

ĐẠI HỌC CÔNG NGHỆ - ĐẠI HỌC QUỐC GIA HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN



Báo cáo cuối kỳ
Cơ sở hệ thống thông tin
Lớp học phần : INT3201 53

PHÂN TÍCH BÀI BÁO

LR-XFL: Học liên kết phân tán giải thích được dựa trên
lý luận logic

Giảng viên hướng dẫn: PGS. TS Hà Quang Thụy

Nhóm thực hiện:

Lê Minh Tâm	22024500
Nguyễn Thị Hương	22024549
Hoàng Thu Hiếu	22024517
Phạm Thu Trang	22024548

Hà Nội, 2024

MỤC LỤC

Lời mở đầu	5
1 Giới thiệu chung	6
1.1 Giới thiệu về bài báo	6
1.2 Giới thiệu về các tác giả	6
1.2.1 Yanci Zhang	6
1.2.2 Han Yu	7
2 Phân tích bài báo	9
2.1 Lý thuyết nền tảng	9
2.1.1 Trí tuệ đám đông	9
2.1.2 Trí tuệ máy móc và Học liên kết phân tán	10
2.2 Chủ đề bài báo	10
2.3 Đóng góp của bài báo	11
2.4 Cấu trúc bài báo	12
2.5 Tập dữ liệu	12
2.6 Công cụ	14
2.7 Công trình liên quan	15
2.7.1 AI giải thích theo khái niệm	15
2.7.2 Học liên kết	15
2.8 Phương pháp được đề xuất	16
2.8.1 Cơ sở của phương pháp	16
2.8.2 Phương pháp LR-XFL được đề xuất	18
2.9 Kịch bản thực nghiệm	24
2.9.1 Mục tiêu của thí nghiệm	24
2.9.2 Thiết lập dữ liệu	25
2.9.3 Thí nghiệm dữ liệu nhiễu	25
2.9.4 Đối chiếu các phương pháp cơ sở	25
2.9.5 Tiêu chí đánh giá	26
2.9.6 Kết quả	28
3 Triển khai tái tạo thực nghiệm của tác giả	31
3.1 Giới thiệu chương trình và mô tả tập dữ liệu	31
3.1.1 Giới thiệu chương trình	31
3.1.2 Mô tả tập dữ liệu	31

3.1.3 Tái tạo thực nghiệm của tác giả	33
4 Triển khai thí nghiệm riêng của sinh viên	40
4.1 Thí nghiệm cải tiến	40
4.1.1 Tập dữ liệu bổ sung	40
4.1.2 Thí nghiệm cải tiến	41
4.1.3 Đề xuất cải tiến	46
4.1.4 Kết quả	49
4.2 Thí nghiệm mở rộng	50
5 Nghiên cứu mở rộng	53
5.1 Giới thiệu về phương pháp	53
5.2 Kiến trúc mô hình	54

DANH MỤC HÌNH ẢNH

2.1	Kiến trúc và quy trình hoạt động của LR-XFL	21
2.2	Kết quả thí nghiệm dưới các mức độ nhiễu khác nhau.	30
3.1	Thư trả lời của tác giả	32
3.2	Ví dụ về việc sửa đường dẫn import các module trong mã nguồn Python trong một tệp	34
3.3	Tạo tệp <code>__init__.py</code>	35
3.4	Cài đặt và chạy các module chính của dự án.	36
3.5	Lệnh chạy mô hình LR-XFL trên ba bộ dữ liệu: MNIST, MIMIC-II, và VDEM.	37
3.6	Sửa mã nguồn để chạy mô hình Học tập trung.	37
3.7	Sửa mã nguồn để chạy mô hình DDT.	37
4.1	Đoạn mã triển khai chiến lược tương tác với máy khách.	51
5.1	Với mỗi lớp i , mạng sử dụng một “đầu” của lớp tuyến tính dựa trên entropy (màu xanh lá) làm lớp đầu tiên, và nó đưa ra dự đoán về lớp f^i và bảng chân trị T^i (Phương trình 6 để đưa ra giải thích vị từ (màu vàng, trên).	54
5.2	Một cái nhìn chi tiết về một “đầu” của lớp tuyến tính dựa trên entropy cho lớp đầu tiên, làm nổi bật vai trò của khái niệm đầu vào thứ k như ví dụ: (i) giá trị vô hướng γ_k^1 (Phương trình 1) được tính từ tập hợp trọng số kết nối khái niệm đầu vào thứ k với các nơ-ron đầu ra của lớp tuyến tính dựa trên entropy; (ii) tầm quan trọng tương đối của mỗi khái niệm được tóm tắt qua phân phối phân loại α^1 (Phương trình 2); (iii) các điểm số mức độ liên quan được thay đổi lại $\tilde{\alpha}^1$ loại bỏ các khái niệm đầu vào không liên quan (Phương trình 3); (iv) trạng thái ẩn h_1 (Phương trình 4) và các khái niệm kiểu Boolean \hat{c}^1 (Phương trình 5) được cung cấp như các đầu ra của lớp tuyến tính dựa trên entropy.	55

DANH MỤC BẢNG BIỂU

2.1 Thiết kế tổng quan của	19
2.2 Kết quả thí nghiệm. Kết quả tốt nhất được đánh dấu bằng chữ in đậm. Ký hiệu ‘-’ cho biết chỉ số đánh giá không phù hợp với phương pháp tương ứng. .	29
2.3 Kết quả nghiên cứu loại bỏ	30
3.1 So sánh hiệu quả trên tập dữ liệu MNIST (Chấn/Lẻ)	37
3.2 Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu MNIST (Chấn/Lẻ), cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.	38
3.3 So sánh hiệu quả trên tập dữ liệu VDEM	38
3.4 Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu VDEM, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.	38
3.5 So sánh hiệu quả trên tập dữ liệu MIMIC-II	38
3.6 Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu MIMIC-II, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.	39
4.1 So sánh hiệu quả mô hình trước và sau khi cải tiến công thức tính chéo. . . .	49
4.2 So sánh hiệu quả mô hình trên hai trường hợp <code>user_engagement_scale</code> trên bộ dữ liệu VDEM.	52

Lời mở đầu

Trước hết, chúng em xin gửi lời cảm ơn chân thành đến PGS.TS Hà Quang Thụy vì sự tận tình chỉ dạy của thầy với chúng em trong suốt một kì học vừa qua. Nhờ có sự hỗ trợ của thầy mà chúng em đều đã tiến bộ trong quá trình tự học và nghiên cứu của mình.

Tiếp đó, chúng em cũng xin gửi lời cảm ơn tới những chủ nhân của bài báo mà chúng em tham khảo vì sự đóng góp vô cùng to lớn cho quá trình chúng em tiến hành công trình nghiên cứu của mình, nhờ có những bài báo của họ mà chúng em tìm ra được những hướng đi đúng đắn để đi tới được kết quả cuối cùng,

Chúng em xin cam kết đã dùng hết công sức, sự nỗ lực và tìm tòi học hỏi, sự trung thực của từng cá nhân để có thể thực hiện được bài tập này. Song chúng em cũng không thể tránh khỏi những sai sót có thể nảy sinh và hy vọng được thầy cũng như các bạn đọc đóng góp một cách thẳng thắn và chân thành. Từ những phản hồi đó mà chúng em có thể cải thiện công trình và nâng cao những kỹ năng phục vụ bản thân sau này. Chúng em xin chân thành cảm ơn!

Chương 1

Giới thiệu chung

1.1 Giới thiệu về bài báo

Bài báo nhóm thực hiện phân tích có tên **LR-XFL: Logical Reasoning-based Explainable Federated Learning** [7], được đăng trên tạp chí *arXiv* tại đường dẫn <https://arxiv.org/abs/2308.12681>, công bố ngày **24/08/2023** trong danh mục **Computer Science**. Theo thông tin trên Google Scholar, bài báo hiện đã có 3 nghiên cứu khác tham chiếu.

1.2 Giới thiệu về các tác giả

Theo công bố trong bài báo **LR-XFL: Logical Reasoning-based Explainable Federated Learning**, nghiên cứu được thực hiện bởi nhóm tác giả **Yanci Zhang** và **Han Yu**.

1.2.1 Yanci Zhang

Yanci Zhang là nghiên cứu sinh tiến sĩ năm cuối và trợ lý nghiên cứu tại Đại học Công nghệ Nanyang (NTU), Singapore. NTU thường nằm trong top 50 trường đại học hàng đầu thế giới theo các bảng xếp hạng uy tín như QS và Times Higher Education, đặc biệt nổi bật trong lĩnh vực kỹ thuật, khoa học, và công nghệ. Yanci tốt nghiệp Đại học Sơn Đông (Shandong University), Trung Quốc với chuyên ngành Khoa học máy tính, đạt GPA xuất sắc 88.63/100. Công trình nghiên cứu của Yanci tập trung vào trí tuệ nhân tạo (AI) và học máy, với những đóng góp nổi bật trong lĩnh vực này. Ngoài việc tham gia vào các dự án nghiên cứu liên ngành tại NTU, Yanci còn tích cực trong việc công bố các nghiên cứu trên các hội nghị và tạp chí quốc tế, thúc đẩy ứng dụng của AI vào giải quyết các vấn đề thực tiễn.

Công bố của tác giả:

1. DBLP:

- Số bài báo tạp chí (j): 30
- Số báo cáo hội nghị (c): 26
- Số ấn phẩm không chính thức (i): 5

Nguồn: <https://dblp.org/pid/30/929.html?q=57220551200>

2. Scopus:

- Số bài báo công bố: 5
- Số tham chiếu: 11
- Chỉ số h-index: 2

Nguồn: <https://www.scopus.com/authid/detail.uri?authorId=57220551200>

3. Google Scholar:

- Số trích dẫn toàn bộ: 33
- Số trích dẫn trong năm gần nhất: 3
- Chỉ số h-index toàn bộ: 1
- Chỉ số h-index trong năm gần nhất: 33
- Chỉ số i10-index toàn bộ: 3
- Chỉ số i10-index trong năm gần nhất: 1

Nguồn: <https://scholar.google.com/citations?user=-o0Ubz4AAAAJ&hl=en>

1.2.2 Han Yu

Dr. Han Yu là Phó Giáo sư chính tại Trường Cao đẳng Khoa học Máy tính và Dữ liệu. (College of Computing and Data Science - CCDS), Đại học Công nghệ Nanyang (NTU), Singapore. Ông từng là Giáo sư trợ lý Nanyang từ 2018 đến 2024 và là Nghiên cứu viên thỉnh giảng tại HKUST. Dr. Yu nhận bằng Tiến sĩ từ NTU năm 2014 và đã từng đạt Học bổng Tiến sĩ Quỹ Thiên niên kỷ Singapore. Nghiên cứu của ông tập trung vào học máy phân tán đáng tin cậy và đã công bố hơn 300 bài báo tại các hội nghị và tạp chí hàng đầu. Ông cũng là đồng tác giả của cuốn sách “Federated Learning” và là Biên tập viên của nhiều tạp chí khoa học uy tín. Dr. Yu được công nhận là một trong Top 2% Nhà khoa học thế giới trong lĩnh vực AI và là một trong Mười Người Trẻ Xuất Sắc Nhất JCI của Singapore.

Công bố của tác giả:

1. DBLP:

- Số sách và tiểu luận (b): 2
- Số bài báo tạp chí (j): 79

- Số báo cáo hội nghị (c): 157
- Số ấn phẩm không chính thức (i): 91
- Các phần trong sách hoặc bộ sưu tập (p): 6
- Số biên tập (e): 5

Nguồn: <https://dblp.org/pid/35/1096-1.html>

2. Scopus:

- Số bài báo công bố: 5
- Số tham chiếu: 11
- Chỉ số h-index: 2

Nguồn: <https://www.scopus.com/authid/detail.uri?authorId=55753004000>

3. Google Scholar:

- Số trích dẫn toàn bộ: 18808
- Số trích dẫn trong năm gần nhất: 16858
- Chỉ số h-index toàn bộ: 54
- Chỉ số h-index trong năm gần nhất: 43
- Chỉ số i10-index toàn bộ: 147
- Chỉ số i10-index trong năm gần nhất: 124

Nguồn: <https://scholar.google.com.sg/citations?user=eXgoTXMAAAAJ&hl=en>

Chương 2

Phân tích bài báo

2.1 Lý thuyết nền tảng

2.1.1 Trí tuệ đám đông

Khi mới đọc tiêu đề bài báo, có một khái niệm hoàn toàn mới lạ đối với nhóm sinh viên. Đó là khái niệm “Học liên kết phân tán”. Vì thế mà trước khi bắt đầu đi vào phân tích bài báo, nhóm đã thực hiện một số nghiên cứu liên quan để hiểu rõ hơn về thuật ngữ này. Quá trình này đã dẫn dắt nhóm đến một khám phá thú vị về khái niệm “Trí tuệ đám đông” - một minh chứng điển hình cho hiệu quả của sự cộng tác và tổng hợp ý kiến [3]. Khái niệm này không phải là mới, mà đã được phát hiện từ hơn một thế kỷ trước thông qua một sự kiện như sau:

Francis Galton (16/2/1822 – 17/1/1911), anh họ của Charles Darwin, là một nhà bác học người Anh nổi tiếng thời Victoria. Vào năm 1906, khi tham dự một hội chợ gia súc, Galton tình cờ chứng kiến một cuộc thi thú vị. Tại đây, người ta trưng bày một con bò đực và toàn bộ dân làng được mời tham gia dự đoán trọng lượng của con bò đó, người chiến thắng sẽ nhận được một phần thưởng hấp dẫn.

Tổng cộng 787 người đã tham gia, bao gồm những người thường xuyên tiếp xúc với gia súc như nông dân, người mổ thịt, và cả những người không có kinh nghiệm liên quan đến gia súc. Các dự đoán dao động từ rất cao đến rất thấp, nhưng cũng có những ước tính khá hợp lý. Sau khi cuộc thi kết thúc, Galton thu thập tất cả các dự đoán và tính toán kết quả trung bình từ toàn bộ đám đông. Điều ngạc nhiên là, trung bình các dự đoán là 1.197 pound, chỉ chênh lệch đúng 1 pound so với trọng lượng chính xác của con bò là 1.198 pound.

Phát hiện này đã dẫn đến một kết luận quan trọng đó là: một đám đông, khi được hợp lại, có thể đưa ra những quyết định hoặc dự đoán cực kỳ chính xác, ngay cả khi từng cá nhân trong đó không có đầy đủ thông tin. Đây chính là cơ sở của khái niệm “Trí tuệ Đám đông”, và nhiều ví dụ khác cũng minh họa cho hiện tượng này.

2.1.2 Trí tuệ máy móc và Học liên kết phân tán

Khái niệm “Trí tuệ Đám đông” không chỉ giới hạn ở con người, mà còn mở ra những gợi ý quan trọng trong lĩnh vực trí tuệ nhân tạo. Nếu một đám đông các cá nhân có thể đưa ra quyết định chính xác khi được tổng hợp thông tin, liệu các hệ thống máy móc, khi hoạt động độc lập và phối hợp với nhau, cũng có thể đạt được điều tương tự?

Trong kỷ nguyên trí tuệ nhân tạo, khái niệm “Wisdom of the Machine”, hay còn được gọi là Trí tuệ Máy móc [1] đã xuất hiện, trong đó các hệ thống máy móc hoặc mô hình độc lập hợp tác với nhau để giải quyết vấn đề một cách hiệu quả và chính xác hơn.

Một ví dụ điển hình của “Trí tuệ Máy móc” là “Học liên kết phân tán” (Federated Learning). Thay vì tập trung toàn bộ dữ liệu vào một nơi, mỗi thiết bị hoặc hệ thống cục bộ sẽ huấn luyện mô hình dựa trên dữ liệu riêng của nó. Sau đó, các kết quả huấn luyện này được gửi về máy chủ trung tâm để tổng hợp thành một mô hình toàn cục. Phương pháp này mang lại nhiều lợi ích thiết thực:

- **Bảo vệ quyền riêng tư và bảo mật:** Dữ liệu không rời khỏi thiết bị cục bộ, giảm nguy cơ bị rò rỉ hoặc xâm phạm.
- **Giảm thiểu chi phí và tận dụng sức mạnh phân tán:** Tận dụng năng lực tính toán của các thiết bị cá nhân, tránh tình trạng quá tải ở trung tâm dữ liệu.
- **Mở ra cơ hội hợp tác mà không cần tập trung hóa dữ liệu:** Các tổ chức hoặc cá nhân có thể hợp tác mà không cần chia sẻ dữ liệu thô, phù hợp với yêu cầu bảo mật trong nhiều lĩnh vực.
- **Thích ứng với thế giới phi tập trung:** Trong bối cảnh dữ liệu ngày càng nhạy cảm và phân tán, phương pháp này phù hợp với các yêu cầu của xã hội hiện đại.

2.2 Chủ đề bài báo

Chủ đề chính của bài báo là **nâng cao khả năng giải thích và bảo mật trong học máy phân tán thông qua phương pháp LR-XFL: Học máy giải thích dựa trên lý luận logic**.

Trong bối cảnh phát triển của trí tuệ nhân tạo, học máy phân tán (Federated Learning - FL) đã trở thành một giải pháp quan trọng cho việc đào tạo mô hình mà không cần tập trung dữ liệu, qua đó đảm bảo quyền riêng tư của người dùng. Tuy nhiên, sự cần thiết phải bảo vệ dữ liệu riêng tư đã dẫn đến những thách thức về tính minh bạch và khả năng giải thích của các mô hình FL. Thiếu sự giải thích rõ ràng có thể gây ra sự nghi ngờ và không tin tưởng từ người dùng, đặc biệt trong các lĩnh vực nhạy cảm như y tế và tài chính.

Để giải quyết các vấn đề này, bài báo **LR-XFL: Logical Reasoning-based Explainable Federated Learning** giới thiệu một cách tiếp cận mới, nhằm cải thiện khả năng giải thích của các mô hình FL mà vẫn đảm bảo tính bảo mật. Nghiên cứu này không chỉ tìm cách

làm rõ cách thức mà các quyết định được đưa ra trong môi trường học máy phân tán mà còn hướng tới việc tạo ra một mô hình có thể được người dùng tin tưởng hơn.

Bằng cách giải quyết các vấn đề tồn đọng về tính minh bạch và khả năng giải thích trong FL, bài báo có tiềm năng mang lại những tiến bộ đáng kể trong việc ứng dụng các mô hình học máy vào các lĩnh vực quan trọng, nơi mà quyền riêng tư dữ liệu và khả năng giải thích là rất cần thiết.

2.3 Đóng góp của bài báo

Tổng hợp quy tắc logic cục bộ một cách linh hoạt

Một trong những đóng góp chính của LR-XFL là khả năng tổng hợp các quy tắc logic cục bộ từ các máy khách trong mạng lưới FL. Các quy tắc này được gửi lên máy chủ FL, nơi chúng được kết hợp thông qua các bộ nối logic, cụ thể là AND hoặc OR, dựa trên thuộc tính của dữ liệu từ từng máy khách. Cách tiếp cận này giúp đảm bảo rằng các quy tắc logic phản ánh đúng đặc điểm của từng nguồn dữ liệu cục bộ, đồng thời duy trì sự linh hoạt trong việc kết nối và sử dụng chúng trên toàn hệ thống.

Phân bố trọng số cho các máy khách dựa trên chất lượng dữ liệu

LR-XFL không chỉ đơn thuần tổng hợp mô hình cục bộ mà còn sử dụng một cơ chế trọng số dựa trên chất lượng dữ liệu của từng máy khách. Máy chủ FL sẽ đánh giá chất lượng dữ liệu cục bộ thông qua các quy tắc logic được gửi lên và sử dụng các trọng số này để tổng hợp các bản cập nhật mô hình. Cách tiếp cận này giúp đảm bảo rằng những đóng góp của các máy khách vào mô hình toàn cục được cân nhắc một cách công bằng, dựa trên giá trị thực của dữ liệu mà họ cung cấp.

Cải thiện độ chính xác và khả năng giải thích của mô hình FL

Thông qua việc tích hợp các quy tắc logic cục bộ và sử dụng trọng số dựa trên chất lượng dữ liệu, LR-XFL không chỉ cải thiện hiệu quả dự đoán mà còn tăng cường khả năng giải thích của mô hình FL. Việc sử dụng các quy tắc logic rõ ràng giúp kết quả của mô hình trở nên dễ hiểu hơn, đặc biệt trong các trường hợp cần đưa ra quyết định dựa trên dữ liệu phức tạp.

Tăng cường tính minh bạch và khả năng giải thích

Một điểm nổi bật khác của LR-XFL là khả năng cung cấp các quy tắc logic minh bạch, dễ dàng kiểm tra và hiệu chỉnh bởi các chuyên gia. Điều này đóng vai trò đặc biệt quan trọng trong các lĩnh vực nhạy cảm như y tế và tài chính, nơi yêu cầu không chỉ tính chính xác mà còn cần khả năng giải thích rõ ràng để đảm bảo tính an toàn và bảo mật dữ liệu. Cách tiếp cận của giúp tăng cường niềm tin vào hệ thống FL và giảm thiểu rủi ro trong việc triển khai thực tế.

2.4 Cấu trúc bài báo

Bài báo LR-XFL: Học liên kết phân tán giải thích được dựa trên lý luận logic được chia thành 6 phần chính theo thứ tự và nội dung như sau:

- **Introduction (Giới thiệu):** Cung cấp bối cảnh và lý do nghiên cứu, nêu rõ sự cần thiết phải cải thiện tính minh bạch và khả năng giải thích trong học liên kết phân tán (Federated Learning). Nó cũng giới thiệu khái quát về phương pháp LR-XFL và các mục tiêu chính của nghiên cứu.
- **Related Work (Công trình liên quan):** Tổng hợp các nghiên cứu trước đó liên quan đến học liên kết phân tán, lý luận logic và khả năng giải thích của mô hình học máy. Nó phân tích những thành tựu và hạn chế của các phương pháp hiện tại, từ đó làm nổi bật đóng góp của bài báo này.
- **Preliminaries (Cơ sở):** Cung cấp các khái niệm cơ bản và lý thuyết cần thiết để hiểu phương pháp được đề xuất.
- **The Proposed LR-XFL Approach (Phương pháp LR-XFL được đề xuất):** Đây là phần chính của bài báo, mô tả chi tiết về phương pháp LR-XFL. Phần này giải thích cách thức hoạt động của phương pháp, bao gồm cách tạo ra quy tắc logic tại các client, cách kết nối các quy tắc logic trên server và cơ chế gán trọng số cho các client.
- **Experimental Evaluation (Đánh giá thực nghiệm):** Trình bày các kết quả thực nghiệm nhằm đánh giá hiệu quả của phương pháp LR-XFL so với các phương pháp nền tảng khác.
- **Conclusions (Kết luận):** Phần này tóm tắt các phát hiện chính của nghiên cứu và nhấn mạnh những đóng góp của phương pháp LR-XFL. Đồng thời phần này đề xuất các hướng nghiên cứu trong tương lai dựa trên những phát hiện và kết quả đạt được trong nghiên cứu.

2.5 Tập dữ liệu

Trong nghiên cứu này, để đánh giá hiệu quả của phương pháp LR-XFL, nhóm nghiên cứu tiến hành các thí nghiệm trên bốn tập dữ liệu chuẩn khác nhau, đồng thời so sánh kết quả với ba phương pháp thay thế. Các tập dữ liệu này bao gồm:

- **MNIST (Even/Odd)**
 - **Mô tả:** Tập dữ liệu MNIST là một trong những tập dữ liệu phổ biến nhất trong lĩnh vực học máy, bao gồm 70.000 hình ảnh số viết tay. Tập này thường được sử dụng để kiểm tra và phát triển các thuật toán phân loại hình ảnh.
 - **Nội dung dữ liệu:**

- Mỗi hình ảnh có kích thước 28x28 pixel và được biểu diễn dưới dạng ma trận số nguyên từ 0 đến 255, thể hiện độ sáng của từng pixel.
- Các hình ảnh đại diện cho các chữ số từ 0 đến 9.
- Để phục vụ cho bài toán chẵn/lẻ, tập dữ liệu này được điều chỉnh thành hai lớp: số chẵn (0, 2, 4, 6, 8) và số lẻ (1, 3, 5, 7, 9), cho phép mô hình học phân loại theo độ chính xác của chữ số và xác định tính chẵn/lẻ.

• CUB (Caltech-UCSD Birds 200)

- **Mô tả:** Tập dữ liệu CUB chứa 11.788 hình ảnh của 200 loài chim khác nhau, với các thuộc tính mô tả chi tiết cho mỗi loài. Đây là tập dữ liệu rất hữu ích trong các bài toán phân loại hình ảnh và nhận diện đặc trưng.
- **Nội dung dữ liệu:**
 - Mỗi loài chim có nhiều hình ảnh, thể hiện các góc độ và điều kiện ánh sáng khác nhau, giúp cải thiện khả năng nhận diện của mô hình.
 - Mỗi loài chim đi kèm với một loạt các thuộc tính mô tả, như màu sắc, hình dạng, và kích thước, cho phép mô hình học các đặc điểm phức tạp hơn của các loài.
 - Mỗi hình ảnh được gán nhãn với loại chim cụ thể, tạo điều kiện cho việc phân loại chính xác.

• V-Dem (Varieties of Democracy)

- V-Dem là một trong những tập dữ liệu quan trọng về chính trị, cung cấp thông tin về thể chế và sự phát triển dân chủ trên toàn thế giới. Tập dữ liệu này tập hợp thông tin từ nhiều nguồn và nghiên cứu để đánh giá các khía cạnh khác nhau của dân chủ.
- **Nội dung dữ liệu:**
 - Tập dữ liệu chứa hàng triệu biến số mô tả các khía cạnh như tự do chính trị, quyền con người, sự tham gia chính trị, và mức độ dân chủ của mỗi quốc gia.
 - Thông tin được thu thập qua nhiều năm, cho phép phân tích sự thay đổi theo thời gian trong các chỉ số dân chủ của mỗi quốc gia.
 - Dữ liệu được phân loại theo từng quốc gia và năm, giúp mô hình học được cách mà các yếu tố chính trị tương tác với nhau để xác định mức độ dân chủ.

• MIMIC-II

- **Mô tả:** MIMIC-II là một tập dữ liệu lâm sàng lớn, chứa thông tin từ hàng triệu bệnh nhân nhập viện tại các bệnh viện ở Mỹ. Đây là một trong những tập dữ liệu y tế phức tạp nhất, cung cấp thông tin chi tiết về tình trạng sức khỏe của bệnh nhân.
- **Nội dung dữ liệu:**

- Tập dữ liệu bao gồm thông tin về lịch sử bệnh lý, các kết quả xét nghiệm, điều trị, và diễn biến bệnh tình.
- Bao gồm độ tuổi, giới tính, và các yếu tố khác như tình trạng sức khỏe trước khi nhập viện.
- Dữ liệu được phân loại theo các chẩn đoán bệnh, cho phép mô hình học từ các mối quan hệ giữa triệu chứng, điều trị và kết quả lâm sàng.

2.6 Công cụ

- **Ngôn ngữ lập trình:** Python

- **Thư viện lập trình:**

- **NumPy:** Thư viện NumPy được sử dụng để thực hiện các phép toán đại số tuyến tính và xử lý mảng. Nó hỗ trợ cho việc xử lý dữ liệu nhanh chóng và hiệu quả, đồng thời là nền tảng của nhiều thư viện khoa học khác.
- **SciPy:** Thư viện SciPy cung cấp các công cụ toán học và thống kê cao cấp, giúp mở rộng khả năng tính toán của NumPy. SciPy được sử dụng để thực hiện các phép tính phức tạp cần thiết cho các thuật toán học máy.
- **Scikit-learn:** Đây là thư viện máy học nổi tiếng trong Python, cung cấp các mô hình và công cụ đánh giá quan trọng để xây dựng và so sánh hiệu quả mô hình. Scikit-learn hỗ trợ các kỹ thuật tiền xử lý dữ liệu, tối ưu hóa mô hình, và đánh giá hiệu quả của các phương pháp.
- **Pandas:** Được sử dụng cho xử lý dữ liệu, Pandas là công cụ mạnh mẽ giúp thao tác với dữ liệu dạng bảng (DataFrame). Với Pandas, dữ liệu có thể dễ dàng được lọc, tổ chức, và chuẩn hóa trước khi sử dụng trong quá trình huấn luyện mô hình.
- **Torch (PyTorch):** PyTorch là thư viện học sâu phổ biến, cung cấp các công cụ để xây dựng và huấn luyện các mô hình mạng nơ-ron. PyTorch được sử dụng như nền tảng cho các mô hình học sâu trong dự án này, hỗ trợ việc triển khai mô hình trên CPU và GPU.
- **SymPy:** Đây là thư viện tính toán ký hiệu trong Python, hỗ trợ thực hiện các phép tính toán học phức tạp dưới dạng biểu thức ký hiệu. Trong dự án, SymPy được sử dụng để xây dựng các mô hình giải thích dựa trên entropy cho mạng nơ-ron.
- **PyTorch Lightning:** Thư viện PyTorch Lightning giúp đơn giản hóa quá trình huấn luyện mô hình bằng cách cung cấp cấu trúc rõ ràng và các công cụ tối ưu hóa tự động. Nó hỗ trợ việc tổ chức mã nguồn và quản lý các quy trình huấn luyện một cách dễ dàng và có thể mở rộng.

2.7 Công trình liên quan

2.7.1 AI giải thích theo khái niệm

XAI dựa trên khái niệm tập trung vào việc nhận diện các khái niệm hoặc trù tượng hóa cấp cao trong dữ liệu. Trong các mô hình học sâu truyền thống, các tầng dưới thường phát hiện các cạnh hoặc kết cấu, trong khi các tầng trên nhận diện các đặc trưng trù tượng hơn như hình dạng hoặc đối tượng.

Học dựa trên khái niệm xem các đặc trưng trù tượng như những “khái niệm” và tập trung làm chúng trở nên dễ hiểu hơn. Các khái niệm thường được hình thành bằng cách kết nối thông tin ẩn từ lớp cuối của mạng nơ-ron thành các đại diện rõ ràng, hoặc thông qua việc điều chỉnh cấu trúc mạng để học các khái niệm này. Mô hình dựa trên khái niệm được thiết kế nhằm mang lại giải thích trực quan và dễ hiểu trong quá trình ra quyết định, giúp tăng tính minh bạch. Ví dụ, nếu một mô hình học từ hình ảnh động vật nhận biết được khái niệm “cánh” và “mỏ”, nó có thể giải thích quyết định phân loại bằng cách chỉ ra rằng sự xuất hiện của cánh trong ảnh là dấu hiệu cho thấy đó thuộc danh mục “chim”.

Các quy tắc logic đóng vai trò như một phương tiện để liên kết các khái niệm đã được trích xuất, từ đó giải thích lý do đằng sau những quyết định của mô hình, dựa trên những khái niệm mà nó đã học được. Những quy tắc logic này thường rất đơn giản và mang tính tất định.

Các hệ thống dựa trên quy tắc truyền thống, như Cây Quyết Định, cung cấp cách giải thích trực quan thông qua các quy tắc logic. Trong DTs, mỗi nhánh quyết định có thể được hiểu như một quy tắc riêng biệt. Tuy nhiên, việc liệt kê tất cả các thuộc tính trong một nhánh quyết định có thể khiến quy tắc trở nên quá dài và chứa nhiều yếu tố không cần thiết, rốt cục lại làm cho mô hình trở nên khó hiểu hơn. Trong lĩnh vực xử lý ngôn ngữ tự nhiên, lý luận logic đã được áp dụng cho các nhiệm vụ như phân tích cảm xúc và dự đoán văn bản. Đối với dữ liệu hình ảnh và dữ liệu bảng, một lớp tuyến tính mới dựa trên entropy đã được phát triển để tạo ra các giải thích theo quy tắc, giúp làm rõ cách mô hình đưa ra quyết định.

Tuy nhiên, các phương pháp này yêu cầu truy cập trực tiếp vào dữ liệu đào tạo, nên không thể áp dụng hiệu quả trong cài đặt FL.

2.7.2 Học liên kết

FL cung cấp giải pháp phân quyền và bảo vệ quyền riêng tư, cho phép đào tạo mô hình AI trên dữ liệu được phân tán và thuộc sở hữu của nhiều nguồn khác nhau. Thay vì chia sẻ dữ liệu gốc có thể chứa thông tin nhạy cảm, chỉ các bản cập nhật mô hình được truyền giữa máy chủ và các máy khách. Tuy nhiên, phương pháp huấn luyện này lại gây khó khăn trong việc giải thích lý do đằng sau các quyết định của mô hình. Để giải quyết thách thức này, đã có nhiều nỗ lực trong việc tích hợp các quy tắc logic vào FL, nhằm tăng tính minh bạch và dễ hiểu cho các quyết định của mô hình.

Một số nghiên cứu đã ứng dụng logic để cải thiện khả năng cá nhân hóa và hiệu quả của

mô hình FL. Chẳng hạn, logic tín hiệu theo thời gian được sử dụng để xác định đặc tính của các máy khách và phân nhóm chúng trong quá trình tổng hợp, giúp tạo ra các mô hình FL cá nhân hóa. Tương tự, logic mờ được áp dụng để tối ưu việc lựa chọn thiết bị khách trong các mạng lưới phương tiện, và quy tắc mờ Takagi–Sugeno được tích hợp vào FL để thực hiện phân cụm mờ liên kết. Tuy nhiên, các phương pháp này vẫn chưa giải quyết được thách thức trong việc phát triển mô hình FL có thể giải thích bằng suy luận logic. Để lấp đầy khoảng trống này, phương pháp LR-XFL đã được đề xuất.

2.8 Phương pháp được đề xuất

2.8.1 Cơ sở của phương pháp

Mô hình cơ sở trong phương pháp LR-XFL được đề xuất là các **giải thích logic dựa trên entropy của mạng nơ-ron**. Chúng ta sẽ có các bước cụ thể mà phương pháp sử dụng mô hình này:

Xác định khái niệm từ dữ liệu đầu vào

LR-XFL bắt đầu bằng cách lấy dữ liệu đầu vào, chẳng hạn như một hình ảnh hoặc một mẫu dữ liệu bảng, rồi biến đổi chúng thành một không gian khái niệm.

Với dữ liệu hình ảnh, mạng xử lý hình ảnh ResNet10 sẽ trích xuất những đặc điểm quan trọng trong ảnh (hình dạng, màu sắc, cấu trúc) và ánh xạ những thông tin này thành các khái niệm dễ hiểu:

- **Dữ liệu đầu vào:** Một ảnh của một con chim.
- **Quá trình trích xuất đặc điểm:** ResNet10 phân tích ảnh và nhận diện các đặc điểm như:
 - **Hình dạng tổng thể:** cơ thể nhỏ, có đầu và đuôi.
 - **Cấu trúc đặc trưng:** có cánh, mỏ nhọn.
- **Ánh xạ thông tin:** Dựa trên các đặc điểm trích xuất được, mô hình ánh xạ chúng thành các khái niệm dễ hiểu:
 - `hasBeak = True` (có mỏ)
 - `hasWing = True` (có cánh)

Còn với dữ liệu bảng, mô hình tuyến tính sẽ trực tiếp ánh xạ các đặc trưng đầu vào thành các khái niệm tương ứng.

Lớp tuyến tính dựa trên entropy

Từ các tham số của lớp tuyến tính dựa trên entropy, nó sẽ suy ra các quy tắc logic. Các tham số này giúp mô hình hiểu khái niệm nào quan trọng và ảnh hưởng đến quyết định phân loại.

Cơ chế hoạt động của nó dựa trên các trọng số entropy, vốn biểu thị mức độ ảnh hưởng của từng khái niệm trong việc dự đoán một lớp cụ thể. Trọng số càng cao, khái niệm đó càng quan trọng đối với lớp được dự đoán.

Ví dụ: Nếu hai khái niệm “có mỏ” và “có cánh” là yếu tố chính giúp nhận diện lớp *Bird*, thì trọng số entropy của chúng sẽ cao hơn khi mô hình dự đoán lớp này. Mô hình sẽ suy ra rằng “có mỏ” và “có cánh” rất quan trọng trong việc xác định loài chim và sẽ tạo ra các quy tắc tương ứng. Khi nhận được một hình ảnh có “mỏ” và “cánh,” mô hình sẽ kích hoạt hai khái niệm này trong lớp tuyến tính dựa trên entropy.

Áp dụng mặt nạ nhị phân

Để lọc ra các khái niệm quan trọng nhất, mô hình áp dụng một mặt nạ nhị phân trên các khái niệm. Mặt nạ nhị phân này sẽ chọn chỉ những khái niệm có trọng số entropy cao hơn một ngưỡng nhất định, bỏ qua những khái niệm không liên quan. Như vậy, các khái niệm còn lại là những khái niệm cốt lõi để xây dựng quy tắc logic cho từng lớp dự đoán.

Minh họa: Trong lớp *Bird*, giả sử mặt nạ nhị phân xác định *hasBeak* và *hasWing* là các khái niệm được kích hoạt. Điều này có nghĩa là các quy tắc sẽ dựa trên hai khái niệm này, vì chúng là những đặc điểm chủ chốt để dự đoán “chim”.

Xây dựng bảng chân lý cho quy tắc logic

Sau khi xác định các khái niệm kích hoạt, mô hình tạo ra một bảng chân lý (truth table) cho từng lớp đầu ra. Bảng chân lý này ghi lại các trạng thái của các khái niệm kích hoạt tương ứng với các điểm dữ liệu cụ thể thuộc lớp đó. Từng hàng trong bảng chân lý tương ứng với một tập hợp các khái niệm kích hoạt cho một dự đoán cụ thể.

Ví dụ:

Khái niệm	hasBeak	hasWing	Kết quả
Mẫu 1	1	1	Bird
Mẫu 2	1	0	Không Bird
Mẫu 3	0	1	Không Bird

Tổng hợp quy tắc logic

Từ bảng chân lý, LR-XFL xây dựng các quy tắc cấp mẫu bằng cách kết nối các khái niệm kích hoạt trong từng hàng bằng toán tử AND. Những quy tắc cấp mẫu này mô tả các điều kiện cần thiết để dự đoán một điểm dữ liệu thuộc lớp cụ thể.

Mỗi hàng là quy tắc cấp mẫu được tạo ra bằng cách kết nối các khái niệm bởi AND

Ví dụ: Nếu một mẫu dự đoán là chim và các khái niệm được kích hoạt là “có cánh” và “có mỏ” thì quy tắc cấp mẫu sẽ là: “Nếu có cánh AND có mỏ, thì đó là chim.”

Sau đó, LR-XFL kết hợp các quy tắc cấp mẫu thành một quy tắc cấp lớp bằng cách dùng toán tử OR hoặc AND tùy thuộc vào ngữ cảnh, để đạt được độ chính xác cao hơn và tổng hợp thông tin từ nhiều mẫu khác nhau.

Ví dụ quy tắc tổng hợp:

- **Quy tắc 1:** $\text{hasBeak} \wedge \text{hasWing} \rightarrow \text{Bird}$
- **Quy tắc tổng hợp:** $\text{hasBeak} \wedge \text{hasWing} \rightarrow \text{Bird}$

Điều chỉnh toán tử

Khi kết nối các quy tắc cấp mẫu bằng toán tử OR (\vee), có thể dẫn đến sự thiếu chính xác trong việc phân loại.

Ví dụ cụ thể được đưa ra là về việc phân loại chim (Bird):

- **Quy tắc 1:** $\text{hasBeak} \leftrightarrow \text{Bird}$ (Nếu có mỏ thì đó là chim).
- **Quy tắc 2:** $\text{hasWing} \leftrightarrow \text{Bird}$ (Nếu có cánh thì đó là chim).

Nếu chúng ta kết hợp hai quy tắc này bằng toán tử OR, ta sẽ có:

$$\text{hasBeak} \vee \text{hasWing} \leftrightarrow \text{Bird} \quad (\text{Nếu có mỏ HOẶC có cánh thì đó là chim}).$$

Vấn đề: Quy tắc kết hợp này không chính xác, vì nó sẽ phân loại sai:

- Những loài chỉ có mỏ mà không có cánh (thú mỏ vịt) cũng bị coi là chim.
- Những loài chỉ có cánh mà không có mỏ (bướm, muỗi, ruồi) cũng bị coi là chim.

Trên thực tế, một con chim phải có cả mỏ và cánh để được phân loại đúng. Vì vậy, trong trường hợp này, toán tử AND (\wedge) sẽ chính xác hơn.

2.8.2 Phương pháp LR-XFL được đề xuất

LR-XFL là khung giải thích dựa trên logic đầu tiên được thiết kế cho Học Liên Kết (FL), trong đó mỗi máy khách học từ dữ liệu cục bộ và gửi các quy tắc logic lên máy chủ trung tâm để tổng hợp thành mô hình toàn cục. LR-XFL kết hợp điểm mạnh của mạng nơ-ron dựa trên entropy với tính linh hoạt của các quy tắc logic, cho phép hệ thống không chỉ đạt được độ chính xác cao mà còn giải thích được các quyết định mà nó đưa ra.

Các thành phần chính của LR-XFL

- **Xác định toán tử logic tự động**

LR-XFL giới thiệu một cơ chế tự động chọn toán tử logic thích hợp (AND hoặc OR) để tạo ra các quy tắc logic từ các khái niệm mà mô hình phát hiện. Điều này giúp mô hình giảm thiểu sai sót khi kết hợp các quy tắc từ nhiều máy khách, nhất là khi có sự khác biệt trong dữ liệu.

- **Kết hợp quy tắc logic từ máy khách**

Một thách thức lớn trong FL là việc tích hợp các quy tắc từ nhiều máy khách với các phân phối dữ liệu khác nhau. LR-XFL giải quyết điều này bằng cách tối ưu hóa các quy tắc logic cục bộ tại mỗi máy khách, sau đó tổng hợp chúng một cách thông minh tại máy chủ để tạo ra các quy tắc logic toàn cục.

- **Trọng số quy tắc logic dựa trên hiệu quả**

LR-XFL không chỉ đơn thuần sử dụng trung bình các mô hình cục bộ từ các máy khách mà nó sử dụng trọng số cho từng mô hình cục bộ dựa trên độ chính xác quy tắc logic của chúng. Điều này đảm bảo rằng các máy khách có quy tắc mạnh hơn sẽ có ảnh hưởng lớn hơn trong mô hình toàn cục.

Server	Lựa chọn kết nối logic tự động ở cấp khách hàng
	Tổng hợp quy tắc khách hàng
	Gán trọng số khách hàng
Client	Lựa chọn kết nối logic tự động ở cấp mẫu
	Mạng dựa trên Entropy

Bảng 2.1: Thiết kế tổng quan của

Quy trình hoạt động của LR-XFL

Đoạn mã giả trong 1 minh họa quy trình thuật toán đằng sau mô hình LR-XFL.

Algorithm 1 Thuật toán trong LR-XFL

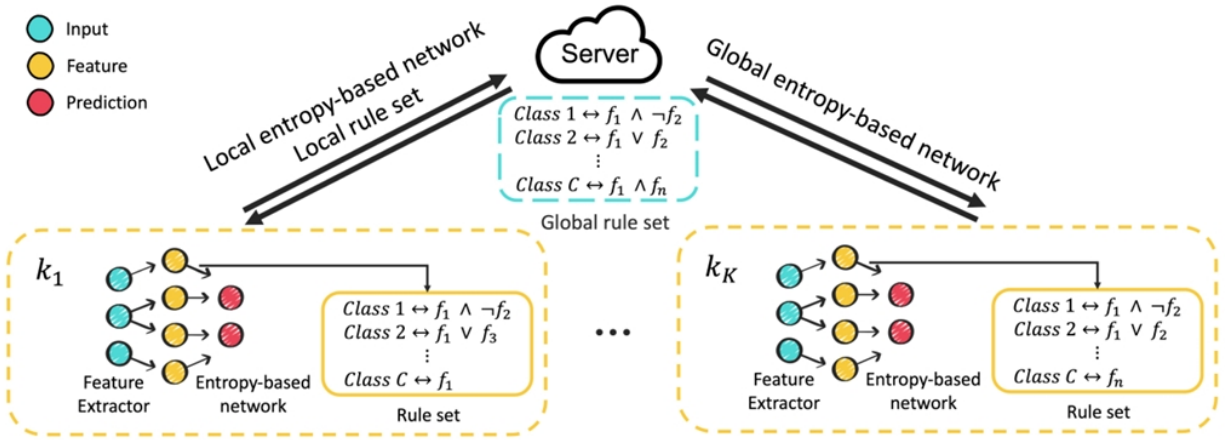
Input: K máy khách, mỗi máy khách có một tập dữ liệu cục bộ; một máy chủ giữ tập dữ liệu để kiểm thử và xác thực logic.

Output: Các quy tắc logic toàn cục cho máy chủ; mô hình và quy tắc logic cục bộ cho từng máy khách.

- 1: **while** Mô hình toàn cục chưa đạt được hiệu năng mục tiêu trên tập dữ liệu xác thực
 and số vòng huấn luyện tối đa chưa được hoàn thành **do**
 - 2: **for** mỗi máy khách FL k , $k \in \{1, \dots, K\}$ **do**
 - 3: Tiến hành huấn luyện mô hình cục bộ;
 - 4: Chọn bộ kết nối logic phù hợp cho các quy tắc logic cục bộ;
 - 5: Tạo ra các quy tắc logic r_k^c cho từng lớp $c \in \{1, \dots, C\}$;
 - 6: Tải lên mô hình cục bộ và các quy tắc logic tới máy chủ FL;
 - 7: **end for**
 - 8: **Tại máy chủ FL:**
 - 9: Lựa chọn bộ kết nối logic toàn cục dựa trên các quy tắc logic từ máy khách;
 - 10: Lựa chọn và tổng hợp các quy tắc logic cục bộ từ máy khách;
 - 11: Tính toán và gán trọng số $\{w_1, \dots, w_K\}$ cho các máy khách dựa trên hiệu quả quy tắc logic của chúng;
 - 12: Tổng hợp các mô hình cục bộ thành mô hình toàn cục dựa trên các trọng số đã gán;
 - 13: Gửi mô hình toàn cục lại cho các máy khách;
 - 14: **for** mỗi máy khách FL k **do**
 - 15: Nhận mô hình toàn cục và tiếp tục huấn luyện cho vòng tiếp theo;
 - 16: **end for**
 - 17: **end while**
-

Tổng quan về LR-XFL

LR-XFL là khung học liên kết dựa trên giải thích logic, cho phép xây dựng các quy tắc từ dữ liệu mà không cần truy cập trực tiếp vào dữ liệu của máy khách. Tại mỗi máy khách, mô hình dựa trên entropy được huấn luyện để tạo ra các quy tắc logic từ bộ dữ liệu cục bộ. Những quy tắc và cập nhật mô hình này sau đó được gửi lên máy chủ. Máy chủ tổng hợp các quy tắc từ máy khách bằng cách chọn phép nối logic phù hợp (AND hoặc OR) và gán trọng số dựa trên chất lượng quy tắc từ mỗi máy khách. Quy trình lặp lại cho đến khi mô hình đạt hiệu quả mong muốn hoặc đủ số vòng lặp đã đặt ra. Hình 2.1 mô tả kiến trúc và quy trình hoạt động của .



Hình 2.1: Kiến trúc và quy trình hoạt động của LR-XFL

Xác định kết nối logic OR hoặc AND

• Tại sao cần xác định kết nối OR hay AND?

- Trong mô hình LR-XFL, việc xác định kết nối logic OR hay AND rất cần thiết để giảm thiểu xung đột giữa các đặc trưng và duy trì hiệu quả của các quy tắc suy luận. Khi các đặc trưng xung đột lẫn nhau và không thể cùng tồn tại trong một mẫu dữ liệu, việc kết nối bằng AND sẽ dẫn đến quy tắc không hợp lý và không phản ánh đúng bản chất của dữ liệu. Ví dụ, trong việc phân loại số, nếu các đặc trưng chỉ ra rằng một số là “số lẻ” hoặc “số chẵn”, thì sử dụng AND sẽ tạo ra một quy tắc không chính xác, vì một số không thể vừa lẻ vừa chẵn. Do đó, trong những trường hợp xung đột như vậy, OR là lựa chọn kết nối phù hợp hơn để đảm bảo tính chính xác của quy tắc.
- Bên cạnh đó, kết nối logic đúng còn đóng vai trò quan trọng trong việc đảm bảo độ chính xác của mô hình. Nếu các đặc trưng có thể cùng tồn tại mà vẫn được nối bằng OR, mô hình sẽ dễ bị quá khái quát và suy giảm tính chính xác. Chẳng hạn, trong bài toán phân loại động vật, hai đặc trưng “cánh” và “mỏ” có thể đồng thời

xuất hiện ở loài chim. Vì vậy, nếu dùng AND để kết nối, mô hình có thể xây dựng một quy tắc chính xác hơn cho loài chim. Trong khi đó, nếu kết nối các đặc trưng này bằng OR, quy tắc sẽ không chỉ áp dụng cho loài chim mà còn cho các loài có một trong hai đặc trưng này, như dơi có cánh nhưng không có mỏ, làm giảm độ chính xác của mô hình.

- Cuối cùng, việc lựa chọn kết nối logic phù hợp không chỉ ảnh hưởng đến từng quy tắc cục bộ mà còn tối ưu hóa hiệu quả của mô hình FL khi tổng hợp các quy tắc từ các máy khách khác nhau. Kết nối đúng giữa các đặc trưng giúp máy chủ FL tổng hợp các quy tắc mà không làm suy giảm hiệu quả hoặc độ chính xác của mô hình toàn cục. Khi các quy tắc cục bộ được kết nối theo cách tối ưu, mô hình FL có thể tận dụng tốt hơn thông tin từ các máy khách và xây dựng quy tắc toàn cục phản ánh chính xác các đặc trưng chung mà vẫn duy trì độ chính xác cao, giúp thực hiện quá trình huấn luyện và dự đoán một cách hiệu quả và chính xác.

• Xác định kết nối phù hợp

Các tác giả đã đưa ra thiết kế cho hai loại ma trận nhằm khám phá các xung đột tiềm ẩn giữa các đặc trưng.

1. **Ma trận đồng hiện dương:** ghi lại số lần các đặc trưng f_i và f_j cùng xuất hiện trong các luật có dạng $f_i \wedge \dots \wedge f_j$. Nếu các đặc trưng f_i và f_j cùng xuất hiện trong các luật t lần, thì $p_{ij} = p_{ji} = t$.

$$M_{\text{pos}} = \begin{bmatrix} p_{11} & \dots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \dots & p_{nn} \end{bmatrix}. \quad (2.1)$$

2. **Ma trận đồng hiện âm:** ghi lại số lần các đặc trưng f_i và f_j cùng xuất hiện trong các luật có dạng $f_i \wedge \dots \wedge \neg f_j$. Nếu các đặc trưng f_i và $\neg f_j$ xuất hiện trong luật t lần, thì $q_{ij} = t$. Tuy nhiên, khác với ma trận đồng hiện dương, trong ma trận đồng hiện âm, $q_{ij} \neq q_{ji}$. Bởi q_{ij} ghi lại số lần f_i và $\neg f_j$ cùng xuất hiện trong một luật, trong khi q_{ji} ghi lại số lần xuất hiện của f_j và $\neg f_i$.

$$M_{\text{neg}} = \begin{bmatrix} q_{11} & \dots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{n1} & \dots & q_{nn} \end{bmatrix}. \quad (2.2)$$

Tác giả đề xuất hai tiêu chí để đánh giá mức độ xung đột đặc trưng thông qua ma trận đồng hiện dương M_{pos} và ma trận đồng hiện âm M_{neg} .

1. **Tính chéo (Diagonality):** Chỉ số này cho biết mức độ độc lập của một đặc trưng trong các quy tắc, phản ánh tần suất mà một đặc trưng có thể xuất hiện trong một quy tắc mà không cần đi kèm các đặc trưng khác. Nếu một đặc trưng có tính chéo cao, ta biết rằng có thể đưa ra giải thích chỉ dựa trên đặc trưng đó mà không cần phụ thuộc vào các đặc trưng khác. Khi tính chéo của một đặc trưng cao, mô

hình có thể cân nhắc sử dụng phép OR để tận dụng khả năng hoạt động độc lập của đặc trưng này, từ đó giúp giảm thiểu xung đột khi kết nối.

$$D = \frac{\sum_{i=1}^n p_{ii}}{\sum_{i=1}^n \sum_{j=1}^n p_{ij}}. \quad (2.3)$$

2. Tính riêng biệt (Exclusivity): Chỉ số này cho biết mức độ mà một đặc trưng loại trừ sự hiện diện của các đặc trưng khác trong các quy tắc. Tính riêng biệt cao cho thấy một đặc trưng thường xuất hiện cùng với dạng phủ định của đặc trưng khác, được biểu diễn dưới dạng $f_i \wedge \neg f_j \wedge \dots \wedge \neg f_n$. Tính riêng biệt cao thể hiện sự xung đột cao giữa các đặc trưng. Ví dụ, trong phân loại số chẵn và lẻ, tính riêng biệt sẽ cao nếu một đặc trưng “số lẻ” xuất hiện thì đi kèm với đó là “ \neg số chẵn”, cho biết đặc trưng “số chẵn” đã bị loại trừ. Khi tính riêng biệt cao, kết nối OR thường được ưu tiên vì các đặc trưng xung đột, khó cùng tồn tại trong một quy tắc.

$$E = \frac{\max_{i=1}^n \left(\sum_{j=1}^n q_{ij} \right)}{\sum_{m=1}^M l_{r_m}}. \quad (2.4)$$

Nếu tính chéo hoặc tính riêng biệt của đặc trưng vượt qua ngưỡng siêu tham số đã thiết lập, nghĩa là khả năng các đặc trưng xung đột nhau rất cao, sẽ chọn kết nối OR. Phép nối OR giúp giảm thiểu xung đột, tránh trường hợp đưa ra các quy tắc thiếu chính xác. Ngược lại, nếu cả hai chỉ số thấp hơn ngưỡng này, điều này cho thấy các đặc trưng không có xung đột, mô hình sẽ chọn AND. Kết nối AND giúp kết hợp đặc trưng khi chúng có thể cùng tồn tại mà không ảnh hưởng đến độ chính xác của mô hình.

Tổng hợp quy tắc liên kết

Trong mỗi vòng huấn luyện FL, máy chủ nhận các mô hình và quy tắc logic cục bộ từ các máy khách. Đầu tiên, máy chủ xác định kết nối logic phù hợp theo phương pháp đã mô tả. Sau đó, máy chủ chọn tập hợp quy tắc tối ưu hóa hiệu quả dựa trên K quy tắc liên quan đến lớp c từ K máy khách. Tuy nhiên, số quy tắc thực tế ít hơn K do:

1. Không phải máy khách nào cũng tạo ra quy tắc đáng tin cậy, vì các quy tắc từ mô hình không đạt ngưỡng độ chính xác sẽ bị loại.
2. Một số quy tắc không hỗ trợ lớp c .
3. Các quy tắc trùng lặp giữa các máy khách.

Để tối ưu hóa tổ hợp quy tắc, thuật toán tìm kiếm beam được áp dụng. Đây là phương pháp tham lam chỉ giữ lại t chuỗi tốt nhất tại mỗi bước, cân bằng giữa chi phí tính toán và chất lượng giải pháp. Trong LR-XFL, các chuỗi quy tắc được xếp hạng dựa trên độ chính xác trên tập kiểm định của máy chủ FL.

Phân bổ trọng số cho máy khách

Trong FL, việc gán trọng số phù hợp cho các máy khách là rất quan trọng, đặc biệt khi xử lý dữ liệu thiên lệch hoặc nhiễu. LR-XFL giới thiệu phương pháp tính trọng số dựa trên các quy tắc logic của từng máy khách. Trọng số của máy khách trong một vòng huấn luyện tỷ lệ thuận với tần suất các quy tắc của nó được máy chủ lựa chọn.

Giả sử máy khách k xây dựng C quy tắc cho C lớp trong một vòng huấn luyện, với p quy tắc được tích hợp vào tập toàn cục. Trọng số của máy khách k được tính như sau:

$$w_k = \frac{p_k}{\sum_{i=1}^K p_i}. \quad (2.5)$$

Nếu máy khách không tạo ra quy tắc hợp lệ hoặc các quy tắc không được chọn, thì $w_k = 0$. Quy tắc có thể bị loại do: (1) độ chính xác thấp, hoặc (2) tích hợp làm giảm hiệu quả của tập quy tắc toàn cục. Giá trị w_k cao phản ánh đóng góp đáng kể của máy khách vào các quy tắc trên nhiều lớp và được ưu tiên gán trọng số cao hơn.

Phân tích độ phức tạp thời gian

Trong quá trình tạo quy tắc cho K máy khách, mỗi máy khách có N điểm dữ liệu, độ phức tạp cho một máy khách tạo ra quy tắc là $O(N)$. Xét cho C lớp, quá trình tổng hợp quy tắc cục bộ bao gồm việc xếp hạng quy tắc với độ phức tạp là $O(N \log N)$. Sau khi các quy tắc được xếp hạng, hệ thống sẽ thực hiện quá trình bao hàm lặp lại để thêm từng quy tắc vào tập hợp tổng hợp với độ phức tạp là $O(N)$. Do đó, độ phức tạp tổng thể cho quá trình tổng hợp quy tắc cục bộ là $O(N \log N)$. Trong giai đoạn tổng hợp quy tắc toàn cục, tìm kiếm beam được sử dụng với độ rộng beam là b , dẫn đến độ phức tạp tệ nhất là $O(bK^2)$ cho mỗi lớp. Tuy nhiên, vì N lớn hơn đáng kể so với K , độ phức tạp thời gian cho toàn bộ quá trình tạo và tổng hợp quy tắc là $O(N \log N)$.

2.9 Kịch bản thực nghiệm

Ngôn ngữ lập trình: Python

2.9.1 Mục tiêu của thí nghiệm

Nhằm đánh giá hiệu quả của trên nhiều loại tập dữ liệu và trong các tình huống dữ liệu khác nhau. Mục tiêu chính bao gồm:

1. So sánh độ chính xác và độ tin cậy của với các phương pháp khác.
2. Đánh giá khả năng hoạt động của trong điều kiện dữ liệu bị nhiễu.

2.9.2 Thiết lập dữ liệu

Bộ dữ liệu được phân chia để tạo điều kiện cho cả hoạt động dựa trên máy chủ và dựa trên máy khách. Cụ thể, 20% dữ liệu được phân bổ cho máy chủ: 10% để kiểm định việc lựa chọn khách hàng dựa trên các quy tắc họ tạo ra, và 10% còn lại để kiểm tra các quy tắc toàn cục sau khi được tổng hợp và đánh giá độ chính xác của chúng. Trong đó:

- **MNIST (Even/Odd) và CUB:**

- Loại dữ liệu: Dạng “ảnh \rightarrow đặc trưng \rightarrow lớp”.
- Mục tiêu: Thử nghiệm khả năng phân loại dựa trên các đặc trưng hình ảnh.

- **V-Dem và MIMIC-II:**

- Loại dữ liệu: Dạng bảng (tabular).
- Mục tiêu: Thử nghiệm phân loại trong môi trường dữ liệu dạng bảng, ánh xạ trực tiếp dữ liệu đầu vào tới lớp phân loại.

Với từng tập dữ liệu, thí nghiệm được chia thành hai thiết lập:

1. **Thiết lập tập trung (Centralised):** Học tập trên dữ liệu không phân tán, có quyền truy cập toàn bộ dữ liệu.
2. **Thiết lập liên kết (Federated):** Mô phỏng môi trường phân tán, trong đó dữ liệu được chia nhỏ và phân phối trên các máy khách.

2.9.3 Thí nghiệm dữ liệu nhiều

Để kiểm tra tính ổn định của LR-XFL khi đối mặt với dữ liệu bị nhiễu, một phần dữ liệu tại các máy khách được làm nhiễu bằng cách xáo trộn nhãn của chúng một cách ngẫu nhiên. Tỷ lệ dữ liệu nhiễu tăng dần từ 20% đến 80%, nhằm đánh giá khả năng hoạt động của LR-XFL trong môi trường dữ liệu không đồng nhất và phức tạp.

2.9.4 Đối chiếu các phương pháp cơ sở

Vì hiện tại chưa có mô hình nào được đặc biệt thiết kế để tạo và tổng hợp các quy tắc từ các mô hình cục bộ trong FL, đồng thời tận dụng các quy tắc này cho việc tổng hợp như LR-XFL, trong thí nghiệm này, để đánh giá hiệu quả mô hình trên cùng một tập dữ liệu, nhóm tác giả so sánh LR-XFL với ba phương pháp cơ sở khác:

1. **Học tập trung (Centralised Learning):** Phương pháp này sử dụng giải thích logic dựa trên entropy của mạng nơ-ron trong môi trường tập trung, nghĩa là toàn bộ dữ liệu đều có thể được truy cập trực tiếp. Mô hình phân tích dữ liệu và tạo ra các quy tắc ở cấp độ mẫu, sau đó tổng hợp các quy tắc này thành quy tắc chung cho từng lớp bằng toán tử OR.

2. **Cây quyết định phân tán (Distributed Decision Tree - DDT):** Trong phương pháp DDT, mỗi máy khách tự xây dựng một cây quyết định riêng. Sau khi huấn luyện cục bộ, các máy khách sẽ gửi cây quyết định của mình lên máy chủ. Máy chủ sẽ thu thập tất cả các cây quyết định cục bộ và đánh giá chúng trên một tập dữ liệu xác thực chung để xác định cây đem lại hiệu quả tốt nhất. Cây cục bộ này sau đó được sử dụng làm cây quyết định toàn cục, áp dụng cho cả quá trình dự đoán và tạo quy tắc. Tuy nhiên, vì mô hình cây quyết định không yêu cầu một quá trình lặp lại, các máy khách sẽ chỉ sử dụng cây quyết định toàn cục mà không thực hiện thêm bước tối ưu hóa nào.
3. **FedAvg-Logic:** Đây là một phiên bản điều chỉnh của FedAvg, sử dụng mạng nơ-ron dựa trên entropy làm mô hình cơ sở. Toán tử OR được sử dụng để kết nối cho việc tổng hợp quy tắc. Các máy khách được gán trọng số như nhau, bất kể chất lượng quy tắc mà chúng tạo ra.

2.9.5 Tiêu chí đánh giá

Nhóm tác giả sử dụng 3 chỉ số đánh giá trong bài báo: độ chính xác mô hình, độ chính xác quy tắc, và độ tin cậy quy tắc. Để đánh giá hiệu năng mô hình, sự nhất quán giữa các chỉ số này rất quan trọng. Tuy nhiên, các chỉ số này đánh giá các khía cạnh khác nhau của mô hình và có thể mang lại kết quả khác nhau trong việc đánh giá tổng thể mô hình.

1. **Độ chính xác mô hình:** Độ chính xác mô hình cho biết tổng số dự đoán đúng với nhãn thật của mô hình trên tổng số dự đoán, bao gồm cả dự đoán đúng và dự đoán sai được đưa ra. Chỉ số này thể hiện khả năng dự đoán đúng nhãn của mô hình đối với một bài toán phân loại cụ thể.

$$\text{Độ chính xác mô hình} = \frac{\text{Số dự đoán đúng}}{\text{Tổng số dự đoán}} \quad (2.6)$$

Mô hình có độ chính xác mô hình cao thường chỉ ra rằng mô hình này hoạt động tốt trong việc phân loại đúng đối tượng. Tuy nhiên, điều này không đảm bảo rằng các quy tắc mà mô hình sinh ra là hợp lý hoặc dễ hiểu.

2. **Độ chính xác quy tắc:** Độ chính xác quy tắc đo lường mức độ chính xác của các quy tắc được sinh ra từ mô hình, tức là kiểm tra xem các quy tắc có phù hợp với nhãn thực tế hay không. Chỉ số này đánh giá khả năng mô hình sinh ra các quy tắc logic phù hợp với dữ liệu gốc.

$$\text{RuleAcc}_c = \frac{p + q}{P + Q} \quad (2.7)$$

- P : Số điểm dữ liệu thuộc lớp c trong tập dữ liệu.
- Q : Số điểm dữ liệu không thuộc lớp c .
- p : Số điểm trong P mà thỏa mãn các điều kiện trong quy tắc r_c .

- q : Số điểm trong Q mà không thỏa mãn các điều kiện trong quy tắc r_c .

$RuleAcc_c$ được tính là độ chính xác của một quy tắc cho lớp c . Để tính độ chính xác quy tắc tổng thể, ta tính trung bình giá trị $RuleAcc_c$ của tất cả các lớp.

$$\text{Độ chính xác quy tắc tổng thể} = \frac{1}{C} \sum_{c=1}^C RuleAcc_c \quad (2.8)$$

3. **Độ tin cậy quy tắc:** Độ tin cậy của quy tắc đánh giá sự nhất quán giữa các dự đoán từ quy tắc và các dự đoán từ mô hình. Đây là một phép đo quan trọng trong trường hợp muốn đảm bảo các quy tắc mà mô hình sinh ra không chỉ chính xác mà còn có tính nhất quán với các dự đoán của mô hình.

$$\text{Độ tin cậy quy tắc}_c = \frac{p' + q'}{P' + Q'} \quad (2.9)$$

- P' là số điểm dữ liệu được mô hình dự đoán thuộc lớp c .
- Q' là số điểm dữ liệu được mô hình không thuộc lớp c .
- p' là số điểm trong P' mà thỏa mãn các điều kiện trong quy tắc r_c .
- q' là số điểm trong Q' mà không thỏa mãn các điều kiện trong quy tắc r_c .

Độ tin cậy quy tắc cao chỉ ra rằng các quy tắc không chỉ phù hợp với nhân thực tế mà còn khớp với cách mô hình đưa ra quyết định. Tuy nhiên, nếu độ tin cậy quy tắc thấp, có thể mô hình dựa vào các quy tắc không nhất quán để đưa ra quyết định, dù mô hình vẫn đạt độ chính xác mô hình cao.

Để đảm bảo có cái nhìn khách quan và toàn diện nhất về hiệu năng của mô hình, chúng ta không nên chỉ xem xét từng chỉ số riêng lẻ, mà còn cần phân tích mối liên kết và tính nhất quán giữa các chỉ số đánh giá.

- **Sự nhất quán giữa độ chính xác mô hình và độ chính xác quy tắc:** Khi độ chính xác mô hình cao nhưng độ chính xác quy tắc thấp, mô hình có thể đưa ra dự đoán chính xác, nhưng các quy tắc được tạo ra để giải thích quyết định lại không đáng tin cậy. Ngược lại, nếu độ chính xác quy tắc cao nhưng độ chính xác mô hình thấp, các quy tắc được sinh ra có thể chính xác nhưng mô hình lại không thực hiện tốt trong việc phân loại. Trong trường hợp này, ta thấy có sự thiếu nhất quán giữa kết quả phân loại và tính giải thích của mô hình.
- **Sự nhất quán giữa độ chính xác quy tắc và độ tin cậy quy tắc:** Độ chính xác quy tắc cao thường thường sẽ đi kèm độ tin cậy quy tắc cao, nghĩa là các quy tắc được tạo ra là hợp lý và được sử dụng đúng cách. Tuy nhiên, vẫn có thể xảy ra trường hợp độ tin cậy của quy tắc thấp dù độ chính xác quy tắc cao. Khi đó ta thấy rằng, dù mô hình có thể học được các quy tắc chính xác, nhưng khi đưa ra dự đoán cuối cùng, mô hình lại không áp dụng chúng một cách phù hợp.

- **Sự nhất quán giữa độ chính xác mô hình và độ tin cậy quy tắc:** Khi độ chính xác mô hình cao nhưng độ tin cậy quy tắc thấp, mô hình đưa ra dự đoán tốt nhưng lại thiếu các quy tắc hợp lý hoặc phù hợp để giải thích cách đưa ra dự đoán. Ngược lại, nếu độ tin cậy quy tắc cao nhưng độ chính xác mô hình thấp, các quy tắc đáng tin cậy nhưng không giúp cải thiện hiệu quả phân loại của mô hình.

Nhóm sinh viên rút ra cái nhìn tổng quan về các chỉ số đánh giá như sau:

- Khi cả ba chỉ số có sự nhất quán cao, mô hình hoạt động tốt. Điều này cho thấy mô hình không chỉ mạnh về hiệu năng mà còn đáng tin cậy và dễ giải thích.
- Khi có sự không nhất quán giữa các chỉ số, mô hình có thể đạt kết quả tốt ở một số chỉ số nhưng lại bộc lộ hạn chế ở khía cạnh khác.
- Các chỉ số đánh giá không chỉ phản ánh chất lượng của mô hình mà còn cho thấy khả năng áp dụng của nó trong thực tế. Một mô hình có chỉ số đồng đều và nhất quán sẽ phù hợp hơn trong các tình huống đòi hỏi độ tin cậy cao, chẳng hạn như y tế, tài chính hoặc tự động hóa.
- Sự không đồng nhất giữa các chỉ số cũng có thể giúp phát hiện vấn đề trong quá trình huấn luyện, như dữ liệu không đồng nhất, thiếu trọng số thích hợp cho các quy tắc hoặc sự phức tạp không cần thiết trong cấu trúc mô hình.

Do đó, việc kết hợp cả ba chỉ số này sẽ mang lại một góc nhìn toàn diện hơn về hiệu năng của mô hình.

2.9.6 Kết quả

Bảng 2.2 trình bày kết quả so sánh giữa LR-XFL và các phương pháp cơ sở. Từ đây, ta có thể rút ra một vài nhận xét như sau.

- **Độ chính xác mô hình:** Kết quả thí nghiệm cho thấy LR-XFL đạt độ chính xác dự đoán cao nhất trong các phương pháp học liên kết. So với Học tập trung, LR-XFL cho thấy hiệu năng ngang bằng hoặc vượt trội, nhờ khả năng tự động chọn kết nối logic ở máy khách. So với FedAvg-Logic, lợi thế của LR-XFL nằm ở cơ chế gán trọng số, ưu tiên các máy khách có nhiều đóng góp chất lượng vào bộ quy tắc toàn cục. Trong khi đó, DDT chỉ chọn mô hình cục bộ tốt nhất mà không tích hợp thông tin từ các máy khách khác. Độ chính xác mô hình trung bình của LR-XFL cao hơn FedAvg-Logic 1.19% và DDT 3.58%.
- **Độ chính xác và độ tin cậy quy tắc:** Xét về độ chính xác và độ tin cậy quy tắc, LR-XFL cho kết quả cao hơn hoặc tương đương so với FedAvg-Logic và DDT trên 3/4 bộ dữ liệu. Kết quả ấn tượng này có được nhờ khả năng tự động lựa chọn kết nối logic thay vì chỉ dùng kết nối OR cố định. Bên cạnh đó, cơ chế chọn lọc quy tắc và gán trọng số cho các máy khách cũng góp phần tạo nên lợi thế này. Độ chính xác quy tắc

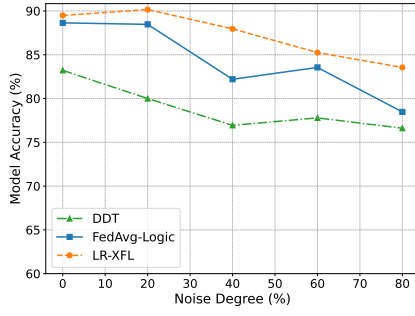
trung bình của LR-XFL cao hơn FedAvg-Logic 5.81% và DDT 0.27%. Về độ tin cậy quy tắc, giá trị trung bình của LR-XFL cao hơn FedAvg-Logic 5.41%. Chỉ số này không được áp dụng với DDT vì mô hình này dựa hoàn toàn vào quy tắc, nên độ tin cậy quy tắc của DDT luôn đạt 100%.

		Học tập trung	DDT	FedAvg-Logic	LR-XFL
MNIST	Độ chính xác mô hình	99.84%	99.78%	99.80%	99.96%
	Độ chính xác quy tắc	99.84%	99.71%	99.84%	99.95%
	Độ tin cậy quy tắc	99.95%	-	99.89%	99.97%
CUB	Độ chính xác mô hình	82.20%	83.22%	88.64%	89.49%
	Độ chính xác quy tắc	91.71%	87.87%	74.89%	90.67%
	Độ tin cậy quy tắc	99.71%	-	98.61%	99.64%
V-Dem	Độ chính xác mô hình	93.32%	92.55%	90.95%	93.08%
	Độ chính xác quy tắc	90.84%	92.52%	86.71%	93.08%
	Độ tin cậy quy tắc	94.59%	-	81.11%	100.00%
MIMIC-II	Độ chính xác mô hình	76.96%	76.40%	80.89%	82.02%
	Độ chính xác quy tắc	66.56%	67.80%	68.27%	65.15%
	Độ tin cậy quy tắc	77.24%	-	79.77%	79.21%

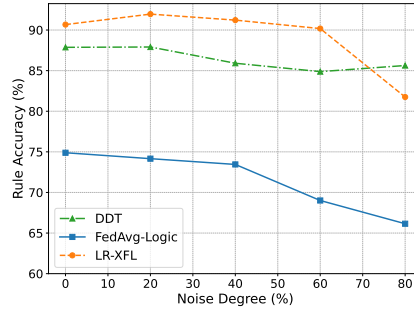
Bảng 2.2: Kết quả thí nghiệm. Kết quả tốt nhất được đánh dấu bằng chữ in đậm. Ký hiệu ‘-’ cho biết chỉ số đánh giá không phù hợp với phương pháp tương ứng.

- **Khả năng kháng nhiễu:** Bài báo cũng tiến hành các thí nghiệm để nghiên cứu khả năng kháng nhiễu của DDT, FedAvg-Logic và LR-XFL trên bộ dữ liệu CUB. Hình 2.2 minh họa kết quả thí nghiệm dưới các thiết lập mức độ nhiễu khác nhau. Có thể thấy, khi mức độ nhiễu tăng lên, cả độ chính xác mô hình và độ chính xác quy tắc đều giảm ở tất cả các phương pháp. Tuy nhiên, độ tin cậy của quy tắc vẫn tương đối ổn định, cho thấy sự liên kết giữa các quy tắc và các dự đoán mô hình vẫn được duy trì. Điểm ấn tượng là LR-XFL luôn đạt độ chính xác cao nhất ở mọi mức nhiễu và suy giảm hiệu năng ít hơn so với FedAvg-Logic và DDT. Đặc biệt, LR-XFL giữ được độ chính xác quy tắc ổn định ở mức nhiễu lên đến 60% nhờ vào cơ chế chọn lọc quy tắc và gán trọng số cho các máy khách quan trọng. Ở mức nhiễu rất cao như (80%), DDT lại cho kết quả tốt nhất vì chỉ sử dụng mô hình từ máy khách có hiệu năng cao nhất, nhưng cách này làm giảm khả năng chia sẻ thông tin giữa các máy khách. Kết quả cho thấy LR-XFL hoạt động ổn định hơn trong môi trường dữ liệu bị nhiễu, còn DDT phù hợp hơn khi mức nhiễu cực cao nhưng phải đánh đổi khả năng tận dụng thông tin từ các máy khách khác.

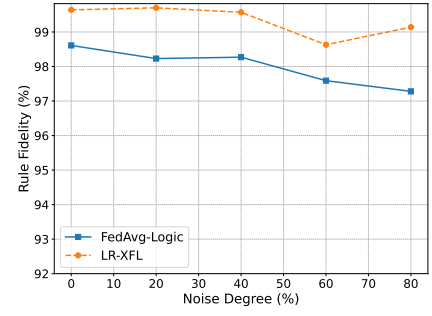
Hình 2.2: Kết quả thí nghiệm dưới các mức độ nhiễu khác nhau.



(a) Độ chính xác mô hình



(b) Độ chính xác quy tắc



(c) Độ tin cậy quy tắc

- **Nghiên cứu loại bỏ (ablation)¹:** Nhóm tác giả cũng đã thực hiện một nghiên cứu để đánh giá xem việc tự động chọn kết nối logic ảnh hưởng thế nào đến hiệu quả của mô hình LR-XFL. Trong nghiên cứu này, họ đã thay thế kết nối logic AND, được tự động chọn cho tập dữ liệu CUB, bằng phép OR rồi đối chiếu kết quả với thiết lập ban đầu. Thí nghiệm cho thấy, khi sử dụng kết nối logic OR, kết quả giảm đáng kể trên cả ba chỉ số đánh giá. Những phát hiện này nhấn mạnh vai trò quan trọng của phương pháp tự động lựa chọn kết nối logic, cũng như sự cần thiết của việc phân tích kỹ lưỡng các đặc trưng trước khi xác định quy tắc kết nối. Việc xem xét mức độ phụ thuộc và tách biệt giữa các đặc trưng khi thiết lập quy tắc là yếu tố cốt lõi để đạt được các kết quả chính xác và đáng tin cậy, cả về chỉ số đánh giá lẫn khả năng giải thích.

Bảng 2.3: Kết quả nghiên cứu loại bỏ

	LR-XFL (ablated)	LR-XFL
Độ chính xác mô hình	87.96%	89.49%
Độ chính xác quy tắc	76.29%	90.67%
Độ tin cậy quy tắc	98.83%	99.64%

¹Hiện chưa có thuật ngữ tiếng Việt chính thức cho từ ablation. Theo một nghiên cứu của Học viện Công nghệ Hoàng gia KTH, nghiên cứu loại bỏ là phương pháp loại bỏ từng phần của mô hình, có thể là các lớp, nơ-ron, hoặc các đặc trưng trong tập dữ liệu sử dụng. Mục đích là để đánh giá tác động của từng thành phần cấu thành đến hiệu năng tổng thể [5].

Chương 3

Triển khai tái tạo thực nghiệm của tác giả

3.1 Giới thiệu chương trình và mô tả tập dữ liệu

3.1.1 Giới thiệu chương trình

3.1.1.1 Mã nguồn

- Mã nguồn do tác giả công bố tại địa chỉ: <https://github.com/Yanci87/LR-XFL>
- Mã nguồn chạy thực nghiệm của nhóm sinh viên: <https://github.com/maple1606/LR-XFL>

3.1.1.2 Môi trường thực nghiệm

- Card đồ họa: RTX 4060
- RAM: 16GB
- CPU: AMD Ryzen 7 7840H
- IDE: Visual Studio Code (VS Code)

3.1.2 Mô tả tập dữ liệu

3.1.2.1 Giới thiệu tập dữ liệu

Tất cả các tập dữ liệu được sử dụng trong nghiên cứu này đều có sẵn miễn phí, ngoại trừ MIMIC-II yêu cầu đăng ký trực tuyến để truy cập. Các tập dữ liệu này có thể dễ dàng được tải xuống từ các liên kết được cung cấp dưới đây. Nghiên cứu ban đầu sử dụng 4 tập dữ liệu: MNIST, CUB, MIMIC-II và V-Dem. Chi tiết về các tập dữ liệu đã được trình bày tại mục 2.5.

- MIMIC-II: <https://archive.physionet.org/mimic2>
- V-Dem: <https://www.v-dem.net/en/data/data/v-dem-dataset-v111>.
- MNIST: <http://yann.lecun.com/exdb/mnist>.
- CUB-200-2011: <http://www.vision.caltech.edu/visipedia/CUB-200-2011.html>.

3.1.2.2 Tiền xử lý dữ liệu

Ban đầu, nhóm sinh viên gặp khó khăn trong việc xử lý dữ liệu vì tác giả của bài báo không đề cập chi tiết về quy trình này. Hơn nữa, các tệp dữ liệu trên GitHub đã bị ẩn do tệp `.gitignore`, khiến việc truy cập thêm phần khó khăn. Sau khi liên hệ trực tiếp với tác giả qua email (Hình 3.1), họ giải thích rằng dữ liệu đã được xử lý hoàn toàn dựa trên phương pháp được trình bày trong một nghiên cứu khác [2], vốn là nghiên cứu tiền đề cho bài báo này mà nhóm sẽ trình bày ở chương 5. Chi tiết như sau:

**Re: Request for Dataset Access for
Replicating Results in [LR-XFL:
Logical Reasoning-based
Explainable Federated Learning]**

Hi Nguyen Thi Huong,

Thanks for your email. For the data preprocessing. I followed this GitHub project. https://github.com/pietrobarbiero/logic_explained_networks/tree/Experiments?tab=readme-ov-file. You can find the preprocessing code in the `./experiments` folder.

Best regards,
Yanci

Hình 3.1: Thư trả lời của tác giả

- **MIMIC-II** Trong thí nghiệm này, nhóm tác giả đã loại bỏ các thông tin không được ẩn danh, các đặc trưng dạng văn bản, các đầu vào chuỗi thời gian, cũng như các quan sát bị thiếu dữ liệu. Các đặc trưng liên tục được rời rạc hóa thành các danh mục mã hóa dạng one-hot. Sau bước tiền xử lý, không gian đầu vào C được tạo thành với $k = 90$ đặc trưng chính. Nhiệm vụ đặt ra là xác định bệnh nhân sẽ phục hồi hay tử vong sau khi nhập ICU.
- **V-Dem** Trong thí nghiệm này, tác giả xây dựng mô hình phân loại nhị phân nhằm mục đích nhận biết nền dân chủ bầu cử và không bầu cử. Ký hiệu C_1 và C_2 là các không gian tương ứng với các mức độ kích hoạt của hai cấp độ khái niệm kể trên. Hai bộ phân loại f_1 và f_2 được huấn luyện để học phép ánh xạ $C_1 \rightarrow C_2 \rightarrow Y$. Các giải thích được đưa ra cho bộ phân loại f_2 theo các khái niệm $c_2 \in C_2$.

- **MNIST (Chẵn/Lẻ)** Giả sử $Y \subset \{0, 1\}^2$, nhóm tác giả quan tâm đến việc xác định liệu một chữ số là số chẵn hay số lẻ, đồng thời giải thích cách phân loại này dựa trên các nhãn chữ số (các khái niệm trong C). Phép ánh xạ $X \rightarrow C$ được cung cấp bởi một bộ phân loại ResNet10 g được huấn luyện từ đầu. Trong khi đó, bộ phân loại f được sử dụng để học đồng thời phép ánh xạ cuối cùng và lời giải thích dưới dạng hàm $C \rightarrow Y$.
- **CUB** Tập dữ liệu này sử dụng 312 thuộc tính nhị phân để mô tả các đặc điểm trực quan (màu sắc, hoa văn, hình dạng) của các bộ phận cụ thể (mỏ, cánh, đuôi, v.v.) trên mỗi hình ảnh chim. Tuy nhiên, các chú thích về thuộc tính này có độ nhiễu khá cao. Để khắc phục, các thuộc tính đã được xử lý để giảm nhiễu bằng cách sử dụng các chú thích ở cấp độ lớp. Sau quá trình xử lý, tổng cộng 108 thuộc tính (tức là các khái niệm nhị phân thuộc C) được giữ lại. Việc ánh xạ từ hình ảnh đến các khái niệm thuộc tính ($X \rightarrow C$) được thực hiện bằng mô hình ResNet10 g được huấn luyện từ đầu. Trong khi đó, bộ phân loại f đảm nhận việc học hàm cuối cùng để ánh xạ từ không gian khái niệm đến các loài chim ($C \rightarrow Y$).

3.1.3 Tái tạo thực nghiệm của tác giả

Để thuận tiện cho việc đào sâu và hiểu rõ hơn về mã nguồn thực nghiệm, cũng như cho việc tùy chỉnh và mở rộng nghiên cứu, trước tiên, nhóm sinh viên đã tiến hành các bước sau:

1. **Fork repository của tác giả:** Repository mã nguồn của tác giả được fork vào tài khoản GitHub cá nhân để dễ dàng quản lý và chỉnh sửa, cũng như lưu lại các thay đổi cần thiết trong quá trình thực nghiệm.
2. **Clone repository về máy cá nhân:** Sau khi fork, repository đã được clone về máy tính cá nhân. Quá trình này giúp nhóm có thể chạy mã nguồn cục bộ và tiến hành các thay đổi, thử nghiệm trên môi trường máy tính riêng.
3. **Khám phá và đọc hiểu mã nguồn:** Sinh viên cố gắng đọc hiểu mã nguồn trong khả năng tốt nhất của mình để hiểu rõ cấu trúc, cách thức triển khai, và các phương pháp mà tác giả đã sử dụng trong nghiên cứu. Điều này cũng hỗ trợ việc tìm hiểu cách xử lý dữ liệu và tái hiện các kết quả thực nghiệm.

3.1.3.1 Khó khăn

Trong quá trình triển khai thực nghiệm, nhóm sinh viên đã gặp phải nhiều vấn đề đáng chú ý:

- **Tương thích phiên bản và thiết bị:** Nhóm gặp phải các lỗi liên quan đến sự không tương thích giữa phiên bản thư viện PyTorch, Torchvision, và pytorch-lightning, đặc biệt khi các phiên bản này không đồng nhất với mã nguồn cung cấp ban đầu.
- **Cấu trúc thư mục và lỗi import:** Do cấu trúc thư mục của mã nguồn không rõ ràng, Python liên tục không tìm thấy các module cần thiết.

- **Khó khăn với mô hình ResNet10:** Khi đưa tập dữ liệu CUB và MNIST vào ResNet10, lượng dữ liệu dạng ảnh là quá lớn, phải mất rất nhiều thời gian chạy mới đưa ra được kết quả.
- **Mô hình FedAdvg Logic:** Nhóm gặp khó khăn khi không biết cách triển khai mô hình này. Việc thiếu tài liệu hướng dẫn chi tiết cũng như cách chỉnh sửa mã nguồn để chạy thử nghiệm đã khiến nhóm không thể thực hiện được các thí nghiệm liên quan.

Bên cạnh đó, nhóm đã gặp phải lỗi liên quan đến việc sử dụng hàm `torch.stack()` để chồng các tensor.

`RuntimeError: Sizes of tensors must match except in dimension 0`

Cụ thể, lỗi xảy ra khi các tensor đầu vào có kích thước không đồng nhất dọc theo các trục, khiến quá trình stack không thể thực hiện được. Nguyên nhân là do dữ liệu đầu vào không được xử lý đồng bộ, dẫn đến sự khác biệt về kích thước giữa các tensor.

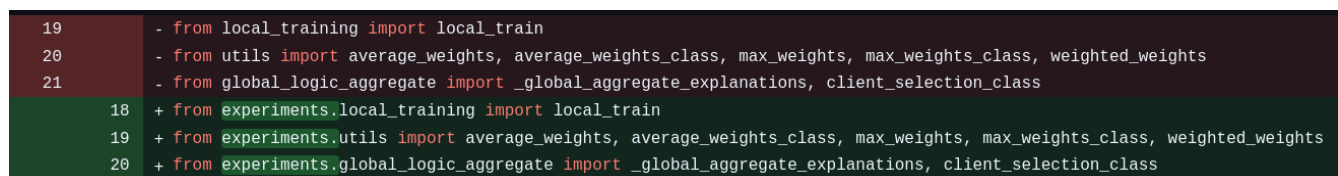
Những vấn đề này dẫn đến việc mã nguồn của tác giả không thể chạy mượt ngay từ 1-2 lần đầu tiên, buộc sinh viên phải can thiệp và chỉnh sửa. Mặt khác, quá trình này lại tạo cơ hội giúp sinh viên hiểu sâu hơn về cấu trúc mã nguồn và phương pháp của bài báo, hiểu rõ hơn về cách tác giả triển khai và áp dụng các ý tưởng nghiên cứu.

3.1.3.2 Giải pháp khắc phục

1. Để đảm bảo mã nguồn hoạt động ổn định và tránh các vấn đề không tương thích thiết bị hoặc khác biệt về API, nhóm đã sử dụng thư viện `pytorch-lightning` phiên bản **1.5.0**.

```
!pip install pytorch-lightning==1.5.0
```

2. Tiến hành sửa đường dẫn import các module trên toàn bộ tệp mã nguồn, một ví dụ được minh họa trong:



```

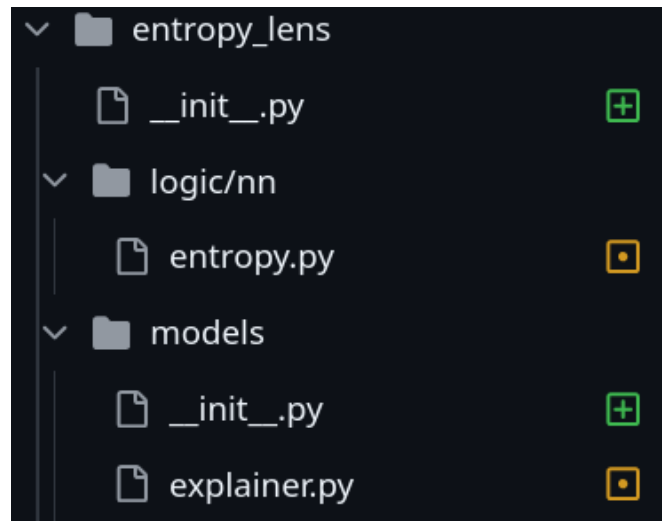
19 - from local_training import local_train
20 - from utils import average_weights, average_weights_class, max_weights, max_weights_class, weighted_weights
21 - from global_logic_aggregate import _global_aggregate_explanations, client_selection_class
18 + from experiments.local_training import local_train
19 + from experiments.utils import average_weights, average_weights_class, max_weights, max_weights_class, weighted_weights
20 + from experiments.global_logic_aggregate import _global_aggregate_explanations, client_selection_class

```

Hình 3.2: Ví dụ về việc sửa đường dẫn import các module trong mã nguồn Python trong một tệp

3. Thêm tệp `__init__.py` để giải quyết các vấn đề liên quan đến lỗi `ModuleNotFoundError` khi import module trong dự án (Hình 3.3):

- Các tệp `__init__.py` đã được thêm vào thư mục `entropy_lens` và `models`.



Hình 3.3: Tạo tệp `__init__.py`

4. Đối với tập dữ liệu MNIST, rất may mắn, nhóm sinh viên đã tìm được một notebook hướng dẫn chạy mô hình ResNet50 trên Kaggle [6]. Trong cell thứ 5 của notebook, chỉ cần thay đổi mô hình từ ResNet50 thành ResNet10 là có thể thu được đầu ra cần thiết để sử dụng trong các thực nghiệm tái tạo.
5. Đối với tập dữ liệu CUB và mô hình FedAvg Logic, nhóm đành phải tạm thời bỏ qua việc triển khai thí nghiệm này. Do kích thước dữ liệu CUB quá lớn, việc triển khai mô hình ResNet10 mất nhiều thời gian hơn dự kiến. Mô hình FedAvg thì khiến nhóm gặp quá nhiều khó khăn trong việc thiết lập và chạy mô hình. Đây là một điều rất đáng tiếc, vì việc thực hiện các thí nghiệm trên các dữ liệu và mô hình này có thể mang lại những kết quả giá trị và góp phần nâng cao chất lượng nghiên cứu.
6. Đối với lỗi liên quan đến hàm `torch.stack()` thì mất nhiều thời gian hơn cả. Nhóm đã thực hiện chuẩn hóa kích thước của các tensor đầu vào để đảm bảo rằng tất cả các tensor có cùng kích thước trong tệp `entropy_lens/logic/nn/entropy.py` (3.1.3.2):

```

1 import torch.nn.functional as F
2
3 def forward(self, input: Tensor) -> Tensor:
4     if not isinstance(input, torch.Tensor):
5         # Find the maximum size in each dimension
6         max_size = [max(tensor.shape[i] for tensor in input) for i in
7                     ↪ range(len(input[0].shape))]
8
9         # Pad all tensors to the same size
10        padded_tensors = []
11        for tensor in input:

```

```

11         # Calculate the padding for each dimension
12         pad = []
13         for i in range(len(tensor.shape) - 1, -1, -1):
14             pad.extend([0, max_size[i] - tensor.shape[i]])
15         padded_tensor = F.pad(tensor, pad, "constant", 0) # Pad with
16             ↪ zeros
17         padded_tensors.append(padded_tensor)
18
19     # Stack the padded tensors into a single tensor
20     input = torch.stack(padded_tensors)

```

Những thay đổi trên đảm bảo rằng mã nguồn có thể được sử dụng một cách ổn định trong các môi trường khác nhau.

3.1.3.3 Khởi chạy mã nguồn tái tạo thực nghiệm

Sau khi debug, chạy lại từng tệp quan trọng bằng lệnh `python3 -m` để đảm bảo các đường dẫn import được xử lý đúng trong cấu trúc thư mục dự án (Hình 3.4).

```

[ ] !pip install -r requirements.txt

[ ] !pip install pytorch-lightning==1.5.0

[ ] !python3 -m entropy_lens.logic.utils
    !python3 -m entropy_lens.logic.metrics
    !python3 -m entropy_lens.logic.nn.utils
    !python3 -m entropy_lens.nn.concepts
    !python3 -m entropy_lens.nn.logic
    !python3 -m entropy_lens.logic.nn.entropy
    !python3 -m entropy_lens.nn.functional.loss
    !python3 -m entropy_lens.models.explainer

[ ] !python3 -m entropy_lens.__init__
    !python3 -m entropy_lens.models.__init__

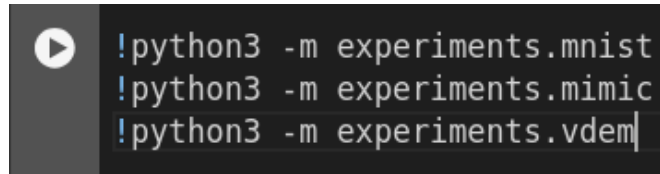
[ ] !python3 -m experiments.local_training
    !python3 -m experiments.global_logic_aggregate

```

Hình 3.4: Cài đặt và chạy các module chính của dự án.

Mô hình LR-XFL: Chạy mô hình LR-XFL bằng các dòng lệnh sau, lần lượt cho ba bộ dữ liệu MNIST, MIMIC-II, và VDEM (Hình 3.5):

Mô hình Học tập trung: Đối với mô hình Học tập trung, điều chỉnh các tệp `experiments/mnist`, `experiments/mimic`, và `experiments/vdem` bằng cách đặt



```
!python3 -m experiments.mnist
!python3 -m experiments.mimic
!python3 -m experiments.vdem
```

Hình 3.5: Lệnh chạy mô hình LR-XFL trên ba bộ dữ liệu: MNIST, MIMIC-II, và VDEM.

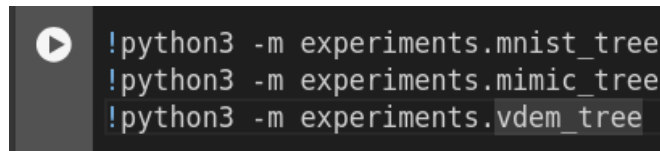
`num_users = 1` để chỉ tập trung vào một người dùng duy nhất (Hình 3.6). Sau đó, lặp lại các câu lệnh giống như lệnh chạy .



```
num_users = 1
```

Hình 3.6: Sửa mã nguồn để chạy mô hình Học tập trung.

Mô hình DDT: Chạy mô hình DDT bằng các dòng lệnh sau, lần lượt cho ba bộ dữ liệu MNIST, MIMIC-II, và VDEM (Hình 3.5):



```
!python3 -m experiments.mnist_tree
!python3 -m experiments.mimic_tree
!python3 -m experiments.vdem_tree
```

Hình 3.7: Sửa mã nguồn để chạy mô hình DDT.

3.1.3.4 Kết quả tái tạo thực nghiệm

MNIST (Chẵn/Lẻ)

- Kết quả tái tạo thực nghiệm trên tập dữ liệu MNIST (Chẵn/Lẻ) được minh họa ở bảng 3.1:

Bảng 3.1: So sánh hiệu quả trên tập dữ liệu MNIST (Chẵn/Lẻ)

MNIST (Chẵn/Lẻ)	Học tập trung	DDT	LR-XFL
Độ chính xác mô hình	98.64%	99.78%	99.98%
Độ chính xác quy tắc	99.34%	99.71%	99.75%
Độ tin cậy quy tắc	99.95%	-	99.97%

- Quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc để phân biệt giữa các số chẵn và số lẻ trong tập dữ liệu MNIST (Chẵn/Lẻ) được thể hiện trong bảng 3.2.

Bảng 3.2: Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu MNIST (Chẵn/Lẻ), cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Odd $\leftrightarrow \neg 0 \wedge \neg 2 \wedge \neg 4 \wedge \neg 6 \wedge \neg 8$	99.95%	99.97%
Even $\leftrightarrow \neg 1 \wedge \neg 3 \wedge \neg 5 \wedge \neg 7 \wedge \neg 9$	99.97%	99.98%

VDEM

- Kết quả tái tạo thực nghiệm trên tập dữ liệu VDEM được minh họa ở bảng 3.3:

Bảng 3.3: So sánh hiệu quả trên tập dữ liệu VDEM

VDEM	Học tập trung	DDT	LR-XFL
Độ chính xác mô hình	92.12%	92.55%	93.08%
Độ chính xác quy tắc	91.04%	92.52%	93.08%
Độ tin cậy quy tắc	93.59%	-	99.98%

- Quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc để phân biệt giữa nền dân chủ bầu cử và dân chủ không bầu cử trong tập dữ liệu VDEM được thể hiện trong bảng 3.4.

Bảng 3.4: Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu VDEM, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Electoral_democracy $\leftrightarrow v2x_freexp_altinf \wedge v2xel_frefair \wedge v2x_cspart$	93.06%	96.67%
Non_electoral_democracy $\leftrightarrow \neg v2xel_frefair \wedge \neg v2x_mpi$	89.98%	97.33%

MIMIC-II

- Kết quả tái tạo thực nghiệm trên tập dữ liệu MIMIC-II được minh họa ở bảng 3.5:

Bảng 3.5: So sánh hiệu quả trên tập dữ liệu MIMIC-II

MIMIC-II	Học tập trung	DDT	LR-XFL
Độ chính xác mô hình	73.46%	76.40%	80.24%
Độ chính xác quy tắc	66.56%	67.80%	64.15%
Độ tin cậy quy tắc	77.24%	-	77.21%

- Quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc để phân biệt khả năng phục hồi của bệnh nhân trong tập dữ liệu MIMIC-II được thể hiện trong bảng 3.6.

Bảng 3.6: Các quy tắc toàn cục cho 2 lớp trong tập dữ liệu MIMIC-II, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
$\text{Recover} \leftrightarrow \neg \text{stroke_flg} \wedge \neg \text{copd_flg} \wedge \neg \text{age_HIGH}$	72.21%	100.00%
$\text{Non_recover} \leftrightarrow \text{copd_flg} \wedge \text{age_HIGH} \wedge \text{stroke_flg} \wedge \neg \text{age_LOW}$	69.93%	100.00%

Chương 4

Triển khai thí nghiệm riêng của sinh viên

4.1 Thí nghiệm cải tiến

4.1.1 Tập dữ liệu bổ sung

Nhằm hiểu rõ hơn về các thực nghiệm được trình bày, nhóm sinh viên đã bổ sung thêm một tập dữ liệu mới, Default of Credit Dataset, để phân tích sâu hơn và thực hiện cải tiến.

Default of Credit Dataset: Liệu tháng sau khách hàng này có thanh toán khoản vay không?

Default of Credit Dataset [4] là một tập dữ liệu công khai phổ biến, được thiết kế để phân tích khả năng vỡ nợ tín dụng của khách hàng. Tập dữ liệu này bao gồm thông tin về các khoản thanh toán chậm, nhân khẩu học, dữ liệu tín dụng, lịch sử thanh toán và hóa đơn của khách hàng sử dụng thẻ tín dụng tại Đài Loan trong khoảng thời gian từ tháng 4 đến tháng 9 năm 2005. Thông tin của mỗi khách hàng được biểu diễn qua 25 đặc trưng:

- Thông tin nhân khẩu học (e.g., giới tính, độ tuổi, tình trạng hôn nhân, trình độ học vấn).
- Lịch sử tín dụng (e.g., trạng thái thanh toán trong 6 tháng gần nhất).
- Số dư tín dụng (e.g., giới hạn tín dụng, số dư hiện tại).
- Mẫu thanh toán (e.g., tổng số tiền thanh toán hàng tháng).

Mục tiêu của tập dữ liệu là dự đoán khả năng một khách hàng sẽ không thể thanh toán khoản vay trong tháng tiếp theo, dựa trên các đặc trưng đã cho. Trong thí nghiệm của sinh viên, dữ liệu được tiền xử lý để xử lý giá trị thiếu và chuẩn hóa các đặc trưng số. Các đặc trưng dạng danh mục (e.g., tình trạng hôn nhân, giới tính) được mã hóa dạng one-hot.

Không gian đầu vào X bao gồm 23 đặc trưng mô tả khách hàng, trong khi không gian đầu ra Y là một biến nhị phân đại diện cho trạng thái vỡ nợ (0: không vỡ nợ, 1: vỡ nợ). Nhiệm vụ chính là xây dựng một mô hình để ánh xạ từ $X \rightarrow Y$, đồng thời cung cấp giải thích về các yếu tố ảnh hưởng chính đến quyết định dự đoán.

4.1.2 Thí nghiệm cải tiến

4.1.2.1 Đặt vấn đề

Hiện tại, các công thức xác định kết nối logic tự động, bao gồm tính chéo và tính riêng biệt, đều giả định rằng tất cả các đặc trưng trong dữ liệu có mức độ quan trọng như nhau.

- **Tính chéo:** Công thức này đánh giá mức độ độc lập của từng đặc trưng, tức là khả năng xuất hiện của một đặc trưng mà không bị phụ thuộc vào các đặc trưng khác trong quy tắc:

$$D = \frac{\sum_{i=1}^n p_{ii}}{\sum_{i=1}^n \sum_{j=1}^n p_{ij}}$$

Trong đó:

- p_{ii} : Số lần đặc trưng f_i xuất hiện độc lập.
- p_{ij} : Số lần đặc trưng f_i và f_j cùng xuất hiện.
- n : Tổng số đặc trưng trong dữ liệu.
- **Tính riêng biệt:** Công thức này tập trung vào việc đánh giá mức độ một đặc trưng loại trừ sự hiện diện của đặc trưng khác trong các quy tắc:

$$E = \frac{\max_{i=1}^n \left(\sum_{j=1}^n q_{ij} \right)}{\sum_{m=1}^M l_m}$$

Trong đó:

- q_{ij} : Số lần đặc trưng i xuất hiện mà đặc trưng j không xuất hiện.
- l_m : Độ dài của quy tắc thứ m trong tập M .
- n : Tổng số đặc trưng.
- M : Tổng số quy tắc.

Tuy nhiên, giả định này không phản ánh đúng bản chất của nhiều dữ liệu thực tế. Trong nhiều trường hợp, một số đặc trưng có thể có vai trò chi phối kết quả dự báo, đặc biệt khi giá trị của chúng đạt mức cực cao hoặc cực thấp.

- Một sinh viên có thể học rất giỏi, thông minh và chăm chỉ, nhưng nếu nghỉ quá số buổi quy định thì chắc chắn bị đánh trượt môn.
- Một khách hàng có điểm tín dụng tốt và lịch sử thanh toán ổn định, nhưng nếu số dư nợ vượt ngưỡng an toàn thì vẫn có thể bị từ chối cấp thêm tín dụng.

Những tình huống này cho thấy rằng một đặc trưng đơn lẻ, khi vượt ngưỡng quan trọng, có thể chi phối hoàn toàn kết quả, bất kể các đặc trưng khác có tác động tích cực như thế nào.

Do đó, việc giả định rằng tất cả các đặc trưng đều có mức độ quan trọng như nhau không chỉ đơn giản hóa vấn đề mà còn tiềm ẩn nguy cơ dẫn đến những dự đoán sai lệch. Thực tế này đặt ra nhu cầu cần thiết phải phân tích sâu hơn các trường hợp mà một hoặc một vài đặc trưng có tác động chi phối mạnh mẽ, đặc biệt trong các kịch bản dự báo thực tế. Nghĩa là, trong những trường hợp có đặc trưng lấn át như trên, muốn phản ánh đúng hơn tác động chi phối của các đặc trưng đó, cần tìm cách tăng xác suất để bộ kết nối tự động chọn phép OR.

4.1.2.2 Chiến lược gốc của tác giả

Trong thí nghiệm gốc, tác giả đặt ngưỡng rất cao để quyết định lựa chọn giữa phép OR hoặc AND trong việc kết nối các đặc trưng. Cụ thể:

- Nếu tính chéo (D) lớn hơn 0.9 hoặc tính riêng biệt (E) lớn hơn 0.8, thì phép OR được áp dụng.
- Ngược lại, nếu cả hai giá trị đều không đạt ngưỡng trên, phép AND được mặc định sử dụng.

Quy định ngưỡng cao này cho thấy rằng chỉ khi các đặc trưng có mức độ **độc lập** hoặc **tách biệt** rất lớn, chúng mới đủ điều kiện để áp dụng phép OR, nhằm phản ánh sự tự do và độc lập thực sự giữa các đặc trưng trong dữ liệu. Trong các trường hợp còn lại, phép AND được sử dụng để đảm bảo rằng mối quan hệ giữa các đặc trưng là đủ chặt chẽ và phụ thuộc lẫn nhau để đưa ra kết quả dự báo chính xác hơn.

4.1.2.3 Lựa chọn dữ liệu cải tiến

Với các tập dữ liệu mà tác giả gốc sử dụng, nhóm gặp một số khó khăn khi triển khai thí nghiệm cải tiến. Chẳng hạn, với tập dữ liệu như MNIST và VDEM, kết quả thực nghiệm cho thấy độ chính xác mô hình, độ chính xác quy tắc, và độ tin cậy quy tắc đã đạt mức rất cao. Trong những trường hợp như vậy, dù có thực hiện cải tiến thì sự thay đổi về kết quả cũng khó có thể được nhận biết một cách rõ ràng do các chỉ số đã tiến gần đến giới hạn tối ưu.

Còn với tập dữ liệu MIMIC-II, một điều đáng chú ý là, trong tất cả các trường hợp, bộ kết nối logic đều chọn toán tử AND. Kết quả này phản ánh đúng đặc tính của dữ liệu y tế:

chỉ dựa vào một chỉ số bình thường, không thể kết luận chính xác tình trạng sức khỏe của bệnh nhân. Thay vào đó, sự kết hợp giữa nhiều chỉ số bình thường là yếu tố thể hiện rõ nhất trạng thái khỏe mạnh của bệnh nhân. Việc lựa chọn toán tử AND cho thấy sự hợp lý, đồng thời minh chứng khả năng giải thích chính xác của mô hình LR-XFL. Song, mục đích của nhóm là cải tiến mô hình để ưu tiên chọn toán tử OR khi xuất hiện các đặc trưng mang tính chất lấn áp, mà cả ba tập dữ liệu có sẵn của tác giả đều không phù hợp để thực hiện thí nghiệm theo hướng này.

Tuy nhiên, một điểm đặc biệt thú vị khi thực nghiệm trên mô hình LR-XFL là sinh viên có thể tùy ý thay đổi bộ dữ liệu để kiểm tra hiệu năng của mô hình cũng như tiến hành các thí nghiệm cải tiến. Điều này mang lại nhiều cơ hội để đánh giá mô hình trên các ngữ cảnh và loại dữ liệu khác nhau. Đó là lý do, sau khi tìm hiểu thêm về các tập dữ liệu phù hợp với tiêu chí đề ra, nhóm đã quyết định bổ sung tập dữ liệu Default of Credit Card Clients Dataset để triển khai thực nghiệm.

Tập dữ liệu này phản ánh rõ các tình huống mà một số đặc trưng có tác động chi phối lớn đến kết quả dự báo, đáp ứng tiêu chí tồn tại nhiều đặc trưng có giá trị ổn định ở mức trung bình, nhưng một đặc trưng khác quan trọng hơn lại đạt giá trị quá thấp hoặc quá cao đến mức cảnh báo, dẫn đến việc khách hàng bị xếp vào nhóm rủi ro. Có thể liên hệ đến các trường hợp thực tế như:

- Khách hàng có hạn mức tín dụng cao nhưng thường xuyên chậm trả, dẫn đến bị đánh giá rủi ro cao.
- Khách hàng có dư nợ thấp nhưng thường xuyên vượt hạn mức tín dụng, làm tăng nguy cơ bị từ chối tín dụng.

4.1.2.4 Tiền xử lý dữ liệu

Phần này sẽ trình bày rõ hơn về cách nhóm tiền xử lý dữ liệu trước khi áp dụng vào mô hình LR-XFL. Qua quan sát thực tế, nhóm nhận thấy rằng các đặc trưng có khả năng lấn áp thường nằm ở các trường PAY_i, PAYATM_i, và BILLATM_i. Ký hiệu i biểu thị mốc thời gian tương ứng với tháng thứ i. Từ nhận xét trên ta sử dụng one hot mã hóa những đặc trưng này thành 3 mức: cực thấp, trung bình, và cực cao.

Lưu ý: Nhóm sinh viên muốn nhấn mạnh rằng việc phân mức và mã hóa các đặc trưng được thực hiện dựa trên hiểu biết cá nhân, không hoàn toàn dựa trên nguồn tin chính thống hoặc quy định cụ thể từ các tổ chức tài chính. Các mức và tiêu chí được đề xuất chỉ nhằm mục đích phục vụ thực nghiệm và phân tích trong phạm vi nghiên cứu này.

Chi tiết như sau:

- PAY_i: Ghi nhận trạng thái thanh toán qua các tháng, bao gồm:
 - PAY_0: Trạng thái thanh toán trong tháng hiện tại.
 - PAY_2 đến PAY_6: Trạng thái thanh toán trong các tháng trước đó, từ 2 tháng trước đến 6 tháng trước.

- Quy ước:
 - -1: Thanh toán đúng hạn.
 - 1: Chậm thanh toán một tháng.
 - 2: Chậm thanh toán hai tháng.
 - ...
 - 8: Chậm thanh toán tám tháng.
 - 9: Chậm thanh toán chín tháng hoặc hơn.
- Phân loại đặc trưng PAY_i thành ba mức như sau (4.1.2.4):
 - **Tốt:** PAY_i = -1
 - **Trung bình:** $1 \leq \text{PAY}_i \leq 5$
 - **Đáng báo động:** PAY_i ≥ 6

```

1 def encode_payment(value):
2     if value == -1: # Paid on time
3         return [1, 0, 0] # GOOD
4     elif 1 <= value <= 5: # Payment delayed by 1-5 months
5         return [0, 1, 0] # AVERAGE
6     elif value >= 6: # Payment delayed by 6 months or more
7         return [0, 0, 1] # ALERT
8     else:
9         return [0, 0, 0] # Invalid value

```

- BILL_AMT_i: Ghi nhận số tiền trên hóa đơn trong các tháng (đơn vị: Tân Đài tệ), bao gồm:
 - BILL_AMT1 đến BILL_AMT6: Số tiền trên hóa đơn từ 1 tháng trước đến 6 tháng trước.
 - Phân loại đặc trưng BILL_AMT_i thành ba mức như sau (4.1.2.4):
 - **Tốt:** BILL_AMT_i < 0.5 × LIMIT_BAL (dưới 50% hạn mức tín dụng).
 - **Trung bình:** $0.5 \times \text{LIMIT_BAL} \leq \text{BILL_AMT}_i < 0.8 \times \text{LIMIT_BAL}$ (từ 50% đến 80% hạn mức tín dụng).
 - **Đáng báo động:** BILL_AMT_i ≥ 0.8 × LIMIT_BAL (trên 80% hạn mức tín dụng).

```

1 def encode_bill_amt(bill_amt, limit_bal):
2     if bill_amt < 0.5 * limit_bal:
3         return [1, 0, 0] # GOOD
4     elif 0.5 * limit_bal <= bill_amt < 0.8 * limit_bal:
5         return [0, 1, 0] # AVERAGE
6     else:
7         return [0, 0, 1] # ALERT

```

- **PAY_AMTi**: Ghi nhận số tiền được thanh toán qua các tháng (đơn vị: Tân Đài tệ), bao gồm:
 - **PAY_AMT1** đến **PAY_AMT6**: Số tiền được thanh toán từ 1 tháng trước đến 6 tháng trước.
 - Phân loại đặc trưng **PAY_AMTi** thành ba mức như sau (4.1.2.4):
 - **Tốt**: $\text{PAY_AMTi} \geq 0.5 \times \text{BILL_AMTi}$
(số tiền thanh toán lớn hơn hoặc bằng 50% số tiền hóa đơn).
 - **Trung bình**: $0.1 \times \text{BILL_AMTi} \leq \text{PAY_AMTi} < 0.5 \times \text{BILL_AMTi}$
(số tiền thanh toán nằm trong khoảng từ 10% đến 50% số tiền hóa đơn).
 - **Đáng báo động**: $\text{PAY_AMTi} < 0.1 \times \text{BILL_AMTi}$ hoặc $\text{PAY_AMTi} = 0$
(số tiền thanh toán nhỏ hơn 10% số tiền hóa đơn hoặc không thanh toán).

```

1 def encode_pay_amt(pay_amt, bill_amt):
2     if pay_amt >= 0.5 * bill_amt:
3         return [1, 0, 0] # Safe
4     elif 0.1 * bill_amt <= pay_amt < 0.5 * bill_amt:
5         return [0, 1, 0] # AVERAGE
6     else:
7         return [0, 0, 1] # Alert

```

Đặc trưng về học vấn (**Education**) và tình trạng hôn nhân (**Marriage**) được mã hóa như theo hướng dẫn của tác giả tập dữ liệu (4.1.2.4):

- **Education** được mã hóa theo các giá trị sau:
 - **1**: Graduate School (Trình độ sau đại học)
 - **2**: University (Trình độ đại học)
 - **3**: High School (Trình độ trung học phổ thông)
 - **4**: Others (Các trình độ khác)
 - **5 hoặc 6**: Unknown (Không rõ hoặc không được cung cấp thông tin)
- **Marriage** được mã hóa theo các giá trị sau:
 - **1**: Married (Đã kết hôn)
 - **2**: Single (Độc thân)
 - **3**: Others (Các tình trạng khác như ly dị, góa, không rõ)

```

1 def encode_education(education):
2     if education == 1:
3         return [1, 0, 0, 0, 0] # GRADUATE SCHOOL
4     elif education == 2:
5         return [0, 1, 0, 0, 0] # UNIVERSITY
6     elif education == 3:
7         return [0, 0, 1, 0, 0] # HIGH SCHOOL
8     elif education == 4:
9         return [0, 0, 0, 1, 0] # OTHERS
10    else:
11        return [0, 0, 0, 0, 1] # UNKNOWN
12
13 def encode_marriage(marriage):
14     if marriage == 1:
15         return [1, 0, 0] # MARRIED
16     elif marriage == 2:
17         return [0, 1, 0] # SINGLE
18     else:
19         return [0, 0, 1] # OTHERS

```

Các đặc trưng khác liên quan đến nhân khẩu học như tuổi tác (**Age**) được mã hóa one hot bằng cách sử dụng hàm `KBinsDiscretizer` từ thư viện **Scikit-learn**, chia các giá trị thành 3 khoảng đều nhau và mã hóa theo phương pháp one-hot dense (4.1.2.4).

```

1 est = KBinsDiscretizer(n_bins=3, encode='onehot-dense',
2                        strategy='uniform')

```

Ngoài ra, nhóm cũng triển khai các bước xử lý dữ liệu bị thiếu và chuẩn hóa:

- **Xử lý dữ liệu bị thiếu:** Sử dụng `SimpleImputer` với chiến lược `mean` để thay thế các giá trị thiếu (`np.nan`) bằng giá trị trung bình của cột.
- **Chuẩn hóa:** Áp dụng `MinMaxScaler` để chuẩn hóa dữ liệu về khoảng giá trị $[0, 1]$.

4.1.3 Đề xuất cải tiến

Nhóm đề xuất cải tiến công thức tính chéo để đánh giá mức độ độc lập của từng đặc trưng trong quy tắc như sau:

$$D = \frac{\sum_{i=1}^n w_i^2 \cdot p_{ii}}{\sum_{i=1}^n \sum_{j=1}^n w_i \cdot w_j \cdot p_{ij}}$$

Trong đó:

- w_i điều chỉnh mức độ ảnh hưởng của từng đặc trưng trong công thức.
- Các trọng số w_i có thể được điều chỉnh để phù hợp với từng bài toán cụ thể, ví dụ như dựa vào mô hình học máy hoặc kiến thức chuyên ngành.
- Sinh viên nhận thấy rằng các đặc trưng có tính chất lẩn áp và mang tính quyết định trong kết quả dự báo rất giống với các đặc trưng có giá trị Gini cao trong các mô hình cây quyết định Decision Tree. Những đặc trưng này không chỉ đóng vai trò quan trọng trong việc phân chia dữ liệu mà còn thường thể hiện mức độ ảnh hưởng mạnh mẽ đến dự đoán cuối cùng. Do đó, trong phạm vi nghiên cứu của môn học, nhóm quyết định sử dụng chỉ số Gini để định lượng và điều chỉnh trọng số w_i trong công thức cải tiến tính chéo.
- Các trọng số w_i được chuẩn hóa để đảm bảo rằng chúng thể hiện mức độ quan trọng tương đối của từng đặc trưng trong mô hình. Việc chuẩn hóa giúp các trọng số trở nên dễ so sánh, giữ cho tổng của chúng bằng 1, từ đó tăng tính ổn định và khả năng giải thích của mô hình. Đồng thời, điều này đảm bảo rằng không có trọng số nào chiếm ưu thế quá mức, duy trì sự cân bằng và phù hợp với các phương pháp tính toán dựa trên xác suất hoặc tỷ lệ.

Như vậy, trước khi đưa vào mô hình LR-XFL, ta cần chạy mô hình **Random Forest** trên các đặc trưng để đánh giá trọng số của từng đặc trưng. Để thực hiện điều này, ta sẽ sửa mã nguồn trong hàm `def explain_class` thuộc tệp `entropy_lens/logic/nn/entropy.py` như sau (4.1.3):

```

1  # Extract correct data points based on the target class using a helper
   ↪ function
2  x_correct, y_correct1h = _get_correct_data(x, y1h, model, target_class)
3
4  # If no correct data points are found, return None for all outputs
5  if x_correct is None:
6      return None, None, None
7
8  # Initialize a concept extraction tool with the chosen activation
   ↪ function
9  activation = 'identity_bool'
10 conceptizator = Conceptizator(activation)
11
12 # Apply the conceptizator to extract concepts (binary activations) for
   ↪ the target class
13 y_correct = conceptizator(y_correct1h[:, target_class]) # On correct
   ↪ samples
14
15 # Convert the correct data and labels from tensors to NumPy arrays for
   ↪ compatibility with sklearn
16 x_correct_np = x_correct.numpy()
17 y_correct_np = y_correct.numpy()
18
19 # Initialize a Random Forest Classifier model
20 rf_model = RandomForestClassifier()
21
22 # Train the Random Forest model using the correct data points and their
   ↪ corresponding labels
23 rf_model.fit(x_correct_np, y_correct_np)
24
25 # Extract feature importance scores from the trained Random Forest
   ↪ model
26 feature_weights = torch.tensor(rf_model.feature_importances_)

```

Sau khi hoàn tất việc tính toán các ma trận đồng hiện, cũng trong hàm này, ta tiến hành nhân các phần tử của ma trận với các trọng số như sau (4.1.3):

```
1 # Scale the 'positive' co-occurrence matrix by the feature importance
  ↳ weights
2 class_concept_co_occ['positive'] = feature_weights.unsqueeze(0) *
  ↳ class_concept_co_occ['positive']
```

Cần nhấn mạnh rằng, nhóm chỉ thực hiện cải tiến trên công thức tính chéo, không thực hiện cải tiến tính riêng biệt. Lý do là mục tiêu nghiên cứu hiện tại tập trung vào việc xác định các đặc trưng có khả năng chi phối mạnh mẽ kết quả dự đoán. Cụ thể, khi giá trị của một đặc trưng đạt mức cực cao hoặc cực thấp, đặc trưng này có thể tự mình quyết định kết quả mà không cần sự hỗ trợ hay kết hợp từ các đặc trưng khác. Ngược lại, công thức tính riêng biệt đánh giá mối quan hệ loại trừ giữa các đặc trưng, tức là khi một đặc trưng xuất hiện thì các đặc trưng còn lại phải không xuất hiện. Điều này không phù hợp với mục tiêu của sinh viên, vốn tập trung vào việc tìm kiếm các đặc trưng độc lập có khả năng quyết định mạnh, bất kể sự xuất hiện của các đặc trưng khác.

4.1.4 Kết quả

Bảng 4.1 so sánh hiệu năng của mô hình trước và sau khi áp dụng cải tiến công thức tính chéo, với các chỉ số cao hơn được in đậm để dễ dàng nhận diện.

Kết quả cho thấy:

- Độ chính xác của mô hình giảm nhẹ sau cải tiến (80.36% xuống 79.98%).
- Độ chính xác quy tắc tăng từ 65.61% lên 66.54%.
- Độ tin cậy quy tắc cũng tăng từ 92.27% lên 93.23%.

	Chưa cải tiến	Sau cải tiến
Độ chính xác mô hình	80.36%	79.98%
Độ chính xác quy tắc	65.61%	66.54%
Độ tin cậy quy tắc	92.27%	93.23%

Bảng 4.1: So sánh hiệu quả mô hình trước và sau khi cải tiến công thức tính chéo.

Kết quả cho thấy các chỉ số hiệu quả sau khi cải tiến công thức tính chéo có sự cải thiện, nhưng mức độ cải thiện chưa thực sự đáng kể. Sau khi xem xét lại quá trình thí nghiệm, sinh viên rút ra một vài nhận xét như sau:

- Mặc dù chỉ số Gini đơn giản và dễ áp dụng, nhưng nó tồn tại một số hạn chế:
 - Trong trường hợp dữ liệu không cân bằng, chỉ số Gini có thể đánh giá thấp mức độ quan trọng của các đặc trưng hiếm.
 - Gini chỉ cung cấp một đánh giá toàn cục về đặc trưng, không thể giải thích cụ thể từng dự đoán của mô hình.
 - Gini có xu hướng ưu tiên các đặc trưng với số lượng giá trị khác biệt lớn, dẫn đến việc đánh giá không công bằng.
- Sinh viên đã nghĩ đến việc sử dụng **SHAP (SHapley Additive exPlanations)** thay cho chỉ số Gini để gán trọng số cho các đặc trưng, bởi SHAP mang lại nhiều ưu điểm vượt trội:
 - SHAP cung cấp một cách tiếp cận lý thuyết vững chắc dựa trên nền tảng toán học của lý thuyết giá trị Shapley, đảm bảo mỗi đặc trưng được gán trọng số dựa trên mức độ đóng góp thực sự của nó vào dự báo.
 - SHAP không chỉ giải thích mức độ quan trọng của đặc trưng toàn cục mà còn cung cấp khả năng giải thích từng dự đoán riêng lẻ, giúp hiểu rõ hơn cách mô hình hoạt động.
 - Chỉ số Gini có thể gặp vấn đề với bias trong dữ liệu không cân bằng, trong khi SHAP không bị ảnh hưởng bởi vấn đề này.
- Tuy nhiên, do giới hạn về thời gian và kiến thức hiện tại của nhóm sinh viên, SHAP vẫn là một khái niệm quá mới mẻ. Vì vậy, nhóm đã không thành công trong việc triển khai SHAP trong thí nghiệm cải tiến này. Chi tiết các lỗi phát sinh và nguyên nhân gây ra lỗi sẽ được nhóm trình bày kỹ lưỡng hơn trong báo cáo gửi đến bộ môn **Khoa học Dữ liệu**.
- Bên cạnh đó, cần tiến hành thêm các phân tích thống kê trên dữ liệu:
 - Đo lường tương quan để phân tích mối quan hệ giữa các đặc trưng nhằm xác định các đặc trưng có tương quan mạnh, từ đó giảm thiểu đặc trưng dư thừa hoặc tìm các mối quan hệ quan trọng.
 - Kiểm tra phân phối của từng đặc trưng để phát hiện ngoại lai có thể ảnh hưởng đến kết quả mô hình.
 - Đánh giá mức độ và cách xử lý giá trị thiếu để tránh bias trong dữ liệu.
 - Sử dụng các chỉ số như trung bình, độ lệch chuẩn để hiểu rõ đặc tính của từng đặc trưng.

4.2 Thí nghiệm mở rộng

Trong phần này, nhóm sẽ trình bày một phát hiện thú vị mà tác giả không đề cập trong bài báo gốc. Phát hiện này được sinh viên khám phá khi phân tích sâu mã nguồn. Nhóm sinh viên đặt tên cho phát hiện này là **Chiến lược tương tác với máy khách**.

Như đã đề cập trong bài báo, không phải giải thích nào từ máy khách cũng được máy chủ lựa chọn để đưa vào giải thích toàn cục. Thông qua phân tích mã nguồn, tiêu chuẩn để một người dùng **chắc chắn** được chọn có thể được chia thành ba điều kiện chính như sau:

- Máy khách có giải thích hợp lệ cho lớp mục tiêu.
- Nếu giải thích đã có từ trước, độ chính xác của họ phải cao hơn hoặc bằng với người dùng khác có cùng giải thích. Nếu cao hơn, những người có độ chính xác thấp hơn sẽ bị loại.
- Nếu đưa ra giải thích mới, máy khách sẽ được giữ lại và giải thích sẽ được thêm vào.

Đến đây, tác giả chia ra hai hướng cho **Chiến lược tương tác với máy khách**. Hình 4.1 chỉ ra đoạn mã nơi tác giả triển khai hai chiến lược này. Hai chiến lược có thể được mô tả bằng lời như sau:

- Khi `user_engagement_scale = 'large'`: Tất cả các máy khách hỗ trợ giải thích sẽ được tính là tham gia. Nói cách khác, khi có nhiều máy khách đưa ra cùng một giải thích, máy chủ sẽ chấp nhận giải thích của tất cả các máy khách.
- Khi `user_engagement_scale = 'small'`: Chỉ những người dùng tạo ra giải thích có độ chính xác cao nhất mới được tính là tham gia. Nói cách khác, máy chủ sẽ chỉ chấp nhận giải thích có độ chính xác cao nhất từ nhóm các máy khách có chung giải thích.

```
for explanation in best_sequence:
    if user_engagement_scale == 'large':
        users_to_keep.update(list(local_explanations_support[explanation][2]))
    else:
        users_to_keep.update(user_id)
```

Hình 4.1: Đoạn mã triển khai chiến lược tương tác với máy khách.

Trong nghiên cứu của tác giả, mã nguồn sử dụng chiến lược `user_engagement_scale = 'large'`. Nhóm sinh viên sau đó đã thử nghiệm thay đổi chiến lược thành `user_engagement_scale = 'small'`. Đúng như dự đoán, kết quả cho thấy chiến lược gốc của tác giả (`user_engagement_scale = 'large'`) đem lại hiệu quả vượt trội hơn hẳn. Kết quả so sánh trên tập dữ liệu VDEM được thể hiện ở bảng 4.2.

Đây là một phát hiện quan trọng, khẳng định hiệu quả của mô hình Học liên kết phân tán, đồng thời củng cố nền tảng của “Trí tuệ đám đông” mà nhóm sinh viên đã giới thiệu trong các phần đầu của bài phân tích. Nhắc lại, ta biết rằng phát hiện của Galton đã chứng minh: **Sự tổng hợp thông tin từ nhiều nguồn có thể vượt qua giới hạn của từng cá nhân**. Đây cũng là nguyên lý cốt lõi trong Học liên kết phân tán, nơi sự cộng tác và tổng hợp giữa các thiết bị hoặc tổ chức mang lại hiệu quả vượt trội. Dù một số cá nhân dự đoán sai rất nhiều, nhưng sự đa dạng trong dự đoán giúp trung bình của đám đông cực kỳ

	‘large’	‘small’
Độ chính xác mô hình	92.12%	79.98%
Độ chính xác quy tắc	91.04%	76.54%
Độ tin cậy quy tắc	93.59%	73.23%

Bảng 4.2: So sánh hiệu quả mô hình trên hai trường hợp `user_engagement_scale` trên bộ dữ liệu VDEM.

chính xác. Hiện tượng này có thể được biểu diễn bằng công thức toán học qua Định lý Đa dạng Dự đoán của James Surowiecki [3] như sau:

$$\text{Error of Crowd} = \frac{1}{n} \sum_{i=1}^n (s_i - \theta)^2 - \frac{1}{n} \sum_{i=1}^n (s_i - c)^2 = \text{Average Error} - \text{Diversity} \quad (4.1)$$

Trong đó:

- **Error of Crowd:** Sai số của dự đoán trung bình (dự đoán của đám đông).
- **Average Error:** Sai số trung bình của từng cá nhân so với giá trị thực.
- **Diversity:** Sự đa dạng trong dữ liệu.

Như vậy, chúng ta có thể giảm thiểu Error of Crowd thành hai cách:

- **Tập hợp một đám đông nhiều chuyên gia:** Average Error sẽ nhỏ, nhưng đồng thời, Diversity cũng sẽ rất nhỏ.
- **Tập hợp một đám đông nhiều người từ nhiều lĩnh vực khác nhau:** Average Error có thể sẽ rất to, nhưng nếu Diversity đủ lớn và cân bằng, ta cũng sẽ thu được sai số nhỏ hơn. Chúng ta sẽ cần một đám đông đủ lớn để làm cho xác suất có hiệu quả.

Chương 5

Nghiên cứu mở rộng

5.1 Giới thiệu về phương pháp

Như tác giả của bài báo LR-XFL đề cập, phương pháp tác giả đưa ra là phương pháp tiên phong và duy nhất cho đến thời điểm hiện tại giải quyết vấn đề tăng khả năng giải thích trong mô hình học liên kết phân tán. Chính vì vậy, nhóm sinh viên không thể tìm được các phương pháp cùng mục đích để đối chiếu kết quả. Thế nhưng trong quá trình đào sâu nghiên cứu về LR-XFL, nhóm sinh viên đã để ý tới cơ sở của là **Giải thích Logic dựa trên Entropy Mạng Nơ-ron** [2] mà tác giả đề cập và tìm đọc bài báo đó. Chính vì sự bất ngờ đối với nền tảng lý thuyết đồ sộ đằng sau phương pháp mà nhóm sinh viên đã quyết định đưa nó vào phần mở rộng của báo cáo này.

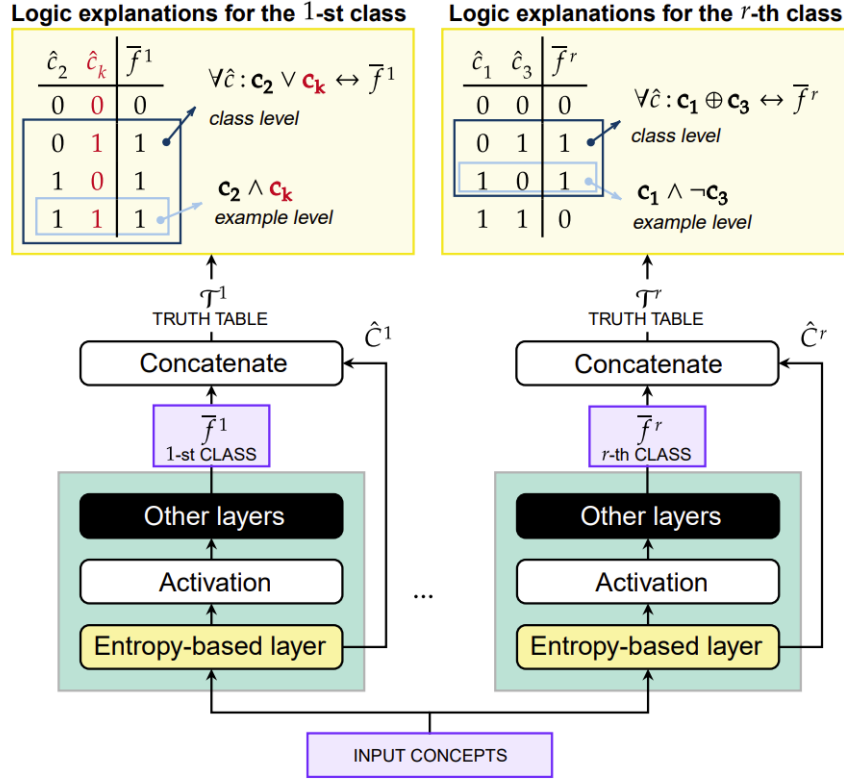
Nghiên cứu này là một đóng góp quan trọng trong lĩnh vực trí tuệ nhân tạo và học máy, đặc biệt là trong việc cung cấp giải thích cho các mô hình học sâu. Được đề xuất bởi một nhóm các nhà nghiên cứu hàng đầu, bài báo này đề cập đến việc sử dụng các khái niệm logic và entropy để giúp giải thích các quyết định của mạng nơ-ron, một trong những mô hình học máy phức tạp và khó hiểu nhất hiện nay.

Đóng góp chính của bài báo

Trong bài báo này, các tác giả đề xuất một lớp mười dựa trên entropy, nhằm xây dựng các mạng nơ-ron theo hướng giải thích dựa trên khái niệm, từ đó đưa ra các giải thích logic vị từ. Cách tiếp cận này không sử dụng một phương pháp post-hoc chỉ bổ sung giải thích sau quá trình huấn luyện, mà được thiết kế để tự giải thích ngay từ đầu. Phương pháp này tích hợp các ràng buộc trực tiếp vào cả thiết kế kiến trúc và quá trình học của mô hình, giúp mô hình tự động tạo ra những giải thích logic đơn giản.

Tiếp theo, các tác giả trình bày cách diễn giải các dự đoán của mô hình nơ-ron được đề xuất, để rút ra các giải thích logic cho từng quan sát cụ thể, cũng như cho toàn bộ nhóm mục tiêu. Phương pháp này được chứng minh là không chỉ cung cấp các giải thích chất lượng cao dựa trên sáu tiêu chí định lượng, mà còn đạt độ chính xác mô hình ngang bằng với các mô hình hộp đen và vượt trội so với các mô hình hộp trắng hiện đại nhất.

5.2 Kiến trúc mô hình



Hình 5.1: Với mỗi lớp i , mạng sử dụng một “đầu” của lớp tuyến tính dựa trên entropy (màu xanh lá) làm lớp đầu tiên, và nó đưa ra dự đoán về lớp f^i và bảng chân trị T^i (Phương trình 6 để đưa ra giải thích vị từ (màu vàng, trên)).

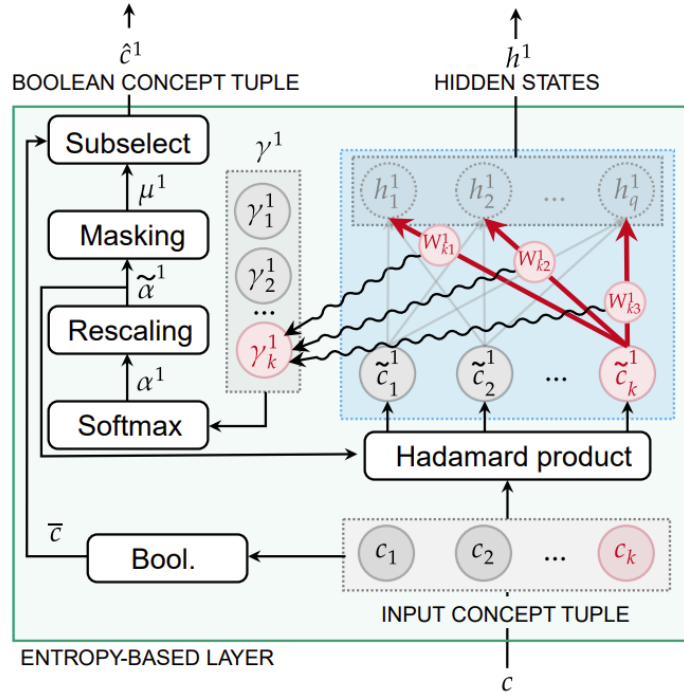
Giải thích Logic dựa trên Entropy của Mạng nơ-ron

Bài báo này giới thiệu một lớp tuyến tính mới, giúp mạng nơ-ron có khả năng giải thích logic dựa trên nguyên lý entropy (Hình 5.1 và Hình 5.2). Đầu vào của lớp này thuộc không gian khái niệm C , và kết quả của các tính toán trong lớp bao gồm:

- Các embedding h_i (tương tự các lớp tuyến tính khác)
- Một bảng chân trị T_i , giải thích cách mạng nơ-ron sử dụng các khái niệm để đưa ra dự đoán cho lớp mục tiêu thứ i .

Mỗi lớp mục tiêu trong bài toán yêu cầu một lớp riêng biệt dựa trên entropy, được ký hiệu bởi chỉ số i . Như vậy, nếu bài toán có nhiều lớp mục tiêu, sẽ cần nhiều lớp như vậy để xử lý tương ứng.

Để dễ hiểu và không làm mất tính tổng quát, từ đoạn này trở đi, ta sẽ tập trung phân tích một quan sát duy nhất, được biểu diễn dưới dạng một bộ khái niệm $c \in C$, và một



Hình 5.2: Một cái nhìn chi tiết về một “đầu” của lớp tuyến tính dựa trên entropy cho lớp đầu tiên, làm nổi bật vai trò của khái niệm đầu vào thứ k như ví dụ: (i) giá trị vô hướng γ^1_k (Phương trình 1) được tính từ tập hợp trọng số kết nối khái niệm đầu vào thứ k với các nơ-ron đầu ra của lớp tuyến tính dựa trên entropy; (ii) tầm quan trọng tương đối của mỗi khái niệm được tóm tắt qua phân phối phân loại α^1 (Phương trình 2); (iii) các điểm số mức độ liên quan được thay đổi lại $\tilde{\alpha}^1$ loại bỏ các khái niệm đầu vào không liên quan (Phương trình 3); (iv) trạng thái ẩn h_1 (Phương trình 4) và các khái niệm kiểu Boolean \hat{c}^1 (Phương trình 5) được cung cấp như các đầu ra của lớp tuyến tính dựa trên entropy.

mạng nơ-ron f_i dự đoán khả năng quan sát đó thuộc về lớp thứ i . Đối với bài toán phân loại đa lớp, mỗi lớp mục tiêu như “mèo”, “chó”, hay “chim”, có thể có các lớp ẩn được thiết kế riêng biệt để xử lý đặc thù. Như vậy sẽ giúp mô hình tối ưu hóa khả năng dự đoán cho từng lớp cụ thể. Tuy nhiên, để tiết kiệm tài nguyên và giảm độ phức tạp, các lớp ẩn này cũng có thể được chia sẻ giữa các mạng khác nhau.

Lớp tuyến tính dựa trên entropy

Nhóm tác giả chỉ ra rằng, theo nhiều nghiên cứu trong triết học và tâm lý học, con người thường có xu hướng ưu tiên những giả thuyết đơn giản nhất khi so sánh một tập hợp các giả thuyết dẫn đến cùng một kết quả. Vì vậy, phương pháp dựa trên entropy được đề xuất trong bài báo mô phỏng xu hướng này bằng cách tích hợp nó vào một mô hình học sâu có thể tối ưu hóa thông qua lan truyền ngược. Lớp tuyến tính dựa trên entropy được thiết kế để giúp mô hình chỉ chọn một số ít các khái niệm đầu vào quan trọng nhất, từ đó đưa ra các giải thích ngắn gọn và dễ hiểu cho dự đoán của mình. Các tham số có thể học được của

bao gồm ma trận trọng số W và vector bias b .

Trong phần tiếp theo, quá trình lan truyền xuôi sẽ được mô tả qua các phép tính được trình bày từ phương trình 1 đến phương trình 4. Còn quá trình tạo ra các bảng chân trị để trích xuất giải thích được trình bày qua phương trình 5 và phương trình 6.

Mức độ liên quan của từng khái niệm đầu vào có thể được tóm tắt một cách xấp xỉ bằng một thước đo phụ thuộc vào giá trị của các trọng số kết nối khái niệm đó với phần mạng phía trên. Đối với mạng f_i (tức là dự đoán lớp thứ i) và khái niệm đầu vào thứ j , nhóm tác giả ký hiệu W_j^i là vector trọng số xuất phát từ đầu vào thứ j (xem Hình 5.2)

$$\gamma_j^i = \|W_j^i\|_1. \quad (1)$$

Giá trị γ_j^i càng lớn thì mức độ liên quan của khái niệm j đối với mạng f_i càng cao. Trong trường hợp giới hạn ($\gamma_j^i \rightarrow 0$), mô hình f_i loại bỏ khái niệm thứ j . Để lựa chọn một số ít các khái niệm có liên quan cho mỗi lớp mục tiêu, các khái niệm được thiết lập để cạnh tranh với nhau. Với mục tiêu này, tầm quan trọng tương đối của mỗi khái niệm đối với lớp thứ i được tóm tắt trong phân phối phân loại α^i , bao gồm các hệ số $\alpha_j^i \in [0, 1]$ (với $\sum_j \alpha_j^i = 1$), được mô hình hóa thông qua hàm softmax:

$$\alpha_j^i = \frac{e^{\gamma_j^i/\tau}}{\sum_{l=1}^k e^{\gamma_l^i/\tau}} \quad (2)$$

Trong đó, $\tau \in R^+$ là một tham số nhiệt độ do người dùng định nghĩa để điều chỉnh hàm softmax. Với một tập hợp các γ_{ij} đã cho, khi sử dụng giá trị nhiệt độ cao ($\tau \rightarrow \infty$), tất cả các khái niệm có độ quan trọng gần như tương đương nhau. Với các giá trị nhiệt độ thấp ($\tau \rightarrow 0$), xác suất của khái niệm có độ quan trọng nhất sẽ tiến gần tới $\alpha_{ij} \approx 1$, trong khi các khái niệm khác sẽ có $\alpha_{ik} \approx 0$, với $k \neq j$. Vì phân phối xác suất α_i làm nổi bật các khái niệm quan trọng nhất, thông tin này sẽ được đưa trực tiếp trở lại đầu vào, điều chỉnh trọng số của các khái niệm dựa trên mức độ quan trọng ước tính. Để tránh sự triệt tiêu số học do các giá trị trong α_i gần bằng không, đặc biệt khi không gian đầu vào có độ chiều lớn, nhóm tác giả thay thế α_i bằng phiên bản chuẩn hóa của nó, $\tilde{\alpha}_i$, vẫn trong phạm vi $[0, 1]^k$, và mỗi mẫu đầu vào $c \in C$ sẽ được điều chỉnh theo mức độ quan trọng ước tính này.

$$\tilde{c}^i = c \odot \tilde{\alpha}^i \quad \text{với} \quad \tilde{\alpha}_j^i = \frac{\alpha_j^i}{\max_u \alpha_u^i}, \quad (3)$$

Trong đó, \odot biểu thị phép nhân Hadamard (nhân theo phần tử). Giá trị cao nhất trong $\tilde{\alpha}_i$ luôn bằng 1 (tức là $\max_j \tilde{\alpha}_{ij} = 1$) và nó tương ứng với khái niệm quan trọng nhất. Các nhúng h_i được tính toán như trong bất kỳ lớp tuyến tính nào thông qua phép biến đổi affine:

$$h^i = W^i \mathbf{z}^i + b^i. \quad (4)$$

Khi $\tilde{\alpha}_{ij} \rightarrow 0$, đầu vào $\tilde{c}_{ij} \rightarrow 0$. Điều này có nghĩa là khái niệm tương ứng có xu hướng bị loại bỏ, và mạng f_i sẽ học cách dự đoán lớp thứ i mà không phụ thuộc vào khái niệm thứ j .

Để tạo ra các giải thích logic, lớp tuyến tính được đề xuất sẽ tạo ra bảng chân trị T_i , biểu diễn một cách chính thức hành vi của mạng nơ-ron dưới dạng các biểu diễn giống Boolean của các khái niệm đầu vào. Cụ thể, nhóm tác giả ký hiệu \bar{c} là diễn giải Boolean của bộ đầu vào $c \in C$, trong khi $\mu_i \in \{0, 1\}^k$ là mặt nạ nhị phân liên kết với $\tilde{\alpha}_i$. Để mã hóa thành kiến suy diễn của con người hướng đến các giải thích đơn giản, mặt nạ μ_i được sử dụng để tạo ra bộ khái niệm nhị phân \hat{c}_i , loại bỏ các khái niệm ít quan trọng nhất khỏi c .

$$\hat{c}^i = \xi(c, \mu^i) \quad \text{với} \quad \mu^i = I_{\alpha^i \geq \epsilon} \quad \text{và} \quad \bar{c} = I_{c \geq \epsilon}, \quad (5)$$

Trong đó, $I_{z \geq \epsilon}$ biểu thị hàm chỉ báo, có giá trị bằng 1 khi tất cả các thành phần của vectơ z lớn hơn hoặc bằng ϵ , và bằng 0 trong các trường hợp khác (đối với trường hợp không thiên lệch, nhóm tác giả đặt $\epsilon = 0.5$). Hàm ξ trả về vectơ chứa các thành phần của \bar{c} tương ứng với các giá trị bằng 1 trong μ_i (tức là nó chọn lọc dữ liệu từ \bar{c}). Do đó, \hat{c}_i thuộc không gian C_i^\wedge gồm m_i đặc trưng Boolean, với $m_i < k$ do tác động của quá trình chọn lọc.

$$\mathcal{T}^i = (\hat{C}^i \parallel \bar{F}^i). \quad (6)$$

Chính xác hơn, bất kỳ bảng chân trị T_i nào cũng giống như một bảng chân trị thực nghiệm hơn là một bảng chân trị cổ điển tương ứng với một hàm Boolean n -ngôi. Thực tế, T_i có thể chứa các hàng lặp lại và thiếu các bộ Boolean. Tuy nhiên, T_i vẫn có thể được sử dụng để tạo ra các giải thích logic theo cách tương tự.

Hàm mất mát

Entropy của phân phối xác suất α_i (Phương trình 7), được tối thiểu hóa khi một α_{ij} bằng 1, đại diện cho trường hợp cực đoan trong đó chỉ có một khái niệm quan trọng, trong khi nó đạt giá trị cực đại khi tất cả các khái niệm đều có độ quan trọng như nhau.

$$\mathcal{H}(\alpha^i) = - \sum_{j=1}^k \alpha_j^i \log \alpha_j^i \quad (7)$$

Khi entropy H được tối thiểu hóa cùng với hàm mất mát thông thường cho học có giám sát $L(f, y)$ (với y là nhãn mục tiêu – nhóm tác giả sử dụng cross-entropy trong các thí nghiệm của mình), nó cho phép mô hình tìm ra sự cân bằng giữa chất lượng khớp và việc kích hoạt tiết kiệm các khái niệm, cho phép mỗi mạng f_i dự đoán sự tham gia của lớp thứ i chỉ bằng cách sử dụng một số ít các khái niệm có liên quan. Tổng thể, hàm mất mát để huấn luyện mạng f được định nghĩa như sau:

$$\mathcal{L}(f, y, \alpha_1, \dots, \alpha_r) = L(f, y) + \lambda \sum_{i=1}^r \mathcal{H}(\alpha^i), \quad (8)$$

Trong đó, $\lambda > 0$ là siêu tham số được sử dụng để cân bằng tầm quan trọng tương đối của các giải pháp có entropy thấp trong hàm mất mát. Các giá trị cao hơn của λ dẫn đến cấu hình thưa thớt hơn của α , buộc mạng phải tập trung vào một tập hợp khái niệm nhỏ hơn cho mỗi nhiệm vụ phân loại (và ngược lại), do đó mã hóa thiên kiến suy diễn của con người hướng tới các giải thích đơn giản.

Tài liệu tham khảo

- [1] A. Alotaibi. Wisdom of the machines: Federated learning using opal. Master’s thesis, 2018. Available at: MIT Libraries.
- [2] Pietro Barbiero, Gabriele Ciravegna, Francesco Giannini, Pietro Lió, Marco Gori, and Stefano Melacci. Entropy-based logic explanations of neural networks. Proceedings of the AAAI Conference on Artificial Intelligence, 36(6):6046–6054, June 2022.
- [3] Gianni A. Di Caro. 15-382 collective intelligence – s19, lecture 33: Wisdom of the crowd. Technical report, 2019. Available at: <https://web2.qatar.cmu.edu/~gdicaro/15382/slides/382-S19-33-WisdomCrowd.pdf>.
- [4] UCI Machine Learning Repository and 1 collaborator. Default of credit card clients dataset. Available at: <https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset>, n.d.
- [5] Sina Sheikholeslami. Ablation programming for machine learning. Master’s thesis, KTH, School of Electrical Engineering and Computer Science (EECS), 2019.
- [6] Donatas Tamosauskas. Using resnet for mnist. Available at: <https://www.kaggle.com/code/donatastamosauskas/using-resnet-for-mnist/notebook>, n.d.
- [7] Yanci Zhang and Han Yu. LR-XFL: Logical reasoning-based explainable federated learning, 2023.

LR-XFL: Học liên kết phân tán giải thích được dựa trên lý luận logic

Yanci Zhang, Han Yu

Khoa Khoa học Máy tính và Kỹ thuật, Đại học Công nghệ Nanyang, Singapore
Trung tâm nghiên cứu chung về công nghệ tài chính của NTU-WeBank, Đại học công nghệ Nanyang, Singapore
{yanci001, han.yu}@ntu.edu.sg

Abstract

Học liên kết là một phương pháp mới nổi để đào tạo các mô hình học máy theo cách phối hợp, trong khi vẫn bảo vệ quyền riêng tư của dữ liệu. Nhu cầu bảo vệ quyền riêng tư khiến cho các mô hình FL khó đạt được tính minh bạch toàn cục và khả năng giải thích. Để khắc phục hạn chế này, chúng tôi kết hợp các lời giải thích dựa trên logic vào FL bằng cách đề xuất phương pháp Học Liên kết giải thích dựa trên Lý luận Logic (LR-XFL). Theo LR-XFL, các máy khách FL tạo các quy tắc logic cục bộ dựa trên dữ liệu cục bộ của chúng và gửi đi, cùng với các bản cập nhật mô hình đến máy chủ FL. Máy chủ FL kết nối các quy tắc logic cục bộ thông qua một trình kết nối logic thích hợp, được xác định dựa trên các đặc tính của dữ liệu khách hàng, mà không cần truy cập vào dữ liệu gốc. Ngoài ra, máy chủ cũng tổng hợp các bản cập nhật mô hình cục bộ từ máy khách với các trọng số được xác định dựa trên chất lượng dữ liệu cục bộ của từng thiết bị, được phản ánh qua các quy tắc logic mà chúng tải lên. Kết quả cho thấy LR-XFL vượt trội hơn so với baseline liên quan nhất với tỷ lệ lần lượt là 1,19%, 5,81% và 5,41% về độ chính xác phân loại, độ chính xác quy tắc và độ tin cậy quy tắc. Việc đánh giá và biểu đạt quy tắc một cách rõ ràng theo phương pháp LR-XFL cho phép các chuyên gia xác thực và sửa các quy tắc ở phía máy chủ, từ đó cải thiện độ chính xác mô hình FL toàn cục. Điều này có tiềm năng nâng cao tính minh bạch của các mô hình FL trong các lĩnh vực như y tế và tài chính, nơi mà cả quyền riêng tư dữ liệu và khả năng giải thích đều rất quan trọng.

Giới thiệu

Học liên kết (FL) (Kairouz, McMahan và cộng sự, 2021) là một mô hình đào tạo hợp tác cùng đào tạo các mô hình trí tuệ nhân tạo (AI) từ một nhóm máy khách FL mà không để lộ dữ liệu cục bộ của chúng. Dưới sự hướng dẫn của máy chủ trung tâm, các máy khách cải thiện mô hình cục bộ của mình dựa trên thông tin thu nhận được từ các máy khách khác. Ý tưởng chung là các máy khách sẽ tải lên các bản cập nhật mô hình của mình thay vì dữ liệu lên máy chủ FL. Máy chủ tổng hợp các mô hình nhận được và trả về mô hình toàn cục đã cập nhật cho máy khách để có thể đào tạo thêm. Tuy nhiên, cơ chế huấn luyện chung nhằm bảo vệ quyền riêng tư lại gây ra những thách thức, đặc biệt là trong việc giải thích quá trình ra quyết định, do máy chủ không có quyền truy cập trực tiếp vào tập dữ liệu máy khách.

Trí tuệ nhân tạo giải thích (XAI) (Gunning và cộng sự,

2019; Xu và cộng sự, 2019; Yu và cộng sự, 2014) đang thu hút sự chú ý trong những năm gần đây. Mục tiêu của XAI là giúp con người hiểu rõ hơn về hành vi của các mô hình AI thông qua việc cung cấp lời giải thích. Một loạt các nghiên cứu đã thành công trong việc cung cấp lời giải thích cho các mô hình AI hộp đen. Chẳng hạn, các phương pháp như Grad-CAM (Selvaraju và cộng sự, 2017) phân tích đầu ra lớp tích chập cuối cùng của một mạng nơ-ron nhất định và tính toán cách mà những thay đổi ở từng vùng của hình ảnh đầu vào ảnh hưởng đến đầu ra như thế nào, từ đó xác định các vùng ảnh hưởng quan trọng nhất cho quá trình ra quyết định. Các thuật toán không phụ thuộc vào mô hình như SHAP (Lundberg and Lee 2017) và LIME (Ribeiro, Singh, and Guestrin 2016) đánh giá tầm quan trọng của các tính năng cho từng trường hợp cụ thể. Tuy nhiên, những phương pháp này không thể làm rõ quá trình ra quyết định của mô hình, mà chủ yếu cung cấp phân tích post-hoc của các mô hình.

Các mô hình dựa trên khái niệm, như một cầu nối giữa trí tuệ nhân tạo biểu tượng (dựa trên luật và logic) và trí tuệ nhân tạo thống kê (ví dụ: mạng nơ-ron), trích xuất các khái niệm từ mô hình bằng cách ánh xạ thông tin ẩn của lớp nơ-ron cuối cùng trong mạng nơ-ron sang các khái niệm mà con người có thể hiểu được (Kim và cộng sự, 2018) hoặc ràng buộc các khái niệm như một phần của mạng nơ-ron (Koh và cộng sự, 2020). Tuy nhiên, các khái niệm này vẫn còn rời rạc và chỉ ở cấp độ từng trường hợp cụ thể. Các luật logic, ngược lại, có thể kết nối các khái niệm được kích hoạt trong quá trình dự đoán để minh họa quá trình suy luận (Lee và cộng sự, 2022). Các luật logic ở cấp độ trường hợp cụ thể có thể được kết nối thêm để hình thành các luật tổng quát ở cấp độ lớp (Barbiero và cộng sự, 2022). Những lời giải thích dựa trên logic như vậy rất hữu ích vì chúng minh họa quá trình ra quyết định theo cách dễ dàng được con người hiểu.

Các mô hình khái niệm dựa trên logic truyền thống được thiết kế cho các framework AI tập trung và không thể áp dụng trực tiếp trong cài đặt FL. Việc tích hợp các quy tắc logic vào FL phải đối mặt với ba thách thức quan trọng:

1. **Độ chính xác cục bộ so với tính đại diện toàn cục:** Các máy khách chỉ có quyền truy cập vào một phần dữ liệu hoặc dữ liệu dễ bị lệch, có thể đưa ra những quy tắc có vẻ đúng trong phạm vi cục bộ của chúng nhưng lại dễ dẫn đến thiên lệch hoặc làm sai lệch bức tranh tổng thể ở cấp độ toàn cục. Máy chủ bắt buộc phải tạo ra các quy tắc toàn diện và chính xác trên toàn cục.

2. **Giải quyết xung đột và kết hợp quy tắc:** Việc hợp nhất các quy tắc logic tại một máy chủ trung tâm có thể dẫn đến xung đột. Việc kết nối đơn giản các quy tắc cục bộ bằng toán tử logic ‘AND’ có thể gây ra xung đột và việc kết hợp các quy tắc với toán tử ‘OR’ có thể làm giảm độ chính xác quy tắc tổng thể.
3. **Phân bổ trọng số cho máy khách FL:** Việc chỉ định trọng số phù hợp cho các bản cập nhật mô hình máy khách trong quá trình tổng hợp dựa trên các quy tắc logic của chúng lại đặt ra một thách thức khác.

Để giải quyết những thách thức này, chúng tôi đề xuất phương pháp Học Liên kết giải thích dựa trên Lý luận Logic (LR-XFL). Theo LR-XFL, các máy khách FL tạo các quy tắc logic cục bộ dựa trên dữ liệu cục bộ của chúng và gửi đi, cùng với các bản cập nhật mô hình đến máy chủ FL. Máy chủ FL kết nối các quy tắc logic cục bộ thông qua một trình kết nối logic (AND hoặc OR) được LR-XFL xác định linh hoạt dựa trên các đặc tính của dữ liệu từ phía máy khách, mà không cần tiết lộ dữ liệu gốc. Ngoài ra, máy chủ cũng tổng hợp các bản cập nhật mô hình cục bộ với các giá trị trọng số được xác định dựa trên chất lượng dữ liệu cục bộ của máy khách, được thể hiện qua các quy tắc logic mà chúng tải lên. Theo hiểu biết của chúng tôi, LR-XFL là phương pháp lý luận FL đầu tiên có khả năng tổng hợp linh hoạt các quy tắc cục bộ từ các máy khách.

Để đánh giá hiệu năng của LR-XFL, chúng tôi tiến hành các thử nghiệm mở rộng trên bốn tập dữ liệu chuẩn trong cài đặt FL¹. So với ba phương pháp thay thế hiện hành, LR-XFL đã chứng minh được những lợi thế đáng kể. Nó vượt trội hơn so với *baseline* liên quan nhất với tỷ lệ lần lượt là 1,19%, 5,81% và 5,41% về độ chính xác phân loại, độ chính xác quy tắc và độ tin cậy quy tắc. Việc đánh giá và biểu đạt quy tắc một cách rõ ràng theo phương pháp LR-XFL cho phép các chuyên gia xác thực và sửa các quy tắc ở phía máy chủ, từ đó cải thiện độ chính xác mô hình FL toàn cục trước các lỗi. Điều này có tiềm năng nâng cao tính minh bạch của các mô hình FL trong các lĩnh vực như y tế và tài chính, nơi mà cả quyền riêng tư dữ liệu và khả năng giải thích đều rất quan trọng.

Công trình liên quan

AI giải thích theo khái niệm

XAI dựa trên khái niệm tập trung vào việc nhận diện các khái niệm hoặc trừu tượng hóa cấp cao trong dữ liệu. Trong các mô hình học sâu truyền thống, các tầng dưới thường phát hiện các cạnh hoặc kết cấu, trong khi các tầng trên nhận diện các đặc trưng trừu tượng hơn như hình dạng hoặc đối tượng. Học dựa trên khái niệm coi các đặc trưng trừu tượng là “khái niệm” và hướng đến việc làm cho chúng trở nên rõ ràng hơn. Các khái niệm thường thu được bằng cách kết nối thông tin ẩn từ lớp cuối cùng của mạng nơ-ron thành những khái niệm dễ hiểu (Kim và cộng sự, 2018; Kazhdan và cộng sự, 2020), hoặc bằng cách ràng buộc cấu trúc của mạng nơ-ron để học các khái niệm này (Chen, Bei, and Rudin 2020; Ciravegna và cộng sự, 2020; Koh và cộng sự, 2020; Stammer, Schramowski, and Kersting 2021; Barbiero và cộng

sự, 2022). Mô hình dựa trên khái niệm được thiết kế để cung cấp những giải thích rõ ràng cho quá trình ra quyết định, nhờ vào tính trực quan và dễ hiểu của các khái niệm. Do đó làm cho quá trình ra quyết định trở nên minh bạch. Chẳng hạn, nếu một mô hình được huấn luyện trên các hình ảnh động vật nhận biết được khái niệm “cánh” và “mỏ”, nó có thể giải thích quyết định phân loại bằng cách lập luận rằng sự xuất hiện của cánh trong một bức ảnh là dấu hiệu rõ ràng cho thấy đó là danh mục “chim”.

Các quy tắc logic đóng vai trò như một phương tiện để liên kết các khái niệm đã được trích xuất, từ đó giải thích lý do đằng sau những quyết định của mô hình, dựa trên những khái niệm mà nó đã học được. Những quy tắc logic này thường rất đơn giản và mang tính tất định. Các hệ thống dựa trên quy tắc truyền thống, như Cây Quyết Định (Quinlan 1986), cung cấp cách giải thích trực quan thông qua các quy tắc logic. Trong Cây Quyết Định, mỗi nhánh quyết định có thể được hiểu như một quy tắc riêng biệt. Tuy nhiên, việc liệt kê tất cả các thuộc tính trong một nhánh quyết định có thể khiến quy tắc trở nên quá dài và chứa nhiều yếu tố không cần thiết, rốt cục lại làm cho mô hình trở nên khó hiểu hơn. Trong lĩnh vực xử lý ngôn ngữ tự nhiên, lý luận logic đã được áp dụng cho các nhiệm vụ như phân tích cảm xúc (Lee và cộng sự, 2022) và dự đoán văn bản (Jain và cộng sự, 2022). Đối với dữ liệu hình ảnh và dữ liệu bảng, một lớp tuyến tính mới dựa trên entropy đã được phát triển để tạo ra các giải thích theo quy tắc, giúp làm rõ cách mô hình đưa ra quyết định (Barbiero và cộng sự, 2022). Tuy nhiên, các phương pháp này yêu cầu truy cập trực tiếp vào dữ liệu đào tạo, nên không thể áp dụng hiệu quả trong cài đặt FL.

Học liên kết

FL cung cấp phương pháp phân quyền và bảo vệ quyền riêng tư để đào tạo các mô hình AI trên dữ liệu được sở hữu phân tán, thuộc sở hữu của nhiều nguồn khác nhau (Kairouz, McMahan và cộng sự, 2021). Thay vì truyền dữ liệu gốc có thể chứa thông tin nhạy cảm, chỉ các bản cập nhật mô hình được gửi qua lại giữa máy chủ FL và các máy khách FL. Tuy nhiên, cách tiếp cận huấn luyện này khiến cho việc giải thích lý do đằng sau các quyết định của mô hình FL trở nên khó khăn. Lĩnh vực FL đã nhận ra tiềm năng của các quy tắc logic để giải quyết thách thức này. Đã có nhiều nỗ lực đáng kể nhằm kết hợp các quy tắc logic vào trong FL.

Trong (An and Ma 2023), logic thời gian tín hiệu được sử dụng để phân biệt các đặc tính của các thiết bị máy khách, từ đó được tận dụng để phân nhóm chúng trong quá trình tổng hợp mô hình, nhằm tạo ra các mô hình FL cá nhân hóa. Tương tự, trong (Cha và cộng sự, 2022), logic mờ được sử dụng để cải thiện việc chọn lựa máy khách trong các mạng phương tiện. Bên cạnh đó, trong (Zhu và cộng sự, 2021), quy tắc mờ Takagi-Sugeno được tích hợp vào FL để thực hiện phân cụm mờ phân tán. Tuy nhiên, các phương pháp này chủ yếu tập trung vào việc chọn lựa máy khách FL và phân nhóm. Chúng không được thiết kế để giải quyết các thách thức trong việc xây dựng mô hình FL có thể giải thích được bằng lý luận logic. Phương pháp LR-XFL đề xuất giúp lấp đầy khoảng trống quan trọng này.

¹Mã nguồn có sẵn tại <https://github.com/Yanci87/LR-XFL>.

Cơ sở

Mô hình cơ sở trong phương pháp LR-XFL được đề xuất là các giải thích logic dựa trên entropy của mạng nơ-ron (Barbiero và cộng sự, 2022). Mô hình này có khả năng trích xuất các giải thích logic từ mạng nơ-ron một cách hiệu quả và thể hiện chúng dưới dạng các quy tắc logic. Được thiết kế để xử lý cả dữ liệu hình ảnh và bảng, mô hình này không chỉ đưa ra kết quả phân loại mà còn kèm theo các giải thích dựa trên quy tắc. Đối với dữ liệu hình ảnh, mô hình đầu tiên sử dụng mạng xử lý hình ảnh ResNet10 (He và cộng sự, 2016) để ánh xạ hình ảnh từ không gian pixel sang không gian khái niệm. Đối với dữ liệu dạng bảng, việc ánh xạ có thể được thực hiện thông qua các mô hình tuyến tính. Sau đó, không gian khái niệm được ánh xạ tới lớp mục tiêu bằng lớp tuyến tính dựa trên entropy. Các khái niệm được kích hoạt từ tầng này sẽ được suy ra từ các tham số của lớp tuyến tính dựa trên entropy, từ đó tạo thành các quy tắc logic.

Quá trình tạo ra các quy tắc logic từ lớp tuyến tính dựa trên entropy dựa vào một bảng chân lý (truth table), ký hiệu là T_c , tương ứng với lớp c . Bảng chân lý T_c phản ánh hoạt động của các mạng nơ-ron thông qua các biểu diễn dạng Boolean của các khái niệm đầu vào. Cụ thể, mỗi hàng của T_c thể hiện các khái niệm được kích hoạt cho một mẫu dự đoán thuộc lớp c . Trạng thái kích hoạt của các khái niệm này đối với một điểm dữ liệu cụ thể được xác định bằng cách xử lý vector khái niệm của nó thông qua một mặt nạ nhị phân, được lấy từ các tham số của lớp tuyến tính dựa trên entropy. Các khái niệm được kích hoạt thông qua mặt nạ nhị phân cho dự đoán thuộc lớp c sẽ được đưa vào, trong khi những khái niệm khác sẽ bị loại bỏ. Đối với một khái niệm được kích hoạt f_i , biểu diễn của nó sẽ là f_i hoặc $\neg f_i$, tùy thuộc vào giá trị của nó trong điểm dữ liệu đầu vào. Mỗi hàng trong bảng chân lý sẽ tạo ra một giải thích dựa trên quy tắc cho cấp mẫu bằng cách kết nối các khái niệm đã được kích hoạt qua toán tử AND. Giải thích cho cấp lớp c được xây dựng bằng cách kết nối các quy tắc cấp mẫu tương ứng với dữ liệu thuộc lớp c bằng toán tử OR.

Tuy nhiên, việc luôn sử dụng toán tử OR để kết nối các quy tắc mẫu có thể vô tình làm giảm độ chính xác quy tắc. Xem xét hai quy tắc cấp mẫu sau: 1) $hasBeak \leftrightarrow Bird$, và 2) $hasWing \leftrightarrow Bird$. Khi kết hợp các quy tắc này bằng toán tử OR, ta có: $hasBeak \vee hasWing \leftrightarrow Bird$. Tuy nhiên, quy tắc tổng hợp này lại phân loại sai các mẫu chỉ có mỏ và không có cánh, hoặc chỉ có cánh và không có mỏ là chim. Trong trường hợp này, toán tử AND sẽ phù hợp hơn vì hai quy tắc mẫu chứa thông tin bổ sung từ các góc độ khác nhau. Phương pháp LR-XFL được đề xuất nhằm giải quyết những tình huống khó khăn như vậy.

Phương pháp LR-XFL được đề xuất

Trong phần này, chúng tôi sẽ trình bày chi tiết thiết kế của LR-XFL (Hình 1), một khung giải thích dựa trên logic đầu tiên được thiết kế riêng cho các cài đặt FL, dựa trên mạng lưới dùng entropy (Barbiero và cộng sự, 2022). Phương pháp này giới thiệu cách tiếp cận mới để tự động xác định toán tử logic phù hợp (AND hoặc OR). Đồng thời, nó cũng giải quyết thách thức trong việc chọn lọc và kết hợp các quy tắc do các máy khách tạo ra. Trong quá trình tổng hợp mô hình FL, LR-XFL xác định trọng số của từng máy khách dựa trên

quy tắc của chúng, giúp thực hiện trung bình có trọng số một cách hiệu quả.

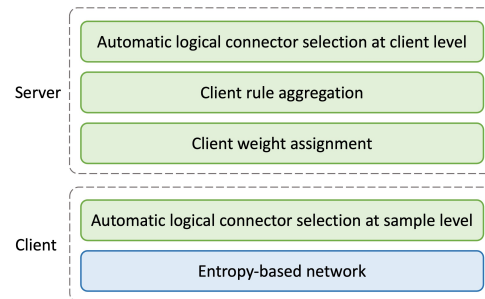
Tổng quan về LR-XFL

Hình 2 mô tả kiến trúc và quy trình hoạt động của LR-XFL. Ở phía máy khách, các mạng dựa trên entropy trong nghiên cứu của (Barbiero và cộng sự, 2022) được sử dụng làm mô hình nền tảng. Mỗi máy khách FL sẽ sở hữu một tập hợp quy tắc được suy ra từ mạng lưới dựa trên entropy này. Máy chủ FL tổng hợp các quy tắc toàn cục từ quy tắc cục bộ và gửi mô hình toàn cục lại cho các máy khách. Thuật toán 1 cung cấp cái nhìn tổng quan về LR-XFL. Trong LR-XFL, các máy khách FL xây dựng các quy tắc logic cục bộ dựa trên bộ dữ liệu riêng của họ. Các quy tắc này, cùng với các bản cập nhật mô hình, được truyền tới máy chủ FL. Đáng chú ý, máy chủ FL tổng hợp các quy tắc logic cục bộ thông qua một phép nối logic phù hợp (\wedge hoặc \vee), được xác định dựa trên đặc điểm vốn có trong dữ liệu của máy khách, mà không cần truy cập vào dữ liệu gốc. Thêm vào đó, máy chủ sẽ tổng hợp các bản cập nhật mô hình cục bộ và gán giá trị trọng số dựa trên chất lượng dữ liệu cục bộ của từng máy khách, được đánh giá dựa trên các quy tắc logic cục bộ của chúng. Quá trình huấn luyện lặp đi lặp lại này sẽ tiếp tục cho đến khi đạt đến số lượng vòng lặp tối đa T đã được định trước, hoặc khi mô hình toàn cục đạt được hiệu năng mong muốn trên tập dữ liệu kiểm định phía máy chủ.

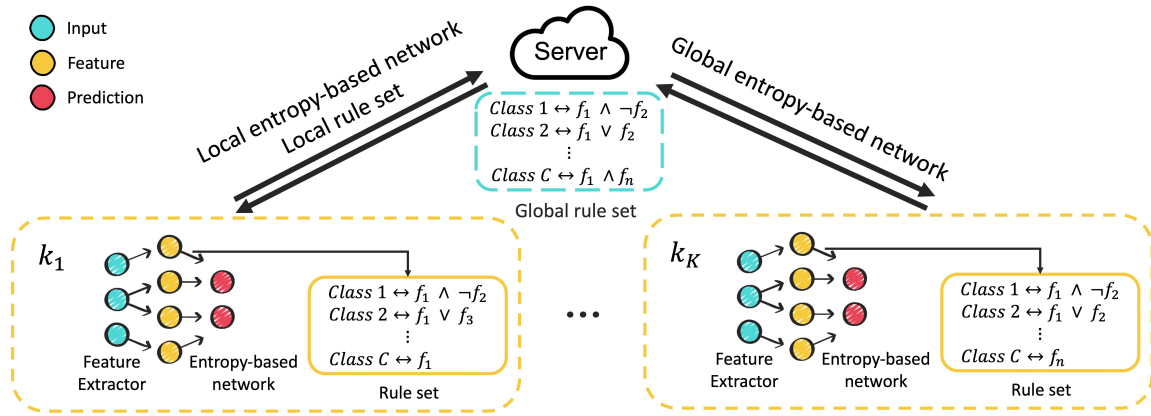
Xác định kết nối logic OR hoặc AND

Lý tưởng nhất, các quy tắc toàn cục chỉ nên bao gồm những quy tắc đúng với kết nối logic chính xác. Khi tích hợp các quy tắc logic cục bộ, bước đầu tiên là phải xác định nên sử dụng kết nối logic AND hay OR. Kết nối các quy tắc cục bộ bằng AND có thể dẫn đến xung đột, trong khi việc sử dụng OR để kết nối các quy tắc sai hoặc chưa hoàn chỉnh có thể làm giảm độ chính xác.

Chúng tôi đưa ra giả thuyết rằng cơ sở lý luận cơ bản để lựa chọn giữa AND và OR phụ thuộc vào khả năng xung đột giữa các đặc trưng cục bộ của các máy khách. Nếu tất cả các đặc trưng loại trừ lẫn nhau (tức là chúng không thể cùng tồn tại trong một điểm dữ liệu), thì nên sử dụng kết nối \vee (OR). Một ví dụ cho tình huống này là chuỗi phân loại $1 \vee 3 \vee 5 \vee 7 \vee 9$, biểu thị cho mục 'số lẻ'. Ngược lại, trong các trường hợp mà các đặc trưng không xung đột về bản chất và có thể xuất hiện cùng nhau trong một điểm dữ liệu, thì kết nối \wedge (AND) là lựa



Hình 1: Thiết kế tổng quan của LR-XFL.



Hình 2: Kiến trúc và quy trình hoạt động của LR-XFL.

Algorithm 1: LR-XFL

Input: K máy khách, mỗi máy khách có một tập dữ liệu cục bộ; một máy chủ giữ tập dữ liệu để kiểm thử và xác thực logic
Output: Các quy tắc logic toàn cục cho máy chủ; mô hình và quy tắc logic cục bộ cho từng máy khách

- 1: **while** Mô hình toàn cục chưa đạt được hiệu năng mục tiêu trên tập dữ liệu xác thực **and** số vòng huấn luyện tối đa chưa được hoàn thành **do**
- 2: **Đối với mỗi máy khách FL** $k, k \in \{1, \dots, K\}$:
- 3: Tiến hành huấn luyện mô hình cục bộ;
- 4: Chọn bộ kết nối logic phù hợp cho các quy tắc logic cục bộ;
- 5: Tạo ra các quy tắc logic r_k^c cho từng lớp $c \in \{1, \dots, C\}$;
- 6: Tải lên mô hình cục bộ và các quy tắc logic tới máy chủ FL;
- 7: **chủ FL**:
- 8: Lựa chọn bộ kết nối logic toàn cục dựa trên các quy tắc logic từ máy khách;
- 9: Lựa chọn và tổng hợp các quy tắc logic cục bộ từ máy khách;
- 10: Tính toán và gán trọng số $\{w_1, \dots, w_K\}$ cho các máy khách dựa trên hiệu năng quy tắc logic của chúng;
- 11: Tổng hợp các mô hình cục bộ thành mô hình toàn cục dựa trên các trọng số đã gán;
- 12: Gửi mô hình toàn cục lại cho các máy khách;
- 13: **Đối với mỗi máy khách FL** k : Nhận mô hình toàn cục và tiếp tục huấn luyện cho vòng tiếp theo;
- 14: **end while**

chọn phù hợp hơn. Ví dụ, trong nhiệm vụ phân loại động vật, các đặc trưng ‘cánh’ và ‘mỏ’ có thể cùng tồn tại trong mục ‘chim’. Do đó, một quy tắc như $Cnh \wedge M \leftrightarrow Chim$ là phù hợp để phân loại chim.

Để xác định kết nối phù hợp, chúng tôi đã thiết kế một ma trận đồng hiện dương và một ma trận đồng hiện âm nhằm khám phá các xung đột tiềm năng giữa các đặc trưng. Ma

trận đồng hiện ghi lại tần suất hai đặc trưng xuất hiện cùng nhau trong cùng một quy tắc. Cụ thể, hãy xem xét tập hợp quy tắc cấp mẫu R_k^c cho lớp c trong khách hàng k , trong đó mỗi quy tắc r được định nghĩa như sau: $f_1 \wedge \neg f_2 \wedge \dots \wedge f_n$:

1. **Ma trận Đồng hiện Dương** ghi lại số lần các đặc trưng f_i và f_j cùng xuất hiện trong các quy tắc dưới dạng $f_i \wedge \dots \wedge f_j$. Nếu các đặc trưng f_i và f_j cùng xuất hiện trong quy tắc t lần, thì $p_{ij} = p_{ji} = t$.

$$M_{pos} = \begin{bmatrix} p_{11} & \dots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \dots & p_{nn} \end{bmatrix}. \quad (1)$$

2. **Ma trận Đồng hiện Âm** ghi lại số lần các đặc trưng f_i và f_j cùng xuất hiện trong các quy tắc dưới dạng $f_i \wedge \dots \wedge \neg f_j$. Nếu các đặc trưng f_i và $\neg f_j$ xuất hiện trong quy tắc t lần, thì $q_{ij} = t$. Tuy nhiên, khác với ma trận đồng hiện dương, trong ma trận đồng hiện âm, $q_{ij} \neq q_{ji}$. q_{ij} ghi lại số lần f_i và $\neg f_j$ xuất hiện trong một quy tắc, trong khi q_{ji} ghi lại số lần f_j và $\neg f_i$ cùng xuất hiện.

$$M_{neg} = \begin{bmatrix} q_{11} & \dots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{n1} & \dots & q_{nn} \end{bmatrix}. \quad (2)$$

Chúng tôi đề xuất hai tiêu chí để đánh giá mức độ xung đột đặc trưng thông qua Ma trận Đồng hiện Dương M_{pos} và Ma trận Đồng hiện Âm M_{neg} .

1. **Tính chéo**, được tính bằng công thức:

$$D = \frac{\sum_{i=1}^n p_{ii}}{\sum_{i=1}^n \sum_{j=1}^n p_{ij}} \quad (3)$$

Trong đó, n là tổng số đặc trưng. Tính chéo cao cho thấy rằng, một đặc trưng có khả năng xuất hiện trong một quy tắc mà không kèm theo đặc trưng nào khác ngoài chính nó (tức là khả năng cao xảy ra xung đột giữa các đặc trưng).

2. **Tính riêng biệt**, được tính bằng công thức:

$$E = \frac{\max_{i=1}^n \left(\sum_{j=1}^n q_{ij} \right)}{\sum_{m=1}^M l_{r_m}} \quad (4)$$

Trong đó, l_{r_m} là độ dài của quy tắc r_m và M là tổng số quy tắc đóng góp vào ma trận đồng hiện âm M_{neg} . Tính riêng biệt E đo lường số lần đồng hiện âm trung bình của các đặc trưng của một máy khách. Tính riêng biệt cao chỉ ra rằng một quy tắc có khả năng có dạng $f_i \wedge \neg f_j \wedge \dots \wedge \neg f_n$, biểu thị các xung đột tiềm ẩn giữa các đặc trưng.

Khi tính chéo hoặc tính riêng biệt vượt quá một ngưỡng siêu tham số được định trước, kết nối logic sẽ được chỉ định là OR do xác suất xung đột giữa các đặc trưng tăng cao. Ngược lại, nếu không vượt ngưỡng, kết nối sẽ được xác định là AND. Cả tính chéo và tính riêng biệt đều được tính toán tại phía máy khách, dựa trên các quy tắc mẫu của chúng. Trong quá trình tổng hợp quy tắc toàn cục, máy chủ FL sử dụng cơ chế bỏ phiếu đa số từ tất cả các kết nối logic của các máy khách để xác định kết nối toàn cục tối ưu. Lưu ý rằng, trong hầu hết các trường hợp, các kết nối logic được áp dụng là giống nhau cho tất cả các máy khách FL. Sự đồng thuận này nảy sinh do quá trình xác định các kết nối logic dựa trên các xung đột tiềm năng giữa các đặc trưng, vốn là một tính chất tự nhiên của các đặc trưng và không bị ảnh hưởng bởi cách dữ liệu được phân phối. Vì tất cả các máy khách đều chia sẻ cùng một không gian đặc trưng, các đánh giá về khả năng xung đột giữa các đặc trưng của chúng thường có xu hướng đồng nhất.

Tổng hợp Quy tắc Liên kết

Trong mỗi vòng huấn luyện FL, máy chủ nhận các mô hình cục bộ và các quy tắc logic cục bộ từ các máy khách. Đầu tiên, máy chủ xác định kết nối logic phù hợp theo phương pháp đã mô tả trong phần trước. Sau đó, máy chủ xác định một tập hợp các quy tắc tối ưu hóa hiệu năng mô hình. Trong một vòng huấn luyện nhất định, máy chủ FL nhận được K quy tắc liên quan đến lớp c từ K máy khách FL. Trên thực tế, số lượng quy tắc riêng biệt tối đa mà máy chủ có thể thu được ít hơn K vì một số lý do. Thứ nhất, không phải máy khách nào cũng có khả năng tạo ra các quy tắc đáng tin cậy. Chúng tôi thiết lập một ngưỡng độ chính xác để lọc các quy tắc (tức là chỉ những mô hình vượt quá ngưỡng này mới được coi là đáng tin cậy). Các quy tắc được lấy từ các mô hình có độ chính xác thấp hơn tiêu chuẩn sẽ không được coi là đáng tin cậy. Thứ hai, ngay cả khi một mô hình máy khách đạt được ngưỡng độ chính xác, nó vẫn có thể chứa dữ liệu thiên lệch không hỗ trợ cho lớp dự đoán c . Do đó, không có quy tắc nào được tạo ra cho lớp c . Cuối cùng, có thể nhiều máy khách tạo ra các quy tắc giống hệt nhau.

Để xác định tổ hợp quy tắc tối ưu, chúng tôi tận dụng thuật toán tìm kiếm beam (Lowerre and Reddy 1976), một phương pháp tham lam thường được áp dụng trong xử lý ngôn ngữ tự nhiên và dịch máy. Thay vì khám phá mọi chuỗi có thể, tìm kiếm beam chỉ giữ lại t chuỗi tốt nhất tại mỗi bước và tiếp tục mở rộng chúng. Phương pháp này dẫn tới sự đánh đổi giữa chi phí tính toán và chất lượng giải pháp. Trong LR-XFL,

các chuỗi quy tắc được xếp hạng dựa trên giá trị độ chính xác quy tắc đối với tập dữ liệu kiểm định trên máy chủ FL.

Phân bổ Trọng số cho Máy khách trong FL

Trong FL, việc phân bổ trọng số phù hợp cho các máy khách là rất quan trọng, đặc biệt khi xử lý các tập dữ liệu thiên lệch hoặc có nhiễu. Trong LR-XFL, chúng tôi giới thiệu một phương pháp mới để tính toán trọng số máy khách dựa trên các quy tắc logic từ mỗi máy khách. Trọng số được gán cho một máy khách trong một vòng huấn luyện nhất định sẽ tỷ lệ thuận với tần suất các quy tắc của máy khách đó được máy chủ lựa chọn. Giả sử trong một vòng huấn luyện, máy khách k xây dựng C quy tắc trên C lớp (một số quy tắc có thể rỗng). Trong số các quy tắc này, p quy tắc được tổng hợp vào tập quy tắc toàn cục. Trọng số cho máy khách k trong vòng này được tính như sau:

$$w_k = \frac{p_k}{\sum_{i=1}^K p_i}. \quad (5)$$

Nếu một máy khách không tạo ra bất kỳ quy tắc hợp lệ nào hoặc nếu các quy tắc của nó không được máy chủ lựa chọn, thì w_k sẽ bằng 0. Có hai lý do khiến một quy tắc bị loại khỏi tập toàn cục: 1) độ chính xác quy tắc thấp, và 2) việc tích hợp quy tắc đó có thể làm suy giảm hiệu quả của các quy tắc toàn cục hiện tại. Một giá trị w_k cao cho thấy máy khách đã có những đóng góp đáng kể vào các quy tắc trên nhiều lớp trong vòng huấn luyện này, và do đó sẽ được gán trọng số cao hơn.

Phân tích Độ phức tạp Thời gian

Trong quá trình tạo quy tắc cho K máy khách, mỗi máy khách có N điểm dữ liệu, độ phức tạp cho một máy khách tạo ra quy tắc là $O(N)$. Xét cho C lớp, quá trình tổng hợp quy tắc cục bộ bao gồm việc xếp hạng quy tắc với độ phức tạp là $O(N \log N)$ và bao hàm lặp lại với độ phức tạp là $O(N)$. Do đó, độ phức tạp tổng thể cho quá trình tổng hợp quy tắc cục bộ là $O(N \log N)$. Trong giai đoạn tổng hợp quy tắc toàn cục, tìm kiếm beam được sử dụng với độ rộng beam là b , dẫn đến độ phức tạp tệ nhất là $O(bK^2)$ cho mỗi lớp. Tuy nhiên, vì N lớn hơn đáng kể so với K , độ phức tạp thời gian cho toàn bộ quá trình tạo và tổng hợp quy tắc là $O(N \log N)$.

Đánh giá thực nghiệm

Để đánh giá hiệu quả của LR-XFL, chúng tôi tiến hành các thí nghiệm trên bốn tập dữ liệu khác nhau và so sánh nó với ba phương pháp thay thế. Ngoài ra, chúng tôi đánh giá hiệu năng của LR-XFL trong điều kiện dữ liệu nhiễu. Kết quả được so sánh dựa trên ba chỉ số: 1) độ chính xác mô hình, 2) độ chính xác quy tắc và 3) độ tin cậy quy tắc. Một nghiên cứu loại bỏ cũng đã được thực hiện để làm nổi bật tầm quan trọng của phương pháp chọn bộ kết nối logic trong LR-XFL.

Thiết lập Thí nghiệm

Theo thiết lập bộ dữ liệu (Barbiero và cộng sự, 2022), chúng tôi áp dụng bốn bộ dữ liệu chuẩn: 1) MNIST(Even/Odd) (Le-Cun 1998), 2) CUB (Wah và cộng sự, 2011), 3) V-Dem (Coppedge và cộng sự, 2021) và 4) MIMIC-II (Saeed và cộng sự, 2011). MNIST và CUB được thiết kế theo dạng

“ảnh → đặc trưng → lớp”. V-Dem và MIMIC là các tập dữ liệu dạng bảng, trực tiếp ánh xạ đầu vào tới các lớp. Cụ thể, trong MNIST(Even/Odd), bộ dữ liệu được tăng cường mở rộng từ mô hình gốc “ảnh → chữ số” trong MNIST thành mô hình “ảnh → chữ số → tính chẵn lẻ”, trong đó các chữ số được trích xuất là đặc trưng và tính chẵn lẻ là dự đoán cuối cùng. Đối với mỗi bộ dữ liệu, chúng tôi tạo hai thiết lập dữ liệu khác nhau: 1) thiết lập tập trung, đóng vai trò là đường cơ sở đại diện cho việc học không phân tán; và 2) thiết lập dữ liệu liên kết (McMahan và cộng sự, 2017), đảm bảo phân bố đồng đều dữ liệu trên các máy khách trong FL. Sử dụng bộ dữ liệu MNIST làm ví dụ minh họa: trong thiết lập dữ liệu liên kết, mỗi máy khách trong số 10 máy khách nắm giữ ngẫu nhiên 10% của toàn bộ tập dữ liệu.

Để mô phỏng các tình huống thực tế, chúng tôi thêm các yếu tố gây nhiễu vào bộ dữ liệu của máy khách. Chúng tôi chọn $t\%$ máy khách từ tổng số K máy khách, sau đó thay thế một phần dữ liệu cục bộ nhất định của họ bằng dữ liệu nhiễu được tạo ra thông qua việc xáo trộn nhãn ngẫu nhiên. Mức độ nhiễu $t\%$ được tăng dần từ 20% lên 80% với mức tăng 20%. Sau đó, chúng tôi đánh giá hiệu năng của cả mô hình toàn cục và các quy tắc toàn cục bằng cách sử dụng bộ dữ liệu kiểm tra của máy chủ.

Đối chiếu các phương pháp cơ sở

Do không có nghiên cứu nào hiện có được thiết kế đặc biệt để tạo ra và tổng hợp các quy tắc về mô hình cục bộ FL và tận dụng các quy tắc đó để tổng hợp mô hình FL như LR-XFL, chúng tôi so sánh LR-XFL với ba phương pháp cơ sở sau trong các thí nghiệm của mình:

1. **Học tập trung:** Phương pháp này áp dụng các giải thích logic dựa trên entropy của mạng nơ-ron thần kinh (Barbiero và cộng sự, 2022) trong thiết lập môi trường tập trung với quyền truy cập trực tiếp vào tất cả dữ liệu. Nó sử dụng toán tử OR để tổng hợp các quy tắc cấp mẫu thành các quy tắc cấp lớp.
2. **Cây quyết định phân tán (DDT)** (Quinlan 1986): Trong phương pháp DDT, mỗi máy khách duy trì cây quyết định của riêng mình. Sau khi huấn luyện cục bộ, máy khách chia sẻ cây quyết định này với máy chủ. Máy chủ sau đó đánh giá các cây quyết định cục bộ đó trên một bộ dữ liệu xác thực để xác định cây tối ưu nhất. Cây cục bộ có hiệu năng cao nhất này sẽ được sử dụng làm cây quyết định toàn cục cho cả dự đoán và tạo quy tắc. Vì cây quyết định không phải là một quá trình lặp đi lặp lại, máy khách bị hạn chế chỉ được phép sử dụng cây quyết định toàn cục, không được thực hiện thêm bước tối ưu hóa nào. Cây quyết định tự nhiên tạo ra các quy tắc. Trong các thí nghiệm của chúng tôi, nếu các đặc trưng (chẳng hạn xem xét f_i) trong các quy tắc vượt quá giá trị chia tách của cây quyết định, chúng được biểu diễn dưới dạng f_i . Ngược lại, các đặc trưng f_j trong các quy tắc mà nằm dưới giá trị chia tách sẽ được biểu diễn dưới dạng $\neg f_j$.
3. **FedAvg-Logic:** Đây là một phiên bản điều chỉnh của FedAvg (McMahan và cộng sự, 2017), sử dụng mạng dựa trên entropy (Barbiero và cộng sự, 2022) làm mô hình cơ sở. Toán tử OR được sử dụng để kết nối cho việc tổng hợp

quy tắc. Các máy khách đều được gán cùng trọng số bất kể hiệu năng của các quy tắc của họ.

Tiêu chí đánh giá

Chúng tôi so sánh hiệu năng của LR-XFL và các phương pháp nền tảng bằng cách sử dụng các tiêu chí đánh giá sau:

1. **Độ chính xác phân loại:** Được tính bằng số dự đoán chính xác chia cho tổng số dự đoán. Chỉ số này đánh giá mức độ mà một phương pháp dự đoán chính xác kết quả mục tiêu.
2. **Độ chính xác quy tắc:** Độ chính xác quy tắc đo lường sự nhất quán giữa các dự đoán quy tắc và các nhãn thực tế. Giả sử r^c đại diện cho một quy tắc của lớp c , trong đó có C lớp khác nhau. Giả sử dữ liệu thực tế có P điểm dữ liệu thuộc lớp c và Q điểm dữ liệu thuộc các lớp khác. Trong số P điểm dữ liệu, p điểm thỏa mãn các điều kiện trong quy tắc r^c , trong khi trong số Q điểm dữ liệu, q điểm không thỏa mãn các điều kiện của quy tắc r^c . độ chính xác quy tắc này, ký hiệu là $RuleAcc_c$, được định nghĩa như sau:

$$RuleAcc_c = \frac{p + q}{P + Q}, \quad (6)$$

Độ chính xác quy tắc tổng thể $RuleAcc$ là trung bình của $RuleAcc_c$ trên tất cả các lớp.

3. **Độ tin cậy quy tắc:** độ tin cậy quy tắc đánh giá sự nhất quán giữa các dự đoán từ quy tắc và các dự đoán từ mô hình. Nó được tính toán bằng cách điều chỉnh công thức độ chính xác quy tắc, trong đó các số liệu thực tế P và Q được thay thế bằng các số lượng dự đoán mô hình của P cho lớp c và Q cho các lớp khác.

Kết quả của các tiêu chí đánh giá được tính toán trên tập dữ liệu kiểm tra được lưu trữ trên máy chủ. Giá trị này càng cao thì phương pháp đó càng có hiệu năng tốt.

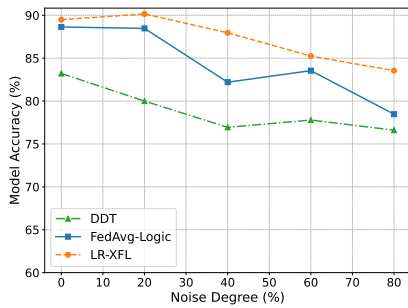
Kết quả và thảo luận

Table 1 hiển thị kết quả so sánh giữa LR-XFL và các phương pháp cơ sở.

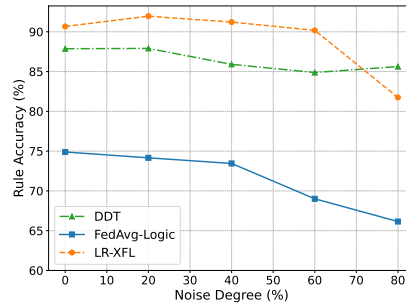
Hiệu năng mô hình Sau tất cả các thiết lập thí nghiệm, LR-XFL đạt được độ chính xác kiểm tra cao nhất trong số các phương pháp học liên kết. So với Học tập trung, phương pháp đóng vai trò là một điểm tham chiếu cho các thiết lập không liên kết với quyền truy cập trực tiếp vào dữ liệu thô, LR-XFL thường đạt hiệu năng tương đương hoặc thậm chí vượt trội hơn. Điều này có thể được giải thích bởi hiệu quả của phương pháp lựa chọn bộ kết nối logic tự động trong LR-XFL khi áp dụng ở phía máy khách trên các quy tắc cấp mẫu cục bộ. Trong các thiết lập FL, ưu điểm của LR-XFL so với FedAvg-Logic chủ yếu đến từ cơ chế gán trọng số cho các máy khách. Cơ chế này hiệu quả trong việc xác định các máy khách có quy tắc đóng góp quan trọng cho tập quy tắc toàn cầu, sau đó tăng trọng số cho các máy khách này. Kết quả là, máy chủ toàn cục dựa nhiều hơn vào các máy khách có hiệu năng cao này. Cả LR-XFL và FedAvg-Logic thường vượt trội hơn so với DDT. Điều này có thể được quy cho khả năng tận dụng thông tin từ nhiều máy khách FL trong giai đoạn huấn luyện. Trong khi đó, DDT chọn mô hình cục bộ

Bảng 1: Kết quả thí nghiệm. Chúng tôi đánh dấu hiệu năng tốt nhất bằng chữ in đậm. ‘-’ biểu thị rằng chỉ số đánh giá được đưa ra không được áp dụng cho phương pháp này.

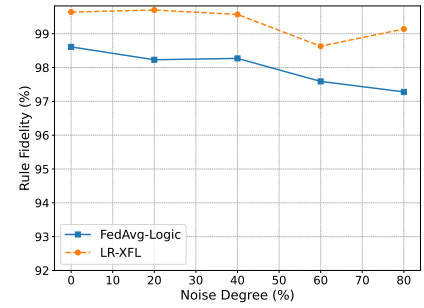
		Học tập trung	Cây quyết định phân tán (DDT)	FedAvg-Logic	LR-XFL
MNIST	độ chính xác mô hình	99.84%	99.78%	99.80%	99.96%
	độ chính xác quy tắc	99.84%	99.71%	99.84%	99.95%
	độ tin cậy quy tắc	99.95%	-	99.89%	99.97%
CUB	độ chính xác mô hình	82.20%	83.22%	88.64%	89.49%
	độ chính xác quy tắc	91.71%	87.87%	74.89%	90.67%
	độ tin cậy quy tắc	99.71%	-	98.61%	99.64%
V-Dem	độ chính xác mô hình	93.32%	92.55%	90.95%	93.08%
	độ chính xác quy tắc	90.84%	92.52%	86.71%	93.08%
	độ tin cậy quy tắc	94.59%	-	81.11%	100.00%



(a) Độ chính xác mô hình



(b) Độ chính xác quy tắc



(c) Độ tin cậy quy tắc

Hình 3: Kết quả thí nghiệm dưới các thiết lập mức độ nhiễu khác nhau..

có độ chính xác quy tắc tốt nhất trên tập dữ liệu kiểm định, mà không tích hợp các mô hình cục bộ từ các máy khách FL khác. Trung bình, LR-XFL đạt được độ chính xác mô hình cao hơn FedAvg-Logic 1.19% và DDT 3.58%.

Hiệu năng tạo quy tắc Độ chính xác quy tắc và độ tin cậy quy tắc là những chỉ số quan trọng để đánh giá tính dễ hiểu và độ tin cậy của một mô hình. Dưới các thiết lập FL, LR-XFL luôn đạt được hiệu năng cao hơn hoặc tương đương so với FedAvg-Logic và DDT trong các khía cạnh này. Có sự khác biệt lớn về độ chính xác quy tắc giữa LR-XFL và FedAvg-Logic đã được quan sát thấy trên các tập dữ liệu CUB và V-Dem. Điều này nhấn mạnh lợi thế của phương pháp lựa chọn bộ kết nối logic tự động của LR-XFL. Không giống như FedAvg-Logic, tổng hợp các quy tắc cục bộ với kết nối logic OR, LR-XFL xác định kết nối logic phù hợp nhất cho bất kỳ tình huống nào. Ngoài ra, lợi thế hiệu năng của LR-XFL so với FedAvg-Logic cũng đến từ cơ chế lựa chọn quy tắc và cơ chế gán trọng số máy khách trong quá trình tổng hợp mô hình FL. DDT, do cấu trúc vốn có không cho phép tổng hợp quy tắc, đôi khi sẽ vượt trội hơn FedAvg-Logic bằng cách tránh các vấn đề tiềm ẩn của việc mở rộng quy tắc không phù hợp. Trung bình, LR-XFL đạt được độ chính xác quy tắc cao hơn FedAvg-Logic 5,81% và DDT 0,27%.

Độ tin cậy quy tắc cao của LR-XFL nhấn mạnh sự nhất quán giữa các quy tắc và các dự đoán mô hình FL. Độ tin cậy quy tắc không áp dụng được cho DDT do đây vốn dĩ là một mô hình dựa trên quy tắc. Chỉ số này, đánh giá mức độ phù

hợp giữa dự đoán dựa trên quy tắc và dự đoán mô hình, luôn đạt 100% đối với các mô hình mà các quy tắc là đại diện trực tiếp của các dự đoán mô hình. Trung bình, LR-XFL đạt được độ tin cậy quy tắc cao hơn 5,41% so với FedAvg-Logic.

Kháng nhiễu Chúng tôi cũng tiến hành các thí nghiệm để nghiên cứu khả năng của DDT, FedAvg-Logic và LR-XFL chống lại nhiễu trong dữ liệu cục bộ của máy khách FL, sử dụng bộ dữ liệu CUB làm tiêu chuẩn. Hình 3 minh họa kết quả thí nghiệm dưới các thiết lập mức độ nhiễu khác nhau. Có thể quan sát thấy rằng khi mức độ nhiễu tăng lên, cả độ chính xác mô hình và độ chính xác quy tắc đều giảm đối với tất cả các phương pháp. Tuy nhiên, độ tin cậy quy tắc vẫn tương đối ổn định, cho thấy sự liên kết giữa các quy tắc và các dự đoán mô hình vẫn được duy trì. LR-XFL liên tục đạt được độ chính xác mô hình cao nhất dưới các mức độ nhiễu khác nhau. Hơn nữa, mức độ suy giảm hiệu năng của nó là vừa phải so với FedAvg-Logic và DDT khi mức độ nhiễu tăng lên. Đáng chú ý, độ chính xác quy tắc của LR-XFL vẫn tương đối ổn định cho đến mức độ nhiễu 60%. Điều này có thể được giải thích bởi cơ chế chọn lọc quy tắc và cơ chế gán trọng số cho máy khách của nó, giúp bảo vệ mô hình FL toàn cục khỏi việc tích hợp thông tin từ các máy khách bị nhiễu. Thụ vị là, mô hình DDT đạt được hiệu năng tốt nhất khi mức độ nhiễu đạt 80%. Điều này có thể được giải thích bởi chiến lược chỉ sử dụng mô hình máy khách tốt nhất làm cây quyết định toàn cục, giảm thiểu tối đa việc tiếp xúc với nhiễu và cái giá phải trả là hạn chế việc chia sẻ kiến thức giữa các máy

Bảng 2: Kết quả nghiên cứu loại bỏ.

	LR-XFL (loại bỏ)	LR-XFL
Độ chính xác mô hình	87.96%	89.49%
Độ chính xác quy tắc	76.29%	90.67%
Độ tin cậy quy tắc	98.83%	99.64%

khách FL.

Nghiên cứu loại bỏ

Chúng tôi thực hiện các nghiên cứu loại bỏ để đánh giá tác động của phương pháp lựa chọn bộ kết nối logic tự động lên hiệu năng của LR-XFL. Trong các thí nghiệm, các giá trị ngưỡng cho tính chéo và tính riêng biệt được đặt lần lượt là 0.9 và 0.8, dựa trên việc tinh chỉnh siêu tham số. Vì kết nối logic được chọn bởi LR-XFL cho bộ dữ liệu CUB là AND, chúng tôi đã tạo ra một phiên bản bị loại bỏ của LR-XFL bằng cách sử dụng OR làm kết nối logic cho các quy tắc trên bộ dữ liệu CUB. Kết quả được trình bày trong Bảng 2. Có thể thấy rằng phiên bản bị loại bỏ của LR-XFL có độ chính xác mô hình và độ tin cậy quy tắc thấp hơn, và có sự sụt giảm đáng kể về độ chính xác quy tắc so với phiên bản đầy đủ của LR-XFL. Kết quả này cho thấy việc sử dụng OR để kết nối các quy tắc có thể làm giảm độ chính xác quy tắc toàn cục khi các quy tắc có thể chứa thông tin bổ sung cho nhau. Điều này nhấn mạnh tầm quan trọng của phương pháp tự động lựa chọn bộ kết nối logic mà chúng tôi đã đề xuất.

Kết luận

Trong bài báo này, chúng tôi đã đề xuất phương pháp LR-XFL, framework học liên kết có thể được giải thích bằng các quy tắc logic, và cũng là framework đầu tiên có khả năng này. Nó có thể suy ra các quy tắc toàn cục chính xác từ các quy tắc cục bộ mà không yêu cầu quyền truy cập vào dữ liệu cục bộ của máy khách. Bộ kết nối logic phù hợp nhất để tổng hợp các quy tắc của máy khách được xác định tự động bởi LR-XFL dựa trên các đặc điểm dữ liệu cục bộ của máy khách. Thiết kế mới này cải thiện đáng kể độ tin cậy của mô hình kết quả. Hơn nữa, các quy tắc được tổng hợp đóng một vai trò quan trọng trong việc xác định trọng số của máy khách trong quá trình tổng hợp mô hình FL. Thiết kế minh bạch này cho phép các chuyên gia trong lĩnh vực tham gia tích cực vào việc xác thực, tinh chỉnh và điều chỉnh các quy tắc, từ đó giúp cải thiện mô hình kết quả FL.

Lời cảm ơn

Nghiên cứu/dự án này được hỗ trợ một phần bởi Quỹ Nghiên cứu Quốc gia Singapore và Phòng thí nghiệm Quốc gia DSO thông qua Chương trình AI Singapore (Giải thưởng AISG Số AISG2-RP-2020-019); Quỹ Chương trình Sản xuất và Kỹ thuật Tiên tiến (AME) RIE 2020 (Số A20G8b0102), Singapore; và Trung tâm Nghiên cứu Chung NTU-WeBank về FinTech, Đại học Công nghệ Nanyang, Singapore.

Tài liệu tham khảo

An, Z.; and Ma, M. 2023. Guiding Federated Learning with Inferenced Formal Logic Properties. In *Proceedings of the*

ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023), 274–275.

Barbiero, P.; Ciravegna, G.; Giannini, F.; Lió, P.; Gori, M.; and Melacci, S. 2022. Entropy-based logic explanations of neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 6046–6054.

Cha, N.; Du, Z.; Wu, C.; Yoshinaga, T.; Zhong, L.; Ma, J.; Liu, F.; and Ji, Y. 2022. Fuzzy logic based client selection for federated learning in vehicular networks. *IEEE Open Journal of the Computer Society*, 3: 39–50.

Chen, Z.; Bei, Y.; and Rudin, C. 2020. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12): 772–782.

Ciravegna, G.; Giannini, F.; Gori, M.; Maggini, M.; and Melacci, S. 2020. Human-Driven FOL Explanations of Deep Learning. In *IJCAI*, 2234–2240.

Coppedge, M.; Gerring, J.; Knutsen, C. H.; Lindberg, S. I.; Teorell, J.; Altman, D.; Bernhard, M.; Cornell, A.; Fish, M. S.; Gastaldi, L.; và cộng sự., 2021. V-dem codebook v11.

Gunning, D.; Stefik, M.; Choi, J.; Miller, T.; Stumpf, S.; and Yang, G.-Z. 2019. XAI—Explainable artificial intelligence. *Science robotics*, 4(37): eaay7120.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jain, R.; Ciravegna, G.; Barbiero, P.; Giannini, F.; Buffelli, D.; and Lio, P. 2022. Extending Logic Explained Networks to Text Classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 8838–8857. Association for Computational Linguistics.

Kairouz, P.; McMahan, H. B.; và cộng sự., 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends in Machine Learning*, 14(1-2): 1–210.

Kazhdan, D.; Dimanov, B.; Jamnik, M.; Lió, P.; and Weller, A. 2020. Now you see me (CME): concept-based model extraction. *arXiv preprint arXiv:2010.13233*.

Kim, B.; Gilmer, J.; Wattenberg, M.; and Viégas, F. 2018. Tcav: Relative concept importance testing with linear concept activation vectors.

Koh, P. W.; Nguyen, T.; Tang, Y. S.; Mussmann, S.; Pierson, E.; Kim, B.; and Liang, P. 2020. Concept bottleneck models. In *International conference on machine learning*, 5338–5348. PMLR.

LeCun, Y. 1998. The MNIST database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>.

Lee, S.; Wang, X.; Han, S.; Yi, X.; Xie, X.; and Cha, M. 2022. Self-explaining deep models with logic rule reasoning. *Advances in Neural Information Processing Systems*, 35: 3203–3216.

Lowerre, B.; and Reddy, R. 1976. The harpy speech recognition system: performance with large vocabularies. *The Journal of the Acoustical Society of America*, 60(S1): S10–S11.

- Lundberg, S. M.; and Lee, S.-I. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.
- McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.
- Quinlan, J. R. 1986. Induction of decision trees. *Machine learning*, 1: 81–106.
- Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144.
- Saeed, M.; Villarroel, M.; Reisner, A. T.; Clifford, G.; Lehman, L.-W.; Moody, G.; Heldt, T.; Kyaw, T. H.; Moody, B.; and Mark, R. G. 2011. Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): a public-access intensive care unit database. *Critical care medicine*, 39(5): 952.
- Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.
- Stammer, W.; Schramowski, P.; and Kersting, K. 2021. Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3619–3629.
- Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.
- Xu, F.; Uszkoreit, H.; Du, Y.; Fan, W.; Zhao, D.; and Zhu, J. 2019. Explainable AI: A brief survey on history, research areas, approaches and challenges. In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II* 8, 563–574. Springer.
- Yu, H.; Miao, C.; An, B.; Shen, Z.; and Leung, C. 2014. Reputation-aware Task Allocation for Human Trustees. volume 1.
- Zhu, X.; Wang, D.; Pedrycz, W.; and Li, Z. 2021. Horizontal federated learning of Takagi–Sugeno fuzzy rule-based models. *IEEE Transactions on Fuzzy Systems*, 30(9): 3537–3547.

Phụ lục

Mô tả bộ dữ liệu

Trong thiết lập học tập liên kết (FL), bộ dữ liệu được phân chia để tạo điều kiện cho cả hoạt động dựa trên máy chủ và dựa trên máy khách. Cụ thể, 20% dữ liệu được phân bổ cho máy chủ: 10% để kiểm định việc lựa chọn khách hàng dựa trên các quy tắc họ tạo ra, và 10% còn lại để kiểm tra các quy tắc toàn cục sau khi được tổng hợp và đánh giá độ chính xác của chúng. Trong các thí nghiệm của chúng tôi, chúng tôi thiết lập 10 máy khách FL, với mỗi máy khách nắm giữ 10% dữ liệu tổng của tất cả các máy khách. Trong phần dữ liệu của mỗi máy khách, chúng tôi chia 90% để huấn luyện, 5% để xác thực và 5% để kiểm tra.

Dựa trên cấu hình bộ dữ liệu được trình bày trong (Barbiero và cộng sự, 2022), chúng tôi đã sử dụng bốn bộ dữ liệu tiêu chuẩn:

1. MNIST(Even/Odd) (LeCun 1998),
2. CUB (Wah và cộng sự, 2011),
3. V-Dem (Coppedge và cộng sự, 2021), and
4. MIMIC-II (Saeed và cộng sự, 2011).

MNIST(Even/Odd)

là một phiên bản biến thể của bộ dữ liệu MNIST (LeCun 1998), tuân theo mô hình “hình ảnh → đặc trưng → lớp”. Bao gồm 60.000 mẫu chữ số viết tay kích thước 28x28 pixel, các đặc trưng của nó là các chữ số từ 0-9 được dự đoán bởi mô hình ResNet10 (He và cộng sự, 2016). Nhiệm vụ phân loại cuối cùng tập trung vào việc xác định tính chẵn lẻ của chữ số, tức là chữ số chẵn hay lẻ.

CUB (Caltech-UCSD Birds-200-2011)

tuân theo mô hình “ảnh → đặc trưng → lớp”. Bao gồm 11.788 hình ảnh trải rộng trên 200 phân loại chim. Mỗi hình ảnh được đánh dấu với 312 thuộc tính nhị phân mô tả các đặc điểm như màu sắc, kích thước, hình dạng và hoa văn. Dựa trên quá trình tiền xử lý từ (Koh và cộng sự, 2020), chúng tôi sử dụng 108 trong số các đặc trưng này trong nghiên cứu của chúng tôi. Các đặc trưng cho CUB được bắt nguồn từ ResNet10 (He và cộng sự, 2016) và phân loại cuối cùng liên quan đến 200 phân loại chim.

V-Dem (Các loại hình dân chủ)

dựa trên mô hình “đầu vào → lớp”. Bộ dữ liệu này bao gồm các đặc điểm tiềm ẩn về các chế độ chính trị ở nhiều mức độ chi tiết khác nhau. Nghiên cứu của chúng tôi, phù hợp với các thiết lập trong (Barbiero và cộng sự, 2022), làm việc với 3.743 điểm dữ liệu sau xử lý. Bộ dữ liệu giữ nguyên 14 đặc trưng để hiểu đối với con người, giúp hướng dẫn việc dự đoán hai loại chế độ dân chủ: bầu cử và không bầu cử.

MIMIC-II (Giám sát Thông minh Đa tham số trong Chăm sóc Đặc biệt II)

cũng sử dụng mô hình “đầu vào → lớp”, cung cấp một cái nhìn toàn diện về các hoạt động chăm sóc tích cực. Bộ dữ liệu này ghi lại các tín hiệu sinh lý, các chỉ số sống còn và dữ liệu lâm sàng chi tiết từ các bệnh nhân trong phòng chăm sóc đặc biệt ICU (Intensive care unit). Với quá trình tiền xử lý dựa trên (Barbiero và cộng sự, 2022), phân tích của chúng tôi sử dụng 1.776 điểm dữ liệu, mỗi điểm bao gồm 90 đặc trưng. Nhiệm vụ phân loại là để dự đoán kết quả hồi phục của bệnh nhân sau khi chuyển vào ICU.

Các quy tắc được tạo bởi $LR-XFL$

Bảng 3 hiển thị quy tắc toàn cục và độ chính xác của nó trên bộ dữ liệu kiểm tra của máy chủ cho lớp Swainson Warbler, được xác định trong vòng đào tạo FL cuối cùng. Bảng cũng trình bày quy tắc của mỗi máy khách và độ chính xác tương ứng của họ trên tập dữ liệu kiểm tra này. Đáng chú ý, các quy tắc từ máy khách 2, 4, 8 và 9 đã được tích hợp để xây dựng quy tắc toàn cục. Bằng cách tổng hợp các quy tắc cục bộ của máy khách bằng toán tử logic AND, quy tắc toàn cục đạt được độ chính xác 100,00%, vượt qua độ chính xác của bất kỳ quy tắc cục bộ cá nhân nào.

Bảng 4 trình bày quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc để phân biệt giữa số chẵn và số lẻ trong bộ dữ liệu MNIST(Even/Odd). Mặc dù toán tử logic cho tập dữ liệu này là OR do $LR-XFL$ quyết định, nhưng điều này không rõ ràng trong các quy tắc. Sự bỏ sót này phát sinh vì một quy tắc cấp mẫu cụ thể đạt được độ chính xác tối ưu. Do đó, trong quá trình tổng hợp quy tắc cấp lớp trong máy khách và tổng hợp quy tắc toàn cục giữa các máy khách, việc thêm các quy tắc bổ sung với quy tắc cấp mẫu tối ưu này không nâng cao độ chính xác. Cuối cùng, quy tắc cấp mẫu này trở thành quy tắc toàn cục.

Bảng 7 hiển thị quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc cho 200 phân loại chim trong bộ dữ liệu CUB. Toán tử logic toàn cục trong bộ dữ liệu CUB là AND.

Bảng 5 trình bày các quy tắc toàn cục, độ chính xác và độ tin cậy của chúng liên quan đến sự hiện diện hoặc vắng mặt của dân chủ bầu cử trong bộ dữ liệu V-Dem. Toán tử logic được sử dụng trong bộ dữ liệu V-Dem là AND.

Bảng 6 phác thảo các quy tắc toàn cục, độ chính xác quy tắc và độ tin cậy quy tắc cho trạng thái hồi phục của bệnh nhân trong bộ dữ liệu MIMIC-II. Bộ dữ liệu MIMIC-II sử dụng AND làm toán tử logic chính.

Bảng 3: Quy tắc toàn cục và cục bộ cho Swainson Warbler trong CUB. Các khách hàng đóng góp vào quy tắc toàn cục được tô đậm. Dấu ‘-’ trong cột quy tắc biểu thị rằng khách hàng tương ứng không tạo ra quy tắc cho lớp này.

	Quy tắc	Độ chính xác
Toàn cục	has_bill_shape-all-purpose \wedge has_tail_shape-notched_tail \wedge has_crown_color-brown \wedge \neghas_throat_color-white \wedge \neghas_bill_length-shorter_than_head	100.00%
Máy khách 1	has_under_tail_color-brown \wedge \neg has_bill_length-shorter_than_head	55.24%
Máy khách 2	has_tail_shape-notched_tail \wedge \neghas_bill_length-shorter_than_head	66.50%
Máy khách 3	has_bill_shape-all-purpose \wedge has_back_color-brown \wedge \neg has_underparts_color-brown	52.81%
Máy khách 4	has_tail_shape-notched_tail \wedge \neghas_bill_length-shorter_than_head	66.50%
Máy khách 5	has_tail_shape-notched_tail \wedge \neg has_bill_shape-cone \wedge \neg has_upperparts_color-grey \wedge \neg has_under_tail_color-black \wedge \neg has_primary_color-yellow	54.53%
Máy khách 6	has_bill_shape-all-purpose \wedge has_belly_pattern-solid \wedge \neg has_upperparts_color-grey \wedge \neg has_upperparts_color-black \wedge \neg has_primary_color-yellow \wedge \neg has_leg_color-black	51.42%
Máy khách 8	has_bill_shape-all-purpose \wedge has_crown_color-brown \wedge \neghas_throat_color-white	59.66%
Máy khách 9	has_tail_shape-notched_tail \wedge \neghas_bill_length-shorter_than_head	66.50%
Máy khách 10	-	0.00%

Bảng 4: Các quy tắc toàn cục cho 2 lớp trong bộ dữ liệu MNIST (Chẵn/Lẻ), cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Odd $\leftrightarrow \neg 0 \wedge \neg 2 \wedge \neg 4 \wedge \neg 6 \wedge \neg 8$	99.95%	99.97%
Even $\leftrightarrow \neg 1 \wedge \neg 3 \wedge \neg 5 \wedge \neg 7 \wedge \neg 9$	99.97%	99.98%

Bảng 5: Các quy tắc toàn cục cho 2 lớp trong bộ dữ liệu VDEM, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Electoral_democracy $\leftrightarrow v2x_freexp_altinf \wedge v2xel_frefair \wedge v2x_elecoff$	93.08%	100.00%
Non_electoral_democracy $\leftrightarrow \neg v2xel_frefair$	93.08%	100.00%

Bảng 6: Các quy tắc toàn cục cho 2 lớp trong bộ dữ liệu MIMIC-II, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Recover $\leftrightarrow \neg stroke_flg \wedge \neg mal_flg \wedge \neg age_HIGH$	78.33%	76.40%
Non_recover $\leftrightarrow resp_flg \wedge age_HIGH \wedge hour_icu_intime_LOW \wedge \neg age_LOW \wedge \neg chloride_first_LOW \wedge \neg map_1st_HIGH \wedge \neg po2_first_HIGH \wedge \neg sapsi_first_LOW$	51.98%	82.02%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Black_footed_Albatross $\leftrightarrow has_bill_shape-hooked_seabird \wedge \neg has_underparts_color-white$	73.33%	97.97%
Laysan_Albatross $\leftrightarrow has_bill_shape-hooked_seabird \wedge \neg has_size-medium_9_16_in \wedge \neg has_crown_color-black$	94.40%	99.83%
Sooty_Albatross $\leftrightarrow has_upper_tail_color-grey \wedge has_back_pattern-solid \wedge has_bill_color-black \wedge \neg has_size-small_5_9_in \wedge \neg has_primary_color-grey$	100.00%	100.00%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Groove_billed_Ani \leftrightarrow has_head_pattern-plain \wedge has_bill_length-shorter_than_head \wedge \neg has_bill_shape-all-purpose \wedge \neg has_bill_shape-cone \wedge \neg has_underparts_color-white \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_nape_color-white \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_shape-perching-like	95.41%	98.98%
Crested_Auklet \leftrightarrow has_underparts_color-grey \wedge has_breast_color-grey \wedge \neg has_eye_color-black \wedge \neg has_shape-perching-like	100.00%	99.83%
Least_Auklet \leftrightarrow has_breast_pattern-striped \wedge \neg has_shape-perching-like	100.00%	100.00%
Parakeet_Auklet \leftrightarrow has_primary_color-white \wedge has_leg_color-grey \wedge \neg has_eye_color-black \wedge \neg has_shape-perching-like \wedge \neg has_bill_color-black	100.00%	99.83%
Rhinoceros_Auklet \leftrightarrow has_leg_color-buff \wedge \neg has_shape-perching-like	100.00%	100.00%
Brewer_Blackbird \leftrightarrow has_bill_shape-all-purpose \wedge has_wing_shape-rounded-wings \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_shape-perching-like \wedge \neg has_crown_color-blue	94.40%	99.15%
Red_winged_Blackbird \leftrightarrow has_wing_pattern-multi-color \wedge \neg has_bill_shape-all-purpose \wedge \neg has_bill_shape-cone \wedge \neg has_bill_length-about_the_same_as_head	69.61%	98.47%
Rusty_Blackbird \leftrightarrow has_bill_shape-all-purpose \wedge has_back_color-brown \wedge \neg has_throat_color-white \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_back_pattern-striped	91.58%	99.66%
Yellow_headed_Blackbird \leftrightarrow has_nape_color-yellow \wedge has_belly_color-black \wedge has_wing_shape-rounded-wings \wedge \neg has_bill_shape-all-purpose \wedge \neg has_nape_color-black	100.00%	100.00%
Bobolink \leftrightarrow has_back_pattern-multi-colored \wedge \neg has_bill_shape-all-purpose \wedge \neg has_underparts_color-yellow \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_under_tail_color-white \wedge \neg has_bill_color-grey	100.00%	100.00%
Indigo_Bunting \leftrightarrow has_head_pattern-plain \wedge has_crown_color-blue \wedge \neg has_bill_shape-all-purpose	89.91%	98.81%
Lazuli_Bunting \leftrightarrow has_upperparts_color-black \wedge has_wing_shape-rounded-wings \wedge has_tail_pattern-multi-colored \wedge \neg has_bill_shape-all-purpose \wedge \neg has_breast_pattern-solid \wedge \neg has_forehead_color-black	100.00%	100.00%
Painted_Bunting \leftrightarrow has_nape_color-blue \wedge has_back_pattern-multi-colored \wedge \neg has_bill_color-black	100.00%	99.49%
Cardinal \leftrightarrow has_forehead_color-red \wedge \neg has_tail_shape-notched_tail	49.62%	98.47%
Spotted_Catbird \leftrightarrow has_leg_color-grey \wedge \neg has_wing_color-brown \wedge \neg has_underparts_color-white \wedge \neg has_breast_pattern-solid \wedge \neg has_nape_color-grey \wedge \neg has_belly_color-white \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_tail_pattern-solid \wedge \neg has_belly_pattern-solid \wedge \neg has_bill_color-grey \wedge \neg has_bill_color-black \wedge \neg has_wing_pattern-striped	100.00%	99.83%
Gray_Catbird \leftrightarrow has_forehead_color-grey \wedge has_under_tail_color-grey \wedge has_wing_pattern-solid \wedge \neg has_bill_shape-cone \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_belly_color-white \wedge \neg has_wing_pattern-multi-color	100.00%	100.00%
Yellow_breasted_Chats \leftrightarrow has_breast_color-yellow \wedge has_leg_color-grey \wedge has_bill_color-black \wedge \neg has_belly_pattern-solid	92.81%	100.00%
Eastern_Towhee \leftrightarrow has_head_pattern-plain \wedge has_breast_color-black \wedge has_wing_pattern-multi-color \wedge \neg has_upperparts_color-white \wedge \neg has_breast_pattern-solid \wedge \neg has_tail_pattern-multi-colored	83.29%	100.00%
Chuck_will_Widow \leftrightarrow has_belly_color-buff \wedge \neg has_underparts_color-buff	88.33%	99.66%
Brandt_Cormorant \leftrightarrow has_bill_shape-hooked_seabird \wedge has_head_pattern-plain \wedge has_forehead_color-black \wedge \neg has_bill_shape-all-purpose \wedge \neg has_underparts_color-white \wedge \neg has_nape_color-white	83.21%	98.98%
Red_faced_Cormorant \leftrightarrow has_belly_color-black \wedge has_wing_shape-rounded-wings \wedge \neg has_head_pattern-plain \wedge \neg has_shape-perching-like	83.29%	99.83%
Pelagic_Cormorant \leftrightarrow has_head_pattern-plain \wedge has_size-medium_9_-_16_in \wedge \neg has_bill_shape-hooked_seabird \wedge \neg has_wing_color-grey \wedge \neg has_wing_color-white \wedge \neg has_nape_color-white \wedge \neg has_size-small_5_-_9_in	74.57%	99.15%
Bronzed_Cowbird \leftrightarrow has_head_pattern-plain \wedge \neg has_bill_shape-all-purpose \wedge \neg has_bill_length-about_the_same_as_head	51.42%	94.41%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Shiny_Cowbird \leftrightarrow has_bill_shape-all-purpose \wedge has_wing_pattern-solid \wedge \neg has_underparts_color-white \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_belly_color-buff \wedge \neg has_shape-perching-like	88.91%	97.63%
Brown_Creeper \leftrightarrow has_nape_color-buff \wedge has_tail_pattern-solid	100.00%	100.00%
American_Crow \leftrightarrow has_shape-perching-like \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_size-small_5_-_9_in	74.74%	98.64%
Fish_Crow \leftrightarrow has_wing_shape-rounded-wings \wedge has_size-small_5_-_9_in \wedge \neg has_bill_shape-hooked_seabird \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_belly_color-white \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_size-small_5_-_9_in \wedge \neg has_shape-perching-like	49.92%	100.00%
Black_billed_Cuckoo \leftrightarrow has_leg_color-grey \wedge has_crown_color-brown \wedge \neg has_wing_color-black \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_wing_shape-rounded-wings	92.81%	99.49%
Mangrove_Cuckoo \leftrightarrow has_back_color-grey \wedge has_belly_color-buff	100.00%	100.00%
Yellow_billed_Cuckoo \leftrightarrow has_back_color-brown \wedge has_upper_tail_color-brown \wedge has_leg_color-grey \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_wing_shape-rounded-wings	100.00%	99.32%
Gray_crowned_Rosy_Finch \leftrightarrow has_back_pattern-striped \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_belly_pattern-solid \wedge \neg has_bill_color-black	100.00%	100.00%
Purple_Finch \leftrightarrow has_underparts_color-white \wedge has_tail_shape-notched_tail \wedge has_forehead_color-red \wedge \neg has_throat_color-white \wedge \neg has_tail_pattern-solid \wedge \neg has_belly_pattern-solid \wedge \neg has_bill_color-black	49.96%	100.00%
Northern_Flicker \leftrightarrow has_primary_color-buff \wedge \neg has_back_color-brown \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Acadian_Flycatcher \leftrightarrow has_wing_pattern-striped \wedge \neg has_upperparts_color-brown \wedge \neg has_bill_color-black	78.44%	99.83%
Great_Crested_Flycatcher \leftrightarrow has_under_tail_color-brown \wedge \neg has_back_color-brown \wedge \neg has_bill_length-shorter_than_head	100.00%	100.00%
Least_Flycatcher \leftrightarrow has_back_color-grey \wedge has_wing_shape-rounded-wings \wedge \neg has_bill_shape-cone \wedge \neg has_wing_color-yellow \wedge \neg has_wing_color-black \wedge \neg has_throat_color-white \wedge \neg has_forehead_color-brown \wedge \neg has_belly_color-yellow \wedge \neg has_size-very_small_3_-_5_in \wedge \neg has_tail_pattern-multi-colored \wedge \neg has_wing_pattern-solid \wedge \neg has_wing_pattern-multi-color	100.00%	100.00%
Olive_sided_Flycatcher \leftrightarrow has_bill_shape-all-purpose \wedge has_underparts_color-grey \wedge \neg has_belly_pattern-solid	100.00%	99.83%
Scissor_tailed_Flycatcher \leftrightarrow has_tail_pattern-multi-colored \wedge has_crown_color-white \wedge \neg has_back_color-white \wedge \neg has_shape-perching-like	100.00%	100.00%
Vermilion_Flycatcher \leftrightarrow has_forehead_color-red \wedge \neg has_bill_shape-cone \wedge \neg has_primary_color-black	100.00%	100.00%
Yellow_bellied_Flycatcher \leftrightarrow has_tail_shape-notched_tail \wedge has_leg_color-black \wedge \neg has_underparts_color-white \wedge \neg has_throat_color-black \wedge \neg has_under_tail_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_bill_color-black	92.81%	100.00%
Frigatebird \leftrightarrow has_bill_shape-hooked_seabird \wedge has_underparts_color-black \wedge has_head_pattern-plain \wedge has_forehead_color-black \wedge \neg has_throat_color-black	100.00%	99.83%
Northern_Fulmar \leftrightarrow has_under_tail_color-grey \wedge \neg has_bill_shape-all-purpose \wedge \neg has_bill_color-black	74.83%	99.49%
Gadwall \leftrightarrow has_wing_color-buff \wedge has_forehead_color-brown \wedge \neg has_upper_tail_color-brown	100.00%	100.00%
American_Goldfinch \leftrightarrow has_upper_tail_color-black \wedge has_upper_tail_color-white \wedge has_tail_pattern-multi-colored \wedge has_crown_color-yellow \wedge \neg has_bill_shape-all-purpose	100.00%	100.00%
European_Goldfinch \leftrightarrow has_breast_pattern-multi-colored \wedge has_under_tail_color-white \wedge has_nape_color-white	49.87%	99.49%
Boat_tailed_Grackle \leftrightarrow has_bill_shape-all-purpose \wedge has_head_pattern-plain \wedge has_bill_length-about_the_same_as_head \wedge has_wing_shape-rounded-wings \wedge \neg has_belly_color-yellow \wedge \neg has_size-small_5_-_9_in \wedge \neg has_leg_color-grey	100.00%	97.12%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Eared_Grebe \leftrightarrow has_wing_color-black \wedge has_belly_color-grey \wedge \neg has_wing_color-grey \wedge \neg has_underparts_color-white \wedge \neg has_upper_tail_color-black \wedge \neg has_breast_color-black \wedge \neg has_throat_color-black \wedge \neg has_eye_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_shape-perching-like \wedge \neg has_bill_color-black	83.29%	100.00%
Horned_Grebe \leftrightarrow has_primary_color-black \wedge \neg has_underparts_color-white \wedge \neg has_breast_pattern-solid	74.74%	98.98%
Pied_billed_Grebe \leftrightarrow has_upper_tail_color-brown \wedge has_back_pattern-solid \wedge \neg has_size-small_5_-_9_in \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Western_Grebe \leftrightarrow has_wing_color-grey \wedge has_belly_pattern-solid \wedge \neg has_eye_color-black	100.00%	100.00%
Blue_Grosbeak \leftrightarrow has_wing_color-black \wedge has_under_tail_color-black \wedge \neg has_upperparts_color-black \wedge \neg has_underparts_color-white \wedge \neg has_upper_tail_color-black	92.81%	99.32%
Evening_Grosbeak \leftrightarrow has_nape_color-brown \wedge \neg has_back_color-buff \wedge \neg has_primary_color-brown \wedge \neg has_crown_color-brown	100.00%	100.00%
Pine_Grosbeak \leftrightarrow has_underparts_color-grey \wedge has_tail_shape-notched_tail \wedge has_under_tail_color-grey \wedge \neg has_underparts_color-white \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Rose_breasted_Grosbeak \leftrightarrow has_head_pattern-plain \wedge \neg has_belly_pattern-solid	66.50%	99.32%
Pigeon_Guillemot \leftrightarrow has_upperparts_color-white \wedge has_bill_color-black \wedge has_wing_pattern-multi-color \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_size-small_5_-_9_in \wedge \neg has_shape-perching-like	100.00%	100.00%
California_Gull \leftrightarrow has_head_pattern-plain \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_tail_pattern-solid	67.17%	97.63%
Glaucous_winged_Gull \leftrightarrow has_upper_tail_color-grey \wedge has_head_pattern-plain \wedge has_under_tail_color-grey \wedge has_back_pattern-solid \wedge \neg has_upperparts_color-black \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_tail_pattern-solid \wedge \neg has_bill_color-black	89.96%	99.66%
Heermann_Gull \leftrightarrow has_breast_color-grey \wedge has_crown_color-grey \wedge \neg has_size-small_5_-_9_in	100.00%	100.00%
Herring_Gull \leftrightarrow has_wing_color-white \wedge has_back_color-grey \wedge \neg has_bill_shape-dagger \wedge \neg has_upper_tail_color-black \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_under_tail_color-grey \wedge \neg has_tail_pattern-solid	65.00%	97.63%
Ivory_Gull \leftrightarrow has_head_pattern-plain \wedge has_tail_pattern-solid \wedge has_leg_color-black \wedge \neg has_upper_tail_color-black \wedge \neg has_under_tail_color-black \wedge \neg has_size-small_5_-_9_in	100.00%	100.00%
Ring_billed_Gull \leftrightarrow has_upper_tail_color-grey \wedge \neg has_under_tail_color-grey \wedge \neg has_under_tail_color-black \wedge \neg has_size-small_5_-_9_in \wedge \neg has_tail_pattern-solid \wedge \neg has_wing_pattern-multi-color	72.01%	99.15%
Slaty_backed_Gull \leftrightarrow has_bill_shape-hooked_seabird \wedge has_back_color-grey \wedge \neg has_upper_tail_color-grey \wedge \neg has_head_pattern-plain \wedge \neg has_bill_color-black	95.41%	99.83%
Western_Gull \leftrightarrow has_head_pattern-plain \wedge has_back_pattern-multi-colored \wedge \neg has_back_color-black \wedge \neg has_upper_tail_color-grey \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_back_pattern-solid	100.00%	100.00%
Anna_Hummingbird \leftrightarrow has_size-very_small_3_-_5_in \wedge \neg has_breast_color-white	57.09%	96.44%
Ruby_throated_Hummingbird \leftrightarrow has_size-very_small_3_-_5_in \wedge \neg has_bill_shape-all-purpose	62.86%	97.80%
Rufous_Hummingbird \leftrightarrow has_underparts_color-white \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_shape-perching-like \wedge \neg has_belly_pattern-solid	83.25%	99.66%
Green_Violetear \leftrightarrow has_nape_color-blue \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Long_tailed_Jaeger \leftrightarrow has_wing_color-grey \wedge has_nape_color-white \wedge \neg has_wing_color-black \wedge \neg has_upperparts_color-grey \wedge \neg has_size-small_5_-_9_in \wedge \neg has_primary_color-white	100.00%	100.00%
Pomarine_Jaeger \leftrightarrow has_belly_color-white \wedge \neg has_breast_pattern-solid \wedge \neg has_size-small_5_-_9_in \wedge \neg has_shape-perching-like \wedge \neg has_tail_pattern-solid \wedge \neg has_primary_color-white	100.00%	100.00%
Blue_Jay \leftrightarrow has_breast_color-grey \wedge has_under_tail_color-black \wedge \neg has_wing_color-brown \wedge \neg has_breast_pattern-solid \wedge \neg has_back_color-black	89.96%	100.00%
Florida_Jay \leftrightarrow has_breast_pattern-multi-colored \wedge has_back_pattern-multi-colored \wedge has_tail_pattern-solid \wedge \neg has_leg_color-grey	92.81%	100.00%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Green_Jay \leftrightarrow has_breast_color-black \wedge \neg has_wing_color-black \wedge \neg has_upperparts_color-brown \wedge \neg has_underparts_color-white \wedge \neg has_back_color-black \wedge \neg has_bill_length-about_the_same_as_head	100.00%	99.83%
Dark_eyed_Junco \leftrightarrow has_underparts_color-white \wedge has_throat_color-grey \wedge \neg has_wing_color-black \wedge \neg has_underparts_color-yellow \wedge \neg has_tail_shape-notched_tail \wedge \neg has_breast_color-white	83.25%	99.83%
Tropical_Kingbird \leftrightarrow has_head_pattern-plain \wedge has_breast_color-yellow \wedge \neg has_wing_color-black \wedge \neg has_under_tail_color-grey \wedge \neg has_leg_color-grey	100.00%	100.00%
Gray_Kingbird \leftrightarrow has_forehead_color-grey \wedge has_under_tail_color-grey \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_back_pattern-solid	100.00%	100.00%
Belted_Kingfisher \leftrightarrow has_breast_pattern-multi-colored \wedge has_wing_shape-rounded-wings \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_shape-perching-like	74.83%	99.49%
Green_Kingfisher \leftrightarrow has_size-small_5_-_9_in \wedge has_primary_color-black \wedge \neg has_head_pattern-plain \wedge \neg has_nape_color-black \wedge \neg has_wing_shape-rounded-wings	57.87%	96.95%
Pied_Kingfisher \leftrightarrow has_bill_shape-dagger \wedge \neg has_head_pattern-capped \wedge \neg has_back_pattern-solid	74.83%	99.49%
Ringed_Kingfisher \leftrightarrow has_wing_color-grey \wedge has_primary_color-grey \wedge \neg has_bill_shape-cone \wedge \neg has_upperparts_color-grey \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_forehead_color-white \wedge \neg has_nape_color-grey	100.00%	100.00%
White_breasted_Kingfisher \leftrightarrow has_throat_color-white \wedge \neg has_wing_color-brown \wedge \neg has_belly_color-white \wedge \neg has_belly_pattern-solid	92.81%	100.00%
Red_legged_Kittiwake \leftrightarrow has_upperparts_color-grey \wedge has_back_pattern-solid \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_size-small_5_-_9_in \wedge \neg has_tail_pattern-solid \wedge \neg has_bill_color-black	100.00%	99.83%
Horned_Lark \leftrightarrow has_throat_color-yellow \wedge has_under_tail_color-buff \wedge \neg has_breast_color-black \wedge \neg has_throat_color-white \wedge \neg has_nape_color-grey \wedge \neg has_leg_color-buff	100.00%	100.00%
Pacific_Loon \leftrightarrow has_back_color-black \wedge has_primary_color-black \wedge \neg has_upper_tail_color-black \wedge \neg has_belly_pattern-solid \wedge \neg has_crown_color-black	94.40%	100.00%
Mallard \leftrightarrow has_underparts_color-brown \wedge has_upper_tail_color-black \wedge \neg has_underparts_color-white \wedge \neg has_back_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_leg_color-black	100.00%	100.00%
Western_Meadowlark \leftrightarrow has_primary_color-yellow \wedge \neg has_shape-perching-like	100.00%	99.66%
Hooded_Merganser \leftrightarrow has_nape_color-black \wedge \neg has_eye_color-black \wedge \neg has_belly_pattern-solid	83.25%	99.66%
Red_breasted_Merganser \leftrightarrow has_upperparts_color-white \wedge \neg has_eye_color-black	100.00%	100.00%
Mockingbird \leftrightarrow has_upper_tail_color-grey \wedge has_shape-perching-like \wedge \neg has_wing_color-grey \wedge \neg has_wing_pattern-multi-color	83.29%	99.83%
Nighthawk \leftrightarrow has_wing_color-white \wedge has_throat_color-white \wedge \neg has_wing_color-black \wedge \neg has_underparts_color-white	49.92%	100.00%
Clark_Nutcracker \leftrightarrow has_throat_color-grey \wedge has_under_tail_color-white \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_shape-perching-like	100.00%	100.00%
White_breasted_Nuthatch \leftrightarrow has_shape-perching-like \wedge has_back_pattern-multi-colored \wedge \neg has_bill_length-shorter_than_head	100.00%	100.00%
Baltimore_Oriole \leftrightarrow has_back_pattern-multi-colored \wedge has_tail_pattern-multi-colored \wedge has_belly_pattern-solid \wedge has_bill_color-grey \wedge \neg has_bill_shape-cone \wedge \neg has_bill_color-black	80.56%	99.15%
Hooded_Oriole \leftrightarrow has_back_color-yellow \wedge has_breast_color-black \wedge \neg has_underparts_color-black \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_wing_shape-rounded-wings	96.62%	100.00%
Orchard_Oriole \leftrightarrow has_under_tail_color-black \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_belly_color-white \wedge \neg has_crown_color-yellow \wedge \neg has_wing_pattern-solid	56.74%	96.44%
Scott_Oriole \leftrightarrow has_under_tail_color-yellow \wedge has_tail_pattern-multi-colored \wedge \neg has_bill_length-shorter_than_head	100.00%	100.00%
Ovenbird \leftrightarrow has_underparts_color-black \wedge has_wing_pattern-solid \wedge \neg has_wing_color-brown \wedge \neg has_breast_pattern-striped \wedge \neg has_back_color-black \wedge \neg has_belly_pattern-solid	89.96%	100.00%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Brown_Pelican \leftrightarrow has_wing_pattern-solid \wedge \neg has_throat_color-white \wedge \neg has_size-small_5_-_9_in \wedge \neg has_size-medium_9_-_16_in	79.11%	98.81%
White_Pelican \leftrightarrow has_tail_pattern-solid \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_bill_length-shorter_than_head \wedge \neg has_crown_color-black	100.00%	99.83%
Western_Wood_Pewee \leftrightarrow has_breast_color-grey \wedge has_belly_color-grey \wedge \neg has_bill_shape-cone \wedge \neg has_upper_tail_color-black \wedge \neg has_crown_color-black \wedge \neg has_wing_pattern-multi-color	74.91%	100.00%
Sayornis \leftrightarrow has_upper_tail_color-brown \wedge \neg has_back_color-brown \wedge \neg has_bill_length-about_the_same_as_head	83.29%	99.83%
American_Pipit \leftrightarrow has_bill_shape-all-purpose \wedge has_belly_color-buff \wedge \neg has_underparts_color-brown \wedge \neg has_breast_pattern-solid \wedge \neg has_back_color-brown \wedge \neg has_forehead_color-blue \wedge \neg has_nape_color-grey \wedge \neg has_primary_color-yellow \wedge \neg has_bill_color-black	100.00%	100.00%
Whip_poor_Will \leftrightarrow has_underparts_color-brown \wedge \neg has_back_color-brown \wedge \neg has_under_tail_color-brown	58.71%	98.98%
Horned_Puffin \leftrightarrow has_throat_color-black \wedge has_primary_color-white \wedge \neg has_throat_color-white \wedge \neg has_shape-perching-like	89.96%	99.83%
Common_Raven \leftrightarrow has_wing_shape-rounded-wings \wedge \neg has_underparts_color-white \wedge \neg has_size-small_5_-_9_in \wedge \neg has_size-very_small_3_-_5_in	72.21%	97.12%
White_necked_Raven \leftrightarrow has_breast_color-black \wedge has_nape_color-white \wedge has_wing_pattern-solid \wedge \neg has_throat_color-white \wedge \neg has_nape_color-black	100.00%	99.15%
American_Redstart \leftrightarrow has_breast_pattern-multi-colored \wedge has_tail_shape-notched_tail \wedge has_tail_pattern-multi-colored \wedge \neg has_wing_shape-rounded-wings	100.00%	100.00%
Geococcyx \leftrightarrow has_nape_color-brown \wedge has_nape_color-black \wedge has_leg_color-grey	100.00%	100.00%
Loggerhead_Shrike \leftrightarrow has_upperparts_color-white \wedge has_nape_color-grey \wedge has_tail_pattern-multi-colored	94.40%	100.00%
Great_Grey_Shrike \leftrightarrow has_forehead_color-grey \wedge has_under_tail_color-black \wedge \neg has_upper_tail_color-black \wedge \neg has_size-small_5_-_9_in \wedge \neg has_tail_pattern-solid	100.00%	99.83%
Baird_Sparrow \leftrightarrow has_under_tail_color-buff \wedge \neg has_back_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_bill_color-black	49.96%	100.00%
Black_throated_Sparrow \leftrightarrow has_throat_color-black \wedge has_bill_length-shorter_than_head \wedge has_crown_color-grey \wedge \neg has_bill_shape-all-purpose \wedge \neg has_underparts_color-yellow \wedge \neg has_tail_shape-notched_tail \wedge \neg has_forehead_color-black	100.00%	100.00%
Brewer_Sparrow \leftrightarrow has_forehead_color-brown \wedge has_belly_color-grey	100.00%	100.00%
Chipping_Sparrow \leftrightarrow has_nape_color-grey \wedge has_crown_color-brown \wedge \neg has_upperparts_color-black	100.00%	100.00%
Clay_colored_Sparrow \leftrightarrow has_upperparts_color-black \wedge has_breast_color-buff \wedge has_forehead_color-brown \wedge has_belly_color-white \wedge \neg has_size-very_small_3_-_5_in	100.00%	100.00%
House_Sparrow \leftrightarrow has_breast_pattern-solid \wedge has_breast_color-grey \wedge has_primary_color-brown \wedge \neg has_primary_color-grey	92.81%	100.00%
Field_Sparrow \leftrightarrow has_size-very_small_3_-_5_in \wedge has_belly_pattern-solid \wedge \neg has_bill_shape-all-purpose \wedge \neg has_bill_color-black	95.41%	100.00%
Fox_Sparrow \leftrightarrow has_breast_pattern-striped \wedge has_throat_color-white \wedge \neg has_bill_shape-all-purpose \wedge \neg has_wing_color-white \wedge \neg has_upperparts_color-buff \wedge \neg has_size-medium_9_-_16_in \wedge \neg has_back_pattern-solid \wedge \neg has_tail_pattern-solid \wedge \neg has_leg_color-grey \wedge \neg has_wing_pattern-striped	100.00%	99.83%
Grasshopper_Sparrow \leftrightarrow has_wing_color-black \wedge has_breast_pattern-solid \wedge has_back_color-buff \wedge has_leg_color-buff \wedge \neg has_forehead_color-brown	100.00%	100.00%
Harris_Sparrow \leftrightarrow has_wing_color-black \wedge has_back_color-buff \wedge has_under_tail_color-brown \wedge has_primary_color-white \wedge \neg has_primary_color-buff	100.00%	100.00%
Henslow_Sparrow \leftrightarrow has_bill_length-shorter_than_head \wedge has_nape_color-brown \wedge has_primary_color-buff \wedge \neg has_breast_pattern-solid \wedge \neg has_tail_shape-notched_tail \wedge \neg has_throat_color-yellow \wedge \neg has_primary_color-brown	100.00%	99.66%
Le_Conte_Sparrow \leftrightarrow has_under_tail_color-buff \wedge has_crown_color-black	100.00%	99.66%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Lincoln_Sparrow \leftrightarrow has_breast_pattern-striped \wedge has_breast_color-black \wedge has_breast_color-buff \wedge \neg has_breast_color-white \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_belly_pattern-solid	83.29%	100.00%
Nelson_Sharp_tailed_Sparrow \leftrightarrow has_underparts_color-buff \wedge has_forehead_color-black \wedge \neg has_back_color-black \wedge \neg has_size-small_5_-_9_in \wedge \neg has_crown_color-black	100.00%	99.83%
Savannah_Sparrow \leftrightarrow has_breast_pattern-striped \wedge has_throat_color-white \wedge \neg has_breast_pattern-striped \wedge \neg has_breast_color-black \wedge \neg has_nape_color-buff \wedge \neg has_primary_color-white \wedge \neg has_bill_color-black \wedge \neg has_wing_pattern-solid	49.92%	100.00%
Seaside_Sparrow \leftrightarrow has_back_color-grey \wedge \neg has_wing_color-grey \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Song_Sparrow \leftrightarrow has_under_tail_color-buff \wedge has_nape_color-buff \wedge \neg has_breast_color-buff \wedge \neg has_belly_pattern-solid \wedge \neg has_primary_color-buff	100.00%	100.00%
Tree_Sparrow \leftrightarrow has_wing_color-black \wedge has_wing_color-white \wedge has_forehead_color-brown \wedge \neg has_forehead_color-black \wedge \neg has_under_tail_color-brown \wedge \neg has_under_tail_color-black	83.29%	99.49%
Vesper_Sparrow \leftrightarrow has_breast_color-white \wedge has_nape_color-brown \wedge \neg has_breast_pattern-solid \wedge \neg has_breast_pattern-striped \wedge \neg has_back_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_bill_color-black	100.00%	100.00%
White_crowned_Sparrow \leftrightarrow has_breast_pattern-solid \wedge has_throat_color-grey \wedge has_forehead_color-white \wedge \neg has_back_color-grey \wedge \neg has_throat_color-white	100.00%	100.00%
White_throated_Sparrow \leftrightarrow has_throat_color-white \wedge has_forehead_color-yellow \wedge has_nape_color-grey \wedge \neg has_bill_shape-all-purpose \wedge \neg has_primary_color-grey	92.81%	100.00%
Cape_Glossy_Starling \leftrightarrow has_crown_color-blue \wedge \neg has_shape-perching-like	83.25%	99.66%
Bank_Swallow \leftrightarrow has_bill_shape-cone \wedge has_back_color-brown \wedge has_belly_pattern-solid \wedge has_bill_color-black \wedge \neg has_shape-perching-like \wedge \neg has_leg_color-black	94.40%	99.83%
Barn_Swallow \leftrightarrow has_underparts_color-buff \wedge has_bill_color-black \wedge \neg has_breast_color-buff \wedge \neg has_shape-perching-like	100.00%	100.00%
Cliff_Swallow \leftrightarrow has_back_color-black \wedge has_crown_color-black \wedge \neg has_wing_color-black \wedge \neg has_head_pattern-capped \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_size-medium_9_-_16_in \wedge \neg has_shape-perching-like \wedge \neg has_leg_color-black	100.00%	99.49%
Tree_Swallow \leftrightarrow has_tail_shape-notched_tail \wedge has_nape_color-blue \wedge \neg has_bill_shape-all-purpose \wedge \neg has_wing_color-black \wedge \neg has_bill_color-grey	49.79%	99.15%
Scarlet_Tanager \leftrightarrow has_head_pattern-plain \wedge has_under_tail_color-black \wedge \neg has_bill_shape-all-purpose \wedge \neg has_upper_tail_color-black \wedge \neg has_bill_length-about_the_same_as_head \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_crown_color-black	100.00%	100.00%
Summer_Tanager \leftrightarrow has_tail_pattern-solid \wedge has_crown_color-yellow \wedge \neg has_bill_shape-all-purpose \wedge \neg has_underparts_color-white \wedge \neg has_tail_shape-notched_tail \wedge \neg has_under_tail_color-black \wedge \neg has_leg_color-buff \wedge \neg has_bill_color-black	49.83%	99.32%
Arctic_Tern \leftrightarrow has_head_pattern-capped \wedge has_eye_color-black \wedge has_nape_color-black \wedge \neg has_size-small_5_-_9_in \wedge \neg has_leg_color-black \wedge \neg has_bill_color-black	94.40%	99.15%
Black_Tern \leftrightarrow has_under_tail_color-grey \wedge has_crown_color-black \wedge \neg has_upperparts_color-black \wedge \neg has_size-small_5_-_9_in	100.00%	100.00%
Caspian_Tern \leftrightarrow has_bill_shape-dagger \wedge has_wing_color-white \wedge has_tail_pattern-solid \wedge \neg has_wing_color-grey \wedge \neg has_nape_color-black \wedge \neg has_crown_color-white	69.87%	99.49%
Common_Tern \leftrightarrow has_upper_tail_color-grey \wedge \neg has_size-small_5_-_9_in \wedge \neg has_primary_color-grey \wedge \neg has_bill_color-black	94.40%	99.83%
Elegant_Tern \leftrightarrow has_bill_shape-dagger \wedge \neg has_forehead_color-black \wedge \neg has_size-small_5_-_9_in \wedge \neg has_bill_color-black	92.81%	99.83%
Forsters_Tern \leftrightarrow has_nape_color-black \wedge has_bill_color-black \wedge \neg has_wing_color-black \wedge \neg has_back_color-grey \wedge \neg has_primary_color-black	83.21%	98.81%
Least_Tern \leftrightarrow has_bill_shape-dagger \wedge has_head_pattern-capped \wedge \neg has_upperparts_color-white \wedge \neg has_nape_color-grey \wedge \neg has_bill_color-black	83.29%	99.83%
Green_tailed_Towhee \leftrightarrow has_belly_color-grey \wedge \neg has_wing_color-brown \wedge \neg has_wing_color-black \wedge \neg has_back_pattern-solid \wedge \neg has_leg_color-black \wedge \neg has_bill_color-black	100.00%	99.83%
Brown_Thrasher \leftrightarrow has_nape_color-brown \wedge \neg has_eye_color-black	89.96%	100.00%
Sage_Thrasher \leftrightarrow has_wing_pattern-striped \wedge \neg has_eye_color-black	89.96%	100.00%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Black_capped_Vireo ↔ has_back_pattern-multi-colored ∧ ¬has_wing_color-grey ∧ ¬has_primary_color-black ∧ ¬has_leg_color-black	83.21%	99.49%
Blue_headed_Vireo ↔ has_under_tail_color-grey ∧ has_primary_color-grey ∧ has_bill_color-black ∧ ¬has_upper_tail_color-grey ∧ ¬has_bill_length-about_the_same_as_head ∧ ¬has_leg_color-black	100.00%	100.00%
Philadelphia_Vireo ↔ has_wing_color-grey ∧ has_crown_color-grey ∧ ¬has_bill_length-about_the_same_as_head ∧ ¬has_belly_color-white ∧ ¬has_size-small_5_-_9_in ∧ ¬has_bill_color-black	83.29%	100.00%
Red_eyed_Vireo ↔ has_breast_pattern-solid ∧ has_bill_length-shorter_than_head ∧ has_forehead_color-grey ∧ has_belly_color-white ∧ ¬has_wing_color-yellow ∧ ¬has_bill_length-about_the_same_as_head ∧ ¬has_bill_color-black	100.00%	99.83%
Warbling_Vireo ↔ has_upperparts_color-grey ∧ has_primary_color-buff	89.91%	99.83%
White_eyed_Vireo ↔ has_forehead_color-yellow ∧ ¬has_wing_color-grey ∧ ¬has_upperparts_color-yellow ∧ ¬has_breast_color-yellow ∧ ¬has_wing_pattern-solid	86.24%	99.49%
Yellow_throated_Vireo ↔ has_back_pattern-multi-colored ∧ has_crown_color-yellow ∧ ¬has_belly_pattern-solid	89.96%	100.00%
Bay_breasted_Warbler ↔ has_wing_color-black ∧ has_breast_pattern-multi-colored ∧ has_wing_shape-rounded-wings ∧ ¬has_under_tail_color-black	100.00%	100.00%
Black_and_white_Warbler ↔ has_forehead_color-white ∧ ¬has_belly_pattern-solid	100.00%	100.00%
Black_throated_Blue_Warbler ↔ has_primary_color-black ∧ has_primary_color-white ∧ ¬has_bill_shape-cone ∧ ¬has_upperparts_color-grey ∧ ¬has_bill_length-about_the_same_as_head ∧ ¬has_size-small_5_-_9_in ∧ ¬has_size-very_small_3_-_5_in ∧ ¬has_tail_pattern-solid	100.00%	100.00%
Blue_winged_Warbler ↔ has_back_color-grey ∧ has_bill_length-shorter_than_head ∧ ¬has_underparts_color-white ∧ ¬has_nape_color-grey ∧ ¬has_nape_color-white ∧ ¬has_belly_color-grey	100.00%	100.00%
Canada_Warbler ↔ has_under_tail_color-grey ∧ ¬has_breast_pattern-solid ∧ ¬has_bill_color-black	100.00%	100.00%
Cape_May_Warbler ↔ has_breast_pattern-striped ∧ has_back_pattern-striped ∧ ¬has_wing_color-brown ∧ ¬has_breast_color-white ∧ ¬has_belly_pattern-solid	100.00%	100.00%
Cerulean_Warbler ↔ has_wing_color-white ∧ has_primary_color-white ∧ ¬has_breast_pattern-solid ∧ ¬has_nape_color-black ∧ ¬has_wing_shape-rounded-wings ∧ ¬has_bill_color-black	100.00%	100.00%
Chestnut_sided_Warbler ↔ has_crown_color-yellow ∧ ¬has_upperparts_color-grey ∧ ¬has_wing_shape-rounded-wings ∧ ¬has_size-small_5_-_9_in	94.40%	100.00%
Golden_winged_Warbler ↔ has_throat_color-black ∧ has_crown_color-yellow ∧ ¬has_underparts_color-yellow	100.00%	100.00%
Hooded_Warbler ↔ has_throat_color-black ∧ has_nape_color-black ∧ ¬has_wing_color-black ∧ ¬has_breast_pattern-solid ∧ ¬has_back_color-black ∧ ¬has_nape_color-grey	100.00%	100.00%
Kentucky_Warbler ↔ has_breast_color-yellow ∧ has_leg_color-buff ∧ ¬has_wing_color-black ∧ ¬has_throat_color-grey ∧ ¬has_forehead_color-yellow ∧ ¬has_wing_shape-rounded-wings ∧ ¬has_back_pattern-solid	95.41%	100.00%
Magnolia_Warbler ↔ has_wing_color-grey ∧ has_wing_color-white ∧ ¬has_belly_pattern-solid	100.00%	100.00%
Mourning_Warbler ↔ has_under_tail_color-yellow ∧ ¬has_breast_pattern-solid ∧ ¬has_bill_color-black	100.00%	99.83%
Myrtle_Warbler ↔ has_upperparts_color-black ∧ has_wing_pattern-striped ∧ ¬has_upperparts_color-brown ∧ ¬has_underparts_color-white	100.00%	100.00%
Nashville_Warbler ↔ has_under_tail_color-grey ∧ has_back_pattern-multi-colored ∧ ¬has_tail_shape-notched_tail ∧ ¬has_upper_tail_color-grey ∧ ¬has_forehead_color-yellow ∧ ¬has_forehead_color-black ∧ ¬has_nape_color-blue ∧ ¬has_back_pattern-solid ∧ ¬has_primary_color-black	95.41%	99.83%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Orange_crowned_Warbler \leftrightarrow has_bill_shape-all-purpose \wedge has_breast_pattern-solid \wedge has_tail_pattern-multi-colored \wedge has_belly_pattern-solid \wedge \neg has_upperparts_color-black \wedge \neg has_throat_color-yellow \wedge \neg has_throat_color-white \wedge \neg has_bill_color-black \wedge \neg has_wing_pattern-solid \wedge \neg has_wing_pattern-multi-color	95.41%	98.64%
Palm_Warbler \leftrightarrow has_throat_color-yellow \wedge \neg has_underparts_color-yellow \wedge \neg has_breast_pattern-solid \wedge \neg has_back_color-buff \wedge \neg has_bill_color-black	100.00%	99.49%
Pine_Warbler \leftrightarrow has_wing_color-white \wedge has_upper_tail_color-grey \wedge \neg has_wing_color-brown \wedge \neg has_back_color-grey \wedge \neg has_belly_color-white \wedge \neg has_bill_color-black \wedge \neg has_wing_pattern-solid	92.81%	99.83%
Prairie_Warbler \leftrightarrow has_tail_pattern-multi-colored \wedge has_crown_color-yellow \wedge \neg has_wing_color-black \wedge \neg has_breast_pattern-solid \wedge \neg has_size-small_5_-_9_in	89.91%	100.00%
Prothonotary_Warbler \leftrightarrow has_head_pattern-plain \wedge has_leg_color-grey \wedge has_bill_color-black \wedge \neg has_wing_color-black \wedge \neg has_underparts_color-white \wedge \neg has_back_pattern-solid \wedge \neg has_wing_pattern-solid	100.00%	100.00%
Swainson_Warbler \leftrightarrow has_bill_shape-all-purpose \wedge has_tail_shape-notched_tail \wedge has_crown_color-brown \wedge \neg has_throat_color-white \wedge \neg has_bill_length-shorter_than_head	100.00%	100.00%
Tennessee_Warbler \leftrightarrow has_back_color-yellow \wedge \neg has_breast_color-yellow	92.81%	100.00%
Wilson_Warbler \leftrightarrow has_head_pattern-capped \wedge \neg has_upperparts_color-grey \wedge \neg has_underparts_color-white	100.00%	100.00%
Worm_eating_Warbler \leftrightarrow has_throat_color-yellow \wedge has_forehead_color-black \wedge \neg has_primary_color-yellow	94.40%	99.83%
Yellow_Warbler \leftrightarrow has_forehead_color-yellow \wedge \neg has_breast_pattern-solid \wedge \neg has_belly_pattern-solid	83.29%	100.00%
Northern_Waterthrush \leftrightarrow has_breast_pattern-striped \wedge \neg has_bill_shape-cone \wedge \neg has_primary_color-black	74.83%	99.32%
Louisiana_Waterthrush \leftrightarrow has_breast_pattern-striped \wedge has_tail_pattern-solid \wedge \neg has_tail_shape-notched_tail \wedge \neg has_bill_color-black	100.00%	99.83%
Bohemian_Waxwing \leftrightarrow has_throat_color-black \wedge has_under_tail_color-yellow \wedge has_tail_pattern-multi-colored \wedge \neg has_bill_shape-all-purpose \wedge \neg has_back_color-black \wedge \neg has_tail_pattern-solid	100.00%	100.00%
Cedar_Waxwing \leftrightarrow has_under_tail_color-grey \wedge has_belly_color-buff \wedge \neg has_upperparts_color-grey	100.00%	100.00%
American_Three_toed_Woodpecker \leftrightarrow has_back_color-white \wedge \neg has_underparts_color-black \wedge \neg has_belly_pattern-solid	100.00%	100.00%
Pileated_Woodpecker \leftrightarrow has_tail_shape-notched_tail \wedge has_forehead_color-red \wedge \neg has_bill_length-shorter_than_head	100.00%	100.00%
Red_bellied_Woodpecker \leftrightarrow has_forehead_color-red \wedge has_wing_pattern-striped \wedge \neg has_tail_pattern-solid	100.00%	100.00%
Red_cockaded_Woodpecker \leftrightarrow has_head_pattern-capped \wedge has_belly_color-black	92.81%	100.00%
Red_headed_Woodpecker \leftrightarrow has_bill_shape-dagger \wedge \neg has_throat_color-white	100.00%	100.00%
Downy_Woodpecker \leftrightarrow has_bill_shape-all-purpose \wedge has_nape_color-white \wedge has_back_pattern-multi-colored	100.00%	99.83%
Bewick_Wren \leftrightarrow has_under_tail_color-black \wedge has_shape-perching-like \wedge has_back_pattern-solid \wedge \neg has_bill_shape-cone \wedge \neg has_bill_color-black	83.29%	100.00%
Cactus_Wren \leftrightarrow has_underparts_color-black \wedge has_under_tail_color-white \wedge has_nape_color-white \wedge \neg has_upperparts_color-black \wedge \neg has_belly_pattern-solid \wedge \neg has_bill_color-black	96.11%	100.00%
Carolina_Wren \leftrightarrow has_size-very_small_3_-_5_in \wedge has_bill_color-grey	83.29%	99.83%
House_Wren \leftrightarrow has_breast_color-buff \wedge \neg has_forehead_color-black \wedge \neg has_wing_shape-rounded-wings \wedge \neg has_size-small_5_-_9_in \wedge \neg has_size-medium_9_-_16_in \wedge \neg has_size-very_small_3_-_5_in	89.96%	99.66%

Bảng 7: Các quy tắc toàn cục cho 200 lớp trong bộ dữ liệu CUB, cùng với độ chính xác và độ tin cậy tương ứng trên tập kiểm tra toàn cục.

Quy tắc	Độ chính xác	Độ tin cậy
Marsh_Wren \leftrightarrow has_wing_color-black \wedge has_nape_color-brown \wedge has_belly_pattern-solid \wedge \neg has_bill_shape-cone \wedge \neg has_upperparts_color-grey \wedge \neg has_forehead_color-grey \wedge \neg has_under_tail_color-black \wedge \neg has_size-very_small_3_-_5_in \wedge \neg has_back_pattern-solid \wedge \neg has_primary_color-white \wedge \neg has_wing_pattern-striped \wedge \neg has_wing_pattern-multi-color	100.00%	100.00%
Rock_Wren \leftrightarrow has_under_tail_color-buff \wedge \neg has_upperparts_color-buff	100.00%	100.00%
Winter_Wren \leftrightarrow has_underparts_color-buff \wedge has_breast_color-brown \wedge \neg has_upperparts_color-black \wedge \neg has_breast_pattern-solid \wedge \neg has_throat_color-white	94.40%	100.00%
Common_Yellowthroat \leftrightarrow has_forehead_color-black \wedge has_under_tail_color-yellow \wedge \neg has_crown_color-black	100.00%	100.00%