ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

--------o0o--------

BÁO CÁO ĐỒ ÁN

ĐỀ TÀI: An effective genetic algorithm-based feature selection method for intrusion detection systems

Giảng viên hướng dẫn : Đỗ Hoàng Hiển

Sinh viên thực hiện:

1. Bùi Quốc Huy – 21520911
2. Nguyễn Long Vũ – 21522800
3. Bùi Đức Anh Tú – 21522735
4. Lê Huy Hiệp – 21522067

Lớp : **NT204.O21.ANTT**

TP. Hồ Chí Minh, 18 tháng 5 năm 2024

LỜI CẢM ƠN

Lời đầu tiên, xin chân thành cảm ơn Ban Giám hiệu Trường Đại học Công nghệ thông tin,khoa Mạng máy tính và truyền thông và quý thầy Đỗ Hoàng Hiển đã tạo nền tảng, điều kiện giúp chúng tôi có hội tiếp cận và nghiên cứu chủ đề này.

Chúng tôi cũng muốn bày tỏ lòng biết ơn sâu sắc đến các tác giả của bài báo khoa học mà chúng tôi đã sử dụng làm cơ sở cho nghiên cứu của mình. Công trình nghiên cứu của quý vị đã cung cấp cho tôi cơ sở lý thuyết và dữ liệu quan trọng để phát triển và thực hiện dự án này.

Trong quá trình nghiên cứu về chủ đề này, có thể do hiểu biết còn nhiều hạn chế nên bài làm khó tránh khỏi những thiếu sót. Chúng tôi rất mong nhận được những lời góp ý chân thành của thầy để bài báo cáo ngày càng hoàn thiện hơn.

Trân trọng.

NHẬN XÉT CỦA GIẢNG VIÊN

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

MỤC LỤC

LỜI CẢM ƠN 2

NHẬN XÉT CỦA GIẢNG VIÊN 3

MỤC LỤC 4

DANH MỤC HÌNH ẢNH 6

DANH MỤC BẢNG 8

DANH MỤC VIẾT TẮT 9

1. Giới thiệu 10

1.1. Tổng quan đề tài 10

1.2. Mục tiêu và Phạm vi nghiên cứu 10

1.3. Phương pháp nghiên cứu 10

2. Cơ sở lý thuyết và các nghiên cứu liên quan 10

2.1. Khái niệm về hệ thống phát hiện xâm nhập (IDS) 10

2.2. Các loại hệ thống phát hiện xâm nhập 11

2.2.1. IDS dựa trên dấu hiệu (Signature-based IDS) 11

2.2.2. IDS dựa trên bất thường (Anomaly-based IDS) 11

2.3. Vai trò của chọn lọc đặc trưng trong IDS 11

2.4. Khái niệm về thuật toán di truyền 11

2.5. Các nghiên cứu liên quan 12

3. Phân tích thiết kế hệ thống 15

3.1. Tổng quan hệ thống 15

3.2. Cấu trúc của thuật toán di truyền 16

3.2.1. Khởi tạo quần thể 17

3.2.2. Tính giá trị thích nghi 18

3.2.3. Lựa chọn "cha mẹ" (Parents Selection) 18

3.2.4. Lai chéo (Crossover) 20

3.2.5. Đột biến (Mutation) 20

3.3. Ứng dụng của thuật toán di truyền trong học máy và khai phá dữ liệu 21

4. Hiện thực hệ thống 21

4.1. Môi trường thực nghiệm 21

4.2. Datasets **22**
4.3. Các bộ phân loại **25**
4.4. Quy trình triển khai **26**
5. Thực nghiệm và đánh giá **26**
5.1. Thực nghiệm **27**
5.1.1. CIRA-CIC-DOHBrw-2020 ..... **27**
5.1.2. UNSW-NB15 ..... **33**
5.1.3. Bot-IoT 5% **40**
5.2. Đánh giá kết quả **46**
5.2.1. CIRA-CIC-DOHBrw-2020 ..... **46**
5.2.2. UNSW-NB15 ..... **46**
5.2.3. Bot-IoT 5% **47**
5.2.4. Đánh giá chung **47**
6. Kết luận và hướng phát triển **48**
6.1. Kết luận **48**
6.2. Hướng phát triển **48**
7. Tài liệu tham khảo **49**

DANH MỤC HÌNH ẢNH

Hình 1: Thiết kế hệ thống của bài báo 15
Hình 2: Minh họa thiết kế hệ thống của đồ án (có GbFS) 16
Hình 3: Minh họa "khởi tạo quần thể ban đầu" 17
Hình 4: Minh họa "bánh xe roulette" 19
Hình 5: Minh họa "đột biến" và "lai chéo" 20
Hình 6: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (1) 22
Hình 7: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (2) 22
Hình 8: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (3) 22
Hình 9: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của bài báo 23
Hình 10: Thống kê chung về các dataset của bài báo 23
Hình 11: Thống kê về dataset Bot-IoT 5% của nhóm (1) 23
Hình 12: Thống kê về dataset Bot-IoT 5% của nhóm (2) 24
Hình 13: Thống kê về dataset Bot-IoT 5% của nhóm (3) 24
Hình 14: Thống kê về dataset Bot-IoT 5% của bài báo 24
Hình 15: Thống kê về dataset UNSW-NB15 của nhóm (1) 24
Hình 16: Thống kê về dataset UNSW-NB15 của nhóm (2) 25
Hình 17: Thống kê về dataset UNSW-NB15 của nhóm (3) 25
Hình 18: Thống kê về dataset UNSW-NB15 của bài báo 25
Hình 19: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset CIRA-CIC-DOHBrw-2020 27
Hình 20: Kết quả đánh giá của 6 mô hình của bài báo với dataset CIRA-CIC-DOHBrw-2020 27
Hình 21: Kết quả chạy GbFS với dataset CIRA-CIC-DOHBrw-2020 28
Hình 22: Kết quả đánh giá mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS 28
Hình 23: Độ chính xác của mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS 28
Hình 24: Kết quả đánh giá mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS 29
Hình 25: Độ chính xác của mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS 29
Hình 26: Kết quả đánh giá mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS 30
Hình 27: Độ chính xác của mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS 30
Hình 28: Kết quả đánh giá mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS 30
Hình 29: Độ chính xác của mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS 31
Hình 30: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset UNSW-NB15 33
Hình 31: Kết quả đánh giá của 6 mô hình của bài báo với dataset UNSW-NB15 33
Hình 32: Kết quả chạy GbFS với dataset UNSW-NB15 34

Hình 33: Kết quả đánh giá mô hình SVM với dataset UNSW-NB15 không dùng GbFS 34
Hình 34: Độ chính xác của mô hình SVM với dataset UNSW-NB15 không dùng GbFS 35
Hình 35: Kết quả đánh giá mô hình SVM với dataset UNSW-NB15 có dùng GbFS 35
Hình 36: Độ chính xác của mô hình SVM với dataset UNSW-NB15 có dùng GbFS 36
Hình 37: Kết quả đánh giá mô hình k-NN với dataset UNSW-NB15 không dùng GbFS 36
Hình 38: Độ chính xác của mô hình k-NN với dataset UNSW-NB15 không dùng GbFS 37
Hình 39: Kết quả đánh giá mô hình k-NN với dataset UNSW-NB15 có dùng GbFS 37
Hình 40: Độ chính xác của mô hình k-NN với dataset UNSW-NB15 có dùng GbFS 38
Hình 41: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset Bot-IoT 5%.....40
Hình 42: Kết quả đánh giá của 6 mô hình của bài báo với dataset Bot-IoT 5%...40
Hình 43: Kết quả chạy GbFS với dataset Bot-IoT 5% 41
Hình 44: Kết quả đánh giá mô hình SVM với dataset Bot-IoT 5% không dùng GbFS 41
Hình 45: Độ chính xác của mô hình SVM với dataset Bot-IoT 5% không dùng GbFS 41
Hình 46: Kết quả đánh giá mô hình SVM với dataset Bot-IoT 5% có dùng GbFS 42
Hình 47: Độ chính xác của mô hình SVM với dataset Bot-IoT 5% có dùng GbFS 42
Hình 48: Kết quả đánh giá mô hình k-NN với dataset Bot-IoT 5% không dùng GbFS 42
Hình 49: Độ chính xác của mô hình k-NN với dataset Bot-IoT 5% không dùng GbFS 43
Hình 50: Kết quả đánh giá mô hình k-NN với dataset Bot-IoT 5% có dùng GbFS 44
Hình 51: Độ chính xác của mô hình k-NN với dataset Bot-IoT 5% có dùng GbFS 44

DANH MỤC BẢNG

Bảng 1: Các mô hình sẽ huấn luyện 27
Bảng 2: So sánh không/có dùng GbFS với dataset CIRA-CIC-DOHBrw-2020....46
Bảng 3: So sánh không/có dùng GbFS với dataset UNSW-NB15 46
Bảng 4: So sánh không/có dùng GbFS với dataset Bot-IoT 5% 47

DANH MỤC VIẾT TẮT

Viết tắt	Viết đầy đủ
GbFS	GA – based Feature Selection
GA	Genetic Algorithm
IDS	Intrusion Detection System
CIRA	Canadian Internet Registration Authority
CIC	Canadian Institute for Cybersecurity
DoH	DNS over HTTPS
DNS	Domain Name System
HTTPS	Hypertext Transfer Protocol Security
Brw	Browser
BoT	BoTnets
IoT	Internet of Things
DoS	Denial of Service
DDoS	Distributed Denial of Service
UNSW	University of New South Wales
ACCS	Australian Centre for Cyber Security
NST	Nhiễm Sắc Thể

1. Giới thiệu

1.1. Tổng quan đề tài

Trong lĩnh vực bảo mật mạng, việc phát hiện và ngăn chặn các cuộc tấn công xâm nhập là một vấn đề quan trọng. Với sự phát triển của công nghệ và kết nối mạng, các mối đe dọa và tác động đến an ninh ngày càng phức tạp hơn. Để đảm bảo an toàn cho hệ thống mạng, việc tìm ra các phương pháp hiệu quả để phát hiện và đối phó với các cuộc tấn công xâm nhập là cực kỳ cần thiết.

Trong bài báo, nhóm tác giả tập trung vào vấn đề chọn đặc trưng trong hệ thống phát hiện xâm nhập và đề xuất một phương pháp chọn đặc trưng dựa trên thuật toán di truyền. Mục tiêu của đề tài là tìm ra một phương pháp chọn đặc trưng hiệu quả, giúp tăng độ chính xác của hệ thống phát hiện xâm nhập và giảm độ phức tạp tính toán.

Đề tài của nhóm sẽ là xây dựng lại và đánh giá phương pháp mà bài báo đề xuất.

1.2. Mục tiêu và Phạm vi nghiên cứu

Mục tiêu của bài báo là thiết kế một phương pháp giữ lại phần lớn thông tin duy nhất liên quan đến dữ liệu với số lượng đặc trưng tối thiểu. Nghiên cứu tập trung vào vấn đề lựa chọn đặc trưng trong lĩnh vực bảo mật mạng và phát hiện xâm nhập và đề xuất phương pháp chọn đặc trưng dựa trên thuật toán di truyền (GbFS) để tăng độ chính xác của các bộ phân loại. Nghiên cứu cũng tập trung vào việc điều chỉnh tham số cho phương pháp chọn đặc trưng dựa trên thuật toán di truyền và đề xuất một hàm thích nghi (fitness) mới.

Phạm vi của đồ án bao gồm thử nghiệm phương pháp trên ba bộ dữ liệu chuẩn và so sánh kết quả với khi không áp dụng phương pháp.

1.3. Phương pháp nghiên cứu

Nhóm đọc, nghiên cứu bài báo; tham khảo các thiết kế GbFS khác trên internet để thiết kế GbFS theo mô tả của bài báo.

Nhóm tìm các dataset qua internet và mô tả cảu bài báo.

Nhóm thiết kế thực nghiệm trên google colab (nền tảng hỗ trợ máy học miễn phí của google).

Dựa trên kết quả nhóm có được để đánh giá phương pháp.

2. Cơ sở lý thuyết và các nghiên cứu liên quan

2.1. Khái niệm về hệ thống phát hiện xâm nhập (IDS)

Hệ thống phát hiện xâm nhập (Intrusion Detection System - IDS) là một thành phần quan trọng trong bảo mật mạng, được sử dụng để giám sát và phát hiện các hoạt động xâm nhập hoặc không hợp lệ trên mạng. Mục tiêu của IDS là cung cấp khả năng phát hiện sớm các cuộc tấn công, bất thường hoặc hành vi đe dọa đến hệ thống mạng.

2.2. Các loại hệ thống phát hiện xâm nhập

Có hai loại chính của hệ thống phát hiện xâm nhập là IDS dựa trên dấu hiệu (Signature-based IDS) và IDS dựa trên bất thường (Anomaly-based IDS).

2.2.1. IDS dựa trên dấu hiệu (Signature-based IDS)

IDS dựa trên dấu hiệu hoạt động dựa trên việc so sánh các dấu hiệu hay mẫu đã biết của các cuộc tấn công với lưu lượng mạng đang đi qua hệ thống. Các dấu hiệu này được gọi là "chữ ký" và được tạo ra từ các mẫu tấn công đã biết trước. Khi một dấu hiệu tương tự được phát hiện trong gói tin hoặc luồng dữ liệu, IDS sẽ kích hoạt cảnh báo để xử lý sự cố.

2.2.2. IDS dựa trên bất thường (Anomaly-based IDS)

IDS dựa trên bất thường xác định các hành vi bất thường hoặc không bình thường trong mạng dựa trên một mô hình học máy hoặc mô hình thống kê. Thay vì dựa vào các chữ ký tấn công đã biết, IDS này tìm hiểu và xác định các mô hình hành vi bình thường của mạng. Bất kỳ hành vi nào không phù hợp với mô hình này sẽ được coi là bất thường và kích hoạt cảnh báo.

2.3. Vai trò của chọn lọc đặc trưng trong IDS

Trong hệ thống phát hiện xâm nhập, chọn lọc đặc trưng đóng vai trò quan trọng để cải thiện hiệu suất và hiệu quả của IDS. Đặc trưng là các thuộc tính của luồng dữ liệu mạng, ví dụ như các thông số giao thức, kích thước gói tin, thời gian phản hồi, v.v. Việc chọn lọc đặc trưng giúp giảm kích thước không gian đặc trưng, loại bỏ thông tin không cần thiết và tăng cường khả năng phát hiện các cuộc tấn công.

Bằng cách áp dụng phương pháp chọn đặc trưng dựa trên thuật toán di truyền, có thể tạo ra một tập hợp đặc trưng tối ưu, chỉ giữ lại những đặc trưng quan trọng nhất cho quá trình phát hiện xâm nhập. Việc chọn đặc trưng hiệu quả cung cấp một cơ sở dữ liệu tốt hơn và giúp giảm độ phức tạp tính toán của hệ thống phát hiện xâm nhập.

2.4. Khái niệm về thuật toán di truyền

Thuật toán di truyền là một thuật toán tối ưu hóa được lấy cảm hứng từ quá trình tiến hoá trong tự nhiên. Thuật toán này dựa trên các nguyên tắc của di truyền học và lý thuyết tiến hoá Darwin. Nó được sử dụng để tìm kiếm và tối ưu hóa các giải pháp trong không gian tìm kiếm lớn.

Thuật toán di truyền tạo ra một tập hợp các NST, mỗi NST đại diện cho một giải pháp trong không gian tìm kiếm. Các NST được biểu diễn dưới dạng chuỗi hoặc vector các gen, trong đó mỗi gen đại diện cho một thuộc tính của giải pháp. Các NST ban đầu được tạo ngẫu nhiên hoặc theo một qui luật cụ thể.

Quá trình tối ưu hóa trong thuật toán di truyền diễn ra qua các thế hệ. Mỗi thế hệ bao gồm các giai đoạn như lai ghép (crossover), đột biến (mutation), và lựa chọn (selection). Qua các giai đoạn này, các cá thể tốt nhất được duy trì và truyền dần từ thế hệ này sang thế hệ tiếp theo, trong khi các cá thể kém hơn bị loại bỏ.

2.5. Các nghiên cứu liên quan

Nghiên cứu của Stein và cộng sự (2005) đã giới thiệu phương pháp "lựa chọn đặc trưng" dựa trên GA. Các tác giả sử dụng Decision tree (DT) với GA. Họ áp dụng GA làm phương pháp "lựa chọn đặc trưng" trên tập dữ liệu KDD và phân loại dữ liệu thử nghiệm bằng bộ phân loại DT. Quần thể ban đầu được tạo ngẫu nhiên và được truyền lại cho quá trình "lựa chọn đặc trưng". Họ thực hiện lựa chọn dựa trên xếp hạng và chéo hai điểm để tạo ra quần thể mới. Cách tiếp cận của họ thực hiện đột biến cấp độ bit ở "con cái" (child) và sau đó giữ lại hai "bố mẹ" (parent) ưu tú và thay thế phần còn lại của quần thể. Độ thích hợp của "nhiễm sắc thể" được đánh giá bằng cách sử dụng tổng tỷ lệ lỗi được xác thực làm hàm fitness. Công trình của họ trình bày 32 đặc trưng tối ưu trong tổng số 41 thuộc tính của bộ dữ liệu KDD. Công trình của (Ho, 2006) cố gắng phát triển một IDS dựa trên "sự bất thường" để phát hiện các cuộc tấn công mới thông qua học tập không giám sát và một mô hình phân cụm ngẫu nhiên và lấy cảm hứng từ sinh học được gọi là Mô hình phân cụm đàn kiến (ACCM). Về học có giám sát, các tác giả đề xuất cơ chế phát hiện xâm nhập "mờ" (fuzzy) di truyền đa mục tiêu. Kỹ thuật của họ hoạt động giống như một phương pháp "lựa chọn đặc trưng" di truyền wrapper để tìm kiếm số đặc trưng tối ưu có thể biểu thị hầu hết thông tin liên quan đến dữ liệu. Để đánh giá độ chính xác của việc phát hiện, họ đưa ra 27 đặc trưng là đặc trưng tối ưu trong số 41 đặc trưng của bộ dữ liệu KDD. Nghiên cứu của họ đạt độ chính xác 99,24% trên bộ dữ liệu lưu lượng mạng KDD.

Trong "Ahmad và cộng sự. (2011)" các tác giả trình bày một phương pháp "lựa chọn đặc trưng" bằng cách sử dụng GA, Phân tích thành phần chính (PCA) và Perceptron đa lớp (MLP) để phát hiện xâm nhập. Mục đích nghiên cứu của họ là nâng cao tỷ lệ phát hiện của các "bộ phân loại" để phát hiện xâm nhập. Họ so sánh cách tiếp cận của họ với việc sử dụng PCA đơn giản. Họ áp dụng GA để tìm kiếm "không gian đặc trưng chính" (principal feature space) có độ nhạy tối ưu với "bộ phân loại". Đề xuất được đánh giá bằng ba thử nghiệm khác nhau với 12, 20 và 27 đặc trưng có độ chính xác cao nhất 99% với 12 tập đặc trưng con. Một nghiên cứu khác được thực hiện ở "Sindhu và cộng sự (2012)" sử dụng GA làm kỹ thuật "lựa chọn đặc trưng". Họ bắt đầu quá trình tối ưu hóa bằng cách tạo ra quần thể ngẫu nhiên và thông qua việc tính toán mức độ phù hợp của "nhiễm sắc thể", họ tạo ra quần thể cho thế hệ tiếp theo. Hàm fitness sẽ xem xét số lượng đặc trưng, độ nhạy và độ đặc hiệu để đánh giá tập đặc trưng con. Họ thực hiện so sánh với các kỹ thuật "lựa chọn đặc trưng" khác và đạt độ chính xác 98,38% với 16 đặc trưng được trích xuất từ 41 đặc trưng gốc của tập dữ liệu KDD.

Nghiên cứu của Kuang và cộng sự (2014) trình bày một mô hình SVM mới với Phân tích thành phần chính kenerl (KPCA) và GA. Phương pháp của họ được triển khai để phát hiện lưu lượng mạng bình thường và độc hại. Họ thử nghiệm phương pháp của

mình trên tập dữ liệu KDD cup. GA được sử dụng trong nghiên cứu của họ để tối ưu hóa các tham số cho SVM và KPCA nhằm "giảm chiều" của không gian đặc trưng. Bằng cách chọn 12 đặc trưng tối ưu, chúng đạt "độ chính xác phát hiện" là 94,22% và sự "hội tụ" nhanh cũng xảy ra với khả năng khái quát hóa tốt hơn. Nghiên cứu của Aslahi- Shahri và cộng sự (2016) đánh giá hiệu suất của "bộ phân loại" SVM trên tập dữ liệu phát hiện xâm nhập. Theo các tác giả, SVM không đạt được độ chính xác cao hơn khi làm việc độc lập. Điều này là do SVM cần một tập dữ liệu có mẫu phù hợp và chọn các đặc trưng tối ưu với độ dư thừa tối thiểu. Do đó, các tác giả triển khai GA để tìm kiếm các đặc trưng tối ưu cho SVM và sau đó thực hiện phân loại trên tập dữ liệu phát hiện xâm nhập. Một "lựa chọn đặc trưng" tổng hợp sử dụng thuật toán di truyền "song mục tiêu" (bi-objective) được đề xuất trong Das và cộng sự (2017). Các tác giả giải quyết vấn đề lựa chọn các đặc trưng tối ưu cho "khai thác dữ liệu". Họ hợp nhất hai khái niệm để cùng nhau phát triển thuật toán di truyền "song mục tiêu" là phân tích vùng biên của "lý thuyết tập thô" (rough set theory) và thông tin "đa biến" lẫn nhau. Phương pháp của họ được thử nghiệm trên các bộ dữ liệu nổi tiếng và được đánh giá về hiệu suất. Trong số các bộ dữ liệu khác, họ đã sử dụng bộ dữ liệu spambase để phân loại thư rác và email hợp pháp và đạt được độ chính xác tốt nhất là 92,6%.

Công trình của Gharaee và Hosseinvand (2016) đề xuất một IDS sử dụng GA để "lựa chọn đặc trưng" với hàm fitness cải tiến. Chúng đạt được độ chính xác dự đoán cao trong khi vẫn duy trì tỷ lệ dương tính giả thấp. Đề xuất của họ đã được thử nghiệm trên bộ dữ liệu KDD cup và UNSW-NB 15. Họ báo cáo độ chính xác của từng lớp trong bài báo của họ. Họ cũng tạo ra một tập dữ liệu riêng cho từng lớp và sau đó áp dụng kỹ thuật của họ. Công trình của Xu và cộng sự (2018) trình bày phương pháp "lựa chọn đặc trưng" dựa trên Thuật toán tối ưu hóa cá voi nhị phân (WOA) cải tiến để phát hiện xâm nhập mạng. Họ thử nghiệm kỹ thuật này trên tập dữ liệu KDD và báo cáo kết quả. Trong công việc của họ, WOA hội tụ chậm và có thể rơi vào trạng thái tối ưu cục bộ trong quá trình cập nhật cơ chế có ảnh hưởng xấu đến việc phân loại. Họ so sánh cơ chế của họ với GA và báo cáo kỹ thuật của họ có độ chính xác cao hơn. Trên tập dữ liệu KDD cup, kỹ thuật của họ chọn ra 5 trên 41 đặc trưng và đạt độ chính xác 97,89%. Trong khi đó, GA chọn lọc 11/41 đặc trưng với độ chính xác 95,58%. Công trình của Yousefi- Azar và cộng sự (2017) đề xuất sử dụng bộ mã hóa tự động làm mô hình tổng quát cho mục đích học đặc trưng. Họ giải thích cách bộ mã hóa tự động có khả năng học cách biểu diễn tiềm ẩn và sự giống nhau về ngữ nghĩa giữa các đặc trưng của tập dữ liệu. Kỹ thuật của họ đã được thử nghiệm để phát hiện xâm nhập cũng như phân loại phần mềm độc hại. Với mục đích này, họ sử dụng tập dữ liệu KDD cup và tập dữ liệu Microsoft Malware Classification Challenge (BIG 2015). Để phát hiện xâm nhập, họ báo cáo kết quả tốt nhất với độ chính xác 83,3% với "trình phân loại" Gaussian naïve Bayes.

Công trình của Tahir và cộng sự (2021) cũng trình bày phương pháp "lựa chọn đặc trưng" dựa trên GA bằng cách kết hợp các "bản đồ hỗn loạn" (chaotic map). Giải pháp của họ được thử nghiệm trên dữ liệu từ "điện toán tình cảm" (affective computing) (Halim và cộng sự, 2021) và hệ thống chăm sóc sức khỏe. Trong công trình của họ, các "bản đồ hỗn loạn" được áp dụng cho quần thể ban đầu của GA, sau đó là các hoạt động tái tạo để tạo ra tập hợp các đặc trưng tối ưu. Phương pháp của họ được đánh giá

dựa trên bài toán "nhận dạng cảm xúc bảy lớp" (seven class emotion identification). Công trình ở "Viharos và cộng sự (2021)" giải quyết yêu cầu tích hợp kỹ thuật nhận dạng thuộc tính phù hợp nhất cho một vấn đề nhất định để đạt được thứ tự đặc trưng tối ưu. Đề xuất của họ về cơ bản là sự kết hợp của nhiều phương pháp "lựa chọn đặc trưng" để có được giải pháp tổng quát. Các thử nghiệm trong công trình của họ được thực hiện bằng cách sử dụng dữ liệu kho lưu trữ UCI và một số bộ dữ liệu thực tế. Công trình của Nouri-Moghaddam và cộng sự (2021) trình bày phương pháp "lựa chọn đặc trưng" wrapper dựa trên cơ chế "tối ưu hóa rừng đa mục tiêu" (multi-objective forest optimization). Mặt trận Pareto trong giải pháp của họ được duy trì thông qua các cơ chế lựa chọn dựa trên kho lưu trữ, lưới và khu vực. Thuật toán của họ được đặt tên là phương pháp multi-objective wrapper dựa trên Forest Optimization (MOFOA). Ngoài các thử nghiệm trên dữ liệu kho lưu trữ UCI, họ còn thực hiện đánh giá trên hai bộ dữ liệu microarray (Uzma và cộng sự, 2021). Một phương pháp "lựa chọn đặc trưng" để nhận dạng xâm nhập mạng được trình bày trong Li và cộng sự (2021). Giải pháp của họ dựa trên thuật toán Krill Herd (KH), một kỹ thuật trí tuệ bầy đàn. Tối ưu hóa bước lasso lân cận tuyến tính gần nhất được thực hiện trong giải pháp của họ để cập nhật vị trí đàn nhuyễn thể trong không gian tìm kiếm. Điều này cho phép rút ra giải pháp tối ưu toàn cục. Tương tự, đề xuất của Dwivedi và cộng sự (2021) là một đóng góp dựa trên trí thông minh bầy đàn cho IDS. Thuật toán châu chấu từ lĩnh vực trí tuệ bầy đàn được sử dụng trong công việc của họ, được tích hợp với phương pháp "lựa chọn đặc trưng" tổng thể.

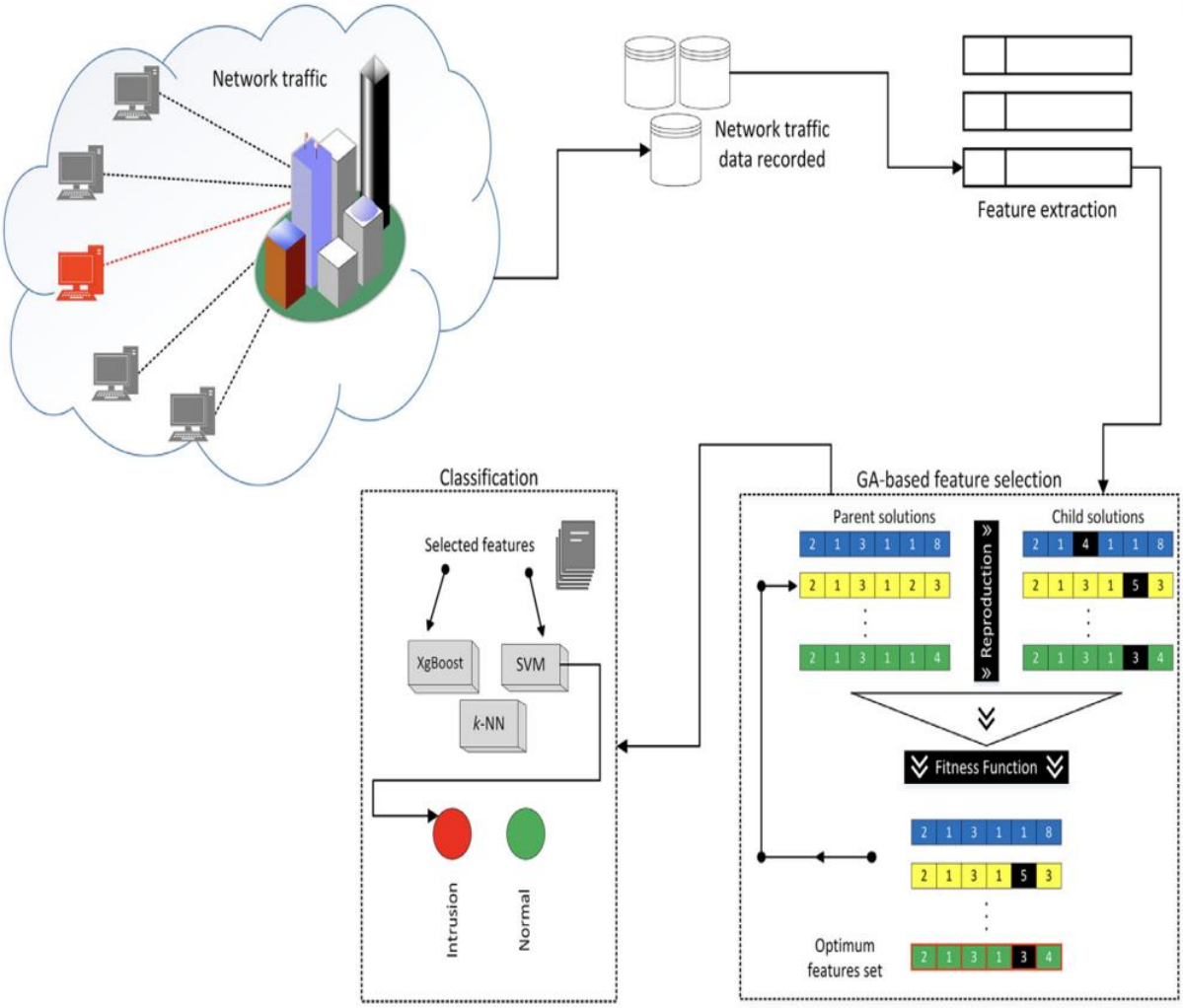
Phương pháp "lựa chọn đặc trưng" hai lớp được trình bày trong Amini và Hu (2021). Lớp đầu tiên trong giải pháp của họ là trình wrapper dựa trên GA, trong khi lớp thứ hai là cơ chế nhúng Elastic Net. Mục đích chính của giải pháp của họ là nâng cao độ chính xác của dự đoán. Lớp thứ hai giải quyết các vấn đề tối ưu do GA ở lớp đầu tiên gây ra. Hiệu suất của giải pháp của họ được đánh giá bằng cách sử dụng tập dữ liệu di truyền Maize từ quần thể NAM. Giải pháp này được đặt tên là nhúng trình wrapper hai lớp (GA-EN). Các kết quả thu được cho thấy rằng GA-EB dẫn đến sai số bình phương trung bình gốc nhỏ hơn với các kích thước không gian đặc trưng khác nhau khi so sánh với phương pháp wrapper được nhúng. Cơ chế "lựa chọn đặc trưng" nâng cao dựa trên GA được trình bày trong Maleki và cộng sự (2021). Công trình của họ sử dụng k-NN để tiên lượng bệnh ung thư phổi được tích hợp với "lựa chọn đặc trưng" dựa trên "điện toán tiến hóa". Thuật toán di truyền trong công trình của họ được áp dụng như một phương pháp lai để lựa chọn thuộc tính hiệu quả. Đề xuất của họ được đánh giá dựa trên bộ dữ liệu về ung thư phổi mang lại hiệu quả tốt hơn. Công trình của Guo và cộng sự (2021) trình bày một phương pháp "lựa chọn đặc trưng" mới sử dụng mạng bộ nhớ ngắn hạn dài (LSTM network). Đề xuất này được thử nghiệm để dự đoán "độ nhám" (roughness) bề mặt như một nghiên cứu điển hình. Công trình của họ trích xuất nhiều đặc trưng trong miền thời gian và tần số từ các tín hiệu gốc và tín hiệu bị phân tách. Kết quả thu được trong công việc của họ cho thấy các phương pháp học sâu thực hiện tốt hơn đối với nhiệm vụ đã chọn. Công trình của Sumaiya Thaseen và cộng sự (2021) trình bày một hệ thống phát hiện xâm nhập tích hợp. Giải pháp của họ sử dụng lựa chọn thuộc tính dựa trên mối tương quan được tích hợp với "mạng thần kinh nhân tạo" (artificial neural network). Giải pháp của họ là một khung dựa trên máy học (Halim và cộng sự, 2020) cho IDS. Việc "lựa chọn đặc trưng" dựa trên mối tương quan cho phép

xếp hạng các thuộc tính theo giá trị tương quan cao nhất giữa các thuộc tính và sự thật cơ bản. Kỹ thuật "lựa chọn đặc trưng" để phát hiện phần mềm độc hại từ Android được trình bày trong Mahindru và Sangal (2021). Giải pháp của họ dựa trên phương pháp học máy. Mô-đun nghiêng máy trong công trình của họ dựa trên Least Square Support Vector Machine (LSSVM) được đánh giá thông qua ba kernel, đó là hàm tuyến tính (linear), hàm radial basis và đa thức (polynomial). Để đánh giá hiệu suất, hai triệu ứng dụng Android riêng biệt đã được sử dụng.

3. Phân tích thiết kế hệ thống

3.1. Tổng quan hệ thống

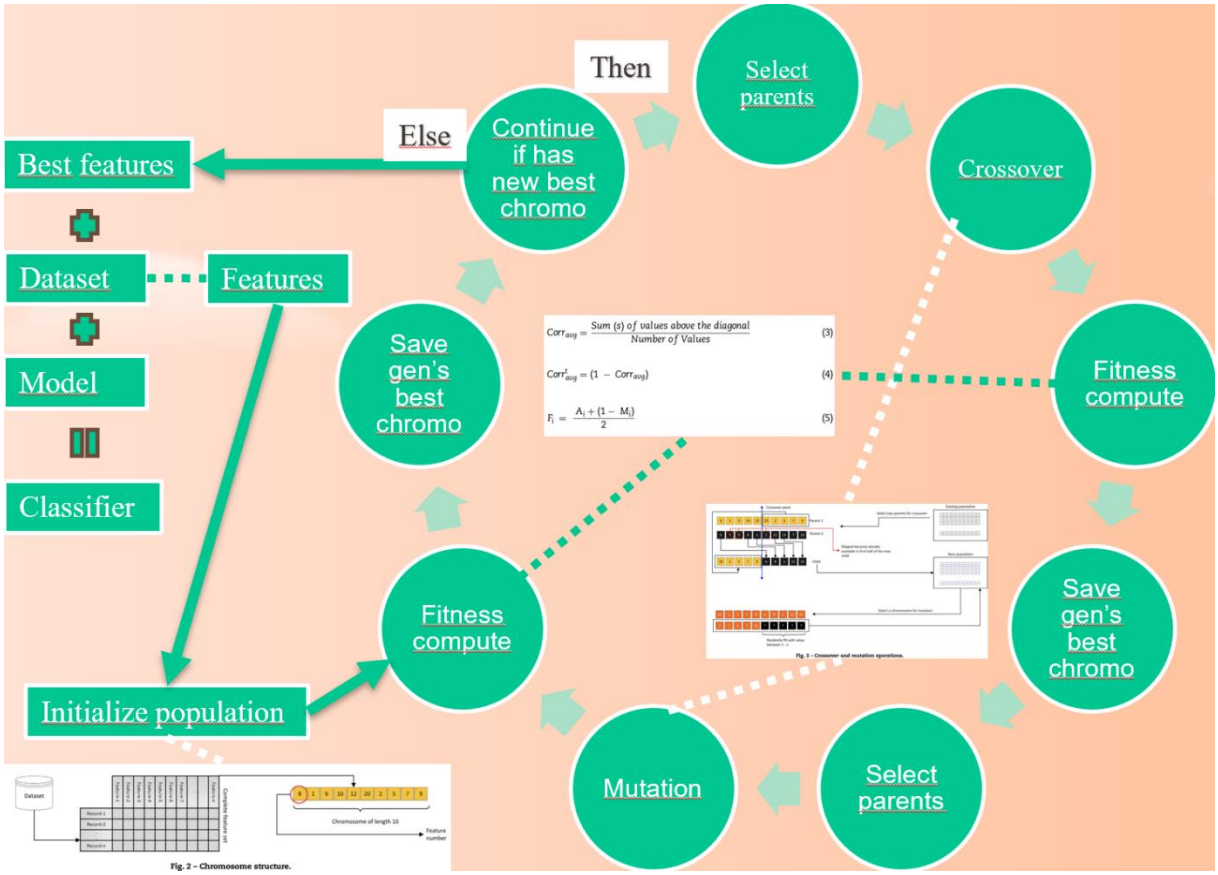
Thiết kế hệ thống của bài báo: Lấy dữ liệu mạng > trích xuất đặc trưng > GbFS > Classification (phân loại). Đồ án của nhóm tập trung vào giai đoạn train Classifier (bộ phân loại) và GbFS.



Hình 1: Thiết kế hệ thống của bài báo

Thiết kế hệ thống của nhóm:

- Chuẩn bị dữ liệu: Dataset sẽ được xử lý giá trị rỗng bằng cách thay chúng bằng 0 (nếu có), điều chỉnh lượng dữ liệu giống như bài báo, mã hóa nhãn bằng LabelEncoder, chuẩn hóa và scaling bằng MinMaxScaler và train – test set được chia bằng Kfold(shuffle=True).
- Đánh giá bộ phân loại không dùng GbFS: dùng k-fold cross validation để đánh giá các bộ phân loại.
- Đánh giá bộ phân loại có dùng GbFS: Đưa dataset vào GbFS để tìm ra best_chromo, giảm chiều bằng cách giữ lại các đặc trưng trong best_chromo và nhãn. Cuối cùng dùng dataset đã giảm chiều để đánh giá bộ phân loại bằng k-fold cross validation.
- Các kết quả cần lưu lại: best_chromo nếu dùng GbFS; báo cáo phân loại; độ chính xác; độ thu hồi.



Hình 2: Minh họa thiết kế hệ thống của đồ án (có GbFS)

3.2. Cấu trúc của thuật toán di truyền

Khởi tạo: Bước này tạo ra một quần thể ban đầu gồm các NST. Các NST có thể được tạo ngẫu nhiên hoặc theo một qui luật cụ thể với độ dài cho trước. Quần thể này đại diện cho không gian tìm kiếm ban đầu và bao gồm một số lượng NST .

Đánh giá (Evaluation): Mỗi NST trong quần thể được đánh giá dựa trên một hàm mục tiêu (hàm fitness) để xác định độ tốt của giải pháp mà nó đại diện. Hàm mục tiêu có thể được thiết kế để đo lường hiệu suất, chất lượng hoặc sự thích nghi của NST với môi trường tìm kiếm.

Lựa chọn (Selection): Bước lựa chọn chọn lọc các NST tốt nhất từ quần thể hiện tại để truyền dần sang thế hệ tiếp theo. Mục tiêu của lựa chọn là ưu tiên các NST có giá trị fitness cao để đảm bảo sự tiến hóa của quần thể. Có nhiều phương pháp lựa chọn khác nhau như Roulette Wheel Selection, Tournament Selection, hoặc Rank Selection. Mỗi phương pháp có cách tiếp cận khác nhau để chọn lọc NST dựa trên giá trị fitness của chúng.

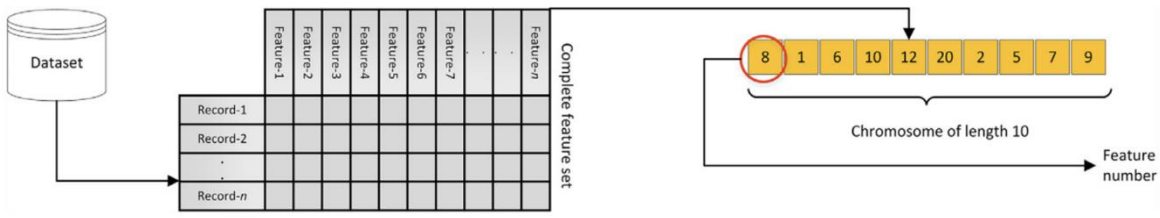
Lai ghép (Crossover): kết hợp các NST được chọn lựa chọn từ thế hệ hiện tại để tạo ra các NST con mới. Quá trình lai ghép thường dựa trên việc trao đổi và kết hợp các gen giữa các NST cha mẹ. Có nhiều phương pháp lai ghép như Single-Point Crossover, Two-Point Crossover, hoặc Uniform Crossover. Mỗi phương pháp có cách tiếp cận khác nhau để kết hợp các gen và tạo ra NST con mới.

Đột biến (Mutation): thay đổi (<tỷ lệ dột biến> * độ dài NST) gen trong các NST con một cách ngẫu nhiên để tạo ra sự đa dạng và khám phá không gian tìm kiếm rộng hơn. Điều này giúp thuật toán tránh rơi vào một cấu trúc giải pháp cục bộ.

Điều kiện dừng (Termination Criteria): Thuật toán di truyền tiếp tục lặp lại các bước trên cho đến khi một điều kiện dừng được đáp ứng. Điều kiện dừng có thể là một số lượng thế hệ tối đa đã đạt được, sự hội tụ của giải pháp, hoặc đạt được một mức độ tối ưu mong muốn. Khi Điều kiện dừng được đáp ứng, thuật toán di truyền kết thúc và trả về giải pháp tốt nhất.

3.2.1. Khởi tạo quần thể

Đây là bước đầu tiên trong việc triển khai thuật toán di truyền (GA). Tại đây quần thể ban đầu được tạo ngẫu nhiên bao gồm nhiều NST. Các NST được tạo ra bằng cách kết hợp các gen duy nhất được chọn ngẫu nhiên. Các gen ở đây là các đặc trưng (thuộc tính) của dữ liệu. Mỗi NST trong một quần thể đại diện cho một giải pháp cho bài toán chọn lọc. Hình bên dưới cho thấy cấu trúc của một NST.



Hình 3: Minh họa "khởi tạo quần thể ban đầu"

NST là 1 mảng một chiều với chiều dài N cho trước. Trong nghiên cứu này giá trị của N được đặt là 10. Mỗi ô của nhiễm sắc thể chứa một giá trị số V, với điều kiện 0 ≤ V ≤ Chiều dài của tập đặc trưng gốc. Ví dụ ở hình trên gen tại chỉ số 0 có giá trị là 8, điều này có nghĩa nó đại diện cho đặc trưng thứ 8 từ tập dữ liệu gốc. Bằng cách này, mỗi nhiễm sắc thể sẽ chứa 10 đặc trưng từ bộ đặc trưng để tạo ra tập con mà sau này sẽ được tối ưu hóa để trở thành giải pháp tối ưu.

3.2.2. Tính giá trị thích nghi

Hàm độ thích nghi (fitness function) dựa trên việc tính toán mối tương quan giữa các đặc trưng được chọn mà không có sự hiện diện của "nhãn lớp", bài báo trình bày một hàm độ thích nghi mới. Mục tiêu là chọn các đặc trưng có độ tương quan thấp để đảm bảo tính đa dạng cao và đại diện cho phần lớn thông tin của tập dữ liệu. Sau khi tính toán tương quan của các đặc trưng đã chọn, hàm thích nghi được đề xuất tính giá trị trung bình tương quan của các đặc trưng đó. Sau đó, tính giá trị trung bình tương quan được biến đổi hay giá trị trung bình phi tương quan bằng cách trừ nó khỏi 1 (giá trị tối đa).

$$\mathbf{Corr_{avg}} = \frac{\boldsymbol{Sum\ of\ values\ above\ the\ diagonal}}{\boldsymbol{Number\ of\ Values}} \quad (\mathbf{1})$$

$$\boldsymbol{Corr^{t}_{avg}} = \mathbf{1} - \boldsymbol{Corr_{avg}} \quad (\mathbf{2})$$

$$\boldsymbol{F_i} = \frac{\boldsymbol{A_i} + (\mathbf{1} - \boldsymbol{M_i})}{\mathbf{2}} \quad (\mathbf{3})$$

Giá trị trung bình của mối tương quan ($Corr_{avg}$) được tính bằng cách lấy tổng của tất cả các giá trị tương quan phía trên đường chéo chính của ma trận tương quan, chia cho số lượng giá trị (Phương trình 1). Giá trị trung bình phi tương quan được tính trong phương trình 2, mục tiêu là có được giá trị không tương quan cao nhất (tức là giá trị $Corr_{avg}$ thấp nhất). Trong phương trình 3, F_i là giá trị thích nghi của nhiễm sắc thể thứ i. A_i, M_i lần lượt là độ chính xác (có từ việc train 1 model cho trước, ở đây là LogisticRegression), giá trị tuyệt đối của trung bình tương quan của các đặc trưng trong NST thứ i.

3.2.3. Lựa chọn "cha mẹ" (Parents Selection)

Chiến thuật lựa chọn bằng bánh xe roulette, mỗi NST trong quần thể được gán một phần trên bánh xe roulette dựa trên giá trị thích nghi của nó. Giá trị thích nghi càng cao thì phần trên bánh xe roulette mà NST chiếm giữ càng lớn. Và do đó, cơ hội để NST đó được chọn làm "cha mẹ" càng cao.

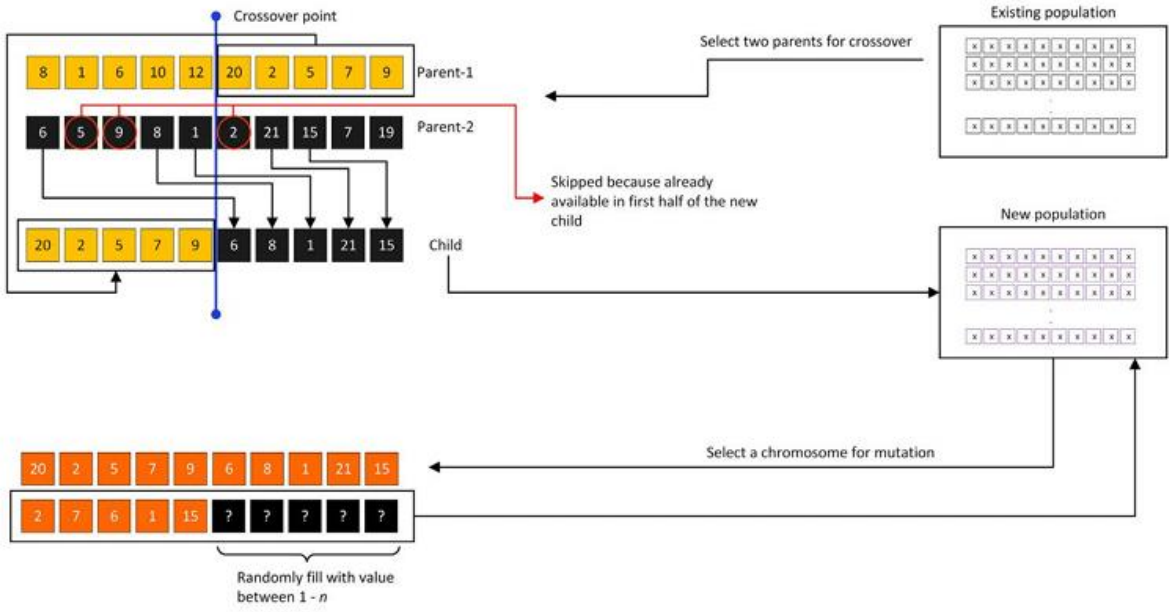
Hình 4: Minh họa "bánh xe roulette"

Cụ thể, quá trình lựa chọn bằng bánh xe roulette được thực hiện như sau:

- Tính giá trị thích nghi: Đầu tiên, giá trị thích nghi của từng NST trong quần thể được tính toán thông qua một hàm thích nghi. Giá trị thích nghi đo lường mức độ tương thích của NST với mục tiêu hoặc yêu cầu của bài toán.
- Xác định phần trên bánh xe roulette: Mỗi NST được gán một phần trên bánh xe roulette dựa trên giá trị thích nghi đã được chuẩn hóa của nó. Các NST có giá trị thích nghi cao sẽ có phần trên bánh xe roulette rộng hơn, tương ứng với khả năng cao hơn để được chọn.
- Lựa chọn "cha mẹ": Cuối cùng, quá trình quay bánh xe roulette được thực hiện để chọn ngẫu nhiên các NST làm "cha mẹ" cho quá trình tiếp tục tiến hóa. Các NST có phần trên bánh xe roulette lớn hơn sẽ có xác suất cao hơn để được chọn.

Ưu điểm của chiến lược lựa chọn bằng bánh xe roulette là khi được triển khai song song, thời gian thực hiện sẽ giảm và không yêu cầu chia tỷ lệ/sắp xếp như các phương pháp lựa chọn khác. Ngoài ra, sự lựa chọn của "cha mẹ" tỷ lệ thuận với nhau với các giá trị thích hợp được tạo ra thông qua hàm thích nghi, đồng nghĩa với việc cơ hội lựa chọn cao hơn đối với các NST có giá trị thích nghi cao.

3.2.4. Lai chéo (Crossover)



Hình 5: Minh họa "đột biến" và "lai chéo"

Ở đây tỷ lệ chéo được chọn là 0.5, có nghĩa là "cắt" NST 1 theo tỷ lệ 50% và "nối" với 50% còn lại lấy từ đầu của NST 2. Quá trình chéo được thực hiện theo các bước sau:

- Chia nửa sau: Nửa sau của NST "cha mẹ" đầu tiên được chuyển trực tiếp cho NST con, không thay đổi.
- Sao chép các gen còn lại: Các gen còn lại từ NST "cha mẹ" thứ hai được sao chép vào NST con theo trình tự tương tự như trong trường hợp của NST "cha mẹ" thứ hai.
- Lặp lại cho đủ số lượng NST : Quá trình trên được lặp lại cho đến khi đủ số lượng NST cần thiết trong quần thể.
- Sự duy nhất của NST con: Chỉ có một NST con được sao chép từ cặp "cha mẹ" đã được chọn. Điều này đảm bảo rằng mỗi NST trong quần thể mới được tạo ra từ cặp "cha mẹ" khác nhau, tạo ra sự đa dạng trong quần thể.

Trên cơ sở các bước trên, quá trình lai chéo được thực hiện để tạo ra các NST mới và tăng cường khả năng tiến hóa của thuật toán di truyền.

3.2.5. Đột biến (Mutation)

Đột biến trong thuật toán di truyền được thiết kế như sau:

- Lựa chọn tỷ lệ đột biến: Ba tỷ lệ đột biến khác nhau (0.25, 0.5 và 0.75) được kiểm tra để tìm ra tỷ lệ tối ưu. Dựa trên mô phỏng của bài báo, tỷ lệ đột biến 0.5 được chọn.

- Đột biến: Giữ ((1 - <tỷ lệ đột biến>) * độ dài NST) gen của từng NST trong quần thể không đổi, còn lại sẽ được biến đổi ngẫu nhiên thành đặc trưng không trùng lặp với các gen đang có trong NST.

Quá trình đột biến giúp cung cấp khả năng khám phá trong thuật toán di truyền, tạo ra sự đa dạng trong không gian tìm kiếm và ảnh hưởng đến tốc độ hội tụ của thuật toán GA. Tuy nhiên, nếu tỷ lệ đột biến quá cao, có thể dẫn đến sự mất mát của các phần có giá trị của nghiệm trước khi hội tụ.

3.3. Ứng dụng của thuật toán di truyền trong học máy và khai phá dữ liệu

Ứng dụng phổ biến của thuật toán di truyền:

- Tối ưu hóa: GA được sử dụng để tìm kiếm các giải pháp tối ưu cho các bài toán tối ưu hóa. Ví dụ, trong tối ưu hóa tham số mô hình học máy, GA có thể được sử dụng để tìm ra các giá trị tối ưu cho các tham số của mô hình.
- Học máy: GA có thể được áp dụng để tìm kiếm và tối ưu các mô hình học máy. Qua việc tạo ra và tiến hóa các tập hợp các NST (mô hình), GA có thể tìm ra mô hình tốt nhất cho một bài toán học máy cụ thể. Điều này đặc biệt hữu ích khi không có một mô hình học máy cụ thể nào đã được biết đến trước đó.
- Clustering: GA có thể được sử dụng để tìm ra cấu trúc phân cụm trong dữ liệu. Các NST trong GA có thể biểu diễn các trung tâm cụm và thuật toán di truyền có thể tìm ra các trung tâm cụm tối ưu dựa trên việc tối thiểu hóa hàm mất mát.
- Feature selection: GA có thể được sử dụng để chọn ra các đặc trưng (features) quan trọng nhất trong dữ liệu. Qua việc tiến hóa các tập hợp các NST đại diện cho các tập con đặc trưng, GA có thể tìm ra các tập con đặc trưng tối ưu giúp cải thiện hiệu suất của mô hình học máy.
- Xử lý tối ưu các vấn đề kết hợp: GA có thể được áp dụng để giải quyết các vấn đề kết hợp phức tạp như lập lịch, tối ưu hóa địa lý, tối ưu hóa mạng lưới và quy hoạch nguyên.
- Khai phá dữ liệu: GA có thể được sử dụng để khai phá dữ liệu bằng cách tìm kiếm các quy tắc, mô hình hoặc cấu trúc dữ liệu tiềm năng trong tập dữ liệu. Điều này có thể áp dụng trong việc phân tích dữ liệu, phát hiện biểu đồ, tìm kiếm khối lượng dữ liệu lớn, và nhiều ứng dụng khác.

4. Hiện thực hệ thống

4.1. Môi trường thực nghiệm

Môi trường thực nghiệm Google Colab (Google Colaboratory) là một nền tảng phát triển mã nguồn mở dựa trên trình duyệt web, cung cấp một môi trường tính toán mạnh mẽ và thuận tiện. Colab cho phép người dùng chạy và triển khai mã Python một cách dễ dàng và miễn phí.

Ngôn ngữ lập trình sử dụng là Python: Python là một ngôn ngữ lập trình phổ biến và mạnh mẽ được sử dụng rộng rãi trong phân tích dữ liệu.

Các thư viện quan trọng được sử dụng: NumPy, Pandas để xử lý dữ liệu; scikit-learn để tiền xử lý dữ liệu, huấn luyện mô hình và các phương thức đánh giá mô hình; scikit-learn-intelex là 1 thư viện đặc biệt nhằm tối ưu việc huấn luyện mô hình, dự đoán khi sử dụng Intel CPU (kết nối local runtime).

4.2. Datasets

Tập dữ liệu CIRA-CIC-DoHBrw-2020 được tạo bởi Viện Nghiên cứu An ninh Mạng Canada (CIC), tập trung vào lưu lượng mạng được tạo ra từ việc sử dụng các triển khai DoH (DNS qua HTTPS) khác nhau. Tập dữ liệu này được thiết kế để giúp các nhà nghiên cứu phân tích và phát hiện lưu lượng DoH độc hại, có thể được sử dụng để vượt qua các cơ chế bảo mật DNS truyền thống. Nó bao gồm lưu lượng từ các kịch bản khác nhau như duyệt web bình thường, duyệt web dựa trên DoH và các hoạt động độc hại như trích xuất dữ liệu qua DoH. Đặc điểm chính:

- Nguồn: https://www.unb.ca/cic/datasets/dohbrw-2020.html
- Cung cấp một bộ dữ liệu lưu lượng mạng toàn diện.
- Bao gồm cả lưu lượng DoH lành tính và độc hại.
- Hữu ích cho việc phát triển và kiểm tra các hệ thống phát hiện xâm nhập (IDS) và các biện pháp an ninh mạng khác.
- Thông qua xử lý bộ dữ liệu (xử lý rỗng) nhóm biết được: Số đặc trưng: 35; số đặc trưng sau khi bỏ nhãn: 34; số mẫu: 1436779.

```
# Record per class
print('Record per class:\n',data.groupby('Label').size())
print('\nSum:\t\t',data['Label'].size)
```

```
Record per class:
 Label
Benign        19807
DoH          269643
Malicious    249836
NonDoH       897493
dtype: int64

Sum:             1436779
```

Hình 6: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (1)

```
# Number of features
print('Number of features: ',data.shape[1])
```

```
Number of features:  35
```

Hình 7: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (2)

```
print('Number of train features: ',X.shape[1])
```

```
Number of train features:  34
```

Hình 8: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của nhóm (3)

Table 4 – Attack classes in CIRA-CIC-DoHBrw-2020.		
Classification of attack	No. of records	Attack name
DoH	269643	DNS over HTTPs
None-DoH	897493	None DNS over HTTPs
Benign-DoH	19807	Benign DNS over HTTPs
Malicious	249836	Malicious

Hình 9: Thống kê về dataset CIRA-CIC-DoHBrw-2020 của bài báo

Table 2 – Datasets summary.			
	CIRA-CIC-DoHBrw-2020	Bot-IoT	UNSW NB-15
No. of features	34	29	49
No. of classes	4	5	10
No. of Samples	~1.4 million	~3 million	~0.25 million
X			

Hình 10: Thống kê chung về các dataset của bài báo

Tập dữ liệu Bot-IoT được tạo ra để giúp các nhà nghiên cứu phát triển và đánh giá các giải pháp bảo mật cho môi trường IoT. Nó chứa một loạt các lưu lượng mạng bao gồm cả lưu lượng bình thường và lưu lượng tấn công. Tập dữ liệu này được tạo ra bằng một môi trường thử nghiệm thực tế, nơi các loại tấn công khác nhau như DoS, DDoS, trộm cắp dữ liệu và do thám được mô phỏng trên các thiết bị IoT. Ở đây nhóm dùng Bot-IoT 5%. Đặc điểm chính:

- Nguồn: https://www.kaggle.com/datasets/vigneshvenkateswaran/bot-iot-5-data
- Tập dữ liệu quy mô lớn với lưu lượng mạng IoT thực tế.
- Bao gồm nhiều loại tấn công để hỗ trợ phân tích toàn diện.
- Hữu ích cho việc phát triển các mô hình học máy để phát hiện và giảm thiểu các mối đe dọa liên quan đến IoT.
- Thông qua xử lý bộ dữ liệu (điều chỉnh lượng dữ liệu như trong bài báo) nhóm biết được: Số đặc trưng: 32; số đặc trưng sau khi bỏ nhãn: 29; số mẫu: 665591.

```
# Record per class
print('Record per class:\n',data.groupby('category').size()
print('\nSum:\t\t',data['category'].size)
```

```
Record per class:
 category
DDoS            240000
DoS             242788
Normal             477
Reconnaissance  182166
Theft              160
dtype: int64

Sum:            665591
```

Hình 11: Thống kê về dataset Bot-IoT 5% của nhóm (1)

```
# Number of features
print('Number of features: ',data.shape[1])

Number of features:  32
```

Hình 12: Thống kê về dataset Bot-IoT 5% của nhóm (2)

```
print('Number of train features: ',X.shape[1])

Number of train features:  29
```

Hình 13: Thống kê về dataset Bot-IoT 5% của nhóm (3)

Table 6 – Attack classes in Bot-IoT dataset.

Classification of attack	No. of records	Attack name
DDoS	240,000	DDos
DoS	242,788	DoS
Reconnaissance	182,166	OS and Service Scan
Theft	160	Keylogging and Data Exfiltration

Hình 14: Thống kê về dataset Bot-IoT 5% của bài báo

Tập dữ liệu UNSW-NB15 được tạo ra bởi Trung tâm An ninh Mạng Úc (ACCS) cho nghiên cứu an ninh mạng. Nó chứa dữ liệu lưu lượng mạng được thu thập bằng công cụ IXIA PerfectStorm, mô phỏng lưu lượng mạng và các kịch bản tấn công thực tế. Tập dữ liệu này bao gồm nhiều loại tấn công như DoS, sâu máy tính, backdoor và khai thác. Đặc điểm chính:

- Nguồn: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15/code
- Tập dữ liệu toàn diện với sự kết hợp của lưu lượng bình thường và lưu lượng độc hại.
- Được tạo ra trong một môi trường kiểm soát để đảm bảo độ chính xác của các mô phỏng tấn công.
- Được sử dụng rộng rãi để đánh giá các hệ thống phát hiện và ngăn chặn xâm nhập.
- Thông qua xử lý bộ dữ liệu (xử lý rỗng, điều chỉnh lượng dữ liệu) nhóm biết được: Số đặc trưng: 49; số đặc trưng sau khi bỏ nhãn: 47; số mẫu: 62994.

```
# Record per class
print('Record per class:\n',data.groupby('attack_cat').size())
print('\nSum:\t\t',data['attack_cat'].size)

Record per class:
 attack_cat
Analysis            677
Backdoor            577
DoS                4089
Exploits           7061
Fuzzers           12062
Generic            5016
Normal            31395
Reconnaissance     1695
Shellcode           378
Worms                44
dtype: int64

Sum:            62994
```

Hình 15: Thống kê về dataset UNSW-NB15 của nhóm (1)

```
# Number of features
print('Number of features: ',data.shape[1])

Number of features:  49
```

Hình 16: Thống kê về dataset UNSW-NB15 của nhóm (2)

```
print('Number of train features: ',X.shape[1])

Number of train features:  47
```

Hình 17: Thống kê về dataset UNSW-NB15 của nhóm (3)

Table 8 – Per class records in UNSW NB-15 dataset.

Classification of attack	No of records
Analysis	677
Backdoor	577
DoS	4089
Exploits	7061
Fuzzers	12,062
Generic	5016
Normal	31,395
Reconnaissance	1695
Shellcode	378
Worms	44

Hình 18: Thống kê về dataset UNSW-NB15 của bài báo

Link drive chứa các dataset nhóm thu thập được: https://drive.google.com/drive/folders/18uNi86CZRJ7QSBSob4F6Xipy4SAWBVZA?usp=sharing

4.3. Các bộ phân loại

Trong đồ án này, nhóm sẽ dùng 3 bộ phân loại được sử dụng trong bài báo để thực nghiệm: SVM, k-NN, XgBoost.

- Support Vector Machines (SVM): SVM là mô hình học có giám sát được sử dụng để phân loại. SVM thường được áp dụng cho dữ liệu có thể phân tách tuyến tính, tuy nhiên, chúng cũng có thể được sử dụng để phân loại phi tuyến tính theo tính toán nhiều chiều đặc trưng. Nó tạo ra một tập hợp các "siêu phẳng" (hyperplanes, mặt phẳng quyết định) trong không gian nhiều chiều để phân loại dữ liệu. Ưu điểm của SVM là tính hiệu quả của nó trong không gian nhiều chiều, tính linh hoạt của nó (các chức năng kernel khác nhau có thể được sử dụng và cả các kernel tùy chỉnh), cũng như hiệu quả bộ nhớ. Tuy nhiên, nếu số lượng đặc trưng lớn hơn số lượng mẫu thì có thể xảy ra hiện tượng overfitting.
- k-Nearest Neighbor (k-NN): k-NN là phương pháp được sử dụng cho cả phân loại và hồi quy. Nó phân loại dữ liệu trên cơ sở "đa số phiếu bầu" của k-neighbors. Đối với trường hợp đơn giản, nếu như k = 1, có một loại dữ liệu. Giá trị tối ưu của k có thể được quyết định bằng cách kiểm tra dữ liệu hoặc thông qua một loạt các thí nghiệm. Giá trị k lớn hơn sẽ tốt hơn vì nó làm giảm nhiễu. "Phiếu bầu" được quyết định trên cơ sở khoảng cách giữa hai điểm. Hàm

"khoảng cách" có thể là Euclidean, Manhattan, Minkowski hoặc bất kỳ hàm nào khác. Việc đo khoảng cách thường được quyết định dựa trên loại dữ liệu.

- XgBoost (eXtreme Gradient Boosting): Đây là một thuật toán phức tạp có khả năng xử lý các loại "hiện diện" bất thường trong một tập dữ liệu. Thuật toán phân loại XgBoost cung cấp regularization, xử lý song song, tính linh hoạt hơn về tiêu chí đánh giá do người dùng tự định nghĩa, thuật toán tối ưu và xử lý các giá trị thiếu. XGBoost cung cấp một cơ sở "tăng cường cây song song" (parallel tree boosting). Điều này cho phép giải quyết nhiều vấn đề về máy học một cách nhanh chóng với độ chính xác tốt hơn.

4.4. Quy trình triển khai

Không dùng GbFS:

- Bước 1: Tải dataset từ drive.
- Bước 2: Tiền xử lý dữ liệu: Nạp dữ liệu, xử lý rỗng, xử lý trùng lặp, mã hóa nhãn, chuẩn hóa và chia train – test set.
- Bước 3: Đánh giá mô hình: dùng k-fold cross validation để đánh giá mô hình phân loại với dataset đầy đủ đặc trưng.
- Bước 4: Ghi nhận kết quả: xuất ra báo cáo phân loại, độ chính xác, độ thu hồi và đồ thị thể hiện độ chính xác của mô hình với các lớp.

Dùng GbFS:

- Bước 1: Tải dataset.
- Bước 2: Tiền xử lý dữ liệu.
- Bước 3: Sử dụng GbFS do nhóm thiết kế để chọn 10 đặc trưng tối ưu của dataset. Dữ liệu được đưa vào mô hình chỉ có 10 đặc trưng tối ưu (10 cột) và nhãn của chúng.
- Bước 4: Đánh giá mô hình.
- Bước 5: Ghi nhận kết quả.

5. Thực nghiệm và đánh giá

Về tiêu chí đánh giá của nhóm: Độ chính xác (accuracy) và độ thu hồi (recall).

Về sản phẩm: 18 mô hình.

	Full feature			**Apply GbFS**		
CIRA-CIC-DOHBrw-2020	SVM	k-NN	XgBoost	SVM	k-NN	XgBoost
UNSW-NB15	SVM	k-NN	XgBoost	SVM	k-NN	XgBoost

Bot-IoT 5%	SVM	k-NN	XgBoost	SVM	k-NN	XgBoost

Bảng 1: Các mô hình sẽ huấn luyện

Link drive chứa các sản phẩm:
https://drive.google.com/drive/folders/1_bBfYT1JkVhPQiCniwSsk1ZNIO2PXrJC?usp=drive_link

5.1. Thực nghiệm

5.1.1. CIRA-CIC-DOHBrw-2020

Kết quả đánh giá 6 mô hình từ bài báo:

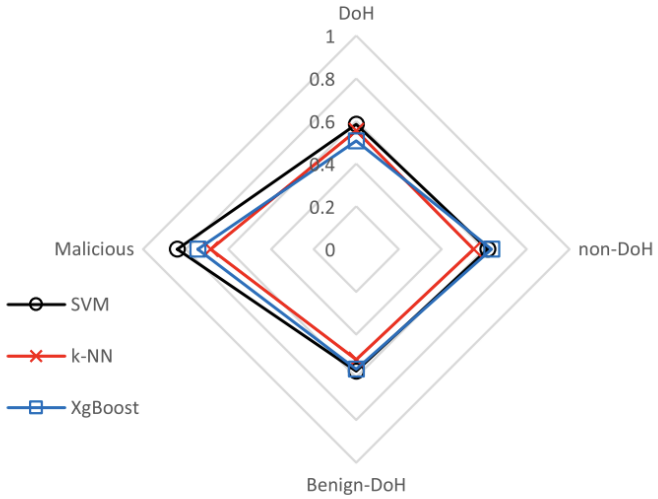
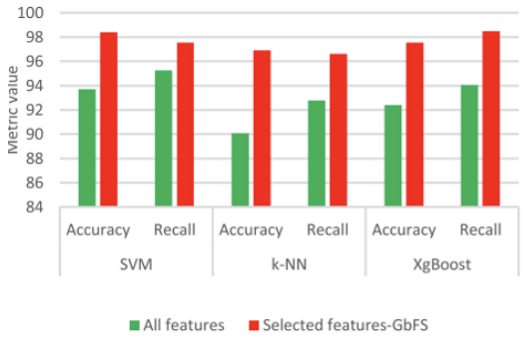


Fig. 4 – Results obtained using CIRA-CIC-DoHBrw-2020 dataset.

Hình 19: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset CIRA-CIC-DOHBrw-2020



Hình 20: Kết quả đánh giá của 6 mô hình của bài báo với dataset CIRA-CIC-DOHBrw-2020

10 đặc trưng tối ưu của dataset từ GbFS (nt204_Doh_GA.ipynb):
PacketTimeStandardDeviation, FlowReceivedRate, PacketLengthSkewFromMode, SourcePort, SourceIP, TimeStamp, ResponseTimeTimeStandardDeviation, ResponseTimeTimeMedian, ResponseTimeTimeSkewFromMedian, PacketLengthMedian.

Hình 21: Kết quả chạy GbFS với dataset CIRA-CIC-DOHBrw-2020

Kết quả đánh giá 2 mô hình SVM của nhóm: [nt204_svm_doh.ipynb]

- Không dùng GbFS: Độ chính xác đạt ~80%, độ thu hồi đạt ~51%.

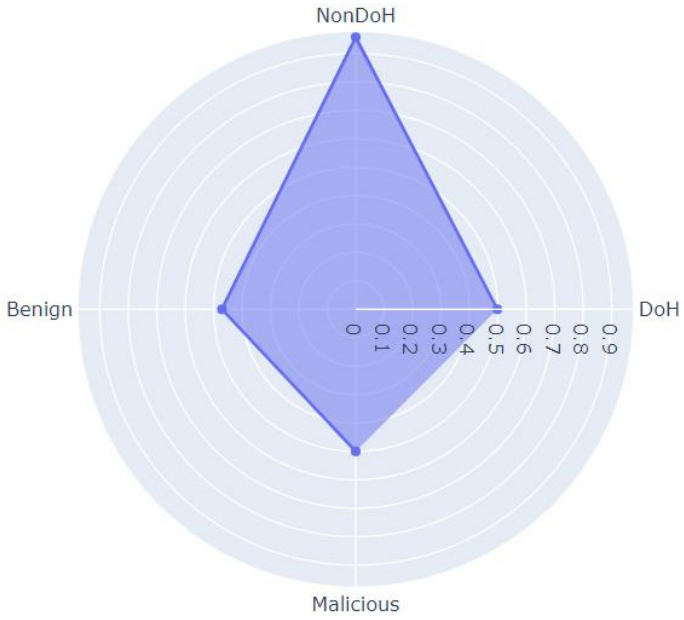
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+-----------+-------------+-----------+------------+-----------+
| Class     |   Precision |    Recall |   F1-Score |   Support |
+===========+=============+===========+============+===========+
| Benign    |      0.4703 | 0.0727521 |   0.126011 |     19807 |
+-----------+-------------+-----------+------------+-----------+
| DoH       |      0.49789 | 0.384534 |   0.433931 |    269643 |
+-----------+-------------+-----------+------------+-----------+
| Malicious |      0.499693 | 0.577067 |   0.5356 |    249836 |
+-----------+-------------+-----------+------------+-----------+
| NonDoH    |      0.956804 | 0.998859 |   0.977379 |    897493 |
+-----------+-------------+-----------+------------+-----------+

Accuracy: 0.7974566721813167    Recall: 0.5083030057123881
```

Hình 22: Kết quả đánh giá mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS



Hình 23: Độ chính xác của mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~79%, độ thu hồi ~49%.

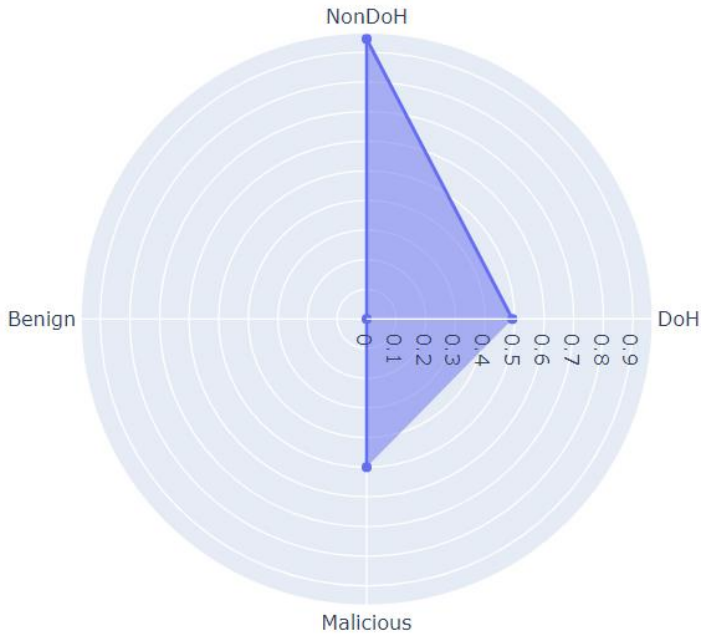
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+-----------+-------------+----------+------------+-----------+
| Class     |   Precision |   Recall |   F1-Score |   Support |
+===========+=============+==========+============+===========+
| Benign    |           0 |        0 |          0 |     19807 |
+-----------+-------------+----------+------------+-----------+
| DoH       |     0.49231 | 0.246912 |   0.328879 |    269643 |
+-----------+-------------+----------+------------+-----------+
| Malicious |    0.499915 |  0.70806 |   0.586055 |    249836 |
+-----------+-------------+----------+------------+-----------+
| NonDoH    |    0.944906 | 0.997746 |   0.970607 |    897493 |
+-----------+-------------+----------+------------+-----------+

Accuracy: 0.7927085515587297    Recall: 0.4881795220919163
```

Hình 24: Kết quả đánh giá mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS



Hình 25: Độ chính xác của mô hình SVM với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS

Kết quả đánh giá 2 mô hình k-NN của nhóm: [nt204_knn_Doh.ipynb]

- Không dùng GbFS: Độ chính xác đạt ~73%, độ thu hồi đạt ~44%.

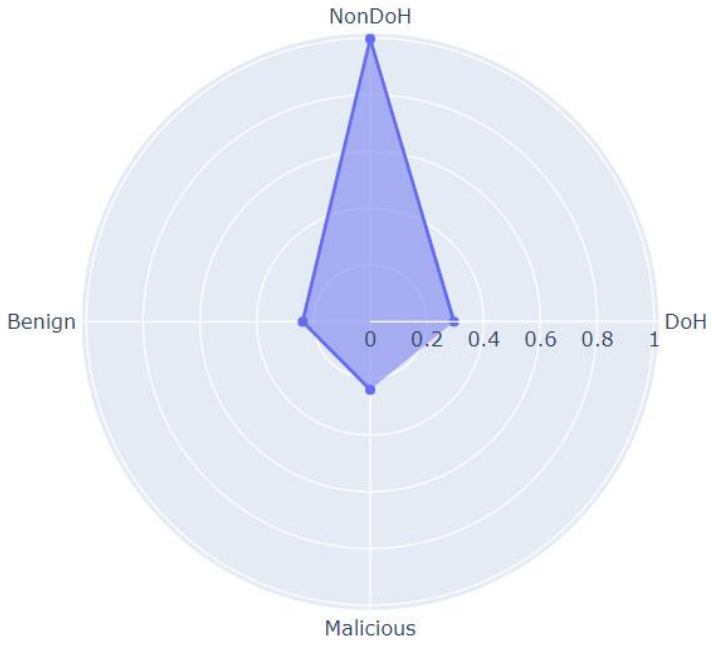
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+-----------+-------------+----------+------------+-----------+
| Class     |   Precision |   Recall |   F1-Score |   Support |
+===========+=============+==========+============+===========+
| Benign    |     0.23786 | 0.216641 |   0.226755 |     19807 |
+-----------+-------------+----------+------------+-----------+
| DoH       |    0.295223 | 0.329421 |   0.311386 |    269643 |
+-----------+-------------+----------+------------+-----------+
| Malicious |    0.240027 | 0.210882 |   0.224513 |    249836 |
+-----------+-------------+----------+------------+-----------+
| NonDoH    |    0.996894 | 0.997858 |   0.997376 |    897493 |
+-----------+-------------+----------+------------+-----------+

Accuracy: 0.7247976202324783    Recall: 0.43870053804266185
```

Hình 26: Kết quả đánh giá mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS



Hình 27: Độ chính xác của mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~72%, độ thu hồi đạt ~42%.

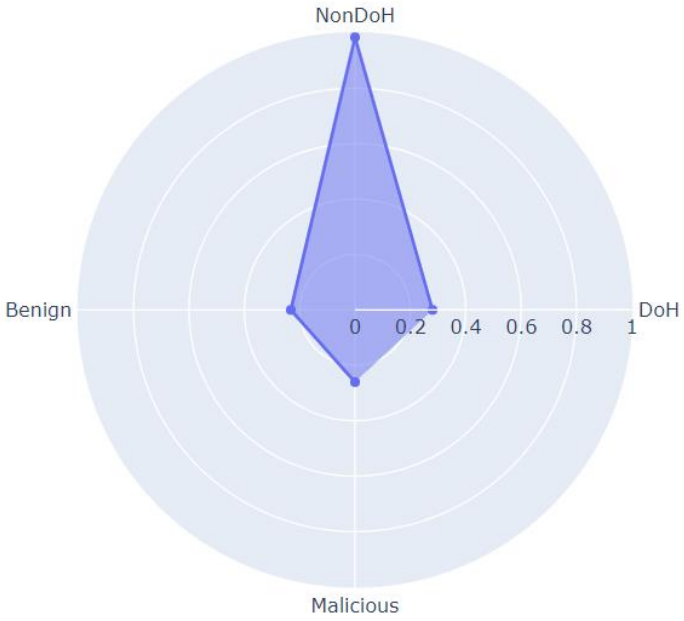
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+-----------+-------------+----------+------------+-----------+
| Class     |   Precision |   Recall |   F1-Score |   Support |
+===========+=============+==========+============+===========+
| Benign    |    0.231798 | 0.155753 |   0.186315 |     19807 |
+-----------+-------------+----------+------------+-----------+
| DoH       |    0.280049 | 0.285299 |   0.28265  |    269643 |
+-----------+-------------+----------+------------+-----------+
| Malicious |    0.260275 | 0.248471 |   0.254236 |    249836 |
+-----------+-------------+----------+------------+-----------+
| NonDoH    |    0.983856 | 0.99786  |   0.990809 |    897493 |
+-----------+-------------+----------+------------+-----------+

Accuracy: 0.7222147595420033    Recall: 0.4218457646037898
```

Hình 28: Kết quả đánh giá mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS



Hình 29: Độ chính xác của mô hình k-NN với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS

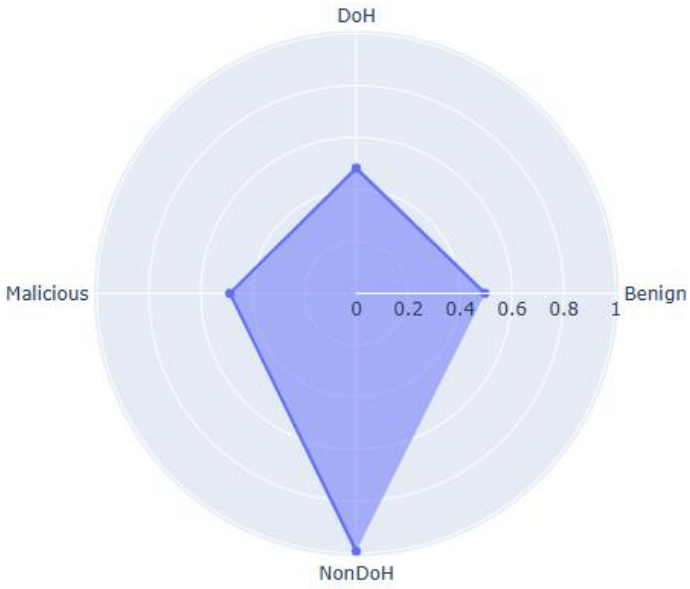
Kết quả đánh giá 2 mô hình XgBoost của nhóm: [nt204_xgboost_doh.ipynb - Colab (google.com)]

- Không dùng GbFS: Độ chính xác đạt ~80%, độ thu hồi đạt ~65%.

Class	Precision	Recall	F1-Score	Support
0	0.496807	0.628515	0.554954	19807
1	0.482747	0.38831	0.430409	269643
2	0.487673	0.567913	0.524743	249836
3	0.992685	0.999754	0.996207	897493

Accuracy: 0.8047939175057542 Recall: 0.6461228021525567

Hình 30: Kết quả đánh giá mô hình xgboost với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS



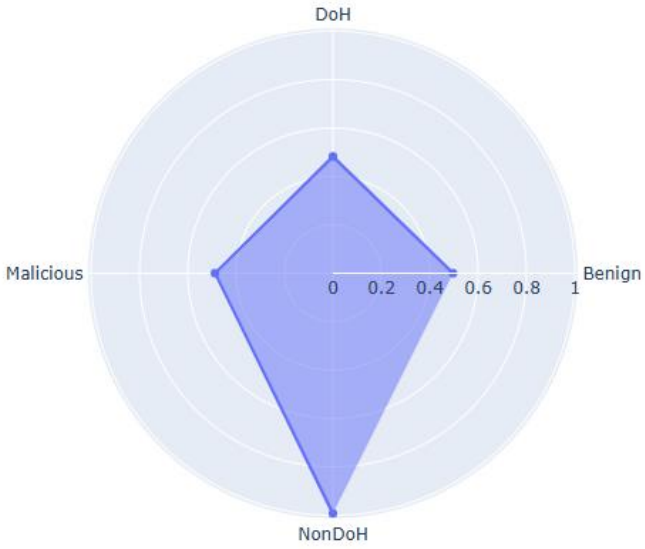
Hình 31: Độ chính xác của mô hình xgboost với dataset CIRA-CIC-DOHBrw-2020 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~79%, độ thu hồi đạt ~49%.

Class	Precision	Recall	F1-Score	Support
0	0.48826	0.0356945	0.0665255	19807
1	0.485726	0.47198	0.478754	269643
2	0.484736	0.47291	0.47875	249836
3	0.965467	0.999982	0.982422	897493

Accuracy: 0.7959470454398345 Recall: 0.49514161765600007

Hình 32: Kết quả đánh giá mô hình xgboost với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS



Hình 33: Độ chính xác của mô hình xgboost với dataset CIRA-CIC-DOHBrw-2020 có dùng GbFS

5.1.2. UNSW-NB15

Kết quả đánh giá 6 mô hình từ bài báo:

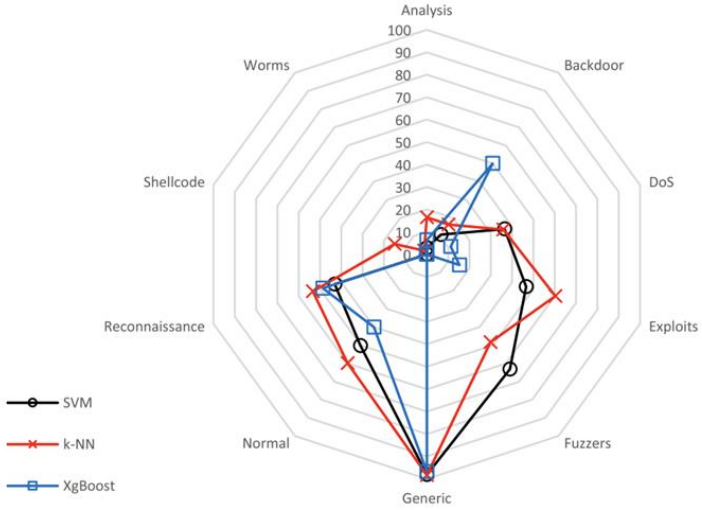
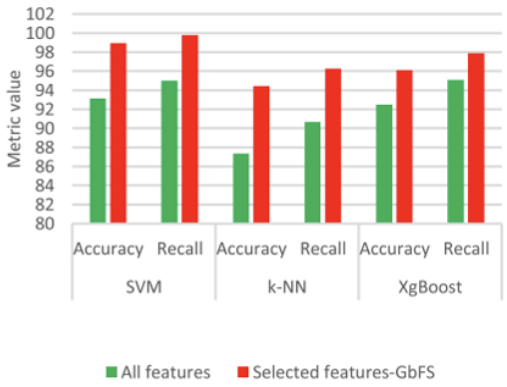


Fig. 6 – Results obtained using UNSW NB-15 dataset.

Hình 30: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset UNSW-NB15



Hình 31: Kết quả đánh giá của 6 mô hình của bài báo với dataset UNSW-NB15

10 đặc trưng tối ưu của dataset từ GbFS (nt204_UNSW_GA.ipynb): Dintpkt, dmeansz, dttl, ct_dst_ltm, srcip, ct_state_ttl, synack, dbytes, dur, ct_dst_sport_ltm.

```
best_chromo, max_fitness = generations(X=X, y=y, pop_size=100, chromo_len=10,
                                       crossover_rate = 0.5, mutation_rate = 0.5)
print("Best feature: ", list(X.columns[best_chromo]))
print("Max fitness: ", max_fitness)
```

```
Max fitness of generation 0 :  0.7737636077004644
Max fitness of generation 1 :  0.7742008979204442
Max fitness of generation 2 :  0.7741007003088096
Best feature:  ['Dintpkt', 'dmeansz', 'dttl', 'ct_dst_ltm', 'srcip', 'ct_state_ttl', 'synack', 'dbytes', 'dur', 'ct_dst_sport_ltm']
Max fitness:  0.7742008979204442
```

Hình 32: Kết quả chạy GbFS với dataset UNSW-NB15

Kết quả đánh giá 2 mô hình SVM của nhóm: nt204_svm_UNSW.ipynb

- Không dùng GbFS: Độ chính xác đạt ~83%, độ thu hồi đạt ~39%.

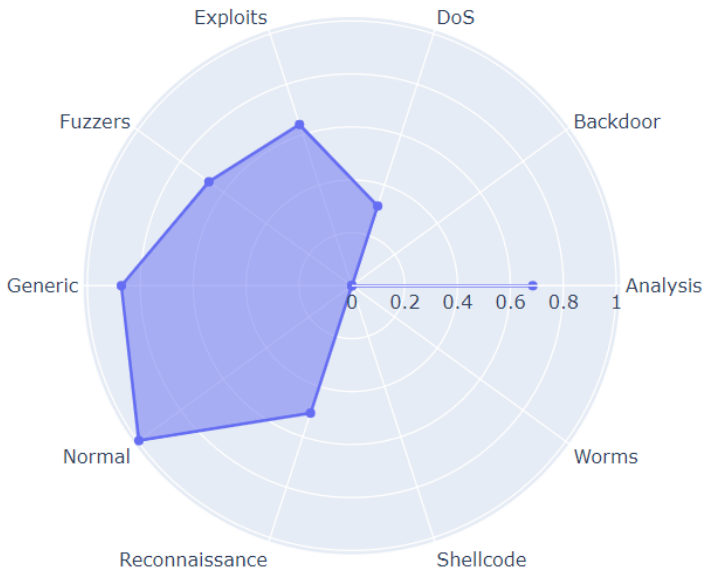
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+----------------+-------------+-----------+------------+-----------+
| Class          |   Precision |    Recall |   F1-Score |   Support |
+================+=============+===========+============+===========+
| Analysis       |    0.684211 | 0.0192024 |  0.0373563 |       677 |
+----------------+-------------+-----------+------------+-----------+
| Backdoor       |    0        | 0         |  0         |       577 |
+----------------+-------------+-----------+------------+-----------+
| DoS            |    0.316445 | 0.0931768 |  0.143964  |      4089 |
+----------------+-------------+-----------+------------+-----------+
| Exploits       |    0.639985 | 0.743946  |  0.688061  |      7061 |
+----------------+-------------+-----------+------------+-----------+
| Fuzzers        |    0.667816 | 0.931189  |  0.777812  |     12062 |
+----------------+-------------+-----------+------------+-----------+
| Generic        |    0.871259 | 0.789274  |  0.828243  |      5016 |
+----------------+-------------+-----------+------------+-----------+
| Normal         |    0.995299 | 0.98452   |  0.98988   |     31395 |
+----------------+-------------+-----------+------------+-----------+
| Reconnaissance |    0.505677 | 0.341593  |  0.407746  |      1695 |
+----------------+-------------+-----------+------------+-----------+
| Shellcode      |    0        | 0         |  0         |       378 |
+----------------+-------------+-----------+------------+-----------+
| Worms          |    0        | 0         |  0         |        44 |
+----------------+-------------+-----------+------------+-----------+

Accuracy: 0.8306505381464901    Recall: 0.39029007240737135
```

Hình 33: Kết quả đánh giá mô hình SVM với dataset UNSW-NB15 không dùng GbFS



Hình 34: Độ chính xác của mô hình SVM với dataset UNSW-NB15 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~78%, độ thu hồi ~31%.

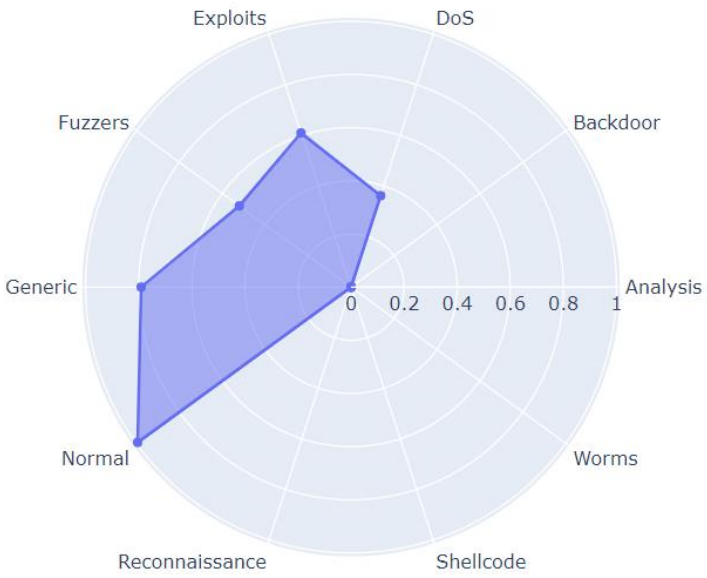
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+----------------+-------------+------------+------------+-----------+
| Class          |   Precision |     Recall |   F1-Score |   Support |
+================+=============+============+============+===========+
| Analysis       |    0        | 0          | 0          |       677 |
+----------------+-------------+------------+------------+-----------+
| Backdoor       |    0        | 0          | 0          |       577 |
+----------------+-------------+------------+------------+-----------+
| DoS            |    0.361702 | 0.00831499 | 0.0162563  |      4089 |
+----------------+-------------+------------+------------+-----------+
| Exploits       |    0.609782 | 0.494406   | 0.546066   |      7061 |
+----------------+-------------+------------+------------+-----------+
| Fuzzers        |    0.519881 | 0.954983   | 0.673252   |     12062 |
+----------------+-------------+------------+------------+-----------+
| Generic        |    0.790483 | 0.649123   | 0.712863   |      5016 |
+----------------+-------------+------------+------------+-----------+
| Normal         |    0.993624 | 0.977926   | 0.985713   |     31395 |
+----------------+-------------+------------+------------+-----------+
| Reconnaissance |    0        | 0          | 0          |      1695 |
+----------------+-------------+------------+------------+-----------+
| Shellcode      |    0        | 0          | 0          |       378 |
+----------------+-------------+------------+------------+-----------+
| Worms          |    0        | 0          | 0          |        44 |
+----------------+-------------+------------+------------+-----------+

Accuracy: 0.7778836079626631    Recall: 0.30847527013313736
```

Hình 35: Kết quả đánh giá mô hình SVM với dataset UNSW-NB15 có dùng GbFS



Hình 36: Độ chính xác của mô hình SVM với dataset UNSW-NB15 có dùng GbFS

Kết quả đánh giá 2 mô hình k-NN của nhóm: [nt204_knn_UNSW.ipynb]

- Không dùng GbFS: Độ chính xác đạt ~84%, độ thu hồi đạt ~46%.

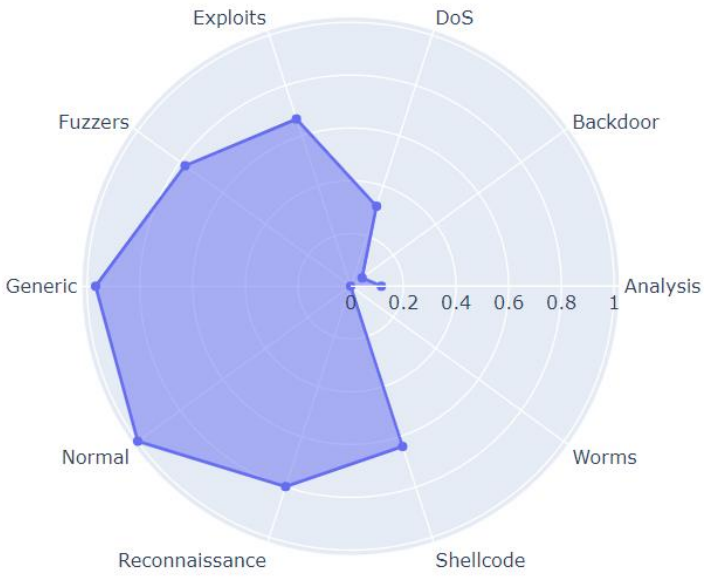
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+----------------+-------------+-----------+------------+-----------+
| Class          |   Precision |    Recall |   F1-Score |   Support |
+================+=============+===========+============+===========+
| Analysis       |    0.116719 |  0.109306 |   0.112891 |       677 |
+----------------+-------------+-----------+------------+-----------+
| Backdoor       |   0.0538117 | 0.0415945 |  0.0469208 |       577 |
+----------------+-------------+-----------+------------+-----------+
| DoS            |    0.318689 |  0.328198 |   0.323373 |      4089 |
+----------------+-------------+-----------+------------+-----------+
| Exploits       |    0.666097 |   0.71732 |    0.69076 |      7061 |
+----------------+-------------+-----------+------------+-----------+
| Fuzzers        |    0.776846 |  0.891229 |   0.830116 |     12062 |
+----------------+-------------+-----------+------------+-----------+
| Generic        |       0.968 |  0.796053 |   0.873646 |      5016 |
+----------------+-------------+-----------+------------+-----------+
| Normal         |    0.999127 |  0.984488 |   0.991754 |     31395 |
+----------------+-------------+-----------+------------+-----------+
| Reconnaissance |    0.798319 |  0.504425 |   0.618221 |      1695 |
+----------------+-------------+-----------+------------+-----------+
| Shellcode      |    0.638462 |  0.219577 |   0.326772 |       378 |
+----------------+-------------+-----------+------------+-----------+
| Worms          |           0 |         0 |          0 |        44 |
+----------------+-------------+-----------+------------+-----------+

Accuracy: 0.8428421754452805    Recall: 0.4592189068629737
```

Hình 37: Kết quả đánh giá mô hình k-NN với dataset UNSW-NB15 không dùng GbFS



Hình 38: Độ chính xác của mô hình k-NN với dataset UNSW-NB15 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~78%, độ thu hồi đạt ~38%.

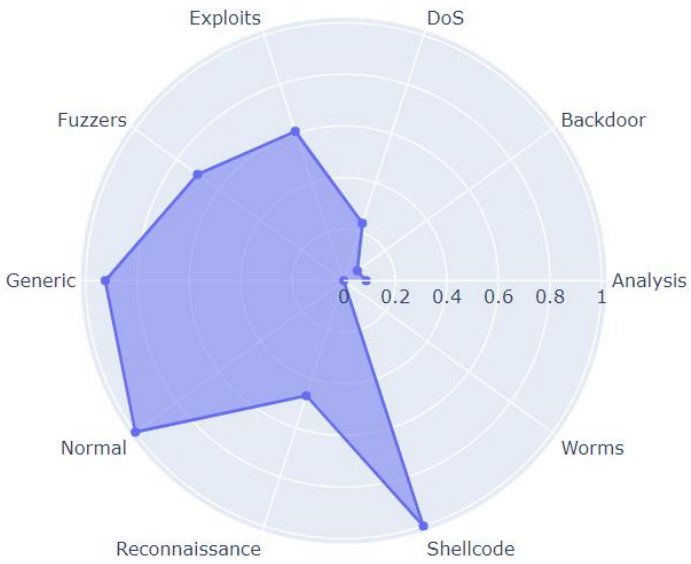
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+----------------+-------------+-----------+------------+-----------+
| Class          |   Precision |    Recall |   F1-Score |   Support |
+================+=============+===========+============+===========+
| Analysis       |   0.0866667 | 0.249631  | 0.128664   |       677 |
+----------------+-------------+-----------+------------+-----------+
| Backdoor       |   0.0651387 | 0.0935875 | 0.0768137  |       577 |
+----------------+-------------+-----------+------------+-----------+
| DoS            |   0.233838  | 0.291025  | 0.259316   |      4089 |
+----------------+-------------+-----------+------------+-----------+
| Exploits       |   0.608493  | 0.61082   | 0.609654   |      7061 |
+----------------+-------------+-----------+------------+-----------+
| Fuzzers        |   0.70073   | 0.763472  | 0.730757   |     12062 |
+----------------+-------------+-----------+------------+-----------+
| Generic        |   0.924636  | 0.645734  | 0.760418   |      5016 |
+----------------+-------------+-----------+------------+-----------+
| Normal         |   0.998772  | 0.984233  | 0.991449   |     31395 |
+----------------+-------------+-----------+------------+-----------+
| Reconnaissance |   0.469163  | 0.125664  | 0.198232   |      1695 |
+----------------+-------------+-----------+------------+-----------+
| Shellcode      |   1         | 0.0026455 | 0.00527704 |       378 |
+----------------+-------------+-----------+------------+-----------+
| Worms          |   0         | 0         | 0          |        44 |
+----------------+-------------+-----------+------------+-----------+

Accuracy: 0.7824237228942439    Recall: 0.3766811033964993
```

Hình 39: Kết quả đánh giá mô hình k-NN với dataset UNSW-NB15 có dùng GbFS



Hình 40: Độ chính xác của mô hình k-NN với dataset UNSW-NB15 có dùng GbFS

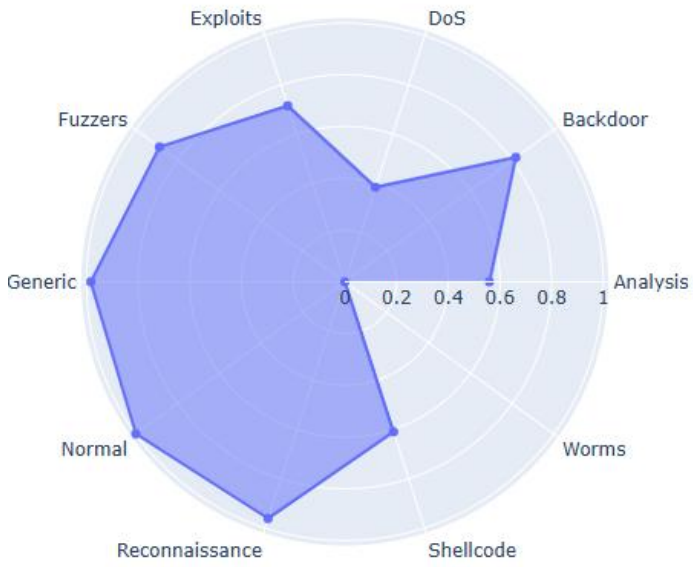
Kết quả đánh giá 2 mô hình XgBoost của nhóm: [nt204_Xgboost_UNSW.ipynb - Colab (google.com)]

- Không dùng GbFS: Độ chính xác đạt ~88%, độ thu hồi đạt ~54%.

Class	Precision	Recall	F1-Score	Support
0	0.559441	0.118168	0.195122	677
1	0.818182	0.0779896	0.142405	577
2	0.384225	0.537295	0.448047	4089
3	0.715152	0.818156	0.763194	7061
4	0.886575	0.907229	0.896783	12062
5	0.98273	0.816786	0.892107	5016
6	0.999774	0.984711	0.992185	31395
7	0.961265	0.717404	0.821622	1695
8	0.609428	0.478836	0.536296	378
9	0	0	0	44

Accuracy: 0.88025843731149 Recall: 0.5456575891042328

Hình 41: Kết quả đánh giá mô hình Xgboost với dataset UNSW-NB15 không dùng GbFS



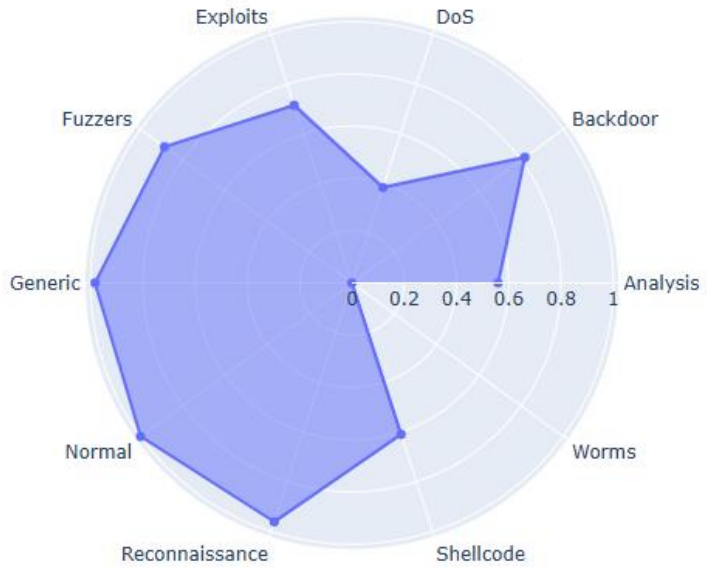
Hình 42: Độ chính xác của mô hình Xboost với dataset UNSW-NB15 không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~78%, độ thu hồi ~31%.

Class	Precision	Recall	F1-Score	Support
0	0.538462	0.0413589	0.0768176	677
1	0.357143	0.017331	0.0330579	577
2	0.505872	0.136953	0.21555	4089
3	0.670846	0.659255	0.665	7061
4	0.617266	0.951998	0.748932	12062
5	0.737107	0.729466	0.733267	5016
6	0.998901	0.984743	0.991772	31395
7	0.710602	0.146313	0.242661	1695
8	0	0	0	378
9	0	0	0	44

Accuracy: 0.8184747753754326 Recall: 0.3667417020046015

Hình 43: Kết quả đánh giá mô hình Xgboost với dataset UNSW-NB15 có dùng GbFS



Hình 44: Độ chính xác của mô hình Xboost với dataset UNSW-NB15 có dùng GbFS

5.1.3. Bot-IoT 5%

Kết quả đánh giá 6 mô hình từ bài báo:

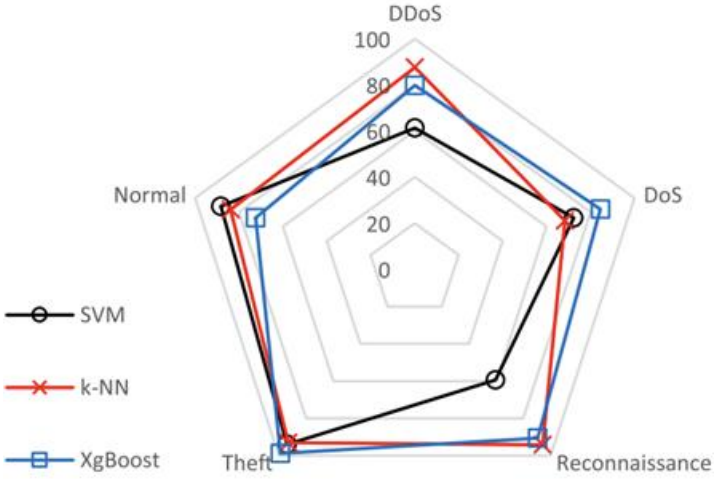
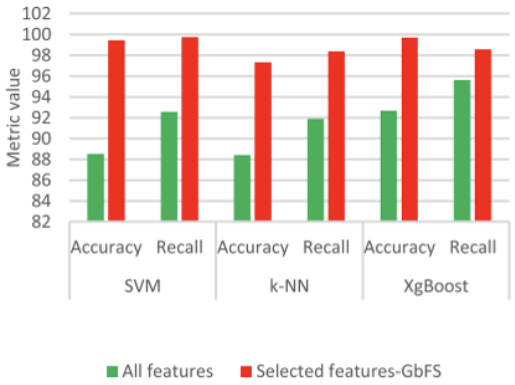


Fig. 5 – Results obtained using Bot-IoT dataset.

Hình 41: Độ chính xác của 3 mô hình không dùng GbFS của bài báo với dataset Bot-IoT 5%



Hình 42: Kết quả đánh giá của 6 mô hình của bài báo với dataset Bot-IoT 5%

10 đặc trưng tối ưu của dataset từ GbFS (nt204_BOT_IOT_005_GA.ipynb): stddev, AR_P_Proto_P_SrcIP, srate, drate, min, state, mean, Pkts_P_State_P_Protocol_P_SrcIP, N_IN_Conn_P_DstIP, flgs.

```
best_chromo, max_fitness = generations(X=X, y=y, pop_size=100, chromo_len=10,
                                       crossover_rate = 0.5, mutation_rate = 0.5)
print("Best feature: ", list(X.columns[best_chromo]))
print("Max fitness: ", max_fitness)
```

```
Max fitness of generation 0 :  0.9027655329615974
Max fitness of generation 1 :  0.904130017402377
Max fitness of generation 2 :  0.9065791076620792
Max fitness of generation 3 :  0.9012291284084405
Best feature:  ['stddev', 'AR_P_Proto_P_SrcIP', 'srate', 'drate', 'min', 'state', 'mean', 'Pkts_P_State_P_Protocol_P_SrcIP', 'N_IN_Conn_P_DstIP', 'flgs']
Max fitness:  0.9065791076620792
```

Hình 43: Kết quả chạy GbFS với dataset Bot-IoT 5%

Kết quả đánh giá 2 mô hình SVM của nhóm: nt204_svm_BOT_IOT_005.ipynb

- Không dùng GbFS: Độ chính xác đạt ~90%, độ thu hồi đạt ~69%.

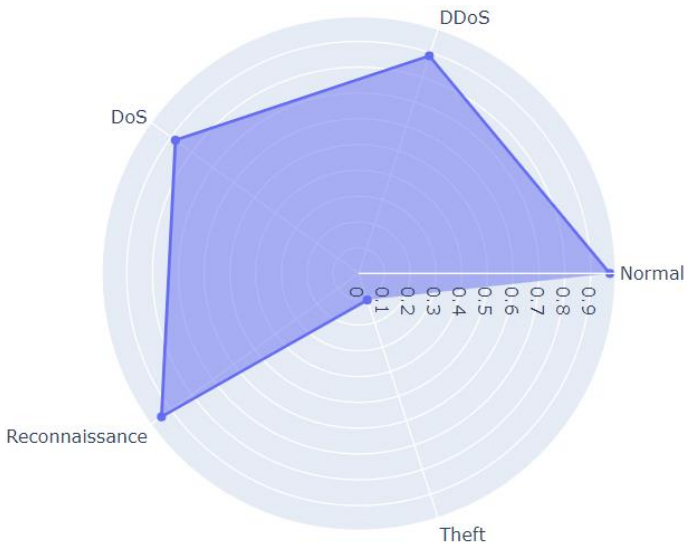
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+----------------+-------------+----------+------------+-----------+
| Class          |   Precision |   Recall |   F1-Score |   Support |
+================+=============+==========+============+===========+
| DDoS           |    0.887121 | 0.868617 |  0.877771  |    240000 |
+----------------+-------------+----------+------------+-----------+
| DoS            |    0.878969 | 0.898623 |  0.888687  |    242788 |
+----------------+-------------+----------+------------+-----------+
| Normal         |    0.974432 | 0.719078 |  0.827503  |       477 |
+----------------+-------------+----------+------------+-----------+
| Reconnaissance |    0.94569  | 0.944781 |  0.945235  |    182166 |
+----------------+-------------+----------+------------+-----------+
| Theft          |    0.108108 | 0.025    |  0.0406091 |       160 |
+----------------+-------------+----------+------------+-----------+

Accuracy: 0.9000978078129062    Recall: 0.6912197717889326
```

Hình 44: Kết quả đánh giá mô hình SVM với dataset Bot-IoT 5% không dùng GbFS



Hình 45: Độ chính xác của mô hình SVM với dataset Bot-IoT 5% không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~82%, độ thu hồi ~50%.

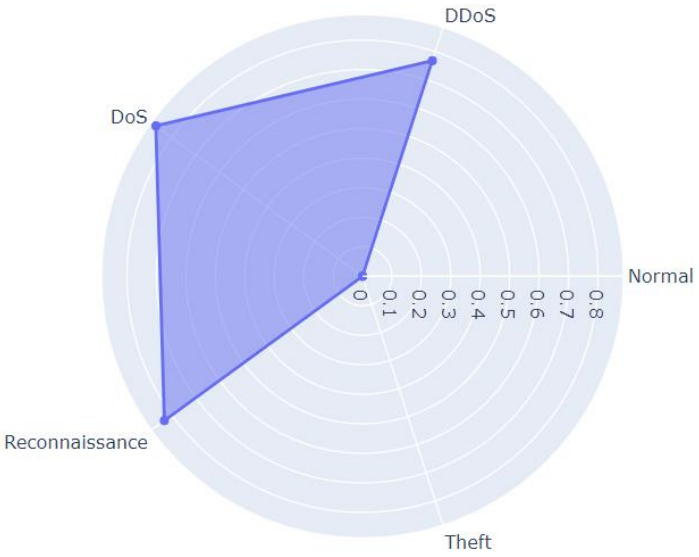
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+----------------+-------------+----------+------------+-----------+
| Class          |   Precision |   Recall |   F1-Score |   Support |
+================+=============+==========+============+===========+
| DDoS           |     0.76764 | 0.835675 |   0.800214 |    240000 |
+----------------+-------------+----------+------------+-----------+
| DoS            |    0.867045 | 0.723372 |   0.788719 |    242788 |
+----------------+-------------+----------+------------+-----------+
| Normal         |    0        | 0        |   0        |       477 |
+----------------+-------------+----------+------------+-----------+
| Reconnaissance |    0.831505 | 0.920957 |   0.873948 |    182166 |
+----------------+-------------+----------+------------+-----------+
| Theft          |    0        | 0        |   0        |       160 |
+----------------+-------------+----------+------------+-----------+

Accuracy: 0.8172511347058479    Recall: 0.49600070807647156
```

Hình 46: Kết quả đánh giá mô hình SVM với dataset Bot-IoT 5% có dùng GbFS



Hình 47: Độ chính xác của mô hình SVM với dataset Bot-IoT 5% có dùng GbFS

Kết quả đánh giá 2 mô hình k-NN của nhóm: [nt204_knn_BOT_IOT_005.ipynb]

- Không dùng GbFS: Độ chính xác đạt ~99.3%, độ thu hồi đạt ~96.6%.

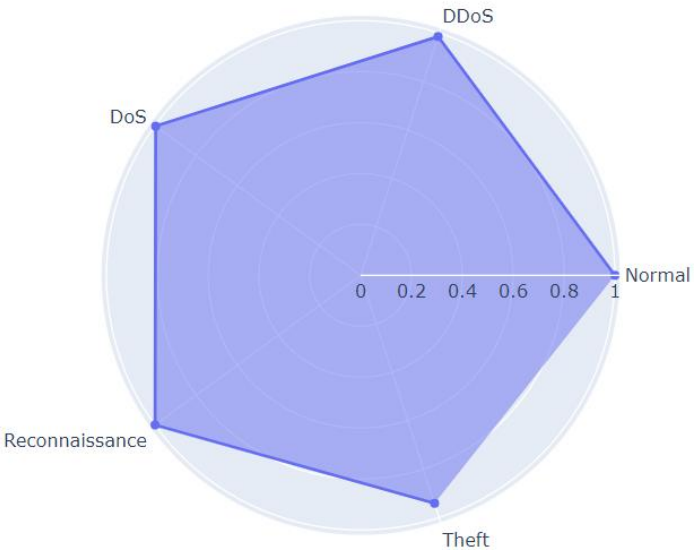
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report['accuracy'],'\tRecall:',report['macro avg']['recall'])
```

```
+----------------+-------------+----------+------------+-----------+
| Class          |   Precision |   Recall |   F1-Score |   Support |
+================+=============+==========+============+===========+
| DDoS           |    0.985911 | 0.996325 |   0.991091 |    240000 |
+----------------+-------------+----------+------------+-----------+
| DoS            |    0.996294 | 0.986655 |   0.991451 |    242788 |
+----------------+-------------+----------+------------+-----------+
| Normal         |    1        | 0.949686 |   0.974194 |       477 |
+----------------+-------------+----------+------------+-----------+
| Reconnaissance |    0.999396 | 0.998545 |   0.99897  |    182166 |
+----------------+-------------+----------+------------+-----------+
| Theft          |    0.941176 | 0.9      |   0.920128 |       160 |
+----------------+-------------+----------+------------+-----------+

Accuracy: 0.9933487682375512    Recall: 0.9662421683202682
```

Hình 48: Kết quả đánh giá mô hình k-NN với dataset Bot-IoT 5% không dùng GbFS



Hình 49: Độ chính xác của mô hình k-NN với dataset Bot-IoT 5% không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~99.5%, độ thu hồi đạt ~94.6%.

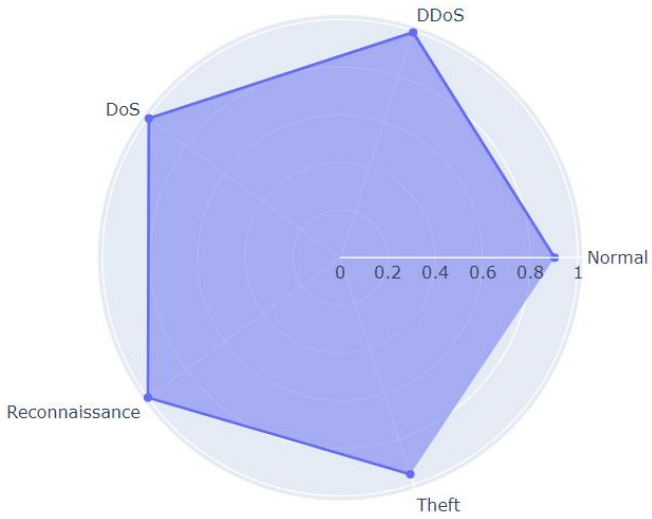
```
from tabulate import tabulate
table = [["Class", "Precision", "Recall", "F1-Score", "Support"]]
for key, value in report_fs.items():
    if key != "accuracy" and key != "macro avg" and key != "weighted avg":
        table.append([key, value['precision'], value['recall'], value['f1-score'], value['support']])

print(tabulate(table, headers="firstrow", tablefmt="grid"))
print('\nAccuracy:',report_fs['accuracy'],'\tRecall:',report_fs['macro avg']['recall'])
```

```
+----------------+-------------+----------+------------+-----------+
| Class          |   Precision |   Recall |   F1-Score |   Support |
+================+=============+==========+============+===========+
| DDoS           |    0.993858 | 0.993871 |   0.993865 |    240000 |
+----------------+-------------+----------+------------+-----------+
| DoS            |    0.993974 | 0.994592 |   0.994283 |    242788 |
+----------------+-------------+----------+------------+-----------+
| Normal         |    0.902    | 0.945493 |   0.923234 |       477 |
+----------------+-------------+----------+------------+-----------+
| Reconnaissance |    0.999    | 0.998172 |   0.998586 |    182166 |
+----------------+-------------+----------+------------+-----------+
| Theft          |    0.955224 | 0.8      |   0.870748 |       160 |
+----------------+-------------+----------+------------+-----------+

Accuracy: 0.9952298032876046    Recall: 0.9464254964992701
```

Hình 50: Kết quả đánh giá mô hình k-NN với dataset Bot-IoT 5% có dùng GbFS



Hình 51: Độ chính xác của mô hình k-NN với dataset Bot-IoT 5% có dùng GbFS

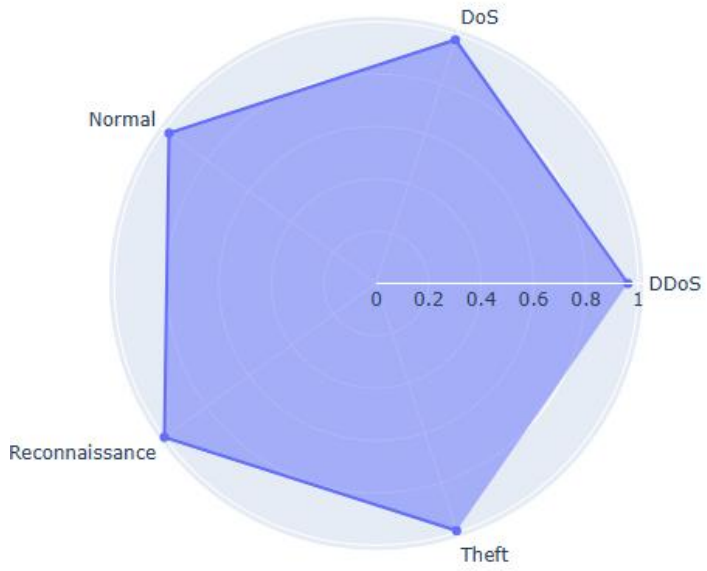
Kết quả đánh giá 2 mô hình XgBoost của nhóm:
[nt204_Xgboost_BOT_IOT_005.ipynb - Colab (google.com)]

- Không dùng GbFS: Độ chính xác đạt ~99.3%, độ thu hồi đạt ~96.6%.

Class	Precision	Recall	F1-Score	Support
0	0.960512	0.978842	0.96959	240000
1	0.978772	0.961139	0.969875	242788
2	0.977221	0.899371	0.936681	477
3	0.999286	0.998419	0.998852	182166
4	0.993333	0.93125	0.96129	160

Accuracy: 0.9776739769618279 Recall: 0.953804139256372

Hình 52: Kết quả đánh giá mô hình Xgboost với dataset Bot-IoT 5% không dùng GbFS



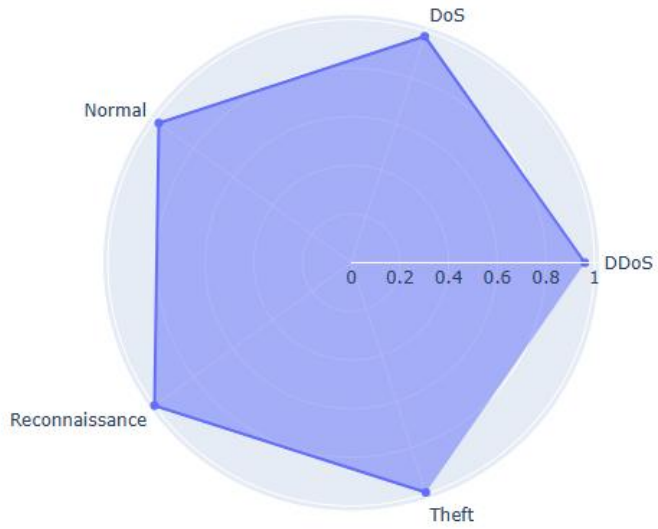
Hình 53: Độ chính xác của mô hình Xgboost với dataset Bot-IoT 5% không dùng GbFS

- Dùng GbFS: Độ chính xác đạt ~99.5%, độ thu hồi đạt ~94.6%.

Class	Precision	Recall	F1-Score	Support
0	0.960512	0.978842	0.96959	240000
1	0.978772	0.961139	0.969875	242788
2	0.977221	0.899371	0.936681	477
3	0.999286	0.998419	0.998852	182166
4	0.993333	0.93125	0.96129	160

Accuracy: 0.9776739769618279 Recall: 0.953804139256372

Hình 54: Kết quả đánh giá mô hình Xgboost với dataset Bot-IoT 5% có dùng GbFS



Hình 55: Độ chính xác của mô hình xgboost với dataset Bot-IoT 5% có dùng GbFS

5.2. Đánh giá kết quả

5.2.1. CIRA-CIC-DOHBrw-2020

	Không dùng GbFS		**Có dùng GbFS**	
	Độ chính xác (%)	Độ thu hồi (%)	Độ chính xác (%)	Độ thu hồi (%)
SVM	~80	~51	~79	~49
k-NN	~73	~44	~72	~42
XgBoost	~80	~65	~79	~50

Bảng 2: So sánh không/có dùng GbFS với dataset CIRA-CIC-DOHBrw-2020

Dựa vào bảng 2, có thể thấy việc áp dụng GbFS của nhóm không giúp tăng độ chính xác và độ thu hồi cho mô hình, mà còn giảm.

5.2.2. UNSW-NB15

	Không dùng GbFS		**Có dùng GbFS**	
	Độ chính xác (%)	Độ thu hồi (%)	Độ chính xác (%)	Độ thu hồi (%)
SVM	~83	~39	~78	~31
k-NN	~84	~46	~78	~38
XgBoost	~88	~55	~82	~37

Bảng 3: So sánh không/có dùng GbFS với dataset UNSW-NB15

Dựa vào bảng 3, có thể thấy việc áp dụng GbFS không giúp tăng độ chính xác và độ thu hồi cho mô hình, mà còn giảm.

5.2.3. Bot-IoT 5%

	Không dùng GbFS		**Có dùng GbFS**	
	Độ chính xác (%)	Độ thu hồi (%)	Độ chính xác (%)	Độ thu hồi (%)
SVM	~90	~69	~82	~50
k-NN	~99.3	~96.6	~99.5	~94.6
XgBoost	~99	~97	~99	~95

Bảng 4: So sánh không/có dùng GbFS với dataset Bot-IoT 5%

Dựa vào bảng 4, có thể thấy việc áp dụng GbFS không giúp tăng độ chính xác và độ thu hồi cho các mô hình, mà còn giảm, ngoại trừ với mô hình k-NN có tăng 1 chút độ chính xác.

5.2.4. Đánh giá chung

Dựa trên thí nghiệm thực tế, kết quả thu được đã không thể chứng minh rằng GbFS sẽ chắc chắn tăng độ chính xác và độ thu hồi cho mô hình học. Điều này có nghĩa là phương pháp mà bài báo đề xuất không hiệu quả như bài báo viết hoặc GbFS mà nhóm thiết kế dựa trên mô tả của bài báo không đúng.

Đánh giá riêng của nhóm:

- Việc giảm số đặc trưng xuống 1 con số, cụ thể là 10, quá thấp so với tổng số đặc trưng của dataset sẽ giảm khiến mô hình học máy không đủ thông tin để phân loại dẫn đến giảm hiệu suất.
- Đối với công thức đánh giá độ fitness mà bài báo đề xuất, nhóm cho rằng ý tưởng tìm kiếm đặc trưng có tính đa dạng cao của tác giả là không phù hợp vì không phải dữ liệu đa dạng có nghĩa là dữ liệu đó có ý nghĩa cho việc phân loại.
- Độ chính xác được trong công thức đánh giá độ fitness được bài báo nhắc tới là kết quả của hàm "mục tiêu" nhưng trong cả bài báo chỉ nói là "đề ra một hàm mục tiêu mới" chứ không miêu tả rõ ràng hàm "mục tiêu" này, nên nhóm đã lấy ý tưởng từ các phương pháp lựa chọn đặc trưng sử dụng thuật toán di truyền trước đó (độ chính xác có từ kết quả đánh giá 1 mô hình phân loại). Đây có thể là nguyên nhân mà kết quả của nhóm quá kém so với bài báo.

6. Kết luận và hướng phát triển

6.1. Kết luận

Có những tập dữ liệu có nhiều đặc trưng nhưng không phải đặc trưng nào cũng có ý nghĩa, có ích trong việc phân loại, các đặc trưng này gây ảnh hưởng tiêu cực (giảm đôh chính xác, độ thu hồi) đến kết quả phân loại của mô hình máy học.

Phương pháp mà bài báo đề ra nhằm giải quyết vấn đề trên bằng cách áp dụng thuật toán di truyền để chọn ra các đặc trưng tối ưu để giảm chiều tập dữ liệu, loại bỏ các đặc trưng "nhiễu", tăng hiệu suất cho mô hình máy học.

Tuy nhiên kết quả thực nghiệm cho thấy "phương pháp lựa chọn đặc trưng dựa trên thuật toán di truyền" mà bài báo đề xuất không chắc chắn gia tăng độ chính xác và độ thu hồi cho các mô hình học, mà còn giảm hiệu suất.

Nguyên nhân kết quả không như mong muốn có thể là do nhóm thiết kế sai (do bài báo viết không rõ ràng về hàm fitness) hoặc phương pháp được viết trong bài báo bài báo không đúng.

6.2. Hướng phát triển

Chúng tôi đề xuất một số hướng nghiên cứu tương lai để tiếp tục cải thiện hiệu suất của hệ thống phát hiện xâm nhập:

- Nghiên cứu sâu hơn về các phương pháp chọn lọc đặc trưng: Cần tiếp tục khám phá và so sánh các phương pháp chọn lọc đặc trưng khác nhau để tìm ra phương pháp hiệu quả nhất. Các phương pháp như phân tích thành phần chính (PCA), LASSO và SVM-RFE có thể được khám phá để xem xét khả năng cải thiện hiệu suất của hệ thống phát hiện xâm nhập.
- Tối ưu hóa thuật toán di truyền: Cần nghiên cứu và tối ưu hóa các tham số của thuật toán di truyền như kích thước quần thể, số lượng thế hệ, phép lai ghép và biến đổi. Sự tinh chỉnh các tham số này có thể cải thiện khả năng tìm ra giải pháp tối ưu và tăng hiệu suất của thuật toán di truyền trong hệ thống phát hiện xâm nhập.
- Áp dụng các phương pháp học sâu: Các mô hình học sâu như mạng nơ-ron sâu (deep neural networks) có thể được áp dụng để cải thiện hiệu suất của hệ thống phát hiện xâm nhập. Nghiên cứu về việc kết hợp các thuật toán di truyền và mô hình học sâu có thể mang lại kết quả tốt hơn trong việc phát hiện và phòng ngừa các cuộc tấn công xâm nhập.
- Nghiên cứu về bảo mật hệ thống dựa trên trí tuệ nhân tạo: Trong tương lai, nên tiếp tục nghiên cứu về việc áp dụng trí tuệ nhân tạo và học máy trong bảo mật hệ thống. Các phương pháp như học tăng cường (reinforcement learning) và mạng gan (generative adversarial networks) có thể đượcsử dụng để phát triển các hệ thống phát hiện xâm nhập thông minh và linh hoạt hơn.
- Tăng cường tập dữ liệu và độ đa dạng: Cần có sự đa dạng và đại diện trong tập dữ liệu được sử dụng để huấn luyện và đánh giá hệ thống phát hiện xâm nhập. Việc tăng cường tập dữ liệu có thể bao gồm việc thu thập dữ liệu từ nhiều

nguồn và mô phỏng các cuộc tấn công khác nhau để đảm bảo tính toàn vẹn và hiệu quả của hệ thống phát hiện xâm nhập.

- Tích hợp hệ thống phát hiện xâm nhập vào mạng lưới IoT: Với sự phát triển của Internet of Things (IoT), việc bảo mật và phát hiện xâm nhập trong mạng lưới IoT trở thành một thách thức quan trọng. Nghiên cứu tương lai nên tập trung vào việc phát triển các phương pháp và công nghệ để tích hợp hệ thống phát hiện xâm nhập vào mạng lưới IoT, để đảm bảo an toàn và bảo mật cho các thiết bị và dữ liệu trong môi trường IoT.

7. Tài liệu tham khảo

- Link bài báo: https://www.sciencedirect.com/science/article/abs/pii/S0167404821002728
- Tham khảo ý tưởng thiết kế GbFS: https://www.kaggle.com/code/tanmayunhale/genetic-algorithm-for-feature-selection