

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Phan Trường An - Phạm Dương Trường Đức

HỌC CÓ GIÁM SÁT VỚI DỮ LIỆU CÓ
PHÂN BỐ THAY ĐỔI BẰNG MÔ HÌNH
DỰA TRÊN QUAN HỆ NHÂN QUẢ

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN
CHƯƠNG TRÌNH CHUẨN

Tp. Hồ Chí Minh, tháng 07/2024

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Phan Trường An - 20120032
Phạm Dương Trường Đức - 20120061

HỌC CÓ GIÁM SÁT VỚI DỮ LIỆU CÓ
PHÂN BỐ THAY ĐỔI BẰNG MÔ HÌNH
DỰA TRÊN QUAN HỆ NHÂN QUẢ

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN
CHƯƠNG TRÌNH CHUẨN

GIẢNG VIÊN HƯỚNG DẪN

ThS. Trần Trung Kiên

TS. Nguyễn Ngọc Thảo

Tp. Hồ Chí Minh, tháng 07/2024

Lời cam đoan

Chúng tôi xin cam đoan đây là công trình nghiên cứu của chúng tôi dưới sự hướng dẫn của thầy Trần Trung Kiên và cô Nguyễn Ngọc Thảo.

Chúng tôi cam kết tất cả nội dung của đề tài khóa luận này là kết quả của quá trình học tập, nghiên cứu của chúng tôi.

Chúng tôi cam kết tất cả tài liệu, thông tin tham khảo từ tác giả khác được trích dẫn đầy đủ và chính xác.

Tp. Hồ Chí Minh, tháng 7 năm 2024

Phan Trường An

Phạm Dương Trường Đức

Nhận xét hướng dẫn

Nhận xét phản biện

Lời cảm ơn

Chúng em xin chân thành cảm ơn thầy Trần Trung Kiên và cô Nguyễn Ngọc Thảo vì đã tận tình hướng dẫn chúng em trong suốt quá trình thực hiện khóa luận tốt nghiệp. Chúng em xin cảm ơn thầy cô vì đã dẫn dắt chúng em. Những kiến thức, lời khuyên và những chỉnh sửa, góp ý của thầy cô là những bài học quý báu mà chúng em sẽ ghi nhớ.

Chúng em xin cảm ơn quý thầy cô Khoa Công nghệ Thông tin nói riêng, quý thầy cô của Trường Đại học Khoa học Tự nhiên - ĐHQG TPHCM nói chung vì đã trang bị cho chúng em một nền tảng kiến thức vững chắc và cho chúng em một môi trường học tập lành mạnh, bổ ích.

Xin gửi lời cảm ơn chân thành đến gia đình, bạn bè, người thân vì đã luôn ủng hộ, động viên cũng như lắng nghe những chia sẻ của chúng tôi.

Tp. Hồ Chí Minh, tháng 7 năm 2024

Phan Trường An

Phạm Dương Trường Đức



fit@hcmus

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

ĐỀ CƯƠNG KHOÁ LUẬN TỐT NGHIỆP
**HỌC CÓ GIÁM SÁT VỚI DỮ LIỆU CÓ PHÂN
BỐ THAY ĐỔI BẰNG MÔ HÌNH DỰA TRÊN
QUAN HỆ NHÂN QUẢ**

*(Supervised learning with data distribution shift using
causality-based model)*

1 THÔNG TIN CHUNG

Người hướng dẫn:

- ThS. Trần Trung Kiên
- TS. Nguyễn Ngọc Thảo (Khoa Công nghệ Thông tin)

Nhóm sinh viên thực hiện:

1. Phan Trường An (MSSV: 20120032)
2. Phạm Dương Trường Đức (MSSV: 20120061)

Loại đề tài: Nghiên cứu

Thời gian thực hiện: Từ 01/2024 đến 06/2024

2 NỘI DUNG THỰC HIỆN

2.1 Giới thiệu về đề tài

Học có giám sát là một loại của học máy mà máy tính sẽ được huấn luyện trên một bộ dữ liệu gồm các cặp đầu vào và đầu ra đúng. Máy tính sẽ học trên bộ dữ liệu đó và đưa ra dự đoán cho dữ liệu đầu vào mới. Ví dụ, để dự đoán giá của một ngôi nhà mới, một mô hình học máy sẽ được huấn luyện trên bộ dữ liệu bao gồm thông tin (vị trí, diện tích, số phòng, số tầng,...) của các ngôi nhà được bán trước đó đại diện cho đầu vào và giá của chúng đại diện cho đầu ra. Sau quá trình huấn luyện, khi đưa thông tin của ngôi nhà mới vào mô hình, mô hình sẽ dự đoán được giá của ngôi nhà đó.

Dữ liệu có phân bố thay đổi (Data distribution shift) là một hiện tượng trong học có giám sát xảy ra khi dữ liệu mà mô hình phải dự đoán có sự thay đổi phân bố so với dữ liệu mà mô hình được huấn luyện; điều này khiến cho độ chính xác của kết quả dự đoán của mô hình giảm.

Bài toán “học có giám sát với dữ liệu có phân bố thay đổi” mà chúng em tập trung làm trong khoá luận là bài toán học có giám sát với dữ liệu có phân bố đầu vào thay đổi. Cụ thể, bài toán này được phát biểu như sau:

- Cho dữ liệu của K miền huấn luyện, trong đó dữ liệu của mỗi miền có dạng $\{(x_i, y_i)_{i=1}^n\}$ với $x \in \mathcal{X}$ là đầu vào và $y \in \mathcal{Y}$ là đầu ra tương ứng, (x, y) được phát sinh từ phân bố $P(X, Y) = P(X)P(Y | X)$. Các miền này có cùng phân bố $P(Y | X)$ nhưng có phân bố $P(X)$ khác nhau. Với mỗi cặp (x, y) thì có thể có thêm thuộc tính a mà giá trị sẽ đặc trưng cho miền tương ứng và ảnh hưởng đến giá trị của x (ví dụ, với dữ liệu ảnh và các miền ứng với các người chụp khác nhau thì ta có thể có các thuộc tính a như góc chụp, giờ chụp,...).
- Yêu cầu của bài toán là tìm một hàm dự đoán $h : \mathcal{X} \rightarrow \mathcal{Y}$ từ dữ liệu của K miền huấn luyện được cho để có thể dự đoán tốt với dữ liệu của miền mục

tiêu, với miền mục tiêu là một miền mới có cùng $P(Y | X)$ nhưng khác $P(X)$ với các miền huấn luyện. Nghĩa là hàm dự đoán h có thể khái quát và chống chịu tốt với sự thay đổi miền.

Các bộ dữ liệu khác nhau có thể có các loại phân bố thay đổi khác nhau, hoặc thậm chí với cùng một bộ dữ liệu có thể xảy ra các loại phân bố thay đổi khác nhau trên các thuộc tính khác nhau. Đây là một thách thức cho các phương pháp giải quyết bài toán. Các phương pháp chỉ tập trung giải quyết một loại phân bố thay đổi sẽ không hoạt động tốt với dữ liệu có các loại phân bố thay đổi khác.

Nếu giải quyết được bài toán này, ta có thể giúp các mô hình học máy khái quát tốt được các miền, từ đó giúp mô hình chống chịu tốt với sự thay đổi phân bố dữ liệu, nhờ vậy các mô hình sẽ không bị giảm hiệu suất và đưa ra dự đoán đủ tốt theo yêu cầu của người dùng. Việc chống chịu tốt với sự thay đổi miền còn giúp mở rộng ứng dụng của các mô hình học máy trong thực tế.

Qua quá trình tìm hiểu, chúng em nhận thấy hướng tiếp cận sử dụng mô hình dựa trên quan hệ nhân quả là một hướng tiếp cận tiềm năng. Vì vậy, chúng em sẽ tập trung tìm hiểu sâu hướng tiếp cận này trong khóa luận.

2.2 Mục tiêu đề tài

- Tìm hiểu bài toán “học có giám sát với dữ liệu có phân bố thay đổi” (tình hình nghiên cứu, các phương pháp tiếp cận/giải quyết). Từ đó, chọn ra một phương pháp tiềm năng để hiểu sâu.
- Cài đặt thành công phương pháp được đề xuất trong bài báo mà chúng em tham khảo và đạt được kết quả như bài báo đó. Tiến hành cài đặt các phương pháp khác để so sánh.
- Tìm hiểu, thí nghiệm để hiểu thêm về ưu/nhược điểm của phương pháp đã chọn. Sau đó, xem xét và cải tiến nếu có thể.
- Rèn luyện kỹ năng nghiên cứu, mở rộng kiến thức liên quan đến bài toán.

- Nâng cao khả năng tự học, làm việc nhóm, kỹ năng trình bày,...

2.3 Phạm vi của đề tài

Trong đề tài này, chúng em sẽ tìm hiểu và cài đặt lại phương pháp từ một bài báo uy tín. Trong quá trình cài đặt, nếu có thời gian và khả năng, chúng em sẽ tiến hành cải tiến phương pháp. Theo chúng em thì việc hiểu rõ cơ sở lý thuyết và cài đặt được phương pháp sẽ cần nhiều thời gian, nên chúng em xem đây là mục tiêu chính cần tập trung thực hiện. Khi đã hiểu rõ và cài đặt thành công phương pháp của bài báo, chúng em sẽ tiến hành các thử nghiệm để tìm ra ưu/khuyết điểm của phương pháp đó.

Trong đề tài này, chúng em sẽ thử nghiệm phương pháp với bài toán phân loại. Đề tài chủ yếu làm về dữ liệu ảnh. Chúng em dự định sẽ thực hiện với 3 bộ dữ liệu là MNIST, small NORB và Waterbirds.

2.4 Cách tiếp cận dự kiến

Dưới đây là một số phương pháp giải quyết bài toán mà chúng em đã tìm hiểu cho đến thời điểm hiện tại.

- Một phương pháp thường được sử dụng là Correlation Alignment (CORAL) được đề xuất bởi B. Sun và cộng sự trong bài báo “Return of Frustratingly Easy Domain Adaptation” [1]. Phương pháp này điều chỉnh hiệp phương sai của dữ liệu từ miền huấn luyện theo dữ liệu của miền mục tiêu. Từ đó, dữ liệu từ miền huấn luyện sẽ có phân bố tương đồng với dữ liệu từ miền mục tiêu. Nhờ vậy mà mô hình huấn luyện được sẽ khai quát hóa tốt hơn. Phương pháp này đòi hỏi phải có được dữ liệu từ miền mục tiêu.
- Các phương pháp tăng cường dữ liệu là một trong những hướng được sử dụng phổ biến trong việc khai quát miền. Với ý tưởng là dữ liệu càng lớn thì mô hình sẽ càng có nhiều sự đa dạng dữ liệu để khai quát các miền. Mixup [2] do Zhang và cộng sự đề xuất là một phương pháp sinh dữ liệu thường được

sử dụng để tăng cường dữ liệu trong bài toán khái quát miền. Mixup sinh dữ liệu mới bằng cách thực hiện nội suy tuyến tính giữa hai đầu vào bất kỳ và đầu ra tương ứng của chúng trong bộ dữ liệu với một trọng số được lấy mẫu từ phân bố Beta. Đối với phương pháp Mixup nói riêng và các phương pháp tăng cường dữ liệu nói chung, đôi khi sự tăng cường dữ liệu sinh ra những đặc trưng không cần thiết trong việc khái quát miền, khiến cho tốn tài nguyên mà không giải quyết được yêu cầu bài toán.

- Một số các chiến lược học cũng có thể giải quyết được yêu cầu khái quát miền như là Ensemble learning, Meta-learning, Gradient operation,... [3] Trong đó meta-learning là một trong các chiến lược được sử dụng và nghiên cứu phổ biến. MLDG [4] (meta-learning for domain generalization) sử dụng chiến lược này trong việc khái quát miền. Ý tưởng của MLDG là chia dữ liệu trong các miền huấn luyện thành các bộ dữ liệu meta-train và meta-test để mô phỏng sự thay đổi miền, từ đó học được biểu diễn của các miền và khái quát chúng. Điểm yếu của phương pháp này là nếu dữ liệu trong các miền huấn luyện không đa dạng thì khó có thể khái quát tốt được trên miền mục tiêu và có thể dẫn đến hiện tượng overfitting trên các đặc điểm cụ thể của miền huấn luyện.
- Trong bài toán khái quát miền, các phương pháp dựa trên mối quan hệ nhân quả là hướng tiếp cận tiềm năng. Các mô hình học máy thông thường có thể dựa vào các biến gây nhiễu trong dữ liệu huấn luyện để đưa ra quyết định khiến cho dự đoán không chính xác với dữ liệu mới của miền mục tiêu (ví dụ trong bài toán nhận diện con vật, các hình ảnh về bò trong dữ liệu huấn luyện thường xuất hiện cùng với nền cỏ; nên nếu ta đưa một hình ảnh con bò ở bãi biển thì mô hình có thể dự đoán không chính xác). Các mô hình học dựa trên mối quan hệ nhân quả có thể chỉ ra những nguyên nhân thật sự dẫn đến đầu ra đúng, chứ không bị ảnh hưởng bởi các biến gây nhiễu trong dữ liệu huấn luyện. Nhờ đó, các mô hình này có thể dự đoán tốt với dữ liệu mới của

miền mục tiêu. Năm 2022, Kaur và cộng sự đã công bố bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” ở hội nghị International Conference on Learning Representations [5]. Mô hình được đề xuất đã rút trích ra các ràng buộc chính xác từ đồ thị nhân quả để tìm ra tất cả các nguyên nhân dẫn đến đầu ra. Từ đó, mô hình có thể khái quát tốt được các miền và đưa ra dự đoán chính xác nhất. Ngoài ra, mô hình của Kaur và cộng sự [5] còn có thể khái quát tốt được sự thay đổi đa thuộc tính trên các miền.

Trong khóa luận này, chúng em quyết định tập trung tìm hiểu và cài đặt lại phương pháp theo thuật toán được đề xuất trong bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” [5] ở hội nghị International Conference on Learning Representations vào năm 2022. Phương pháp này có độ chính xác cao hơn các phương pháp còn lại trong nhiều trường hợp. Hơn nữa, nó còn có thể giải quyết bài toán dữ liệu có phân bố thay đổi trên nhiều thuộc tính. Trong khi các phương pháp khác thường chỉ tập trung vào dữ liệu có phân bố thay đổi trên một thuộc tính [5]. Ngoài ra, học dựa trên mối quan hệ nhân quả có thể được ứng dụng mở rộng ra rất nhiều bài toán. Tuy nhiên, kiến thức nền tảng của học dựa trên mối quan hệ nhân quả theo chúng em nghĩ là không dễ để có thể làm chủ. Chúng em xác định hiểu rõ bài báo này cũng sẽ là cơ hội để hiểu rõ các kiến thức học dựa trên mối quan hệ nhân quả.

2.5 Kết quả dự kiến của đề tài

Qua khóa luận này, chúng em mong muốn đạt được các kết quả sau:

- Cài đặt từ đầu phương pháp được đề xuất và đạt được các kết quả đầu ra như trong bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” [5].
- Có các thử nghiệm để hiểu rõ tiềm năng phát triển của hướng tiếp cận này

cũng như hạn chế của nó.

- Thủ nghiệm phương pháp trên các bài toán hồi quy, hoặc trên các bộ dữ liệu văn bản và cải tiến phương pháp nếu có đủ thời gian.

2.6 Kế hoạch thực hiện

Thời gian	Công việc	Người thực hiện
Từ 01/01/2024 đến 31/01/2024	<ul style="list-style-type: none">- Tìm hiểu các bài báo khái quát miền dựa trên học nhân quả và chọn ra một phương pháp tiềm năng.- Lấy dữ liệu được sử dụng trong bài báo đó.	Phan Trưởng An Phạm Dương Trường Đức
Từ 01/02/2024 đến 15/03/2024	<ul style="list-style-type: none">- Tìm hiểu một số bài báo giải quyết vấn đề khái quát miền và chọn ra một số phương pháp để so sánh với bài báo chính.- Viết đề cương thực hiện khóa luận.- Nắm rõ ý tưởng thực hiện của bài báo chính.	Phan Trưởng An Phạm Dương Trường Đức

Từ 16/03/2024 đến 15/05/2024	<ul style="list-style-type: none"> - Hiểu sâu về kiến thức nền tảng của phương pháp học dựa trên quan hệ nhân quả. - Hiểu sâu bài báo chính. - Cài đặt lại phương pháp và tiến hành các thí nghiệm để tái hiện kết quả trong bài báo chính. - Tiến hành các thí nghiệm mở rộng (nếu còn thời gian). 	Phan Trường An Phạm Dương Trường Đức
Từ 16/05/2024 đến 31/06/2024	<ul style="list-style-type: none"> - Viết báo cáo khóa luận tốt nghiệp và chuẩn bị bài thuyết trình. 	Phan Trường An Phạm Dương Trường Đức

Tài liệu

- [1] B. Sun, J. Feng, and K. Saenko, “Return of frustratingly easy domain adaptation,” in *Proceedings of the AAAI conference on artificial intelligence*, 2016.
- [2] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *ICLR*, 2017.
- [3] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, “Generalizing to unseen domains: A survey on domain generalization,” *IEEE Transactions on Knowledge and Data Engineering*, 2022.
- [4] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales, “Learning to generalize: Meta-learning for domain generalization,” in *AAAI*, 2018.
- [5] J. N. Kaur, E. Kiciman, and A. Sharma, “Modeling the data-generating process is necessary for out-of-distribution generalization,” *International Conference on Learning Representations*, 2022.

XÁC NHẬN
CỦA NGƯỜI HƯỚNG DẪN
(Ký và ghi rõ họ tên)

Kiên
Trần Trung Kiên

Nguyễn Ngọc Thảo

TP. Hồ Chí Minh, ngày 04 tháng 04 năm 2024
NHÓM SINH VIÊN THỰC HIỆN
(Ký và ghi rõ họ tên)

Anh

Phan Trường An

Duc

Phạm Dương Trường Đức

Mục lục

Lời cam đoan	ii
Nhận xét của GV hướng dẫn	iii
Nhận xét của GV phản biện	iv
Lời cảm ơn	v
Đề cương	xiv
Danh sách hình	xvi
Danh sách bảng	xvii
Tóm tắt	xvii
1 Giới thiệu	1
2 Kiến thức nền tảng	6
2.1 Mô hình mạng nơ-ron đơn giản	6
2.1.1 Dạng mô hình	6
2.1.2 Huấn luyện mô hình bằng phương pháp ERM . . .	9
2.2 Phương pháp căn chỉnh phân bố dữ liệu CORAL	14
3 Mô hình mạng nơ-ron với phương pháp huấn luyện dựa trên quan hệ nhân quả CACM	17

3.1	Dạng mô hình	18
3.1.1	Kiến trúc CNN cho bài toán phân lớp ảnh	18
3.2	Huấn luyện mô hình bằng phương pháp CACM	21
3.2.1	Xác định đồ thị nhân quả ứng với dữ liệu đang xét	22
3.2.2	Huấn luyện mô hình bằng phương pháp CACM từ dữ liệu huấn luyện và đồ thị nhân quả	27
4	Thí nghiệm	35
4.1	Các thiết lập thí nghiệm	36
4.1.1	Các bộ dữ liệu	36
4.1.2	Các kiến trúc cụ thể của mạng nơ-ron	40
4.2	Các thí nghiệm	43
4.2.1	So sánh kết quả cài đặt của khóa luận với bài báo gốc	43
4.2.2	So sánh CACM sử dụng ràng buộc đúng với CACM sử dụng ràng buộc sai	46
4.2.3	Áp dụng CACM cho dữ liệu ảnh thực tế	48
5	Tổng kết và hướng phát triển	50
5.1	Tổng kết	50
5.2	Hướng phát triển	51
Tài liệu tham khảo		52

Danh sách hình

2.1	Ví dụ về kiến trúc của một mạng nơ-ron nhân tạo (ANN)	7
2.2	Kiến trúc của một FCFFNN đơn giản	9
2.3	Ý tưởng chính của phương pháp CORAL [14]	15
3.1	Cấu trúc chung của CNN (Theo [5])	19
3.2	Phép tính tích chập trên ma trận	20
3.3	(a) Đồ thị canonical nhân quả [10] mô tả quá trình sinh dữ liệu phổ biến có thể tạo tập dữ liệu có sự thay đổi đa thuộc tính. (b) Đồ thị canonical với mối quan hệ tương quan $E - \mathbf{X}_c$. (c) Các cấu trúc đồ thị khác nhau dẫn tới sự thay đổi <i>Causal</i> , <i>Confounded</i> và <i>Selected</i>	24
3.4	Các đồ thị nhân quả [10] cho từng loại phân bố thay đổi dựa trên mối quan hệ $Y - \mathbf{A}$	25
3.5	Ba đồ thị DAG đặc trưng cho độc lập có điều kiện [7] . . .	28
4.1	Một số mẫu của các bộ dữ liệu MNIST	37
4.2	Một số mẫu của bộ dữ liệu Light+Azi small NORB	38
4.3	Một số mẫu của bộ dữ liệu Camelyon17	40
4.4	Khối phần dư [9]	41

Danh sách bảng

4.1	Các giá trị để điều chỉnh siêu tham số	44
4.2	Kết quả thí nghiệm trên các bộ dữ liệu MNIST và small NORB. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.	45
4.3	Kết quả thí nghiệm CACM trên các bộ dữ liệu MNIST với ràng buộc sai. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.	47
4.4	Kết quả thí nghiệm CACM trên các bộ dữ liệu small NORB với ràng buộc sai. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.	47
4.5	Kết quả thí nghiệm trên bộ dữ liệu Camelyon17. Các kết quả ở đây là độ chính xác dự đoán(%).	49

Tóm tắt

“Dữ liệu có phân bố thay đổi” là một hiện tượng trong học có giám sát, khi dữ liệu mà mô hình phải dự đoán có sự thay đổi phân bố so với dữ liệu mà mô hình được huấn luyện. Điều này thường khiến cho độ chính xác của kết quả dự đoán của mô hình giảm. Ví dụ, trong bài toán nhận dạng, dữ liệu huấn luyện chỉ gồm những hình ảnh bò trên những đồng cỏ, khi ta đưa hình một chú bò trên bãi biển, mô hình sẽ đưa ra dự đoán sai. “Dữ liệu có phân bố thay đổi” là một thử thách mà hầu hết các mô hình học có giám sát phải đối mặt. Để có thể giải quyết vấn đề này, các mô hình học có giám sát phải có khả năng chống chịu và khái quát tốt được các phân bố dữ liệu khác nhau. Các bộ dữ liệu khác nhau có thể có các loại phân bố thay đổi khác nhau, hoặc thậm chí với cùng một bộ dữ liệu có thể xảy ra các loại phân bố thay đổi khác nhau trên các thuộc tính khác nhau. Đã có nhiều phương pháp được đề xuất để giải quyết bài toán “Dữ liệu có phân bố thay đổi” nhưng chưa có phương pháp nào có hiệu suất tốt và ổn định với nhiều loại phân bố thay đổi khác nhau. Một phương pháp đạt hiệu suất tốt với loại thay đổi này thì có hiệu suất kém với loại thay đổi khác. Khó khăn này tìm hiểu và cài đặt lại thành công phương pháp Causally Adaptive Constraint Minimization (CACM) được đề xuất trong bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization”. Đây là một phương pháp dựa trên mối quan hệ nhân quả giữa các biến có trong đồ thị nhân quả mô tả quá trình sinh một tập dữ liệu. Trong đồ thị nhân quả, giữa các biến sẽ có mối quan hệ khác nhau, dẫn đến các ràng buộc khác nhau. CACM sẽ sử dụng các ràng buộc phù

hợp với từng bộ dữ liệu để định hướng việc học. CACM có khả năng giải quyết bài toán trên những bộ dữ liệu khác nhau mà có các loại phân bố thay đổi khác nhau. Đồng thời, CACM cũng có thể giải quyết được cả những bộ dữ liệu có nhiều loại phân bố thay đổi khác nhau trên các thuộc tính khác nhau trong cùng một bộ dữ liệu. Bên cạnh đó, khóa luận cũng tìm hiểu các kiến trúc mô hình mà CACM có thể áp dụng và cài đặt lại các phương pháp khác (như ERM, CORAL) để so sánh hiệu suất với CACM. Khóa luận này chứa các thí nghiệm nhằm so sánh kết quả cài đặt CACM so với bài báo gốc, đánh giá hiệu suất của CACM dựa trên các ràng buộc đúng và ràng buộc sai; đồng thời, áp dụng CACM trên một bộ dữ liệu ảnh thực tế. Qua các thí nghiệm, các kết quả giúp rút ra kết luận rằng CACM đạt được hiệu suất tương đương hoặc tốt hơn ERM và CORAL trên các bộ dữ liệu có phân bố thay đổi trên đơn thuộc tính và đạt được hiệu suất vượt trội trên các bộ dữ liệu có phân bố thay đổi trên đa thuộc tính.

Chương 1

Giới thiệu

Học có giám sát là một loại của học máy mà máy tính sẽ được huấn luyện trên bộ dữ liệu gồm các cặp đầu vào và đầu ra đúng. Máy tính sẽ học trên bộ dữ liệu đó và đưa ra dự đoán cho dữ liệu đầu vào mới. Ví dụ, chúng ta cần xây dựng một mô hình nhận diện các loài động vật từ các hình ảnh được chụp bởi các máy ảnh đặt trên thảo nguyên. Chúng ta sẽ huấn luyện mô hình dựa vào một bộ dữ liệu hình ảnh của các con vật (đầu vào) và tên gọi đúng của chúng (đầu ra). Sau quá trình huấn luyện, khi chúng ta đưa vào một hình ảnh của một con vật nào đó được chụp trên thảo nguyên, mô hình sẽ đưa ra được tên gọi của con vật đó. Với sự tò mò và đam mê vô tận mong muốn máy tính có thể học và giải quyết được các yêu cầu của con người, các nhà nghiên cứu đã đề xuất ra vô số công trình nghiên cứu về các mô hình học có giám sát để máy tính có thể học và tự đưa ra dự đoán sau khi được huấn luyện từ dữ liệu. Các mô hình học có giám sát đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, dự đoán tài chính, y học, và nhiều lĩnh vực khác. Các mô hình học máy truyền thống được huấn luyện trên giả định rằng dữ liệu huấn luyện và dữ liệu kiểm tra sẽ độc lập và có cùng phân bố với nhau (IID - Independent and Identically Distributed). Tuy nhiên, điều đó không phải lúc nào cũng đúng trong thực tế. Khi dữ liệu huấn luyện và dữ liệu kiểm tra có phân bố khác nhau, hiệu suất của các mô hình học máy thường giảm [16]. Hiện tượng này được gọi là “Dữ

liệu có phân bố thay đổi” (Data distribution shift). Đây là vấn đề của hầu hết các mô hình học máy và cần được giải quyết để có thể áp dụng được các mô hình học máy này vào thực tế.

Bài toán “học có giám sát với dữ liệu có phân bố thay đổi” được tập trung tìm hiểu trong khoá luận là bài toán học có giám sát với dữ liệu có phân bố đầu vào thay đổi. Cụ thể, bài toán này được phát biểu như sau:

- Cho dữ liệu của K miền huấn luyện, trong đó dữ liệu của mỗi miền có dạng $\{(x_i, y_i)_{i=1}^n\}$ với $x \in \mathcal{X}$ là đầu vào và $y \in \mathcal{Y}$ là đầu ra tương ứng, n là số điểm dữ liệu, (x, y) được phát sinh từ phân bố $P(X, Y) = P(X)P(Y | X)$. Các miền này có cùng phân bố $P(Y | X)$ nhưng có phân bố $P(X)$ khác nhau. Với mỗi cặp (x, y) thì có thể có thêm thuộc tính a mà giá trị sẽ đặc trưng cho miền tương ứng và ảnh hưởng đến giá trị của x (ví dụ, với dữ liệu ảnh và các miền ứng với các người chụp khác nhau thì ta có thể có các thuộc tính a như góc chụp, giờ chụp,...).
- Yêu cầu của bài toán là tìm một hàm dự đoán $g : \mathcal{X} \rightarrow \mathcal{Y}$ từ dữ liệu của K miền huấn luyện được cho để có thể dự đoán tốt với dữ liệu của miền mục tiêu, với miền mục tiêu là một miền mới có cùng $P(Y | X)$ nhưng khác $P(X)$ với các miền huấn luyện. Nghĩa là hàm dự đoán g có thể khái quát và chống chịu tốt với sự thay đổi miền.

Trong phần trình bày của khóa luận, đôi khi chúng em gọi bài toán “học có giám sát với dữ liệu có phân bố thay đổi” ở trên là bài toán “khái quát hóa miền”.

Các bộ dữ liệu khác nhau có thể có các loại phân bố thay đổi khác nhau, hoặc thậm chí với cùng một bộ dữ liệu có thể xảy ra các loại phân bố thay đổi khác nhau trên các thuộc tính khác nhau. Đây là một thách thức cho các phương pháp giải quyết bài toán. Các phương pháp chỉ tập trung giải quyết một loại phân bố thay đổi sẽ không hoạt động tốt với dữ liệu có các loại phân bố thay đổi khác. Nếu giải quyết được bài toán này, ta có thể giúp các mô hình học máy khái quát tốt được các miền, từ đó

giúp mô hình chống chịu tốt với sự thay đổi phân bố dữ liệu, nhờ vậy các mô hình sẽ không bị giảm hiệu suất và đưa ra dự đoán đủ tốt theo yêu cầu của người dùng. Việc chống chịu tốt với sự thay đổi miền còn giúp mở rộng ứng dụng của các mô hình học máy trong thực tế.

“Dữ liệu có phân bố thay đổi” là một vấn đề luôn tồn tại trong các mô hình học có giám sát. Từ đó, đã có vô số nghiên cứu để giải quyết vấn đề này. Các phương pháp để giải quyết bài toán này có thể được chia thành ba nhóm: Xử lý dữ liệu (data manipulation), sử dụng các chiến lược học (learning strategy) và học biểu diễn (representation learning) [16]. Đại diện cho nhóm xử lý dữ liệu có thể được kể đến là phương pháp Mixup [18], đây là một phương pháp tăng cường dữ liệu do Zhang và cộng sự đề xuất. Mixup sinh dữ liệu mới bằng cách thực hiện nội suy tuyến tính giữa hai đầu vào bất kỳ và đầu ra tương ứng của chúng trong bộ dữ liệu với một trọng số được lấy mẫu từ phân bố Beta. Ý tưởng của Mixup nói riêng và các phương pháp khác trong nhóm này nói chung là dữ liệu càng nhiều và càng đa dạng thì mô hình sẽ có thể học được các đặc trưng chung của dữ liệu, từ đó khai quát tốt hơn. Tuy nhiên, điểm yếu của nhóm phương pháp này là có thể những đặc trưng mới được tạo ra không cần thiết cho việc khai quát khiến cho tiêu tốn tài nguyên mà không giải quyết được yêu cầu bài toán. Một số các chiến lược học cũng có thể giải quyết được yêu cầu khai quát miền như là Esemble learning, Meta-learning, Gradient operation,... [16]. Esemble learning khai thác mối quan hệ giữa các miền huấn luyện bằng cách sử dụng các kiến trúc mạng và các chiến lược huấn luyện cụ thể để cải thiện khả năng khai quát hóa với giả định rằng mỗi mẫu dữ liệu có thể được xem như sự kết hợp của nhiều miền huấn luyện khác nhau và kết quả dự đoán cuối cùng là tổng hợp kết quả từ các mạng được huấn luyện trên từng miền huấn luyện. Từ đó, mô hình có thể đưa ra dự đoán chính xác hơn bằng cách tận dụng thông tin từ nhiều nguồn khác nhau. Tuy nhiên, Esemble learning đòi hỏi sự tính toán phức tạp và không gian để huấn luyện và lưu trữ các mô hình khác nhau. Meta-learning cũng là một chiến lược được sử dụng để khai quát miền. MLDG [12] (meta-

learning for domain generalization) dựa trên ý tưởng của meta-learning, bằng cách chia dữ liệu trong các miền huấn luyện thành các bộ dữ liệu meta-train và meta-test để mô phỏng sự thay đổi miền, từ đó học được biểu diễn của các miền và khái quát chúng. Điểm yếu của phương pháp này là nếu dữ liệu trong các miền huấn luyện không đa dạng thì khó có thể khái quát tốt được trên miền mục tiêu và có thể dẫn đến hiện tượng overfitting trên các đặc điểm cụ thể của miền huấn luyện. Một phương pháp để giải quyết bài toán khái quát miền sử dụng chiến lược Gradient operation đó là *Correlation Alignment* (CORAL)[14]. CORAL điều chỉnh hiệp phương sai của dữ liệu từ miền huấn luyện theo dữ liệu của miền mục tiêu. Từ đó, dữ liệu từ miền huấn luyện sẽ có phân bố tương đồng với dữ liệu từ miền mục tiêu. Nhờ vậy mà mô hình huấn luyện được sẽ khái quát hóa tốt hơn. Phương pháp này đòi hỏi phải có được dữ liệu từ miền mục tiêu.

Các phương pháp dựa trên mối quan hệ nhân quả là hướng tiếp cận tiềm năng thuộc nhóm học biểu diễn. Các mô hình học máy thông thường có thể dựa vào các biến gây nhiễu trong dữ liệu huấn luyện để đưa ra quyết định khiến cho dự đoán không chính xác với dữ liệu mới của miền mục tiêu (ví dụ trong bài toán nhận diện con vật, các hình ảnh về bò trong dữ liệu huấn luyện thường xuất hiện cùng với nền cỏ; nên nếu ta đưa một hình ảnh con bò ở bãi biển thì mô hình có thể dự đoán không chính xác). Các mô hình học dựa trên mối quan hệ nhân quả có thể chỉ ra những nguyên nhân thật sự dẫn đến đầu ra đúng, chứ không bị ảnh hưởng bởi các biến gây nhiễu trong dữ liệu huấn luyện. Nhờ đó, các mô hình này có thể dự đoán tốt với dữ liệu mới của miền mục tiêu. Năm 2022, Kaur và cộng sự đã công bố bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” ở hội nghị International Conference on Learning Representations [10]. Trong bài báo này, nhóm tác giả đã đề xuất phương pháp *Causally Adaptive Constraint Minimization* (CACM). Điểm mạnh của CACM là không sử dụng các ràng buộc cố định để định hướng việc học, mà sẽ xác định đồ thị nhân quả (mô tả mối quan hệ giữa biến

đầu vào, biến đầu ra, biến nguyên nhân, biến gây nhiễu) phù hợp với từng bộ dữ liệu cụ thể, từ đó rút ra các ràng buộc phù hợp cho mỗi bộ dữ liệu. Nhờ đó, CACM có thể hoạt động tốt với các bộ dữ liệu khác nhau mà có các loại phân bố thay đổi khác nhau, cũng như là với một bộ dữ liệu mà chứa đồng thời các loại phân bố thay đổi khác nhau trên các thuộc tính khác nhau.

Trong khoá luận này, phương pháp được tập trung tìm hiểu sâu và cài đặt lại là phương pháp CACM được đề xuất trong bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” [10]. Phương pháp này có độ chính xác cao hơn các phương pháp còn lại trong nhiều trường hợp. Phần còn lại của khoá luận được trình bày như sau:

- Chương 2 trình bày các kiến thức cơ bản của các kiến trúc mô hình được sử dụng trong khoá luận. Đồng thời, trình bày các phương pháp được sử dụng để so sánh với CACM.
- Chương 3 trình bày chi tiết về một kiến trúc mô hình mà ta có thể áp dụng phương pháp CACM và việc huấn luyện mô hình bằng phương pháp CACM.
- Chương 4 trình bày chi tiết các thí nghiệm và phân tích về kết quả đạt được.
- Chương 5 trình bày tổng kết và hướng phát triển.

Chương 2

Kiến thức nền tảng

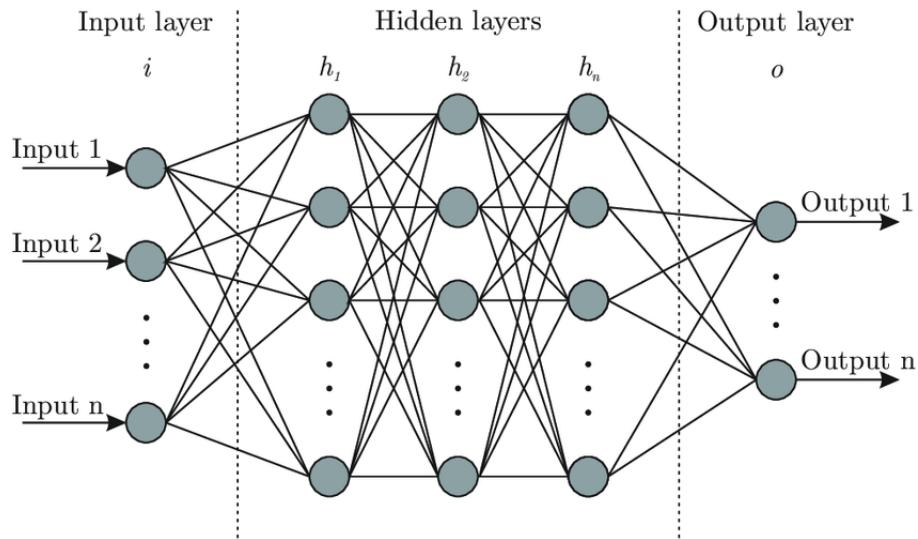
Chương này trình bày về các kiến thức nền tảng của khóa luận. Phần đầu tiên của chương này trình bày về mô hình mạng nơ-ron đơn giản, bao gồm: dạng/kiến-trúc mô hình và cách huấn luyện mô hình bằng phương pháp ERM. Mô hình đơn giản này là nền tảng cho mô hình chính mà khóa luận tập trung tìm hiểu (được trình bày ở chương 3); kiến trúc của mô hình đơn giản này và phương pháp ERM cũng sẽ được sử dụng trong phần thí nghiệm (được trình bày ở chương 4). Phần thứ hai của chương này trình bày về CORAL - một phương pháp căn chỉnh phân bố dữ liệu để huấn luyện mô hình tốt hơn với dữ liệu có phân bố thay đổi. Mô hình với phương pháp CORAL sẽ dùng trong phần thí nghiệm (ở chương 4) để so sánh với mô hình chính mà khóa luận tập trung tìm hiểu.

2.1 Mô hình mạng nơ-ron đơn giản

2.1.1 Dạng mô hình

Mạng nơ-ron nhân tạo (Artificial Neural Network - ANN) là một chủ đề nóng được quan tâm từ những năm 1980. Mạng nơ-ron nhân tạo mô phỏng mạng lưới thần kinh của não người trong quá trình xử lý thông tin,

nó là một mô hình tính toán được tạo bởi một số lượng lớn các nút (hoặc nơ-ron) được liên kết với nhau. Mỗi nút đại diện cho một hàm đầu ra cụ thể, gọi là hàm kích hoạt (activation function) và mỗi liên kết giữa hai nút bất kỳ đại diện cho trọng số thể hiện mức độ liên kết mạnh hay yếu giữa các nút (trọng số tương ứng với trí nhớ trong mạng nơ-ron nhân tạo). Đầu ra của mạng sẽ khác nhau dựa vào cách mạng được kết nối, giá trị trọng số và các hàm chức năng khác nhau trong mạng [17]. Mô hình dự đoán được tập trung tìm hiểu trong khóa luận là các mô hình dựa trên nền tảng mạng nơ-ron nhân tạo với cấu trúc cơ bản như hình 2.1. Các đơn vị dữ liệu x được truyền vào lớp đầu vào (Input layer) là dữ liệu mà ta cần mô hình dự đoán hoặc phân lớp. Các đơn vị ẩn trong các lớp ẩn (Hidden layers) là biểu diễn các cấp của dữ liệu đầu vào và không thể được quan sát ở ngoài mô hình. Lớp đầu ra (Output layer) sẽ đưa ra dự đoán y của mô hình về dữ liệu được cho.



Hình 2.1: Ví dụ về kiến trúc của một mạng nơ-ron nhân tạo (ANN)

Đã có hàng loạt các công trình nghiên cứu dựa trên ý tưởng của mạng nơ-ron và đưa ra rất nhiều các kiến trúc mô hình khác nhau. Tiêu biểu có mạng truyền thẳng kết nối đầy đủ (Fully-Connected Feed-Forward Neural Network - FCFFNN), mạng tích chập (Convolutional Neural Network - CNN), mạng hồi quy (Recurrent Neural Network - RNN), mạng dựa trên

kiến trúc Transformer. Đối với từng loại dữ liệu và độ phức tạp của dữ liệu mà chúng ta có thể sử dụng các kiến trúc mô hình khác nhau để giải quyết bài toán. FCFFNN là kiến trúc đơn giản và có thể áp dụng cho dữ liệu nói chung. Đối với dữ liệu dạng ảnh, chúng ta có thể sử dụng các kiến trúc dựa trên CNN như LeNet-5, VGGNet, ResNet,... Đối với dữ liệu dạng văn bản, chúng ta có thể sử dụng các mô hình RNN như LSTM. Đối với các bài toán xử lý ngôn ngữ như phân tích văn bản và nhận diện giọng nói, các mô hình tiêu biểu là các mô hình dựa trên kiến trúc Transformer như BERT. Với sự phát triển nhanh chóng và đa dạng, mạng nơ-ron nhân tạo đã được ứng dụng vào rất nhiều lĩnh vực như là khoa học máy tính, trí tuệ nhân tạo, khoa học nhận thức và các lĩnh vực khác như y khoa,...

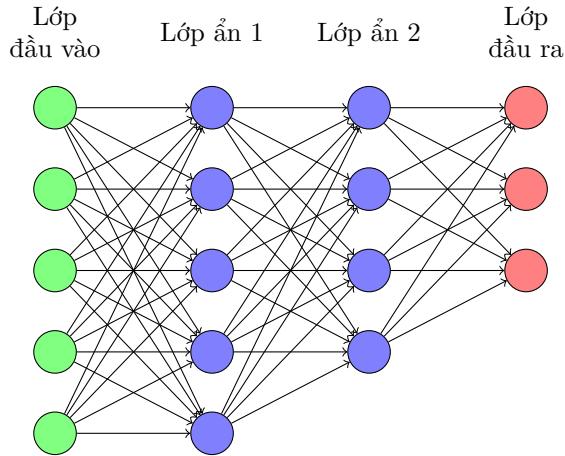
Sau đây là phần giới thiệu về mạng truyền thẳng kết nối đầy đủ dành cho bài toán phân lớp, được sử dụng trong thí nghiệm ở chương 4.

Mạng truyền thẳng kết nối đầy đủ cho bài toán phân lớp

Mạng truyền thẳng kết nối đầy đủ là một kiến trúc mạng nơ-ron nhân tạo đơn giản. Truyền thẳng nghĩa là các lớp của mạng có thứ tự trước sau, kết quả đầu ra của lớp trước được truyền qua lớp sau tuần tự cho đến lớp cuối cùng của mạng. Kết nối đầy đủ nghĩa là mỗi một nơ-ron sẽ có liên kết tới tất cả nơ-ron của lớp kế tiếp. Hình 2.2 là một ví dụ đơn giản của mạng truyền thẳng kết nối đầy đủ.

Mạng truyền thẳng kết nối đầy đủ bao gồm ba thành phần chính: lớp đầu vào, một hoặc nhiều lớp ẩn, lớp đầu ra.

- Lớp đầu vào nhận các đặc trưng từ dữ liệu đầu vào. Trong lớp này, với mỗi đặc trưng đầu vào, có một nơ-ron tương ứng để nhận vào đặc trưng đó. Nghĩa là số nơ-ron trong lớp này bằng với số đặc trưng của dữ liệu đầu vào. Ví dụ, với đầu vào là ảnh có kích thước 96x96 điểm ảnh, lớp đầu vào sẽ có 9216 nơ-ron (tương ứng với 9216 điểm ảnh).
- Các lớp ẩn có nhiệm vụ trích xuất những khuôn mẫu từ đặc trưng của dữ liệu đầu vào. Số lượng lớp ẩn và số nơ-ron của mỗi lớp được



Hình 2.2: Kiến trúc của một FCFFNN đơn giản

định nghĩa tùy theo từng bài toán. Hàm kích hoạt của các lớp ẩn là những hàm kích hoạt phi tuyến tính như ReLU, sigmoid,...

- Lớp đầu ra có nhiệm vụ đưa ra kết quả cuối cùng. Tùy vào bài toán để quyết định số lượng nơ-ron và hàm kích hoạt được sử dụng.

Đối với bài toán phân lớp được tập trung trong khóa luận, lớp đầu ra sẽ có hai loại như sau:

- Trong bài toán phân loại chỉ có 2 lớp, lớp đầu ra sẽ chỉ gồm 1 nơ-ron với hàm kích hoạt thường là sigmoid.
- Trong bài toán phân loại nhiều hơn 2 lớp, lớp đầu ra sẽ có số lượng nơ-ron bằng với số lớp cần phân loại và sử dụng hàm kích hoạt softmax. Mỗi nơ-ron cho ra kết quả là xác suất của mỗi lớp cần phân loại.

2.1.2 Huấn luyện mô hình bằng phương pháp ERM

Các thuật toán học truyền thống thường giả định rằng dữ liệu huấn luyện và dữ liệu kiểm tra sẽ độc lập và có cùng phân bố với nhau (IID). Nghĩa là phân bố của miền mục tiêu và miền huấn luyện sẽ giống nhau. Dựa vào giả định này, Empirical Risk Minimization (ERM) sẽ tìm cách tối

thiểu độ lỗi trung bình trên các miền dữ liệu huấn luyện. Từ đó, có thể đưa ra được mô hình tối ưu có thể khái quát được miền mục tiêu.

Cho dữ liệu từ các miền huấn luyện, chúng ta có thể ước lượng độ lỗi thực nghiệm (empirical risk) bằng cách tính trung bình của hàm lỗi trên n mẫu dữ liệu huấn luyện:

$$R(g) = \frac{1}{n} \sum_{i=1}^n L(g(x_i), y_i) \quad (2.1)$$

Với g là hàm dự đoán và L là hàm lỗi đo sự khác biệt giữa dự đoán $g(x)$ và đầu ra y .

Mục tiêu cuối cùng của ERM là tìm ra hàm g^* sao cho độ rủi ro $R(g)$ là tối thiểu.

$$g^* = \underset{g \in G}{\operatorname{argmin}} R(g) \quad (2.2)$$

Phương pháp ERM thường được sử dụng trong các thí nghiệm để so sánh với các phương pháp giải quyết bài toán khái quát miền khác. Vì vậy, phương pháp này sẽ được cài đặt và thử nghiệm để so sánh với phương pháp chính của khóa luận.

Hàm lỗi là một phần không thể thiếu trong việc huấn luyện mô hình học có giám sát. Đối với bài toán phân lớp, hàm lỗi thường được sử dụng là cross-entropy. Sau đây là phần trình bày về hàm lỗi cross-entropy và cách để cực tiểu hóa hàm lỗi này bằng gradient descent.

Hàm lỗi Cross-Entropy

Hàm lỗi cross-entropy, hay còn được gọi là hàm lỗi lô-ga-rít, là hàm lỗi được dùng phổ biến cho bài toán phân lớp. Hàm lỗi cross-entropy đo sự khác biệt giữa hai phân bố là phân bố thật và phân bố được dự đoán. Sự khác biệt này giúp đánh giá độ tốt của phân bố được dự đoán.

Với bài toán phân lớp nhị phân (chỉ có hai lớp 0 và 1), hàm lỗi cross-entropy được tính như sau:

$$\text{Loss} = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \quad (2.3)$$

Trong đó:

- y là nhãn đúng (0 hoặc 1)
- \hat{y} là xác suất được dự đoán của nhãn 1

Ví dụ, ta có nhãn đúng $y = 1$, xác suất được dự đoán $\hat{y} = 0.9$ thì:

$$\text{Loss} = -(1 \log(0.9) + (1 - 1) \log(1 - 0.9)) = -\log(0.9) \approx 0.105$$

Với bài toán phân lớp nhiều lớp, hàm lỗi cross-entropy được tính theo công thức sau:

$$\text{Loss} = - \sum_{i=1}^C y_i \log(\hat{y}_i) \quad (2.4)$$

Trong đó:

- C là số lớp của bài toán
- y_i là xác suất đúng của lớp thứ i (bằng 1 nếu lớp thứ i là lớp đúng, ngược lại bằng 0)
- \hat{y}_i là xác suất được dự đoán của lớp thứ i

Ví dụ, cho bài toán phân lớp có ba lớp (0, 1 hoặc 2). Với nhãn đúng là 1, xác suất đúng của ba lớp lần lượt là 0.0, 1.0, 0.0. Xác suất được dự đoán của 3 lớp lần lượt là 0.1, 0.6, 0.3. Vậy độ lỗi được tính như sau:

$$\text{Loss} = -(0.0 \cdot \log(0.1) + 1.0 \cdot \log(0.6) + 0.0 \cdot \log(0.3)) = -\log(0.6) \approx 0.511$$

Hàm lỗi cross-entropy luôn cho ra kết quả không âm, nó sẽ cho ra độ lỗi bằng 0 nếu xác suất dự đoán khớp hoàn toàn với xác suất đúng. Các xác

suất dự đoán càng lệch thì độ lỗi tính được bởi cross-entropy càng tăng. Những lớp sai nhưng có xác suất dự đoán cao sẽ làm cho độ lỗi tăng mạnh.

Hàm lỗi cross-entropy phù hợp cho các mô hình đưa ra xác suất dự đoán (như các mạng nơ ron có lớp đầu ra sử dụng hàm softmax). Cross-entropy cũng thích hợp với các phương pháp tối ưu hóa dựa trên gradient do có thể tính được đạo hàm. Sau đây là phần trình bày cách cực tiểu hóa hàm lỗi cross-entropy bằng gradient descent.

Cực tiểu hóa hàm lỗi Cross-Entropy bằng Gradient Descent

Gradient descent là một thuật toán tối ưu, được sử dụng rộng rãi để tối ưu hóa tham số của mạng nơ-ron. Ý tưởng chính của gradient descent là cập nhật tham số của mạng nơ-ron đi theo hướng ngược lại với gradient của hàm mất mát so với các tham số nhằm tối thiểu hàm mất mát.

Cho một hàm $f(\theta)$, với θ là các tham số (bao gồm trọng số W và bias b), gradient descent cập nhật tham số theo công thức sau:

$$\theta \leftarrow \theta - \eta \nabla_{\theta} f(\theta) \quad (2.5)$$

Trong đó:

- η là tốc độ học (learning rate), siêu tham số quyết định bước nhảy của quá trình cập nhật
- $\nabla_{\theta} f(\theta)$ là gradient của hàm f so với các tham số

Cho rằng lớp đầu ra của mạng nơ-ron sử dụng hàm softmax để tính xác suất dự đoán của các lớp, hàm softmax cho lớp j được tính như sau:

$$\hat{y}_j = \frac{e^{z_j}}{\sum_{k=1}^C e^{z_k}} \quad (2.6)$$

với z_j là điểm số của lớp j và $z_j = W \cdot x + b$

Đạo hàm của hàm lỗi cross-entropy so với z_j là:

$$\frac{\partial \text{Loss}}{\partial z_j} = \hat{y}_j - y_j \quad (2.7)$$

Đạo hàm của z_j so với W và b lần lượt là:

$$\frac{\partial z_j}{\partial W} = x \quad (2.8)$$

$$\frac{\partial z_j}{\partial b} = 1 \quad (2.9)$$

Sử dụng quy tắc chuỗi, ta tính được đạo hàm của hàm lỗi cross-entropy so với W và b như sau:

$$\frac{\partial \text{Loss}}{\partial W} = \frac{\partial \text{Loss}}{\partial z_j} \cdot \frac{\partial z_j}{\partial W} = (\hat{y}_j - y_j) \cdot x \quad (2.10)$$

$$\frac{\partial \text{Loss}}{\partial b} = \frac{\partial \text{Loss}}{\partial z_j} \cdot \frac{\partial z_j}{\partial b} = \hat{y}_j - y_j \quad (2.11)$$

Từ đạo hàm trên, theo công thức 2.5, ta cập nhật giá trị của W và b :

$$W \leftarrow W - \eta \cdot (\hat{y}_j - y_j) \cdot x \quad (2.12)$$

$$b \leftarrow b - \eta \cdot (\hat{y}_j - y_j) \quad (2.13)$$

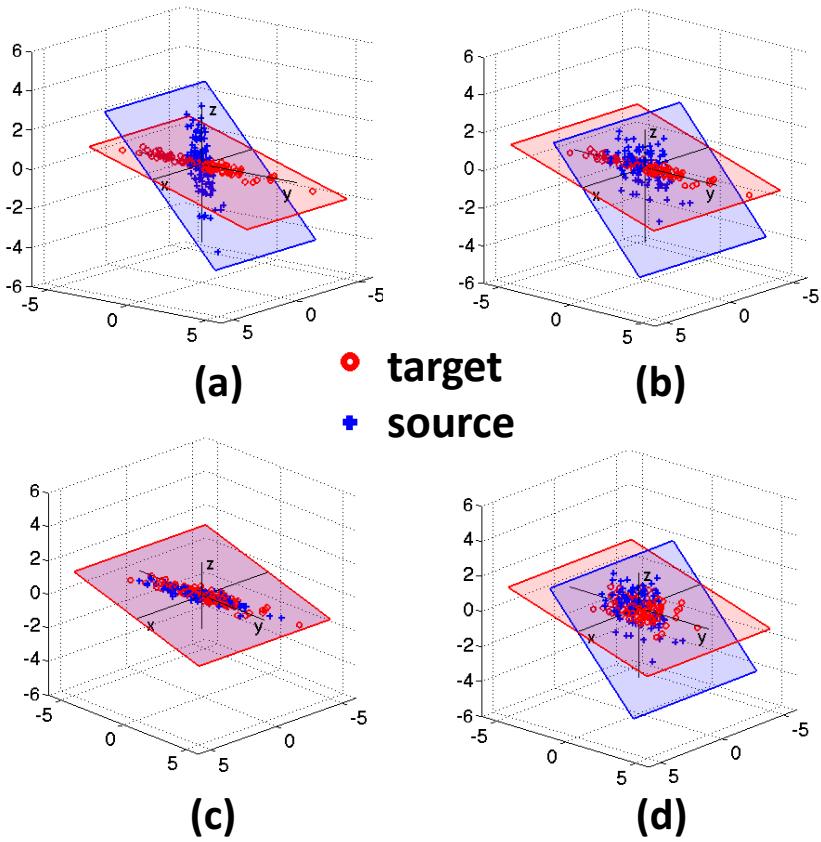
Quá trình cập nhật tham số được lặp lại nhiều lần cho đến khi hội tụ. Mỗi lần lặp bao gồm truyền thẳng (forward pass, tính ra giá trị dự đoán và độ lỗi) và truyền ngược (backward pass, tính gradient và cập nhật tham số cho mô hình). Quá trình này giúp giảm dần độ lỗi và tăng hiệu suất của mạng nơ-ron.

2.2 Phương pháp căn chỉnh phân bố dữ liệu CORAL

Correlation Alignment (CORAL) là một phương pháp dùng để giải quyết bài toán khái quát miền được đề xuất trong bài báo “Return of Frustratingly Easy Domain Adaptation” của Sun và cộng sự [14]. Phương pháp CORAL căn chỉnh phân bố dữ liệu huấn luyện theo dữ liệu mục tiêu dựa trên hiệp phương sai. Cụ thể hơn, các đặc trưng từ dữ liệu huấn luyện sẽ được “làm trắng” (whitening) và “tô màu lại” (re-coloring) theo hiệp phương sai của dữ liệu mục tiêu. Quá trình “làm trắng” và “tô màu lại” đơn giản là các phép biến đổi tuyến tính. Sau khi có được dữ liệu huấn luyện đã căn chỉnh, ta huấn luyện mô hình học có giám sát bằng dữ liệu này. Bởi vì dữ liệu huấn luyện đã căn chỉnh có phân bố tương đồng với dữ liệu mục tiêu nên mô hình sẽ có hiệu suất tốt, chống chịu tốt với thay đổi miền.

Ý tưởng của phương pháp CORAL được minh họa trong hình 2.3 (a-c). Hình 2.3a cho thấy dữ liệu của miền nguồn và miền mục tiêu có phân bố khác nhau, có thể dẫn đến giảm hiệu suất của mô hình được huấn luyện. Hình 2.3b minh họa quá trình “làm trắng” trên miền nguồn, loại bỏ các tương quan của đặc trưng trong miền dữ liệu. Hình 2.3c minh họa quá trình “tô màu lại”, thêm các tương quan của miền mục tiêu vào miền nguồn, dữ liệu của hai miền đã có phân bố tương đồng nhau nên mô hình sẽ có hiệu suất tốt. Hình 2.3d cho thấy nếu ta thực hiện “làm trắng” trên cả hai miền thì sẽ không thành công.

Giả sử trong một bài toán học có giám sát, chúng ta được cho dữ liệu miền nguồn D_S cùng với nhãn Y_S , và dữ liệu miền mục tiêu D_T không kèm nhãn. Ta gọi C_S và C_T lần lượt là ma trận hiệp phương sai của dữ liệu miền nguồn và miền mục tiêu. Ta đặt $C_{\hat{S}} = A^\top C_S A$. Mục tiêu của phương pháp CORAL là tìm ra ma trận A sao cho khoảng cách giữa ma trận $C_{\hat{S}}$ và C_T là nhỏ nhất. Từ đó, ta tính ra được dữ liệu miền nguồn



Hình 2.3: Ý tưởng chính của phương pháp CORAL [14]

mới $D_S^* = D_S A$ và sử dụng dữ liệu này để huấn luyện mô hình.

Qua những chứng minh của mình, Sun và cộng sự [14] đã đưa ra thuật toán CORAL như sau:

Thuật toán 1 CORAL

Đầu vào: Dữ liệu miền nguồn D_S , Dữ liệu miền mục tiêu D_T

Đầu ra: Dữ liệu miền nguồn đã điều chỉnh D_S^*

$$C_S = cov(D_S) + eye(size(D_S, 2))$$

$$C_T = cov(D_T) + eye(size(D_T, 2))$$

$$D_S = D_S * C_S^{-\frac{1}{2}} \quad \% \text{ làm trắng}$$

$$D_S^* = D_S * C_T^{\frac{1}{2}} \quad \% \text{ tô màu lại}$$

Trong đó, $eye(size(D_S, 2))$ là ma trận $I_{n \times n}$ với n là số cột của D_S . Phép cộng với $I_{n \times n}$ nhằm mục đích tăng hạng của ma trận C_S bởi vì hạng của ma trận này thường nhỏ. Phép tính cũng được thực hiện tương tự với C_T .

Phương pháp CORAL có thể được áp dụng trong bài toán nhận diện vật thể hoặc xử lý ngôn ngữ tự nhiên. Bởi vì phương pháp này chỉ cần những phép tính đơn giản trên ma trận, nên có thể dễ dàng cài đặt. Phương pháp CORAL cũng thường được cài đặt trong các thí nghiệm để so sánh với các phương pháp giải quyết bài toán khai quát miền khác, cho nên phương pháp này sẽ được cài đặt và thí nghiệm để so sánh với phương pháp chính được chọn của khóa luận là CACM.

Chương 3

Mô hình mạng nơ-ron với phương pháp huấn luyện dựa trên quan hệ nhân quả CACM

Chương này trình bày về mô hình mạng nơ-ron với phương pháp huấn luyện dựa trên quan hệ nhân quả CACM (Causally Adaptive Constraint Minimization) đã được Kaur và cộng sự [10] đề xuất để giải quyết bài toán học có giám sát với dữ liệu có phân bố thay đổi; đây là phương pháp chính được tập trung tìm hiểu trong khóa luận. Trong chương này, phần đầu tiên trình bày về dạng của mô hình mạng nơ-ron; chương 2 đã trình bày về một kiến trúc mạng đơn giản, phần này sẽ trình bày về một kiến trúc mạng phức tạp hơn mà cũng sẽ được sử dụng trong các thí nghiệm ở chương 4. Phần thứ hai của chương sẽ trình bày về cách huấn luyện mô hình mạng nơ-ron bằng phương pháp CACM; sẽ có 2 bước thực hiện: (1) xác định đồ thị nhân quả ứng với dữ liệu đang xét, (2) huấn luyện mô hình bằng phương pháp CACM từ dữ liệu huấn luyện và đồ thị nhân quả.

3.1 Dạng mô hình

Như đã trình bày ở mục 2.1.1, mỗi loại dữ liệu khác nhau cần kiến trúc mạng nơ-ron phù hợp. Phương pháp CACM không giới hạn loại mô hình mạng nơ-ron sử dụng mà người dùng sẽ chọn tùy theo bài toán và dữ liệu cần xử lý. Đó có thể là các kiến trúc mạng như FCFFNN, CNN, RNN, Transformer,...

Sau đây là phần trình bày cụ thể kiến trúc của CNN cho dữ liệu ảnh. Kiến trúc này cũng được sử dụng trong phần thí nghiệm ở chương 4.

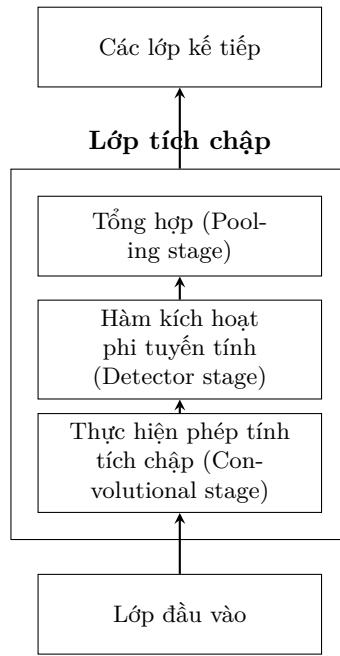
3.1.1 Kiến trúc CNN cho bài toán phân lớp ảnh

Mạng nơ-ron tích chập (Convolutional Neural Network - CNN) là một kiểu kiến trúc phổ biến dành cho dữ liệu dạng ảnh. Mạng tích chập chia ảnh đầu vào thành các phần nhỏ hơn để tìm ra những khuôn mẫu trong ảnh. Các khuôn mẫu này giúp nhận diện những thành phần có trong ảnh như góc cạnh, hình dạng,... Kiến trúc CNN có nhiều ứng dụng như như nhận diện khuôn mặt, nhận diện vật thể, phân lớp hình ảnh,... CNN thường có kiến trúc như hình 3.1.

Phép tính tích chập trên ma trận

Phép tính tích chập trên ma trận chính là nền tảng của mạng nơ-ron tích chập. Phép tính tích chập có thể giúp thực hiện các thao tác xử lý ảnh như làm mờ, phát hiện đường biên của vật thể. Trong mạng nơ-ron tích chập, phép tính tích chập giúp trích xuất các đặc trưng của ảnh đầu vào, từ đó giúp đưa ra các dự đoán.

Thành phần không thể thiếu để tính tích chập là nhân (kernel), hay còn được gọi là bộ lọc (filter). Nhân là một ma trận kích thước nhỏ, được trượt qua các vùng trên ma trận đầu vào để thực hiện tính toán. Giá trị sải bước (stride) sẽ quyết định khoảng cách mà kernel di chuyển sau mỗi bước.



Hình 3.1: Cấu trúc chung của CNN (Theo [5])

Với mỗi vị trí đặt nhân lên ma trận đầu vào, ta thực hiện phép nhân theo từng phần tử của vùng ma trận đầu vào tương ứng với nhân. Sau đó, ta cộng các giá trị lại để có kết quả trong ma trận đầu ra tại vị trí tương ứng.

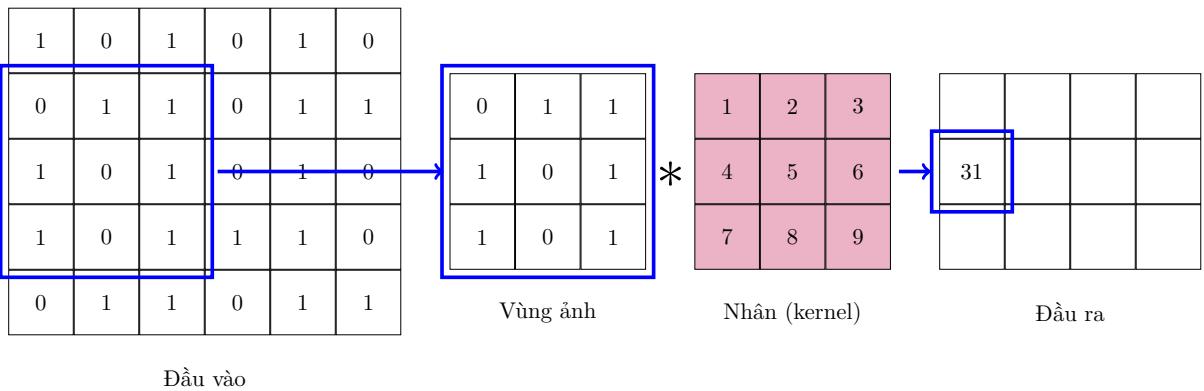
Ví dụ, trong hình 3.2, phép nhân từng phần tử của vùng ảnh và nhân sẽ là:

$$\begin{bmatrix} 0 * 1 & 1 * 2 & 1 * 3 \\ 1 * 4 & 0 * 5 & 1 * 6 \\ 1 * 7 & 0 * 8 & 1 * 9 \end{bmatrix}$$

Cộng các giá trị của ma trận, ta được kết quả là 31, kết quả được điền vào vị trí tương ứng trong ma trận đầu ra.

Kiến trúc CNN cho bài toán phân lớp ảnh

Kiến trúc CNN dành cho bài toán phân lớp ảnh bao gồm những thành phần sau:



Hình 3.2: Phép tính tích chập trên ma trận

- **Lớp đầu vào:** dùng để nhận dữ liệu ảnh đầu vào. Ảnh đầu vào thường ở dạng ma trận ba chiều: chiều rộng, chiều dài, các kênh màu.
- **Lớp tích chập (Convolutional Layer):** có nhân (hoặc bộ lọc) được trượt qua ảnh đầu vào và thực hiện phép tính tích chập nhằm mục đích nhận diện các góc cạnh, kết cấu hay khuôn mẫu trong ảnh. Kết quả của lớp tích chập sẽ được đưa qua hàm kích hoạt phi tuyến tính như ReLU. Sau đó, giai đoạn tổng hợp thực hiện giảm kích thước không gian của các đặc trưng từ lớp tích chập, giúp giảm lượng tài nguyên tính toán cần dùng. Có hai kiểu tổng hợp là lấy giá trị lớn nhất (max pooling) và lấy giá trị trung bình (average pooling).
- **Lớp kết nối đầy đủ:** các đặc trưng trích xuất được từ lớp tổng hợp được đưa qua một hoặc nhiều lớp kết nối đầy đủ để thực hiện các lý luận cấp cao hơn, từ đó giúp phân lớp ảnh.
- **Lớp đầu ra:** tương tự như FCFFNN được trình bày ở 2.1.1, số lượng nơ-ron và hàm kích hoạt của lớp đầu ra phụ thuộc vào số lớp của bài toán phân lớp.
 - Trong bài toán phân lớp chỉ có 2 lớp, lớp đầu ra sẽ chỉ gồm 1 nơ-ron với hàm kích hoạt thường là sigmoid.

- Trong bài toán phân lớp nhiều hơn 2 lớp, lớp đầu ra sẽ có số lượng nơ-ron bằng với số lớp và sử dụng hàm kích hoạt softmax. Mỗi nơ-ron cho ra kết quả là xác suất của mỗi lớp cần phân loại.

Số lượng lớp tích chập, lớp tổng hợp cũng như các nhân bên trong của chúng là khác nhau tùy vào từng kiến trúc cụ thể.

LeNet-5, VGGNet, ResNet là một số kiến trúc CNN phổ biến cho bài toán phân lớp ảnh.

3.2 Huấn luyện mô hình bằng phương pháp CACM

Phương pháp CACM, được đề xuất trong bài báo “Modeling the Data-Generating Process is Necessary for Out-of-Distribution Generalization” của Kaur và cộng sự [10], là phương pháp học biểu diễn dựa trên mối quan hệ nhân quả để khái quát miền. Các phương pháp khái quát miền như ERM, CORAL thường không đạt được kết quả nhất quán trên các loại thay đổi phân bố khác nhau do chỉ tập trung vào một loại thay đổi. Từ đó, Kaur và cộng sự [10] đã đề xuất ra phương pháp CACM để giải quyết bài toán thay đổi phân bố trên nhiều loại thay đổi thuộc tính khác nhau. Ngoài ra, phương pháp CACM có khả năng khái quát tốt sự thay đổi phân bố đa thuộc tính trên các miền. Sau đây là phần trình bày về khái quát đa thuộc tính trên các miền.

Mục đích của bài toán khái quát miền là học được một hàm phân loại $g(x)$ có thể khái quát được các miền dữ liệu huấn luyện và đạt được độ rủi ro nhỏ và tương tự trên miền dữ liệu mục tiêu như trên các miền dữ liệu huấn luyện. Trong bài báo [10], nhóm tác giả đã định nghĩa hàm dự đoán rủi ro bất biến (risk-invariant predictor) cho một tập hợp các phân bố D như sau,

Hàm dự đoán rủi ro bất biến tối ưu cho D . Định nghĩa độ rủi ro cho hàm dự đoán g cho phân bố $P \in D$ là $\mathcal{R}_{P(g)} = E_{x,y \sim P} \ell(g(x), y)$ với ℓ

là cross-entropy hoặc bất kỳ hàm lỗi phân loại nào khác. Từ đó, tập hợp những hàm dự đoán rủi ro bất biến bao gồm những hàm dự đoán rủi ro có cùng độ rủi ro qua tất cả các phân bố $P \in D$ và tập hợp những hàm dự đoán rủi ro bất biến tối ưu được định nghĩa là các hàm rủi ro bất biến có độ rủi ro nhỏ nhất trên tất cả các phân bố.

$$g_{\text{rinv}} \in \arg \min_{g \in G_{\text{rinv}}} \mathcal{R}_{P(g)} \forall P \in D, \text{ với } G_{\text{rinv}} = \{g : \mathcal{R}_{P(g)} = \mathcal{R}_{P'(g)} \forall P, P' \in D\} \quad (3.1)$$

Theo nhóm tác giả, một cách trực quan để chỉ ra được một hàm dự đoán rủi ro bất biến là chỉ cần xem xét một phần đặc trưng đầu vào (\mathbf{X}) gây ra nhãn Y và bỏ qua mọi biến thể được gây ra bởi các thuộc tính giả (spurious attributes). Trong không gian ẩn (latent space), xét đặc trưng nhân quả không quan sát được \mathbf{X}_c . Kaur và cộng sự giả định rằng $P(Y | \mathbf{X}_c)$ là bất biến trên tất cả các phân bố khác nhau. Sử dụng định nghĩa độ rủi ro bất biến (risk invariance), nhóm tác giả đã định nghĩa vấn đề khái quát đa thuộc tính như sau,

Khái quát dưới sự thay đổi phân bố của đa thuộc tính. Cho nhãn mục tiêu Y , các đặc trưng đầu vào \mathbf{X} , các thuộc tính \mathbf{A} và các đặc trưng nhân quả \mathbf{X}_c . Xét một tập hợp các phân bố D sao cho $P(Y | \mathbf{X}_c)$ luôn bất biến trong khi $P(\mathbf{A} | Y)$ thay đổi qua từng phân bố riêng lẻ. Sử dụng một tập huấn luyện $(\mathbf{x}_i, \mathbf{a}_i, y_i)_{i=1}^n$ được lấy mẫu từ một tập con các phân bố $D_{Train} \subset D$, mục tiêu của sự khái quát là học được một hàm dự đoán rủi ro tối ưu trên D .

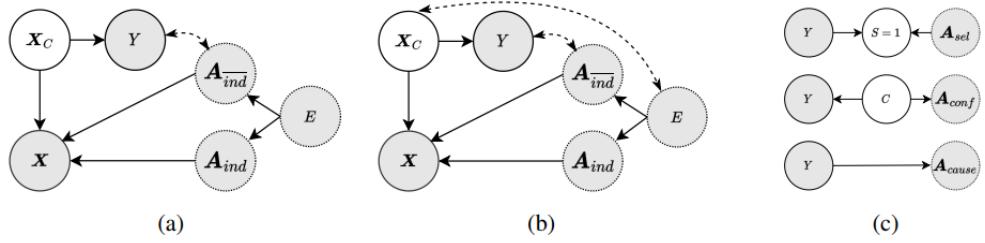
Trong trường hợp $|\mathbf{A}| = 1$, chúng ta sẽ có được bài toán khái quát trên đơn thuộc tính.

3.2.1 Xác định đồ thị nhân quả ứng với dữ liệu đang xét

Dựa vào quá trình sinh dữ liệu của một bộ dữ liệu, chúng ta có thể mô hình hóa mối quan hệ giữa các biến trong dữ liệu thông qua một đồ

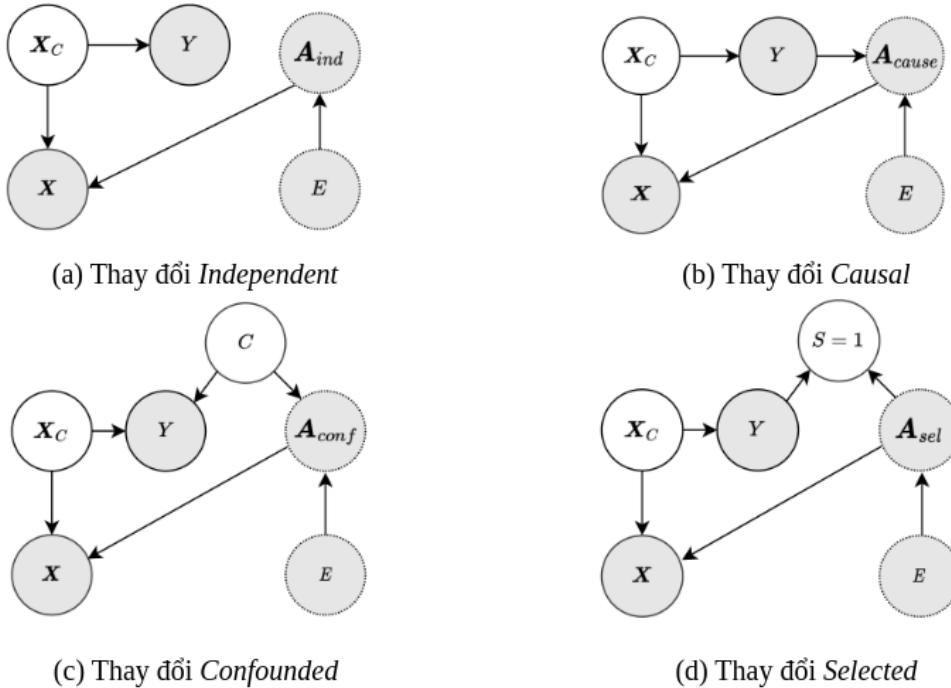
thị nhân quả. Đồ thị nhân quả thường được biểu diễn bằng một đồ thị có hướng không chu trình DAG (Directed acyclic graph) với các đỉnh thể hiện các biến có trong dữ liệu và các cạnh của đồ thị thể hiện mối quan hệ nhân quả trực tiếp (ví dụ, cạnh có hướng $x \rightarrow y$ thể hiện x là nguyên nhân sinh ra y).

Trong bài báo [10], Kaur và cộng sự nhấn mạnh sự cần thiết của việc mô hình hoá quá trình sinh ra dữ liệu (data-generating process) trong khái quát sự thay đổi phân bố. Để có thể sử dụng phương pháp CACM, đầu tiên, chúng ta cần xác định đồ thị nhân quả mô tả quá trình sinh ra dữ liệu ứng với dữ liệu đang xét. Kaur và cộng sự đã đề xuất một đồ thị canonical nhân quả (canonical causal graph) mô tả quá trình sinh dữ liệu phổ biến mà có thể tạo tập dữ liệu có sự thay đổi phân bố trên đa thuộc tính như hình 3.3. Đồ thị DAG này bao gồm các nút $\mathbf{X}_c, \mathbf{X}, \mathbf{A}, Y$ tương ứng với bộ dữ liệu $(\mathbf{x}_i, \mathbf{a}_i, y_i)_{i=1}^n$ được sinh từ đồ thị này. Trong đó, \mathbf{X}_c là tập hợp tất cả nguyên nhân gây ra Y ($\mathbf{X}_c \rightarrow Y$) và \mathbf{X}_c không quan sát được. \mathbf{X}_c và \mathbf{A} cùng gây ra \mathbf{X} ($\mathbf{X}_c \rightarrow \mathbf{X}$ và $\mathbf{A} \rightarrow \mathbf{X}$). \mathbf{A} là tập hợp các thuộc tính của bộ dữ liệu và có thể chia thành các tập hợp các thuộc tính $\mathbf{A}_{\overline{ind}}, \mathbf{A}_{ind}$ và E sao cho $\mathbf{A}_{\overline{ind}} \cup \mathbf{A}_{ind} \cup E = \mathbf{A}$. $\mathbf{A}_{\overline{ind}}$ thể hiện các thuộc tính tương quan với nhau, \mathbf{A}_{ind} là các thuộc tính độc lập với nhau, E là một thuộc tính đặc biệt thể hiện miền hoặc môi trường nơi mà điểm dữ liệu được thu thập. Không phải tất cả các thuộc tính cần được quan sát. Ví dụ trong vài trường hợp, chỉ có E và một tập con của $\mathbf{A}_{\overline{ind}}, \mathbf{A}_{ind}$ có thể được quan sát. Trong các trường hợp khác, chỉ có $\mathbf{A}_{\overline{ind}}$ và \mathbf{A}_{ind} được quan sát còn E thì không.



Hình 3.3: (a) Đồ thị canonical nhân quả [10] mô tả quá trình sinh dữ liệu phô biến có thể tạo tập dữ liệu có sự thay đổi đa thuộc tính. (b) Đồ thị canonical với mối quan hệ tương quan $E - \mathbf{X}_c$. (c) Các cấu trúc đồ thị khác nhau dẫn tới sự thay đổi *Causal*, *Confounded* và *Selected*.

Trong đồ thị 3.3, các nút xám thể hiện các biến có thể quan sát được như \mathbf{X}, Y . Đối với \mathbf{A} , không phải tất cả các thuộc tính đều quan sát được nên nhóm tác giả đã sử dụng viền nét đứt. Các cạnh liền thể hiện mối quan hệ nhân quả trực tiếp, còn các cạnh nét đứt thể hiện mối quan hệ tương quan như giữa Y và \mathbf{A}_{ind} hoặc giữa \mathbf{X}_c và E ở hình 3.3b. Dựa vào đồ thị canonical mà Kaur và cộng sự đã đề xuất, họ mô tả đặc điểm của các loại thay đổi phân bố khác nhau dựa vào mối quan hệ giữa các thuộc tính giả \mathbf{A} và nhãn phân loại Y . Cụ thể, \mathbf{A}_{ind} độc lập với nhãn và sự thay đổi trong $P(\mathbf{A}_{ind})$ dẫn tới một sự thay đổi phân bố *Independent*. Đối với \mathbf{A}_{ind} có ba cấu trúc đồ thị (hình 3.3c) thể hiện mối tương quan giữa \mathbf{A}_{ind} và Y đó là: mối quan hệ nhân quả trực tiếp (Y gây ra \mathbf{A}_{ind}), nhiều giữa Y và \mathbf{A}_{ind} bởi một nguyên nhân chung hoặc mối quan hệ lựa chọn trong quá trình sinh dữ liệu. Mỗi quan hệ tương quan giữa E và \mathbf{X}_c (hình 3.3) cũng có thể có các mối tương quan tương tự giữa \mathbf{A}_{ind} và Y . Nhìn chung, nhóm tác giả đã định nghĩa bốn loại thay đổi dựa vào đồ thị nhân quả: *Independent*, *Causal*, *Confounded*, và *Selected*. Đồ thị canonical trong hình 3.3a là đồ thị chung, ứng với mỗi cạnh giữa Y và \mathbf{A} sẽ dẫn đến một đồ thị nhận dạng nhân quả (realized causal) DAG thể hiện mối quan hệ giữa một thuộc tính và nhãn cho một bộ dữ liệu cụ thể. Các đồ thị nhận dạng nhân quả cho từng loại thay đổi phân bố dựa trên mối quan hệ $Y - \mathbf{A}$ được thể hiện như trong hình 3.4.



Hình 3.4: Các đồ thị nhận dạng [10] cho từng loại phân bố thay đổi dựa trên mối quan hệ $Y - A$.

Các đồ thị nhận dạng ở hình 3.4 thể hiện mối quan hệ giữa các thuộc tính phụ trợ (auxiliary attributes) và nhãn trong quá trình sinh dữ liệu của các bộ dữ liệu trong thực tế.

Đồ thị nhận dạng nhãn quả ở hình 3.4a thể hiện mối quan hệ *Independent* giữa Y và A_{ind} nghĩa là nhãn Y sẽ độc lập với thuộc tính A_{ind} và không có bất kỳ mối tương quan nào giữa A_{ind} và Y (sự thay đổi của A_{ind} sẽ không ảnh hưởng đến Y). Ví dụ, trong bộ dữ liệu có phân bố thay đổi đa thuộc tính Col+Rot MNIST [10] được Kaur và cộng sự kết hợp từ hai bộ dữ liệu Colored MNIST [1] và Rotated MNIST [4], thuộc tính *rotation* được tạo ra một cách độc lập không có sự tương quan với nhãn. Hoặc trong bộ dữ liệu gồm các hình ảnh chụp vệ tinh FMoW [3] thuộc tính *time* được xem là một thuộc tính độc lập (A_{ind}) bởi vì thời gian chụp ảnh sẽ không có sự tương quan với nhãn của hình ảnh.

Hình 3.4b thể hiện mối quan hệ *Causal* giữa Y và A_{cause} , cụ thể giữa Y và A_{cause} có mối quan hệ tương quan nhãn quả trực tiếp Y sẽ gây ra A_{cause} .

Ví dụ, trong bộ dữ liệu Col+Rot MNIST được đề cập ở trên. Thuộc tính *color* được tạo ra có sự tương quan với nhãn. Trên các miền huấn luyện, những số nhỏ hơn 5 sẽ có nhãn 0 và sẽ được “tô màu” đỏ trong khi những số lớn hơn hoặc bằng 5 sẽ có nhãn bằng 1 và được “tô màu” xanh lá. Vì vậy, thuộc tính *color* sẽ là thuộc tính \mathbf{A}_{cause} được gây ra trực tiếp bởi nhãn Y . Hoặc trong bộ dữ liệu Waterbirds [13] được tạo ra bằng cách dán hình những loài chim nước (waterbird) vào hình nền nước (biển, hồ tự nhiên) và dán những loài không phải chim nước vào hình nền trên cạn (rừng tre, rừng lá rộng). Bộ dữ liệu này có nhãn là chim nước (waterbird) và chim cạn (landbird), do hình nền (*background*) được lựa chọn dựa vào nhãn nên thuộc tính *background* (land/water) sẽ là thuộc tính \mathbf{A}_{cause} .

Đối với đồ thị nhận dạng ở hình 3.4c, mỗi quan hệ tương quan giữa Y và \mathbf{A}_{conf} được gây ra bởi một nguyên nhân chung C . Trong bộ dữ liệu FMoW [3], thuộc tính *region* biểu thị khu vực nơi ảnh được chụp được coi là thuộc tính \mathbf{A}_{conf} . Điều này là do các khu vực nhất định thường có nhãn Y xác định được đại diện quá mức, do sự thuận lợi trong việc thu thập dữ liệu từ các khu vực đó. *Region* không thể dẫn đến thay đổi *Causal* bởi vì quyết định chụp ảnh không được xác định bởi nhãn cuối cùng, cũng không thể dẫn đến thay đổi *Selected* do quyết định chụp ảnh không dựa vào giá trị của Y [10]. Một ví dụ khác thể hiện mối quan hệ này là: vào thời kỳ COVID-19, bệnh nhân sẽ được chụp CT ngực để xác định nhiễm bệnh. Một nghiên cứu ở MIT Technology Review¹ đã thảo luận về vấn đề có sự tương quan giả (spurious correlation) giữa vị trí của một người khi chụp CT ngực với kết quả dự đoán bệnh của họ. Đó là, những bệnh nhân nằm để chụp thường có bệnh nặng hơn những người đứng để chụp. Khi đó, thuộc tính *vị trí* (\mathbf{A}_{conf}) tương quan giả với nhãn vì cùng bị ảnh hưởng bởi tình trạng của bệnh nhân.

Đồ thị nhận dạng ở hình 3.4d, thể hiện mối quan hệ tương quan giữa Y và \mathbf{A}_{sel} khi có một điều kiện (sự kiện) S phụ thuộc vào cả Y và \mathbf{A}_{sel} , sao

¹<https://www.technologyreview.com/2021/07/30/1030329/machine-learning-ai-failed-hospital-diagnosis-pandemic/>

cho một điểm dữ liệu từ tổng thể chỉ được đưa vào mẫu nếu $S = 1$ đúng [15]. Mỗi tương quan này xảy ra trong quá trình sinh dữ liệu, ví dụ, trong bài toán chấm điểm phim, dữ liệu huấn luyện của chúng ta chỉ có những bộ phim có trên 100 lượt đánh giá (S). Chỉ những phim kinh dị xuất sắc mới có nhiều lượt đánh giá, trong khi hầu hết các phim hài hoặc tình cảm đều có nhiều lượt đánh giá. Như vậy thì sự lựa chọn (S) trong quá trình tạo dữ liệu huấn luyện dẫn đến mối liên hệ *Selection* giữa *thể loại* (\mathbf{A}_{sel}) và *điểm đánh giá* (Y).

3.2.2 Huấn luyện mô hình bằng phương pháp CACM từ dữ liệu huấn luyện và đồ thị nhân quả

Trong thực tế, đặc trưng nhân quả \mathbf{X}_c không quan sát được và mục đích của chúng ta là học được \mathbf{X}_c thông qua các biến quan sát được ($\mathbf{X}, Y, \mathbf{A}$). Nhờ vào sự độc lập và ổn định của cấu trúc nhân quả, Kaur và cộng sự [10] đề xuất thuật toán CACM dựa vào đồ thị nhân quả thể hiện quá trình sinh dữ liệu để khái quát miền. Thuật toán CACM gồm hai pha:

- **Pha I:** Rút ra các ràng buộc độc lập chính xác.
- **Pha II:** Áp dụng Regularization penalty sử dụng các ràng buộc đã rút ra.

Pha I: Rút ra các ràng buộc độc lập chính xác

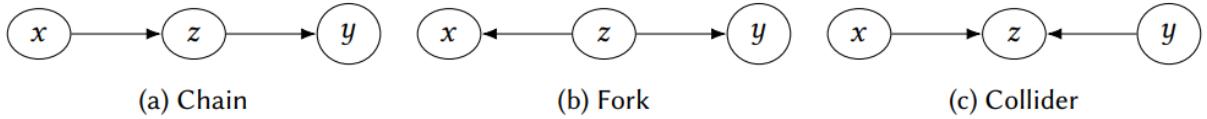
Như đã trình bày ở trên, đồ thị canonical ở hình 3.3 là một đồ thị nhân quả mô tả quá trình sinh dữ liệu phổ biến. Trong đồ thị này, nhóm tác giả đã mô tả các loại thay đổi phân bố khác nhau dựa trên mối quan hệ của thuộc tính giả \mathbf{A} với nhãn phân loại Y . Đối với từng loại phân bố được biểu diễn bởi một đồ thị nhận dạng nhân quả ở hình 3.4, Kaur và cộng sự đã chỉ ra được các ràng buộc độc lập chính xác thỏa mãn bởi \mathbf{X}_c ứng với từng loại thay đổi phân bố. Ở **pha I**, nhiệm vụ của thuật toán là tìm ra được những ràng buộc độc lập này. Ứng với từng loại thay đổi phân bố

được thể hiện qua các đồ thị nhận dạng (hình 3.4), Kaur và cộng sự đã đưa ra các ràng buộc độc lập có điều kiện thỏa mãn bởi \mathbf{X}_c ứng với từng trường hợp như sau:

1. *Independent*: $X_c \perp\!\!\!\perp A_{ind}$; $X_c \perp\!\!\!\perp E$; $X_c \perp\!\!\!\perp A_{ind} | Y$; $X_c \perp\!\!\!\perp A_{ind} | E$; $X_c \perp\!\!\!\perp A_{ind} | Y, E$
2. *Causal*: $X_c \perp\!\!\!\perp A_{cause} | Y$; $X_c \perp\!\!\!\perp E$; $X_c \perp\!\!\!\perp A_{cause} | Y, E$
3. *Confounded*: $X_c \perp\!\!\!\perp A_{conf}$; $X_c \perp\!\!\!\perp E$; $X_c \perp\!\!\!\perp A_{conf} | E$
4. *Selected*: $X_c \perp\!\!\!\perp A_{sel} | Y$; $X_c \perp\!\!\!\perp A_{sel} | Y, E$

Tất cả các ràng buộc được rút ra thỏa mãn cho đồ thị hình 3.3a. Tuy nhiên, nếu có sự tồn tại của mối tương quan giữa E và \mathbf{X}_c (đồ thị 3.3b), chỉ những ràng buộc điều kiện trên E là chính xác [10].

Việc xác định các ràng buộc độc lập trong đồ thị nhân quả là vô cùng cần thiết để xác định mối quan hệ giữa các biến. Một biến A nếu độc lập với B thì sự thay đổi của biến A sẽ không làm ảnh hưởng đến biến B và ngược lại. Điều đó cũng đúng khi ta xét các ràng buộc độc lập có điều kiện trong đồ thị. Ba đồ thị DAG đặc trưng cho độc lập có điều kiện được thể hiện như hình 3.5 [7]. Để có xác định được các ràng buộc độc lập có điều kiện chúng ta có thể sử dụng d-separation (Pearl, 2009).



Hình 3.5: Ba đồ thị DAG đặc trưng cho độc lập có điều kiện [7]

Định nghĩa d-separation [8]:

Nếu G là một đồ thị có hướng trong đó X, Y và S là các tập hợp đỉnh rời rạc. Một đường đi vô hướng p giữa X và Y được xem là bị chặn (d-separated) bởi S khi và chỉ khi có ít nhất một trong hai điều kiện sau thỏa mãn: (1) p chứa một chain $x \rightarrow z \rightarrow y$ hoặc một fork $x \leftarrow z \rightarrow y$ sao cho nút nằm giữa z thuộc S , (2) p chứa một collider $x \rightarrow z \leftarrow y$ sao cho

nút nằm giữa z không thuộc S và không có "con cháu" nào của z thuộc S . Nếu S chặn mọi đường đi từ một nút trong X đến một nút trong Y thì X và Y d-separated bởi S nghĩa là X và Y độc lập có điều kiện trên S ($X \perp\!\!\!\perp Y | S$).

Kaur và cộng sự đã sử dụng d-separation để rút ra các ràng buộc độc lập có điều kiện thỏa mãn bởi \mathbf{X}_c cho từng đồ thị DAG ứng với từng loại thay đổi ở hình 3.4.

Independent: Chúng ta có thể thấy ở hình 3.4a, chỉ có một đường đi duy nhất từ \mathbf{X}_c đến \mathbf{A}_{ind} và từ \mathbf{X}_c đến E , trên đường đi đó tồn tại collider \mathbf{X} nên chúng ta sẽ có các ràng buộc độc lập $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind}$ và $\mathbf{X}_c \perp\!\!\!\perp E$. Hơn nữa, điều kiện trên Y hoặc E không tạo ra kết nối cho đường đi từ \mathbf{X}_c đến \mathbf{A}_{ind} nên chúng ta sẽ rút ra được các ràng buộc: $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | Y$; $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | E$; $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | Y, E$. Từ đó, chúng ta rút được các ràng buộc trong trường hợp này:

$$\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind}; \mathbf{X}_c \perp\!\!\!\perp E; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | Y; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | E; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{ind} | Y, E$$

Causal: Chúng ta có thể thấy trong hình 3.4b, có 2 đường đi từ \mathbf{X}_c đến \mathbf{A}_{cause} . Vì không có collider trên đường $\mathbf{X}_c \rightarrow Y \rightarrow \mathbf{A}_{cause}$ nên $\mathbf{X}_c \not\perp\!\!\!\perp \mathbf{A}_{cause}$. Tuy nhiên, khi xét điều kiện trên Y , theo d-separation, chúng ta sẽ có đường đi $\mathbf{X}_c \rightarrow Y \rightarrow \mathbf{A}_{cause}$ là một đường bị chặn và đường $\mathbf{X}_c \rightarrow X \rightarrow \mathbf{A}_{cause}$ cũng là một đường bị chặn vì tồn tại collider \mathbf{X} và chúng ta không xét điều kiện trên X . Từ đó, chúng ta rút ra được ràng buộc độc lập có điều kiện $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{cause} | Y$. Ngoài ra, nếu xét điều kiện trên E cũng không tạo ra kết nối cho các đường đi từ \mathbf{X}_c đến \mathbf{A}_{cause} , nên chúng ta rút ra được ràng buộc $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{cause} | Y, E$. Chúng ta cũng rút được ràng buộc độc lập $\mathbf{X}_c \perp\!\!\!\perp E$ vì tất cả các đường đi từ \mathbf{X}_c đến E đều tồn tại collider (collider \mathbf{X} trong $\mathbf{X}_c \rightarrow \mathbf{X} \rightarrow \mathbf{A}_{cause} \rightarrow E$ và collider \mathbf{A}_{cause} trong $\mathbf{X}_c \rightarrow Y \rightarrow \mathbf{A}_{cause} \rightarrow E$). Trong trường hợp này các ràng

buộc được rút ra là:

$$\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{cause} \mid Y; \mathbf{X}_c \perp\!\!\!\perp E; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{cause} \mid Y, E$$

Confounded: Từ hình 3.4c, chúng ta có thể thấy 2 đường đi từ \mathbf{X}_c đến \mathbf{A}_{conf} đều chứa collider (collider \mathbf{X} trong $\mathbf{X}_c \rightarrow \mathbf{X} \rightarrow \mathbf{A}_{conf}$, collider Y trong $\mathbf{X}_c \rightarrow Y \rightarrow C \rightarrow \mathbf{A}_{conf}$) nên $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{conf}$. Hơn nữa, khi xét điều kiện trên E vẫn thỏa mãn vì E không là collider trên bất kì đường đi nào từ \mathbf{X}_c đến \mathbf{A}_{conf} nên $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{conf} \mid E$. Chúng ta cũng có được ràng buộc $\mathbf{X}_c \perp\!\!\!\perp E$ vì tất cả các đường từ \mathbf{X}_c đến \mathbf{A}_{conf} đều tồn tại collider (collider \mathbf{X} trong $\mathbf{X}_c \rightarrow \mathbf{X} \rightarrow \mathbf{A}_{conf} \rightarrow E$, collider Y trong $\mathbf{X}_c \rightarrow Y \rightarrow C \rightarrow \mathbf{A}_{conf} \rightarrow E$). Từ đó, các ràng buộc được rút ra trong trường hợp này là:

$$\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{conf}; \mathbf{X}_c \perp\!\!\!\perp E; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{conf} \mid E$$

Selected: Với dữ liệu quan sát được, chúng ta luôn xét điều kiện trên biến lựa chọn S . Từ đồ thị 3.4d, chúng ta có 2 đường đi từ \mathbf{X}_c đến \mathbf{A}_{sel} . Dù tồn tại collider \mathbf{X} trên đường đi $\mathbf{X}_c \rightarrow \mathbf{X} \rightarrow \mathbf{A}_{sel}$, nhưng ở đường đi còn lại $\mathbf{X}_c \rightarrow Y \rightarrow S \rightarrow \mathbf{A}_{sel}$ tồn tại collider S và chúng ta luôn xét điều kiện trên S nên theo d-separation đây là một đường đi không bị chặn suy ra $\mathbf{X}_c \not\perp\!\!\!\perp \mathbf{A}_{sel}$. Khi ta xét điều kiện trên Y , đường đi từ \mathbf{X}_c đến \mathbf{A}_{sel} bị chặn, nên chúng ta rút ra được $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{sel} \mid Y$. Hơn nữa, khi xét điều kiện trên E vẫn thỏa mãn vì E không là collider trên bất kì đường đi nào từ \mathbf{X}_c đến \mathbf{A}_{sel} suy ra $\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{sel} \mid Y, E$. Trong trường hợp này chúng ta rút ra được các điều kiện:

$$\mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{sel} \mid Y; \mathbf{X}_c \perp\!\!\!\perp \mathbf{A}_{sel} \mid Y, E$$

Việc rút ra các ràng buộc độc lập có điều kiện trên là cần thiết trong bài toán khái quát miền. Kaur và cộng sự [10] đã chứng minh được rằng đối với đồ thị nhân quả DAG \mathcal{G} trên $\langle \mathbf{X}_c, \mathbf{X}, \mathbf{A}, Y \rangle$ và bộ dữ liệu tương

ứng $(\mathbf{x}_i, \mathbf{a}_i, y_i)_{i=1}^n$ được sinh từ đồ thị này, các tính chất của $\mathbf{X}_c, \mathbf{X}, \mathbf{A}$ và Y tương tự như đồ thị ở hình 3.3 (ngoại trừ việc chia \mathbf{A} thành các tập con nhỏ hơn). Cho $\mathcal{P}_{\mathcal{G}}$ là tập hợp các phân bố tương ứng với đồ thị \mathcal{G} được tạo ra bằng cách thay đổi $P(\mathbf{A} | Y)$ nhưng không thay đổi $P(Y | \mathbf{X}_c)$. Khi đó, các ràng buộc độc lập có điều kiện thỏa mãn bởi \mathbf{X}_c là cần thiết cho một hàm dự đoán rủi ro (cross-entropy) trên $\mathcal{P}_{\mathcal{G}}$. Nghĩa là, nếu một hàm dự đoán cho Y không thỏa bất kỳ ràng buộc nào trong số các ràng buộc này thì tồn tại một phân bố dữ liệu $P' \in \mathcal{P}_{\mathcal{G}}$ sao cho độ rủi ro của hàm dự đoán sẽ cao hơn độ rủi ro của nó trong các phân bố khác.

Do đó, cho một đồ thị nhân quả DAG, sử dụng d-separation trên \mathbf{X}_c và các biến quan sát được, chúng ta có thể rút ra được các ràng buộc regularization chính xác để áp dụng cho ϕ (ϕ là biểu diễn của dữ liệu).

Tóm lại, ở **pha I** chúng ta cần rút ra các điều kiện độc lập chính xác thỏa mãn bởi \mathbf{X}_c và các biến quan sát được ứng với đồ thị DAG thể hiện quá trình sinh bộ dữ liệu mà chúng ta cần giải quyết. Nếu quá trình sinh dữ liệu thỏa mãn đồ thị canonical graph (hình 3.3), CACM yêu cầu người dùng xác định loại quan hệ cho từng thuộc tính và sử dụng các ràng buộc chính xác Kaur và cộng sự đã chỉ ra được ứng với từng loại thay đổi phân bố đã được trình bày ở trên. Đối với những bộ dữ liệu khác không thỏa mãn đồ thị canonical graph (hình 3.3), CACM yêu cầu một đồ thị nhân quả miêu tả quá trình sinh dữ liệu và sử dụng các bước sau để rút ra các ràng buộc độc lập. Cho \mathcal{V} là tập hợp các biến quan sát được trong trong đồ thị ngoại trừ Y , và C là tập hợp các ràng buộc.

1. Đối với từng biến quan sát được $V \in \mathcal{V}$, kiểm tra: Nếu (\mathbf{X}_c, V) d-separated, thêm $\mathbf{X}_c \perp\!\!\!\perp V$ vào C .
2. Nếu không, kiểm tra nếu (\mathbf{X}_c, V) d-separated điều kiện trên bất kỳ tập con Z nào của các biến quan sát được còn lại trong $\mathcal{Z} = \{Y\} \cup \mathcal{V} \setminus \{V\}$. Nếu (\mathbf{X}_c, V) d-separated điều kiện trên Z , thêm $\mathbf{X}_c \perp\!\!\!\perp V | Z$ vào C .

Pha II: Áp dụng Regularization penalty sử dụng các ràng buộc đã rút ra

Trong **pha II**, CACM áp dụng các ràng buộc đã được rút ra từ **pha I** để tính penalty trong kỹ thuật regularization và cộng vào hàm lỗi ERM tiêu chuẩn, $g_1, \phi = \operatorname{argmin}_{g_1, \phi} \ell(g_1(\phi(x)), y) + \operatorname{RegPenalty}$, với ℓ là hàm lỗi cross-entropy. Kaur và cộng sự sử dụng Maximum Mean Discrepancy (MMD) để tính penalty. Công thức tính regularization penalty ứng với đồ thị canonical (hình 3.3) $\mathcal{V} = \mathbf{A}$ tương ứng với từng loại thay đổi phân bố như sau:

$$\operatorname{RegPenalty}_{\mathbf{A}_{ind}} = \sum_{i=1}^{|E|} \sum_{j>i} \operatorname{MMD}(P(\phi(x) | a_{i,ind}), P(\phi(x) | a_{j,ind})) \quad (3.2)$$

$$\operatorname{RegPenalty}_{\mathbf{A}_{cause}} = \sum_{|E|} \sum_{y \in Y} \sum_{i=1}^{|\mathbf{A}_{cause}|} \sum_{j>i} \operatorname{MMD}(P(\phi(x) | a_{i,cause}, y), P(\phi(x) | a_{j,cause}, y)) \quad (3.3)$$

$$\operatorname{RegPenalty}_{\mathbf{A}_{conf}} = \sum_{|E|} \sum_{i=1}^{|\mathbf{A}_{conf}|} \sum_{j>i} \operatorname{MMD}(P(\phi(x) | a_{i,conf}), P(\phi(x) | a_{j,conf})) \quad (3.4)$$

$$\operatorname{RegPenalty}_{\mathbf{A}_{sel}} = \sum_{|E|} \sum_{y \in Y} \sum_{i=1}^{|\mathbf{A}_{conf}|} \sum_{j>i} \operatorname{MMD}(P(\phi(x) | a_{i,sel}, y), P(\phi(x) | a_{j,sel}, y)) \quad (3.5)$$

Trong \mathbf{A} bao gồm nhiều thuộc tính khác nhau, CACM sẽ tính penalty dựa vào loại thay đổi ứng với từng thuộc tính. Ví dụ, trong trường hợp *Independent* hoặc *Confounded* vì $A \in \mathbf{A}_{ind}$ hoặc $A \in \mathbf{A}_{conf}$ nên để $\phi(x) \perp\!\!\!\perp A$ nhóm tác giả sẽ tối thiểu sự khác biệt phân bố giữa $P(\phi(x) | A = a_i)$ và $P(\phi(x) | A = a_j)$ cho tất cả giá trị i, j của A . Tuy nhiên, vì cùng một ràng buộc được áp dụng trên E ($\phi(x) \perp\!\!\!\perp E$), nên việc áp dụng ràng buộc trên E (nếu có) sẽ hiệu quả về mặt thống kê vì có thể có nhiều giá trị liên quan chặt chẽ của A trong một miền xác định. Từ đó, Kaur và cộng sự áp dụng ràng buộc lên phân bố $P(\phi(x) | E = E_i)$ và $P(\phi(x) | E = E_j)$ nếu

E quan sát được (A có thể quan sát được hoặc không), nếu không thì họ sẽ áp dụng ràng buộc trên A [10]. Trong trường hợp *Causal* hoặc *Seleted*, vì $\phi(x) \not\perp\!\!\!\perp A$ mà $\phi(x) \perp\!\!\!\perp A|Y$ nên nhóm tác giả sẽ áp dụng ràng buộc lên phân bố $P(\phi(x) | A = a_i, Y = y)$ và $P(\phi(x) | A = a_j, Y = y)$. Trong các trường hợp nếu không thể xác định chắc chắn rằng \mathbf{X}_c độc lập với E thì Kaur và cộng sự sẽ xét điều kiện trên E .

Thuật toán CACM cho một đồ thị tổng quát \mathcal{G} :

Thuật toán 2 CACM

Đầu vào: Bộ dữ liệu $(\mathbf{x}_i, \mathbf{a}_i, y_i)_{i=1}^n$, đồ thị nhân quả DAG \mathcal{G}

Đầu ra: Hàm $g(x) = g_1(\phi(x)) : \mathbf{X} \rightarrow Y$

$\mathcal{A} \leftarrow$ tập hợp các biến quan sát được trong \mathcal{G} ngoại trừ Y, E

$C \leftarrow \{\}$ ▷ ánh xạ của A tới A_s

Pha I: Rút ra các ràng buộc độc lập chính xác

for $A \in \mathcal{A}$ **do**

if (X_c, A) d-separated **then**

$X_c \perp\!\!\!\perp A$ là một ràng buộc độc lập hợp lệ

else if (X_c, A) d-separated bởi bất kỳ tập con A_s của các biến quan sát còn lại trong $A \setminus \{\mathcal{A}\} \cup \{Y\}$ **then**

$X_c \perp\!\!\!\perp A | A_s$ là một ràng buộc độc lập hợp lệ

$C[A] = A_s$

Pha II: Áp dụng Regularization penalty sử dụng các ràng buộc đã rút ra

for $A \in \mathcal{A}$ **do**

if $X_c \perp\!\!\!\perp A$ **then**

$RegPenalty_A = \sum_{|E|} \sum_{i=1}^{|A|} \sum_{j>i} MMD(P(\phi(x) | A_i), P(\phi(x) | A_j))$

else if $A \in C$ **then**

$A_s = C[A]$

$RegPenalty_A = \sum_{|E|} \sum_{a \in A_s} \sum_{i=1}^{|A|} \sum_{j>i} MMD(P(\phi(x) | A_i, a), P(\phi(x) | A_j, a))$

$RegPenalty = \sum_{A \in \mathcal{A}} \lambda_A RegPenalty_A$

$g_1, \phi = \arg \min_{g_1, \phi} \ell(g_1(\phi(x)), y) + RegPenalty$

Xác định mối quan hệ giữa thuộc tính và nhãn

Như đã trình bày ở trên, nếu một bộ dữ liệu thỏa đồ thị 3.3, **pha I** của thuật toán CACM sẽ yêu cầu người dùng xác định mối quan hệ giữa các thuộc tính với nhãn của dữ liệu. Đối với dữ liệu quan sát được, chúng ta có thể xác định một thuộc tính sẽ thuộc vào A_{ind} hay $A_{\overline{ind}}$, vì A_{ind} sẽ độc lập với Y . Tuy nhiên, trong trường hợp của đồ thị 3.3b, mối quan hệ giữa A_{cause} , A_{conf} và A_{sel} với nhãn không thể xác định được từ dữ liệu quan sát được mà cần phải được nhập vào từ người dùng [10]. Đối với đồ thị nhãn quả đầy đủ, chúng ta cần phải xác định mối quan hệ giữa tất cả nút trong đồ thị. Tuy nhiên, ở đồ thị canonical của Kaur và cộng sự (hình 3.3), chúng ta chỉ thật sự quan tâm mối quan hệ giữa các thuộc tính với nhãn, mối quan hệ này có thể được xác định bởi kiến thức miền. Ví dụ, thuộc tính *region* trong bộ dữ liệu FMoW [3] sẽ có mối quan hệ *Confounded* với nhãn, hay thuộc tính *hospital* trong bộ dữ liệu Camelyon17 [2] sẽ có mối quan hệ *Independence* với nhãn vì sự khác biệt trong việc nhuộm tiêu bản hoặc thu nhận hình ảnh dẫn đến sự khác biệt về tiêu bản mô giữa các bệnh viện (*hospital*) [11]. Điều này xảy ra độc lập không có sự tương quan với nhãn nên *hospital* là một thuộc tính *Independence*. Tuy nhiên, nếu không thật sự biết được quá trình sinh ra dữ liệu, người dùng có thể xác định sai mối quan hệ giữa thuộc tính với nhãn. Ví dụ, nếu không biết được quá trình sinh ra bộ dữ liệu Waterbirds [13], người dùng có thể xác định nhầm mối quan hệ giữa thuộc tính *background* và nhãn là *Confounded*. Việc xác định sai mối quan hệ giữa thuộc tính với nhãn có thể sẽ ảnh hưởng đến hiệu suất của mô hình.

Chương 4

Thí nghiệm

Chương này trình bày các thí nghiệm để đánh giá mô hình mạng nơ-ron với phương pháp huấn luyện CACM đã được trình bày ở chương 3. Phần đầu tiên trình bày các thiết lập thí nghiệm, bao gồm các bộ dữ liệu và các kiến trúc cụ thể của mạng nơ-ron được sử dụng. Phần tiếp theo trình bày về các thí nghiệm. Thí nghiệm đầu tiên được thực hiện để so sánh kết quả cài đặt của khóa luận với kết quả của bài báo gốc. Thí nghiệm thứ hai so sánh giữa phương pháp CACM sử dụng ràng buộc đúng với phương pháp CACM sử dụng ràng buộc sai. Bài báo gốc chỉ thí nghiệm trên các bộ dữ liệu ảnh giả lập nên ở thí nghiệm thứ ba, phương pháp CACM được thử áp dụng cho bộ dữ liệu ảnh thực tế. Thí nghiệm thứ hai và thứ ba là thí nghiệm mở rộng ngoài bài báo gốc nhằm nhìn nhận rõ hơn về hiệu quả của phương pháp CACM.

4.1 Các thiết lập thí nghiệm

4.1.1 Các bộ dữ liệu

MNIST

MNIST là bộ dữ liệu ảnh viết tay của các chữ số từ 0 đến 9. Bài toán đặt ra là phân loại các ảnh theo hai lớp: lớp 0 là ảnh của chữ số bé hơn 5, lớp 1 là ảnh của chữ số lớn hơn hoặc bằng 5. Bộ dữ liệu này được sử dụng để tạo ra một bộ dữ liệu Colored MNIST [1] (hình 4.1a) có $\mathbf{A}_{cause} = color$ với $color$ là màu của chữ số trong ảnh (nghĩa là, màu của chữ số trong ảnh sẽ bị ảnh hưởng bởi nhãn tương ứng). Kế tiếp, bộ dữ liệu MNIST được sử dụng để tạo một bộ dữ liệu Rotated MNIST [4] (hình 4.1b) có $\mathbf{A}_{ind} = rotate$ với $rotate$ là độ xoay của chữ số trong ảnh (nghĩa là, độ xoay của chữ số trong ảnh sẽ độc lập với nhãn tương ứng). Trong đó, thuộc tính $rotate$ của các ảnh khác nhau giữa các miền. Cuối cùng, bộ dữ liệu MNIST được dùng để tạo ra bộ dữ liệu Col+Rot MNIST (hình 4.1c) với đa thuộc tính gồm $\mathbf{A}_{cause} = color$ và $\mathbf{A}_{ind} = rotate$, là kết hợp hai thuộc tính từ hai bộ dữ liệu trước. Mỗi bộ dữ liệu bao gồm 4 miền dữ liệu: $D_1, D_2 \in \mathcal{D}_{train}$, $D_3 \in \mathcal{D}_{val}$, $D_4 \in \mathcal{D}_{test}$. Bên cạnh đó, mỗi miền dữ liệu đều có 25% nhãn nhiễu.

Đối với bộ dữ liệu có $\mathbf{A}_{cause} = color$, mỗi miền dữ liệu D_i có tương quan $corr_i$ giữa $color$ với nhãn Y , và $corr_i = P(color = 0|Y = 0) = P(color = 1|Y = 1)$ với $color = 0$ là màu đỏ, $color = 1$ là màu xanh lá cây. Trong các thí nghiệm của khóa luận thì $corr_1 = 0.9$, $corr_2 = 0.8$, $corr_3 = corr_4 = 0.1$.

Đối với bộ dữ liệu có $\mathbf{A}_{ind} = rotate$, tất cả ảnh của mỗi miền dữ liệu D_i bị xoay một góc r_i . Trong các thí nghiệm của khóa luận thì $r_1 = 15^\circ$, $r_2 = 60^\circ$, $r_3 = r_4 = 90^\circ$.

		Train			Test
	0.9	0.8	0.1		
Y=0					
Y=1					

(a) Colored MNIST

		Train			Test
	15°	60°	90°		
Y=0					
Y=1					

(b) Rotated MNIST

		Train			Test
	(0.9, 15°)	(0.8, 60°)	(0.1, 90°)		
Y=0					
Y=1					

(c) Col+Rot MNIST

Hình 4.1: Một số mẫu của các bộ dữ liệu MNIST

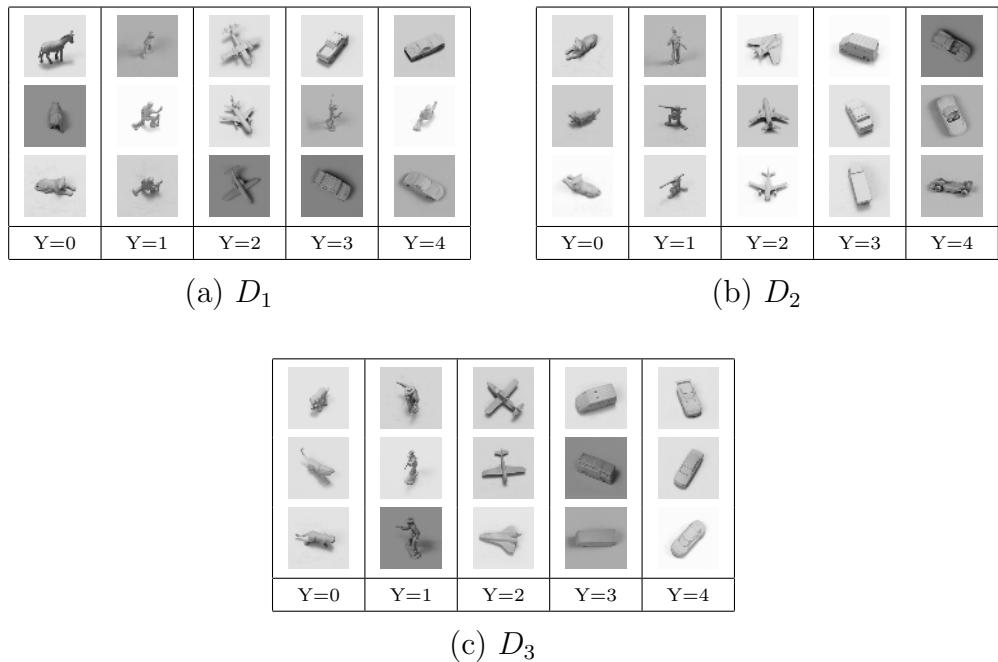
small NORB

small NORB là bộ dữ liệu ảnh của 50 món đồ chơi gồm 5 lớp: động vật bốn chân, nhân vật, máy bay, xe tải, xe ô tô. Các vật thể được chụp bởi 2 camera, dưới 6 điều kiện ánh sáng, 9 độ cao và 18 góc phương vị. Từ bộ dữ liệu small NORB, bộ dữ liệu Lighting small NORB được tạo ra có $\mathbf{A}_{cause} = lighting$ với *lighting* là điều kiện ánh sáng của ảnh (nghĩa là, điều kiện ánh sáng của ảnh bị ảnh hưởng bởi nhãn tương ứng), bộ dữ liệu Azimuth small NORB được tạo ra có $\mathbf{A}_{ind} = azimuth$ với *azimuth* là góc phương vị (nghĩa là, góc phương vị của vật thể được chụp độc lập với nhãn tương ứng). Thêm vào đó, bộ dữ liệu Light+Azi small NORB được tạo ra (hình 4.2) với đa thuộc tính gồm $\mathbf{A}_{cause} = lighting$ và $\mathbf{A}_{ind} = azimuth$. Mỗi bộ dữ liệu bao gồm 4 miền dữ liệu: $D_1, D_2 \in \mathcal{D}_{train}$, $D_3 \in \mathcal{D}_{val}$, $D_4 \in \mathcal{D}_{test}$. Bên cạnh đó, mỗi miền dữ liệu đều có 5% nhãn nhiễu.

Đối với bộ dữ liệu có $\mathbf{A}_{cause} = lighting$, giá trị của *lighting* là từ 0 tới 4, mỗi miền dữ liệu D_i có tương quan $corr_i$ giữa *lighting* với nhãn Y , và $corr_i = P(lighting = 0|Y = 0) = P(lighting = 1|Y = 1) = P(lighting = 2|Y = 2) = P(lighting = 3|Y = 3) = P(lighting = 4|Y = 4)$. Trong các

thí nghiệm của khóa luận thì $corr_1 = 0.9$, $corr_2 = 0.95$, $corr_3 = corr_4 = 0.0$.

Đối với bộ dữ liệu có $A_{ind} = azimuth$, tất cả ảnh của cùng một miền dữ liệu sẽ có giá trị azimuth thuộc về cùng một nhóm. Nghĩa là, tất cả ảnh của miền D_1 sẽ có giá trị azimuth thuộc về nhóm 1 (từ 0 đến 5), tất cả ảnh của miền D_2 sẽ có giá trị azimuth thuộc về nhóm 2 (từ 6 đến 11), tất cả ảnh của miền D_3 và miền D_4 sẽ có giá trị azimuth thuộc về nhóm 3 (từ 12 đến 17).



Hình 4.2: Một số mẫu của bộ dữ liệu Light+Azi small NORB

Camelyon17

Mô hình cho các ứng dụng y khoa thường chỉ được huấn luyện từ dữ liệu của một số nhỏ các bệnh viện và mong muốn có thể áp dụng được cho nhiều bệnh viện khác. Trong các ứng dụng mô bệnh học, khi nghiên cứu các tiêu bản mô dưới kính hiển vi sẽ có sự khác biệt giữa việc nhuộm tiêu bản hoặc thu thập hình ảnh về các tiêu bản mô giữa các bệnh viện. Điều này khiến cho mô hình sẽ bị giảm hiệu suất khi áp dụng lên bệnh viện mới [11]. Bộ dữ liệu Camelyon17 được điều chỉnh từ hình ảnh tiêu bản (whole-slide

images - WSIs) ở các phần hạch bạch huyết, thu được từ CAMELYON17 Challenge [2]. Khóa luận này sử dụng bộ dữ liệu Camelyon17 và thiết lập từ bài báo WILDS[11], với mục tiêu dự đoán chính xác sự xuất hiện của các mô khốiu trong các mảnh hình ảnh tiêu bản (WSIs) được lấy từ các bệnh viện mà không nằm trong bộ dữ liệu huấn luyện. Trong thiết lập của WILDS, Camelyon17 chứa 455 954 mảnh hình ảnh được tách từ 50 hình ảnh tiêu bản WSIs ung thư vú di căn ở các phần hạch bạch huyết từ 5 bệnh viện khác nhau. Trong đó:

- **Dữ liệu huấn luyện.** 302 436 mảnh từ 30 hình ảnh tiêu bản (WSIs) từ 3 bệnh viện khác nhau.
- **Dữ liệu Validation (OOD).** 34 904 mảnh từ 10 hình ảnh tiêu bản (WSIs) của bệnh viện thứ tư.
- **Dữ liệu kiểm tra.** 85 054 mảnh từ 10 hình ảnh tiêu bản (WSIs) của bệnh viện thứ năm.
- **Dữ liệu Validation (IID).** 33 560 mảnh từ 30 hình ảnh tiêu bản (WSIs) từ các bệnh viện giống dữ liệu huấn luyện.

Nhãn của mỗi mảnh sẽ có giá trị nhãn nhị phân thể hiện có sự xuất hiện của các mô khốiu hoặc không. Vì có sự khác biệt trong việc nhuộm tiêu bản hoặc thu thập hình ảnh dẫn đến nhiều biến thể của các tiêu bản mô giữa các bệnh viện (*hospital*) và điều này xảy ra độc lập không có sự tương quan với nhãn nên $A_{ind} = hospital$. Trong khóa luận này, dữ liệu huấn luyện được chia thành 3 miền ứng với từng bệnh viện trong dữ liệu huấn luyện $D_1, D_2, D_3 \in \mathcal{D}_{train}$, $D_4 \in \mathcal{D}_{val}$ và $D_5 \in \mathcal{D}_{test}$. Một số mẫu dữ liệu của bộ Camelyon17 được trình bày trong hình 4.3.

	Train			Val (OOD)	Test (OOD)
	$d = \text{Hospital 1}$	$d = \text{Hospital 2}$	$d = \text{Hospital 3}$	$d = \text{Hospital 4}$	$d = \text{Hospital 5}$
$y = \text{Normal}$					
$y = \text{Tumor}$					

Hình 4.3: Một số mẫu của bộ dữ liệu Camelyon17

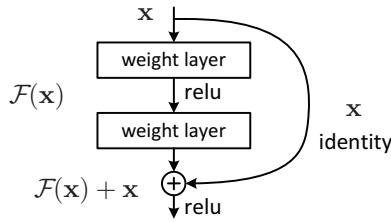
4.1.2 Các kiến trúc cụ thể của mạng nơ-ron

Trong các thí nghiệm với bộ dữ liệu MNIST và small NORB, mô hình được sử dụng là mô hình tuần tự gồm hai thành phần: bộ học đặc trưng để trích xuất đặc trưng của ảnh, bộ phân loại dựa vào các đặc trưng học được để dự đoán nhãn đầu ra. Bộ học đặc trưng dành cho bộ dữ liệu MNIST là một FCFFNN đã được trình bày ở mục 2.1.1, nó sẽ làm phẳng hình ảnh đầu vào và đưa qua mạng mã hóa để rút ra đặc trưng. Bộ học đặc trưng dành cho bộ dữ liệu small NORB là mạng ResNet-18 (được huấn luyện trước trên tập ImageNet). Đối với bộ dữ liệu Camelyon17, kiến trúc mô hình được sử dụng là DenseNet-121, một biến thể của DenseNet.

Sau đây là phần trình bày chi tiết các cấu trúc mạng nơ-ron được sử dụng trong các thí nghiệm.

Kiến trúc ResNet-18

ResNet là một kiến trúc CNN được đề xuất trong bài báo “Deep Residual Learning for Image Recognition” của nhóm tác giả Microsoft Research [9]. Đặc điểm của ResNet có các kết nối phần dư (residual connections),



Hình 4.4: Khối phần dư [9]

hay còn được gọi là đường tắt (shortcuts). Các kết nối này giúp giải quyết vấn đề gradient biến mất.

Thành phần chính của ResNet là các khối phần dư (residual blocks) (hình 4.4). Trong khối phần dư, có các đường tắt nhảy qua một hoặc vài lớp. Các khối phần dư cho phép xây dựng mạng nơ-ron sâu hơn với rất nhiều lớp (152 lớp trong ResNet-152) mà vẫn có thể được huấn luyện hiệu quả. Mạng nơ-ron sâu, nhiều lớp có thể học được các đặc trưng phức tạp hơn nên có thể hoạt động tốt hơn. Trong khối phần dư, đầu ra của một chuỗi các lớp sẽ được cộng thêm giá trị đầu vào. Đây gọi là ánh xạ danh tính (identity mapping), có thể được biểu diễn bằng công thức:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \mathbf{x} \quad (4.1)$$

ResNet-18 là phiên bản đơn giản của ResNet. ResNet-18 cần tương đối ít chi phí tính toán mà vẫn có hiệu suất tốt nên thường được sử dụng làm mô hình cơ sở. ResNet-18 bao gồm 18 lớp, có cấu trúc như sau:

- **Lớp tích chập đầu tiên.** Bao gồm 64 nhân có kích thước 7×7 và tổng hợp theo giá trị lớn nhất (max pooling).
- **Các khối phần dư.** Tại đây có 4 đoạn là conv2_x, conv3_x, conv4_x, conv5_x. Mỗi đoạn bao gồm 2 khối phần dư, mỗi khối phần dư bao gồm 2 lớp, mỗi lớp bao gồm các nhân có kích thước 3×3 . Số lượng nhân của mỗi lớp trong 4 đoạn lần lượt là 64, 128, 256, 512. Các khối phần dư này bao gồm tổng cộng 16 lớp.

- **Lớp cuối cùng.** Tại đây có tổng hợp theo giá trị trung bình (average pooling), lớp kết nối đầy đủ, hàm softmax.

ResNet-18 cũng như ResNet nói chung được sử dụng rỗng rãi cho các tác vụ phân lớp ảnh, nhận diện vật thể,... Trong thí nghiệm ở chương này, với bộ dữ liệu small NORB, kiến trúc mô hình được sử dụng là kiến trúc ResNet-18 đã được huấn luyện trước trên bộ dữ liệu ImageNet.

Kiến trúc DenseNet-121

Mạng tích chập dày đặc (Dense Convolutional Network - DenseNet) là một kiến trúc CNN được biết đến với độ hiệu quả và chính xác với bài toán phân lớp ảnh.

Các kết nối dày đặc là một đặc trưng của DenseNet. Trong các kiến trúc CNN truyền thống, mỗi lớp nhận đầu vào từ lớp trước và truyền kết quả qua lớp kế tiếp. Trong khi đó, một lớp trong DenseNet sẽ nhận đầu vào từ tất cả các lớp trước nó và truyền kết quả cho tất cả các lớp sau nó. Các kết nối dày đặt này giúp giải quyết vấn đề gradient biến mất, đồng thời cho phép các lớp sau tái sử dụng đặc trưng từ các lớp trước, và chúng cũng giúp giảm đáng kể số lượng tham số của mạng.

Kiến trúc DenseNet bao gồm các khối dày đặc (dense block), mỗi khối dày đặc bao gồm các lớp có kết nối dày đặc như nêu ở trên. Mỗi lớp trong khối dày đặc bao gồm chuẩn hóa hàng loạt (batch normalization), hàm kích hoạt ReLU, lớp tích chập. Các lớp này giúp biến đổi đặc trưng đầu giúp mạng học được các khuôn mẫu phức tạp trong dữ liệu. Theo sau mỗi khối dày đặc là một lớp chuyển tiếp (transition layer) được dùng để giảm kích thước của đặc trưng học được từ các khối dày đặc. Mỗi lớp chuyển tiếp bao gồm chuẩn hóa hàng loạt, lớp tích chập, lớp tổng hợp theo giá trị trung bình (average-pooling).

DenseNet-121 là một biến thể khá phổ biến của DenseNet, gồm 121 lớp. DenseNet-121 có kiến trúc mạng như sau:

- **Lớp tích chập.** Bao gồm các nhân có kích thước 7×7 và tổng hợp

theo giá trị lớn nhất (max-pooling).

- **Lớp dày đặc 1.** Bao gồm 6 lớp.
- **Lớp chuyển đổi 1.**
- **Lớp dày đặc 2.** Bao gồm 12 lớp.
- **Lớp chuyển đổi 2.**
- **Lớp dày đặc 3.** Bao gồm 24 lớp.
- **Lớp chuyển đổi 3.**
- **Lớp dày đặc 4.** Bao gồm 16 lớp.
- **Lớp phân loại.** Bao gồm lớp tổng hợp theo giá trị trung bình và một lớp kết nối đầy đủ sử dụng hàm kích hoạt softmax để tính toán xác suất đầu ra.

DenseNet-121 thường đạt được kết quả tốt hơn so sánh với các kiến trúc khác có số lượng tham số tương tự. Vì vậy, kiến trúc này thường được sử dụng cho các bài toán phân lớp ảnh.

4.2 Các thí nghiệm

4.2.1 So sánh kết quả cài đặt của khóa luận với bài báo gốc

Khóa luận đã cài đặt lại các thí nghiệm sử dụng hai bộ dữ liệu MNIST và small NORB dựa theo các mô tả của bài báo gốc [10] và DomainBed [6]. Với các bộ MNIST, kích thước của mỗi batch là 64, mô hình được huấn luyện 5000 bước. Còn với các bộ small NORB, kích thước mỗi batch là 128, mô hình được huấn luyện 2000 bước.

Với mỗi thí nghiệm, 20 bộ siêu tham số được tìm kiếm ngẫu nhiên và được sử dụng để huấn luyện mô hình. Quá trình này được lặp lại 3 lần với 3 random seeds khác nhau. Với mỗi random seed, một mô hình được huấn luyện có độ chính xác cao nhất sẽ được chọn ra. Sau đó kết quả của 3 mô hình này được dùng để tính kết quả trung bình và độ lệch chuẩn. Các siêu tham số được tìm kiếm ngẫu nhiên dựa trên các phân bố được nêu ra trong bảng 4.1. Các thí nghiệm này được chạy trên Kaggle với 2 GPU NVIDIA T4. Mã nguồn của thí nghiệm được viết dựa vào thư viện DoWhy¹ đối với phương pháp ERM và CACM; phương pháp CORAL được cài đặt dựa vào DomainBed [6].

Trường hợp Siêu tham số

FCFFNN	learning rate: [1e-2, 1e-3, 1e-4, 1e-5] dropout: 0
ResNet	learning rate: [1e-2, 1e-3, 1e-4, 1e-5] dropout: [0, 0.1, 0.5]
MNIST	weight decay: 0 generator weight decay: 0
khác MNIST	weight decay: $10^{\text{Uniform}(-6, -2)}$ generator weight decay: $10^{\text{Uniform}(-6, -2)}$
CORAL	λ : [0.1, 1, 10, 100]
CACM	λ : [0.1, 1, 10, 100] γ : [0.01, 0.0001, 0.000001]

Bảng 4.1: Các giá trị để điều chỉnh siêu tham số

Kết quả thí nghiệm

Các kết quả thí nghiệm trên bộ MNIST và small NORB được trình bày trong bảng 4.2. Các kết quả này cho thấy được sự tương quan giữa các

¹<https://github.com/py-why/dowhy>

Algo	MNIST			small NORB		
	color	Accuracy rotation	col+rot	lighting	azimuth	light+azi
Các kết quả trong bài báo gốc [10]						
ERM	30.9 \pm 1.6	61.9 \pm 0.5	25.2 \pm 1.3	65.5 \pm 0.7	78.6 \pm 0.7	64.0 \pm 1.2
CORAL	28.5 \pm 0.8	62.5 \pm 0.7	23.5 \pm 1.1	64.7 \pm 0.5	77.2 \pm 0.7	62.9 \pm 0.3
CACM	70.4 \pm 0.5	62.4 \pm 0.4	54.1 \pm 1.3	85.4 \pm 0.5	80.5 \pm 0.6	69.6 \pm 1.6
Các kết quả từ thí nghiệm của khóa luận						
ERM	30.8 \pm 0.6	61.6 \pm 0.2	25.5 \pm 1.0	70.8 \pm 1.7	77.6 \pm 0.7	64.1 \pm 4.3
CORAL	30.7 \pm 0.3	61.3 \pm 0.9	24.9 \pm 1.6	72.2 \pm 2.5	77.8 \pm 1.3	68.0 \pm 3.2
CACM	72.0 \pm 0.7	61.9 \pm 0.6	49.5 \pm 0.3	86.8 \pm 3.4	77.5 \pm 1.8	76.7 \pm 4.7

Bảng 4.2: Kết quả thí nghiệm trên các bộ dữ liệu MNIST và small NORB. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.

kết quả thí nghiệm trong bài báo gốc của phương pháp CACM [10] và kết quả thí nghiệm mà khóa luận này đã thực hiện. Sự khác biệt xảy ra nhiều nhất ở tập Light+Azi small NORB, điều này có thể là do quá trình xử lý dữ liệu khác biệt giữa khóa luận này với tác giả của CACM.

So sánh với mô hình được huấn luyện bằng phương pháp ERM và CORAL, kết quả của khóa luận và bài báo gốc đều cho thấy mô hình huấn luyện bằng phương pháp CACM có độ chính xác cao hơn trong đa số trường hợp, và vượt trội trong vài trường hợp.

Dầu tiên, trong trường hợp Colored MNIST và Lighting small NORB, CACM cho ra mô hình có độ chính xác cao vượt trội so với ERM lẫn CORAL trên cả thí nghiệm của khóa luận và bài báo gốc. Điều này cho thấy độ hiệu quả của ràng buộc mà CACM sử dụng đối với thay đổi *Causal* có trong dữ liệu.

Kế tiếp, với trường hợp Rotated MNIST, trong cả thí nghiệm của khóa luận lẫn bài báo gốc, các mô hình được huấn luyện bởi cả 3 phương pháp có độ chính xác gần nhau, không có sự chênh lệch quá nhiều. Điều này cho thấy có thể các ràng buộc của CACM đối với thay đổi *Independent* không cải thiện được hiệu suất của mô hình so với ERM hay CORAL. Tuy nhiên, với trường hợp Azimuth small NORB, kết quả của khóa luận và

bài báo gốc có sự khác biệt. Trong bài báo gốc, phương pháp CACM cho ra mô hình có độ chính xác tốt hơn so với ERM và CORAL khoảng 2%. Trong khi đó, kết quả của khóa luận cho thấy mô hình huấn luyện bởi cả 3 phương pháp cho ra độ chính xác không quá khác biệt.

Trong trường hợp Col+Rot MNIST với sự thay đổi trên đa thuộc tính, kết quả của khóa luận và bài báo gốc đều chỉ ra rằng phương pháp ERM và CORAL cho ra mô hình có hiệu suất thấp trong khi các ràng buộc của CACM giúp cho hiệu suất của mô hình tăng đáng kể (hơn 20%), mặc dù cũng không quá cao.

Đối với trường hợp Light+Azi small NORB với sự thay đổi trên đa thuộc tính, mô hình huấn luyện bởi phương pháp CACM có độ chính xác cao hơn vài phần trăm so với ERM và CORAL trên cả thí nghiệm của khóa luận lẫn kết quả của bài báo gốc. Bên cạnh đó, độ chính xác của mô hình huấn luyện bằng phương pháp CACM của khóa luận cao hơn khoảng 7% so với kết quả của bài báo gốc.

Hai trường hợp Col+Rot MNIST và Light+Azi small NORB cho thấy CACM hiệu quả hơn trong bài toán khái quát miền dưới sự thay đổi phân bố trên đa thuộc tính so với ERM và CORAL. Lí do chính là vì CACM áp dụng các ràng buộc khác nhau cho từng loại thay đổi khác nhau trong khi các phương pháp khác chỉ có ràng buộc cố định.

4.2.2 So sánh CACM sử dụng ràng buộc đúng với CACM sử dụng ràng buộc sai

Khi sử dụng phương pháp CACM, việc xác định đúng quan hệ giữa nhãn và thuộc tính là rất quan trọng. Xác định đúng các quan hệ này là cần thiết để sử dụng ràng buộc phù hợp trong quá trình huấn luyện nhằm nâng cao hiệu suất của mô hình. Dưới đây là kết quả của một số thí nghiệm sử dụng phương pháp CACM với ràng buộc sai. Đối với các trường hợp thay đổi trên đơn thuộc tính, tất cả các thuộc tính này được xác định là \mathbf{A}_{conf} dù các thuộc tính này vốn là \mathbf{A}_{cause} (*color*, *lighting*) hoặc \mathbf{A}_{ind}

(*rotation*, *azimuth*). Đối với thay đổi đa thuộc tính, tất cả các thuộc tính là \mathbf{A}_{cause} (*color*, *lighting*) sẽ được xác định là \mathbf{A}_{ind} và các thuộc tính là \mathbf{A}_{ind} (*rotation*, *azimuth*) được xác định là \mathbf{A}_{cause} .

Kết quả thí nghiệm

Kết quả của các thí nghiệm phương pháp CACM với ràng buộc sai trên các bộ dữ liệu MNIST và small NORB được trình bày trong bảng 4.3 và bảng 4.4.

Algo	MNIST		
	Accuracy		
	<i>color</i>	<i>rotation</i>	<i>col+rot</i>
CACM ràng buộc đúng	72.0 ±0.7	61.9 ±0.6	49.5 ±0.3
CACM ràng buộc sai	67.2 ±3.4	61.6 ±0.4	27.9 ±1.3

Bảng 4.3: Kết quả thí nghiệm CACM trên các bộ dữ liệu MNIST với ràng buộc sai. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.

Algo	small NORB		
	Accuracy		
	<i>lighting</i>	<i>azimuth</i>	<i>light+azi</i>
CACM ràng buộc đúng	86.8 ±3.4	77.5 ±1.8	76.7 ±4.7
CACM ràng buộc sai	85.3 ±3.4	78.1 ±1.4	66.1 ±5.0

Bảng 4.4: Kết quả thí nghiệm CACM trên các bộ dữ liệu small NORB với ràng buộc sai. Các kết quả ở đây là độ chính xác dự đoán (%) trên tập kiểm tra.

Đối với hai bộ dữ liệu Colored MNIST và Col+Rot MNIST, ràng buộc sai làm giảm đáng kể hiệu suất của mô hình. Đặc biệt với Col+Rot MNIST, vì cả hai thuộc tính đều được xác định sai, hiệu suất của mô hình giảm đến hơn 20%. Trong khi đó, ràng buộc sai trên bộ dữ liệu Rotated MNIST có hiệu suất giảm rất ít so với ràng buộc đúng.

Trên bộ dữ liệu Lighting small NORB, hiệu suất của mô hình giảm hơn 1% khi sử dụng ràng buộc sai. Tuy nhiên, ràng buộc sai giúp tăng hiệu suất của mô hình một ít trên bộ dữ liệu Azimuth small NORB. Có thể, trong quá trình sinh bộ dữ liệu, đã có một số yếu tố làm cho thuộc tính *azimuth* có quan hệ *Confounded* với nhãn, nên xác định *azimuth* là \mathbf{A}_{conf} là chính xác hơn. Đối với bộ dữ liệu Light+Azi small NORB, hiệu suất mô hình giảm khoảng 10% với ràng buộc sai.

Các kết quả thí nghiệm trên đa phần cho thấy hiệu suất của mô hình sẽ giảm khi sử dụng sai ràng buộc do xác định sai quan hệ giữa thuộc tính và nhãn. Do đó, việc hiểu quá trình sinh ra dữ liệu, xác định đúng quan hệ giữa thuộc tính và nhãn có vai trò quyết định khi sử dụng phương pháp CACM.

4.2.3 Áp dụng CACM cho dữ liệu ảnh thực tế

Khóa luận này đã áp dụng thuật toán CACM cho dữ liệu ảnh thực tế Camelyon17. Trong thí nghiệm này, kiến trúc được sử dụng là DenseNet-121 với các thiết lập tương tự như trong bài báo WILDS [11] như sử dụng SGD optimizer với momentum là 0.9, learning rate là 10^{-3} và batch size là 32. Điểm khác biệt là mô hình được huấn luyện với 1 epoch và kết quả được tổng hợp với 3 random seeds khác nhau. Trong thí nghiệm của khóa luận, thuật toán CACM được áp dụng với siêu tham số λ là 250 và γ là 0.01.

Kết quả thí nghiệm

Kết quả của thí nghiệm trên bộ dữ liệu Camelyon17 được trình bày trong bảng 4.5. Kết quả này cho thấy CACM đạt được kết quả tốt hơn ERM và CORAL trên cả 3 tập dữ liệu IID validation, OOD validation và kiểm tra. Trong tập dữ liệu kiểm tra, CACM đạt kết quả tốt hơn khoảng 1.7% so với ERM và khoảng 11.4% so với CORAL. Kết quả của thí nghiệm này cho thấy khả năng của CACM trong việc giúp mô hình

khái quát và chống chịu tốt với sự thay đổi phân bố trong dữ liệu ảnh y tế như Camelyon17.

Algo	Camelyon17		
	Accuracy		
	Validation (IID)	Validation (OOD)	Test (OOD)
ERM	92.0 \pm 0.3	91.2 \pm 1.5	81.5 \pm 2.6
CORAL	85.3 \pm 8.1	71.0 \pm 18.1	71.8 \pm 5.7
CACM	92.2 \pm 0.9	93.3 \pm 0.5	83.2 \pm2.6

Bảng 4.5: Kết quả thí nghiệm trên bộ dữ liệu Camelyon17. Các kết quả ở đây là độ chính xác dự đoán(%).

Chương 5

Tổng kết và hướng phát triển

5.1 Tổng kết

Khóa luận này đã tập trung tìm hiểu và cài đặt lại thành công phương pháp CACM từ bài báo “Modeling the data-generating process is necessary for out-of-distribution generalization” [10] và tiến hành thí nghiệm trên hai bộ dữ liệu MNIST và small NORB để so sánh với kết quả với bài báo gốc. Thí nghiệm của khóa luận đã có những kết quả khá tương quan so với bài báo gốc. Đối với những bộ dữ liệu dựa trên sự thay đổi đơn thuần tính (Colored MNIST, Rotated MNIST, Lighting small NORB, Azimuth small NORB), CACM có kết quả tương đương hoặc cao hơn so với các phương pháp ERM và CORAL. Còn đối với những bộ dữ liệu có sự thay đổi của đa thuần tính (Col+Rot MNIST và Light+Azi small NORB) CACM đạt được kết quả vượt trội. Ngoài ra, CACM cũng đạt được kết quả tốt hơn ERM và CORAL trên bộ dữ liệu ảnh thực tế Camelyon17. Việc áp dụng trên một bộ dữ liệu thực tế nhằm tăng độ tin cậy khi áp dụng CACM vào các ứng dụng thực tế.

Phương pháp CACM cho thấy sự quan trọng trong việc xác định mối quan hệ nhân quả trong quá trình sinh dữ liệu. Tuy nhiên, nó cũng có điểm yếu là không thể loại bỏ được ảnh hưởng của các thuộc tính tương quan mà không quan sát được. Ngoài ra, việc xác định được các ràng buộc

đúng sẽ ảnh hưởng lớn đến hiệu suất của mô hình. CACM yêu cầu người dùng phải xác định mối quan hệ giữa các thuộc tính và nhãn dựa vào quá trình sinh dữ liệu. Tuy nhiên, nếu không biết được quá trình sinh dữ liệu thì người dùng sẽ gặp khó khăn và sai sót trong việc xác định được các ràng buộc đúng.

5.2 Hướng phát triển

Thí nghiệm phương pháp CACM với các loại dữ liệu khác

Thí nghiệm của khóa luận và bài báo gốc đều được thực hiện trên các bộ dữ liệu ảnh. Do đó, các thí nghiệm trên các loại dữ liệu khác như văn bản, âm thanh,... có thể được thực hiện nhằm đánh giá thêm về độ hiệu quả của phương pháp CACM.

Thí nghiệm để đánh giá ưu, nhược điểm của phương pháp CACM

Do thời gian không đủ và độ phức tạp của phương pháp CACM nên khóa luận chưa tiến hành được các thí nghiệm nhằm đánh giá cụ thể điểm mạnh và điểm yếu của phương pháp này. Phương pháp CACM là một phương pháp khá mới, nếu đánh giá được điểm yếu thì ta sẽ có thể cải tiến phương pháp nhằm giải quyết tốt hơn bài toán dữ liệu có phân bố thay đổi.

Tài liệu tham khảo

Tiếng Anh

- [1] Arjovsky, Martin et al. “Invariant risk minimization”. In: *ArXiv* (2019). arXiv: 1907.02893.
- [2] Bandi, Peter et al. “From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge”. In: *IEEE transactions on medical imaging* 38.2 (2018), pp. 550–560.
- [3] Christie, Gordon et al. “Functional map of the world”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6172–6180.
- [4] Ghifary, Muhammad et al. “Domain generalization for object recognition with multi-task autoencoders”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2551–2559.
- [5] Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- [6] Gulrajani, Ishaan and Lopez-Paz, David. “In search of lost domain generalization”. In: *International Conference on Learning Representations*. 2020.
- [7] Guo, Ruocheng et al. “A survey of learning causality with data: Problems and methods”. In: *ACM Computing Surveys (CSUR)* 53.4 (2020), pp. 1–37.

- [8] Hayduk, Leslie et al. “Pearl’s D-separation: One more step into causal thinking”. In: *Structural Equation Modeling* 10.2 (2003), pp. 289–311.
- [9] He, Kaiming et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.
- [10] Kaur, Jivat Neet, Kiciman, Emre, and Sharma, Amit. “Modeling the data-generating process is necessary for out-of-distribution generalization”. In: *International Conference on Learning Representations*. 2022.
- [11] Koh, Pang Wei et al. “Wilds: A benchmark of in-the-wild distribution shifts”. In: *International conference on machine learning*. PMLR. 2021, pp. 5637–5664.
- [12] Li, Da et al. “Learning to generalize: Meta-learning for domain generalization”. In: *AAAI*. 2018.
- [13] Sagawa, Shiori et al. “Distributionally Robust Neural Networks”. In: *International Conference on Learning Representations*. 2020. URL: <https://openreview.net/forum?id=ryxGuJrFvS>.
- [14] Sun, Baochen, Feng, Jiashi, and Saenko, Kate. “Return of frustratingly easy domain adaptation”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.
- [15] Veitch, Victor et al. “Counterfactual invariance to spurious correlations in text classification”. In: *Advances in neural information processing systems* 34 (2021), pp. 16196–16208.
- [16] Wang, Jindong et al. “Generalizing to unseen domains: A survey on domain generalization”. In: *IEEE Transactions on Knowledge and Data Engineering* (2022).
- [17] Wu, Yu-chen and Feng, Jun-wen. “Development and application of artificial neural network”. In: *Wireless Personal Communications* 102 (2018), pp. 1645–1656.

- [18] Zhang, Hongyi et al. “mixup: Beyond empirical risk minimization”. In: *ICLR* (2017).