóa bằng kỹ thuật one-hot encoding nhằm chuyển đổi dữ liệu về dạng số để phù hợp với mô hình học máy. Sau quá trình mã hóa, tổng số đặc trưng của tập dữ liệu tăng lên 231 cột. Cuối cùng, tập dữ liệu được chia thành tập huấn luyện và tập kiểm tra với tỷ lệ lần lượt là 80 % và 20 %.

Bảng 1 thu được như sau: quy trình tiền xử lý và trích xuất đặc trưng, tập dữ liệu đã được làm sạch, chuẩn hóa và mở rộng với các đặc trưng có ý nghĩa về mặt bảo mật, giúp tăng khả năng phân biệt giữa URL hợp lệ và URL độc hại. Tuy nhiên, vấn đề mất cân bằng dữ liệu vẫn là một thách thức lớn cần được xử lý nhằm đảm bảo hiệu quả học của mô hình.

2.2 Phương pháp đề xuất

Nghiên cứu đề xuất một quy trình phân loại URL độc hại gồm hai giai đoạn chính: xử lý dữ liệu và huấn luyện mô hình, được minh họa trong Hình 3.

Giai đoạn 1 – Nghiên cứu tập trung làm sạch, loại bỏ các cột không cần thiết và xử lý giá trị thiếu. Tiếp theo, 9 đặc trưng mới được trích xuất trực tiếp từ chuỗi URL nhằm tăng khả năng biểu diễn thông tin bảo mật.



Hình 3 Tổng quan phương pháp nghiên cứu

Các biến phân loại được mã hóa bằng one-hot encoding và frequency encoding. Tập dữ liệu sau đó được chia theo tỷ lệ 80/20 cho huấn luyện và kiểm tra. Giai đoạn 2 – Nghiên cứu tiến hành xử lý mất cân bằng và huấn luyện mô hình. Do tỷ lệ URL không độc hại chiếm tới 95,3 %, kỹ thuật SMOTE được áp dụng trên tập huấn luyện để tạo các mẫu tổng hợp cho lớp thiểu số, tránh hiện tượng mô hình thiên lệch.

Ba mô hình học máy gồm Random Forest, XGBoost và CatBoost được lựa chọn để thực nghiệm so sánh. Random Forest hoạt động dựa trên cơ chế xây dựng nhiều cây quyết định độc lập và tổng hợp kết quả bằng phương pháp bỏ phiếu nhằm tăng tính ổn định của mô hình. Trong khi đó, XGBoost sử dụng kỹ thuật gradient boosting kết hợp regularization L1/L2 để giảm overfitting và tăng tốc độ huấn luyện. Đối với CatBoost, mô hình được tối ưu hóa cho dữ liệu phân loại và có khả năng xử lý hiệu quả các đặc trưng dạng chuỗi mà không cần mã hóa thủ công phức tạp.

Thực nghiệm được tiến hành theo hai kịch bản: không sử dụng SMOTE và có sử dụng SMOTE, nhằm đánh giá tác động của kỹ thuật cân bằng dữ liệu đến từng mô hình.

2.3 Các chỉ số đánh giá

Bài toán phân loại nhị phân trong nghiên cứu được đánh giá xoay quanh các chỉ số mô tả trong Bảng 2.

Bảng 2 Các thành phần của ma trận nhầm lẫn (Confusion Matrix)

Ký hiệu	Tên gọi	Mô tả
TP	True Positive	Số mẫu dương tính được dự đoán đúng
TN	True Negative	Số mẫu âm tính được dự đoán đúng
FP	False Positive	Số mẫu âm tính bị dự đoán nhầm thành dương tính
FN	False Negative	Số mẫu dương tính bị dự đoán nhầm thành âm tính

Accuracy đo lường tỷ lệ dự đoán đúng trên toàn bộ tập dữ liệu. Tuy nhiên, trong trường hợp dữ liệu mất cân bằng (ví dụ số lượng URL an toàn quá nhiều so với URL độc hại), Accuracy cao chưa chắc mô hình thực sự tốt.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Precision cho biết có bao nhiêu phần trăm thực sự đúng.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Recall cho biết mô hình phát hiện được bao nhiêu phần trăm.

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

F1-Score là trung bình điều hòa giữa Precision và Recall, cho phép đánh giá cân bằng khi cần tối ưu cả hai.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

ROC Curve (Receiver Operating Characteristic Curve) là đồ thị biểu diễn mối quan hệ giữa tỷ lệ dương tính đúng (True Positive Rate – TPR) và tỷ lệ dương tính giả (False Positive Rate – FPR) tại các ngưỡng phân loại khác nhau.

$$TPR = \frac{TP}{TP + FN} \quad (5)$$

$$FPR = \frac{FP}{FP + TN} \quad (6)$$

AUC (Area Under Curve) là diện tích dưới đường cong ROC, dùng để đo lường khả năng phân biệt giữa hai lớp của mô hình. Giá trị AUC nằm trong khoảng từ 0 đến 1. Chỉ số này càng gần 1 thể hiện mô hình phân loại tốt.

3 Kết quả nghiên cứu và thảo luận

3.1 Kết quả nghiên cứu

3.1.1 Kịch bản 1 – Không sử dụng SMOTE

Bảng 3 Phân tích kết quả của các mô hình với dữ liệu chưa sử dụng SMOTE

Model	Accuracy	Precision	Recall	F1	AUC-ROC	Time(s)
RF	0,992	0,993	0,998	0,995	0,992	34,250
XGB	0,990	0,991	0,998	0,995	0,994	9,030
CatBoost	0,989	0,991	0,998	0,994	0,992	22,990

Trong Bảng 3, kết quả cho thấy cả ba mô hình đều đạt hiệu suất rất cao trên tập kiểm tra. Random Forest đạt độ chính xác cao nhất với Accuracy = 99,2 %, trong khi XGBoost nổi bật với AUC-ROC cao nhất (99,4 %) và thời gian huấn luyện ngắn nhất chỉ 9,03 giây, nhanh hơn gần 4 lần so với Random Forest (34,25 giây) và gần 2,5 lần so với CatBoost (22,99 giây). Đáng chú ý, cả ba mô hình đều đạt Recall = 99,8 %, phản ánh khả năng phát hiện URL độc hại rất tốt ngay cả khi dữ liệu chưa được cân bằng.

3.1.2 Kịch bản 2 – Sử dụng SMOTE

Bảng 4 Phân tích kết quả của các mô hình với dữ liệu sử dụng SMOTE

Model	Accuracy	Precision	Recall	F1	AUC-ROC	Time(s)
RF	0,992	0,994	0,996	0,995	0,994	64,780
XGB	0,986	0,994	0,990	0,992	0,991	13,240
CatBoost	0,989	0,994	0,993	0,994	0,993	38,600

Trong Bảng 4, sau khi áp dụng SMOTE, Random Forest tiếp tục duy trì vị trí dẫn đầu với Accuracy = 99,2 %, đồng thời cải thiện AUC-ROC từ 99,2 % lên 99,4 %. CatBoost cũng cho thấy sự ổn định với các chỉ số được giữ vững. Tuy nhiên, XGBoost có sự sụt giảm nhẹ ở Accuracy (từ 99,0 % xuống 98,6 %) và AUC-ROC (từ 99,4 % xuống 99,1 %), cho thấy mô hình này nhạy cảm hơn với các mẫu tổng hợp do SMOTE sinh ra. Bên cạnh đó, thời gian huấn luyện của tất cả các mô hình đều tăng lên đáng kể do tập dữ liệu được mở rộng, trong đó Random Forest tăng gần gấp đôi (từ 34,25 giây lên 64,78 giây).

3.2 Thảo luận

Khi so sánh kết quả của nghiên cứu này với các nghiên cứu khác, ta có bảng so sánh hiệu suất theo Bảng 5 như sau:

Bảng 5 So sánh các nghiên cứu phát hiện website lừa đảo và nghiên cứu đề xuất

Nghiên cứu	Phương pháp/Đặc trưng	Xử lý mất cân bằng	Hiệu suất (Acc/AUC) %
[7] Aljofey et al. (2022)	Kết hợp đặc trưng URL và nội dung HTML	Không đề cập	Acc ~ 98
[8] Prasad & Chandra (2024)	Chỉ số tương đồng (Similarity index) & Học tăng dần (Incremental learning)	Học tăng dần	Acc ~ 97,5
[11] Liu et al. (2024)	TransURL (Kiến trúc Transformer đa tầng)	Xử lý mất cân bằng	AUC 0,99+
Nghiên cứu này	**9 đặc trưng thủ công tối ưu + SMOTE + So sánh 3 mô hình Ensemble**	**SMOTE (có phân tích tác động riêng biệt lên từng họ thuật toán)**	**Acc > 98,6 AUC > 99,1**

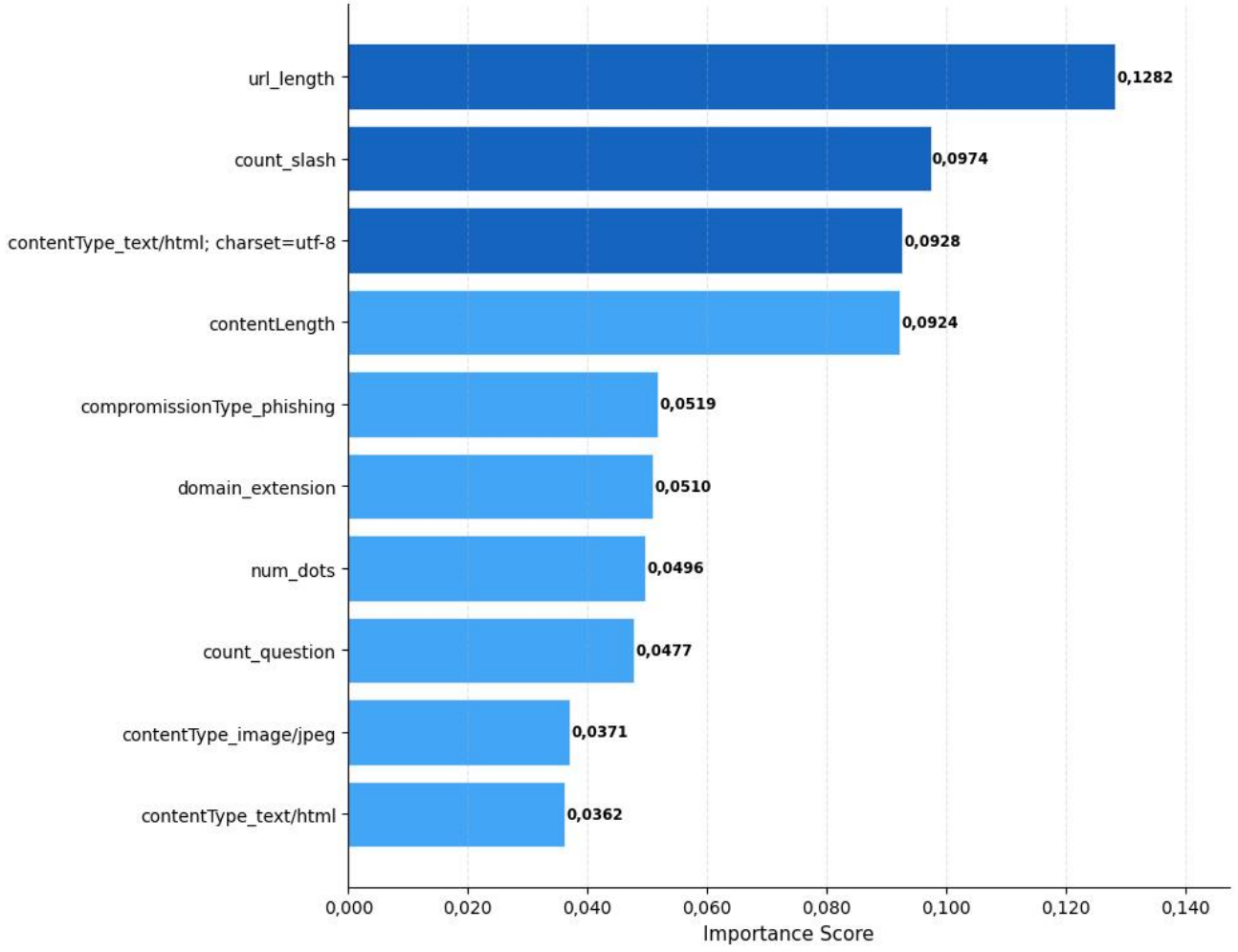
Dựa trên bảng so sánh trên, nghiên cứu này khẳng định tính mới thông qua hai điểm sau:

1. Tính robust (bền vững) trước các mối đe dọa Zero-day: trong khi các phương pháp dựa trên "chỉ số tương đồng" [8] có thể thất bại trước các URL lừa đảo hoàn toàn mới, các đặc trưng do chúng tôi thiết kế (như count_slash, url_length, domain_extension) khai thác các quy luật cấu trúc bất biến của kỹ thuật

obfuscation (làm rối) URL. Do đó, mô hình có khả năng tổng quát hóa tốt hơn ngay cả khi chưa từng gặp mẫu tấn công đó trước đây.

2. Phân tích thực nghiệm sâu về tác động của SMOTE lên các họ thuật toán khác nhau: đây là điểm đóng góp khoa học đặc thù của nghiên cứu. Thay vì áp dụng SMOTE một cách máy móc như một "hộp đen", nghiên cứu đã chứng minh thực nghiệm rằng SMOTE có tác động phân hóa rõ rệt: nó cải thiện đáng kể hiệu suất của mô hình dựa trên cơ chế bỏ phiếu (Bagging – Random Forest) nhờ khả năng kháng nhiễu, nhưng lại gây suy giảm nhẹ ở mô hình Gradient Boosting (XGBoost). Nguyên nhân là do các mẫu tổng hợp được nội suy tuyến tính có thể tạo ra nhiễu tại vùng biên quyết định (decision boundary), ảnh hưởng đến quá trình tối ưu hóa gradient. Phát hiện này cung cấp một hướng dẫn thực tiễn quý giá cho các nhà nghiên cứu: việc lựa chọn kỹ thuật cân bằng dữ liệu phải được điều chỉnh tương thích với bản chất toán học của thuật toán phân loại.

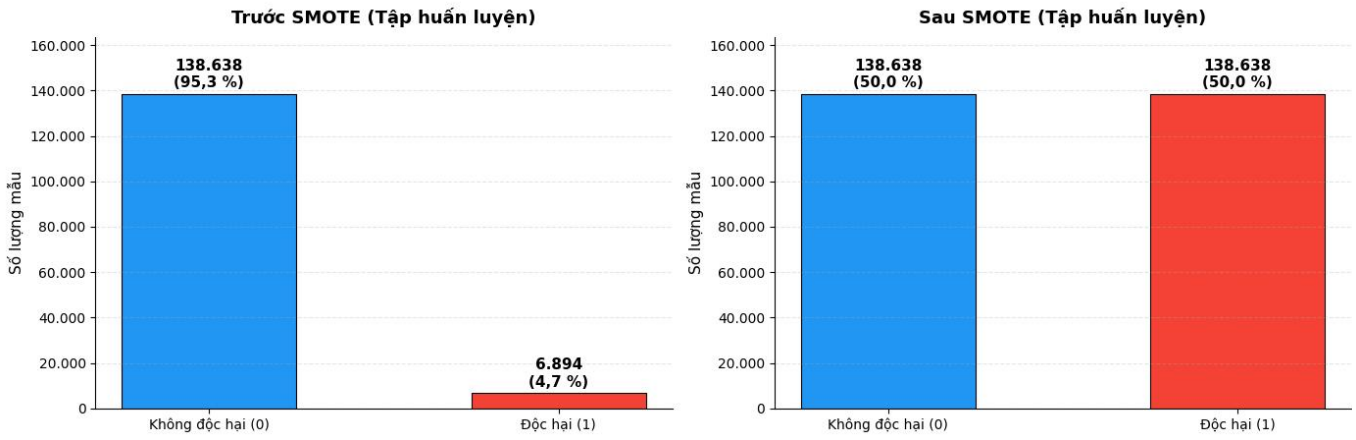
3.2.1 Về hiệu quả của quy trình trích xuất đặc trưng



Hình 4 Xếp hạng 10 đặc trưng quan trọng của Random Forest (SMOTE)

Một điểm đáng chú ý là cả ba mô hình đều đạt Recall rất cao (99,8 %) ngay cả ở kịch bản không sử dụng SMOTE, dù tập dữ liệu gốc mất cân bằng nghiêm trọng với tỷ lệ 95,3 % URL an toàn. Qua đó thấy rằng các đặc trưng được thiết kế trong nghiên cứu mang lại tín hiệu phân biệt đủ mạnh để các mô hình nhận diện URL độc hại mà không cần phải dựa hoàn toàn vào việc cân bằng dữ liệu.

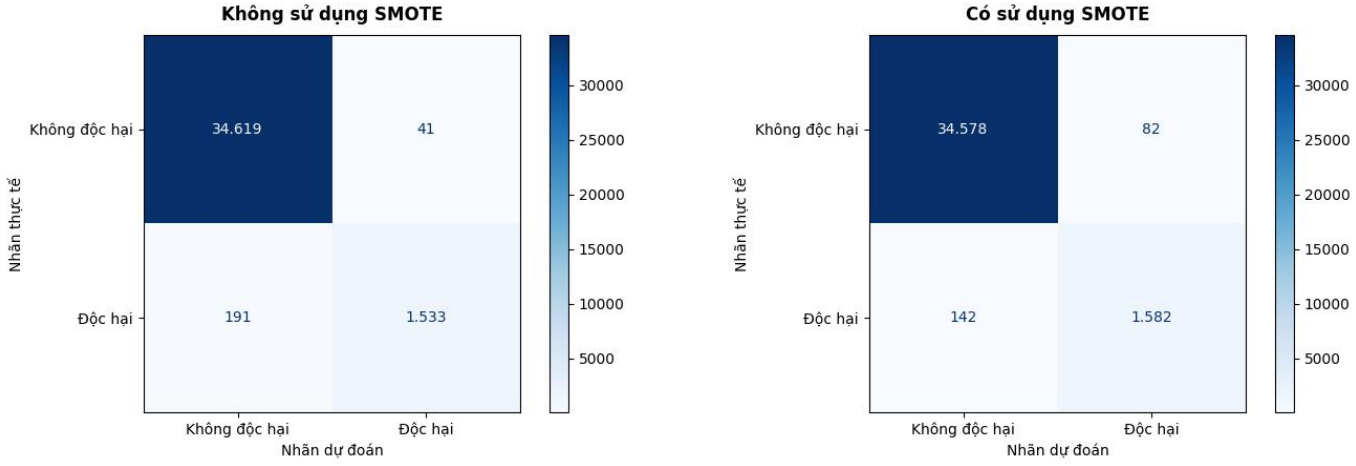
Như mô tả trong Hình 4, đặc trưng *url_length* (0,1282) giữ vị trí then chốt bởi các URL độc hại thường được kéo dài nhằm che giấu tên miền thực hoặc nhúng thêm các tham số lừa đảo.



Hình 5 Cải thiện phân phối nhãn khi dùng SMOTE

Tiếp đến là *count_slash* (0,0974), nơi mật độ dấu gạch chéo cao phản ánh một cấu trúc đường dẫn bất thường, dấu hiệu điển hình của các trang web giả mạo. Bên cạnh đó, các yếu tố kỹ thuật như *contentType_text/html;charset=utf-8* (0,0928) giúp mô hình thiết lập cơ sở dữ liệu về tài nguyên chuẩn, trong khi *contentLength* (0,0924) hỗ trợ phát hiện các thay đổi nội dung đột ngột do bị tấn công thay đổi giao diện hoặc lừa đảo. Những kết quả này một lần nữa khẳng định rằng trong xây dựng mô hình học máy, chất lượng của việc trích xuất đặc trưng đóng vai trò quyết định, quan trọng tương đương với việc lựa chọn và tối ưu hóa thuật toán.

Sử dụng SMOTE giúp cân bằng nhãn từ đó cải thiện hiệu suất mô hình, mô tả trong Hình 5 cho thấy SMOTE can thiệp vào phân phối nhãn. Kết quả thực nghiệm cho thấy, SMOTE mang lại lợi ích rõ rệt nhất đối với Random Forest và CatBoost, trong khi XGBoost lại có sự sụt giảm nhẹ sau khi áp dụng. Điều này có thể được lý giải bởi bản chất của gradient boosting trong XGBoost: thuật toán vốn nhạy cảm với phân phối dữ liệu huấn luyện, do đó các mẫu tổng hợp do SMOTE sinh ra bằng phép nội suy tuyến tính có thể tạo ra nhiễu cục bộ, ảnh hưởng đến quá trình tối ưu hóa gradient.



Hình 6 So sánh ma trận nhầm lẫn của mô hình Random Forest dưới sự tác động của SMOTE

Ngược lại, Random Forest với cơ chế bỏ phiếu đa số có khả năng hấp thụ nhiễu tốt hơn, nên hưởng lợi nhiều hơn từ việc cân bằng dữ liệu. Trong bối cảnh bảo mật mạng, nơi bỏ sót một URL độc hại thường gây hậu quả nghiêm trọng hơn nhiều so với cảnh báo nhầm, việc ưu tiên Recall cao là hoàn toàn hợp lý. Do đó, kịch bản sử dụng SMOTE vẫn được khuyến nghị, dù đánh đổi bằng thời gian huấn luyện dài hơn, kết quả được chỉ ra trong Hình 6.

3.2.2 Về sự lựa chọn mô hình phù hợp theo ngữ cảnh triển khai

Ba mô hình thể hiện những ưu thế khác nhau phù hợp với từng kịch bản triển khai thực tế. Random Forest phù hợp nhất khi yêu cầu độ ổn định và độ tin cậy cao, chấp nhận thời gian huấn luyện dài hơn. XGBoost là lựa chọn lý tưởng khi cần cân bằng giữa hiệu suất và tốc độ, đặc biệt phù hợp với các hệ thống phát hiện thời gian thực do thời gian huấn luyện ngắn nhất (9,03 giây). CatBoost thể hiện lợi thế khi tập dữ liệu có nhiều biến phân loại, phù hợp với các hệ thống mà dữ liệu đầu vào thường xuyên thay đổi cấu trúc mà không cần tái mã hóa thủ công.

4 Kết luận và đề xuất

4.1 Kết luận

Nghiên cứu này đề xuất và đánh giá một pipeline phát hiện URL độc hại dựa trên học máy, bao gồm các giai đoạn tiền xử lý, trích xuất đặc trưng, cân bằng dữ liệu bằng SMOTE và huấn luyện ba mô hình ensemble là Random Forest, XGBoost và CatBoost. Kết quả thực nghiệm trên tập dữ liệu 181 916 mẫu cho thấy cả ba mô hình đều đạt hiệu suất vượt trội so với các phương pháp truyền thống, với Accuracy trên 98,6 % và AUC-ROC trên 99,1 %. Trong đó, Random Forest thể hiện độ ổn định cao nhất với Accuracy = 99,21 % và AUC-ROC = 99,4 %, XGBoost cân bằng tốt giữa hiệu suất và tốc độ huấn luyện chỉ 9,03 giây, còn CatBoost phù hợp với dữ liệu có nhiều biến phân loại với Accuracy = 98,9 %.

Việc áp dụng SMOTE giúp duy trì Recall ở mức cao mà không làm suy giảm đáng kể các chỉ số khác, phù hợp với yêu cầu của bài toán bảo mật nơi việc bỏ sót URL độc hại gây hậu quả nghiêm trọng hơn cảnh báo nhầm. Bên cạnh đó, quy trình trích xuất 9 đặc trưng bổ sung từ chuỗi URL được chứng minh là yếu tố then chốt giúp nâng cao khả năng phân biệt giữa URL an toàn và URL độc hại.

4.2 Đề xuất

Thứ nhất, việc cải tiến kỹ thuật cân bằng dữ liệu là cần thiết, trong đó các phương pháp như ADASYN hoặc sự kết hợp giữa oversampling và undersampling có thể thay thế SMOTE để xử lý tốt hơn các mẫu phi tuyến phức tạp và giảm thiểu nhiễu.

Thứ hai, tính minh bạch của hệ thống cần được chú trọng thông qua các kỹ thuật giải thích mô hình như SHAP hoặc LIME, giúp chuyên gia bảo mật hiểu rõ các đặc trưng quyết định trong từng phân loại. Tích hợp có chọn lọc các mô hình học sâu và hệ thống giải thích (XAI): thay vì thay thế hoàn toàn, nghiên cứu đề xuất kiến trúc lai (Hybrid Architecture). Cụ thể, sử dụng các mô hình ngôn ngữ tiền huấn luyện chuyên biệt cho URL như các mô hình Bert để trích xuất vector embedding ngữ cảnh, sau đó nối (concatenate) vector này với 9 đặc trưng thủ công đã được chứng minh hiệu quả trong nghiên cứu này. Đầu ra sẽ được đưa vào XGBoost để phân loại, giúp tận dụng cả sức mạnh biểu diễn sâu và tốc độ suy luận của cây quyết định. Đồng thời, tích hợp thư viện SHAP (Shapley Additive exPlanations) vào pipeline. Thay vì chỉ đưa ra nhãn "độc hại", hệ thống sẽ xuất ra biểu đồ SHAP waterfall cho từng URL, ví dụ: chỉ ra chính xác việc count_slash > 5 và domain_extension lạ đóng góp + 0,45 vào điểm số độc hại, giúp chuyên gia bảo mật ra quyết định nhanh chóng và minh bạch.

Thứ ba, triển khai thực tế nghiên cứu này, pipeline này được thiết kế để dễ dàng đóng gói dưới dạng dịch vụ API RESTful hoặc module plugin, tích hợp trực tiếp vào hệ thống proxy, firewall hoặc trung tâm điều hành an ninh mạng (SOC) của doanh nghiệp. Với ưu thế về tốc độ huấn luyện và suy luận, mô hình XGBoost đặc biệt phù hợp cho các hệ thống yêu cầu phát hiện và chặn URL độc hại theo thời gian thực (real-time), trong khi Random Forest có thể được sử dụng ở chế độ phân tích sâu (deep analysis) cho các luồng traffic đáng ngờ.

Cuối cùng, hướng tới việc triển khai hệ thống trong thời gian thực với cơ chế cập nhật liên tục sẽ là chìa khóa để thích nghi với các chiến thuật tấn công mạng ngày càng tinh vi và luôn thay đổi.

Tài liệu tham khảo

1. Asiri, S., Xiao, Y., Alzahrani, S., Li, S., & Li, T. (2023). A survey of intelligent detection designs of HTML URL phishing attacks. *IEEE Access*, *11*, 6421-6443. https://doi.org/10.1109/ACCESS.2023.3237798.
2. Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., & Fujita, H. (2022). Deep learning for phishing detection: Taxonomy, current challenges and future directions. *IEEE Access*, *10*, 36429-36463. https://doi.org/10.1109/ACCESS.2022.3151903.
3. Tang, L., & Mahmoud, Q. H. (2022). A deep learning-based framework for phishing website detection. *IEEE Access*, *10*, 1509-1521. https://doi.org/10.1109/ACCESS.2021.3137636.
4. Rezaie, A., Moattar, M. H., & Ghaffari, M. (2025). Leveraging machine learning to proactively identify phishing campaigns before they strike. *Journal of Big Data*, *12*, 68. https://doi.org/10.1186/s40537-025-01174-x.
5. Uddin, M. A., Alkhamis, A., Alotaibi, F., Alshehri, A., Alam, M. M., & Ahmad, N. (2025). Real-time phishing detection for brand protection using temporal convolutional network-driven URL sequence modeling. *Electronics*, *14*(18), 3746. https://doi.org/10.3390/electronics14183746.
6. Safi, A., & Singh, S. (2023). A systematic literature review on phishing website detection techniques. *Journal of King Saud University – Computer and Information Sciences*, *35*(3), 590-611. https://doi.org/10.1016/j.jksuci.2023.01.004.
7. Aljofey, A., Jiang, Q., Rasool, A., Chen, H., Liu, W., Qu, Q., & Wang, Y. (2022). An effective detection approach for phishing websites using URL and HTML features. *Scientific Reports*, *12*, 8842. https://doi.org/10.1038/s41598-022-10841-5.
8. Prasad, A., & Chandra, S. (2024). PhiUSIIL: A diverse security profile empowered phishing URL detection framework based on similarity index and incremental learning. *Computers & Security*, *136*, 103545. https://doi.org/10.1016/j.cose.2023.103545.
9. Almujahid, N. F., Haq, M. A., & Alshehri, M. (2024). Comparative evaluation of machine learning algorithms for phishing site detection. *PeerJ Computer Science*, *10*, e2131. https://doi.org/10.7717/peerj-cs.2131.
10. Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., & Faheem, M. H. (2023). A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. *Electronics*, *12*(1), 232. https://doi.org/10.3390/electronics12010232.
11. Liu, R., Wang, Y., Guo, Z., Xu, H., Qin, Z., Ma, W., & Zhang, F. (2024). TransURL: Improving malicious URL detection with multi-layer Transformer encoding and multi-scale pyramid features. *Computer Networks*, *253*, 110707. https://doi.org/10.1016/j.comnet.2024.110707.
12. Li, Y., Liu, Y., Li, P., Jia, Y., & Wang, Y. (2025). Continuous multi-task pre-training for malicious URL detection and webpage classification. *Computer Networks*, *120*, 111513. https://doi.org/10.1016/j.comnet.2025.111513.
13. Yi, L., Omotosho, A., & Balogun, H. (2025). Phishing URL detection and interpretability with machine learning: A cross-dataset approach. *Security and Privacy*, *9*(1), e70175. https://doi.org/10.1002/spy2.70175.
14. Abdalla, H. B., & Shekh, A. (2023). Analysis of the performance impact of fine-tuned machine learning model for phishing URL detection. *Electronics*, *12*(7), 1642. https://doi.org/10.3390/electronics12071642.
15. Catal, C., Giray, G., Tekinerdogan, B., Kumar, S., & Shukla, S. (2022). Applications of deep learning for phishing detection: A systematic literature review. *Knowledge and Information Systems*, *64*(6), 1457-1500. https://doi.org/10.1007/s10115-022-01672-x.

A Machine Learning Pipeline for Malicious URL Detection with Feature Engineering and Class Imbalance Handling

Duong Hon Minh
Pharmacy Department, Nguyen Tat Thanh University, Ho Chi Minh City, Viet Nam
dhminh@ntt.edu.vn

Abstract Malicious URLs are one of the most common forms of cyber attacks, causing serious financial losses and security risks. Traditional approaches such as blacklist and heuristic methods have limitations in scalability and are not effective in detecting new threats. This study proposes a machine learning pipeline for malicious URL detection, including data preprocessing, extraction of nine additional features from URL strings, and the use of SMOTE to address class imbalance. Three models, namely Random Forest, XGBoost, and CatBoost, are trained and evaluated on a dataset of 181,916 samples. The results show that all three models achieve an accuracy above 98.6% and an AUC-ROC above 0.991, outperforming baseline methods. Random Forest provides the most stable performance, while XGBoost offers a good balance between performance and training speed. In addition, SMOTE helps maintain high recall without significantly reducing other evaluation metrics.

Keywords Malicious URL; Random Forest; XGBoost; CatBoost; SMOTE.