


# THÔNG TIN CHUNG CỦA BÁO CÁO

- Link YouTube video của báo cáo (tối đa 5 phút):  
<https://youtu.be/GgQTKuZkY8Y>
- Link slides (dạng .pdf đặt trên Github):  
<https://github.com/hieulx/CS2205.MAR2024/blob/main/Phan%20loan%20ma%20doc%20PE%20su%20dung%20machine%20learning.pdf>

|  |   |
|--|---|
| <ul style="list-style-type: none"><li>• Họ và Tên: Lê Xuân Hiếu</li><li>• MSSV: 230202026</li></ul>  | <ul style="list-style-type: none"><li>• Lớp: CS2205.MAR2024</li><li>• Tự đánh giá (điểm tổng kết môn): 7.5/10</li><li>• Số buổi vắng: 0</li><li>• Số câu hỏi QT cá nhân: 3</li><li>• Link Github:<br/><a href="https://github.com/hieulx/CS2205.MAR2024/">https://github.com/hieulx/CS2205.MAR2024/</a></li></ul> |
|--|---|

# ĐỀ CƯƠNG NGHIÊN CỨU

## TÊN ĐỀ TÀI (IN HOA)

PHÂN LOẠI MÃ ĐỘC PE SỬ DỤNG MACHINE LEARNING

## TÊN ĐỀ TÀI TIẾNG ANH (IN HOA)

PE MALWARE CLASSIFICATION USING MACHINE LEARNING

## TÓM TẮT

Trong thế giới kỹ thuật số ngày nay, việc phân loại mã độc là một thách thức không ngừng được quan tâm do sự gia tăng liên tục của các mối đe dọa an ninh mạng. Số lượng mối đe dọa từ phần mềm độc hại (malware) ngày càng gia tăng khiến các phương pháp phát hiện dựa theo signatures truyền thống không đủ khả năng để chống lại các cuộc tấn công mới và phức tạp. Do đó, việc áp dụng học máy đã nổi lên như một phương pháp hữu hiệu để phát hiện mã độc.

Nghiên cứu sau đây tập trung vào việc triển khai các phương pháp học máy để phân loại và phát hiện malware. Nghiên cứu đánh giá hiệu quả của một số thuật toán, bao gồm: K-Nearest Neighbor (KNN), Decision Tree và Logistic Regression. Đánh giá này được thực hiện thông qua việc kiểm tra một bộ dữ liệu công khai có chứa cả tệp tin lành tính và các mẫu malware (dataset BODMAS). Đồng thời, nghiên cứu sử dụng một mô hình ensemble kết hợp ba mô hình đã huấn luyện thành một mô hình tốt hơn.

Ngoài ra nghiên cứu còn khám phá ảnh hưởng của các tập tính năng và kỹ thuật tiền xử lý khác nhau đến hiệu suất của bộ phân loại. Kết quả từ nghiên cứu này không chỉ cung cấp cái nhìn sâu sắc về hiệu suất của các thuật toán trong việc phát hiện mã độc mà còn đề xuất hướng phát triển tiềm năng cho các phương pháp phân loại mã độc trong tương lai.

## GIỚI THIỆU (*Tối đa 1 trang A4*)

Phần mềm antivirus truyền thống sử dụng phương pháp signatures để phát hiện mã độc. Các phần mềm này thường quét và nhận diện mã độc dựa trên một cơ sở dữ liệu lớn đã được phân tích từ các mã độc được phát hiện trước đó. Khi quét một tệp,

signatures sẽ được tính toán và so sánh với cơ sở dữ liệu đã biết, nếu trùng khớp, tệp đó sẽ được đánh dấu là mã độc. Tuy nhiên với sự phát triển của các cuộc tấn công hiện đại và tinh vi, phần mềm antivirus truyền thống đang khó khăn trong việc phát hiện các malware mới. Việc nghiên cứu sử dụng các kỹ thuật tiên tiến để chọn ra các đặc trưng có liên quan, cải thiện độ chính xác trong việc phát hiện phần mềm độc hại, đồng thời giảm kích thước dữ liệu đã và đang được nghiên cứu và cải thiện. Vì thế, trong nghiên cứu này, tôi sẽ sử dụng các thuật toán học máy phân loại khác nhau để phát hiện malware. Các kỹ thuật này cải thiện độ chính xác, cho phép phát hiện tự động, hiệu quả, tiết kiệm thời gian và tài nguyên.

- + Input: tập tin PE malware và benign (lành tính) .

Sử dụng dataset BODMAS và mô hình học máy: K-Nearest Neighbor, Decision Tree, Logistic Regression và Ensemble

- + Output: phân loại được mẫu đưa vào là malware hay benign, hiệu suất của các mô hình học máy sử dụng trong phát hiện và phân loại malware

## **MỤC TIÊU**

- + Phát hiện được tập tin PE đưa vào là malware hay benign (lành tính)
- + Phân loại tập tin malware thuộc họ nào của malware
- + Đánh giá hiệu suất phân loại malware tốt nhất trong các mô hình học máy Logistic Regression (LR), K-Nearest Neighbor (KNN), Decision Tree và Ensemble

## **NỘI DUNG VÀ PHƯƠNG PHÁP**

Nội dung:

- + Các công trình liên quan và hạn chế
- + Sử dụng bộ dataset BODMAS để huấn luyện và kiểm thử .
- + Tiền xử lý các dữ liệu trong bộ dataset, trích xuất các đặc trưng và đánh nhãn . Tạo ra file metadata để tăng thêm hiệu quả đánh giá
- + Chia các dữ liệu để huấn luyện và kiểm thử ( 80%-20%) cho các mô hình LR,

KNN, Decision Tree và Ensemble (kết hợp các mô hình trên)

+ Đánh giá hiệu suất của các mô hình khi có và không sử dụng metadata, trong cả phân loại nhị phân và phân loại đa nhãn

Phương pháp thực hiện:

+Tìm hiểu các bài báo có nội dung liên quan tới malware classification sử dụng machine learning, hỏi giảng viên hướng dẫn

+ Download bộ dataset BODMAS [1] của tác giả Yang Linmin[1] theo hướng dẫn của giảng viên hướng dẫn

+ Trích xuất đặc trưng dựa trên dự án LIEF [2], kỹ thuật trích xuất giống với dự án Ember [3] ( ByteHistogram, ByteEntropyHistogram, StringExtractor, GeneralFileInfo, HeaderFileInfo, SectionInfo, ImportsInfo, ExportsInfo, DataDirectories) , đưa ra dưới dạng vector.

+ Tập đưa vào bodmas.npz chứa hai phần, đặc trưng và nhãn. Tạo thêm file metadata.csv ( timestamp, family) để đưa vào huấn luyện và kiểm thử các mô hình

+ Chọn tiêu chí đánh giá hiệu suất mô hình Accuracy score, Precision score, Recall score, sử dụng hàm đánh giá của thư viện Sklearn

+ Xuất kết quả ra dạng bảng biểu và tiến hành so sánh khi có và không sử dụng metadata, trong trường hợp phân loại nhị phân và phân loại đa nhãn

## **KẾT QUẢ MONG ĐỢI**

+Áp dụng thành công mô hình LR, KNN, Decision Tree, Ensemble trong việc phân loại mã độc.

+ Bản biểu đồ tổng quan mô hình thực nghiệm.

+ Xuất được bảng biểu hiển thị thông tin hiệu suất mô hình qua các tiêu chí đánh giá, trong phân loại nhị phân và phân loại đa nhãn ( có và không sử dụng metadata)

## **TÀI LIỆU THAM KHẢO**

[1]. Y. Limin, A. Ciptadi, I. Laziuk, A. Ahmadzadeh and G. Wang, “BODMAS,” 2019. [Online]. Available:

[https://liminyang.web.illinois.edu/slides/DLS21\\_BODMAS\\_LiminYang.pdf](https://liminyang.web.illinois.edu/slides/DLS21_BODMAS_LiminYang.pdf).

[2] R. Thomas, “LIEF - Library to Instrument Executable Formats,” April 2017. [Online]. Available: <https://lief.re/>.

[3]H. Anderson and P. Roth, “EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models,” April 2018. [Online]. Available: <https://github.com/elastic/ember/blob/master/ember/features.py>.