**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

Tel. (+84.0236) 3736949, Fax. (84-511) 3842771

Website: http://dut.udn.vn/khoacntt, E-mail: cntt@dut.udn.vn



**BÁO CÁO MÔN HỌC**

**AN TOÀN VÀ BẢO MẬT THÔNG TIN**

**ĐỀ TÀI :**

**ỨNG DỤNG TRÍ TUỆ NHÂN TẠO TRONG PHÁT HIỆN VÀ PHÂN LOẠI CÁC HÀNH VI BẤT THƯỜNG DỰA TRÊN LƯU LƯỢNG MẠNG**

|  |  |  |
| --- | --- | --- |
| HỌ TÊN SINH VIÊN | MÃ SINH VIÊN | NHÓM |
| Võ Hoàng Bảo | 102200246 | 20.Nh15 |
| Trương Bảo Ngọc | 102200276 | 20.Nh15 |
| Trần Thị Mỹ Linh | 102200268 | 20.Nh15 |

**CBHD : PGS. TS. Nguyễn Tấn Khôi**

**Đà Nẵng, 11/2024**

**MỤC LỤC**

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 2](#_Toc181763517)

[1.1. CƠ SỞ LÝ THUYẾT 2](#_Toc181763518)

[1.1.1. Gói dữ liệu 2](#_Toc181763519)

[1.1.3. Ứng dụng của AI trong phát hiện hành vi bất thường 3](#_Toc181763520)

[1.2. PHÁT BIỂU BÀI TOÁN 5](#_Toc181763521)

[1.3. DỮ LIỆU 5](#_Toc181763522)

[1.3.1. Nội dung và cấu trúc dữ liệu: 5](#_Toc181763523)

[1.3.2. Tấn công được mô phỏng: 6](#_Toc181763524)

[1.3.3. Số lượng: 6](#_Toc181763525)

[1.4. MÔ HÌNH 7](#_Toc181763526)

[1.4.1. K-Nearest Neighbors (KNN) 7](#_Toc181763527)

[1.4.2. Random Forest 9](#_Toc181763528)

[1.4.3. Neural Network Classification 10](#_Toc181763529)

[CHƯƠNG 2: TRIỂN KHAI 12](#_Toc181763530)

[2.1. THU THẬP DỮ LIỆU LƯU LƯỢNG THỰC 12](#_Toc181763531)

[2.2. TIỀN XỬ LÝ DỮ LIỆU 12](#_Toc181763532)

[2.3. HUẤN LUYỆN MÔ HÌNH 12](#_Toc181763533)

[2.3.1. K-Nearest Neighbors (KNN) 12](#_Toc181763534)

[2.3.2. Random Forest 16](#_Toc181763535)

[2.3.3. Neural Network Classification 20](#_Toc181763536)

[2.4. XÂY DỰNG MODULE BẮT GÓI TIN 24](#_Toc181763537)

[CHƯƠNG 3: ĐÁNH GIÁ KẾT QUẢ 26](#_Toc181763538)

[3.1. KẾT QUẢ THỰC NGHIỆM 26](#_Toc181763539)

[3.1.1. Trường hợp bình thường 26](#_Toc181763540)

[3.1.2. Trường hợp bị PortScan với nmap 26](#_Toc181763541)

[3.1.3. Trường hợp bị tấn công DDos 27](#_Toc181763542)

[3.2. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ 28](#_Toc181763543)

**DANH SÁCH HÌNH ẢNH**

[Hình 1. Giới thiệu KNN 8](#_Toc181763412)

[Hình 2. Giới thiệu Random Forest 9](#_Toc181763413)

[Hình 3. Giới thiệu Neural Network 10](#_Toc181763414)

[Hình 4. Ma trận nhầm lẫn huấn luyện KNN 13](#_Toc181763415)

[Hình 5. Các thông số đánh giá huấn luyện KNN 14](#_Toc181763416)

[Hình 6. Ma trận nhầm lẫn kiểm thử KNN 15](#_Toc181763417)

[Hình 7. Thông số đánh giá kiểm thử KNN 16](#_Toc181763418)

[Hình 8. Ma trận nhẫm lẫn huấn luyện Random Forest 17](#_Toc181763419)

[Hình 9. Thông số đánh giá huấn luyện Random Forest 18](#_Toc181763420)

[Hình 10. Ma trận nhầm lẫn kiểm thử Random Forest 19](#_Toc181763421)

[Hình 11. Thông số đánh giá kiểm thử Random Forest 20](#_Toc181763422)

[Hình 12. Mô tả Neural Network 21](#_Toc181763423)

[Hình 13. Tham số mô hình Neural Network 22](#_Toc181763424)

[Hình 14. Ma trận nhầm lẫn huấn luyện Neural Network 22](#_Toc181763425)

[Hình 15. Thông số đánh giá huấn luyện Neural Network 22](#_Toc181763426)

[Hình 16. Ma trận nhầm lẫn kiển thử Neural Network 23](#_Toc181763427)

[Hình 17. Thông số đánh giá kiểm thử Neural Network 24](#_Toc181763428)

[Hình 18. Mẫu gói tin bắt được 25](#_Toc181763429)

[Hình 19. Kết quả thực nghiệm mô hình trường hợp bình thường 26](#_Toc181763430)

[Hình 20. Kết quả thực nghiệm mô hình trường hợp PortScan 27](#_Toc181763431)

[Hình 21. Kết quả thực nghiệm mô hình trường hợp Ddos 28](#_Toc181763432)

**DANH SÁCH TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | Nội dung | Giải thích |
| IP | Internet Protocol | Giao thức liên mạng |
| CSV | Comma-Separated Values | Giá trị Phân tách bằng Dấu phẩy |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| ML | Machine Learning | Học máy |
| TCP | Transmission Control Protocol | Giao thức điều khiển truyền dẫn, đảm bảo truyền dữ liệu chính xác giữa các thiết bị mạng |
| UDP | User Datagram Protocol | Giao thức truyền dữ liệu không đảm bảo, nhanh hơn TCP nhưng không kiểm tra độ tin cậy |
| ICMP | Internet Control Message Protocol | Giao thức kiểm soát tin nhắn Internet |
| DDos | Distributed Denial of Service | Tấn công từ chối dịch vụ phân tán |
| CNN | Convolutional Neural Network | Mạng nơ-ron tích chập |
| RNN | Recurrent Neural Network | Mạng nơ-ron hồi quy |
| KNN | K-Nearest Neighbors | Thuật toán hàng xóm gần nhất K |
| IDS | Intrusion Detection System | Hệ thống phát hiện xâm nhập |
| XXS | Cross-Site Scripting | Kịch bản chéo trang, một lỗ hổng bảo mật cho phép kẻ tấn công chèn mã độc |
| FTP | File Transfer Protocol | Giao thức truyền tập tin |
| SSH | Secure Socket Shell | Giao thức kết nối giữa máy khách (local) và máy chủ (server) để điều khiển từ xa |

**BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** | **Các nhiệm vụ** | **Đánh giá** |
| Võ Hoàng Bảo | Thu thập, xử lý dữ liệu | Đã hoàn thành |
| Cài đặt máy ảo | Đã hoàn thành |
| Lập trình thực nghiệm bắt gói tin | Đã hoàn thành |
| Tích hợp mô hình AI đã huấn luyện vào hệ thống | Đã hoàn thành |
| Trực quan hóa | Đã hoàn thành |
| Tổng hợp, đánh giá mô hình | Đã hoàn thành |
| Viết báo cáo | Đã hoàn thành |
| Trương Bảo Ngọc | Xử lý dữ liệu | Đã hoàn thành |
| Nghiên cứu huấn luyện mô hình KNNs | Đã hoàn thành |
| Nghiên cứu huấn luyện mô hình Random Forest | Đã hoàn thành |
| Nhận xét, đánh giá mô hình | Đã hoàn thành |
| Viết báo cáo | Đã hoàn thành |
| Trần Thị Mỹ Linh | Xử lý dữ liệu | Đã hoàn thành |
| Nghiên cứu huấn luyện mô hình Neural Network Classification | Đã hoàn thành |
| Thử nghiệm, điều chỉnh tham số | Đã hoàn thành |
| Nhận xét, đánh giá mô hình | Đã hoàn thành |
| Viết báo cáo | Đã hoàn thành |

**MỞ ĐẦU**

Để ứng phó với các mối đe dọa an ninh mạng ngày càng phức tạp, việc phát hiện và phân loại các hành vi bất thường trong lưu lượng mạng trở thành một yêu cầu thiết yếu trong hệ thống bảo mật hiện đại. Các kỹ thuật phát hiện truyền thống dựa trên quy tắc và chữ ký ngày càng khó khăn trong việc theo dõi và phát hiện các loại tấn công mới, chưa kể đến những biến thể tấn công không ngừng thay đổi.

Với sự tiến bộ của công nghệ trí tuệ nhân tạo (AI) và học máy (ML), các hệ thống phát hiện bất thường giờ đây có khả năng tự học từ dữ liệu và thích ứng với các mẫu tấn công mới một cách nhanh chóng, hiệu quả. Nhóm chúng tôi đã tiến hành nghiên cứu và triển khai một hệ thống phát hiện và phân loại hành vi dị thường dựa trên dữ liệu lưu lượng mạng. Nghiên cứu bao gồm hai nhiệm vụ chính: chuyển đổi dữ liệu lưu lượng mạng thành các đặc trưng phù hợp để phân tích và xây dựng hệ thống phát hiện sử dụng các thuật toán ML tiên tiến nhằm cải thiện khả năng phát hiện chính xác các hành vi bất thường.

Với sự kết hợp của các phương pháp AI/ML trong phân tích lưu lượng mạng, hệ thống này kỳ vọng sẽ cung cấp giải pháp mạnh mẽ trong việc phát hiện các hành vi bất thường, từ đó nâng cao khả năng bảo vệ hệ thống mạng trước các mối đe dọa tiềm ẩn.

.

# CƠ SỞ LÝ THUYẾT

## CƠ SỞ LÝ THUYẾT

### Gói dữ liệu

Gói dữ liệu là đơn vị cơ bản trong truyền thông mạng, là cấu trúc chứa thông tin cần thiết để truyền tải dữ liệu từ một thiết bị đến thiết bị khác. Mỗi gói dữ liệu thường bao gồm ba thành phần chính:

* **Header** (Phần đầu): Chứa thông tin điều khiển như địa chỉ IP nguồn và đích, loại giao thức (TCP, UDP, ICMP, v.v.), và các thông tin cần thiết khác để đảm bảo gói được xử lý đúng cách.
* **Payload** (Phần tải): Là phần dữ liệu thực sự mà gói chứa, có thể là các tập tin, văn bản, hình ảnh, hoặc bất kỳ loại dữ liệu nào. Kích thước của payload có thể thay đổi tùy theo loại dữ liệu và giao thức.
* **Trailer** (Phần đuôi): Đôi khi bao gồm thông tin kiểm tra lỗi (Checksum), giúp đảm bảo rằng dữ liệu trong gói không bị hỏng trong quá trình truyền tải.

Quá trình gửi gói dữ liệu qua mạng bao gồm các bước sau:

* **Tạo Gói**: Dữ liệu từ ứng dụng được chia thành các gói nhỏ hơn, mỗi gói có kích thước tối ưu cho việc truyền tải.
* **Địa Chỉ và Gửi**: Các gói được đánh dấu bằng địa chỉ IP nguồn và đích. Gói được chuyển tới thiết bị mạng gần nhất (router/switch) để bắt đầu quá trình truyền tải.
* **Chuyển Tiếp**:Tại mỗi router, gói dữ liệu được xem xét và chuyển tiếp tới router tiếp theo cho đến khi đến đích cuối cùng. Quá trình này có thể thay đổi tùy thuộc vào trạng thái của mạng.
* **Tái Tạo Dữ Liệu**: Khi các gói đến đích, chúng được ghép lại theo thứ tự đúng để tái tạo dữ liệu ban đầu. Nếu một hoặc nhiều gói bị mất, thiết bị nhận sẽ yêu cầu gửi lại các gói bị thiếu.

Có nhiều giao thức khác nhau liên quan đến việc truyền tải gói dữ liệu, trong đó nổi bật là:

* **TCP** (Transmission Control Protocol): Đảm bảo truyền tải dữ liệu đáng tin cậy. Nó xác nhận rằng các gói được nhận đúng thứ tự và không bị mất.
* **UDP** (User Datagram Protocol): Một giao thức không đáng tin cậy, không đảm bảo thứ tự gói hoặc xác nhận nhận gói. Thích hợp cho các ứng dụng cần tốc độ như video streaming.
* **ICMP** (Internet Control Message Protocol): Sử dụng để gửi các thông điệp lỗi và thông tin điều khiển trong mạng.
  + 1. **Hành vi bất thường**

Hành vi bất thường trên mạng đề cập đến các hoạt động không điển hình mà có thể chỉ ra các vấn đề bảo mật hoặc sự cố trong mạng. Các hành vi này có thể gây ra rủi ro cho hệ thống và dữ liệu. Một số ví dụ phổ biến bao gồm:

* **Tấn công từ chối dịch vụ (DDoS)**: Nhiều gói dữ liệu được gửi đến một địa chỉ đích với tốc độ cao nhằm làm ngạt băng thông, gây gián đoạn dịch vụ.
* **Tấn công man-in-the-middle**: Kẻ tấn công can thiệp vào giao tiếp giữa hai thiết bị mà không bị phát hiện, có thể đánh cắp hoặc thay đổi dữ liệu.
* **Thâm nhập trái phép**: Các gói dữ liệu không hợp lệ hoặc độc hại cố gắng truy cập vào mạng, có thể là từ các thiết bị đã bị xâm nhập.
* **Chuyển tiếp gói không hợp lệ**: Các gói không tuân theo quy tắc mạng có thể chỉ ra rằng một thiết bị trong mạng đã bị xâm phạm hoặc bị lỗi.

### Ứng dụng của AI trong phát hiện hành vi bất thường

AI có khả năng phân tích lượng dữ liệu lớn và phát hiện các mẫu không bình thường mà phương pháp truyền thống có thể bỏ sót. Một số phương pháp chính:

1. **Học máy giám sát**:

Học máy giám sát là một phương pháp mạnh mẽ trong AI, trong đó các mô hình được huấn luyện trên các tập dữ liệu đã được gán nhãn để phân loại hành vi vào các nhóm cụ thể. Quá trình này bao gồm một số bước quan trọng:

* **Xây dựng Dữ liệu Huấn luyện**: Để mô hình học máy hoạt động hiệu quả, cần thu thập một tập dữ liệu phong phú, bao gồm cả các hành vi bình thường và bất thường. Việc gán nhãn cho dữ liệu này là rất quan trọng, vì nó sẽ định hình cách mà mô hình hiểu và phân loại các mẫu.
* **Chọn Thuật Toán Phù Hợp**: Các thuật toán như láng giềng gần nhất, hồi quy logistic, hoặc Random Forest có thể được sử dụng để xây dựng mô hình. Tùy thuộc vào loại dữ liệu và tính chất của bài toán, một thuật toán cụ thể có thể cho kết quả tốt hơn những thuật toán khác.
* **Huấn luyện và Tinh chỉnh Mô Hình**: Sau khi chọn thuật toán, mô hình sẽ được huấn luyện trên dữ liệu đã gán nhãn. Quá trình này có thể bao gồm điều chỉnh các tham số của mô hình để cải thiện độ chính xác và khả năng tổng quát của nó.
* **Kiểm Tra và Đánh Giá**: Sau khi huấn luyện, mô hình cần được kiểm tra với một tập dữ liệu khác để đánh giá hiệu suất. Các chỉ số như độ chính xác, độ nhạy và độ đặc hiệu sẽ được sử dụng để xác định khả năng phát hiện hành vi bất thường của mô hình.

1. **Mạng nơ-ron**:

Mạng nơ-ron, một lĩnh vực con trong học máy, sử dụng các kiến trúc phức tạp để mô phỏng cách thức hoạt động của não người. Các mạng này có khả năng xử lý và phân tích dữ liệu phi cấu trúc rất hiệu quả. Đặc điểm nổi bật của mạng nơ-ron trong phát hiện hành vi bất thường bao gồm:

* **Kiến trúc Đa tầng**: Mạng nơ-ron bao gồm nhiều lớp nơ-ron, mỗi lớp có thể học các đặc trưng khác nhau từ dữ liệu. Các mạng như Mạng Nơ-ron Tích Chập (CNN) và Mạng Nơ-ron Hồi Tiếp (RNN) đã được chứng minh là rất hiệu quả trong việc phát hiện hành vi bất thường trong dữ liệu phức tạp như hình ảnh và tín hiệu thời gian.
* **Khả năng Tự Học**: Mạng nơ-ron có khả năng tự học và cải thiện hiệu suất theo thời gian khi có thêm dữ liệu. Điều này rất quan trọng trong môi trường mạng, nơi mà các hành vi và mối đe dọa liên tục thay đổi.
* **Xử Lý Dữ Liệu Lớn**: Với khả năng xử lý song song, mạng nơ-ron có thể xử lý một lượng lớn dữ liệu đồng thời, giúp phát hiện hành vi bất thường nhanh chóng và hiệu quả.
* **Giải Thích và Độ Tin Cậy**: Mặc dù mạng nơ-ron cung cấp độ chính xác cao trong việc phát hiện, nhưng một trong những thách thức lớn là khả năng giải thích các quyết định của nó. Việc cải thiện khả năng giải thích của mạng nơ-ron là một lĩnh vực nghiên cứu đang phát triển, nhằm làm tăng độ tin cậy trong việc phát hiện các hành vi bất thường.

## PHÁT BIỂU BÀI TOÁN

Bài toán đặt ra là phát hiện và phân loại các hành vi bất thường trong lưu lượng mạng bằng cách sử dụng các kỹ thuật trí tuệ nhân tạo. Mục tiêu là xây dựng một hệ thống có khả năng tự động nhận diện các hành vi không bình thường trong lưu lượng mạng, từ đó giúp cải thiện khả năng bảo mật và quản lý mạng.

## DỮ LIỆU

Dữ liệu nhóm dùng để huấn luyện mô hình chính là dữ liệu CIC IDS 2017 (Canadian Institute for Cybersecurity Intrusion Detection System 2017) là một tập dữ liệu nổi bật được sử dụng để nghiên cứu và phát triển các hệ thống phát hiện xâm nhập (IDS). Tập dữ liệu này bao gồm thông tin về nhiều loại tấn công mạng khác nhau và hành vi bình thường, giúp các nhà nghiên cứu và kỹ sư an ninh mạng phát triển và kiểm tra các mô hình học máy cho mục đích phát hiện và phân loại các hành vi bất thường trong lưu lượng mạng.

### Nội dung và cấu trúc dữ liệu:

|  |  |
| --- | --- |
| **Đặc Điểm** | **Mô Tả** |
| **Ghi chú** | Mỗi bản ghi trong tập dữ liệu đại diện cho một phiên lưu lượng mạng với nhiều đặc điểm khác nhau. |
| **Thời gian** | Timestamp ghi lại thời điểm mà gói dữ liệu được thu thập. |
| **Địa chỉ IP** | Địa chỉ IP nguồn và đích, cho biết nguồn và đích của lưu lượng. |
| **Cổng** | Cổng nguồn và cổng đích, xác định giao thức và dịch vụ được sử dụng. |
| **Kích thước gói** | Kích thước của gói dữ liệu, có thể ảnh hưởng đến hành vi của mạng. |
| **Giao thức** | Thông tin về giao thức sử dụng trong phiên lưu lượng (TCP, UDP, ICMP, v.v.). |
| **Tình trạng kết nối** | Thông tin về trạng thái của kết nối (có thể là mở, đóng hoặc đang thiết lập). |
| **Số lượng gói** | Số lượng gói đã được gửi trong phiên. |
| **Thời gian giữa các gói** | Thời gian giữa các gói trong phiên, giúp phân tích tính liên tục của lưu lượng. |
| **Các tính năng khác** | Bao gồm các thông tin như số byte gửi đi, số byte nhận về, và các thông số khác liên quan đến lưu lượng. |

### Tấn công được mô phỏng:

Tập dữ liệu này mô phỏng các loại tấn công khác nhau, như:

|  |  |
| --- | --- |
| **Tấn công** | **Mô tả** |
| **DDoS** | Tấn công từ chối dịch vụ phân tán |
| **Brute Force** | Tấn công thử tất cả các mật khẩu khả thi |
| **XSS** | Tấn công lừa đảo bằng cách thực thi mã độc trên trình duyệt của người dùng |
| **SQL Injection** | Tấn công thông qua việc chèn mã SQL độc hại vào các truy vấn của cơ sở dữ liệu |
| **Port Scanning** | Quét cổng để tìm kiếm lỗ hổng bảo mật |

### Số lượng:

Số lượng được tổng hợp theo nhãn như sau:

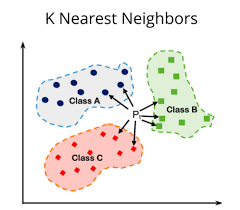
|  |  |
| --- | --- |
| **Hành vi mạng** | **Số lượng** |
| **BENIGN (hành vi bình thường)** | 2,095,057 |
| **DoS Hulk** | 172,846 |
| **DDoS** | 128,014 |
| **PortScan** | 90,694 |
| **DoS GoldenEye** | 10,286 |
| **FTP-Patator** | 5,931 |
| **DoS slowloris** | 5,385 |
| **DoS Slowhttptest** | 5,228 |
| **SSH-Patator** | 3,219 |
| **Bot** | 1,948 |
| **Web Attack - Brute Force** | 1,47 |
| **Web Attack - XSS** | 652 |
| **Infiltration** | 36 |
| **Web Attack - Sql Injection** | 21 |
| **Heartbleed** | 11 |

Có thể thấy số lượng nhãn các hành vi không đều nhau nên để đảm bảo không bị mất cân bằng dữ liệu, nhóm đã lựa chọn ngưỡng 3000, các tấn công có số lượng mẫu dưới 3000 sẽ được phân loại vào nhóm các cuộc tấn công khác (Others)

## MÔ HÌNH

Nhóm đã thử nghiệm 4 mô hình có độ phức tạp từ thấp tới cao là KNN, Random Forest, Neural Network Classification.

### K-Nearest Neighbors (KNN)

KNN là một trong những thuật toán phân loại đơn giản nhất. Mô hình này hoạt động dựa trên nguyên tắc rằng các điểm dữ liệu gần nhau có khả năng thuộc về cùng một lớp. Khi nhận được một điểm dữ liệu mới, KNN sẽ tìm K điểm gần nhất trong tập huấn luyện và phân loại điểm đó dựa trên lớp chiếm ưu thế trong số K điểm. KNN dễ triển khai và không cần quá trình huấn luyện phức tạp, nhưng độ chính xác của nó có thể bị ảnh hưởng bởi các điểm nhiễu và không hiệu quả với tập dữ liệu lớn.

Các bước triển khai:

**Bước 1**: Tính toán khoảng cách giữa điểm dữ liệu mới và tất cả các điểm dữ liệu trong tập huấn luyện. Một số cách đo khoảng cách phổ biến là:

**Khoảng cách Euclidean (thường dùng nhất)**:

Hình 1. Giới thiệu KNN

**Khoảng cách Manhattan:** 

**Khoảng cách Minkowski và Khoảng cách Hamming** cũng có thể dùng cho các bài toán đặc thù.

**Bước 2**: Lựa chọn K điểm dữ liệu gần nhất (các hàng xóm gần nhất) dựa trên khoảng cách vừa tính.

**Bước 3:** Dựa vào lớp của các điểm trong K hàng xóm gần nhất để dự đoán lớp của điểm mới:

**Nếu KNN được sử dụng cho phân loại**: Lớp chiếm ưu thế trong số K hàng xóm sẽ là lớp của điểm mới.

**Nếu KNN được sử dụng cho hồi quy:** Kết quả sẽ là trung bình hoặc trung vị của các giá trị tại K điểm gần nhất.

### Random Forest

Random Forest là một mô hình ensemble, kết hợp nhiều cây quyết định để cải thiện độ chính xác và giảm thiểu hiện tượng overfitting. Mỗi cây trong rừng được xây dựng trên một mẫu ngẫu nhiên của tập dữ liệu và đưa ra dự đoán riêng. Dự đoán cuối cùng được đưa ra dựa trên sự đồng thuận của tất cả các cây. Mô hình này rất linh hoạt, có khả năng xử lý cả dữ liệu phân loại và hồi quy, nhưng có thể khó giải thích hơn so với các mô hình đơn giản như KNN.



Hình 2. Giới thiệu Random Forest

Nguyên tắc hoạt động:

**Cây Quyết Định:** Random Forest xây dựng một tập hợp lớn các cây quyết định (decision trees) trong quá trình huấn luyện. Mỗi cây trong rừng được huấn luyện trên một mẫu ngẫu nhiên (bootstrap sample) của tập dữ liệu huấn luyện.

**Lựa Chọn Đặc Trưng Ngẫu Nhiên:** Khi xây dựng mỗi cây, Random Forest chọn một tập con ngẫu nhiên các đặc trưng (features) để xem xét tại mỗi nút phân nhánh. Điều này giúp tăng tính đa dạng của các cây và giảm thiểu mối quan hệ tương quan giữa chúng.

**Dự Đoán:** Khi dự đoán, mỗi cây trong rừng sẽ đưa ra một dự đoán độc lập. Kết quả cuối cùng được quyết định bằng cách lấy đối số đa số cho bài toán phân loại (lớp chiếm ưu thế) hoặc trung bình cho bài toán hồi quy.

Cấu trúc của Random forest:

**Bootstrap Aggregating (Bagging):** Random Forest áp dụng kỹ thuật bagging, trong đó nó sử dụng nhiều mẫu ngẫu nhiên từ tập huấn luyện để xây dựng cây quyết định. Kỹ thuật này giúp giảm thiểu độ phương sai và cải thiện độ chính xác.

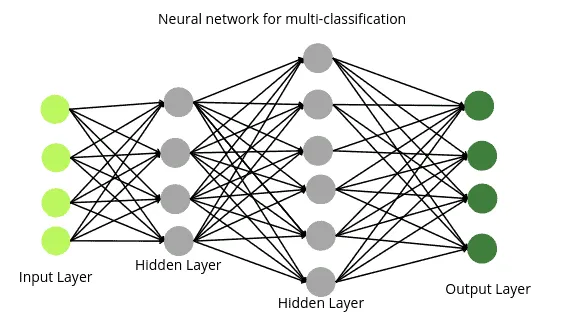
**Sự Ngẫu Nhiên:** Việc chọn ngẫu nhiên các đặc trưng tại mỗi nút giúp mỗi cây trong rừng có khả năng học các mẫu khác nhau, do đó tăng tính chính xác và giảm thiểu overfitting.

### Neural Network Classification

Mạng nơ-ron là một mô hình học máy phức tạp, lấy cảm hứng từ cách hoạt động của bộ não con người. Mô hình này bao gồm nhiều lớp nơ-ron kết nối với nhau, cho phép nó học được các đặc trưng phức tạp từ dữ liệu. Neural Network được sử dụng rộng rãi trong nhiều ứng dụng như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên và nhiều lĩnh vực khác. Tuy nhiên, việc huấn luyện mạng nơ-ron yêu cầu tài nguyên tính toán lớn và có thể gặp khó khăn trong việc điều chỉnh tham số.

Một mô hình mạng nơ-ron cơ bản cho bài toán phân loại hành vi mạng có thể có cấu trúc gồm các lớp sau:

Hình 3. Giới thiệu Neural Network



**Lớp đầu vào (Input layer)**: Dữ liệu đầu vào, cụ thể đổi với bài toán này thì là dữ liệu mạng, chẳng hạn như các đặc trưng mạng (IP addresses, port numbers, packet lengths, protocols, v.v.). Các đặc trưng này được chuẩn hóa hoặc tiền xử lý trước khi đưa vào mô hình.

**Lớp ẩn (Hidden layers)**: Các lớp này có thể có từ 1 đến nhiều lớp nơ-ron. Mỗi lớp ẩn có thể sử dụng các hàm kích hoạt như **ReLU** (Rectified Linear Unit) để tạo ra một hàm phi tuyến tính, cho phép mạng học các mối quan hệ phức tạp trong dữ liệu.

**Lớp đầu ra (Output layer)**: Dành cho phân loại, với số đơn vị bằng số lớp cần phân loại (ví dụ: hành vi bình thường, tấn công DoS, tấn công DDoS, v.v.). Lớp này thường sử dụng hàm kích hoạt *Softmax* trong bài toán phân loại đa lớp, vì nó cung cấp xác suất phân loại cho mỗi lớp.

Các tham số, môi trường huấn luyện cho bài toán sử dụng mạng nơ-ron là:

**Hàm mất mát (Loss function)**: Mạng nơ-ron học từ dữ liệu thông qua quá trình tối ưu hóa, trong đó *loss function* là một hàm được sử dụng để đo lường sai lệch giữa giá trị dự đoán và giá trị thực tế. Trong bài toán phân loại, *sparse categorical crossentropy* là một hàm mất mát phổ biến, đặc biệt khi sử dụng các lớp phân loại đa lớp.

**Tối ưu hóa (Optimization)**: *Adam* là một thuật toán tối ưu hóa phổ biến cho mạng nơ-ron. Adam là sự kết hợp của hai thuật toán *Momentum* và *RMSProp,* giúp tăng tốc quá trình huấn luyện bằng cách điều chỉnh tỷ lệ học (learning rate) cho mỗi tham số.

**Hàm kích hoạt (Activation functions)**:

- *ReLU (Rectified Linear Unit):* Thường được sử dụng trong các lớp ẩn, giúp mạng học các quan hệ phi tuyến tính. ReLU có thể giúp mạng học nhanh hơn và giảm vấn đề gradient vanishing.

- *Softmax:* Được sử dụng trong lớp đầu ra để chuyển đổi các giá trị đầu ra thành xác suất phân loại.

# CHƯƠNG 2: TRIỂN KHAI

## THU THẬP DỮ LIỆU LƯU LƯỢNG THỰC

Nhóm đã sử dụng thư viện pyshark để thu thập và phân tích gói tin mạng từ một giao diện mạng nhất định. Chương trình xác định địa chỉ IP cục bộ, tính toán các đặc trưng mạng (như số gói tin, chiều dài gói tin, cờ TCP, thời gian giữa các gói tin) và ghi các thông tin này vào tệp CSV. Chương trình lặp lại quá trình thu thập và ghi lại dữ liệu này liên tục.

## TIỀN XỬ LÝ DỮ LIỆU

Dữ liệu các file theo ngày được nhập lại thành 1 dữ liệu thống nhất. Các xử lý được áp dụng là xóa lặp, xóa các dữ liệu thiếu, xóa các giá trị vô hạn.

Dữ liệu sau đó được lấy mẫu theo số lượng mẫu của tấn công có mẫu nhỏ nhất là 3120 với mỗi mẫu, ta có được 10 loại tấn công với 3120 mẫu mỗi loại tấn công bao gồm: BENIGN(Hành vi bình thường), DoS Hulk, PortScan, DoS GoldenEye, FTP-Patator, DoS slowloris, DoS Slowhttptest, SSH-Patator, Other.

## HUẤN LUYỆN MÔ HÌNH

Dữ liệu sau khi được tiền xử lý được chia thành 2 phần gồm train và test với tỉ lệ 8/2 và được đưa vào huấn luyện lần lượt 3 mô hình đã kể trên

### K-Nearest Neighbors (KNN)

Thư viện sử dụng: Scikit-learn

Khởi tạo mô hình:

* Số lượng hàng xóm gần nhất (k) được đặt là 10.

Quá trình huấn luyện:

* Mô hình KNN được huấn luyện với dữ liệu X\_train và nhãn Y\_train.
* Thời gian huấn luyện được ghi lại để đánh giá hiệu suất:

Kết quả huấn luyện:

* Thời gian huấn luyện: 0.03926873207092285s
* Accuracy: 0.98

***Nhận xét:*** Độ chính xác cao và quá trình huấn luyện ngắn cho thấy mô hình hoạt động hiệu quả, có khả năng phân loại tốt, nhận diện chính xác các lớp trong dữ liệu.

Ma trận nhầm lẫn: cho thấy mô hình dự đoán bị nhầm lẫn giữa các nhãn rất thấp.

A screenshot of a computer screen

Description automatically generated

Hình 4. Ma trận nhầm lẫn huấn luyện KNN

Nhiều nhất chỉ 11-13 lần dự đoán sai

Các thông số khác:

A screenshot of a computer screen

Description automatically generated

Hình 5. Các thông số đánh giá huấn luyện KNN

**Nhận xét:**

*Precision:* Đo lường tỷ lệ giữa các trường hợp dự đoán là dương tính so với tổng số trường hợp dương tính được dự đoán. Mô hình đạt được độ chính xác cao nhất với lớp DDoS (0.997), cho thấy số ít trường hợp bị phân loại sai vào lớp này. Các lớp khác cũng có độ chính xác cao.

*Recall:* Đo lường tỷ lệ giữa số trường hợp dương tính thực sự mà mô hình nhận diện được so với tổng số trường hợp dương tính thực tế. Lớp DDoS đạt recall cao nhất (0.991), trong khi lớp BENIGN có recall thấp nhất (0.923), điều này cho thấy một số trường hợp dương tính thực tế không được nhận diện.

*F1-score:* Là trung bình hài hòa của precision và recall, cung cấp cái nhìn tổng quan về hiệu suất của mô hình. Các lớp như DDoS, DoS GoldenEye, và DoS Hulk có F1-score cao (gần 0.99), cho thấy mô hình hoạt động rất hiệu quả trên những lớp này. Lớp BENIGN có F1-score là 0.95, cho thấy mô hình vẫn hoạt động tốt nhưng có thể cần được cải thiện hơn nữa.

*Support:* Đề cập đến số trường hợp trong mỗi lớp. Tất cả các lớp đều có số lượng mẫu tương đối lớn, cho thấy tập dữ liệu khá cân bằng, đủ để đào tạo mô hình.

Kết quả kiểm thử:

A screenshot of a computer screen

Description automatically generatedMa trận nhầm lẫn:

Hình 6. Ma trận nhầm lẫn kiểm thử KNN

Các thông số khác:

A screenshot of a computer screen

Description automatically generatedHình 7. Thông số đánh giá kiểm thử KNN

Nhận xét: Tương tự như kết quả của quá trình huấn luyện, quá trình kiểm thử cũng đưa ra kết quả cao tương đương.

### Random Forest

Thư viện sử dụng: Scikit-learn

Khởi tạo mô hình:

* Max\_depth=40 – giới hạn độ sâu của từng cây để giảm thiểu khả năng overfitting và tăng tốc quá trình huấn luyện.
* Số cây quyết định: giá trị mặc định 100, đảm bảo mô hình được xây dựng từ nhiều cây để cải thiện độ chính xác.

Quá trình huấn luyện:

* Random Forest được huấn luyện với dữ liệu X\_train và nhãn Y\_train.
* Thời gian huấn luyện được ghi lại để đánh giá hiệu suất

Kết quả huấn luyện:

* Thời gian huấn luyện: 0.99s
* Accuracy: 0.99

Ma trận nhầm lẫn: cho thấy mô hình có độ chính xác rất cao, hầu như không dự đoán sai nhãn.

Hình 8A screenshot of a computer generated image

Description automatically generated. Ma trận nhẫm lẫn huấn luyện Random Forest

Các thông số khác:

A screenshot of a computer screen

Description automatically generated

Hình 9. Thông số đánh giá huấn luyện Random Forest

**Nhận xét:**

* Precision: Hầu hết các lớp đều đạt precision tuyệt đối (1.000), nghĩa là khi mô hình dự đoán một mẫu thuộc lớp đó, dự đoán đó hầu như luôn đúng.
* Recall: Hầu hết các lớp đạt recall hoàn hảo (1.000), cho thấy mô hình có thể nhận diện hầu như toàn bộ các mẫu thuộc mỗi lớp.
* Lớp “BENIGN” có recall là 0.998 chưa hoàn toàn chính xác 100%
* F1-score: Hầu hết các lớp đạt F1-score là 1.000, cho thấy sự cân bằng rất tốt giữa precision và recall.
* Lớp “Other” và “BENIGN” có F1-score lần lượt là 0.999
* Support: Đề cập đến số lượng trường hợp trong mỗi lớp. Tất cả các lớp đều có số lượng mẫu tương đối lớn, cho thấy tập dữ liệu khá cân bằng và đủ để đào tạo mô hình.

Kiểm thử mô hình:

Hình 10A screenshot of a computer generated image

Description automatically generated. Ma trận nhầm lẫn kiểm thử Random Forest

Hình 11A screenshot of a computer screen

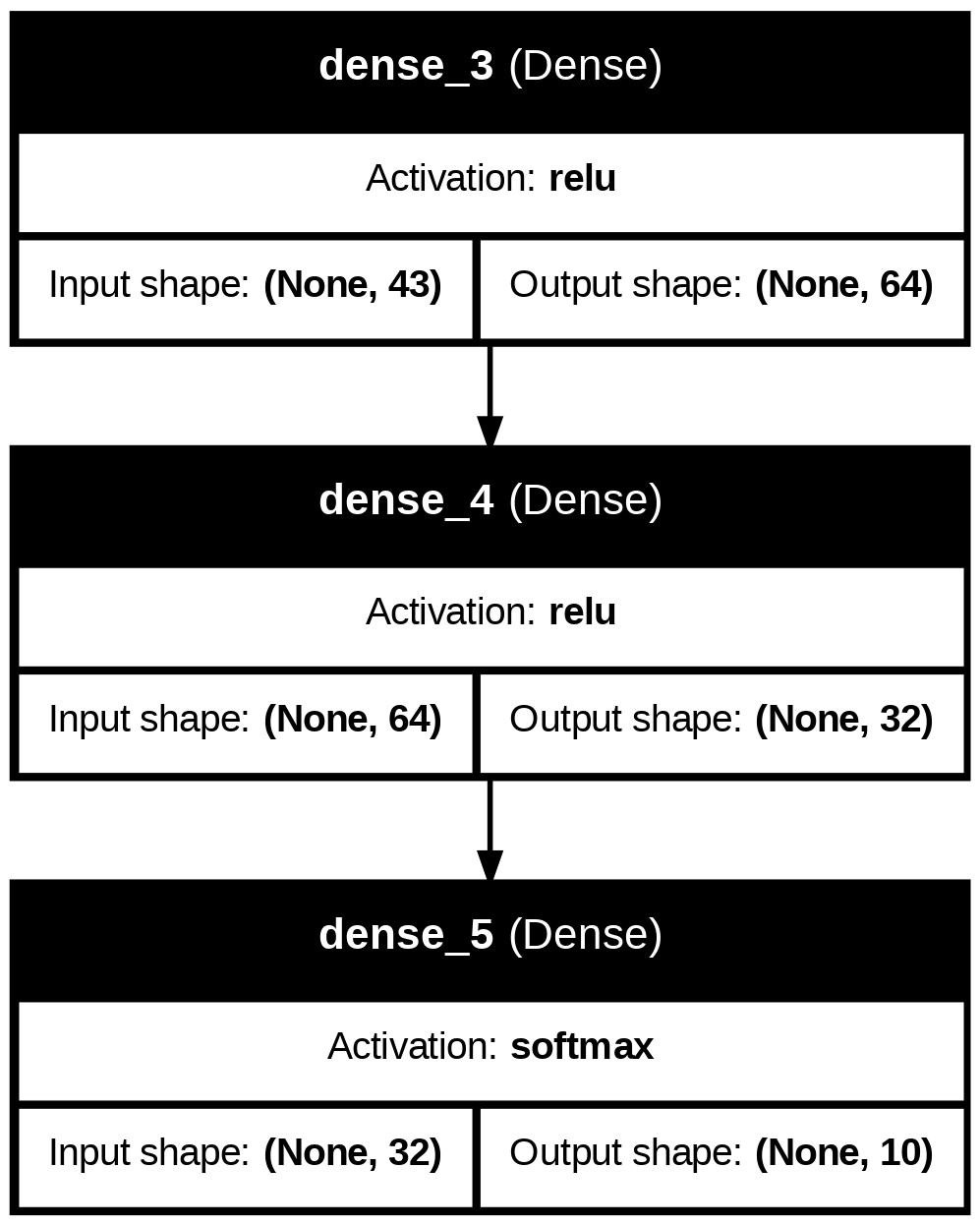
Description automatically generated. Thông số đánh giá kiểm thử Random Forest

### Neural Network Classification

Sau khi xem xét về tài nguyên và độ phù hợp, môi trường và các tham số được nhóm sử dụng để huấn luyện mô hình Neural Network Classification như sau:

* Môi trường và Framework: Colab, Keras, Sklearn
* Loss function: sparse categorical crossentropy
* Optimizer: adam
* Learning rate: 0.001
* Metric: accuracy
* Epochs: 50
* Batch\_size: 32

Kiến trúc mô hình neural network được sử dụng cho bài toán:

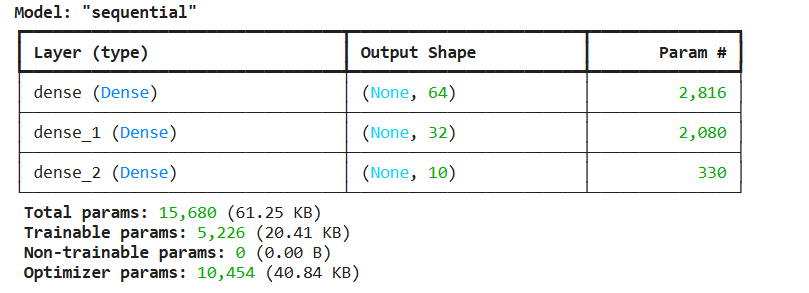


Hình 12. Mô tả Neural Network

Mô hình gồm 3 lớp Dense với các đặc điểm lần lượt như sau:

* *dense\_3:* input\_shape = 43, units = 64, activation: relu
* *dense\_4:* input\_shape = 64, units = 32, activation: relu
* *dense\_5:* input\_shape = 32, units = 10, activation: softmax → đảm nhiệm vai trò phân loại các hành vi bất thường, cụ thể ở đây đầu ra có 10 loại.

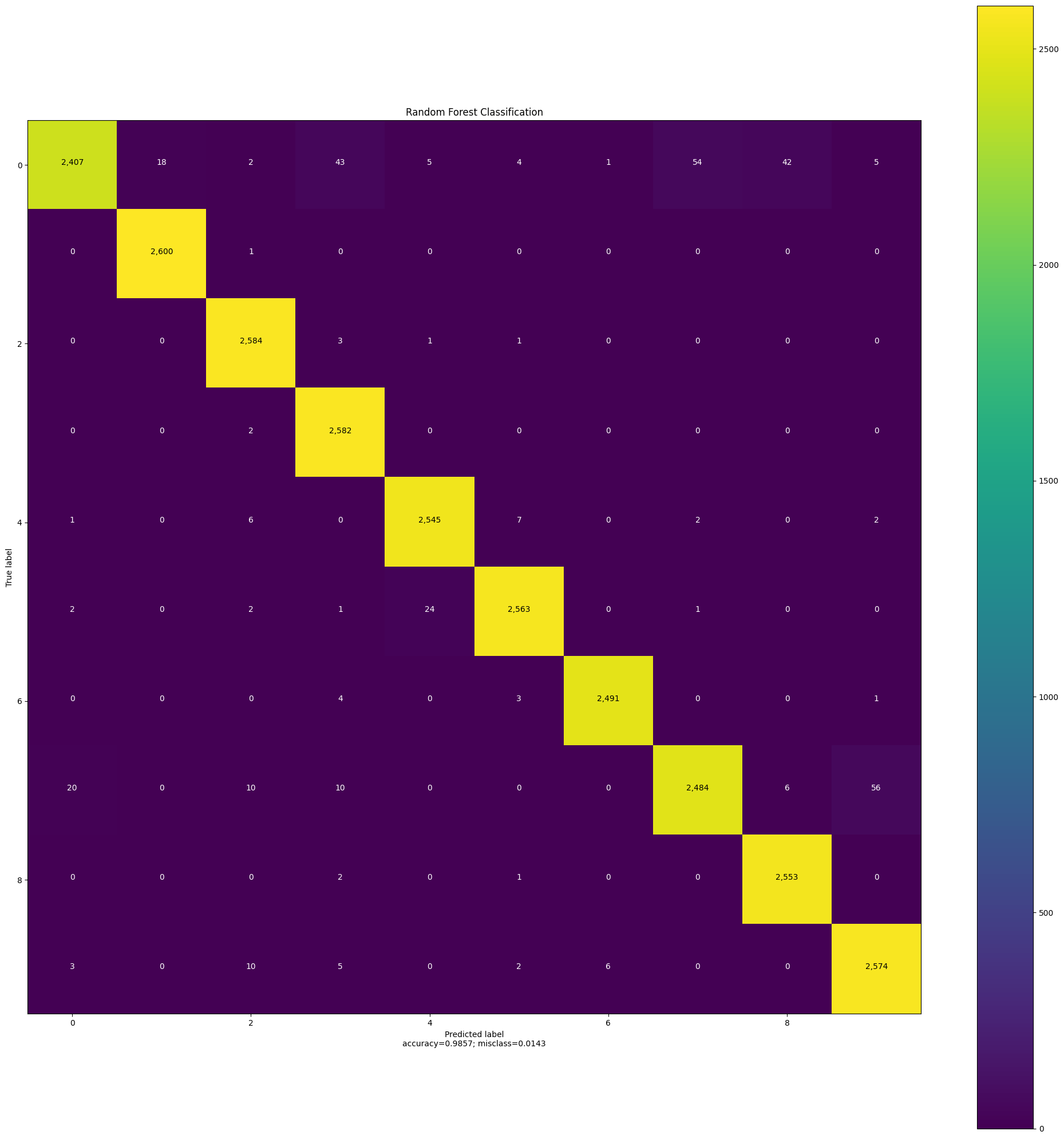
Tổng hợp số lượng tham số của mô hình:



Hình 13. Tham số mô hình Neural Network

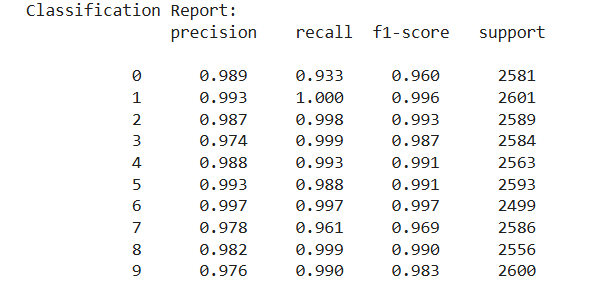
**Kết quả huấn luyện mô hình Neural Network:**

* Confusion matrix trên tập validation của mô hình:



Hình 14. Ma trận nhầm lẫn huấn luyện Neural Network

* Các metrics đánh giá:

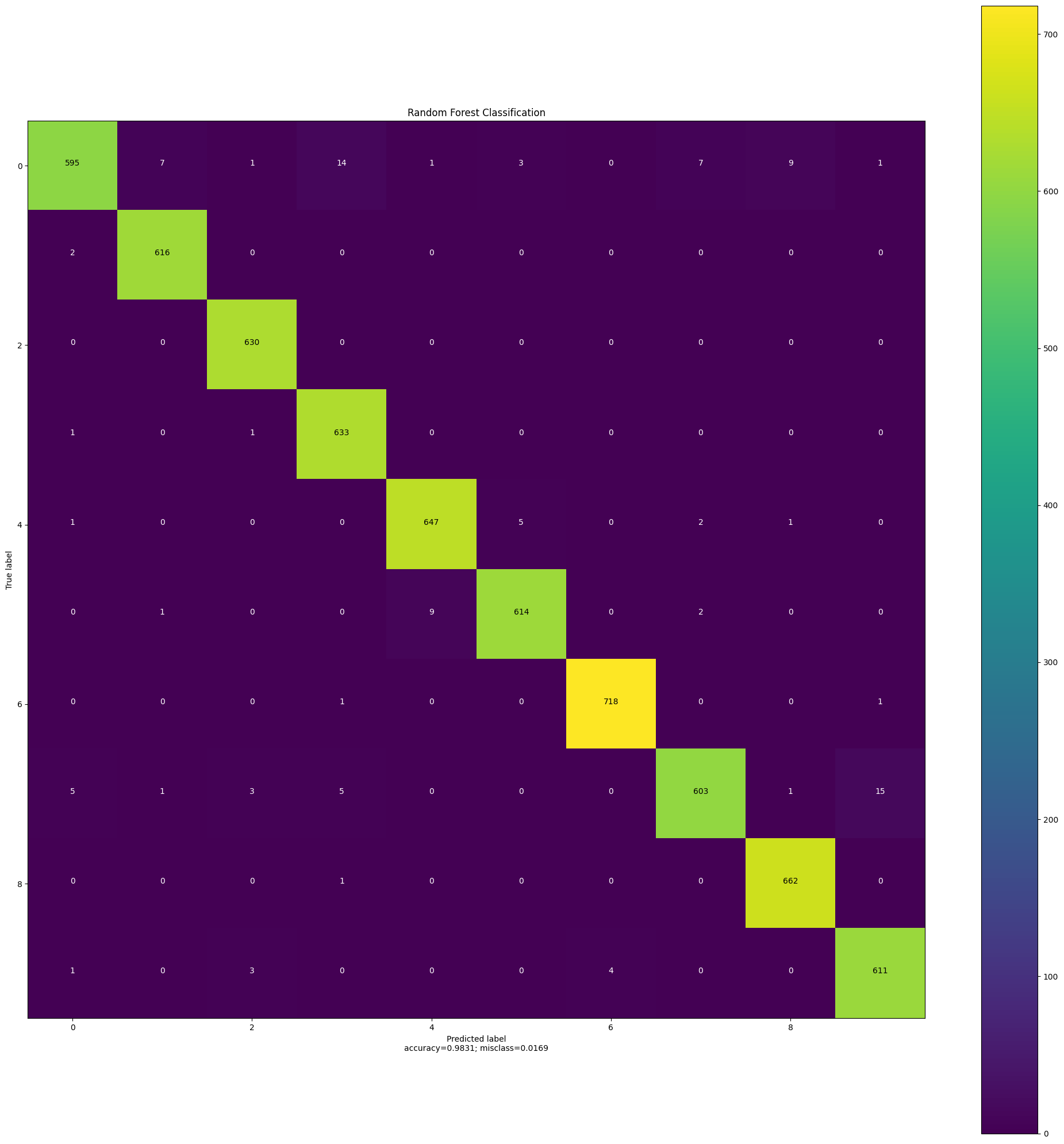


Hình 15. Thông số đánh giá huấn luyện Neural Network

*Nhận xét:*Kết quả trên tập validation cho thấy độ chính xác phân loại cực cao, nhầm lẫn giữa các loại tấn công là cực kỳ nhỏ, trong đó recall cho loại 0 (BENIGN) là thấp nhất 0.933.

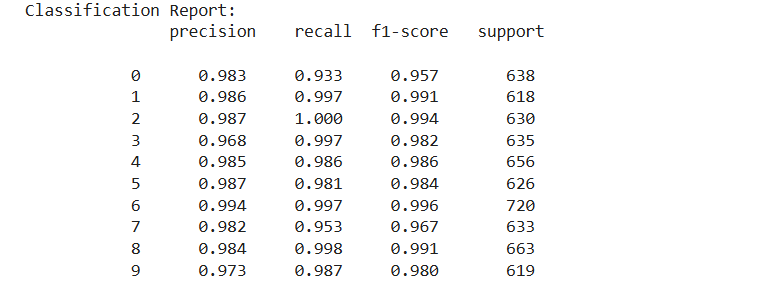
**Kết quả kiểm thử mô hình Neural Network:**

* Confusion matrix trên tập validation của mô hình:



Hình 16. Ma trận nhầm lẫn kiển thử Neural Network

* Các metrics đánh giá:



Hình 17. Thông số đánh giá kiểm thử Neural Network

*Nhận xét:*Kết quả trên tập kiểm thử cho thấy độ chính xác phân loại cực cao, nhầm lẫn giữa các loại tấn công là cực kỳ nhỏ, trong đó recall cho loại 0 (BENIGN) là thấp nhất 0.933.

* + 1. **Tổng hợp kết quả huấn luyện của các mô hình**

|  |  |  |  |
| --- | --- | --- | --- |
| Mô hình | Pecision | Recall | F1-score |
| KNN | 0.983 | 0.983 | 0.982 |
| Random Forest | 0.994 | 0.994 | 0.994 |
| Neural Network | 0.983 | 0.983 | 0.983 |

**Nhận xét:** Kết quả cho thấy mô hình Random Forest độ chính xác cao nhất cho thấy sự hiệu quả của mô hình trên tập dữ liệu

## XÂY DỰNG MODULE BẮT GÓI TIN

Nhóm sử dụng thư viện PyShark hỗ trợ cho việc bắt gói tin. PyShark là một thư viện Python dùng để tương tác với công cụ phân tích gói tin Wireshark, cho phép người dùng bắt và phân tích lưu lượng mạng. PyShark giúp người dùng phân tích dữ liệu gói tin theo thời gian thực hoặc từ các tệp lưu trữ gói tin (PCAP), tận dụng khả năng phân tích mạnh mẽ của Wireshark trong Python. Thư viện này rất hữu ích cho việc xây dựng các công cụ phân tích mạng, tự động hóa việc giám sát lưu lượng và thực hiện các đánh giá an ninh mạng.

Các Tính Năng Chính của PyShark:

* **Bắt Lưu Lượng Trực Tiếp**: PyShark có khả năng bắt lưu lượng mạng theo thời gian thực từ một giao diện mạng, hữu ích cho việc giám sát trực tiếp.
* **Đọc từ Tệp PCAP**: PyShark có thể phân tích các gói tin được lưu trong tệp PCAP, thường được sử dụng cho điều tra và xử lý sự cố mạng.
* **Phân Tích Gói Tin Dễ Dàng**: Mỗi gói tin được phân tích thành một đối tượng Python, giúp dễ dàng truy cập các thông tin chi tiết như địa chỉ IP nguồn và đích, các lớp giao thức (như TCP, UDP, HTTP), và các giá trị trường cụ thể.
* **Hỗ Trợ Nhiều Lớp Giao Thức**: PyShark tự động giải mã nhiều lớp giao thức, cung cấp quyền truy cập đến các trường trong các giao thức như IP, TCP, UDP, ICMP, HTTP, DNS, và nhiều hơn nữa.
* **Lọc và Hiển Thị Cao Cấp**: Người dùng có thể lọc các gói tin dựa trên giao thức, trường cụ thể hoặc điều kiện để cách ly thông tin liên quan từ các tệp lưu trữ lớn.

A screen shot of a computer

Description automatically generatedA screen shot of a computer code

Description automatically generatedKết quả ta có được các thông tin của mỗi gói tin mà máy nhận được:

Hình 18. Mẫu gói tin bắt được

# ĐÁNH GIÁ KẾT QUẢ

## KẾT QUẢ THỰC NGHIỆM

Môi trường:

Máy tấn công: VM Kali với các phương tiện như nmap, hping3

Máy nạn nhân: Window 10 với tưởng lửa và Window Defender bị tắt đi để dễ dàng cho việc thâm nhập

### Trường hợp bình thường

* A screen shot of a computer

  Description automatically generatedMô hình KNN
* A black background with white text

  Description automatically generatedMô hình Random Forest
* A black background with white text

  Description automatically generatedMô hình Neutral Network

Hình 19. Kết quả thực nghiệm mô hình trường hợp bình thường

### Trường hợp bị PortScan với nmap

* A screen shot of a computer

  Description automatically generatedMô hình KNN
* Mô hình Random Forest

A screenshot of a computer code

Description automatically generated

* Mô hình Neutral Network

Hình 20A screen shot of a computer program

Description automatically generated. Kết quả thực nghiệm mô hình trường hợp PortScan

### Trường hợp bị tấn công DDos

* A screen shot of a computer

  Description automatically generatedMô hình KNN
* A screen shot of a computer

  Description automatically generatedMô hình Random Forest
* Mô hình Neutral Network

Hình 21A screen shot of a computer

Description automatically generated. Kết quả thực nghiệm mô hình trường hợp Ddos

## NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ

Tình trạng mà cả ba mô hình đều nhận diện cùng một nhãn cho tất cả các trường hợp thực nghiệm cho thấy rằng các mô hình này đã gặp phải vấn đề nghiêm trọng về overfitting. Overfitting xảy ra khi mô hình quá nhạy cảm với dữ liệu huấn luyện, dẫn đến việc chúng học thuộc lòng các đặc điểm và nhiễu của dữ liệu này, thay vì học được các quy luật tổng quát có thể áp dụng cho dữ liệu mới.

Một số nguyên nhân có thể dẫn đến tình trạng overfitting này có thể bao gồm:

* **Tối ưu hóa mô hình**: Các mô hình có thể chưa được tối ưu hóa đúng cách, dẫn đến việc chúng không đạt được khả năng tổng quát tốt nhất.
* **Hiệu suất của module bắt gói tin**: Nếu module bắt gói tin không hoạt động tốt, nó có thể dẫn đến việc thu thập dữ liệu không đầy đủ hoặc không chính xác. Điều này không chỉ ảnh hưởng đến chất lượng dữ liệu huấn luyện mà còn làm cho các mô hình đọc được thông tin sai lệch, từ đó gây ra sự thiên lệch trong dự đoán.

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. **KIẾN NGHỊ VÀ HƯỚNG PHÁT TRIỂN**

Tối ưu hóa mô hình và tìm hiểu các phương pháp hiệu quả hơn:

* **Tuning siêu tham số (Hyperparameter Tuning)**: Cải thiện các mô hình bằng cách tối ưu hóa các siêu tham số. Việc tìm ra các siêu tham số tối ưu có thể giúp mô hình học được các đặc trưng tổng quát hơn và nâng cao độ chính xác trên tập dữ liệu chưa thấy.
* **Sử dụng các mô hình phức tạp hơn**: Tìm hiểu và thử nghiệm các mô hình học sâu phức tạp hơn như các mạng neural hồi tiếp (RNN), mạng hồi tiếp dài ngắn (LSTM) hoặc Transformer, đặc biệt khi xử lý các dữ liệu dạng chuỗi thời gian, giúp mô hình học được các mối quan hệ lâu dài trong dữ liệu gói tin mạng.
* **Kỹ thuật ensemble**: Kết hợp nhiều mô hình với nhau (bagging, boosting) có thể giúp cải thiện độ chính xác và độ ổn định của dự đoán. Phương pháp này có thể giúp giảm thiểu lỗi của từng mô hình riêng lẻ.

Cải thiện module bắt gói tin và chuyển đổi đặc trưng:

* **Nâng cấp module bắt gói tin**: Tối ưu hóa module bắt gói tin để đảm bảo rằng dữ liệu thu thập được chính xác và đầy đủ, giúp cung cấp thông tin rõ ràng và đáng tin cậy hơn cho mô hình học máy. Việc cải thiện khả năng bắt gói tin sẽ giúp loại bỏ các sai sót trong quá trình thu thập, giảm thiểu nhiễu và các lỗi không mong muốn trong dữ liệu.
* **Tăng cường dữ liệu**: Tạo ra các phiên bản dữ liệu mạng khác nhau thông qua các phương pháp như thêm nhiễu vào dữ liệu, hoặc áp dụng các phương pháp tăng cường như SMOTE (Synthetic Minority Over-sampling Technique) để cân bằng dữ liệu giữa các lớp.

**Kết luận**: Với những cải tiến trên, nhóm có thể nâng cao hiệu suất của các mô hình, đảm bảo rằng mô hình không chỉ hoạt động tốt trên dữ liệu huấn luyện mà còn có khả năng tổng quát tốt khi đối mặt với các tình huống thực tế. Đồng thời, việc tối ưu hóa các phương pháp thu thập dữ liệu và xử lý đặc trưng sẽ giúp cung cấp những đầu vào chính xác và có giá trị cho các mô hình, từ đó nâng cao độ tin cậy của các kết quả phân tích.

**TÀI LIỆU THAM KHẢO**

1. *Towards Data Science. (2020). K-Nearest Neighbors Algorithm: Intuition, Implementation, and Application.* [*https://towardsdatascience.com/k-nearest-neighbors-algorithm-intuition-implementation-and-application-bfcaebf4d7e8*](https://towardsdatascience.com/k-nearest-neighbors-algorithm-intuition-implementation-and-application-bfcaebf4d7e8)
2. *Scikit-learn Documentation. (n.d.). Random Forest Classifier.* [*https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html*](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)
3. *Towards Data Science. (2020). A Brief Introduction to Neural Networks in Python.* [*https://towardsdatascience.com/a-brief-introduction-to-neural-networks-in-python-cf47a03a6f38*](https://towardsdatascience.com/a-brief-introduction-to-neural-networks-in-python-cf47a03a6f38)
4. *Shaghi, N., & Mahmoudi, M. (2017). CICIDS 2017: A Data Set for Intrusion Detection and Classification. Canadian Institute for Cybersecurity.* [*https://www.unb.ca/cic/datasets/malmem-2020.html*](https://www.unb.ca/cic/datasets/malmem-2020.html)