BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC SINH VIÊN NĂM HỌC 2024**

**ỨNG DỤNG GIẢI THUẬT HỌC CHUYỂN GIAO CHO BÀI TOÁN PHÂN LOẠI LƯU LƯỢNG MẠNG MÃ HÓA TRONG MẠNG SDN ĐA MIỀN**

**<Mã số: CNTT-2024-04>**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| NGUYỄN HUY HIỆU | 65CS2 | 1520065 |
| HOÀNG QUANG DUY | 67CS1 | 0190267 |
| NGUYỄN XUÂN PHÚC | 67CS1 | 0182467 |
| ĐẶNG HOÀNG HẢI | 67CS1 | 0047167 |
| TRIỆU QUỐC ĐẠT | 67IT1 | 0004467 |

**Giáo viên hướng dẫn:** HOÀNG NAM THẮNG

**Hà Nội, 06/2024**

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC SINH VIÊN NĂM HỌC 2024**

**ỨNG DỤNG GIẢI THUẬT HỌC CHUYỂN GIAO CHO BÀI TOÁN PHÂN LOẠI LƯU LƯỢNG MẠNG MÃ HÓA TRONG MẠNG SDN ĐA MIỀN**

**<Mã số: CNTT-2024-04>**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| NGUYỄN HUY HIỆU | 65CS2 | 1520065 |
| HOÀNG QUANG DUY | 67CS1 | 0190267 |
| NGUYỄN XUÂN PHÚC | 67CS1 | 0182467 |
| ĐẶNG HOÀNG HẢI | 67CS1 | 0047167 |
| TRIỆU QUỐC ĐẠT | 67IT1 | 0004467 |

**Giáo viên hướng dẫn:** HOÀNG NAM THẮNG

|  |  |
| --- | --- |
| **Cán bộ hướng dẫn**  (Ký, họ tên) | **Sinh viên trưởng nhóm**  (Ký, họ tên) |

**Hà Nội, 06/2024**

**MỤC LỤC**

[TÓM TẮT 3](#_Toc169566383)

[LỜI CẢM ƠN 4](#_Toc169566384)

[LỜI CAM ĐOAN 5](#_Toc169566385)

[CHƯƠNG 1 : PHÂN LOẠI LƯU LƯỢNG MẠNG MÃ HÓA 6](#_Toc169566386)

[**1.1 Phân loại lưu lượng mạng (Network Traffic Classification)** 6](#_Toc169566387)

[**1.1.1 Định nghĩa lưu lượng mạng** 6](#_Toc169566388)

[**1.1.2 Phân loại lưu lượng mạng** 7](#_Toc169566389)

[**1.1.3 Các phương pháp phân loại lưu lượng mạng** 9](#_Toc169566390)

[**1.2 Phân loại lưu lượng mạng mã hóa (Encrypted Traffic Classification)** 10](#_Toc169566391)

[**1.2.1 Định nghĩa** 10](#_Toc169566392)

[**1.2.2 Điểm khác biệt với phân loại lưu lượng mạng thông thường** 10](#_Toc169566393)

[**1.2.3 Vai trò của phân loại lưu lượng mạng mã hóa** 10](#_Toc169566394)

[**1.2.4 Thách thức của phân loại lưu lượng mạng mã hóa** 12](#_Toc169566395)

[**1.2.5 Các phương pháp phân loại lưu lượng mạng mã hóa** 12](#_Toc169566396)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 15](#_Toc169566397)

[**2.1 Mạng định nghĩa bằng tri thức (Knowledge-Defined Networking)** 15](#_Toc169566398)

[**2.2 Tổng quan về mạng KDN** 16](#_Toc169566399)

[**2.3 Khảo sát một số phương pháp phân loại lưu lượng mạng trên mạng SDN** 18](#_Toc169566400)

[**2.4 Học chuyển giao trong bài toán phân loại lưu lượng mạng** 19](#_Toc169566401)

[CHƯƠNG 3 : PHƯƠNG PHÁP TRANSFER LEARNING VÀ THUẬT TOÁN TRADABOOST 20](#_Toc169566402)

[**3.1 Tìm hiểu về phương pháp Transfer Learning** 20](#_Toc169566403)

[**3.1.1 Khái niệm Transfer Learning** 20](#_Toc169566404)

[**3.1.2 Ưu điểm** 20](#_Toc169566405)

[**3.1.3 Nhược điểm** 21](#_Toc169566406)

[CHƯƠNG 4 : THUẬT TOÁN ĐỀ XUẤT 26](#_Toc169566407)

[**4.1 Mô tả bài toán** 26](#_Toc169566408)

[CHƯƠNG 5 : THỰC NGHIỆM 32](#_Toc169566409)

[**5.1 Bộ dữ liệu thực nghiệm** 32](#_Toc169566410)

[**5.1.1 Cách thu thập dữ liệu** 32](#_Toc169566411)

[**5.1.2 Tiền xử lý dữ liệu** 33](#_Toc169566412)

[**5.1.3 Mô tả dữ liệu** 34](#_Toc169566413)

[**5.2 Các độ đo và cấu hình thực nghiệm** 36](#_Toc169566414)

[**5.2.1 Giới thiệu độ đo được sử dụng** 36](#_Toc169566415)

[**5.2.2 Cấu hình thực nghiệm** 37](#_Toc169566416)

[**5.3 Kết quả thực nghiệm** 38](#_Toc169566417)

[**5.3.1 Lựa chọn tham số đầu vào tối ưu** 38](#_Toc169566418)

[**5.3.2 Lựa chọn mô hình CNN tối ưu** 39](#_Toc169566419)

[**5.3.3 Đánh giá so sánh các thuật toán:** 40](#_Toc169566420)

[KẾT LUẬN VÀ KIẾN NGHỊ 42](#_Toc169566421)

[PHỤ LỤC 43](#_Toc169566422)

[**6.1 Wireshark và Tshark** 43](#_Toc169566423)

[**6.2 Tensorflow** 44](#_Toc169566424)

[Tài liệu tham khảo 45](#_Toc169566425)

# **TÓM TẮT**

Vấn đề phân loại lưu lượng đã được nghiên cứu rộng rãi trong gần 20 năm qua do tầm quan trọng của nó trong nhiều khía cạnh, bao gồm cải thiện giám sát hiệu suất, tối ưu hóa băng thông, nâng cao chất lượng dịch vụ (QoS), lập kế hoạch và mở rộng năng lực, xử lý sự cố và chẩn đoán, thực thi chính sách, tối ưu hóa tài nguyên, thanh toán và kế toán, cũng như dự báo và phân tích xu hướng.

Gần đây, việc phân loại lưu lượng mạng đã gặp phải hai thách thức chính. Thứ nhất, việc sử dụng phổ biến lưu lượng mã hóa do sự bùng nổ của các thiết bị di động đã đóng vai trò quan trọng trong việc bảo vệ quyền riêng tư và ẩn danh của người dùng trực tuyến. Tuy nhiên, sự gia tăng lưu lượng mã hóa này cũng dẫn đến sự phức tạp của các ứng dụng và giao thức ẩn, khiến việc phát hiện và phân loại chính xác trở thành một nhiệm vụ khó khăn. Thứ hai, Internet hiện đang trải qua một sự chuyển đổi sang các hệ thống mạng tự động với các tài nguyên phân tán và sự kiểm soát tập trung. Các hệ thống mạng hiện đại được phân chia thành các miền độc lập và đa dạng, tất cả đều được quản lý tập trung bởi một bộ điều khiển theo mô hình Mạng Định nghĩa Bằng Phần Mềm (SDN). Môi trường mạng phi tập trung và không đồng nhất này đặt ra một thách thức đáng kể trong việc phân loại lưu lượng mạng.

Hai thách thức này đã khiến các phương pháp phân loại lưu lượng truyền thống, chẳng hạn như kỹ thuật dựa trên cổng và dựa trên tải trọng, trở nên không hiệu quả. Do đó, đã có một sự chuyển đổi đáng chú ý sang việc sử dụng các phương pháp phân loại dịch vụ dựa trên Học Máy (ML). Sự tiến bộ đáng kể trong các thuật toán ML, đặc biệt là những thuật toán sử dụng học bán giám sát, đã thúc đẩy sự phổ biến của các phương pháp này. Các nghiên cứu trước đây trong lĩnh vực này chủ yếu tập trung vào việc tận dụng cả các thuật toán ML truyền thống và học sâu, mang lại những kết quả hứa hẹn. Tuy nhiên, các phương pháp này chưa tính đến tính chất động của các hệ thống mạng, nơi các miền mới liên tục xuất hiện với tập dữ liệu hạn chế. Sự khan hiếm dữ liệu này đặt ra một thách thức trong việc huấn luyện các mô hình học máy đủ tốt cho các miền mới nổi này.

Để giải quyết bài toán này, chúng tôi sẽ áp dụng kỹ thuật học chuyển giao (Transer Learning) với việc đề xuất một thuật toán học chuyển giao mới. Thuật toán của chúng tôi đề xuất, thuật toán Multi-Class TrAdaBoost-CNN được sử dụng để vượt qua các khó khăn về sự thiếu hụt dữ liệu cũng như mất cần bằng phân phối dữ liệu trên miền đích. Các kết quả thực nghiệm trên bộ dữ liệu thực cho thấy thuật toán của chúng tôi có thể cải thiện độ chính xác của mô hình sau khi được huấn luyện chuyển giao lên đến 16% so với các phương pháp thông thường, kể cả khi gặp phải sự thiếu hụt lớn về dữ liệu.

# **LỜI CẢM ƠN**

Lời đầu tiên, chúng tôi xin gửi gửi lời cảm ơn chân thành và sâu sắc đến thầy ThS. Hoàng Nam Thắng - Giảng viên hướng dẫn, đã quan tâm hướng dẫn tận tình chúng tôi trong suốt thời gian thực hiện đề tài lần này.

Cuối cùng, chúng tôi xin cảm kích các thầy cô thuộc khoa *CÔNG NGHỆ THÔNG TIN,* *TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI* đã trang bị kiến thức quý báu trong suốt thời gian học tập, tạo mọi điều kiện thuận lợi cho chúng tôi được hoàn thành bài báo cáo khoa học lần này.

Chúng tôi xin trân thành cảm ơn.

Hà Nội, ngày 18 tháng 6 năm 2024

Nhóm sinh viên

# **LỜI CAM ĐOAN**

Trong quá trình làm báo cáo này chúng tôi đã đọc và tham khảo rất nhiều tài liệu khác nhau từ giáo trình, sách tham khảo cho tới rất nhiều các bài báo uy tín được đăng tải trong và ngoài nước. **Chúng tôi xin cam đoan những gì chúng tôi viết dưới đây là hoàn toàn chính thống, chân thực. Những kết quả đo đạc mô phỏng đã đạt được trong khóa luận không sao chép từ bất kì tài liệu nào dưới mọi hình thức.** Những kết quả đó là những gì chúng tôi đã nghiên cứu, tích lũy được trong quá trình làm khóa luận này. Các tài liệu trích dẫn là chính xác và ghi rõ nguồn gốc.

Chúng tôi xin hoàn toàn chịu trách nhiệm nếu có bất kì dấu hiệu nào của sự sao chép kết quả từ các tài liệu khác.

Hà Nội, ngày 18 tháng 6 năm 2024

Nhóm sinh viên

# **CHƯƠNG 1 : PHÂN LOẠI LƯU LƯỢNG MẠNG MÃ HÓA**

## **1.1 Phân loại lưu lượng mạng (Network Traffic Classification)**

### **1.1.1 Định nghĩa lưu lượng mạng**

**"Lưu lượng mạng"** : thuật ngữ dùng để chỉ dữ liệu và thông tin được truyền tải qua mạng máy tính từ một điểm tới điểm khác. Điều này bao gồm mọi thứ từ các gói tin dữ liệu, tin nhắn email, tệp tin, đến các loại dịch vụ trực tuyến như web browsing, streaming video, game online, và các ứng dụng khác. Dữ liệu này được truyền qua các thiết bị mạng như router, switch, máy chủ, và các thiết bị kết nối khác [1].

Vì cấu trúc liên kết của Internet không phân cấp, nên không có điểm đo duy nhất nào cho tổng lưu lượng truy cập Internet. Dữ liệu lưu lượng truy cập được lấy từ các điểm trao đổi lưu lượng ngang hàng của các mạng giao thức Internet Cấp 1 (tier 1 network) cho các chỉ dẫn về khối lượng và tăng trưởng. Tuy nhiên, dữ liệu đó không bao gồm lưu lượng truy cập trong mạng của một nhà cung cấp dịch vụ Internet duy nhất cũng như lưu lượng đi qua các điểm tiên phong riêng tư.

**Ví dụ:**

1. Khi bạn truy cập vào một trang web như Facebook, máy tính của bạn gửi yêu cầu đến máy chủ của Facebook thông qua mạng. Yêu cầu này được mã hóa thành gói tin dữ liệu và được gửi đi thông qua mạng Internet. Sau đó, máy chủ của Facebook nhận yêu cầu, xử lý nó và gửi lại dữ liệu cần thiết cho bạn, chẳng hạn như trang web Facebook. Dữ liệu này sau đó được truyền lại qua mạng Internet và được máy tính của bạn nhận và hiển thị trên trình duyệt web.

2. Bạn gửi một email cho một người khác, email của bạn cũng được chuyển thành gói tin dữ liệu và được gửi qua mạng Internet đến máy chủ email của người nhận. Máy chủ email này sau đó nhận và lưu trữ email của bạn cho đến khi người nhận đăng nhập vào hộp thư đến của mình để đọc.

Tất cả các hoạt động trên mạng này đều tạo ra lưu lượng mạng.

Truyền phát video đang đứng đầu trên thế giới về lượng lưu lượng truy cập Internet và video chiếm hơn 60% lưu lượng tải về trên toàn thế giới. Các thể loại video được truyền phát thông qua Internet trở nên đa dạng hơn nhờ vào số lượng các nhà cung cấp đang ngày càng tăng lên. HTTP Media Stream chiếm 23,8% lưu lượng phát trực tuyến video toàn cầu, dẫn trước 2 nhà cung cấp lớn khác là Netflix và YouTube. Từ năm 2018 đến 2019, tỷ lệ lưu lượng chia sẻ tệp tăng khoảng 50% trong khi tỷ lệ lưu lượng tải lên tăng 35%. Lưu lượng BitTorrent, chiếm phần lớn nhất trong tất cả lưu lượng chia sẻ tệp, đã tăng chủ yếu ở các khu vực EMEA (Châu Âu, Trung Đông và Châu Phi) và APAC (Châu Á-Thái Bình Dương).

**1.1.2 Phân loại lưu lượng mạng**

**"Phân loại lưu lượng mạng"**: quá trình nhận diện, phân biệt và nhóm các loại dữ liệu được truyền tải qua mạng máy tính dựa trên các đặc điểm cụ thể như giao thức, ứng dụng, địa chỉ nguồn và đích, cũng như các thuộc tính khác như kích thước gói tin. Quá trình này cung cấp cái nhìn tổng quan về cách mà dữ liệu di chuyển và được sử dụng trên mạng, từ đó giúp tổ chức hiểu rõ hơn về mô hình lưu lượng mạng của mình và áp dụng các biện pháp quản lý, kiểm soát và bảo mật phù hợp.

Trong lĩnh vực phân loại lưu lượng Internet, những phương pháp truyền thống có một số hạn chế nhất định. Đầu tiên, đánh dấu gói (packet marking) được đề xuất để phân biệt lưu lượng dựa trên lớp QoS của nó. Một số ví dụ về các trường được sử dụng để đánh dấu gói là Loại dịch vụ (Type of Sevice - ToS), Điểm mã dịch vụ phân biệt (Differentiated Services Code Point - DSCP) và Thông báo tắc nghẽn rõ ràng (Explicit Congestion Notification - ECN). Từ đấy, một số giao thức đã được đề xuất để phân loại lưu lượng bao gồm Dịch vụ khác biệt (DiffServ), Dịch vụ tích hợp (IntServ) và Chuyển mạch nhãn đa giao thức (MPLS). Tuy nhiên, các giao thức này không được triển khai và sử dụng một cách rộng rãi do sự phức tạp và các vấn đề tương thích với hệ thống của chúng.

**Việc phân loại này giúp các nhà quản lý mạng :**

1. Hiểu rõ hơn về mô hình lưu lượng: Nhận biết được dạng lưu lượng nào chiếm ưu thế trong mạng, từ đó có thể dự đoán và quản lý tài nguyên mạng một cách tốt hơn.

2. Ưu tiên lưu lượng: Một số loại lưu lượng như VoIP và video conferencing cần được ưu tiên hơn so với các loại lưu lượng khác để đảm bảo chất lượng dịch vụ (QoS). Phân loại lưu lượng cho phép thiết lập các chính sách ưu tiên phù hợp.

Trong phân loại lưu lượng mạng, các gói tin dữ liệu được phân loại dựa trên các thuộc tính như giao thức, ứng dụng hoặc địa chỉ IP. Tương tự, trong phân luồng giao thông, các loại phương tiện như xe cứu thương, xe cứu hỏa hoặc xe công cộng được phân loại dựa trên mục đích và ưu tiên trong việc sử dụng đường phố.

A group of emergency vehicles

Description automatically generated

**Hình 1-1**: Các loại xe ưu tiên

3. An ninh mạng: Phát hiện và ngăn chặn các hoạt động độc hại hoặc không mong muốn bằng cách phân biệt lưu lượng hợp lệ với lưu lượng độc hại như virus, worm, hoặc các cuộc tấn công DDoS.

4. Thực thi chính sách: Kiểm soát việc sử dụng mạng bằng cách hạn chế hoặc ngăn chặn truy cập đến các dịch vụ không được phép hoặc không phù hợp với chính sách của tổ chức.

### **1.1.3 Các phương pháp phân loại lưu lượng mạng**

1. Phân loại dựa trên Cổng (Port-based Classification):

- Đây là cách đơn giản nhất, dựa vào cổng TCP/UDP mà gói tin sử dụng để xác định loại ứng dụng hoặc dịch vụ.

- Ví dụ, cổng 80 thường được dùng cho HTTP, trong khi cổng 443 dành cho HTTPS.

- Tuy nhiên, phương pháp này không còn chính xác hoàn toàn do nhiều ứng dụng hiện nay có thể thay đổi cổng động hoặc sử dụng các cổng chung để né tránh sự phát hiện.

2. Phân loại dựa trên Gói tin (Packet Inspection):

- Deep Packet Inspection (DPI): Phân tích nội dung gói tin để xác định ứng dụng hoặc dịch vụ. Phương pháp này có thể cung cấp thông tin chính xác nhưng đòi hỏi tài nguyên lớn và có thể vi phạm quyền riêng tư.

- Shallow Packet Inspection (SPI): Kiểm tra các phần header của gói tin mà không cần phải phân tích sâu vào nội dung, giảm bớt vấn đề về quyền riêng tư và yêu cầu về tài nguyên.

3. Phân loại dựa trên Hành vi (Behavior-based Classification):

- Phân tích mô hình lưu lượng mạng, như tần suất, độ lớn của gói tin, và thời gian truyền để xác định loại ứng dụng.

- Ví dụ, lưu lượng VoIP có mẫu đặc trưng khác biệt so với lưu lượng web browsing.

4. Phân loại dựa trên Đặc điểm (Feature-based Classification):

- Sử dụng các đặc điểm cụ thể của lưu lượng, như tỉ lệ bit, thời gian tồn tại của luồng, và số lượng gói tin để phân loại.

- Cách này thường yêu cầu việc phân tích thống kê và xử lý dữ liệu nâng cao.

Mỗi phương pháp phân loại có ưu nhược điểm và áp dụng tốt nhất trong các tình huống cụ thể. Lựa chọn phương pháp phụ thuộc vào yêu cầu cụ thể về chính xác, tốc độ xử lý, và quyền riêng tư cũng như tài nguyên sẵn có.

## **1.2 Phân loại lưu lượng mạng mã hóa (Encrypted Traffic Classification)**

### **1.2.1 Định nghĩa**

Phân loại lưu lượng mạng mã hóa là quá trình phân loại các gói tin dữ liệu mạng được mã hóa thành các loại lưu lượng khác nhau dựa trên các đặc điểm nội dung của gói tin mà không cần phải giải mã hoặc xử lý giải mã dữ liệu. Điều này làm cho việc phân loại trở nên khó khăn hơn so với phân loại lưu lượng mạng thông thường, nơi các gói tin có thể được phân loại dễ dàng bằng cách phân tích các đặc điểm không gian địa chỉ, cổng, hoặc header.

### **1.2.2 Điểm khác biệt với phân loại lưu lượng mạng thông thường**

**Điểm khác biệt chính** giữa phân loại lưu lượng mạng mã hóa và phân loại lưu lượng mạng thông thường là ở khả năng phân loại dựa trên nội dung của gói tin. Trong phân loại lưu lượng mạng thông thường, thông tin về loại dịch vụ hoặc ứng dụng thường có sẵn trong header của gói tin, cho phép các phương pháp phân loại dựa trên thông tin này. Tuy nhiên, khi dữ liệu được mã hóa, thông tin này không còn có sẵn trong header mà được bảo vệ bởi mã hóa.

Do đó, trong phân loại lưu lượng mạng mã hóa, các phương pháp phân loại phải dựa vào các đặc điểm khác của gói tin, như kích thước gói tin, tần suất truy cập, phân bố thời gian của gói tin, hoặc các đặc điểm thống kê khác của lưu lượng mạng. Các phương pháp như học máy và phân tích thống kê có thể được sử dụng để phát triển các mô hình phân loại có khả năng xử lý lưu lượng mạng mã hóa. Tuy nhiên, độ chính xác của các phương pháp này thường thấp hơn so với phân loại lưu lượng mạng thông thường do sự hạn chế của việc trích xuất thông tin từ dữ liệu đã được mã hóa.

### **1.2.3 Vai trò của phân loại lưu lượng mạng mã hóa**

1. Bảo vệ dữ liệu nhạy cảm

- Nhằm đảm bảo an toàn cho dữ liệu nhạy cảm khi được truyền tải qua mạng bằng cách xác định, phân loại và bảo vệ các loại dữ liệu được mã hóa.

- Ví dụ: Một công ty tài chính sử dụng phân loại lưu lượng mạng mã hóa để xác định và bảo vệ thông tin tài chính của khách hàng khi chúng được truyền tải qua mạng.

2. Phát hiện và ngăn chặn các mối đe dọa bảo mật

- Nhấn mạnh vào việc phát hiện và ngăn chặn các cuộc tấn công hoặc mối đe dọa bảo mật được ẩn dưới dạng dữ liệu mã hóa.

- Ví dụ: Một hệ thống phân loại lưu lượng mạng mã hóa có khả năng phát hiện và ngăn chặn các cuộc tấn công mạng như tấn công phá hoại dịch vụ (DoS) mà sử dụng dữ liệu mã hóa để che giấu.

3. Quản lý rủi ro và tuân thủ quy định

- Liên quan đến việc đánh giá và quản lý rủi ro liên quan đến việc truyền tải dữ liệu mã hóa và đảm bảo rằng các quy định và chuẩn mạng được tuân thủ.

- Ví dụ: Một tổ chức y tế sử dụng phân loại lưu lượng mạng mã hóa để đảm bảo rằng dữ liệu y tế cá nhân của bệnh nhân được truyền tải qua mạng một cách an toàn và tuân thủ các quy định HIPAA.

4. Giám sát và phản ứng nhanh

- Bao gồm việc giám sát hoạt động mạng mã hóa và phản ứng nhanh đối với bất kỳ hoạt động bất thường hoặc đe dọa được phát hiện.

- Ví dụ: Một tổ chức tài chính sử dụng hệ thống phân loại lưu lượng mạng mã hóa để giám sát các giao dịch tài chính và phản ứng nhanh đối với bất kỳ giao dịch bất thường nào có thể biểu hiện một cuộc tấn công hoặc lừa đảo.

5. Tối ưu hóa hiệu suất mạng

- Nhằm cải thiện hiệu suất và tối ưu hóa việc sử dụng tài nguyên mạng bằng cách ưu tiên và quản lý lưu lượng dữ liệu mã hóa.

- Ví dụ: Một nhà cung cấp dịch vụ internet sử dụng phân loại lưu lượng mạng mã hóa để ưu tiên dữ liệu quan trọng như truyền phim trực tuyến HD hoặc truy cập an toàn vào ngân hàng trực tuyến.

### **1.2.4 Thách thức của phân loại lưu lượng mạng mã hóa**

Bài toán phân loại lưu lượng mạng mã hóa đối diện với nhiều thách thức :

1. Mất thông tin: Mã hóa dữ liệu khiến thông tin trong gói tin trở nên không truy cập được mà không cần phải giải mã, làm giảm khả năng phân loại dựa trên nội dung của gói tin.

2. Tính ẩn danh: Việc mã hóa làm cho nội dung của gói tin trở nên không đọc được, làm giảm khả năng xác định loại ứng dụng hoặc dịch vụ được sử dụng.

3. Tính đồng nhất: Các gói tin mã hóa thường có xu hướng trông giống nhau, khiến cho các mẫu lưu lượng khác nhau trở nên khó phân biệt.

4. Kỹ thuật chống phát hiện: Những người gửi có thể sử dụng các kỹ thuật để làm cho dữ liệu mã hóa trông giống như dữ liệu không mã hóa, làm cho việc phát hiện và phân loại trở nên khó khăn hơn.

5. Sự gia tăng của lưu lượng mã hóa: Sự phổ biến của việc sử dụng các giao thức mã hóa như SSL/TLS đã làm tăng lượng dữ liệu mã hóa, làm cho việc phân loại trở nên ngày càng khó khăn hơn.

Bài toán phân loại lưu lượng mạng mã hóa đối mặt với các thách thức đặc biệt, đặt ra thêm các thách thức đối với việc xử lý dữ liệu đã được mã hóa và không thể truy cập được.

### **1.2.5 Các phương pháp phân loại lưu lượng mạng mã hóa**

Có một số phương pháp tiếp cận để giải quyết bài toán phân loại lưu lượng mạng mã hóa, mỗi phương pháp có những ưu điểm và nhược điểm riêng. Dưới đây là một số phương pháp phổ biến:

1. Phân tích thống kê

**Ưu điểm:**

- Không yêu cầu giải mã dữ liệu: Phân tích thống kê có thể thực hiện trên dữ liệu đã được mã hóa mà không cần phải giải mã, do đó bảo vệ sự riêng tư của người dùng.

- Dễ triển khai: Phương pháp này thường dễ triển khai và có thể được thực hiện trên cả dữ liệu lưu lượng lớn.

**Nhược điểm:**

- Độ chính xác thấp: Phân tích thống kê có thể không hiệu quả đối với lưu lượng mạng phức tạp và đa dạng.

- Dễ bị lừa: Các kẻ tấn công có thể dễ dàng tạo ra các mẫu lưu lượng giả mạo để lừa hệ thống phân loại.

**Ví dụ**: Một tổ chức muốn xác định các hoạt động mạng bất thường trong mạng nội bộ của họ mà không cần phải giải mã dữ liệu. Họ có thể sử dụng phân tích thống kê để xác định các biểu hiện không bình thường, như lưu lượng mạng ngoại vi cao đột ngột hoặc mẫu lưu lượng không thường xuyên.

2. Sử dụng mạng nơ-ron tích hợp chập (CNN)

**Ưu điểm:**

- Khả năng học mẫu phức tạp: CNNs có khả năng học được các đặc trưng phức tạp từ dữ liệu mã hóa.

- Đa dạng và linh hoạt: Có thể tinh chỉnh các kiến trúc mạng để phù hợp với loại dữ liệu cụ thể và cải thiện hiệu suất.

**Nhược điểm:**

- Yêu cầu tài nguyên cao: Huấn luyện một mạng CNN phức tạp có thể yêu cầu nhiều tài nguyên tính toán và thời gian.

- Cần có dữ liệu lớn và đa dạng: Đòi hỏi một lượng lớn dữ liệu đã được gán nhãn để huấn luyện mạng CNN.

**Ví dụ**: Một công ty bảo mật muốn xây dựng một hệ thống phát hiện xâm nhập có khả năng phát hiện các mẫu lưu lượng mạng bất thường. Họ có thể sử dụng mạng CNNs để học và nhận diện các biểu hiện của các cuộc tấn công mạng, như lưu lượng mã hóa không bình thường được gửi từ một máy chủ ngoài vào mạng nội bộ.

3. Sử dụng Machine Learning và Deep Learning

**Ưu điểm:**

- Phù hợp với dữ liệu phức tạp: Các mô hình học máy và học sâu có thể học được các mẫu phức tạp từ dữ liệu mã hóa.

- Khả năng tổng quát hóa: Có thể áp dụng mô hình đã huấn luyện trên các dữ liệu mới và không được gán nhãn.

**Nhược điểm:**

- Yêu cầu dữ liệu lớn: Đòi hỏi một lượng lớn dữ liệu đã được gán nhãn để huấn luyện mô hình học máy hoặc học sâu.

- Khả năng giải thích kém: Mô hình học sâu có thể trở nên rất phức tạp và khó hiểu, làm cho việc giải thích kết quả trở nên khó khăn.

**Ví dụ**: Một tổ chức ngân hàng muốn phân loại các giao dịch mạng đáng ngờ, bao gồm cả các giao dịch được mã hóa. Họ có thể sử dụng các mô hình học máy và học sâu để xây dựng một hệ thống phát hiện gian lận, có khả năng nhận diện các mẫu giao dịch không bình thường dựa trên các đặc trưng của lưu lượng mạng mã hóa.

4. Kỹ thuật kết hợp

**Ưu điểm:**

- Tận dụng ưu điểm của nhiều phương pháp: Kết hợp các kỹ thuật khác nhau có thể giúp cải thiện hiệu suất và độ chính xác của hệ thống phân loại.

**Nhược điểm:**

- Độ phức tạp tăng lên: Khi kết hợp nhiều phương pháp, có thể làm tăng độ phức tạp của hệ thống và yêu cầu nhiều tài nguyên hơn.

**Ví dụ**: Một nhóm nghiên cứu muốn xây dựng một hệ thống phân loại lưu lượng mạng mã hóa mạnh mẽ và linh hoạt. Họ quyết định kết hợp phân tích thống kê với CNNs và một mô hình học máy để tận dụng ưu điểm của từng phương pháp. Kết quả là hệ thống có khả năng nhận diện hiệu quả các mẫu lưu lượng mạng mã hóa, đồng thời giảm thiểu sai sót và tăng cường hiệu suất.

Tùy thuộc vào yêu cầu cụ thể của bài toán và tài nguyên có sẵn, người triển khai có thể lựa chọn và kết hợp các phương pháp để đạt được hiệu suất tốt nhất trong việc phân loại lưu lượng mạng mã hóa.

# **CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT**

**2.1 Mạng định nghĩa bằng tri thức (Knowledge-Defined Networking)**

Mạng định nghĩa bằng phần mềm (SDN) là một công nghệ mới giúp nâng cao hiệu quả và giảm chi phí cho việc vận hành và quản trị hệ thống mạng cho các tổ chức doanh nghiệp. Các nhà nghiên cứu đã phát triển một kiến trúc mạng mà ở đó nhiệm vụ điều khiển mạng được xử lý bởi các bộ điều khiển và các bộ điều khiển đó có thể tác động tới phần cứng, bộ nhớ và các chức năng của các thiết bị định tuyến (router), chuyển mạch (switch) để đạt được mục đích của người sử dụng. Do đó, mạng lưới trở nên linh hoạt hơn, hiệu suất sử dụng cao hơn và dễ quản lý hơn [2].

A diagram of a switch

Description automatically generatedMạng SDN (Software Defined Network) là một kiến trúc mạng mới, năng động, dễ quản lý, dễ thích nghi, chi phí hiệu quả và rất phù hợp với nhu cầu mạng ngày càng tăng hiện nay. Kiến trúc này phân tách **phần điều khiển mạng (Control Plane)** và **Forwarding Plane (hay Data Plane - chức năng vận chuyển dữ liệu)**, điều này cho phép việc điều khiển mạng trở nên có thể lập trình được dễ dàng và cơ sở hạ tầng mạng độc lập với các ứng dụng và dịch vụ mạng”.

**Hình 2-1:** Sơ đồ mạng SDN

Về cơ bản, SDN được chia làm ba lớp: lớp ứng dụng (Application Layer), lớp điều khiển (Control Layer) và lớp thiết bị hạ tầng (Infrastructure Layer). Các lớp sẽ liên kết với nhau thông qua giao thức hoặc các API (Hình 1.2)

A diagram of a software application

Description automatically generated

**Hình 2-2**: Kiến trúc mạng SDN

Lớp ứng dụng SDN là các chương trình giao tiếp với bộ điều khiển SDN thông qua các giao diện lập trình ứng dụng API, cho phép lớp ứng dụng lập trình (cấu hình) mạng (ví dụ như điều chỉnh các tham số trễ, băng thông, định tuyến, …) qua lớp điều khiển để tối ưu hoạt động của mạng lưới theo một yêu cầu cụ thể của người quản trị.

## **2.2 Tổng quan về mạng KDN**

KDN (Knowledge-Defined Networking) là một khái niệm mới kết hợp giữa SDN (Software-Defined Networking) với dữ liệu phân tích (NA) và máy học (ML) để tạo ra một hệ thống mạng thông minh tích hợp các mô hình học máy hiệu quả hơn trong quá trình phân loại [3]. KDN thêm một tầng kiến thức (Knowledge Plane - KP) vào hai tầng truyền thống của SDN: tầng điều khiển (Control Plane) và tầng dữ liệu (Data Plane). KP giúp thu thập và xử lý dữ liệu thành tri thức để tự động hóa và tối ưu hóa mạng. Kiến trúc của mạng KDN gồm các tầng như sau:

- Tầng dữ liệu (Data Plane): Chịu trách nhiệm chuyển tiếp, xử lý và chỉnh sửa gói tin. Tầng này bao gồm các thiết bị vật lý và ảo.

- Tầng điều khiển (Control Plane): Cung cấp thông tin và cập nhật quy tắc xử lý cho tầng dữ liệu thông qua API phía Nam. Tầng này thu thập dữ liệu từ tầng dữ liệu và cập nhật bảng luồng để thực hiện các hành động.

- Tầng quản lý (Management Plane): Đảm bảo mạng hoạt động tối ưu và quản lý các dịch vụ hỗ trợ và cấu hình thiết bị mạng.

- Tầng kiến thức (Knowledge Plane): Thu thập dữ liệu và chuyển đổi dữ liệu thành tri thức thông qua các thuật toán ML. Tri thức này sau đó được sử dụng để đưa ra các quyết định nhằm tối ưu hóa hoạt động mạng.

A diagram of a computer process

Description automatically generated

**Hình 2-3**: Kiến trúc mạng KDN

## **2.3 Khảo sát một số phương pháp phân loại lưu lượng mạng trên mạng SDN**

Ở phần này chúng tôi sẽ khảo sát một số phương pháp đã được áp dụng trong bài toán phân loại lưu lượng mạng trong mạng SDN.

**Bảng 2-1:** Khảo sát một số phương pháp phân loại lưu lượng mạng trong mạng SDN (Trích [4])

|  |  |  |  |
| --- | --- | --- | --- |
| **Kỹ thuật sử dụng** | **Công cụ** | **Đầu ra mô hình** | **Sự chính xác** |
| MLP, SAE và CNN | TensorFlow | 6 ứng dụng | CNN: 93.35%  MLP: 93.21%  SAE: 93.13% |
| MLP, SAE and CNN | Keras và TensorFlow | 15 ứng dụng | CNN: 99.30%  SAE: 99.14%  MLP: 97.14% |
| ML-LSTM và  CNN-LSTM | Keras và TensorFlow | 8 ứng dụng | ML-LSTM: 99.65%  CNN-LSTM: 98.86% |
| RF | Không đề cập | 8 ứng dụng | RF: 96.0% |
| CNN | Matlab | 6 ứng dụng | CNN: 99.0% |
| k-NN, SVM, DT,  RF, DNN và CNN | Scikit-learn và PyTorch | 3 lớp | k-NN: 98.00%  RF: 97.00%  CNN: 95.00%  DNN: 94.00%  DT: 88%  SVM: 86.00% |
| SAE | Weka và Matlab | 10 lớp | SAE: 91.21% |
| NB | Không đề cập | 5 lớp | NB: 97.6% |
| SVM và K-Means | Không đề cập | 8 lớp | SVM: 98.7%  K-M: 88.0% |
| SVM, NB và NC | Scikit-learn | 3 lớp | NB: 96.79%  SVM: 92.3%  NC: 91.02% |
| DT, k-NN, NB và SVM | Không đề cập | 12 lớp | DT: 99.39%  k-NN: 98.34%  NB: 96.71%  SVM: 96.75% |
| LapSVM | Không đề cập | 4 lớp | LapSVM: >90.0% |
| SVM NC, B-NB và MUTIL-CLASS-SVM | Không đề cập | 5 lớp | NC, B-NB và MUTIL-CLASS-SVM: > 90.0% |
| DNN, SVM, k-NN  và DT | TensorFlow và  Scikit-learn | 4 lớp | DNN: 88.00%  DT: 85.00%  SVM: 80.0%  k-NN: 79.0% |
| DNN | Không đề cập | 10 lớp | DNN: 96.00% |
| KNN, RF và DT | Scikit-learn | 4 lớp | DT: 87.20%  RF: 85.10%  k-NN: 79.50% |
| KNN, RF và DT | Scikit-learn và Keras | 10 lớp | k-NN: 97.14%  RF: 96.69%  DT: 95.80% |
| CNN và autoen-coder, CNN, DNN | TensorFlow và Keras | 24 ứng dụng | CNN và autoen-  coder: 97.42%  CNN: 96.03%  DNN: 94.36% |
| RF và k-NN | Weka | 40 ứng dụng di động | RF: 95.00%  k-NN: >90.00% |
| DT | Weka | 2 lớp | DT: >90.00% |
| ResNet và GRU | Không đề cập | 2 lớp | ResNet: >93.63%  GRU: >86.53% |
| MLP | Keras và TensorFlow | 2 lớp | MLP: >96.00% |
| RL-RF, SVM, RF  và NB | Không đề cập | 2 lớp | RL-RF: 99.54%  SVM: 98.18%  RF: 97.18%  NB: 96.42% |

## **2.4 Học chuyển giao trong bài toán phân loại lưu lượng mạng**

Ngày nay, các phương pháp sử dụng kỹ thuật học chuyển giao đang trở nên ngày càng hứa hẹn trong vấn đề phân loại lưu lượng mạng được mã hóa. Trong bài báo [8], dữ liệu đã được chuyển từ mô hình CNN được huấn luyện trước sang một mô hình khác dùng để phân loại các ứng dụng của Google. Mô hình ban đầu được huấn luyện trên một tập dữ liệu và sau đó được chuyển và tinh chỉnh trên một tập dữ liệu nhỏ hơn. Tuy nhiên, mô hình này bị giới hạn về độ chính xác, chỉ đạt 85% do sự khác biệt giữa hai tập dữ liệu. Sự khác biệt này chủ yếu xuất phát từ sự khác biệt trong phân phối nhãn, sự thiếu hụt dữ liệu và các yếu tố khác.

Sun và các cộng sự [9] đã giới thiệu một giải pháp cho vấn đề khác biệt phân phối dữ liệu được giả định bởi phương pháp học máy truyền thống sử dụng thuật toán TrAdaBoost. Phương pháp này sử dụng mô hình Maxent làm bộ phân loại cơ bản và cập nhật trọng số mẫu dựa trên tỷ lệ lỗi. Phương pháp này đạt được độ chính xác cao trong phân loại trên một tập dữ liệu mới không hoàn toàn giống với tập dữ liệu huấn luyện, với độ chính xác lên đến 98,7%.

# **CHƯƠNG 3 : PHƯƠNG PHÁP TRANSFER LEARNING VÀ THUẬT TOÁN TRADABOOST**

**3.1 Tìm hiểu về phương pháp Transfer Learning**

**3.1.1 Khái niệm Transfer Learning**

Transfer Learning (TL) là một kỹ thuật trong lĩnh vực học máy và học sâu, nơi mà kiến thức đã học được từ một nhiệm vụ được sử dụng lại để cải thiện hiệu suất của một nhiệm vụ mới. Đây là một phương pháp rất hữu ích khi dữ liệu cho nhiệm vụ mới khan hiếm hoặc khó thu thập. Transfer Learning có thể được định nghĩa là quá trình cải thiện hiệu suất của một mô hình trên một nhiệm vụ mục tiêu bằng cách sử dụng kiến thức từ một hoặc nhiều nhiệm vụ nguồn. Các nhiệm vụ nguồn thường có sẵn dữ liệu phong phú và mô hình đã được huấn luyện trước trên những dữ liệu này.

### **3.1.2 Ưu điểm**

1. Tiết kiệm thời gian và tài nguyên: Việc huấn luyện mô hình từ đầu thường đòi hỏi nhiều tài nguyên tính toán và thời gian. TL giúp tận dụng mô hình đã được huấn luyện trước, giảm thiểu đáng kể công sức và thời gian.

2. Hiệu suất cao hơn với ít dữ liệu: TL cho phép mô hình đạt hiệu suất tốt hơn ngay cả khi có ít dữ liệu trong nhiệm vụ mới.

3. Giảm nhu cầu về dữ liệu nhãn: Trong nhiều trường hợp, việc gán nhãn dữ liệu rất tốn kém và mất thời gian. TL giảm nhu cầu này bằng cách sử dụng các mô hình đã học trước đó.

4. Khả năng áp dụng rộng rãi: TL có thể được áp dụng trong nhiều lĩnh vực khác nhau, từ nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên đến hệ thống khuyến nghị.

### **3.1.3 Nhược điểm**

1. Sự không tương thích giữa nhiệm vụ nguồn và nhiệm vụ mục tiêu: Nếu nhiệm vụ nguồn và nhiệm vụ mục tiêu quá khác biệt, TL có thể không hiệu quả và thậm chí gây ra hiệu ứng tiêu cực.

2. Quá trình tinh chỉnh: Việc tinh chỉnh (fine-tuning) mô hình đã được huấn luyện trước đôi khi phức tạp và đòi hỏi kiến thức sâu về cấu trúc của mô hình.

3. Hiệu ứng overfitting: Nếu mô hình không được tinh chỉnh đúng cách, nó có thể dẫn đến overfitting, tức là mô hình hoạt động tốt trên dữ liệu huấn luyện nhưng kém trên dữ liệu mới.

5. Chuyển đổi miền (Domain adaptation): Khi dữ liệu từ nhiệm vụ nguồn và nhiệm vụ mục tiêu đến từ các miền khác nhau, sự khác biệt này có thể làm giảm hiệu quả của TL.

6. Sự tương tự giữa các nhiệm vụ: Để TL hiệu quả, nhiệm vụ nguồn và nhiệm vụ mục tiêu cần phải có mức độ tương tự nhất định.

**3.2 Tìm hiểu về thuật toán TrAdaBoost**

**3.2.1 Tìm hiểu về thuật toán AdaBoost**

AdaBoost, viết tắt của Adaptive Boosting, là một thuật toán nổi bật trong lĩnh vực học máy Ensemble learning. Ensemble learning là một kỹ thuật kết hợp nhiều mô hình học máy để tạo ra một mô hình mạnh mẽ hơn, nhằm cải thiện hiệu suất và độ chính xác so với từng mô hình đơn lẻ. Ý tưởng chính của ensemble learning là "nhiều cái đầu tốt hơn một cái đầu," nghĩa là nhiều mô hình kết hợp với nhau có thể bù đắp cho những điểm yếu của nhau.

AdaBoost được giới thiệu bởi Yoav Freund và Robert Schapire vào năm 1995. Thuật toán này hoạt động bằng cách kết hợp nhiều mô hình học yếu (weak learners) - những mô hình có hiệu suất chỉ tốt hơn một chút so với dự đoán ngẫu nhiên - để tạo ra một mô hình học mạnh (strong learner).

A screenshot of a computer

Description automatically generated

**Hình 3-1:** Minh họa của Ensemble Learning

**Nguyên lý hoạt động của AdaBoost**

AdaBoost hoạt động bằng cách huấn luyện một chuỗi các weak learners trên cùng một tập dữ liệu, nhưng với một sự điều chỉnh về trọng số của các mẫu trong tập dữ liệu qua mỗi lần huấn luyện. AdaBoost là một thuật toán mạnh mẽ, nổi bật với khả năng kết hợp nhiều mô hình học yếu để tạo ra một mô hình học mạnh với hiệu suất cao. Mặc dù có một số nhược điểm như nhạy cảm với nhiễu và khả năng overfitting, nhưng với sự điều chỉnh và lựa chọn đúng đắn, AdaBoost có thể được áp dụng thành công trong nhiều bài toán thực tế.

**3.2.2 Thuật toán học chuyển giao thích ứng (TrAdaBoost)**

TrAdaBoost là một kỹ thuật để giải quyết các vấn đề học chuyển giao bằng cách áp dụng AdaBoost [5]. Mục tiêu của TrAdaBoost là cải thiện khả năng học của một mô hình trên một bài toán đích bằng cách giảm thiểu độ lệch phân phối của dữ liệu giữa bài toán nguồn và bài toán mục tiêu. TrAdaBoost giải quyết vấn đề này bằng cách sử dụng một cơ chế chọn lọc và đánh độ ảnh hưởng cao cho những mẫu dữ liệu có ích từ bài toán nguồn, mở rộng ý tưởng từ thuật toán AdaBoost để xử lý bài toán Transfer Learning, nơi dữ liệu nguồn có thể giúp cải thiện hiệu suất trên dữ liệu mục tiêu.

**A screenshot of a computer

Description automatically generated**

**Hình 3-2:** Sơ đồ mã giả của Thuật toán TrAdaBoost

TrAdaBoost mở rộng AdaBoost cho bài toán Transfer Learning bằng cách sử dụng hai loại trọng số khác nhau cho dữ liệu nguồn và dữ liệu mục tiêu. Quá trình hoạt động của TrAdaBoost có thể được tóm tắt như sau:

- Khởi tạo: Tất cả các mẫu dữ liệu nguồn và dữ liệu mục tiêu được gán trọng số ban đầu bằng nhau.

- Lặp lại (iteration): Tại mỗi vòng lặp, một mô hình yếu được huấn luyện trên toàn bộ dữ liệu. Sau đó, các trọng số của mẫu dữ liệu được cập nhật dựa trên lỗi phân loại của chúng.

- Đối với các mẫu dữ liệu nguồn, trọng số của chúng giảm dần theo từng vòng lặp để chúng ít ảnh hưởng hơn đến mô hình cuối cùng.

- Đối với các mẫu dữ liệu mục tiêu, trọng số của chúng được cập nhật tương tự như trong AdaBoost, tăng trọng số cho các mẫu bị phân loại sai.

- Kết hợp mô hình: Sau một số vòng lặp, các mô hình yếu được kết hợp để tạo thành một mô hình mạnh, trong đó trọng số của các mô hình yếu được xác định dựa trên hiệu suất của chúng.

A diagram of a fault and fault

Description automatically generated

**Hình 3-3:** Biểu diễn cách hoạt động của giải thuật học chuyển giao – TrAdaBoost

1. Data samples in the source domain : Mẫu dữ liệu trong miền nguồn
2. Data samples in the target domain : Mẫu dữ liệu trong miền đích
3. Incorrectly identified data samples : Mẫu phân loại không chính xác
4. Classification boundary : Đường phân loại

**3.3 Thuật toán Dynamic TrAdaBoost**

Dynamic TrAdaBoost là một thuật toán được cải tiến từ thuật toán TrAdaBoost gốc [6]. Nó được thiết kế để khắc phục những hạn chế của TrAdaBoost thông qua việc sử dụng một yếu tố động trong việc cập nhật trọng số của các mẫu nguồn. Dynamic TrAdaBoost chỉ ra 2 vấn đề của Thuật toán TrAdaBoost gốc như sau:

1. **Mất cân bằng trong cập nhật trọng số**: TrAdaBoost có xu hướng làm mất cân bằng trọng số giữa các lớp khác nhau, dẫn đến việc mô hình cuối cùng có thể chỉ dự đoán một nhãn cho tất cả các mẫu​​.
2. **Hội tụ nhanh chóng của trọng số nguồn**: TrAdaBoost giảm trọng số của các mẫu nguồn quá nhanh, ngay cả khi các mẫu này có thể có ích cho việc học tập đích trong các vòng lặp sau đó​​.

Để giải quyết các vấn đề trên thì thuật toán Dynamic TrAdaBoost được đề xuất như sau:

**A paper with mathematical equations

Description automatically generated**

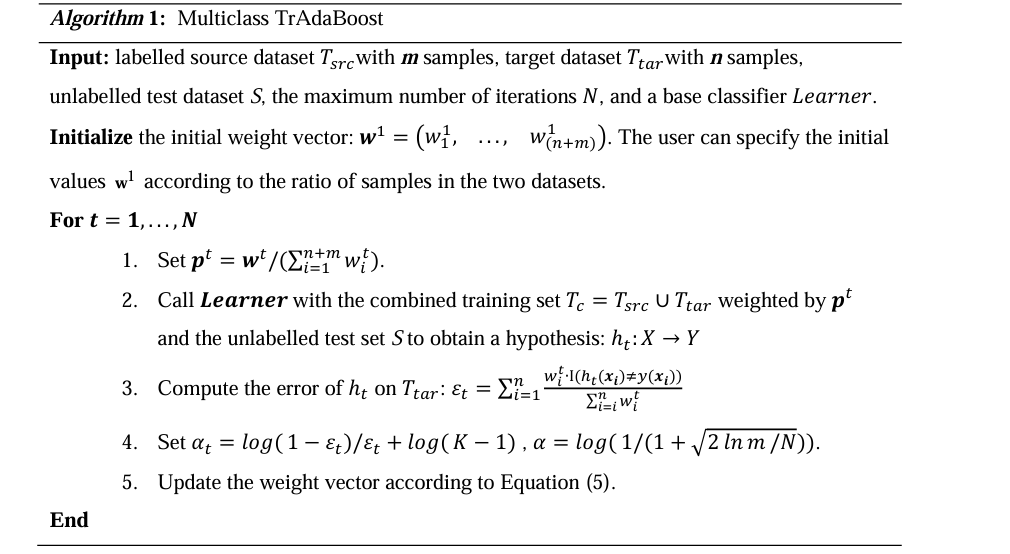
**Hình 3-4:** Giả mã của thuật toán Dynamic TrAdaBoost

Kết quả trên các bộ dữ liệu thực tế cho thấy Dynamic TrAdaBoost cải thiện đáng kể độ chính xác phân loại so với TrAdaBoost và các phương pháp cố định khác. Kết quả cho thấy Dynamic-TrAdaBoost đạt độ chính xác 99,3% so với 97,5% của TrAdaBoost gốc.​

**3.4 Thuật toán Multiclass TrAdaBoost**

Multiclass TrAdaBoost là một phiên bản mở rộng của TrAdaBoost gốc, được thiết kế để xử lý các bài toán phân loại đa lớp thay vì chỉ phân loại nhị phân [7]. Trong khi TrAdaBoost gốc tập trung vào việc cải thiện hiệu suất phân loại giữa hai lớp bằng cách điều chỉnh trọng số của các mẫu nguồn và đích dựa trên lỗi phân loại, Multiclass TrAdaBoost phải xử lý sự phức tạp của nhiều lớp. Điều này yêu cầu cập nhật trọng số không chỉ dựa trên lỗi phân loại mà còn phải cân nhắc hiệu suất trên từng lớp cụ thể. Quá trình huấn luyện của Multiclass TrAdaBoost bao gồm việc kết hợp dữ liệu từ các lớp khác nhau, đảm bảo mô hình cuối cùng có thể phân biệt chính xác giữa nhiều lớp, và tính toán lỗi cũng như điều chỉnh trọng số một cách hiệu quả để cải thiện hiệu suất trên tất cả các lớp.

Mã giả của thuật toán Multiclass TrAdaBoost :



**Hình 3-5**: Giả mã của thuật toán Multiclass TrAdaBoost

Kết quả thực nghiệm cho thấy: Multiclass TrAdaBoost đạt được những thông số tốt hơn hẳn về các độ đo như Độ chính xác precision, recall và F1 – score so với các mô hình mới nhất, đặc biệt là trên tập dữ liệu bị mất cân bằng phân phối.

# **CHƯƠNG 4 : THUẬT TOÁN ĐỀ XUẤT**

## **4.1 Mô tả bài toán**

Cho hai miền riêng miền riêng biệt trên một mạng KDN: miền nguồn (S) và miền đích (T). Dữ liệu từ hai miền này, được biểu diễn lần lượt là và đại diện cho dữ liệu lưu lượng mạng với các nhãn tương ứng. Giả định rằng miền nguồn có dữ liệu rất đầy đủ và phân phối của các nhãn là tương đối đều trong khi miền đích có lượng dữ liệu cực kì hạn chế và phân phối cũng là không đồng đều. Nhiệm vụ đặt ra là phát triển một mô hình phân loại có khả năng phân loại dữ liệu từ miền đích vào nhóm dịch vụ mạng khác nhau dựa trên các mẫu học được từ miền nguồn. Bài toán có thể được mô hình hóa như sau:

Có 𝐷s = {(, ), (, ), . . ., (,)} và Dt = {(, ), (, ), . . ., (,)} đại diện cho các tập dữ liệu từ miền nguồn và miền đích, tương ứng, trong đó:

* ∈ và ∈ payload được mã hóa với 𝑣 byte cho mỗi nhóm dịch vụ trong miền nguồn và miền đích.
* và ∈ {0, 1, . . ., 𝐾} là các nhãn tương ứng chỉ định nhóm dịch vụ mạng.
* và là số lượng mẫu trong miền nguồn và miền đích, tương ứng.
* là số lượng gói của mỗi luồng.
* 𝐾 là số lượng nhóm dịch vụ mạng.

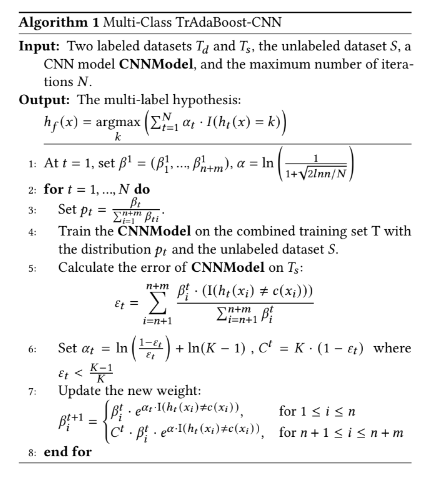
**Mục tiêu**: Mục tiêu là xây dựng một mô hình phân loại sử dụng tập dữ liệu kết hợp từ cả hai miền, 𝐷s và 𝐷t, với mục tiêu cuối cùng là giảm thiểu tỷ lệ lỗi 𝜀 khi phân loại dữ liệu đích.

**Điều kiện**: Tập dữ liệu miền đích 𝐷t bị thiếu hụt dữ liệu.

Mục tiêu của nghiên cứu này là xây dựng một mô hình phân loại sử dụng tập dữ liệu kết hợp từ cả hai miền và với mục tiêu cuối cùng là giảm thiểu tỷ lệ lỗi khi phân loại dữ liệu miền đích. Tuy nhiên, vấn đề gặp phải là tập dữ liệu miền đích bị thiếu hụt dữ liệu nghiêm trọng. Để giải quyết vấn đề này, chúng tôi đề xuất một thuật toán mới tên là Multi-Class TrAdaBoost-CNN, cải tiến từ tư tưởng của thuật toán Multi-Class TrAdaBoost gốc. Thuật toán này dựa trên các nguyên lý của TrAdaBoost, nhằm chuyển giao kiến thức từ miền nguồn sang miền đích bằng cách giảm trọng số của các điểm dữ liệu không tương đồng trong miền nguồn, từ đó giảm thiểu ảnh hưởng của chúng trong quá trình chuyển giao ở miền đích.

**4.2 Thuật toán đề xuất**

Trong phần này, chúng tôi giới thiệu về thuật toán **Multi-class TrAdaBoost-CNN** được thiết kế để giải quyết vấn đề phân loại lưu lượng mạng cho mạng KDN đa miền.



**Hình 4-1**: Sơ đồ giả mã thuật toán đề xuất

**Bảng 4-1**: Chú thích và giải thích của thuật toán

|  |  |
| --- | --- |
| **Ký Hiệu** | **Mô Tả** |
|  | Tập dữ liệu đã gán nhãn cho miền nguồn |
|  | Tập dữ liệu đã gán nhãn cho miền đích |
| S | Tập dữ liệu không gán nhãn |
| CNNModel | Mô hình học sâu mạng nơ-ron tích chập |
| N | Số lần lặp tối đa |
| β | Vector trọng số |
| n | Số mẫu trong miền đích |
| m | Số mẫu trong miền nguồn |
|  | Phân phối xác suất tại lần lặp t |
|  | Lỗi trên tại lần lặp t |
| K | Số nhãn |
|  | Dự đoán cho mẫu tại lần lặp t |
|  | Nhãn lớp thực tế cho mẫu |
|  | Đầu ra giả thuyết cho nhiều nhãn |
| I (a ≠ b) | Trả về 1 nếu a ≠ b, ngược lại trả về 0 |
|  | Yếu tố điều chỉnh |
| α | Tham số cập nhật trọng số dựa trên số lần lặp tối đa |
|  | Tham số cập nhật trọng số dựa trên mất mát của nhiều nhãn |

Các bước của thuật toán có thể được mô tả như sau:

1. Tiền xử lý dữ liệu gốc và dữ liệu mục tiêu.
2. (Dòng 1) Khởi tạo trọng số tại t = 1, tính 𝜶.
3. (Dòng 4) Huấn luyện mô hình CNN dùng dữ liệu được gộp từ dữ liệu gốc và dữ liệu đích.
4. (Dòng 5 - 6) Tính tỉ lệ lỗi, và cho mô hình CNN trên tập dữ liệu đích.
5. (Dòng 7) Nếu tất cả N vòng lặp không được hoàn thành, dùng mô hình CNN ở vòng lặp trước để khởi tạo cho vòng lặp tiếp theo và cập nhật trọng số tổng hợp cho mô hình cuối cùng.
6. Kết thúc vòng lặp và dự đoán cho mục tiêu.

Trong thuật toán được đề xuất của chúng tôi, mỗi dữ liệu đầu vào được xác định như một ma trận 2D, mỗi gói tin đầu vào có thể được hình dung như một vector có độ dài khác nhau. Chúng tôi đạt được biểu diễn ma trận 2D bằng cách nối liền 𝑛𝑝 gói tin đầu tiên của mỗi luồng lại với nhau, cắt ngắn các gói tin vượt quá 𝑣 byte và điền thêm số 0, hình thành nền tảng cho định dạng ma trận 2D của chúng tôi (lưu ý rằng, 𝑣, 𝑛𝑝 đại diện cho kích thước của ma trận). Thay vì sử dụng Cây Quyết định làm weak learner như thuật toán gốc, chúng tôi chọn một Mạng Nơ-ron Tích chập (CNN). Lựa chọn này được thúc đẩy bởi sự phù hợp của CNN trong việc xử lý dữ liệu đầu vào được biểu diễn dưới dạng ma trận 2D, phù hợp với định dạng dữ liệu của chúng tôi, cho phép chúng tôi tận dụng sức mạnh của CNN trong việc phân tích hiệu quả các luồng đầu vào. Khía cạnh này cũng đặc biệt quan trọng vì Multi Class-TrAdaboost yêu cầu tỷ lệ lỗi dưới (K-1)/K để ngăn chặn tham số cập nhật trọng số 𝛼𝑡 âm.

Như vậy cấu trúc của thuật toán **Multi-Class TrAdaBoost-CNN** có thể được diễn tả như sở đồ dưới đây:

A diagram of a program

Description automatically generated

**Hình 4-2**: Sơ đồ thuật toán đề xuất

# **CHƯƠNG 5 : THỰC NGHIỆM**

## **5.1 Bộ dữ liệu thực nghiệm**

### **5.1.1 Cách thu thập dữ liệu**

Sử dụng TShark để thu thập dữ liệu lưu lượng mạng

**Bước 1**: Cài đặt Tshark

Trước khi bắt đầu phân tích, chúng ta cần cài đặt Tshark. Tshark có thể được cài đặt trên nhiều hệ điều hành khác nhau.

* Trên Ubuntu/Debian:

***sudo apt-get update***

***sudo apt-get install tshark***

* Trên Windows:

Tải và cài đặt Wireshark từ trang web chính thức, Tshark sẽ được cài đặt kèm theo.

**Bước 2**: Bắt lưu lượng mạng

Sau khi cài đặt, chúng ta có thể bắt đầu thu thập dữ liệu lưu lượng mạng. giao diện mạng và lưu vào tệp tin:

***tshark -i eth0 -w capture.pcap***

Lệnh này sẽ bắt lưu lượng mạng từ giao diện eth0 và lưu vào tệp tin `capture.pcap`

**Bước 3**: Phân tích tệp tin lưu lượng

Khi đã thu thập được lưu lượng mạng, chúng ta bắt đầu phân tích dữ liệu. Tshark cung cấp nhiều tùy chọn để lọc và hiển thị thông tin hữu ích.

Đọc tệp tin lưu lượng:

***tshark -r capture.pcap***

Lệnh này sẽ hiển thị toàn bộ các gói tin trong tệp tin `capture.pcap`.

Lọc các gói tin HTTP:

***tshark -r capture.pcap -Y http***

Lệnh này chỉ hiển thị các gói tin liên quan đến giao thức HTTP.

Hiển thị thông tin cụ thể từ các gói tin:

***tshark -r capture.pcap -T fields -e ip.src -e ip.dst -e frame.len***

Lệnh này sẽ hiển thị địa chỉ IP nguồn, địa chỉ IP đích và độ dài khung của mỗi gói tin.

**Bước 4**: Phân tích nâng cao

Tshark cũng hỗ trợ nhiều tính năng phân tích nâng cao, giúp chúng ta hiểu rõ hơn về lưu lượng mạng.

Thống kê lưu lượng theo giao thức:

***tshark -r capture.pcap -q -z io,phs***

Lệnh này sẽ hiển thị thống kê lưu lượng theo các giao thức khác nhau.

Theo dõi các luồng TCP:

***tshark -r capture.pcap -q -z follow,tcp,ascii,1***

Lệnh này sẽ theo dõi và hiển thị nội dung của luồng TCP đầu tiên trong tệp tin.

Tập dữ liệu thô ban đầu sau khi thu thập là các file pcap -> sẽ được xử lý và trích xuất ra thành file feather.

Các luồng dữ liệu mạng mã hóa được thu thập trên các miền mạng SDN không đồng nhất ( như ONOS, RYU và FAUCET).

### **5.1.2 Tiền xử lý dữ liệu**

Trước khi được lưu thành file data.zip, dữ liệu ETC đã được tiền xử lý theo quy trình sau:

1. **Loại bỏ tiêu đề (Header Removal):** Loại bỏ phần tiêu đề của gói tin, chỉ giữ lại phần tải trọng (payload) vì CNN trích xuất đặc trưng từ chính nội dung dữ liệu.
2. **Chuyển đổi sang byte (Conversion to Bytes):** Chuyển đổi dữ liệu tải trọng sang dạng chuỗi byte, đơn vị cơ bản để CNN xử lý.
3. **Chuẩn hóa (Normalization):** Chuẩn hóa giá trị của mỗi byte về khoảng [0, 1] bằng cách chia cho 255 (giá trị lớn nhất của một byte). Bước này giúp ổn định quá trình huấn luyện mô hình.
4. **Thêm số 0 (Zero Padding):** Cố định độ dài của mỗi gói tin bằng cách thêm các byte 0 vào cuối những gói tin có độ dài nhỏ hơn ngưỡng. Bước này đảm bảo tất cả các gói tin đều có cùng kích thước đầu vào cho mô hình CNN.
5. **Tổng hợp gói tin (Packet Aggregation):** Tổng hợp một số gói tin liên tiếp trong cùng một luồng thành một đơn vị dữ liệu duy nhất. Bước này giúp nắm bắt được thông tin ngữ cảnh từ chuỗi gói tin trong một flow.
6. **Feather Saving:** Lưu dữ liệu dưới dạng feather file.

### **5.1.3 Mô tả dữ liệu**

Bộ dữ liệu được sử dụng trong bài toán được nhóm thu thập theo phương pháp như đã trình bày ở trên ở Đông Nam Á vào khoảng tháng 1 và tháng 3 năm 2023. Dữ liệu bao gồm nhiều services khác nhau và được chia thành 2 miền như ở bảng dưới đây:

**Bảng 5-1:** Mô tả dữ liệu

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Services** | **Nums Flows** | **Type** |
|  | Amazon | 834 | E-Commerce |
| eBay | 1836 | E-Commerce |
| Facebook video | 1379 | Video-on-Demand |
| Voice Call | 2198 | Interactive Data |
|  | Thegioididong | 509 | E-Commerce |
| Tiki | 1205 | E-Commerce |
| Youtube | 4645 | Video-on-Demand |
| Chat | 2556 | Interactive Data |

Trong đó:

* là tập dữ liệu đã gán nhãn cho miền đích
* là tập dữ liệu đã gán nhãn cho miền nguồn

Sau đó các services trên phân loại thành ba dịch vụ khác nhau bao gồm:

* Nhãn 0: Nhóm E-commerce được ghép từ nhiều trang thương mại điện tử
* Nhãn 1: Nhóm Video-on-Demand được ghép từ Facebook, Youtube
* Nhãn 2: Nhóm Interactive Data ghép từ Voice Call, Chat

**5.1.2 Chi tiết mô hình**

Mô hình CNN được chúng tôi sử dụng trong thử nghiệm gồm 8 lớp tích chập 2D, mỗi lớp được thiết kế để trích xuất các tính năng từ dữ liệu đầu vào. Cấu hình của các lớp trong mô hình như sau:

**Bảng 5-2:** Thông tin các lớp trong mô hình

|  |  |  |  |
| --- | --- | --- | --- |
| **Layer** | **Number of Filters** | **Filter Size** | **Padding** |
| Layer 1 | 128 | 5x5 | Same |
| Layer 2 | 64 | 5x5 | Same |
| Layer 3 | 64 | 3x3 | Same |
| Layer 4 | 32 | 3x3 | Same |
| Layer 5 | 32 | 3x3 | Same |
| Layer 6 | 16 | 3x3 | Same |
| Layer 7 | 16 | 3x3 | Same |
| Layer 8 | 8 | 3x3 | Same |

Sau mỗi 2 lớp tích chập, một lớp max-pooling với kích cỡ 2x2 và một lớp dropout 25% được thêm vào để tránh trường hợp overfitting.

Phần cuối cùng của mô hình CNN gồm 2 lớp Fully connected cho phân loại. Lớp Fully connected gồm 256 nơ-ron với hàm kích hoạt ReLU. Một lớp Dropout với tỉ lệ 0.1 được thêm vào để chống việc quá khớp dữ liệu. Lớp Fully connected cuối cùng có 3 lớp nơ-ron chính là 3 nhóm dịch vụ cần phân loại cho dữ liệu phân loại SDN của chúng tôi.

**Bảng 5-3:** Các lớp thêm của mô hình

|  |  |  |
| --- | --- | --- |
| **Layer** | **Number of Neutrons** | **Activation function** |
| Fully connected layer | 256 | ReLU |
| Dropout | 0.1 |  |
| Fully connected layer | 3 | Softmax |

## **5.2 Các độ đo và cấu hình thực nghiệm**

### **5.2.1 Giới thiệu độ đo được sử dụng**

Trong nghiên cứu này, chúng tôi sử dụng các độ đo precision, điểm F1 và độ chính xác làm các chỉ số đo để so sánh hiệu suất của mô hình của chúng tôi với các phương pháp khác. Các chỉ số liệu này được sử dụng rộng rãi trong việc đánh giá hiệu quả của các bài toán phân loại, đặc biệt với dữ liệu bị mất cân bằng.

* **Precision** (P): Đánh giá mức độ chính xác của các dự đoán dương tính của mô hình, tức là trong số các mẫu mà mô hình dự đoán là dương tính, có bao nhiêu mẫu thực sự là dương tính, được thể hiện trong phương trình sau:

Trong đó:

* TP (True Positive): Số lượng dự đoán dương tính đúng (mẫu được dự đoán là dương tính và thực sự là dương tính).
* FP (False Positive): Số lượng dự đoán dương tính sai (mẫu được dự đoán là dương tính nhưng thực sự là âm tính).

Nói cách khác, precision cho biết trong số tất cả các trường hợp mà mô hình dự đoán là tích cực, bao nhiêu trường hợp là thực sự tích cực. Precision càng cao, tức là số lượng false positive (dự đoán sai) càng thấp, thì chất lượng của mô hình trong việc dự đoán tích cực càng tốt.

* **Recall** (R): Đánh giá khả năng của mô hình trong việc phát hiện các mẫu dương tính thực sự, tức là trong số các mẫu thực sự là dương tính, có bao nhiêu mẫu được mô hình dự đoán đúng là dương tính., như được hiển thị trong phương trình dưới đây:

Trong đó:

* FP (False Positive): Số lượng dự đoán dương tính sai (mẫu được dự đoán là dương tính nhưng thực sự là âm tính).

Recall cao cho thấy rằng mô hình có khả năng “bắt” được một tỷ lệ lớn các trường hợp tích cực từ dữ liệu.

* **F1-Score** (F1) là một chỉ số trung bình trung hòa giữa chỉ số Precision và chỉ số Recall, cung cấp một cách tổng quát để đánh giá hiệu suất của một mô hình phân loại. F1-Score được biểu diễn như sau:
* **AFCT** (Average Flow Classification Time): Là thời gian trung bình mà mô hình mất để phân loại một flow dữ liệu. Được tính bằng cách chia tổng thời gian để phân loại tất cả các flow cho tổng số flow.

### **5.2.2 Cấu hình thực nghiệm**

Để tìm ra các tham số tối ưu cho thuật toán, chúng tôi thực hiện lần lượt các thực nghiệm theo trình tự dưới đây để tìm ra bộ tham số tối ưu nhất cho bài toán:

**(1)** Đầu tiên mô hình CNN sẽ được huấn luyện tập chung các bộ dữ liệu với kích thước của mỗi packet là khác nhau lần lượt là: 128, 256, 512, 1400 byte. Ngoài ra với mỗi bộ dữ liệu sẽ được kiểm thử với số lượng packet cho mỗi flow khác nhau: 5, 10 và 20 để tìm ra số lượng packet tối ưu nhất cho việc phân loại. Các kết quả sẽ được đánh giá dựa trên hai tham số Accuracy và AFCT.

**(2)** Tiếp theo chúng tôi sẽ tiến các thực nghiệm để tìm ra số epoch tối ưu cho mô hình cũng như số layer tối ưu nhất.

**(3)** Cuối cùng, thuật toán được đề xuất của chúng tôi, **Multi-Class TrAdaBoost-CNN** được thực nghiệm và so sánh với các phương pháp**:**

* là mô hình được huấn luyện trên dữ liệu từ miền đích
* đại diện cho một mô hình CNN được huấn luyện bằng cách sử dụng dữ liệu từ cả miền nguồn và miền đích

Để giả lập sử thiếu hụt dữ liệu trên miền đích, tập dữ liệu trên miền đích sẽ chia thành 2 tập train và test trong đó tập train chỉ chiếm lượng nhỏ dữ liệu (1 – 10% tổng dữ liệu)

## **5.3 Kết quả thực nghiệm**

### **5.3.1 Lựa chọn tham số đầu vào tối ưu**

Bảng 5-6 trình bày kết quả đánh giá hiệu suất của mô hình CNN 8 lớp tích chập được trình bày ở mục trước khi thay đổi số lượng gói tin và số lượng byte. Chỉ số được sử dụng để đánh giá là độ chính xác (Accuracy) và thời gian phân loại luồng trung bình (AFCT).

**Bảng 5-6:** Đánh giá mô hình CNN với số gói và số bytes khác nhau

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Number of bytes** | **5 Packets** | | **10 Packets** | | **20 Packets** | |
| **Accuracy** | **AFCT** | **Accuracy** | **AFCT** | **Accuracy** | **AFCT** |
| 128 Bytes  256 Bytes  512 Bytes  1400 Bytes | 0.57  0.57  0.59  0.66 | 0.0406  0.0505  0.0666  0.0942 | 0.64  0.69  0.71  0.73 | 0.0518  0.0596  0.0817  0.1070 | 0.67  0.76  0.77  **0.78** | 0.0573  0.0649  0.0893  0.1190 |

Trong Bảng 5, mô hình thành thạo nhất tương ứng với cấu hình tập dữ liệu với (Số byte) được đặt là 1400 và (Số gói) là 20, mang lại độ chính xác là 78%. Tuy nhiên, cần lưu ý rằng cấu hình này thể hiện Thời gian phân loại luồng trung bình (AFCT) tương đối cao là 0,119 giây. Sau đó, các biến thể của với giá trị là 5, 10 và 20 gói được kiểm tra. Độ chính xác cho thấy sự gia tăng đáng kể trong các thí nghiệm này, trong khi AFCT vẫn được duy trì hợp lý. Do đó, lựa chọn cho tham số kích thước đầu vào được thiết lập là = 20. Đáng chú ý, đối với = 20, người ta quan sát thấy rằng mức độ chính xác của 256, 512 và 1400 không khác biệt đáng kể; tuy nhiên, thời gian xử lý của chúng cao hơn đáng kể. Do đó, một sự đánh đổi hợp lý được thực hiện để ưu tiên các tham số đầu vào với = 256 và = 20 do hiệu suất và độ chính xác hợp lý của chúng.

### **5.3.2 Lựa chọn mô hình CNN tối ưu**

**Bảng 5-7:** Hiệu suất mô hình cho các lớp mạng và epochs khác nhau

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Layers** | **100 Epochs** | | **200 Epochs** | | **300 Epochs** | | **400 Epochs** | | **500 Epochs** | |
| **Accuracy** | **F1 Score** | **Accuracy** | **F1 Score** | **Accuracy** | **F1 Score** | **Accuracy** | **F1 Score** | **Accuracy** | **F1 Score** |
| **4 Layers** | 0.72 | 0.68 | 0.72 | 0.68 | 0.71 | 0.66 | 0.72 | 0.68 | 0.69 | 0.65 |
| **6 Layers** | 0.73 | 0.69 | 0.76 | 0.75 | 0.70 | 0.65 | 0.75 | 0.72 | 0.72 | 0.70 |
| **8 Layers** | 0.63 | 0.54 | 0.72 | 0.70 | 0.69 | 0.64 | 0.76 | 0.75 | 0.75 | 0.75 |

Thông qua các thí nghiệm liên quan đến số lượng các lớp tích chập và số lần huấn luyện, như được thể hiện trong bảng trên, hai mô hình đạt được mức đánh giá cao nhất với kết quả thống kê gần như ngang bằng đó là mô hình CNN với 8 lớp tích chập được huấn luyện trong 400 Epochs, và mô hình còn lại là mô hình với 6 lớp tích chập, huấn luyện trong 200 Epochs. Cả hai mô hình đều đạt được độ chính xác là 76%, cao hơn từ 3 đến hơn 10% so với các mô hình còn lại, chúng cũng cùng đạt được chỉ số F1-Score là 0.75, chỉ số cao nhất so với các mô hình khác. Tuy nhiên, trong quá trình huấn luyện. mô hình với 6 lớp tích chập cho thấy khả năng hội tụ nhanh hơn so với mô hình còn lại. Do đó, trong các thực nghiệm tiếp theo, mô hình 6 lớp tích chập và 200 Epochs huấn luyện sẽ được sử dụng cho thuật toán MC TrAdaBoost - CNN.

### **5.3.3 Đánh giá so sánh các thuật toán:**

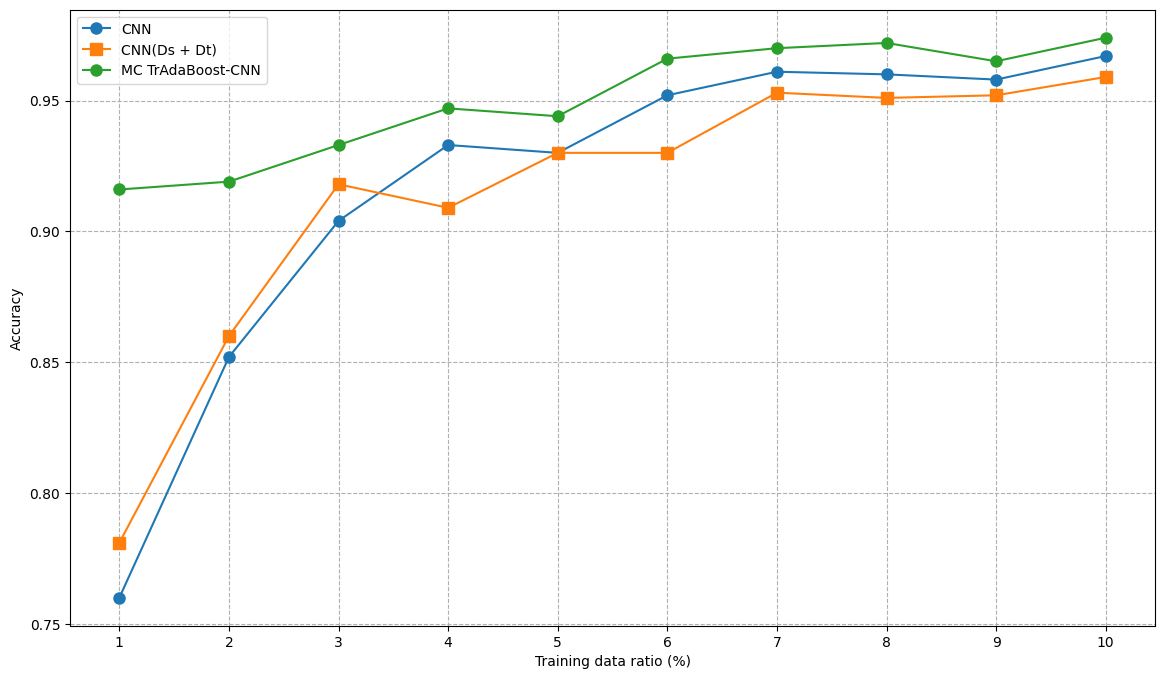
**Bảng 5-8:** Kết quả thực nghiệm của thuật toán

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithm** | **F1-Score** | **Precision** | **Accuracy** | **Recall** | **E-commerce** | **Video-onDemand** | **Interactive Data** |
| **Multi-Class TrAdaBoost-CNN** | 0.92 | 0.93 | 0.92 | 0.92 | 1.00 | 0.92 | 0.86 |
| **CNN (Ds + DT)** | 0.78 | 0.82 | 0.78 | 0.78 | 0.96 | 0.76 | 0.71 |
| **CNN (DT)** | 0.76 | 0.79 | 0.76 | 0.76 | 0.95 | 0.76 | 0.63 |

Kết quả thí nghiệm này được trình bày trong bảng trên. Dễ thấy rằng, khi sử dụng thuật toán Multi-Class- TrAdaboost CNN, các chỉ số F1-Score, Precision, Recall đều được cải thiện đáng kể so với hai mô hình còn lại. Cụ thể F1 - Score được cải thiện 16% so với mô hình CNN và 14% so với mô hình CNN, Precision được cải thiện 14% so với mô hình CNN, 11% so với mô hình CNN. Recall được cải thiện 16% so với mô hình CNN, 14% so với mô hình CNN. Độ chính xác cũng được gia tăng đáng kể khi đặt lên bàn cân với hai mô hình so sánh còn lại. Ngoài ra, thí nghiệm cho thấy thời gian phân loại luồng trung bình (AFCT) của 3 mô hình MULTI-CLASS- TrAdaboost CNN, CNN, CNN lần lượt là 0.06427, 0.0651 và 0.0649. Điều này chỉ ra rằng thời gian để phân loại mẫu của ba mô hình trên không có sự chênh lệch đáng kể, lý do của việc này có thể là do sử dụng cùng một kiến trúc mô hình CNN và phân loại trên cùng một tập dữ liệu.

Độ chính xác của dự đoán trên cả 3 nhãn của thuật toán Mutil-Class- TrAdaboost CNN đều đạt trên 85%, trong đó, nhãn E-commerce được dự đoán chính xác 100% cao hơn 4% so với CNN, 5% so với CNN , nhãn Video-on Demand được dự đoán chính xác 92% cao hơn 16% so với CNN, 14% so với CNNvà nhãn Interactive Data 86%, cao hơn so vói CNN và CNN lần lượt là 15% và 23%. Từ kết quả được thống kê, dễ thấy rằng thuật toán đề xuất đã cải thiện đáng kể khả năng dự đoán chính xác của mô hình, Tuy nhiên, độ chính xác của việc dự đoán các nhãn có sự chênh lệch, nguyên nhân của sự chênh lệch có thể đến từ sự phân bố không đều của dữ liệu được dùng để huấn luyện mô hình.

Để thử nghiệm mang tính khách quan hơn, phạm vi thử nghiệm đã được mở rộng bằng cách thay đổi tỷ lệ dữ liệu huấn luyện và thử nghiệm để nhấn mạnh hiệu quả của thuật toán được đề xuất. Đặc biệt là trong những trường hợp khan hiếm dữ liệu từ miền đích trong quá trình huấn luyện mô hình. Kết quả được thể hiện trong sơ đồ dưới đây:



**Hình 5-1:** Ảnh hưởng của dữ liệu huấn luyện đến độ chính xác mô hình

Đồ thị minh họa độ chính xác của các thuật toán trong các tỉ lệ dữ liệu huấn luyện khác nhau. Dễ thấy trong các tình huống dữ liệu cực kỳ khan hiếm, các thuật toán kết hợp dữ liệu bổ sung từ miền nguồn (D) (Multi-ClassTrAdaBoost - CNN và CNN () cho thấy khả năng dự đoán tốt hơn so với thuật toán chỉ sử dụng dữ liệu từ miền đích (D) như truyền thống. Tuy nhiên theo sự gia tăng của dữ liệu từ miền đích, thuật toán CNN () lại thể hiện kém hơn thuật toán CNN truyền thống. Mặt khác, thuật toán Mutil-CLassTrAdaBoost - CNN luôn thể hiện độ chính xác phân loại cao trên 90% trong mọi tình huống dữ liệu đa dạng. Đáng chú ý, khi dữ liệu từ miền đích ở tình trạng cực kỳ khan hiếm (chỉ có 1% dữ liệu dùng để huấn luyện là dữ liệu miền đích), thuật toán Multi-ClassTrAdaboost - CNN cũng cho thấy khả năng phân loại chính xác các mẫu vượt trội hơn tới 14% so với CNN () và 16% so với CNN ().

# **KẾT LUẬN VÀ KIẾN NGHỊ**

Trong bài báo này, một phương pháp mới được gọi là Multi-Class-TrAdaboost CNN đã được giới thiệu để phân loại đa miền các dịch vụ mạng được mã hóa trong Mạng điều khiển bằng kiến ​​thức (KDN), đặc biệt là khi xử lý dữ liệu hạn chế trong các miền mới nổi. Thuật toán được đề xuất của chúng tôi luôn thể hiện sự vượt trội so với mô hình CNN thông thường, thể hiện khả năng nâng cao độ chính xác đáng kể, thường với mức tăng đáng chú ý, khi xử lý các tập dữ liệu cực kỳ thưa thớt.

Mặc dù đạt được độ chính xác vượt trội, thuật toán được đề xuất của chúng tôi đã cho thấy một số hạn chế đáng kể. Ví dụ, nếu dữ liệu giữa miền nguồn và miền đích quá khác biệt, quá trình học chuyển giao có thể dẫn đến kết quả thấp hơn so với không có học chuyển giao (Negative Transfer).

Trong các tương lai, chúng tôi sẽ cố gắng so sánh phương pháp của mình với các phương pháp tiếp cận gần đây hơn và một loạt các tập dữ liệu sẽ được tiến hành, và các chỉ số bổ sung, chẳng hạn như thời gian suy luận, để cải thiện đề xuất. Chúng tôi có thể phát triển một thuật toán để chọn miền trước quá trình chuyển giao để giảm thiểu sự cố này.

# **PHỤ LỤC**

## **6.1 Wireshark và Tshark**

Wireshark là một công cụ phân tích mạng và ghi lại giao thức mạng. Nó cho phép người dùng theo dõi và phân tích các gói tin mạng trong thời gian thực. Wireshark có khả năng xem và phân tích dữ liệu trong các giao thức mạng phổ biến như TCP, UDP, IP, HTTP và nhiều giao thức khác.

Với Wireshark, người dùng có thể xem các gói tin mạng được truyền qua mạng, xác định nguồn gốc và đích của các gói tin, kiểm tra nội dung và cấu trúc của các giao thức, và phân tích lưu lượng mạng để tìm hiểu về hiệu suất và vấn đề liên quan đến mạng.

Phần mềm này cung cấp một giao diện đồ họa dễ sử dụng, cho phép người dùng lọc và phân loại các gói tin mạng theo nhiều tiêu chí khác nhau. Nó cung cấp các tính năng mạnh mẽ như tìm kiếm, phân tích thống kê, và khả năng xuất dữ liệu phân tích để tạo báo cáo và chia sẻ thông tin với người khác. Wireshark là một công cụ quan trọng trong việc phân tích và xử lý sự cố mạng, tìm hiểu và kiểm tra hiệu suất mạng, và nghiên cứu về các giao thức mạng.

A blue and black logo

Description automatically generated

**Hình 4-3**: Logo của WireShark

**Tshark** là phiên bản dòng lệnh của Wireshark, cung cấp các chức năng tương tự nhưng không có giao diện đồ họa. Điều này làm cho Tshark trở thành lựa chọn lý tưởng cho các kịch bản tự động hóa hoặc khi làm việc trên các hệ thống không có giao diện đồ họa.

Các tính năng chính của Tshark:

* Bắt gói dữ liệu: Giống như Wireshark, Tshark có thể bắt các gói dữ liệu từ nhiều giao diện mạng.
* Phân tích giao thức: Tshark cũng hỗ trợ phân tích hơn 1.000 giao thức mạng.
* Bộ lọc: Tshark sử dụng cùng một bộ lọc mạnh mẽ như Wireshark để lọc các gói dữ liệu.
* Xuất dữ liệu: Tshark cho phép xuất dữ liệu dưới nhiều định dạng khác nhau, hỗ trợ dễ dàng tích hợp với các công cụ khác.
* Dòng lệnh: Làm việc thông qua dòng lệnh, giúp dễ dàng tích hợp vào các tập lệnh hoặc các quy trình tự động hóa.

## **6.2 Tensorflow**



TensorFlow là một thư viện mã nguồn mở phổ biến trong lĩnh vực học máy và trí tuệ nhân tạo. Ban đầu được phát triển bởi nhóm nghiên cứu của Google Brain, TensorFlow cung cấp một cách tiếp cận mạnh mẽ và linh hoạt để xây dựng và triển khai các mô hình học sâu.

Một số điểm nổi bật về TensorFlow:

* Khả năng tính toán phân tán: TensorFlow cho phép bạn phân tán các phép tính trên nhiều thiết bị, từ CPU đến GPU và thậm chí là TPU (Tensor Processing Unit), giúp tăng tốc độ tính toán đối với các mô hình lớn.
* Biểu đồ tính toán động: Mô hình được biểu diễn trong TensorFlow thông qua một biểu đồ tính toán. Điều này cho phép bạn xây dựng và điều chỉnh các phần của mô hình một cách linh hoạt, và TensorFlow sẽ tự động tối ưu hóa và thực hiện các phép tính theo cách hiệu quả nhất.
* API đa ngôn ngữ: TensorFlow cung cấp API cho Python, C++, Java và các ngôn ngữ khác, giúp cho các nhà phát triển có thể sử dụng TensorFlow trong nhiều môi trường và dự án khác nhau.
* Kích cỡ mô hình đa dạng: TensorFlow hỗ trợ xây dựng các mô hình từ nhỏ đến lớn, từ mạng nơ-ron nông (shallow neural networks) cho đến các mô hình học sâu phức tạp như mạng nơ-ron tích chập (CNN) và mạng nơ-ron hồi quy (RNN).
* Cộng đồng lớn và hỗ trợ mạnh mẽ: TensorFlow có một cộng đồng lớn và tích cực, với nhiều tài liệu, tài nguyên học tập, và mã nguồn mở được chia sẻ trên Internet. Điều này làm cho việc học và phát triển với TensorFlow trở nên dễ dàng hơn.
* TensorFlow Extended (TFX): Là một bộ công cụ mở rộng của TensorFlow, TFX cung cấp các công cụ để triển khai và quản lý các dòng sản phẩm học máy trên quy mô lớn.
* TensorFlow Lite và TensorFlow.js: TensorFlow Lite là một phiên bản nhỏ gọn của TensorFlow được thiết kế để chạy trên các thiết bị di động và thiết bị nhúng, trong khi TensorFlow.js cho phép chạy các mô hình TensorFlow trực tiếp trong trình duyệt web.

Keras là một thư viện cao cấp trong Python được sử dụng để xây dựng và huấn luyện mô hình học máy và mạng nơ-ron. Nó cung cấp một API dễ sử dụng và linh hoạt, giúp người dùng tạo ra các mô hình một cách nhanh chóng và hiệu quả. Ban đầu, Keras được phát triển như một thư viện độc lập, nhưng sau đó đã được tích hợp chính thức vào TensorFlow từ phiên bản TensorFlow 2.0 trở đi. Từ đó, Keras trở thành một phần của gói TensorFlow và là một phần quan trọng của TensorFlow API.

# **Tài liệu tham khảo**

[1] Phân loại lưu lượng mạng

* <https://vi.wikipedia.org/wiki/L%C6%B0u_l%C6%B0%E1%BB%A3ng_truy_c%E1%BA%ADp>
* https://ptithcm.edu.vn/wp-content/uploads/2023/07/2020\_HTTT\_LeHoangBao\_LV.pdf

[2] Tổng quan về mạng SDN (Software Defined Network)

<https://opennetworking.org/sdn-definition/?nab=0>

[3] Nam-Thang Hoang, Hai-Nam Nguyen, Hai-Anh Tran, and Sami Souihi. 2022. A Novel Adaptive East–West Interface for a Heterogeneous and Distributed SDN Network. Electronics 11, 7 (2022). https://doi.org/10.3390/ electronics11070975

[4] Núñez-Agurto, Daniel & Fuertes, Walter & Marrone, Luis & Benavides, Eduardo & Vasquez-Bermudez, Mitchell. (2023). Traffic Classification in Software-Defined Networking by Employing Deep Learning Techniques: A Systematic Literature Review. 10.1007/978-3-031-45682-4\_6.

[5] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In Proceedings of the 24th international conference on Machine learning (ICML '07). Association for Computing Machinery, New York, NY, USA, 193–200. <https://doi.org/10.1145/1273496.1273521>

[6] Al-Stouhi, S., & Reddy, C. K. (2011). Adaptive boosting for transfer learning using dynamic updates. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I 11 (pp. 60-75). Springer Berlin Heidelberg.

[7] He, H., Khoshelham, K., & Fraser, C. (2020). A multiclass TrAdaBoost transfer learning algorithm for the classification of mobile lidar data. ISPRS Journal of Photogrammetry and Remote Sensing, 166, 118-127.

[8] Bar-Yanai, R., Langberg, M., Peleg, D., & Roditty, L. (2010). Realtime Classification for Encrypted Traffic. The Sea.

[9] Sun, G., Liang, L., Chen, T., Xiao, F., & Lang, F. (2018). Network traffic classification based on transfer learning. Comput. Electr. Eng., 69, 920-927.