LỜI CẢM ƠN

MỤC LỤC

[LỜI CẢM ƠN 1](#_Toc135255588)

[MỤC LỤC 2](#_Toc135255589)

[DANH MỤC CÁC TỪ VIẾT TẮT 5](#_Toc135255590)

[DANH MỤC BẢNG BIỂU 6](#_Toc135255591)

[DANH MỤC HÌNH ẢNH 7](#_Toc135255592)

[DANH MỤC PHƯƠNG TRÌNH 8](#_Toc135255593)

[LỜI MỞ ĐẦU 9](#_Toc135255594)

[CHƯƠNG 1. TỔNG QUAN VỀ TRÍ TUỆ NHÂN TẠO 11](#_Toc135255595)

[1.1. Trí tuệ nhân tạo (Artificial intelligence) 11](#_Toc135255596)

[1.1.1. Lịch sử hình thành 11](#_Toc135255597)

[1.1.2. Khái niệm 11](#_Toc135255598)

[1.2. Học máy (Machine learning) 12](#_Toc135255599)

[1.2.1. Khái niệm về học máy 12](#_Toc135255600)

[1.2.2. Những nhiệm vụ học máy và kỹ thuật xử lý 13](#_Toc135255601)

[1.2.3. Các loại hệ thống học máy 14](#_Toc135255602)

[1.2.4. Những thách thức chính của học máy 18](#_Toc135255603)

[1.2.5. Kiểm tra và xác thực mô hình học máy 22](#_Toc135255604)

[1.2.6. Những thông số, thước đo để đánh giá mô hình học máy 25](#_Toc135255605)

[1.3. Học sâu (Deep learning) 30](#_Toc135255606)

[1.3.1. Khái niệm về học sâu 30](#_Toc135255607)

[1.3.2. Cách hoạt động của học sâu 31](#_Toc135255608)

[1.3.3. Những hình thức học máy khác học sâu 33](#_Toc135255609)

[1.3.4. Bối cảnh học máy hiện đại 37](#_Toc135255610)

[1.3.5. Lý do để lựa chọn học sâu 39](#_Toc135255611)

[1.4. Kết luận 40](#_Toc135255612)

[CHƯƠNG 2. MÔ HÌNH HỌC MÁY & CÔNG NGHỆ ỨNG DỤNG TRONG BÀI TOÁN PHÂN LOẠI HOA 41](#_Toc135255613)

[2.1. Kiến trúc tổng thể của ứng dụng 41](#_Toc135255614)

[2.2. Mạng thần kinh tích chập (Convolutional neural network - CNN) 42](#_Toc135255615)

[2.2.1. Khái niệm 42](#_Toc135255616)

[2.2.2. Lớp tích chập (Convolutional Layer) 42](#_Toc135255617)

[2.2.3. Lớp tổng hợp 47](#_Toc135255618)

[2.2.4. Kiến trúc mạng CNN 50](#_Toc135255619)

[2.3. Bài toán phát hiện đối tượng 51](#_Toc135255620)

[2.3.1. Khái niệm 51](#_Toc135255621)

[2.3.2. Các thuật toán phát hiện đối tượng 52](#_Toc135255622)

[2.4. Image Classification (Phân loại ảnh) 59](#_Toc135255623)

[2.4.1. Khái niệm 59](#_Toc135255624)

[2.4.2. Phân loại 59](#_Toc135255625)

[2.4.3. Các hướng giải quyết 61](#_Toc135255626)

[2.5. Các mô hình CNN được đào tạo trước (pre-trained CNN model) 61](#_Toc135255627)

[2.5.1. Học chuyển giao (Transfer learning) 61](#_Toc135255628)

[2.5.2. Mô hình VGG-19 63](#_Toc135255629)

[2.5.3. Mô hình InceptionV3 (GoogLeNet) 64](#_Toc135255630)

[2.5.4. Mô hình ResNet50 64](#_Toc135255631)

[2.5.5. Mô hình EfficientNet 67](#_Toc135255632)

[2.5.6. Tăng cường dữ liệu 68](#_Toc135255633)

[2.6. Mô hình hồi quy Logistic (Logistic Regression) 69](#_Toc135255634)

[2.6.1. Khái niệm 69](#_Toc135255635)

[2.6.2. Nguyên lý hoạt động 69](#_Toc135255636)

[2.6.3. Đào tạo và hàm chi phí 70](#_Toc135255637)

[2.6.4. Hồi quy Softmax 71](#_Toc135255638)

[2.7. Tensorflow & Keras 73](#_Toc135255639)

[2.8. Kết luận 73](#_Toc135255640)

[CHƯƠNG 3. XÂY DỰNG ỨNG DỤNG 74](#_Toc135255641)

[3.1. Tổ chức dữ liệu 74](#_Toc135255642)

[3.1.1. Tổ chức dữ liệu cho mô hình YOLO 74](#_Toc135255643)

[3.1.2. Tổ chức dữ liệu cho mô hình CNN 74](#_Toc135255644)

[3.2. Giao diện ứng dụng 74](#_Toc135255645)

[3.3. Những kết quả đạt được 74](#_Toc135255646)

[3.3.1. Kết quả đạt được của mô hình YOLO 74](#_Toc135255647)

[3.3.2. Kết quả đạt được của mô hình CNN 74](#_Toc135255648)

[3.4. Kết luận 74](#_Toc135255649)

[KẾT LUẬN VÀ KIẾN NGHỊ 75](#_Toc135255650)

[DANH MỤC TÀI LIỆU THAM KHẢO 76](#_Toc135255651)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **STT** | **Viết tắt** | **Ý nghĩa** |
| 1 | ML | Machine learning |
| 2 | OCR | Optical character recognition |
| 3 | CNN | Convolutional neural network |
| 4 | RNN | Recurrent neural networks |
| 5 | NLP | Natural language processing |
| 6 | NLU | Natural language understanding |
| 7 | RL | Reinforcement learning |
| 8 | AI | Artificial intelligence |
|  |  |  |
|  |  |  |
| 5 |  |  |

# DANH MỤC BẢNG BIỂU

# DANH MỤC HÌNH ẢNH

[**Hình 1. 1** Trí tuệ nhân tạo, học máy và học sâu 13](#_Toc135079819)

[**Hình 1. 2** Tập huấn luyện được dán nhãn để phân loại thư rác 15](#_Toc135079820)

[**Hình 1. 3** Vấn đề hồi quy: dự đoán một giá trị, được cung cấp một tính năng đầu vào 16](#_Toc135079821)

[**Hình 1. 4** Bộ đào tạo không được gắn nhãn cho học không giám sát 17](#_Toc135079822)

[**Hình 1. 5** Bộ đào tạo sau khi phân cụm 17](#_Toc135079823)

[**Hình 1. 7** Phát hiện bất thường 18](#_Toc135079824)

[**Hình 1. 8** Học tăng cường 19](#_Toc135079825)

[**Hình 1. 9** Tầm quan trọng của dữ liệu so với thuật toán 20](#_Toc135079826)

[**Hình 1. 10** Chính quy hóa làm giảm nguy cơ quá khớp 22](#_Toc135079827)

[**Hình 1. 11** Lựa chọn mô hình bằng cách sử dụng xác thực holdout 24](#_Toc135079828)

[**Hình 1. 12** Sử dụng tập phát triển – đào tạo 25](#_Toc135079829)

[**Hình 1. 13** Ma trận nhầm lẫn 26](#_Toc135079830)

[**Hình 1. 14** Ma trận nhầm lẫn minh họa các ví dụ về âm thực (trên cùng bên trái), dương giả (trên cùng bên phải), âm giả (dưới cùng bên trái) và dương thật (dưới cùng bên phải) 27](#_Toc135079831)

[**Hình 1. 15** Đường cong ROC vẽ tỷ lệ dương giả so với tỷ lệ dương thực cho tất cả các ngưỡng có thể 28](#_Toc135079832)

DANH MỤC PHƯƠNG TRÌNH

[**Phương trình 1. 1** Precision 24](#_Toc135079954)

[**Phương trình 1. 2** Recall 24](#_Toc135079955)

[**Phương trình 1. 3** F1 score 25](#_Toc135079956)

[**Phương trình 1. 4** False positive rate (FPR) 26](#_Toc135079957)

[**Phương trình 1. 5** Mean Squared Error (MSE) 27](#_Toc135079958)

[**Phương trình 1. 6** Mean Absolute Error **(**MAE) 27](#_Toc135079959)

[**Phương trình 1. 7** Root Mean Square Deviation (RMSD) 28](#_Toc135079960)

LỜI MỞ ĐẦU

Trong vài năm trở lại đây, trí tuệ nhân tạo (AI) là lĩnh vực được tất cả các quốc gia trên thế giới đầu tư, nghiên cứu và phát triển một cách cực kỳ mãnh mẽ. Theo Stanford, lượng tiền đầu tư tư nhân AI toàn cầu vào năm 2022 là 91,9 tỷ USD, nếu xét trong suốt thập kỷ qua, số tiền đầu tư tư nhân năm 2022 gấp 18 lần so với năm 2013. Tỷ lệ thuận với lượng tài chính đầu tư, AI đã tạo ra những bước đột phá mà trước đó là những thử thách khó khăn trong lịch sử:

* Phân loại ảnh gần mức con người
* Phiên âm giọng nói ở cấp độ con người
* Phiên âm chữ viết tay ở cấp độ con người
* Bản dịch máy được cải thiện đáng kể
* Chuyển đổi văn bản thành giọng nói được cải thiện đáng kể
* Các trợ lý kỹ thuật số như Google Assistant và Amazon Alexa, Chat GPT
* Lái xe tự động gần như ở cấp độ con người
* Kết quả tìm kiếm được cải thiện trên web
* Có khả năng trả lời các câu hỏi ngôn ngữ tự nhiên
* Chơi cờ vây siêu phàm

Ngoài ra trí tuệ nhân tạo còn có thể tự động sao chép hàng chục nghìn bản thảo cổ được lưu giữ trong Văn khố Tông đồ của Vatican, phát hiện và phân loại các bệnh thực vật trong các lĩnh vực sử dụng điện thoại thông minh đơn giản, hỗ trợ bác sĩ ung thư hoặc bác sĩ X quang giải thích dữ liệu hình ảnh y tế, dự đoán các thảm họa thiên nhiên như lũ lụt, bão hoặc thậm chí là động đất, v.v. Với mỗi cột mốc quan trọng, chúng ta đang tiến gần hơn đến thời đại AI hỗ trợ con người trong mọi hoạt động và mọi lĩnh vực như: khoa học, y học, sản xuất, năng lượng, giao thông vận tải, phát triển phần mềm, nông nghiệp và thậm chí cả sáng tạo nghệ thuật [?].

Sự phát triển của trí tuệ nhân tạo vừa tạo ra thách thức, vừa tạo cơ hội để Việt Nam có thể phần nào bắt kịp công nghệ với các nước trên thế giới, góp phần không nhỏ trong việc phát triển đất nước, phát triển con người, nâng cao chất lượng cuộc sống. Vì vậy, việc học tập, tìm hiểu, nghiên cứu và ứng dụng trí tuệ nhân tạo là việc rất quan trọng và cần thiết. Nhận thức được điều này, em đã lên ý tưởng, tìm hiểu và quyết định nghiên cứu, ứng dụng các phương pháp học máy, học sâu trong nhiệm vụ phân loại hình ảnh để có thể hiểu thêm những phần tuy nhỏ, nhưng rất quan trọng trong trí tuệ nhân tạo.

Đồ án này nhằm mục đích xây dựng một ứng dụng thám hoa (xác định tên các loài hoa) thông qua ảnh. Người dùng chỉ cần đưa một tấm ảnh bất kì về loài hoa mà họ muốn biết tên, sau đó ứng dụng sẽ xác định được trong ảnh có những loài hoa nào và tên của những loài hoa đó là gì.

Đồ án sẽ tập trung vào việc ứng dụng mô hình học máy (Hồi quy Logistic) và các mô hình học sâu (Yolo, Mạng nơ ron tích chập) cùng với thư viện xử lý ảnh (OpenCV) để tạo ra một ứng dụng xác định được tên các loài hoa với tỷ lệ chính xác cao.

Đồ án này được chia thành ba chương như sau:

CHƯƠNG 1: TỔNG QUAN VỀ TRÍ TUỆ NHÂN TẠO.

CHƯƠNG 2: CÁC MÔ HÌNH HỌC MÁY ỨNG DỤNG TRONG BÀI TOÁN PHÂN LOẠI HOA.

CHƯƠNG 3: XÂY DỰNG VÀ CÀI ĐẶT.

# TỔNG QUAN VỀ TRÍ TUỆ NHÂN TẠO

## Trí tuệ nhân tạo (Artificial intelligence)

### Lịch sử hình thành

Trí tuệ nhân tạo ra đời vào những năm 1950, khi một số ít những người tiên phong trong lĩnh vực khoa học máy tính non trẻ bắt đầu đặt câu hỏi liệu máy tính có thể được tạo ra để “suy nghĩ” hay không - một câu hỏi mà chúng ta vẫn đang khám phá ngày nay.

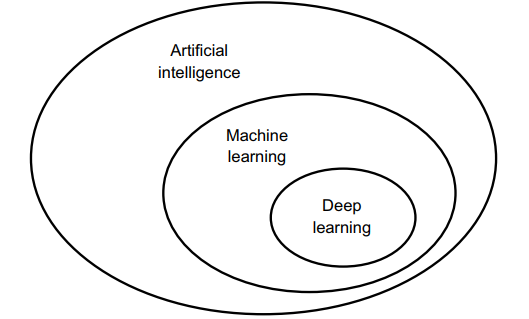
Trong khi nhiều ý tưởng cơ bản đã được hình thành từ nhiều năm và thậm chí nhiều thập kỷ trước đó, thì “trí tuệ nhân tạo” cuối cùng đã kết tinh thành một lĩnh vực nghiên cứu vào năm 1956, khi John McCarthy, lúc đó là Trợ lý Giáo sư Toán học trẻ tuổi tại Đại học Dartmouth, tổ chức một hội thảo mùa hè theo đề xuất sau:

*Nghiên cứu được tiến hành trên cơ sở phỏng đoán rằng mọi khía cạnh của học tập hoặc bất kỳ đặc điểm nào khác của trí thông minh về nguyên tắc có thể được mô tả chính xác đến mức có thể tạo ra một cỗ máy để mô phỏng nó. Một nỗ lực sẽ được thực hiện để tìm cách làm cho máy móc sử dụng ngôn ngữ, hình thành các khái niệm và khái niệm trừu tượng, giải quyết các loại vấn đề hiện dành cho con người và cải thiện bản thân. Chúng tôi nghĩ rằng một hoặc nhiều vấn đề này có thể đạt được tiến bộ đáng kể nếu một nhóm các nhà khoa học được lựa chọn cẩn thận cùng nhau giải quyết vấn đề đó trong một mùa hè.*

### Khái niệm

Nói một cách chính xác, AI có thể được mô tả là *nỗ lực tự động hóa các nhiệm vụ trí tuệ thường được thực hiện bởi con người*. Như vậy, AI là một lĩnh vực chung bao gồm học máy và học sâu, nhưng cũng bao gồm nhiều cách tiếp cận khác có thể không liên quan đến bất kỳ việc học nào. Trên thực tế, trong một thời gian khá dài, hầu hết các chuyên gia đều tin rằng trí tuệ nhân tạo ở cấp độ con người có thể đạt được bằng cách để các lập trình viên tạo thủ công một bộ quy tắc rõ ràng đủ lớn để thao tác kiến thức được lưu trữ trong cơ sở dữ liệu rõ ràng. Cách tiếp cận này được gọi là *AI tượng trưng*. Đó là mô hình thống trị trong AI từ những năm 1950 đến cuối những năm 1980, và nó đã đạt đến mức phổ biến cao nhất trong thời kỳ phát triển mạnh mẽ *hệ thống chuyên gia* của những năm 1980.

Dù AI tượng trưng phù hợp để giải quyết các vấn đề logic, được xác định rõ ràng, chẳng hạn như chơi cờ vua, nhưng lại khó tìm ra các quy tắc rõ ràng để giải quyết các vấn đề phức tạp, kém rõ ràng hơn, chẳng hạn như phân loại hình ảnh, nhận dạng giọng nói hoặc dịch ngôn ngữ tự nhiên. Một cách tiếp cận mới đã xuất hiện để thay thế vị trí của AI tượng trưng: *học máy*.



**Hình 1. 1** Trí tuệ nhân tạo, học máy và học sâu

## Học máy (Machine learning)

Trước đây học máy được coi là viễn tưởng, viễn vông, nhưng tại thời điểm hiện tại học máy không còn là khoa học viễn tưởng: hàng tỷ người sử dụng nó mỗi ngày. Nó đã thực sự tồn tại hàng thập kỷ trong một số ứng dụng chuyên biệt, chẳng hạn như nhận dạng ký tự quang học (OCR). Ứng dụng ML đầu tiên thực sự trở thành xu hướng, cải thiện cuộc sống của hàng trăm triệu người, đã chiếm lĩnh thế giới vào những năm 1990: bộ lọc thư rác. Nó thực sự đã học tốt đến mức chúng ta hiếm khi cần gắn cờ một email là thư rác nữa. Tiếp theo là hàng trăm ứng dụng ML hiện đang được sử dụng trong hàng trăm sản phẩm và tính năng mà con người sử dụng thường xuyên như: lời nhắc bằng giọng nói, dịch tự động, tìm kiếm hình ảnh, đề xuất sản phẩm, v.v. []

### Khái niệm về học máy

Học máy là khoa học lập trình máy tính để chúng có thể học hỏi từ dữ liệu. Khái niệm về học máy còn khá trừu tượng và chưa thực sự có một khái niệm nào chính xác và đầy đủ hoàn toàn để mô tả về học máy.

Một định nghĩa tổng quát:

**Học máy là lĩnh vực nghiên cứu cung cấp cho máy tính khả năng học hỏi mà không cần lập trình rõ ràng (Arthur Samuel, 1959).**

Một định nghĩa mang tính kỹ thuật:

**Một chương trình máy tính được cho là học hỏi từ kinh nghiệm E đối với một số nhiệm vụ T và một số phép đo hiệu suất P, nếu hiệu suất của nó trên T, được đo bằng P, cải thiện theo kinh nghiệm E (Tom Mitchell, 1997).**

Bộ lọc thư rác là một chương trình máy học, được cung cấp ví dụ về email rác (do người dùng gắn cờ) và ví dụ về email thông thường (không phải thư rác, còn được gọi là "ham"), có thể học cách gắn cờ thư rác. Các mẫu mà hệ thống sử dụng để học được gọi là *training set* (tập huấn luyện). Mỗi mẫu đào tạo được gọi là một *training instance* (hoặc *sample* (mẫu)). Một phần của hệ thống học máy học và đưa ra dự đoán được gọi là *model* (mô hình).

Trong trường hợp bộ lọc thư rác, nhiệm vụ T là gắn cờ thư rác cho các email mới, kinh nghiệm E là dữ liệu huấn luyện và thước đo hiệu suất P cần được xác định; ví dụ: chúng ta có thể sử dụng tỷ lệ email được phân loại chính xác. Thước đo hiệu suất cụ thể này được gọi là độ chính xác và nó thường được sử dụng trong các nhiệm vụ phân loại.

### Những nhiệm vụ học máy và kỹ thuật xử lý

Phân tích hình ảnh của sản phẩm trên dây chuyền sản xuất để tự động phân loại chúng : Đây là phân loại hình ảnh, thường được thực hiện bằng cách sử dụng mạng thần kinh tích chập (CNN) hoặc máy biến áp.

Phát hiện khối u khi quét não: Đây là phân đoạn hình ảnh ngữ nghĩa, trong đó mỗi pixel trong hình ảnh được phân loại (khi chúng ta muốn xác định vị trí và hình dạng chính xác của khối u), thường sử dụng CNN hoặc máy biến áp.

Tự động phân loại các bài báo: Đây là xử lý ngôn ngữ tự nhiên (NLP), và cụ thể hơn là phân loại văn bản, có thể được giải quyết bằng cách sử dụng mạng thần kinh hồi quy (RNN), CNN và chủ yếu là máy biến áp.

Tự động gắn cờ các bình luận xúc phạm trên các diễn đàn thảo luận: Đây cũng là phân loại văn bản, sử dụng cùng các công cụ NLP.

Tự động tóm tắt các tài liệu dài: Đây là một nhánh của NLP được gọi là tóm tắt văn bản, và sử dụng các công cụ tương tự.

Tạo một chatbot hoặc trợ lý cá nhân: Điều này liên quan đến nhiều thành phần NLP, bao gồm hiểu ngôn ngữ tự nhiên (NLU) và các mô-đun trả lời câu hỏi.

Dự báo doanh thu của công ty trong năm tới, dựa trên nhiều chỉ số hiệu suất: Đây là nhiệm vụ hồi quy (nghĩa là dự đoán các giá trị) có thể được giải quyết bằng cách sử dụng bất kỳ mô hình hồi quy nào, chẳng hạn như mô hình hồi quy tuyến tính hoặc hồi quy đa thức, hồi quy máy véc tơ hỗ trợ, rừng ngẫu nhiên hồi quy hoặc mạng nơ ron nhân tạo. Nếu cần xem xét các chuỗi số liệu hiệu suất trong quá khứ, có thể sử dụng RNN, CNN hoặc máy biến áp.

Làm cho ứng dụng phản ứng với lệnh thoại: Đây là nhận dạng giọng nói, yêu cầu xử lý các mẫu âm thanh: vì chúng là các chuỗi dài và phức tạp nên chúng thường được xử lý bằng RNN, CNN hoặc máy biến áp.

Phát hiện gian lận thẻ tín dụng: Đây là phát hiện bất thường, có thể được giải quyết bằng cách sử dụng rừng cô lập, mô hình hỗn hợp Gaussian hoặc bộ mã hóa tự động.

Phân khúc khách hàng dựa trên giao dịch mua hàng của họ để có thể thiết kế chiến lược tiếp thị khác nhau cho từng phân khúc: Đây là phân cụm, có thể đạt được bằng cách sử dụng k-mean, DBSCAN, v.v..

Biểu diễn tập dữ liệu phức tạp, nhiều chiều trong một sơ đồ rõ ràng và dễ hiểu: Đây là trực quan hóa dữ liệu, thường liên quan đến các kỹ thuật giảm kích thước.

Đề xuất một sản phẩm mà khách hàng có thể quan tâm, dựa trên các lần mua trước: Đây là một hệ thống đề xuất. Một cách tiếp cận là cung cấp các giao dịch mua trong quá khứ (và các thông tin khác về khách hàng) cho một mạng thần kinh nhân tạo và khiến nó đưa ra giao dịch mua tiếp theo có nhiều khả năng xảy ra nhất. Mạng lưới thần kinh này thường được đào tạo dựa trên các chuỗi mua hàng trong quá khứ của tất cả các khách hàng.

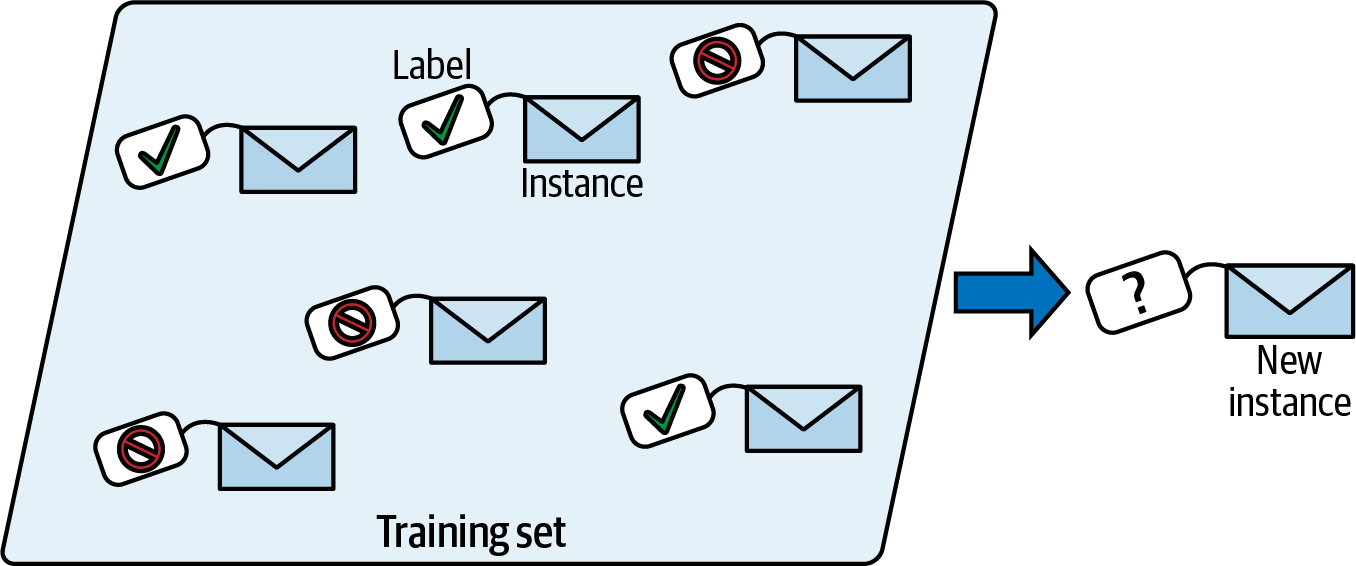
Xây dựng bot thông minh cho trò chơi: Điều này thường được giải quyết bằng cách sử dụng phương pháp học tăng cường (RL), là một nhánh của máy học huấn luyện các tác nhân (chẳng hạn như bot) chọn các hành động sẽ tối đa hóa phần thưởng của họ theo thời gian (ví dụ: bot có thể nhận được phần thưởng mỗi khi người chơi mất một số điểm sinh mệnh), trong một môi trường nhất định (chẳng hạn như trò chơi). Chương trình AlphaGo nổi tiếng đã đánh bại nhà vô địch thế giới trong trò chơi cờ vây được xây dựng bằng RL.

### Các loại hệ thống học máy

Dựa vào số lượng và loại hình giám sát mà các hệ thống ML nhận được trong quá trình đào tạo, gồm những loại chính: học có giám sát, học không giám sát và học tăng cường.

#### a. Học có giám sát

Trong học có giám sát, tập huấn luyện mà chúng ta cung cấp cho thuật toán bao gồm các giải pháp mong muốn, được gọi là nhãn.

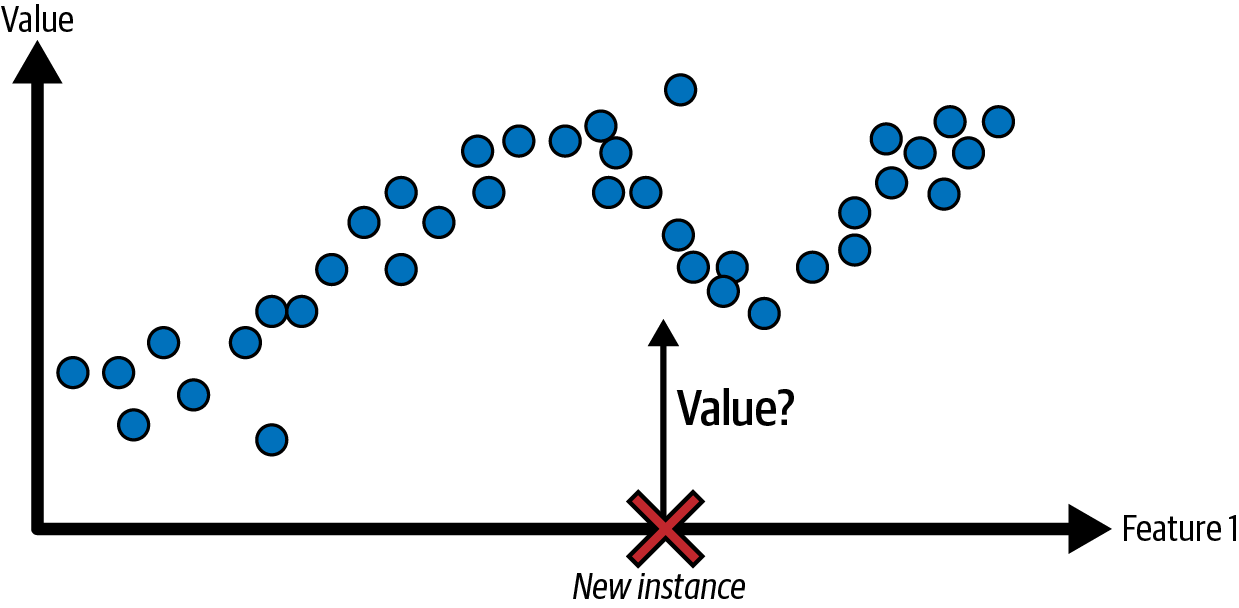


##### **Hình 1. 2** Tập huấn luyện được dán nhãn để phân loại thư rác

Một nhiệm vụ học tập có giám sát điển hình là *phân loại*. Bộ lọc thư rác là một ví dụ điển hình về điều này: nó được đào tạo với nhiều email mẫu cùng với *lớp* của chúng (thư rác hoặc không phải thư rác), và nó phải học cách phân loại email mới.

Một nhiệm vụ điển hình khác là dự đoán một giá trị số *mục tiêu*, chẳng hạn như giá của một chiếc ô tô, được cung cấp một tập hợp các *thông số* (số dặm, tuổi, nhãn hiệu, v.v.). Loại nhiệm vụ này được gọi là *hồi quy*. Để đào tạo hệ thống, chúng ta cần cung cấp cho nó nhiều ví dụ về ô tô, bao gồm cả thông số và mục tiêu của chúng (tức là giá của chúng).

Một số mô hình hồi quy cũng có thể được sử dụng để phân loại và ngược lại. Ví dụ: *hồi quy logistic* thường được sử dụng để phân loại, vì nó có thể xuất ra một giá trị tương ứng với xác suất thuộc về một lớp nhất định (ví dụ: 20% khả năng là thư rác).



##### **Hình 1. 3** Vấn đề hồi quy: dự đoán một giá trị với một tính năng đầu vào

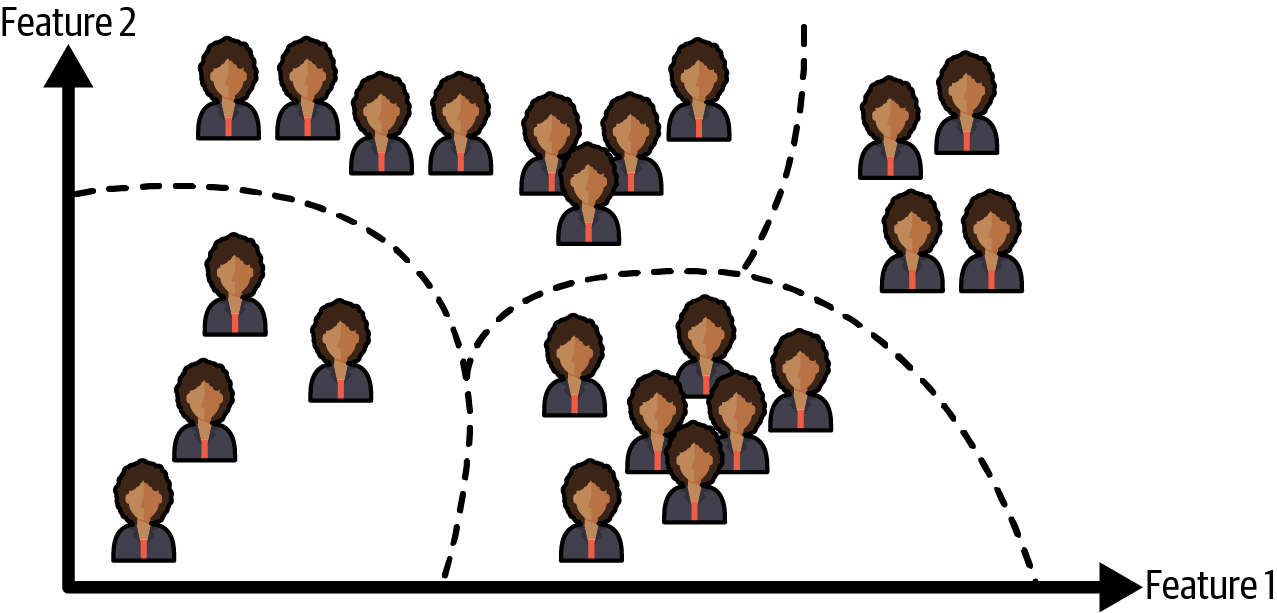
Các từ *mục tiêu* và *nhãn* thường được coi là từ đồng nghĩa trong học có giám sát, nhưng *mục tiêu* phổ biến hơn trong các tác vụ hồi quy và *nhãn* phổ biến hơn trong các tác vụ phân loại. *Các đặc trưng* thường được gọi là *yếu tố dự đoán* hoặc *thuộc tính*.

#### b. Học không giám sát

Trong học không giám sát, dữ liệu huấn luyện không được gắn nhãn. Hệ thống cố gắng học mà không có người dạy. Ví dụ: giả sử chúng ta dữ liệu về khách truy cập trang web của mình. Ta sẽ chạy thuật toán *phân cụm* (Thuật toán sẽ dựa trên độ tương tự giữa các ví dụ để xếp chúng thành những nhóm, mỗi nhóm gồm các ví dụ tương tự nhau) để phát hiện các nhóm khách truy cập tương tự. Chúng ta không cần cho thuật toán biết khách truy cập thuộc nhóm nào: thuật toán sẽ tự tìm thấy các kết nối đó. Ví dụ: có thể nhận thấy rằng 40% khách truy cập là thanh thiếu niên yêu thích truyện tranh và 20% là người lớn thích khoa học viễn tưởng.



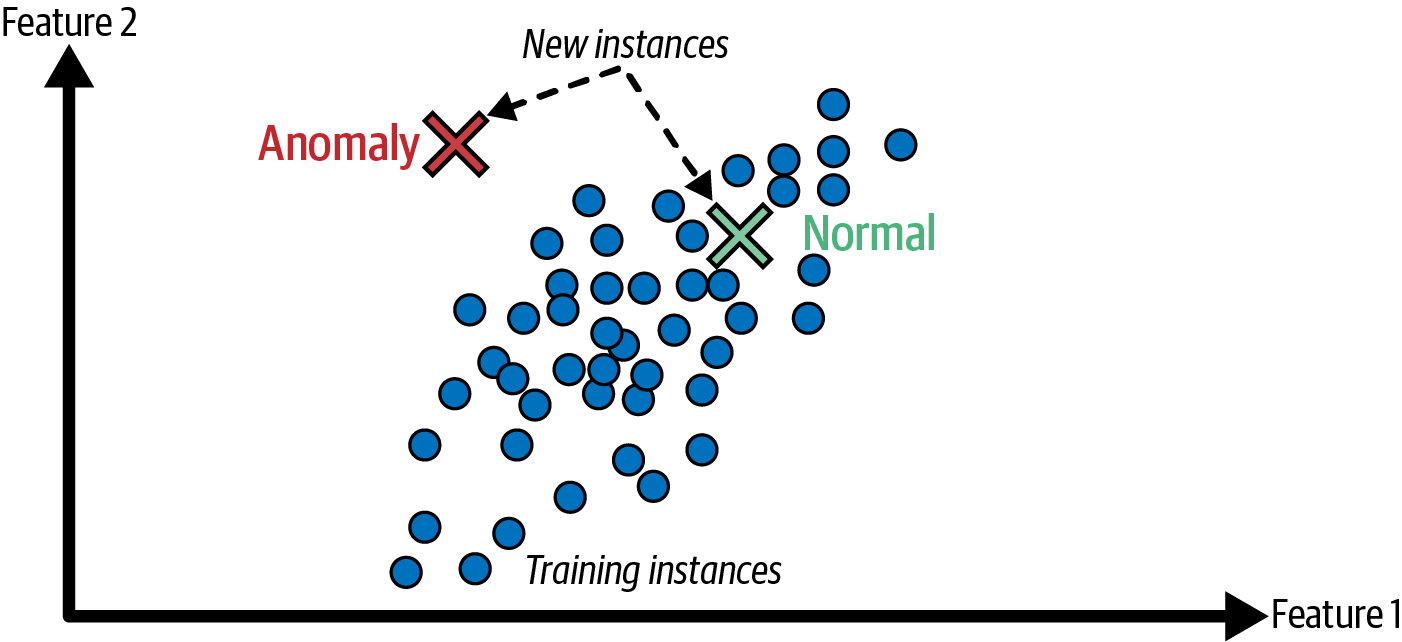
##### **Hình 1. 4** Bộ đào tạo không được gắn nhãn cho học không giám sát



##### **Hình 1. 5** Bộ đào tạo sau khi phân cụm

Một nhiệm vụ liên quan là *giảm kích thước*, trong đó mục tiêu là đơn giản hóa dữ liệu mà không làm mất quá nhiều thông tin. Một cách để làm điều này là hợp nhất một số tính năng có liên quan thành một. Ví dụ: quãng đường đi được của một chiếc ô tô có thể tương quan chặt chẽ với tuổi của nó, vì vậy, thuật toán giảm kích thước sẽ hợp nhất chúng thành một đặc điểm thể hiện mức độ hao mòn của ô tô. Điều này được gọi là *trích xuất tính năng*.

Một nhiệm vụ học không giám sát quan trọng khác là *phát hiện bất thường* - ví dụ: phát hiện các giao dịch thẻ tín dụng bất thường để ngăn chặn gian lận, phát hiện các lỗi sản xuất hoặc tự động loại bỏ các giá trị ngoại lệ khỏi tập dữ liệu trước khi đưa nó vào một thuật toán học tập khác. Hệ thống được hiển thị hầu hết các trường hợp bình thường trong quá trình đào tạo, vì vậy nó học cách nhận ra chúng; sau đó, khi nhìn thấy một trường hợp mới, nó có thể cho biết trường hợp mới trông giống như một trường hợp bình thường hay là một trường hợp bất thường. Một nhiệm vụ tương tự là *phát hiện tính mới*: nó nhằm mục đích phát hiện các trường hợp mới trông khác với tất cả các trường hợp trong tập huấn luyện.

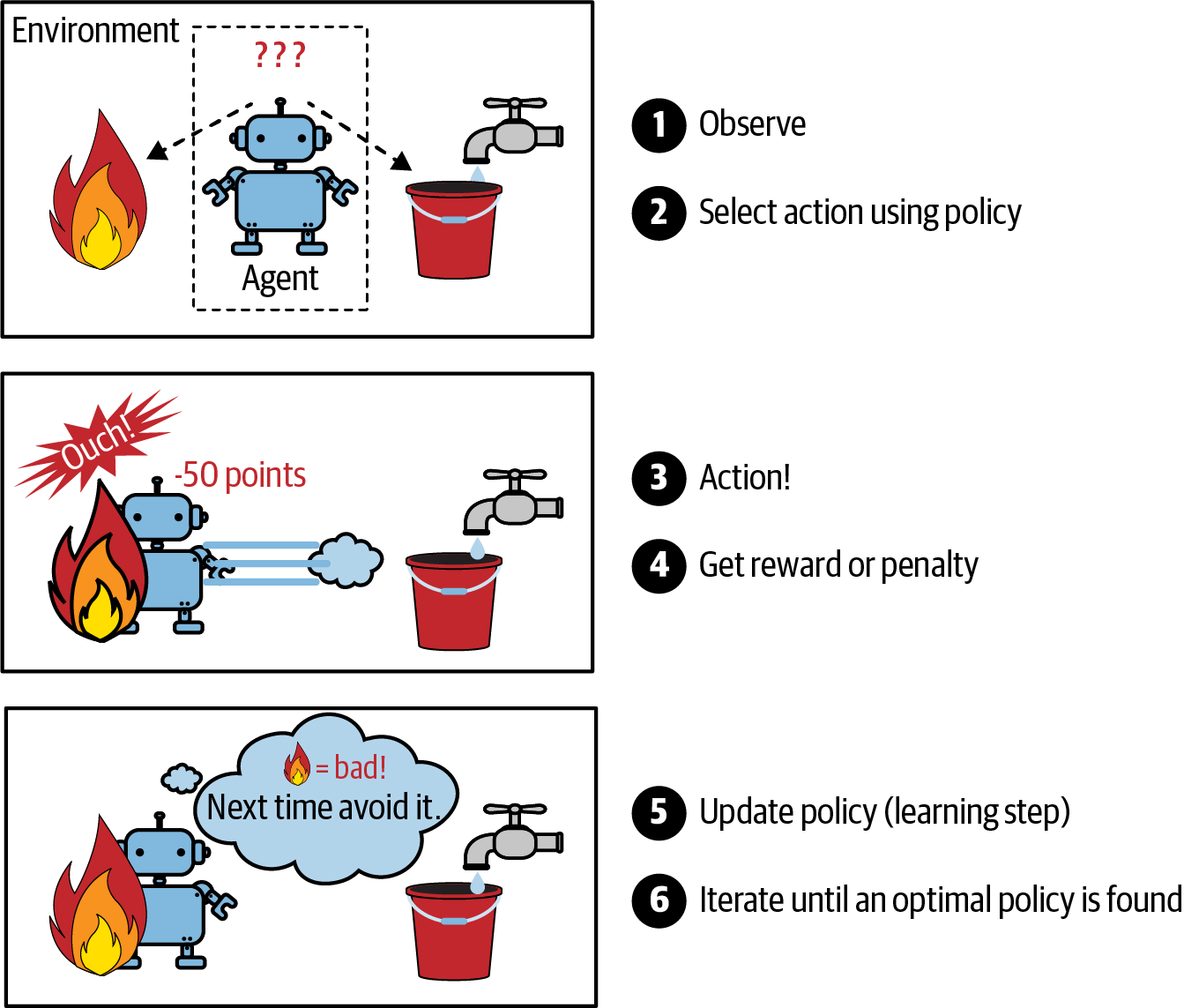


##### **Hình 1. 6** Phát hiện bất thường

Một nhiệm vụ không giám sát phổ biến khác là *học quy tắc kết hợp*, trong đó mục tiêu là đào một lượng lớn dữ liệu và khám phá các mối quan hệ giữa các thuộc tính. Ví dụ, giả sử ta làm giám đốc bán hàng của một rạp chiếu phim, việc chạy quy tắc kết hợp trên nhật ký bán hàng cho ta biết rằng những người mua vé xem phim cũng có xu hướng mua bỏng ngô. Vì vậy, ta có thể đặt các mục này gần nhau.

#### b. Học tăng cường

*Học tăng cường* rất khác so với các hệ thống học trước đó. Hệ thống học tập, được gọi là *tác nhân* trong ngữ cảnh này, có thể quan sát môi trường, lựa chọn và thực hiện các hành động, đồng thời nhận lại *phần thưởng* (hoặc *hình phạt* dưới dạng phần thưởng tiêu cực). Sau đó, nó phải tự tìm hiểu đâu là chiến lược tốt nhất, được gọi là *chính sách*, để nhận được nhiều phần thưởng nhất theo thời gian. Một chính sách xác định hành động nào mà tác nhân nên chọn khi ở trong một tình huống nhất định.



##### **Hình 1. 7** Học tăng cường

Ví dụ, nhiều robot thực hiện các thuật toán học tăng cường để học cách đi bộ. Chương trình AlphaGo của DeepMind cũng là một ví dụ điển hình về học tăng cường: nó đã gây chú ý vào tháng 5 năm 2017 khi đánh bại Ke Jie, kỳ thủ xếp hạng số một thế giới vào thời điểm đó, trong trò chơi cờ vây. Nó đã học được chính sách chiến thắng của mình bằng cách phân tích hàng triệu trò chơi và sau đó chơi nhiều trò chơi chống lại chính nó. Việc học đã bị tắt trong các trận đấu với nhà vô địch; AlphaGo chỉ đang áp dụng chính sách mà nó đã học được.

### Những thách thức chính của học máy

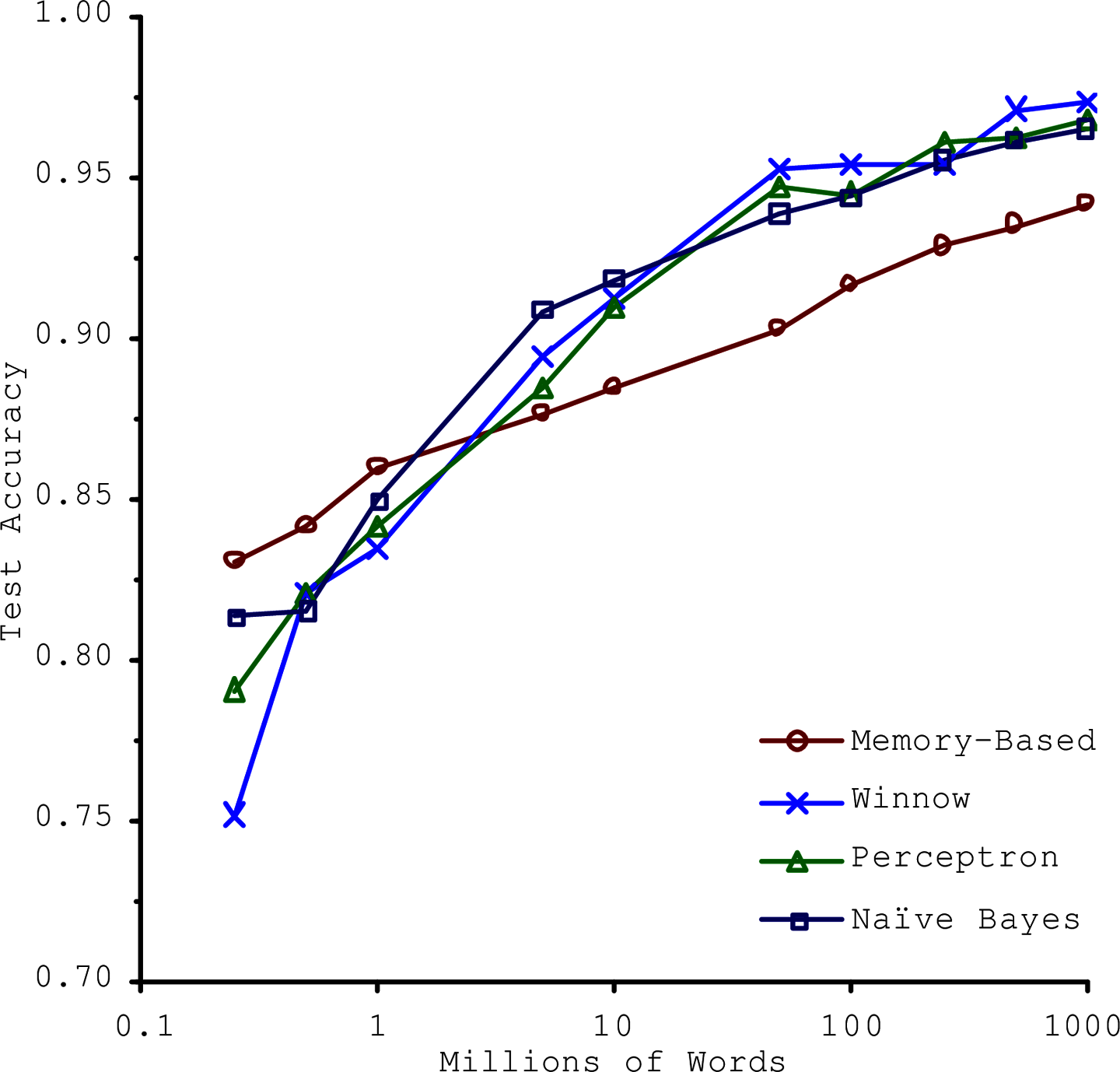
#### a. Không đủ số lượng dữ liệu đào tạo

Học máy cần rất nhiều dữ liệu để hầu hết các thuật toán học máy hoạt động bình thường. Ngay cả đối với các vấn đề rất đơn giản, thường cần hàng nghìn mẫu và đối với các vấn đề phức tạp như nhận dạng hình ảnh hoặc giọng nói, có thể cần hàng triệu mẫu (trừ khi có thể sử dụng lại các phần của mô hình hiện có).

Trong một bài báo nổi tiếng xuất bản năm 2001, các nhà nghiên cứu của Microsoft là Michele Banko và Eric Brill đã chỉ ra rằng các thuật toán học máy rất khác nhau, kể cả những thuật toán khá đơn giản, hoạt động gần như giống hệt nhau đối với một vấn đề phức tạp về định hướng ngôn ngữ tự nhiên một khi chúng được cung cấp đủ dữ liệu.

Như các tác giả đã nói, “những kết quả này gợi ý rằng chúng ta có thể muốn xem xét lại sự đánh đổi giữa việc dành thời gian và tiền bạc cho việc phát triển thuật toán so với việc dành nó cho việc phát triển kho dữ liệu”

Ý tưởng rằng dữ liệu quan trọng hơn thuật toán cho các vấn đề phức tạp đã được phổ biến rộng rãi hơn bởi Peter Norvig et al. trong một bài báo có tiêu đề “**The Unreasonable Effectiveness of Data**”, xuất bản năm 2009. Tuy nhiên, cần lưu ý rằng các tập dữ liệu vừa và nhỏ vẫn còn rất phổ biến và không phải lúc nào cũng dễ dàng hoặc rẻ để có thêm dữ liệu đào tạo, vì vậy không nên từ bỏ các thuật toán.



##### **Hình 1. 8** Tầm quan trọng của dữ liệu so với thuật toán

#### b. Dữ liệu đào tạo không đại diện

Để khái quát hóa tốt, điều quan trọng là dữ liệu đào tạo phải đại diện cho các trường hợp mới mà chúng ta muốn khái quát hóa.

Điều quan trọng là sử dụng một tập huấn luyện đại diện cho các trường hợp chúng ta muốn khái quát hóa. Điều này thường khó: nếu mẫu quá nhỏ, sẽ gặp *nhiễu lấy mẫu* (nghĩa là dữ liệu không mang tính đại diện do ngẫu nhiên), nhưng ngay cả những mẫu rất lớn cũng có thể không mang tính đại diện nếu phương pháp lấy mẫu có sai sót. Điều này được gọi là *thiên vị lấy mẫu*.

#### c. Dữ liệu kém chất lượng

Nếu dữ liệu đào tạo chứa đầy lỗi, ngoại lệ và nhiễu (ví dụ: do phép đo kém chất lượng), thì hệ thống sẽ khó phát hiện các mẫu cơ bản hơn, do đó hệ thống ít có khả năng hoạt động tốt hơn. Việc dành thời gian dọn dẹp dữ liệu đào tạo rất quan trọng và cần thiết. Dưới đây là những cách tiền xử lý dữ liệu:

* Nếu một số trường hợp rõ ràng là ngoại lệ, chỉ cần loại bỏ chúng hoặc sửa lỗi theo cách thủ công.
* Nếu một số trường hợp thiếu một số tính năng, ta phải quyết định xem có nên bỏ qua thuộc tính này hoàn toàn hay không, bỏ qua các trường hợp này, điền vào các giá trị còn thiếu (ví dụ: bằng giá trị trung vị, trung bình) hoặc huấn luyện một mô hình có tính năng này và một mô hình không có tính năng này.

#### d. Tính năng không liên quan

Như câu nói: garbage in, garbage out (rác vào, rác ra). Hệ thống của chúng ta sẽ chỉ có khả năng học hỏi nếu dữ liệu đào tạo chứa đủ các tính năng liên quan và không có quá nhiều tính năng không liên quan. Một phần quan trọng trong sự thành công của một dự án máy học là có một bộ tính năng tốt để đào tạo. Quá trình này, được gọi là *kỹ thuật tính năng*, bao gồm các bước sau:

* Lựa chọn tính năng (lựa chọn các tính năng hữu ích nhất để đào tạo trong số các tính năng hiện có).
* Trích xuất tính năng (kết hợp các tính năng hiện có để tạo ra một tính năng hữu ích hơn như việc sử dụng các thuật toán giảm kích thước).
* Tạo các tính năng mới bằng cách thu thập dữ liệu mới.

#### e. Quá khớp dữ liệu huấn luyện

Trong học máy, *trang bị quá mức* (hoặc *quá khớp*): có nghĩa là mô hình hoạt động tốt trên dữ liệu huấn luyện, nhưng nó không tổng quát hóa tốt (hoạt động kém trên dữ liệu mới).

Các mô hình phức tạp như mạng nơ-ron sâu có thể phát hiện các mẫu tinh tế trong dữ liệu, nhưng nếu tập huấn luyện bị nhiễu hoặc nếu nó quá nhỏ, dẫn đến nhiễu lấy mẫu, thì mô hình có khả năng phát hiện các mẫu trong chính nhiễu đó. Những mẫu này sẽ không khái quát hóa cho các trường hợp mới. Trong trường hợp đó, một mô hình phức hợp có thể phát hiện các mẫu như thực tế. Trường hợp này xảy ra trong dữ liệu huấn luyện một cách tình cờ, nhưng mô hình không có cách nào để biết liệu một mẫu là có thật hay chỉ đơn giản là kết quả của nhiễu trong dữ liệu.

Hạn chế một mô hình để làm cho nó đơn giản hơn và giảm nguy cơ trang bị quá mức được gọi là *chính quy hóa*. Ví dụ: Một mô hình tuyến tính có hai tham số là chiều cao θ0 và độ dốc θ1. Thuật toán học gôm hai bậc tự do để điều chỉnh mô hình với dữ liệu huấn luyện: nó có thể điều chỉnh cả θ0 và θ1 của đường thẳng. Nếu chúng ta buộc θ1 = 0, thuật toán sẽ chỉ có một bậc tự do và sẽ khó khớp dữ liệu hơn: tất cả những gì nó có thể làm là di chuyển đường thẳng lên hoặc xuống để càng gần với các trường hợp đào tạo, vì vậy nó sẽ kết thúc xung quanh giá trị trung bình. Nếu chúng ta cho phép thuật toán sửa đổi θ1 nhưng chúng ta buộc nó phải giữ nó ở mức nhỏ, thì thuật toán học sẽ có tkhoảng từ một đến hai bậc tự do. Nó sẽ tạo ra một mô hình đơn giản hơn mô hình có hai bậc tự do, nhưng phức tạp hơn mô hình chỉ có một.



##### **Hình 1. 9** Chính quy hóa làm giảm nguy cơ quá khớp

Hình 1.10 cho thấy ba mô hình. Đường nét chấm chấm thể hiện mô hình ban đầu được đào tạo trên các quốc gia được biểu thị dưới dạng hình tròn (không có quốc gia nào được biểu thị dưới dạng hình vuông), đường nét liền là mô hình thứ hai được đào tạo với tất cả các quốc gia (hình tròn và hình vuông) và đường nét đứt là mô hình được đào tạo với cùng dữ liệu như mô hình đầu tiên nhưng có ràng buộc chính quy hóa. Ta có thể thấy rằng việc chính quy hóa buộc mô hình phải có độ dốc nhỏ hơn: mô hình này không khớp với dữ liệu đào tạo (vòng tròn) như mô hình đầu tiên, nhưng nó thực sự khái quát tốt hơn cho các ví dụ mới mà nó không thấy trong quá trình đào tạo (hình vuông) .

Lượng chính quy hóa để áp dụng trong quá trình học có thể được kiểm soát bởi một *siêu tham số*. Siêu tham số là một tham số của thuật toán học (không phải của mô hình). Như vậy, nó không bị ảnh hưởng bởi chính thuật toán học tập, nó phải được đặt trước khi đào tạo và không đổi trong quá trình đào tạo. Nếu ta đặt siêu tham số chuẩn hóa thành một giá trị rất lớn, ta sẽ nhận được một mô hình gần như bằng phẳng (độ dốc gần bằng 0), thuật toán học chắc chắn sẽ không bị quá khớp với dữ liệu đào tạo, nhưng sẽ ít có khả năng tìm ra giải pháp tốt. Điều chỉnh siêu tham số là một phần quan trọng trong việc xây dựng hệ thống máy học.

Việc trang bị quá mức xảy ra khi mô hình quá phức tạp so với số lượng và độ nhiễu của dữ liệu huấn luyện. Dưới đây là các giải pháp khả thi:

* Đơn giản hóa mô hình bằng cách chọn một mô hình có ít tham số hơn (ví dụ: mô hình tuyến tính thay vì mô hình đa thức bậc cao), bằng cách giảm số lượng thuộc tính trong dữ liệu huấn luyện hoặc bằng cách hạn chế mô hình.
* Thu thập thêm dữ liệu đào tạo.
* Giảm nhiễu trong dữ liệu đào tạo (ví dụ: sửa lỗi dữ liệu và loại bỏ các giá trị ngoại lệ).

#### f. Không khớp dữ liệu huấn luyện

Không khớp thì ngược lại với quá khớp: nó xảy ra khi mô hình quá đơn giản để tìm hiểu cấu trúc cơ bản của dữ liệu.

Dưới đây là các giải pháp chính để khắc phục sự cố này:

* Chọn một mô hình mạnh mẽ hơn, với nhiều thông số hơn.
* Cung cấp các tính năng tốt hơn cho thuật toán học (kỹ thuật tính năng).
* Giảm các ràng buộc trên mô hình (ví dụ bằng cách giảm siêu tham số chính quy hóa).

### Kiểm tra và xác thực mô hình học máy

Cách duy nhất để biết một mô hình sẽ khái quát hóa tốt như thế nào đối với dữ liệu mới là chia dữ liệu thành hai tập hợp*: tập huấn luyện* và *tập kiểm tra*. Huấn luyện mô hình bằng cách sử dụng tập huấn luyện và kiểm tra nó bằng tập kiểm tra. Tỷ lệ lỗi đối với các trường hợp mới được gọi là *lỗi tổng quát hóa* (*hoặc lỗi ngoài mẫu*) và bằng cách đánh giá mô hình trên tập kiểm tra, ta sẽ ước tính được lỗi này. Giá trị này cho ta biết mô hình sẽ hoạt động tốt như thế nào trong các trường hợp mà nó chưa từng thấy trước đây.

Nếu lỗi huấn luyện thấp (tức là mô hình mắc ít lỗi trên tập huấn luyện) nhưng lỗi tổng quát hóa cao, điều đó có nghĩa là mô hình khớp quá nhiều dữ liệu huấn luyện.

Người ta thường sử dụng 80% dữ liệu để đào tạo và giữ lại 20% để kiểm tra. Tuy nhiên, điều này còn phụ thuộc vào kích thước của tập dữ liệu.

#### a. Điều chỉnh siêu tham số và lựa chọn mô hình

Khi ta cần lựa chọn giữa hai loại mô hình (ví dụ: mô hình tuyến tính và mô hình đa thức), ta có thể đào tạo cả hai và so sánh mức độ tổng quát của chúng bằng cách sử dụng tập kiểm tra.

Sau khi tìm được mô hình có lỗi tổng quát hoá thấp, ta cần lựa chọn giá trị của siêu tham số để tránh vấn đề quá khớp, và chúng ta cần đào tạo 100 mô hình khác nhau bằng cách sử dụng 100 giá trị khác nhau cho siêu tham số này. Giả sử tìm thấy giá trị siêu tham số tốt nhất tạo ra một mô hình có sai số tổng quát hóa thấp nhất - giả sử chỉ sai số 5%. Khi đưa mô hình này vào thực tế, nó hoạt động kém và tạo ra 15% lỗi. Vấn đề xảy ra là chúng ta đã đo lỗi tổng quát hóa nhiều lần trên tập kiểm tra và ta đã điều chỉnh mô hình và siêu tham số để tạo ra mô hình tốt nhất cho *tập cụ thể đó*. Điều này có nghĩa là mô hình không có khả năng hoạt động tốt trên dữ liệu mới.

Một giải pháp phổ biến cho vấn đề này được gọi là *xác thực holdout*: chỉ cần giữ lại một phần của tập huấn luyện để đánh giá một số mô hình ứng cử viên và chọn mô hình tốt nhất. Tập tổ chức mới được gọi là *tập xác thực* (hoặc tập phát triển). Cụ thể hơn, khi ta huấn luyện nhiều mô hình với các siêu tham số khác nhau trên tập huấn luyện rút gọn (tức là tập huấn luyện đầy đủ trừ đi tập xác thực) và ta chọn mô hình hoạt động tốt nhất trên tập xác thực. Sau quá trình xác thực holdout này, chúng ta huấn luyện mô hình tốt nhất trên tập huấn luyện đầy đủ (bao gồm cả tập hợp xác thực) và điều này mang lại cho ta mô hình cuối cùng. Cuối cùng, chỉ cần đánh giá mô hình cuối cùng này trên tập kiểm tra để ước tính lỗi tổng quát hóa.



##### **Hình 1. 10** Lựa chọn mô hình bằng cách sử dụng xác thực holdout

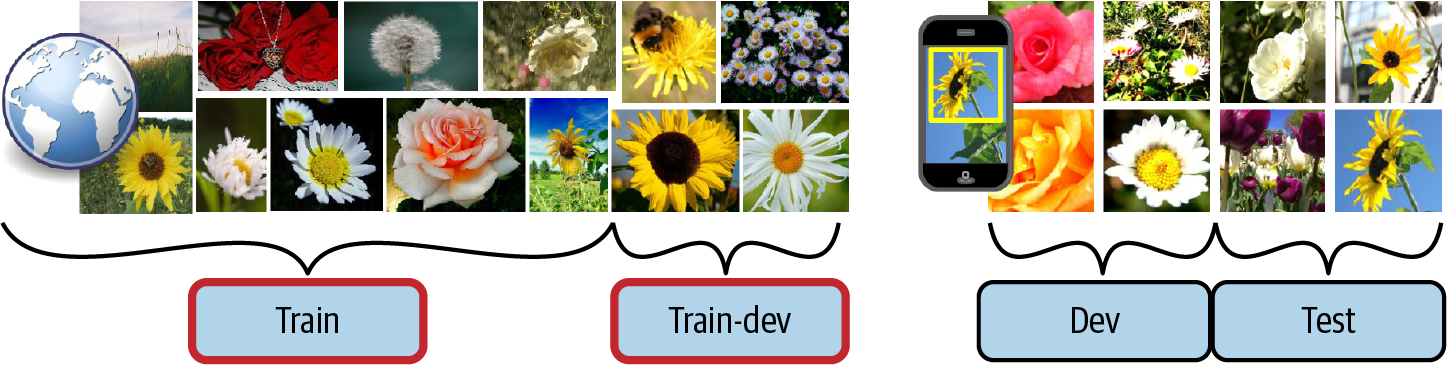
Giải pháp này thường hoạt động khá tốt. Tuy nhiên, nếu tập xác thực quá nhỏ, thì việc đánh giá mô hình sẽ không chính xác: cuối cùng có thể chọn nhầm một mô hình dưới mức tối ưu. Ngược lại, nếu tập xác thực quá lớn thì tập huấn luyện còn lại sẽ nhỏ hơn nhiều so với tập huấn luyện đầy đủ. Điều này là tệ, vì mô hình cuối cùng sẽ được đào tạo trên tập huấn luyện đầy đủ, nên sẽ không lý tưởng nếu so sánh các mô hình ứng viên được đào tạo trên tập huấn luyện nhỏ hơn nhiều.

Một cách để giải quyết vấn đề này là thực hiện *xác thực chéo* lặp đi lặp lại, sử dụng nhiều bộ xác thực nhỏ. Mỗi mô hình được đánh giá một lần cho mỗi bộ xác thực sau khi được huấn luyện trên phần còn lại của dữ liệu. Bằng cách tính trung bình tất cả các đánh giá của một mô hình, ta sẽ có được thước đo chính xác hơn nhiều về hiệu suất của nó. Tuy nhiên, nhược điểm của phương pháp này là thời gian đào tạo rất lâu vì chúng phải thực hiện xác thực nhiều lần tuỳ thuộc vào số lượng bộ xác thực.

#### b. Dữ liệu không khớp

Dữ liệu không khớp xảy ra khi chúng ta không chuẩn bị được lượng dữ liệu đủ để đại diện cho đa số các trường hợp thực tế của đối tượng. Ví dụ: giả sử ta muốn tạo ứng dụng dành cho thiết bị di động để chụp ảnh hoa và tự động xác định loài của chúng. Chúng ta có thể tải xuống hàng triệu bức ảnh về hoa trên web, nhưng chúng sẽ không đại diện hoàn hảo cho những bức ảnh thực sự sẽ được chụp bằng ứng dụng trên thiết bị di động. Có thể chỉ có 1.000 ảnh đại diện (ảnh thực sự được chụp bằng ứng dụng).

Một giải pháp là đưa ra một số hình ảnh đào tạo (từ trang web) trong một tập hợp khác mà Andrew Ng đặt tên là *tập phát triển-đào tạo*. Sau khi mô hình được đào tạo (trên tập huấn luyện), ta có thể đánh giá nó trên tập phát triển - đào tạo. Nếu mô hình hoạt động kém, thì nó chắc chắn khớp quá mức với tập huấn luyện, vì vậy cần đơn giản hóa hoặc chuẩn hóa mô hình, lấy thêm dữ liệu huấn luyện và làm sạch dữ liệu huấn luyện. Nhưng nếu nó hoạt động tốt trên tập phát triển đào tạo, thì ta có thể đánh giá mô hình trên tập phát triển. Nếu nó hoạt động kém, thì vấn đề phải đến từ việc: dữ liệu không khớp. Có thể giải quyết vấn đề này bằng cách xử lý trước các hình ảnh trên web để làm cho chúng trông giống những bức ảnh sẽ được chụp bởi ứng dụng dành cho thiết bị di động, sau đó đào tạo lại mô hình. Sau khi có một mô hình hoạt động tốt trên cả tập phát triển - đào tạo và tập phát triển, ta có thể đánh giá mô hình đó lần cuối trên tập kiểm tra để biết nó có khả năng hoạt động của nó trong thực tế.



##### **Hình 1. 11** Sử dụng tập phát triển – đào tạo

Khi dữ liệu thực khan hiếm (phải), ta có thể sử dụng dữ liệu dồi dào tương tự (trái) để huấn luyện và giữ lại một số dữ liệu đó trong tập phát triển - đào tạo để đánh giá quá khớp; dữ liệu thực sau đó được sử dụng để đánh giá sự không phù hợp của dữ liệu (tập phát triển) và để đánh giá hiệu suất của mô hình cuối cùng (tập kiểm tra).

### Những thông số, thước đo để đánh giá mô hình học máy

#### a. Thước đo bài toán phân loại

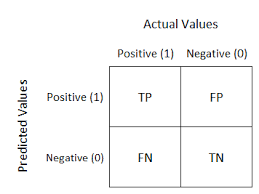
**Đo lường độ chính xác bằng cách sử dụng xác thực chéo (*k-fold*):**

Xác thực chéo *k*-fold có nghĩa là chia tập huấn luyện thành *k* phần không lặp, sau đó huấn luyện mô hình k lần, mỗi lần sử dụng k – 1 phần để học và đưa ra phần còn lại (không lặp) để đánh giá.

Trong thực tế, các nhà nghiên cứu học máy thường sử dụng: 5 folds hoặc 10 folds. Tuy nhiên, khi bộ dữ liệu có sai lệch, việc sử dụng độ chính xác không được ưu tiên, thay vào đó là *ma trận nhầm lẫn* (CM)

**Ma trận nhầm lẫn**

Ý tưởng chung của ma trận nhầm lẫn là đếm số lần các thể hiện của lớp A được phân loại thành lớp B, cho tất cả các cặp A/B. []



Mỗi hàng trong ma trận nhầm lẫn đại diện cho một *actual class* (lớp thực tế), trong khi mỗi cột đại diện cho một *predicted class* (lớp dự đoán). []

*TP* (*True Positive* – Dương thật): Biểu thị giá trị dự đoán của mô hình phù hợp với giá trị thực tế. Giá trị thực tế là dương và mô hình dự đoán giá trị dương.

**Hình 1. 12** Ma trận nhầm lẫn

*TN* (*True Negative* – Âm thật): Biểu thị giá trị dự đoán của mô hình phù hợp với giá trị thực tế. Giá trị thực tế là âm và mô hình dự đoán giá trị âm.

*FP* (*False Positive* – Dương giả): Mô hình đưa ra dự đoán sai. Giá trị thực tế là âm nhưng mô hình đưa ra dự đoán là dương. Đây còn được gọi là *type I error* (lỗi loại 1).

*FN* (*False Negative* – Âm giả): Mô hình đưa ra dự đoán sai. Giá trị thực tế là dương nhưng mô hình đã đưa ra dự đoán là âm. Đây còn được gọi là *type II error* (lỗi loại 2).[?]

Một thước đo được rút gọn từ ma trận nhầm lẫn là tính chính xác của các dự đoán dương: đây được gọi là *precision* của bộ phân loại.

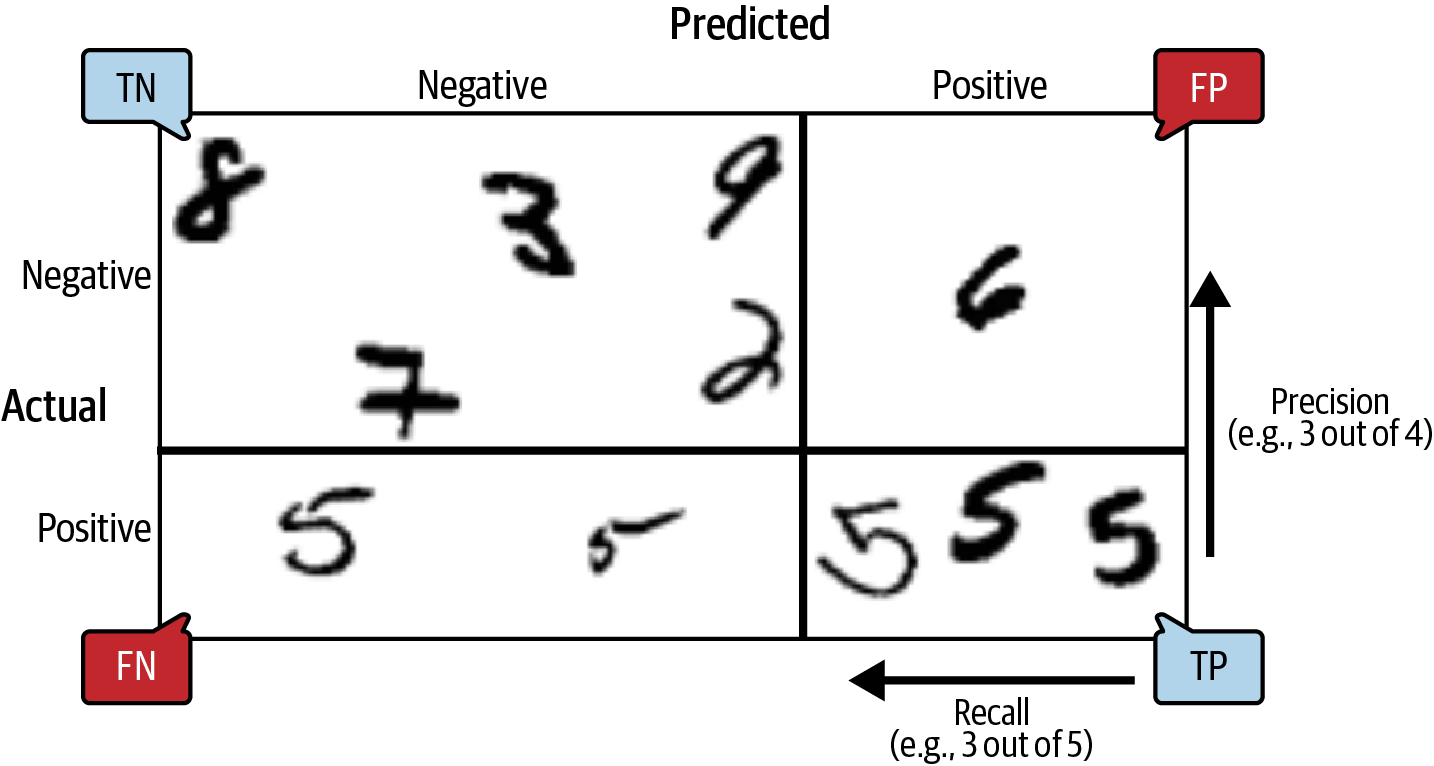
**Phương trình 1. 1** Precision

*TP* là số lượng dương thật và *FP* là số lượng dương giả.

Precision thường được sử dụng cùng với một số liệu khác có tên *recall*, còn được gọi là *sensitivity* (độ nhạy*)* hoặc *true positive rate* (TPR *-* tỷ lệ dương thực*)*: đây là tỷ lệ của các trường hợp dương được bộ phân loại phát hiện chính xác.

**Phương trình 1. 2** Recall

*FN* là số lượng âm giả.



##### **Hình 1. 13** Ma trận nhầm lẫn minh họa các ví dụ về âm thực (trên cùng bên trái), dương giả (trên cùng bên phải), âm giả (dưới cùng bên trái) và dương thật (dưới cùng bên phải)

**F1 score**

*F1 score* là thông số khi kết hợp precision và recall thành một số liệu duy nhất. F1 score là *harmonic mean* (trung bình điều hoà) của precision và recall. Giá trị trung bình hài hòa mang lại nhiều trọng lượng (tầm quan trọng) hơn cho các giá trị thấp. Bộ phân loại sẽ chỉ nhận được F1 score cao nếu cả recall và precision đều cao.

**Phương trình 1. 3** F1 score

F1 score thể hiện bộ phân loại có presicion và recall tương đồng. Trong một số ngữ cảnh nhất định, chúng ta sẽ quan tâm tới precision hoặc recall hơn là cái còn lại. Việc tăng precision sẽ làm giảm recall và ngược lại. Điều này được gọi là *precision/recall trade-off* (sự đánh đổi precision/recall).

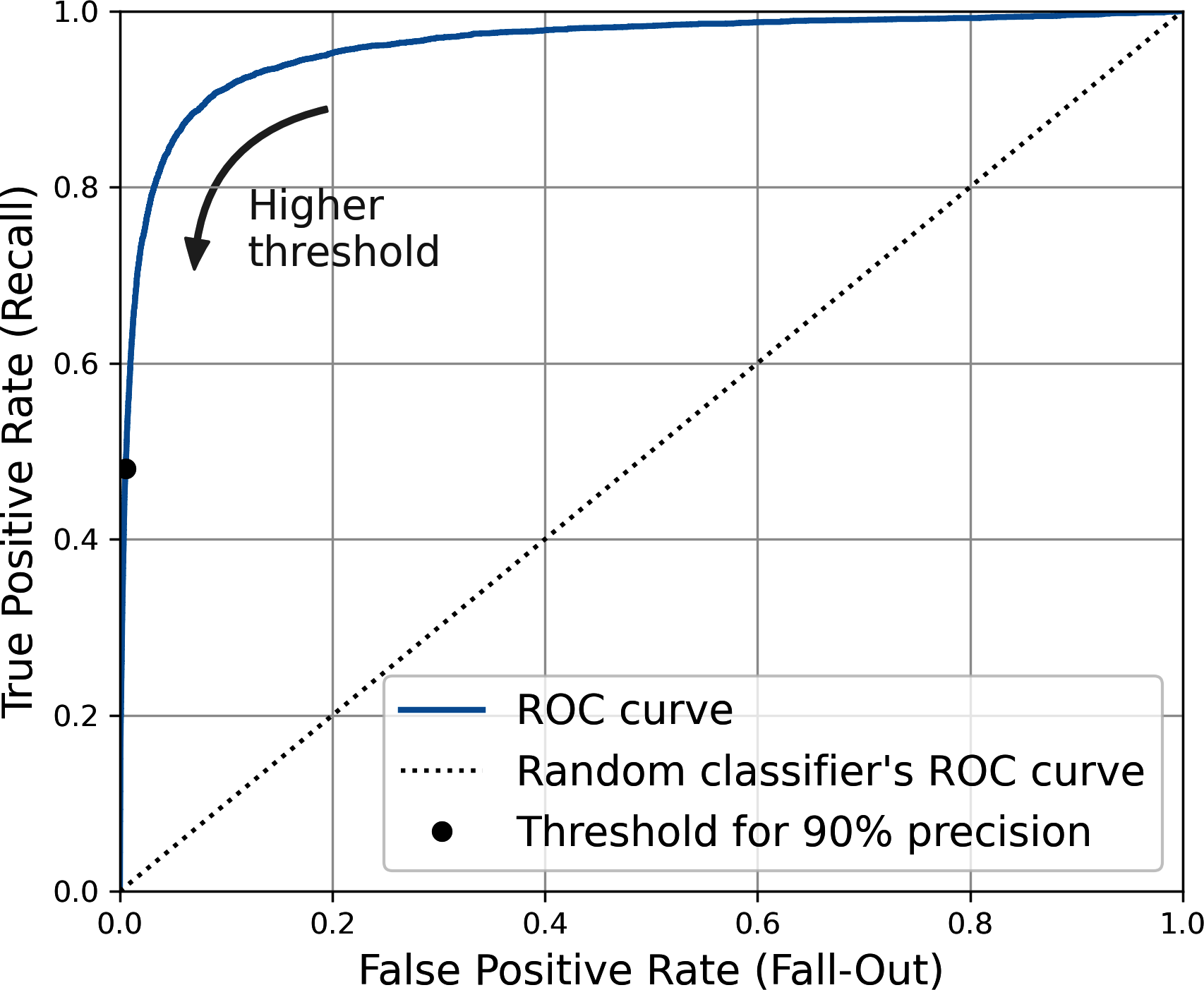
**ROC Curve** (Đường cong ROC)

*Receiver operating characteristic* curve(ROC – đường cong đặc tính vận hành máy thu) là biểu đồ thể hiện hiệu suất của mô hình phân loại ở tất cả các ngưỡng phân loại. Đường cong này vẽ hai tham số: *true positive rate* (tỷ lệ dương thực – tên gọi khác của recall) và *false positive rate* (FPR - tỷ lệ dương giả). FPR (còn được gọi là *fall-out* (tỷ lệ giảm)) là tỷ lệ các trường hợp âm được phân loại không chính xác thành dương.

**Phương trình 1. 4** False positive rate (FPR)

TNR: *true negative reate* (tỷ lệ âm thực): Là tỷ lệ của các trường hợp âm được phân loại chính xác là âm, TNR còn được gọi là *specificity* (tính đặc hiệu).

Tóm lại, đường con ROC biểu thị độ nhạy (recall) so với 1 – độ đặc hiệu.



##### **Hình 1. 14** Đường cong ROC vẽ tỷ lệ dương giả so với dương thực cho các ngưỡng

Vòng tròn màu đen làm nổi bật tỷ lệ đã chọn (với precision 90% và recall 48%). Recall (TPR) càng cao thì bộ phân loại tạo ra càng nhiều kết quả dương giả (FPR). Đường chấm chấm biểu thị đường cong ROC của một bộ phân loại hoàn toàn ngẫu nhiên, một bộ phân loại tốt ở càng xa đường cong ROC này càng tốt (về phía góc trên cùng bên trái).

Một cách để so sánh các bộ phân loại là đo *area under the curve* (AUC - diện tích dưới đường cong). Một bộ phân loại hoàn hảo sẽ có ROC AUC bằng 1, trong khi một bộ phân loại hoàn toàn ngẫu nhiên sẽ có ROC AUC bằng 0,5.

#### b. Thước đo bài toán hồi quy []

**Mean Squared Error** (Lỗi bình phương trung bình)

*Mean Squared Error* (MSE) là số liệu phổ biến nhất được sử dụng cho các bài toán hồi quy. Về cơ bản, nó tìm thấy sai số bình phương trung bình giữa các giá trị được dự đoán và thực tế. MSE là thước đo chất lượng của một công cụ ước tính - nó luôn không âm và các giá trị càng gần 0 càng tốt.

**Phương trình 1. 5** Mean Squared Error (MSE)

: số điểm dữ liệu

: giá trị quan sát

: giá trị dự đoán

Trong phân tích hồi quy, vẽ biểu đồ là một cách tự nhiên để xem xu hướng chung của toàn bộ dữ liệu. MSE cho ta biết mức độ gần của đường hồi quy với một tập hợp các điểm. Nó thực hiện điều này bằng cách lấy khoảng cách từ các điểm đến đường hồi quy (những khoảng cách này là “sai số”) và bình phương chúng. Bình phương rất quan trọng để giảm độ phức tạp với các dấu hiệu tiêu cực. Nó cũng tạo ra nhiều trọng lượng hơn cho sự khác biệt lớn hơn.

Việc giảm thiểu MSE giúp mô hình có thể chính xác hơn, có nghĩa là mô hình gần với dữ liệu thực tế hơn. Một ví dụ về hồi quy tuyến tính sử dụng phương pháp này là - phương pháp bình phương nhỏ nhất đánh giá sự phù hợp của mô hình hồi quy tuyến tính với tập dữ liệu hai biến, nhưng giới hạn của nó liên quan đến phân phối dữ liệu đã biết.

MSE càng thấp thì dự báo càng tốt.

**Mean Absolute Error** (Lỗi tuyệt đối trung bình)

*Mean Absolute Error* (MAE) đo độ lớn trung bình của các lỗi trong một tập hợp các dự đoán mà không cần xem xét hướng của chúng. Đó là giá trị trung bình trên mẫu thử nghiệm về sự khác biệt tuyệt đối giữa dự đoán và quan sát thực tế, trong đó tất cả các khác biệt riêng lẻ có trọng số bằng nhau.

**Phương trình 1. 6** Mean Absolute Error **(**MAE)

: số điểm dữ liệu

: giá trị thực

: giá trị dự đoán

Có thể diễn đạt MAE là tổng hòa của hai thành phần: Bất đồng về số lượng và Bất đồng về phân bổ.

MAE được biết đến là mạnh mẽ hơn đối với các yếu tố ngoại lai so với MSE. Lý do chính là trong MSE bằng cách bình phương các sai số, các giá trị ngoại lai (thường có sai số cao hơn các mẫu khác) được chú ý nhiều hơn và chiếm ưu thế trong sai số cuối cùng và tác động đến các tham số của mô hình.

**Root Mean Square Error (Lỗi trung bình bình phương gốc)**

*Root Mean Square Error* (RMSE) hoặc *Root Mean Square Deviation* (RMSD - Độ lệch gốc trung bình bình phương) là căn bậc hai của mức trung bình của các sai số bình phương. RMSE là độ lệch chuẩn của các phần dư (sai số dự đoán).

Phần dư là thước đo khoảng cách từ các điểm dữ liệu đường hồi quy; RMSE là thước đo mức độ dàn trải của những phần dư này, nói cách khác, nó cho ta biết mức độ tập trung của dữ liệu xung quanh đường phù hợp nhất.

**Phương trình 1. 7** Root Mean Square Deviation (RMSD)

: Số điểm dữ liệu không bị thiếu

: Chuỗi thời gian quan sát thực tế

: Chuỗi thời gian ước tính

Ảnh hưởng của mỗi lỗi đối với RMSE tỷ lệ với kích thước của lỗi bình phương, do đó các sai số lớn hơn có ảnh hưởng lớn đến RMSE một cách không cân xứng. Vì vậy, RMSE nhạy cảm với các yếu tố ngoại lai. Sai số bình phương trung bình gốc thường được sử dụng trong khí hậu học, dự báo và phân tích hồi quy để xác minh kết quả thực nghiệm.

Khi các quan sát và dự báo chuẩn hóa được sử dụng làm đầu vào RMSE, có mối quan hệ trực tiếp với hệ số tương quan. Ví dụ, nếu hệ số tương quan là 1, RMSE sẽ bằng 0, bởi vì tất cả các điểm nằm trên đường hồi quy (và do đó không có sai số).

RMSE luôn không âm và giá trị 0 (hầu như không bao giờ đạt được trong thực tế) sẽ chỉ ra sự phù hợp hoàn hảo với dữ liệu. Nói chung, RMSD thấp hơn sẽ tốt hơn RMSD cao hơn.

## Học sâu (Deep learning)

### Khái niệm về học sâu

Học sâu là một lĩnh vực con cụ thể của học máy: một bước tiến mới trong việc học các biểu diễn từ dữ liệu, nhấn mạnh vào việc học các lớp kế tiếp của các biểu diễn ngày càng có ý nghĩa. Từ “sâu” trong “học sâu” không ám chỉ đến bất kỳ loại hiểu biết sâu hơn nào mà phương pháp này đạt được, nó đại diện cho ý tưởng về các lớp biểu diễn liên tiếp này. Có bao nhiêu lớp đóng góp vào một mô hình dữ liệu được gọi là *depth* (độ sâu) của mô hình. Các tên thích hợp khác cho trường có thể là *layered representations learning* (học biểu diễn phân lớp) hoặc *hierarchical representations learning* (học biểu diễn phân cấp). Học sâu hiện đại thường liên quan đến hàng chục hoặc thậm chí hàng trăm lớp biểu diễn liên tiếp và tất cả chúng đều được học tự động khi tiếp xúc với dữ liệu huấn luyện. Trong khi đó, các cách tiếp cận khác đối với học máy có xu hướng chỉ tập trung vào việc học một hoặc hai lớp biểu diễn dữ liệu. Do đó, đôi khi chúng được gọi là *shallow learning* (học nông).

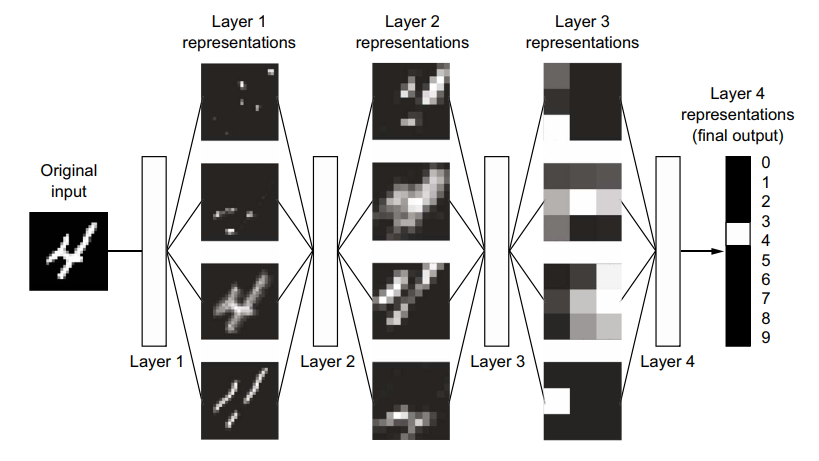
Trong học sâu, các biểu diễn theo lớp này được học thông qua các mô hình được gọi là *neural network* (mạng lưới thần kinh), được cấu trúc theo các lớp xếp chồng lên nhau. Thuật ngữ “neural network” đề cập đến sinh học thần kinh, nhưng mặc dù một số khái niệm trung tâm trong học sâu được phát triển một phần bằng cách lấy cảm hứng từ sự hiểu biết của chúng ta về bộ não (đặc biệt là vỏ não thị giác), các mô hình học sâu không phải là mô hình của não. Không có bằng chứng nào cho thấy bộ não thực hiện bất kỳ thứ gì giống như cơ chế học tập được sử dụng trong các mô hình học sâu hiện đại. Chúng ta có thể bắt gặp các bài báo khoa học đại chúng tuyên bố rằng học sâu hoạt động giống như bộ não hoặc được mô phỏng theo bộ não, nhưng thực tế không phải vậy. Có thể nói học sâu là một khung toán học để học các biểu diễn từ dữ liệu.



##### **Hình 1. 15** Một mạng lưới thần kinh sâu để phân loại chữ số

Mạng trên chuyển đổi hình ảnh số thành các biểu diễn ngày càng khác biệt so với hình ảnh gốc và ngày càng nhiều thông tin về kết quả cuối cùng. Có thể coi mạng sâu như một quá trình *information distillation* (chắt lọc thông tin) nhiều tầng, trong đó thông tin đi qua các bộ lọc liên tiếp và ngày càng được *purified* (lọc sạch - hữu ích đối với một số tác vụ).

**Vì vậy, về mặt kỹ thuật, học sâu là : một cách nhiều tầng để học cách biểu diễn dữ liệu**.

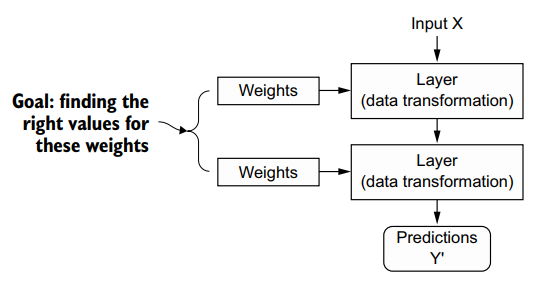


##### **Hình 1. 16** Biểu diễn dữ liệu được học bởi mô hình phân loại chữ số

### Cách hoạt động của học sâu

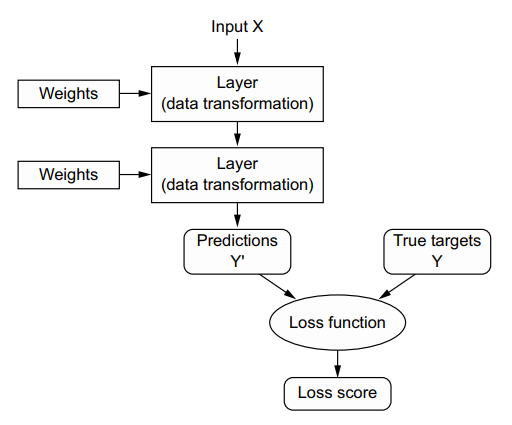
Chúng ta đã biết rằng học máy là ánh xạ các đầu vào (chẳng hạn như hình ảnh) tới các mục tiêu (chẳng hạn như nhãn “hoa hồng”), được thực hiện bằng cách quan sát nhiều ví dụ về đầu vào và mục tiêu. Và các mạng nơ-ron sâu thực hiện ánh xạ đầu vào đến đích thông qua một chuỗi sâu các phép biến đổi dữ liệu đơn giản (các lớp) và các phép biến đổi dữ liệu này được học bằng cách tiếp xúc với các ví dụ.

Thông số kỹ thuật về những gì một lớp thực hiện đối với dữ liệu đầu vào của nó được lưu trữ trong các *weights* (trọng số) của lớp, về bản chất là một loạt các con số. Về mặt kỹ thuật, phép biến đổi được thực hiện bởi một lớp được *parameterized* (tham số hóa) bởi các trọng số của nó. (Trọng số đôi khi còn được gọi là *parameters* (tham số) của một lớp). Trong ngữ cảnh này, *learning* (học) có nghĩa là tìm một tập hợp các giá trị cho trọng số của tất cả các lớp trong mạng, sao cho mạng sẽ ánh xạ chính xác đầu vào mẫu tới các mục tiêu được liên kết của chúng. Nhưng vấn đề ở đây là: một mạng nơ-ron sâu có thể chứa hàng chục triệu tham số. Tìm các giá trị chính xác cho tất cả chúng là một nhiệm vụ khó khăn, đặc biệt là khi sửa đổi giá trị của một tham số sẽ ảnh hưởng đến hành vi của tất cả các tham số khác!



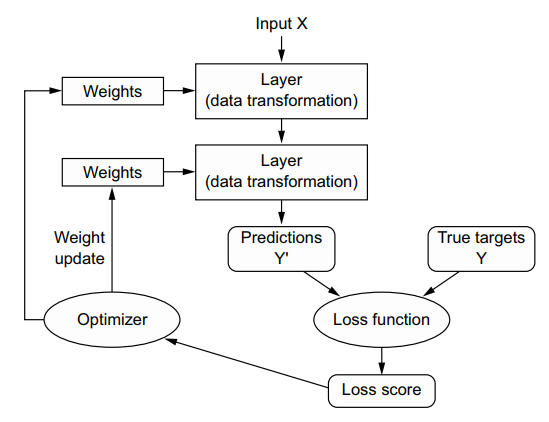
##### **Hình 1. 17** Mạng lưới thần kinh được tham số hóa bởi các trọng số của nó.

Để kiểm soát đầu ra của mạng nơ-ron, chúng ta cần đo lường xem đầu ra này khác bao nhiêu so với những gì được mong đợi. Đây là công việc của *loss function* (hàm mất mát) của mạng, đôi khi còn được gọi là *objective function* (hàm mục tiêu) hoặc *cost fuction* (hàm chi phí). Hàm mất mát lấy các dự đoán của mạng và mục tiêu thực (cái chúng ta muốn mạng xuất ra) và tính điểm khoảng cách, nắm bắt xem mạng đã hoạt động tốt như thế nào.



##### **Hình 1. 18** Hàm mất mát đo lường chất lượng đầu ra của mạng

Thủ thuật cơ bản trong học sâu là sử dụng điểm số này làm tín hiệu phản hồi để điều chỉnh giá trị của các trọng số, theo hướng sẽ hạ thấp điểm số tổn thất. Việc điều chỉnh này là công việc của *optimizer* (trình tối ưu hóa), thực hiện cái được gọi là thuật toán *Backpropagation* (Lan truyền ngược): thuật toán trung tâm trong học sâu.



##### **Hình 1. 19** Điểm mất mát được sử dụng làm tín hiệu phản hồi để điều chỉnh trọng số.

Ban đầu, các trọng số của mạng được gán các giá trị ngẫu nhiên, vì vậy mạng chỉ thực hiện một loạt các phép biến đổi ngẫu nhiên. Vì vậy, đầu ra của nó khác xa so với mức lý tưởng và điểm mất mát rất cao. Nhưng với mọi mẫu mà mạng xử lý, các trọng số được điều chỉnh từng chút một theo hướng chính xác và điểm mất mát giảm xuống. Đây là *training loop* (vòng lặp huấn luyện), được lặp lại đủ số lần (thường là hàng chục lần lặp trên hàng nghìn mẫu), mang lại các giá trị trọng số làm giảm thiểu hàm mất mát. Mạng có tổn thất tối thiểu là mạng mà đầu ra càng gần với mục tiêu càng tốt: mạng được đào tạo.

### Những hình thức học máy khác học sâu

Học sâu đã đạt được mức độ chú ý của công chúng và đầu tư trong ngành chưa từng thấy trong lịch sử AI, nhưng đây không phải là hình thức học máy thành công đầu tiên. Có thể nói rằng hầu hết các thuật toán học máy được sử dụng trong ngành hiện nay không phải là thuật toán học sâu. Học sâu không phải lúc nào cũng là công cụ phù hợp cho công việc - đôi khi không có đủ dữ liệu để học sâu có thể áp dụng được và đôi khi vấn đề được giải quyết tốt hơn bằng một thuật toán khác. Có nhiều cách tiếp cận khác có thể hữu ích hơn học sâu khi vào một bài toán cụ thể nào đó.

#### a. Probabilistic modeling (Mô hình xác suất)

*Probabilistic modeling* là việc áp dụng các nguyên tắc thống kê để phân tích dữ liệu. Đây là một trong những hình thức học máy sớm nhất và nó vẫn được sử dụng rộng rãi cho đến ngày nay. Một trong những thuật toán nổi tiếng nhất trong danh mục này là thuật toán Naive Bayes.

Một mô hình có liên quan chặt chẽ là *logistic regression* (logreg - hồi quy logistic) đôi khi được coi là “Hello World” của học máy hiện đại. Logreg là một thuật toán phân loại chứ không phải là một thuật toán hồi quy. Giống như Naive Bayes, logreg có trước máy tính từ rất lâu, nhưng nó vẫn hữu ích cho đến ngày nay, nhờ bản chất đơn giản và linh hoạt của nó. Đây thường là thuật toán đầu tiên được sử dụng để hiểu về nhiệm vụ phân loại.

#### b. Early neural networks (Mạng lưới thần kinh ban đầu)

Các phiên bản ban đầu của mạng lưới thần kinh đã bị thay thế hoàn toàn bởi các biến thể hiện đại. Mặc dù những ý tưởng cốt lõi của mạng nơ-ron đã được nghiên cứu từ những năm 1950, nhưng trong một thời gian dài, mảnh ghép còn thiếu là một cách hiệu quả để huấn luyện các mạng thần kinh lớn. Điều này đã thay đổi vào giữa những năm 1980, khi nhiều người khám phá lại thuật toán Lan truyền ngược - một cách để huấn luyện các chuỗi phép toán tham số bằng cách sử dụng tối ưu hóa giảm dần độ dốc và bắt đầu áp dụng nó vào hệ thần kinh các mạng.

Ứng dụng thực tế thành công đầu tiên của mạng nơ-ron đến từ Bell Labs vào năm 1989, khi Yann LeCun kết hợp các ý tưởng trước đó về mạng nơ-ron tích chập và lan truyền ngược, rồi áp dụng chúng vào bài toán phân loại các chữ số viết tay. Mạng kết quả, được đặt tên là *LeNet*, được Bưu điện Hoa Kỳ sử dụng vào những năm 1990 để tự động đọc mã ZIP trên phong bì thư.

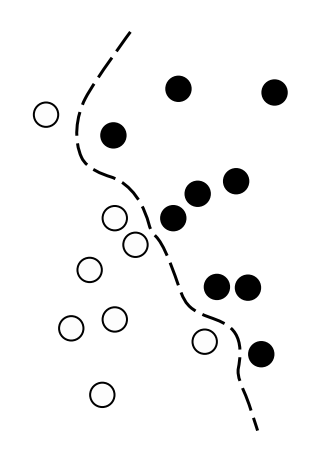
#### c. Kernel methods (Phương pháp Kernel)

Vào những năm 1990, một cách tiếp cận mới đối với máy học đã trở nên nổi tiếng: phương pháp kernel. *Phương pháp kernel* là một nhóm các thuật toán phân loại, trong đó nổi tiếng nhất là *Support Vector Machine* (SVM - Máy vectơ hỗ trợ). Công thức hiện đại của SVM được phát triển bởi Vladimir Vapnik và Corinna Cortes vào đầu những năm 1990 tại Bell Labs và được xuất bản vào năm 1995, và công thức tuyến tính cũ hơn đã được Vapnik và Alexey Chervonenkis công bố vào đầu năm 1963.

SVM là một thuật toán phân loại hoạt động bằng cách tìm “decision boundaries” (ranh giới quyết định) phân tách hai lớp. SVM tiến hành tìm các ranh giới này theo hai bước:

* Dữ liệu được ánh xạ tới một biểu diễn nhiều chiều mới, trong đó ranh giới quyết định có thể được biểu thị dưới dạng siêu phẳng (nếu dữ liệu là hai chiều, siêu phẳng sẽ là một đường thẳng).
* Một ranh giới quyết định tốt (một siêu phẳng phân tách) được tính bằng cách cố gắng tối đa hóa khoảng cách giữa siêu phẳng và các điểm dữ liệu gần nhất từ ​​mỗi lớp, một bước được gọi là *maximizing the margin* (tối đa hóa lề). Điều này cho phép ranh giới khái quát hóa tốt cho các mẫu mới bên ngoài tập dữ liệu huấn luyện.

Kỹ thuật ánh xạ dữ liệu sang biểu diễn nhiều chiều trong đó vấn đề phân loại trở nên đơn giản hơn trên thực tế thường khó tính toán. Đó là nơi *kernel trick* (thủ thuật kernel) xuất hiện (ý tưởng chính mà các phương thức hạt nhân được đặt theo tên).

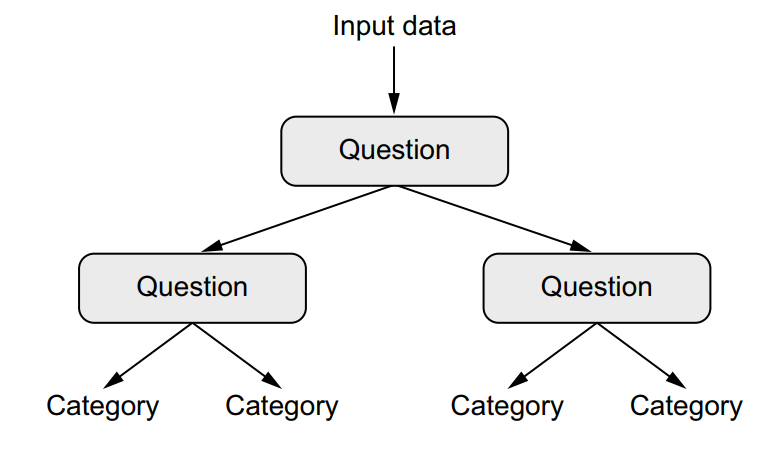
Đây là ý chính của nó: để tìm các siêu phẳng quyết định tốt trong không gian biểu diễn mới, ta không cần phải tính toán rõ ràng tọa độ của các điểm trong không gian mới mà chỉ cần tính toán khoảng cách giữa các cặp điểm trong không gian đó, điều này có thể được thực hiện một cách hiệu quả bằng cách sử dụng hàm kernel. *Kernel function* (Hàm kernel) là một hoạt động có thể điều khiển bằng tính toán, ánh xạ hai điểm bất kỳ trong không gian ban đầu với khoảng cách giữa các điểm này trong không gian biểu diễn đích, hoàn toàn bỏ qua tính toán rõ ràng của biểu diễn mới. Các hàm kernel thường được tạo thủ công thay vì học từ dữ liệu - trong trường hợp SVM, chỉ siêu phẳng phân tách được học.

##### **Hình 1. 20** Ranh giới quyết định

SVM tỏ ra khó mở rộng quy mô thành các tập dữ liệu lớn và không mang lại kết quả tốt cho các vấn đề về nhận thức như phân loại hình ảnh. Bởi vì SVM là một phương pháp nông, nên việc áp dụng SVM cho các vấn đề về tri giác đòi hỏi trước tiên phải trích xuất các biểu diễn hữu ích theo cách thủ công (một bước được gọi là *kỹ thuật tính năng*), điều này rất khó và dễ hỏng.

#### d. Decision trees, random forests, and gradient boosting machines (Cây quyết định, rừng ngẫu nhiên và máy tăng cường độ dốc)

*Decision tree* (cây quyết định) là các cấu trúc giống như lưu đồ cho phép phân loại các điểm dữ liệu đầu vào hoặc dự đoán các giá trị đầu ra cho các đầu vào. Chúng rất dễ hình dung và diễn giải. Cây quyết định học được từ dữ liệu bắt đầu nhận tập trung nghiên cứu đáng kể vào những năm 2000 và đến năm 2010, chúng thường được ưa chuộng hơn các phương pháp kernel.



##### **Hình 1. 21** Cây quyết định: các tham số được học là các câu hỏi về dữ liệu.

Thuật toán *Random Forest* (Rừng ngẫu nhiên) đã giới thiệu một phương pháp học cây quyết định mạnh mẽ, thiết thực bao gồm việc xây dựng nhiều cây quyết định chuyên biệt rồi tập hợp các kết quả đầu ra của chúng. Rừng ngẫu nhiên có thể áp dụng cho nhiều loại vấn đề - có thể nói rằng chúng hầu như luôn là thuật toán tốt thứ hai cho bất kỳ nhiệm vụ học máy nông nào. Khi trang web cuộc thi học máy phổ biến Kaggle (<http://kaggle.com>) bắt đầu vào năm 2010, rừng ngẫu nhiên nhanh chóng trở thành mục yêu thích trên nền tảng - cho đến năm 2014, khi các *gradient boosting machines* (máy tăng cường độ dốc) tiếp quản. Giống như rừng ngẫu nhiên, máy tăng cường độ dốc là một kỹ thuật máy học dựa trên tập hợp các mô hình dự đoán yếu, thường là cây quyết định. Nó sử dụng tăng cường độ dốc, một cách để cải thiện bất kỳ mô hình máy học nào bằng cách đào tạo lặp đi lặp lại các mô hình mới chuyên giải quyết các điểm yếu của các mô hình trước đó. Áp dụng cho các cây quyết định, việc sử dụng kỹ thuật tăng cường độ dốc dẫn đến các mô hình hoàn toàn vượt trội so với các khu rừng ngẫu nhiên trong hầu hết thời gian, trong khi có các thuộc tính tương tự. Nó có thể là một trong những thuật toán tốt nhất để xử lý dữ liệu phi tri giác hiện nay. Bên cạnh học sâu, đây là một trong những kỹ thuật được sử dụng phổ biến nhất trong các cuộc thi Kaggle.

#### e. Neural networks (Mạng lưới thần kinh)

Vào năm 2011, Dan Ciresan từ IDSIA đã bắt đầu giành chiến thắng trong các cuộc thi phân loại hình ảnh mang tính học thuật với các mạng nơ-ron sâu do GPU đào tạo - thành công thực tế đầu tiên của học sâu hiện đại. Bước ngoặt vào năm 2012, với sự tham gia của nhóm Hinton trong thử thách phân loại hình ảnh quy mô lớn hàng năm ImageNet (Thử thách nhận dạng hình ảnh quy mô lớn ImageNet - ILSVRC). Thử thách ImageNet nổi tiếng là khó vào thời điểm đó, bao gồm việc phân loại các hình ảnh màu có độ phân giải cao thành 1.000 danh mục khác nhau sau khi đào tạo trên 1,4 triệu hình ảnh. Năm 2011, năm độ chính xác hàng đầu của mô hình chiến thắng, dựa trên các phương pháp tiếp cận cổ điển đối với thị giác máy tính, chỉ là 74,3%. Sau đó, vào năm 2012, một nhóm do Alex Krizhevsky đứng đầu và được cố vấn bởi Geoffrey Hinton đã có thể đạt được độ chính xác hàng đầu năm độ chính xác là 83,6% - một bước đột phá đáng kể. Cuộc thi đã được thống trị bởi các mạng thần kinh tích chập sâu hàng năm kể từ đó. Đến năm 2015, người chiến thắng đạt độ chính xác 96,4% và nhiệm vụ phân loại trên ImageNet được coi là một vấn đề được giải quyết hoàn toàn.

Kể từ năm 2012, các mạng thần kinh tích chập sâu (*convnets*) đã trở thành thuật toán cho tất cả các tác vụ thị giác máy tính, chúng hoạt động trên tất cả các nhiệm vụ nhận thức. Tại bất kỳ hội nghị thị giác máy tính lớn nào sau năm 2015, gần như không thể tìm thấy các bài thuyết trình nào không liên quan đến convnets dưới một hình thức nào đó. Đồng thời, học sâu cũng đã tìm thấy các ứng dụng trong nhiều loại vấn đề khác, chẳng hạn như xử lý ngôn ngữ tự nhiên. Nó đã thay thế hoàn toàn SVM và cây quyết định trong nhiều ứng dụng. Ví dụ, trong vài năm, Tổ chức Nghiên cứu Hạt nhân Châu Âu, CERN, đã sử dụng các phương pháp dựa trên cây quyết định để phân tích dữ liệu hạt từ máy dò ATLAS tại Máy Va chạm Hadron Lớn (LHC), nhưng CERN cuối cùng đã chuyển sang sử dụng mạng lưới thần kinh sâu dựa trên Keras do hiệu suất cao hơn và dễ đào tạo trên các tập dữ liệu lớn.

#### f. Điểm khác biệt của học sâu

Lý do chính khiến học sâu phát triển nhanh chóng là vì nó mang lại hiệu suất tốt hơn cho nhiều vấn đề. Nhưng đó không phải là lý do duy nhất. Học sâu cũng giúp việc giải quyết vấn đề trở nên dễ dàng hơn nhiều, vì nó tự động hóa hoàn toàn cái bước mà trước đó từng là bước quan trọng nhất trong quy trình học máy: kỹ thuật tính năng.

Các kỹ thuật học máy trước đây - học nông - chỉ liên quan đến việc chuyển đổi dữ liệu đầu vào thành một hoặc hai không gian biểu diễn liên tiếp, thường thông qua các phép biến đổi đơn giản như phép chiếu phi tuyến tính nhiều chiều (SVM) hoặc cây quyết định. Nhưng các biểu diễn tinh tế được yêu cầu bởi các vấn đề phức tạp thường không thể đạt được bằng các kỹ thuật như vậy. Do đó, con người đã phải nỗ lực rất nhiều để làm cho dữ liệu đầu vào ban đầu dễ xử lý hơn bằng các phương pháp này: họ phải thiết kế thủ công các lớp biểu diễn tốt cho dữ liệu của mình. Điều này được gọi là *feature engineering* (kỹ thuật tính năng). Mặt khác, học sâu hoàn toàn tự động hóa bước này: với học sâu, chúng ta học tất cả các tính năng trong một lượt thay vì phải tự thiết kế chúng. Điều này đã đơn giản hóa rất nhiều quy trình học máy, thường thay thế các quy trình nhiều tầng phức tạp bằng một mô hình học sâu đầu cuối, đơn giản, duy nhất.

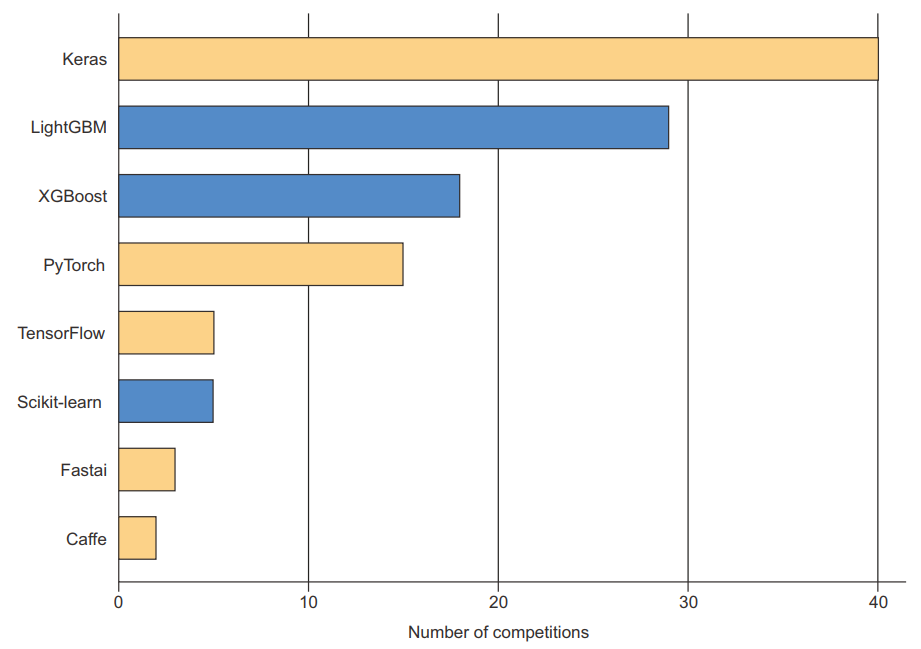
Điểm đáng chú ý là học sâu cho phép một mô hình học tất cả các lớp biểu diễn cùng một lúc, thay vì liên tiếp. Với tính năng học tập chung, bất cứ khi nào mô hình điều chỉnh một trong các tính năng bên trong của nó, tất cả các tính năng khác phụ thuộc vào nó sẽ tự động thích ứng với thay đổi. Mọi thứ được giám sát bởi một tín hiệu phản hồi duy nhất: mọi thay đổi trong mô hình đều phục vụ mục tiêu cuối cùng. Điều này mạnh mẽ hơn nhiều so với việc xếp chồng các mô hình nông, bởi vì nó cho phép học các biểu diễn trừu tượng, phức tạp bằng cách chia chúng thành các chuỗi dài các lớp trung gian, mỗi lớp chỉ là một phép biến đổi đơn giản so với cái trước đó.

Đây là hai đặc điểm cơ bản của cách học sâu học từ dữ liệu: cách tăng dần, từng lớp trong đó các biểu diễn ngày càng phức tạp được phát triển và thực tế là các biểu diễn gia tăng trung gian này được học cùng nhau, mỗi lớp được cập nhật để tuân theo cả hai nhu cầu biểu diễn của tầng bên trên và tầng bên dưới. Cùng với nhau, hai thuộc tính này đã làm cho học sâu thành công hơn rất nhiều so với các phương pháp học máy trước đây.

### Bối cảnh học máy hiện đại

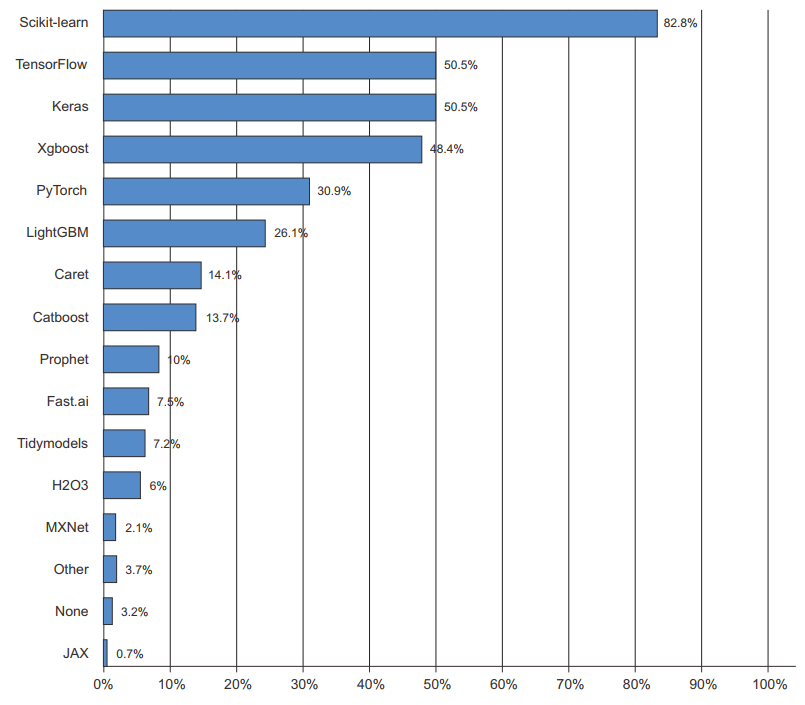
Để hiểu được bối cảnh, xu thế hiện tại của các thuật toán và công cụ học máy, ta nên xem xét các cuộc thi học máy trên Kaggle. Do môi trường cạnh tranh cao (một số cuộc thi có hàng nghìn người tham gia và giải thưởng hàng triệu đô la) và do có nhiều vấn đề về máy học được đề cập, Kaggle cung cấp một cách thực tế để đánh giá điều gì hiệu quả và điều gì không.

Vào đầu năm 2019, Kaggle đã thực hiện một cuộc khảo sát hỏi các đội lọt vào top 5 của bất kỳ cuộc thi nào kể từ năm 2017 về công cụ phần mềm chính mà họ đã sử dụng trong cuộc thi. Kết quả chỉ ra rằng: các nhóm hàng đầu có xu hướng sử dụng phương pháp học sâu (thường xuyên nhất là qua thư viện Keras) hoặc cây tăng cường độ dốc (thường xuyên nhất là qua thư viện LightGBM hoặc XGBoost).



##### **Hình 1. 22** Các công cụ học máy được các nhóm hàng đầu trên Kaggle sử dụng

Kaggle cũng thực hiện một cuộc khảo sát hàng năm giữa các chuyên gia khoa học dữ liệu và máy học trên toàn thế giới. Với hàng chục nghìn người trả lời, cuộc khảo sát này là một trong những nguồn đáng tin cậy nhất về tình trạng của ngành.



##### **Hình 1. 23** Việc sử dụng công cụ trong ngành khoa học dữ liệu và học máy (Nguồn: <www.kaggle.com/kaggle-survey-2020>)

Từ năm 2016 đến năm 2020, toàn bộ ngành khoa học dữ liệu và máy học đã bị chi phối bởi hai cách tiếp cận: học sâu và cây tăng cường độ dốc. Cụ thể, cây tăng cường độ dốc được sử dụng cho các vấn đề có sẵn dữ liệu có cấu trúc, trong khi học sâu được sử dụng cho các vấn đề về tri giác như phân loại hình ảnh.

Người dùng cây tăng cường độ dốc có xu hướng sử dụng Scikit-learn, XGBoost hoặc LightGBM. Trong khi đó, hầu hết những người thực hành học sâu đều sử dụng Keras, thường kết hợp với khung gốc của nó là TensorFlow. Điểm chung của các công cụ này là chúng đều là thư viện Python: Python cho đến nay là ngôn ngữ được sử dụng rộng rãi nhất cho học máy và khoa học dữ liệu.

### Lý do để lựa chọn học sâu

Học sâu có một số thuộc tính chứng minh trạng thái của nó như một cuộc cách mạng AI và nó sẽ tồn tại ở đây. Những thuộc tính quan trọng này có thể được sắp xếp thành ba loại:

* *Simplicity* (tính đơn giản): Học sâu loại bỏ yêu cầu về kỹ thuật tính năng, thay thế các đường ống phức tạp, dễ hỏng, nặng về kỹ thuật bằng các mô hình đơn giản, có thể huấn luyện từ đầu đến cuối thường được xây dựng chỉ bằng năm hoặc sáu thao tác tensor khác nhau.
* *Scalability* (Khả năng mở rộng): Học sâu rất phù hợp với quá trình song song hóa trên GPU hoặc TPU, do đó, nó có thể tận dụng tối đa định luật Moore. Ngoài ra, các mô hình học sâu được đào tạo bằng cách lặp lại các lô dữ liệu nhỏ, cho phép chúng được đào tạo trên các bộ dữ liệu có kích thước tùy ý.
* *Versatility and reusability* (Tính linh hoạt và khả năng tái sử dụng): Không giống như nhiều phương pháp học máy trước đây, các mô hình học sâu có thể được đào tạo dựa trên dữ liệu bổ sung mà không cần bắt đầu lại từ đầu, khiến chúng trở nên khả thi để học trực tuyến liên tục - một thuộc tính quan trọng đối với các mô hình sản xuất rất lớn. Hơn nữa, các mô hình học sâu được đào tạo có thể tái sử dụng và do có thể tái sử dụng.

## Kết luận

Chúng ta đã tìm hiểu tổng quan về trí tuệ nhân tạo, học máy và học sâu.

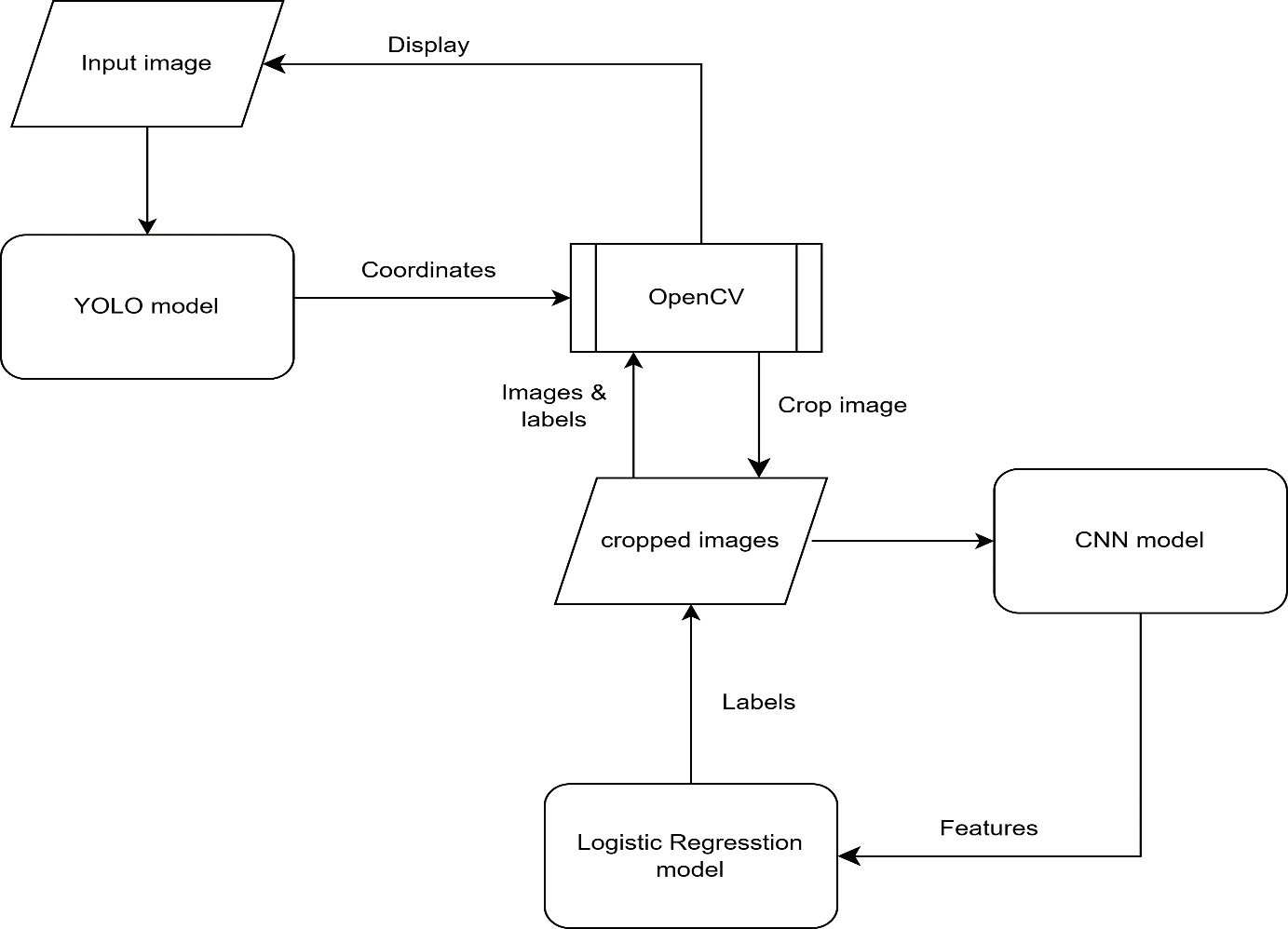
# MÔ HÌNH HỌC MÁY & CÔNG NGHỆ ỨNG DỤNG TRONG BÀI TOÁN PHÂN LOẠI HOA

## Kiến trúc tổng thể của ứng dụng

Ứng dụng phân loại hoa được xây dựng bằng việc áp dụng lần lượt các mô hình học máy, gồm:

* Mô hình YOLO
* Mạng thần kinh tích chập
* Mô hình Hồi quy Logistic

Ý tưởng được đưa ra là: Sử dụng mô hình **YOLO** để xác định *vùng chứa đối tượng hoa*, sau đó tiến hành bóc tách vùng chứa đối tượng hoa thành ảnh bằng **OpenCV**. Những ảnh này được chuyển qua **mạng thần kinh tích chập (CNN)** để *trích xuất* ra *những đặc trưng* của các đối tượng hoa trong ảnh. Cuối cùng là đưa những đặc trưng này qua mô hình **hồi quy Logistic** để tiến hành phân loại đối tượng hoa. Kết quả sau khi phân loại sẽ được đưa về hình ảnh ban đầu bằng **OpenCV** để hiển thị tên hoa do mô hình đã phân loại được.



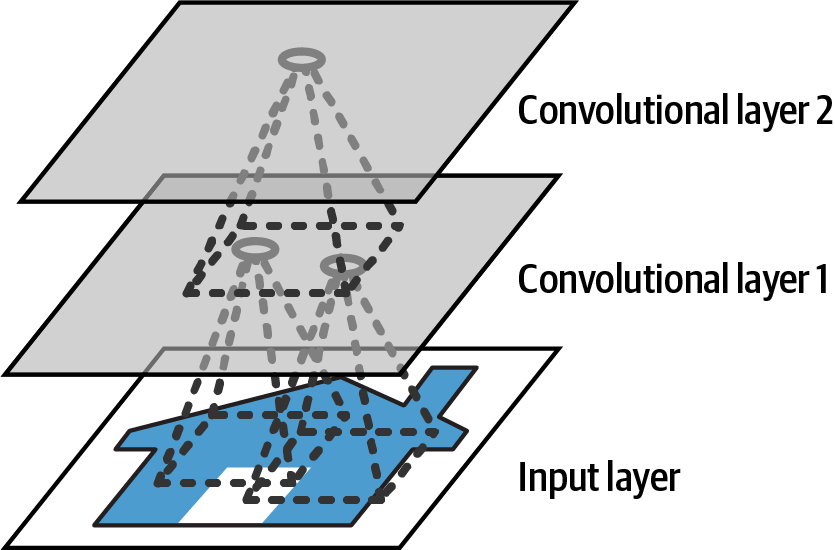
##### **Hình 2. 1** Minh hoạ về kiến trúc tổng thể của ứng dụng

## Mạng thần kinh tích chập (Convolutional neural network - CNN)

### Khái niệm

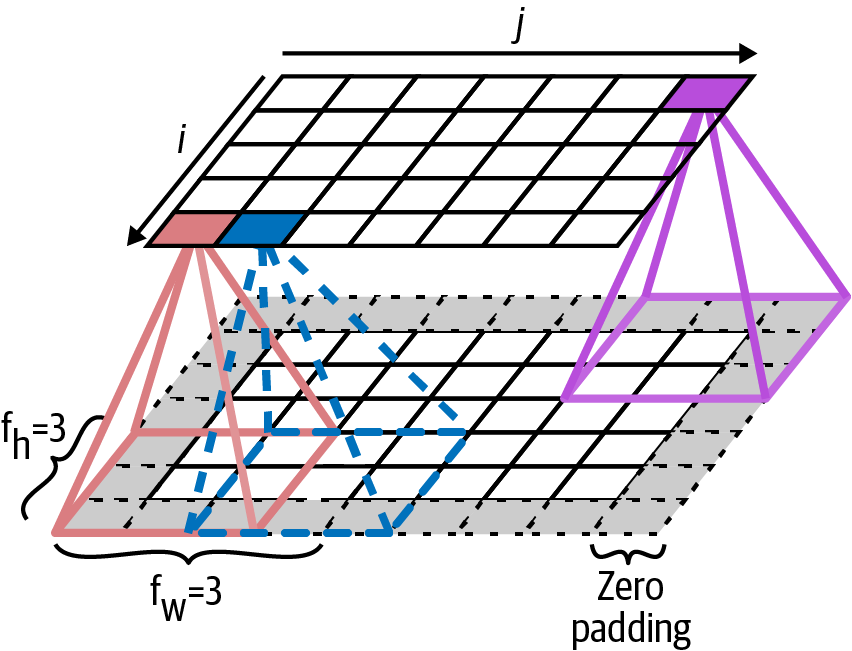
### Lớp tích chập (Convolutional Layer)

Khối xây dựng quan trọng nhất của CNN là *lớp tích chập*: nơ-ron trong lớp tích chập đầu tiên không được kết nối với mọi pixel đơn lẻ trong ảnh đầu vào, mà chỉ kết nối với các pixel trong trường tiếp nhận của chúng. Lần lượt, mỗi nơ-ron trong lớp tích chập thứ hai chỉ được kết nối với các nơ-ron nằm trong một hình chữ nhật nhỏ ở lớp đầu tiên. Kiến trúc này cho phép mạng tập trung vào các tính năng cấp thấp nhỏ trong lớp ẩn đầu tiên, sau đó tập hợp chúng thành các tính năng cấp cao hơn lớn hơn trong lớp ẩn tiếp theo, v.v. Cấu trúc phân cấp này phổ biến trong các hình ảnh trong thế giới thực, đó là một trong những lý do tại sao CNN hoạt động rất tốt để nhận dạng hình ảnh.



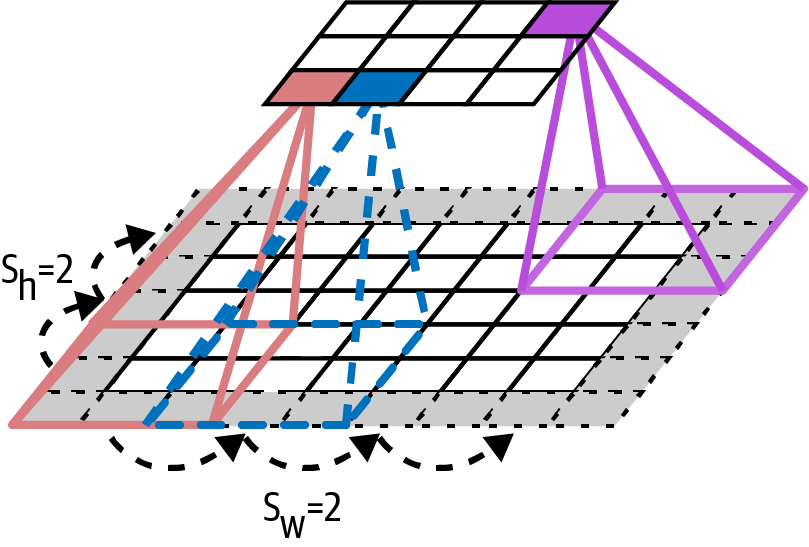
##### **Hình 2. 9** Các lớp CNN với các trường tiếp nhận cục bộ hình chữ nhật

Một nơ-ron nằm ở hàng *i*, cột *j* của một lớp nhất định được kết nối với đầu ra của các nơ-ron ở lớp trước nằm ở hàng *i* đến *i + fh* – 1, cột *j* đến *j + f*w – 1, trong đó *fh* và *fw* là chiều cao và chiều rộng của trường tiếp nhận. Để một lớp có cùng chiều cao và chiều rộng như lớp trước đó, người ta thường thêm các số 0 xung quanh các đầu vào, điều này được gọi là *phần đệm không*.



##### **Hình 2. 10** Kết nối giữa các lớp và phần đệm bằng không

Cũng có thể kết nối một lớp đầu vào lớn với một lớp nhỏ hơn nhiều bằng cách tách các trường tiếp nhận ra. Điều này làm giảm đáng kể độ phức tạp tính toán của mô hình. Kích thước bước ngang hoặc dọc từ trường tiếp nhận này sang trường tiếp theo được gọi là sải bước.

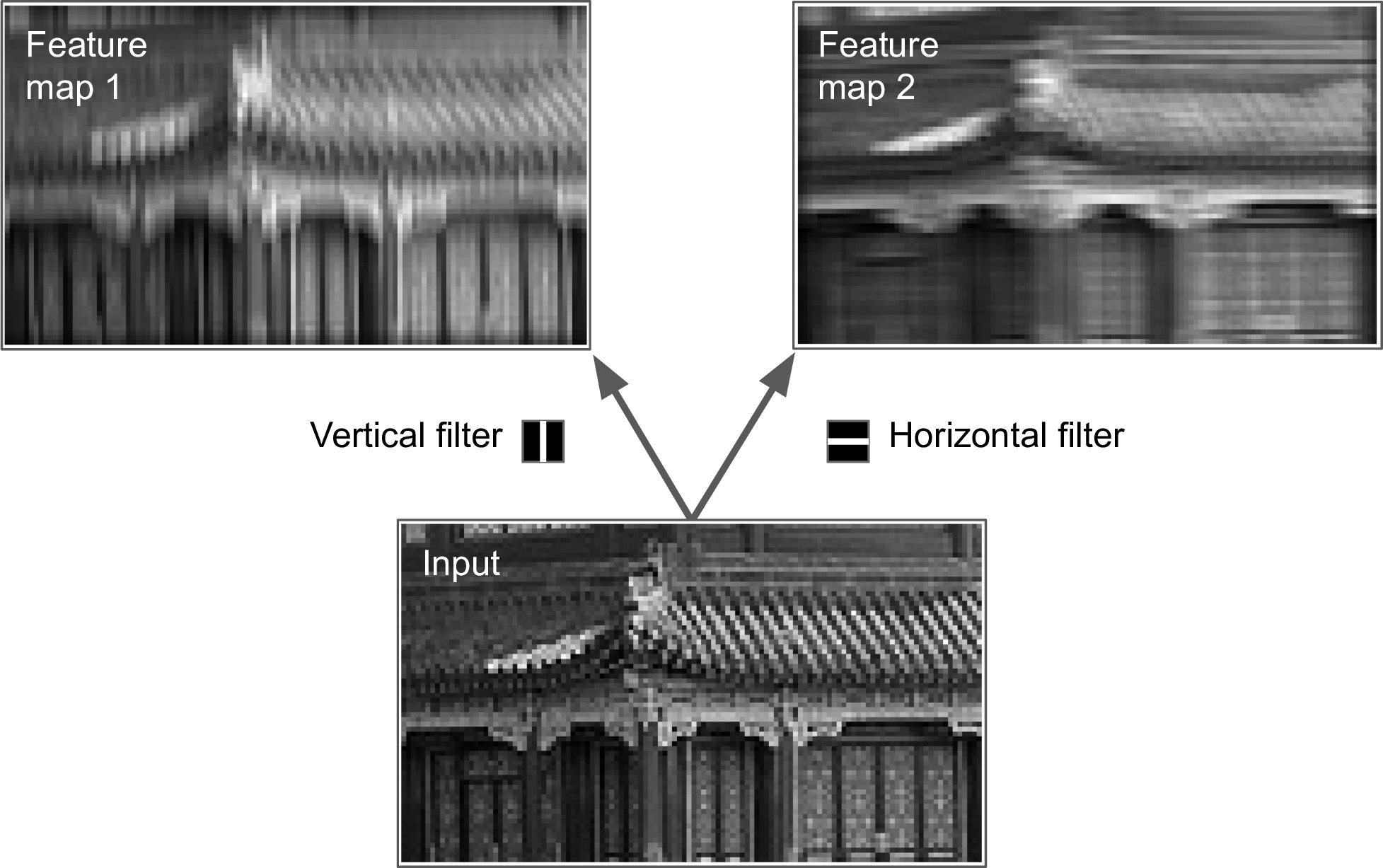


##### **Hình 2. 11** Giảm kích thước bằng cách sử dụng một bước 2

Trong hình trên, lớp đầu vào 5 × 7 (cộng với phần đệm bằng 0) được kết nối với lớp 3 × 4, sử dụng các trường tiếp nhận 3 × 3 và sải bước là 2. Một nơ ron nằm ở hàng *i*, cột *j* ở lớp bên trên được kết nối với đầu ra của các nơ ron trong lớp trước đó nằm trong các hàng *i × sh* đến i × *sh + fh* – 1, các cột *j* × *sw* đến *j × sw + fw* – 1, trong đó *sh* và *sw* là các bước dọc và ngang.

#### a. Bộ lọc

Trọng số của nơ-ron có thể được biểu diễn dưới dạng một hình ảnh nhỏ bằng kích thước của trường tiếp nhận, các bộ trọng số có thể có được gọi là *bộ lọc* (hoặc *nhân tích chập*, hoặc *nhân*).



##### **Hình 2. 12** Áp dụng hai bộ lọc khác nhau để có được hai bản đồ đặc trưng

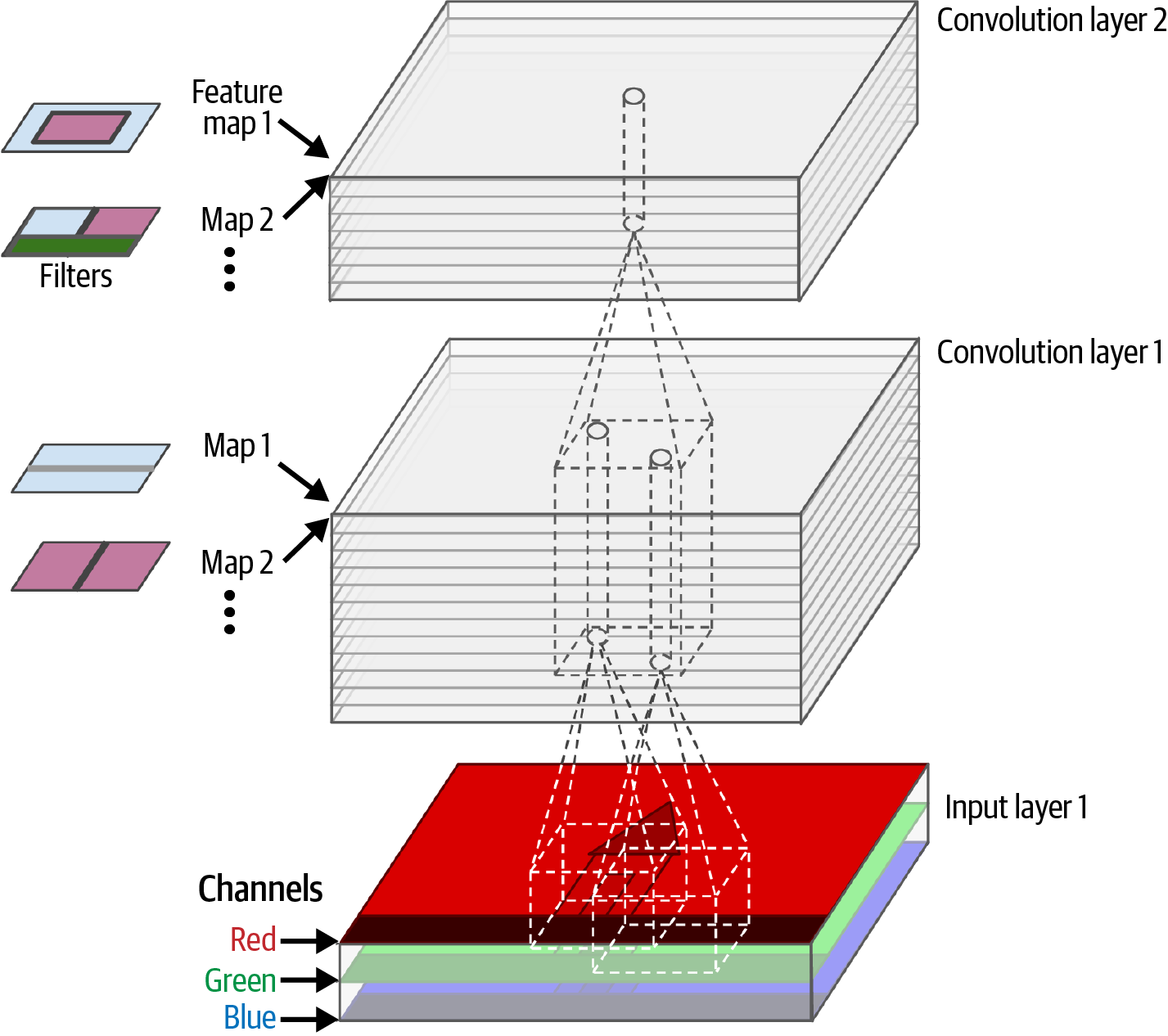
Bộ lọc đầu tiên được biểu diễn dưới dạng một hình vuông màu đen với một đường thẳng đứng màu trắng ở giữa (đó là ma trận 7 × 7 chứa đầy các số 0 ngoại trừ cột trung tâm chứa đầy các số 1); các nơ-ron sử dụng các trọng số này sẽ bỏ qua mọi thứ trong trường tiếp nhận của chúng ngoại trừ đường dọc trung tâm (vì tất cả các đầu vào sẽ được nhân với 0, ngoại trừ các đầu vào trong đường thẳng đứng trung tâm). Bộ lọc thứ hai là một hình vuông màu đen với một đường kẻ ngang màu trắng ở giữa. Các nơ ron sử dụng bộ trọng số này sẽ bỏ qua mọi thứ trong trường tiếp nhận của chúng ngoại trừ đường ngang trung tâm.

Nếu tất cả các nơ-ron trong một lớp sử dụng cùng một bộ lọc đường thẳng đứng (và cùng một thuật ngữ phân cực), và nếu ta cung cấp cho mạng hình ảnh đầu vào như hình dưới cùng, thì lớp sẽ xuất ra hình ảnh trên cùng bên trái: các đường trắng dọc được tăng cường trong khi phần còn lại bị mờ. Tương tự, hình ảnh phía trên bên phải là những gì nhận được nếu tất cả các nơ-ron sử dụng cùng một bộ lọc đường ngang: các đường màu trắng nằm ngang được tăng cường trong khi phần còn lại bị mờ. Do đó, một lớp chứa đầy các nơ-ron sử dụng cùng một bộ lọc sẽ xuất ra một *bản đồ đặc trưng*, bản đồ này làm nổi bật các vùng trong ảnh kích hoạt bộ lọc nhiều nhất. Trong quá trình đào tạo, lớp tích chập sẽ tự động tìm hiểu các bộ lọc hữu ích nhất cho tác vụ của nó và các lớp bên trên sẽ học cách kết hợp chúng thành các mẫu phức tạp hơn.

#### b. Xếp chồng nhiều bản đồ đặc trưng

Trên thực tế, một lớp tích chập có nhiều bộ lọc và xuất ra một bản đồ đặc trưng cho mỗi bộ lọc, vì vậy nó được biểu diễn chính xác hơn ở dạng 3D. Nó có một nơ-ron trên mỗi pixel trong mỗi bản đồ đặc trưng và tất cả các nơ-ron trong một bản đồ đặc trưng nhất định đều có chung các tham số (nghĩa là cùng nhân và thuật ngữ phân cực). Các nơ-ron trong các bản đồ đặc trưng khác nhau sử dụng các tham số khác nhau. Trường tiếp nhận của nơ-ron mở rộng trên tất cả các bản đồ đặc trưng của lớp trước. Tóm lại, một lớp tích chập đồng thời áp dụng nhiều bộ lọc có thể huấn luyện cho các đầu vào của nó, khiến nó có khả năng phát hiện nhiều tính năng ở bất kỳ đâu trong các đầu vào của nó.

Thực tế là tất cả các nơ-ron trong một bản đồ đặc trưng đều có chung các tham số làm giảm đáng kể số lượng tham số trong mô hình. Khi CNN đã học cách nhận dạng một mẫu ở một vị trí, nó có thể nhận ra mẫu đó ở bất kỳ vị trí nào khác. Ngược lại, một khi mạng thần kinh được kết nối đầy đủ đã học cách nhận dạng một mẫu ở một vị trí, thì nó chỉ có thể nhận ra mẫu đó ở vị trí cụ thể đó.

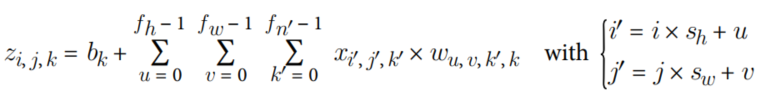


##### **Hình 2. 13** Hai lớp tích chập với nhiều bộ lọc mỗi lớp, xử lý ảnh màu với ba kênh màu

Hình ảnh đầu vào cũng bao gồm nhiều lớp con: một lớp trên mỗi kênh màu, thường có ba màu: đỏ, lục và lam (RGB). Hình ảnh thang độ xám chỉ có một kênh, nhưng một số hình ảnh có thể có nhiều kênh khác - ví dụ: hình ảnh vệ tinh thu được các tần số ánh sáng bổ sung (chẳng hạn như tia hồng ngoại).

Cụ thể, một nơ-ron nằm ở hàng *i*, cột *j* của bản đồ đặc trưng *k* trong một lớp tích chập nhất định *l* được kết nối với đầu ra của các nơ-ron ở lớp trước đó *l* – 1, nằm ở các hàng *i × sh* đến *i × sh + fh* – 1 và các cột *j × sw* đến *j × sw + fw* – 1, trên tất cả các bản đồ đặc trưng (trong lớp *l – 1*). Trong một lớp, tất cả các nơ-ron nằm trong cùng một hàng *i* và cột *j* nhưng trong các bản đồ đặc trưng khác nhau được kết nối với đầu ra của chính các nơ-ron đó trong lớp trước.

Phương trình . Tính toán đầu ra của một nơ ron nhất định trong lớp tích chập



Trong phươg trình này:

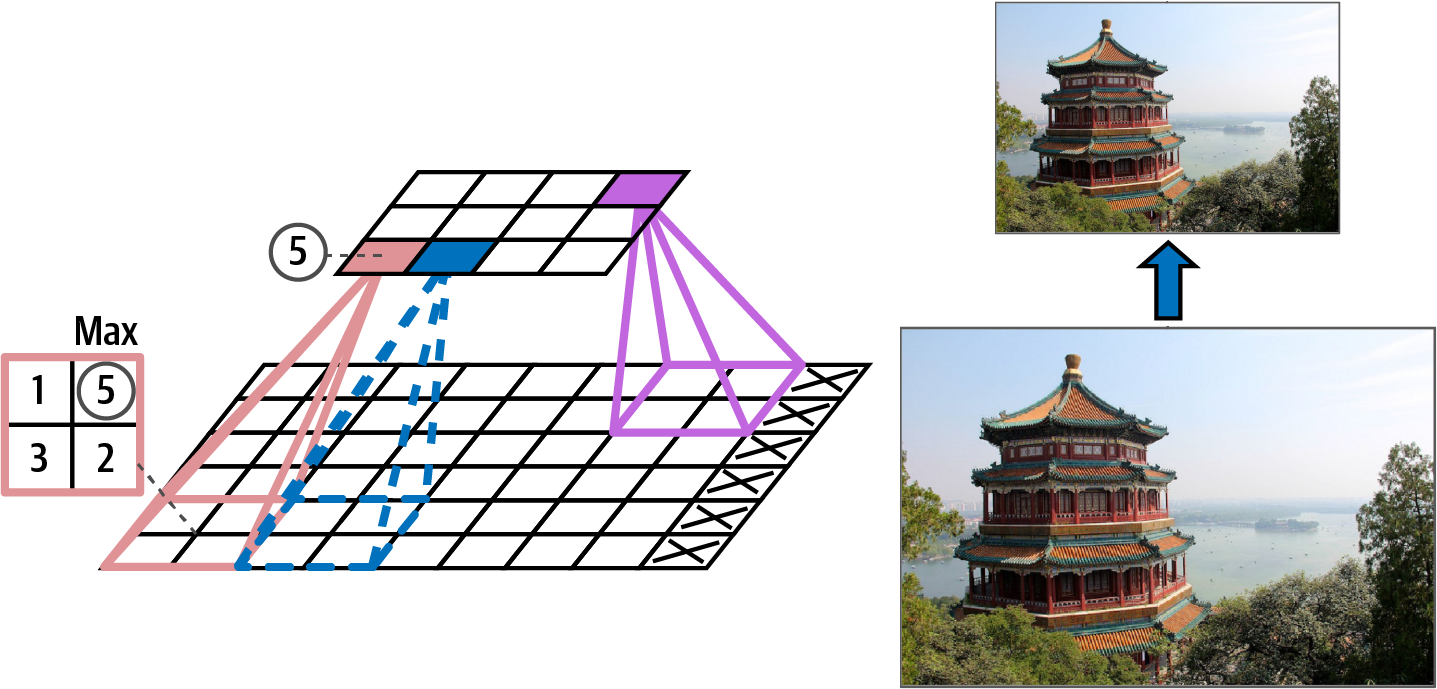
* *zi, j, k*  là đầu ra của nơ ron nằm ở hàng *i*, cột *j* trong bản đồ đặc trưng *k* của lớp tích chập (lớp *l*).
* *sh* và *sw* là các bước dọc và ngang, *fh* và *fw* là chiều cao và chiều rộng của trường tiếp nhận và *fn′* là số bản đồ đặc trưng trong lớp trước (lớp *l* – 1).
* xi′, j′, k′ là đầu ra của nơ-ron nằm trong lớp *l* – 1, hàng *i′*, cột *j′*, bản đồ đặc trưng *k′* (hoặc kênh *k′* nếu lớp trước đó là lớp đầu vào).
* *bk* là thuật ngữ phân cực cho bản đồ đặc trưng *k* (trong lớp *l*). Có thể coi nó như một núm điều chỉnh độ sáng tổng thể của bản đồ đặc trưng *k*.
* *wu, v, k′, k* là trọng số kết nối giữa bất kỳ nơ-ron nào trong bản đồ đặc trưng *k* của lớp *l* và đầu vào của nó ở hàng *u*, cột *v* (so với trường tiếp nhận của nơ-ron) và bản đồ đặc trưng *k′*.

Tất cả những gì phương trình trên thực hiện là tính tổng trọng số của tất cả các đầu vào, cộng với thuật ngữ phân cực.

### Lớp tổng hợp

Mục tiêu của *lớp tổng hợp* là lấy *mẫu con* (tức là thu nhỏ) hình ảnh đầu vào để giảm tải tính toán, mức sử dụng bộ nhớ và số lượng tham số (do đó hạn chế rủi ro quá khớp).

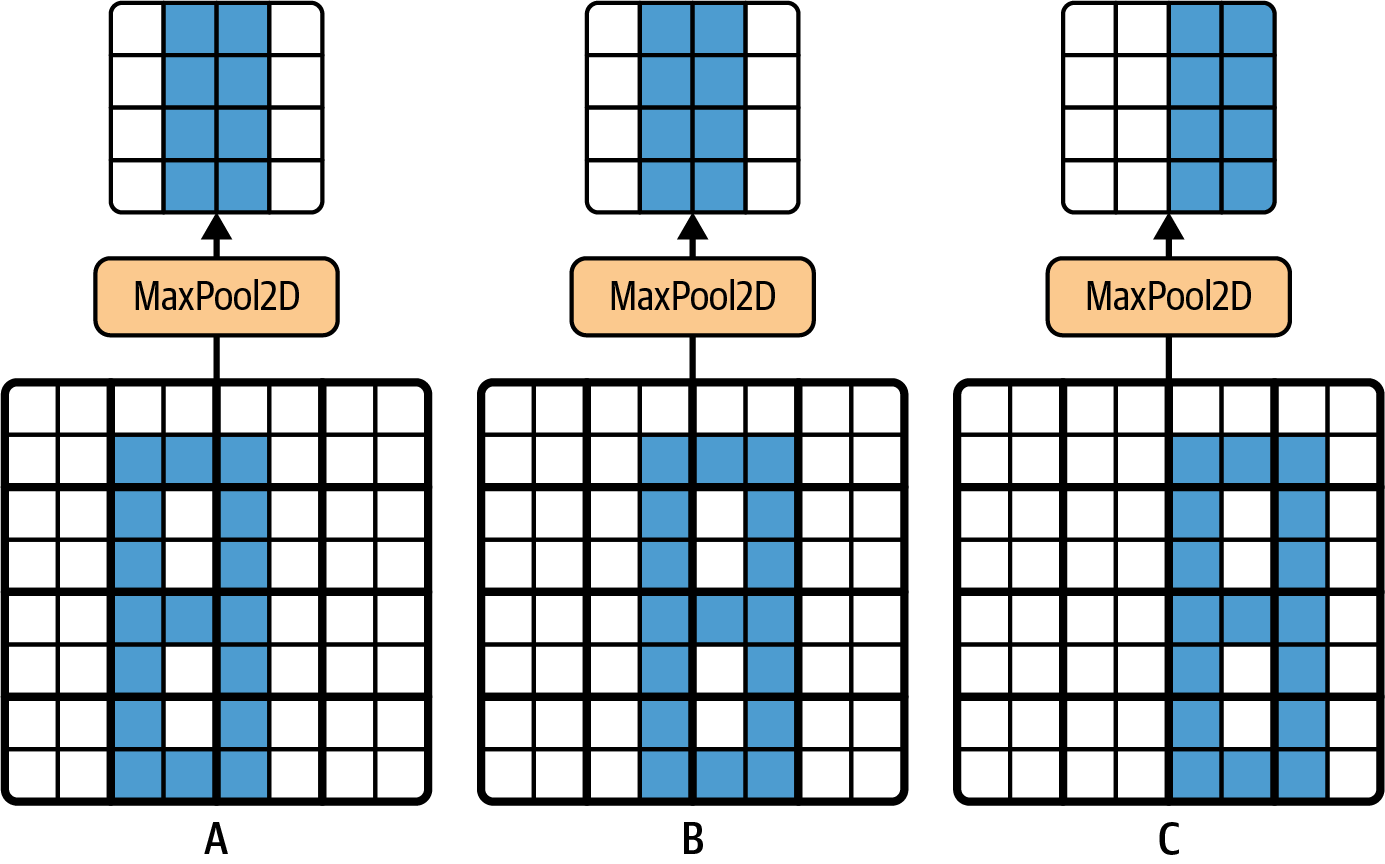
Giống như trong các lớp tích chập, mỗi nơ-ron trong lớp tổng hợp được kết nối với đầu ra của một số nơ-ron giới hạn ở lớp trước đó, nằm trong một trường tiếp nhận hình chữ nhật nhỏ. Ta phải xác định kích thước, sải bước và loại đệm. Tuy nhiên, một nơ-ron tổng hợp không có trọng số; tất cả những gì nó làm là tổng hợp các đầu vào bằng cách sử dụng hàm tổng hợp, chẳng hạn như giá trị tối đa hoặc giá trị trung bình. Lớp tổng hợp tối đa là loại lớp tổng hợp phổ biến nhất.



##### **Hình 2. 14** Lớp tổng hợp tối đa (nhân tổng hợp 2 × 2, sải bước 2, không có phần đệm)

Trong hình trên sử dụng nhân tổng hợp 2 × 2, với sải bước là 2 và không có phần đệm. Chỉ giá trị đầu vào tối đa trong mỗi trường tiếp nhận mới được chuyển sang lớp tiếp theo, trong khi các đầu vào khác bị loại bỏ. Trong trường tiếp nhận phía dưới bên trái trong hình trên, các giá trị đầu vào là 1, 5, 3, 2, vì vậy chỉ giá trị tối đa 5, được truyền tới lớp tiếp theo. Do sải bước là 2, hình ảnh đầu ra có một nửa chiều cao và một nửa chiều rộng của hình ảnh đầu vào (được làm tròn xuống vì không sử dụng phần đệm).

Khác với việc giảm tính toán, mức sử dụng bộ nhớ và số lượng tham số, lớp tổng hợp tối đa cũng đưa ra một số mức độ *bất biến* đối với các bản dịch nhỏ.

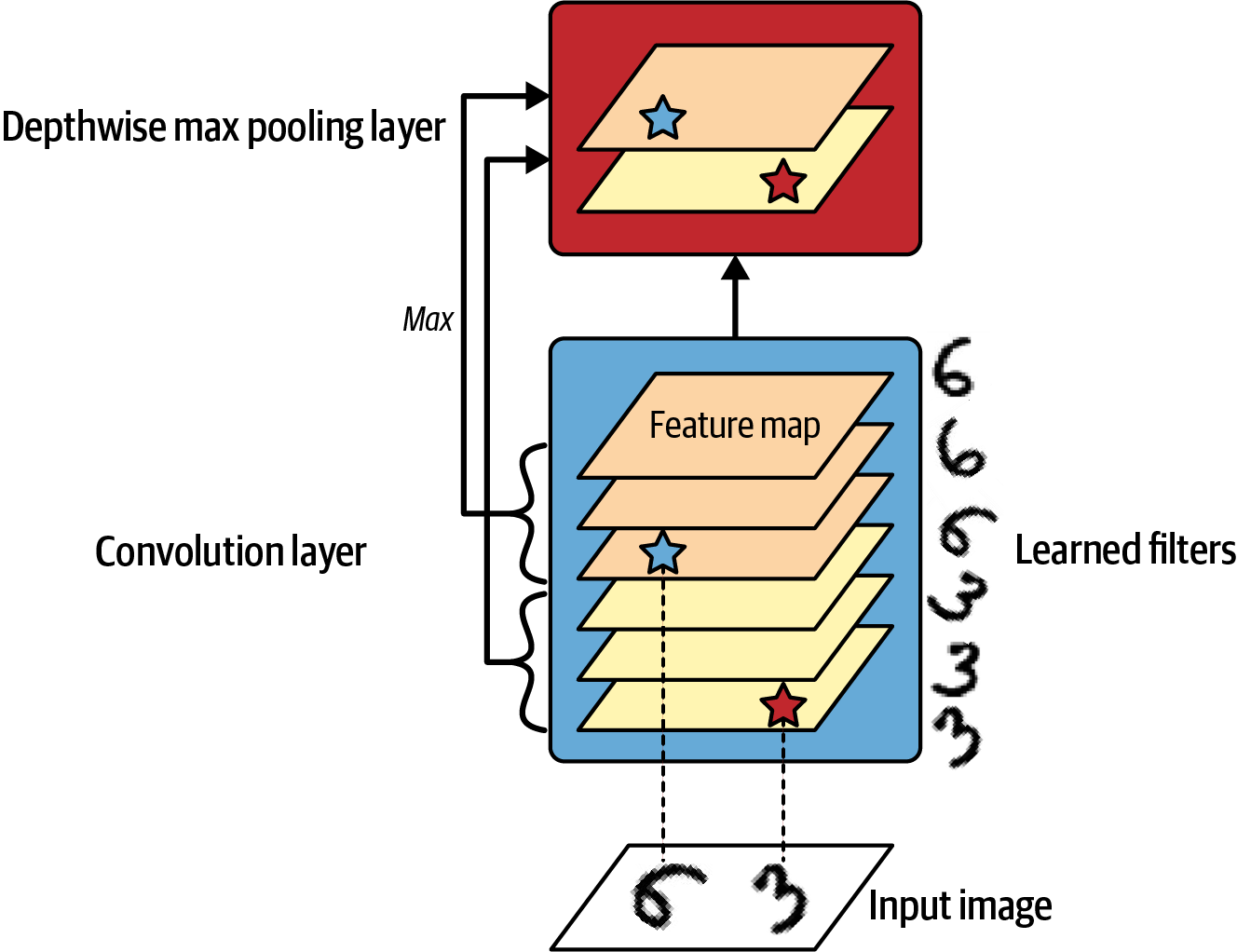


##### **Hình 2. 15** Bất biến đối với các bản dịch nhỏ

Ở đây, ta giả định rằng các pixel sáng có giá trị thấp hơn các pixel tối và xem xét ba hình ảnh (A, B, C) đi qua một lớp tổng hợp tối đa với nhân 2 × 2 và sải bước 2. Hình ảnh B và C giống nhau như hình A, nhưng dịch chuyển một và hai điểm ảnh sang bên phải. Ta có thể thấy, đầu ra của lớp tổng hợp tối đa cho hình ảnh A và B giống hệt nhau. Đây là ý nghĩa của bất biến dịch thuật. Đối với hình ảnh C, đầu ra khác: nó được dịch chuyển một pixel sang bên phải (nhưng vẫn có 50% bất biến). Bằng cách chèn một lớp tổng hợp tối đa cứ sau vài lớp trong CNN, có thể nhận được một số mức độ bất biến dịch thuật ở quy mô lớn hơn. Hơn nữa, tổng hợp tối đa cung cấp một lượng nhỏ bất biến quay và bất biến tỷ lệ nhỏ. Tính bất biến như vậy (ngay cả khi nó bị hạn chế) có thể hữu ích trong trường hợp dự đoán không phụ thuộc vào các chi tiết này, chẳng hạn như trong các nhiệm vụ phân loại.

Tuy nhiên, tổng hợp tối đa cũng có một số nhược điểm. Việc mất dữ liệu về thông tin ảnh rất lớn: ngay cả với nhân 2 × 2 nhỏ và bước tiến là 2, đầu ra sẽ nhỏ hơn hai lần theo cả hai hướng (vì vậy diện tích của nó sẽ nhỏ hơn bốn lần), giảm mất 75% giá trị đầu vào. Và trong một số ứng dụng, tính bất biến là không mong muốn. Thực hiện phân đoạn ngữ nghĩa (nhiệm vụ phân loại từng pixel trong một hình ảnh theo đối tượng mà pixel đó thuộc về): rõ ràng, nếu hình ảnh đầu vào được dịch bởi một pixel sang phải, đầu ra cũng cần được dịch một pixel sang bên phải. Mục tiêu trong trường hợp này là tương đương, không phải bất biến: một thay đổi nhỏ đối với đầu vào sẽ dẫn đến một thay đổi nhỏ tương ứng ở đầu ra.

Các lớp tổng hợp trung bình từng rất phổ biến, nhưng hiện tại mọi người chủ yếu sử dụng các lớp tổng hợp tối đa vì chúng thường hoạt động tốt hơn. Điều này có vẻ mâu thuẫn, vì tính toán giá trị trung bình thường mất ít thông tin hơn so với tính toán giá trị tối đa. Nhưng mặt khác, tổng hợp tối đa chỉ bảo tồn các đặc trưng mạnh nhất, loại bỏ tất cả những đặc trưng vô nghĩa, vì vậy các lớp tiếp theo sẽ nhận được tín hiệu rõ ràng hơn để hoạt động. Ngoài ra, tổng hợp tối đa mang lại tính bất biến dịch thuật mạnh hơn so với tổng hjpư trung bình và nó yêu cầu tính toán ít hơn. Cần chú ý rằng tổng hợp tối đa và tổng hợp trung bình có thể được thực hiện dọc theo chiều sâu thay vì các chiều không gian, mặc dù nó không phổ biến. Điều này có thể cho phép CNN học cách bất biến đối với các đặc trưng khác nhau. Ví dụ: nó có thể học nhiều bộ lọc, mỗi bộ lọc phát hiện một vòng quay khác nhau của cùng một mẫu (chẳng hạn như các chữ số viết tay) và lớp tổng hợp tối đa theo chiều sâu sẽ đảm bảo rằng đầu ra giống nhau bất kể góc quay. Tương tự như vậy, CNN có thể học cách bất biến đối với bất kỳ thứ gì: độ dày, độ sáng, độ lệch, màu sắc, v.v.

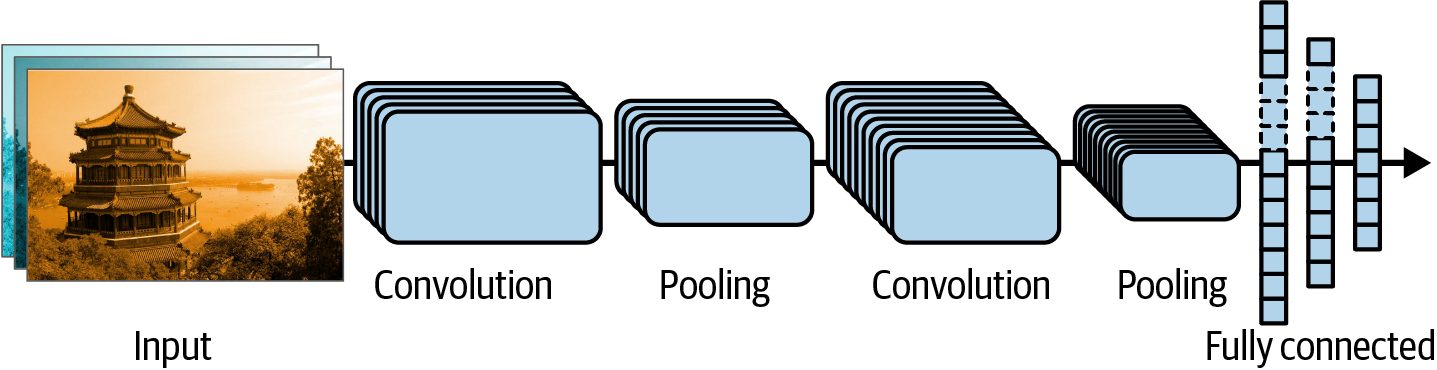


##### **Hình 2. 16** Tổng hợp tối đa theo chiều sâu có thể giúp CNN học cách bất biến (để xoay vòng trong trường hợp này)

Một loại lớp tổng hợp cuối cùng thường thấy trong các kiến trúc hiện đại là *lớp tổng hợp trung bình toàn cục*. Nó hoạt động rất khác: tất cả những gì nó làm là tính toán giá trị trung bình của từng bản đồ tính năng (nó giống như một lớp tổng hợp trung bình sử dụng nhân tổng hợp có cùng kích thước không gian như đầu vào). Điều này có nghĩa là nó chỉ xuất ra một số duy nhất cho mỗi bản đồ tính năng và mỗi thể hiện. Mặc dù điều này là cực kỳ mất mát thông tin (hầu hết thông tin trong bản đồ đặc trưng bị mất), nhưng nó có thể hữu ích ngay trước lớp đầu ra.

### Kiến trúc mạng CNN

Các kiến trúc CNN điển hình xếp chồng một vài lớp tích chập (mỗi lớp thường được theo sau bởi một lớp ReLU), sau đó là một lớp tổng hợp, rồi một vài lớp tích chập khác (+ReLU), sau đó là một lớp tổng hợp khác, v.v. Hình ảnh ngày càng nhỏ hơn khi nó đi qua mạng, nhưng nó cũng thường ngày càng sâu hơn (nghĩa là có nhiều bản đồ đặc trưng hơn), nhờ các lớp tích chập.



##### **Hình 2. 17** Kiến trúc CNN điển hình

Ở đầu ngăn xếp, một mạng nơ-ron chuyển tiếp thông thường được thêm vào, bao gồm một vài lớp được kết nối đầy đủ (+ReLU) và lớp cuối cùng đưa ra dự đoán (ví dụ: lớp softmax đưa ra xác suất lớp ước tính).

Một sai lầm phổ biến là sử dụng hạt nhân tích chập quá lớn. Ví dụ: thay vì sử dụng một lớp tích chập với nhân 5 × 5, hãy xếp chồng hai lớp với nhân 3 × 3: nó sẽ sử dụng ít tham số hơn và yêu cầu ít tính toán hơn, đồng thời thường sẽ hoạt động tốt hơn. Một ngoại lệ dành cho lớp tích chập đầu tiên: nó thường có thể có nhân lớn (ví dụ: 5 × 5), thường có sải bước từ 2 trở lên. Điều này sẽ làm giảm kích thước không gian của hình ảnh mà không làm mất quá nhiều thông tin và vì hình ảnh đầu vào nói chung chỉ có ba kênh nên sẽ không quá tốn kém.

Qua nhiều năm, các biến thể của kiến trúc cơ bản này đã được phát triển, dẫn đến những tiến bộ đáng kinh ngạc trong lĩnh vực này. Một thước đo tốt cho tiến trình này là tỷ lệ lỗi trong các cuộc thi như thử thách ILSVRC ImageNet. Trong cuộc thi này, 5 tỷ lệ lỗi hàng đầu để phân loại hình ảnh - tức là số lượng hình ảnh thử nghiệm mà 5 dự đoán hàng đầu của hệ thống không bao gồm câu trả lời đúng - đã giảm từ hơn 26% xuống dưới 2,3% chỉ trong sáu năm. Hình ảnh khá lớn (ví dụ: cao 256 pixel) và có 1.000 lớp. Xem xét sự phát triển của các mục chiến thắng là một cách hay để hiểu cách thức hoạt động của CNN và tiến trình nghiên cứu về học sâu.

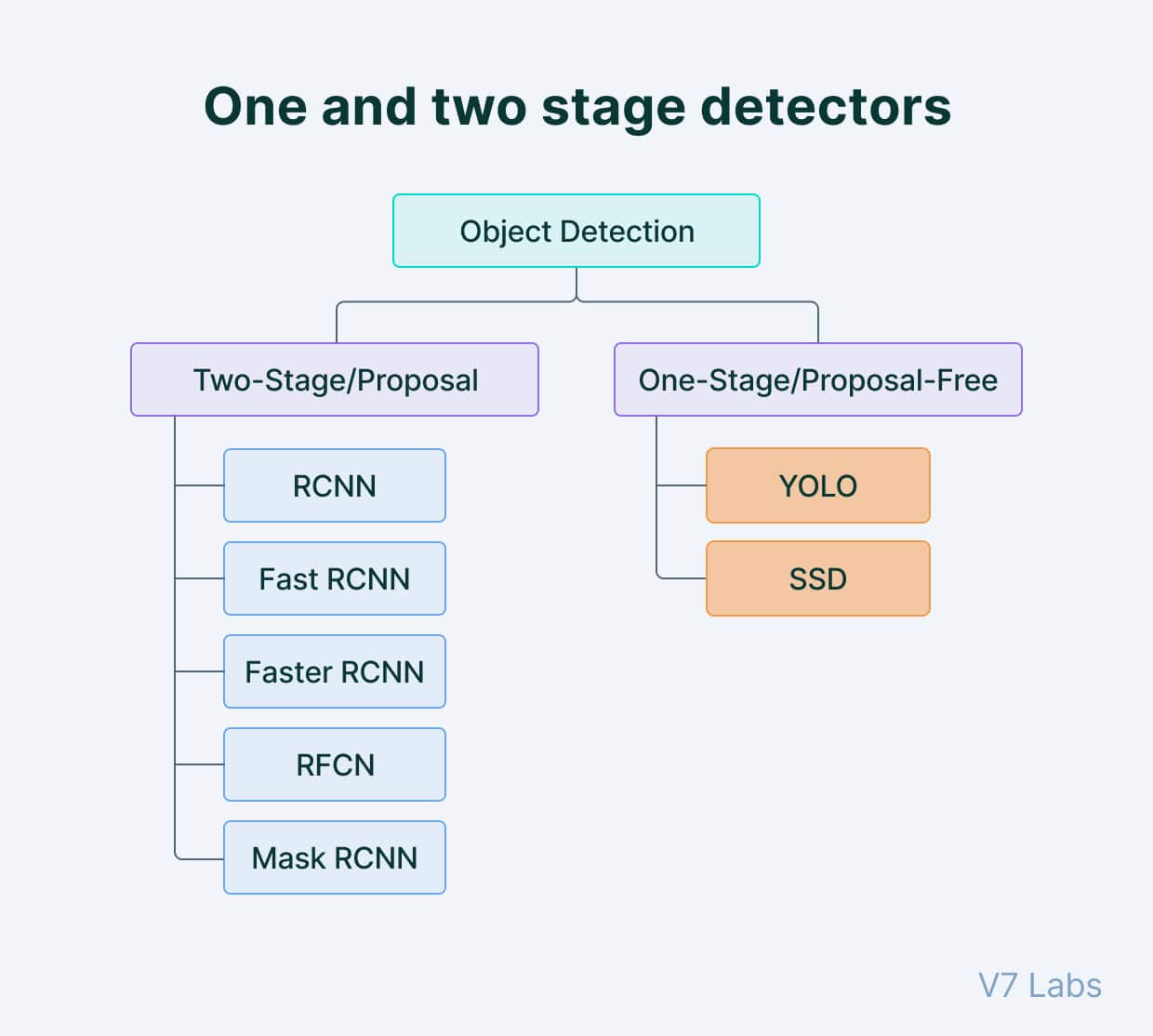
## Bài toán phát hiện đối tượng

### Khái niệm

Phát hiện đối tượng (*Object detection*) là một nhiệm vụ thuộc lĩnh vực *Computer vision* (Thị giác máy tính) liên quan đến việc xác định và định vị các đối tượng trong ảnh hoặc video [[?](https://www.v7labs.com/blog/object-detection-guide)]. Nó là một trong những thành tựu lớn nhất của việc học sâu và xử lý ảnh. Chúng ta có thể đào tạo một mô hình phát hiện đối tượng để xác định và phát hiện nhiều hơn một đối tượng cụ thể, vì vậy nó rất linh hoạt.[[?](https://neptune.ai/blog/object-detection-algorithms-and-libraries)]

### Các thuật toán phát hiện đối tượng

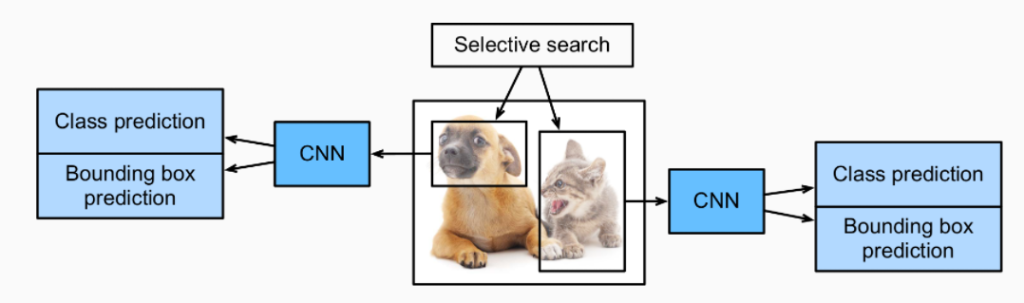
Kể từ khi học sâu phổ biến vào đầu những năm 2010, đã có sự tiến triển và cải thiện liên tục về chất lượng của các thuật toán được sử dụng để giải quyết phát hiện đối tượng. Các thuật toán phổ biến nhất hiện tại sẽ được mô tả bên dưới.



##### **Hình 2. 2** Phân loại các thuật toán phát hiện đối tượng

#### a. Mạng nơ-ron tích chập theo vùng (R-CNN)

Mạng thần kinh tích chập theo vùng (*region-based convolutional neural networks* – R-CNN) là một cải tiến mới trong kỹ thuật phát hiện đối tượng từ các phương pháp trước đây của HOG và SIFT. Trong các mô hình R-CNN thường trích xuất các đặc trưng cần thiết nhất của đối tượng (thường là khoảng 2000 đặc trưng) bằng cách sử dụng 1 giải thuật chọn lọc (*selective search*).[]



##### **Hình 2. 3** Chu trình phát hiện đối tượng vớiR-CNN. “Nguồn: [d2l.ai](https://d2l.ai/chapter_computer-vision/rcnn.html)”

Với R-CNN, việc trích xuất các vùng đề xuất khu vực được thực hiện thông qua thuật toán Selective Search để trích chọn ra các vùng có khả năng chứa đối tượng (khoảng 2000 vùng). Sau đó, các vùng (ảnh) này được thay đổi kích thước về 1 kích thước cố định và đưa qua 1 mô hình CNN được đào tạo trước, rồi từ đó tiến hành xác định offset và nhãn đối tượng[.[?]](https://vinbigdata.com/camera-ai/gioi-thieu-mot-so-thuat-toan-phat-hien-doi-tuong-pho-bien.html)

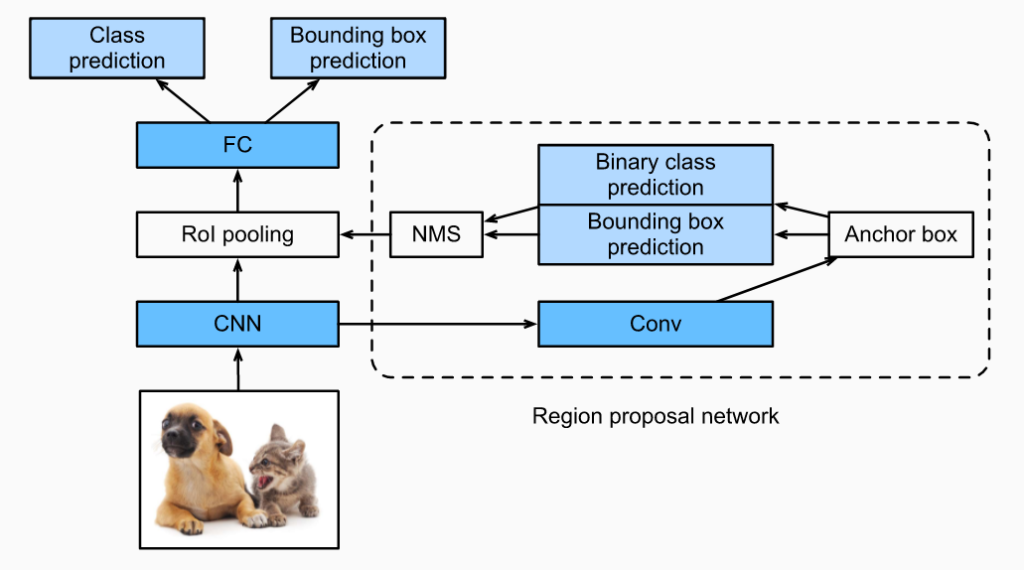
Mặc dù tạo ra kết quả hiệu quả để trích xuất tính năng với các mô hình CNN được đào tạo trước, quy trình trích xuất tổng thể của tất cả các đề xuất khu vực và cuối cùng là các vùng tốt nhất với các thuật toán hiện tại, là cực kỳ chậm. Một nhược điểm lớn khác của mô hình R-CNN không chỉ là tốc độ đào tạo chậm mà còn cả thời gian dự đoán cao. Giải pháp đòi hỏi phải sử dụng các tài nguyên tính toán lớn, tăng tính khả thi chung của quy trình. Do đó, kiến ​​trúc tổng thể có thể được coi là khá tốn kém. Đôi khi, các lựa chọn ứng cử viên xấu có thể xảy ra ở bước đầu tiên do thiếu những cải tiến có thể được thực hiện trong bước cụ thể này. Rất nhiều vấn đề trong mô hình được đào tạo có thể được gây ra bởi điều này. [[?](https://neptune.ai/blog/object-detection-algorithms-and-libraries)]

#### b. Faster R-CNN

Mặc dù mô hình R-CNN có thể thực hiện tính toán phát hiện đối tượng và đạt được hiệu quả nhưng vẫn tồn đọng nhiều điểm bất cập, điển hình là tốc độ mô hình. Vì vậy, một số phương pháp đã được phát triển để giải quyết vấn đề này và xử lý các nhược điểm của R-CNN. Trong số đó nổi bật nhất chính là mô hình *Fast R-CNN* và *Faster R-CNN*.[?]

Đầu tiên, Fast R-CNN đã được giới thiệu để chống lại một số vấn đề tồn tại từ trước của R-CNN. Trong phương pháp Fast R-CNN, toàn bộ hình ảnh được truyền qua mạng lưới thần kinh tích chập được đào tạo trước thay vì xem xét tất cả các phân đoạn phụ. Tập hợp vùng quan tâm (*Region of interest* - RoI) là một phương pháp đặc biệt: lấy hai đầu vào của mô hình được đào tạo trước và thuật toán tìm kiếm có chọn lọc để cung cấp một lớp được kết nối đầy đủ với đầu ra.[?]

Với Faster-RCNN, thay vì việc sử dụng Selective Search, mô hình được thiết kế thêm 1 mạng con gọi là RPN (*Region Proposal Network* – Mạng lưới đề xuất khu vực) để trích rút các vùng có khả năng chứa đối tượng của ảnh. Nhìn chung, sau khi thực hiện RPN, các bước xử lý sau tương tự như Fast-RCNN nhưng nhanh hơn nhiều (vì không sử dụng Selective Search) và được thiết kế như 1 mạng có thể đào tạo từ đầu tới cuối.



##### **Hình 2. 4** Mô hình Faster R-CNN. “Nguồn: d2l.ai”

Mô hình Faster R-CNN là một trong những phiên bản tốt nhất trong Fast R-CNN và cải thiện tốc độ hoạt động rất nhiều so với các phiên bản tiền nhiệm vì sử dụng RPN.

Mạng RPN giảm thời gian tính toán để trích chọn đặc trưng, thường là 10ms cho mỗi hình ảnh. Mạng này bao gồm các lớp tích hợp mà từ đó chúng ta có thể thu được các đặc trưng cần thiết thông qua từng lớp tích chập liên tiếp nhau. Để đưa ra các vùng đặc trưng, chúng ta sử dụng các hộp neo (*anchor box*) với các tỉ lệ, kích thước và độ lớn khác nhau. Đối với mỗi anchor box tại RPN, chúng ta thực hiện 1 bộ phân loại nhị phân để phân loại vùng trích chọn đó có khả năng chứa đối tượng hay không, và dự đoán ra các hộp giới hạn (*bounding box*) tương ứng.

Sau đó, các vùng trích chọn sẽ được đưa qua 1 bộ lọc gọi là *Non maximum suppression* (NMS) để loại bỏ các bounding box dư thừa. Đầu ra của NMS được cho qua 1 lớp gọi là *RoI Align* (Căn chỉnh RoI) để cố định kích thước đầu ra của các vùng đặc trưng đã trích chọn được. Sau đó, phần xử lý tiếp theo của mô hình sẽ tương tự như mô hình Fast R-CNN.[?]

Một trong những hạn chế chính của phương pháp Faster R-CNN là lượng thời gian trễ trong đề xuất của các đối tượng khác nhau. Đôi khi, tốc độ phụ thuộc vào loại hệ thống đang được sử dụng.

Mask R-CNN bổ sung nhánh dự đoán mặt nạ trên Faster R-CNN, có thể phát hiện các vật thể và dự đoán mặt nạ của chúng cùng một lúc.[[?](https://www.v7labs.com/blog/object-detection-guide)]

R-FCN thay thế các lớp được kết nối đầy đủ bằng các bản đồ điểm nhạy cảm với vị trí để phát hiện các đối tượng tốt hơn.[[?](https://www.v7labs.com/blog/object-detection-guide)]

#### c. Máy dò một lần (SSD)

Máy dò một lần (*Single Shot Detector* - SSD) cho các dự đoán đa hộp là một trong những cách nhanh nhất để đạt được tính toán thời gian thực của các tác vụ phát hiện đối tượng. Mặc dù các phương pháp Faster R-CNN có thể đạt được độ chính xác dự đoán cao, nhưng toàn bộ quá trình này khá tốn thời gian và nó yêu cầu tác vụ thời gian thực chạy ở tốc độ khoảng 7 khung hình mỗi giây, đây là điều không mong muốn.[]

SSD giải quyết vấn đề này bằng cách cải thiện số khung hình mỗi giây lên gần gấp năm lần so với kiểu Faster R-CNN. Nó loại bỏ việc sử dụng mạng đề xuất khu vực và thay vào đó sử dụng các tính năng đa tỷ lệ và hộp mặc định.[]



##### **Hình 2. 5** Kiến trúc SSD. “Nguồn: [arxiv.org](https://arxiv.org/pdf/1512.02325.pdf)”

Kiến trúc máy dò đa hộp một lần có thể được chia thành ba thành phần chủ yếu. Giai đoạn đầu tiên của bộ phát hiện một lần là bước trích xuất tính năng, trong đó tất cả các bản đồ tính năng quan trọng được chọn. Vùng kiến trúc này chỉ bao gồm các lớp tích chập đầy đủ và không có lớp nào khác. Sau khi trích xuất tất cả các bản đồ tính năng cần thiết, bước tiếp theo là quá trình phát hiện các đầu. Bước này cũng bao gồm các mạng thần kinh tích chập hoàn toàn.[]

Tuy nhiên, trong giai đoạn thứ hai của đầu phát hiện, nhiệm vụ không phải là tìm ý nghĩa ngữ nghĩa cho hình ảnh. Thay vào đó, mục tiêu chính là tạo các bản đồ giới hạn phù hợp nhất cho tất cả các bản đồ đối tượng. Giai đoạn cuối cùng là chuyển nó qua các lớp triệt tiêu không tối đa để giảm tỷ lệ lỗi do các hộp giới hạn lặp lại gây ra.[?]

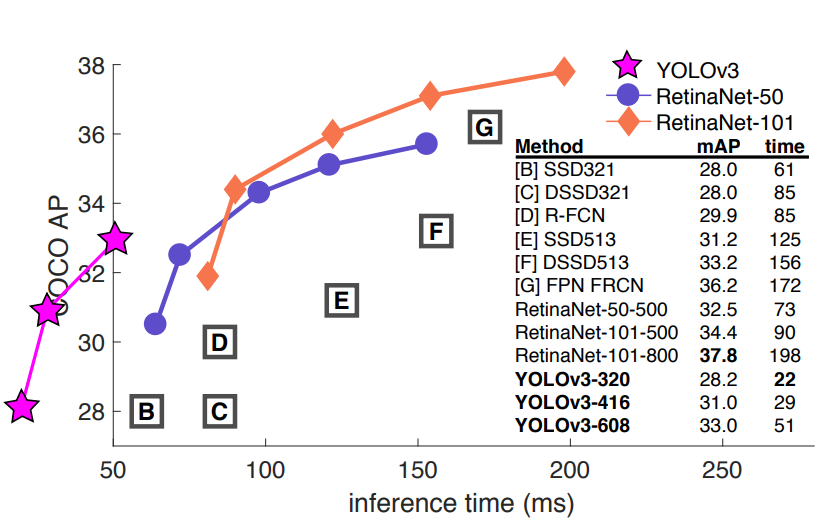
SSD làm giảm độ phân giải của hình ảnh xuống chất lượng thấp hơn.[]

Kiến trúc SSD thường sẽ hoạt động kém hơn Faster R-CNN đối với các đối tượng quy mô nhỏ.[]

#### e. YOLO

**Khái niệm:**

You only look once (YOLO) là một trong những kiến trúc mô hình và thuật toán phát hiện đối tượng phổ biến nhất. Thông thường, khái niệm đầu tiên được tìm thấy khi tìm kiếm thuật toán phát hiện đối tượng trên Google là kiến trúc YOLO. Mô hình YOLO sử dụng một trong những nguyên mẫu mạng thần kinh tốt nhất để tạo ra độ chính xác cao và tốc độ xử lý tổng thể. Tốc độ và độ chính xác này là lý do chính cho sự phổ biến của nó.[[?](https://neptune.ai/blog/object-detection-algorithms-and-libraries)]

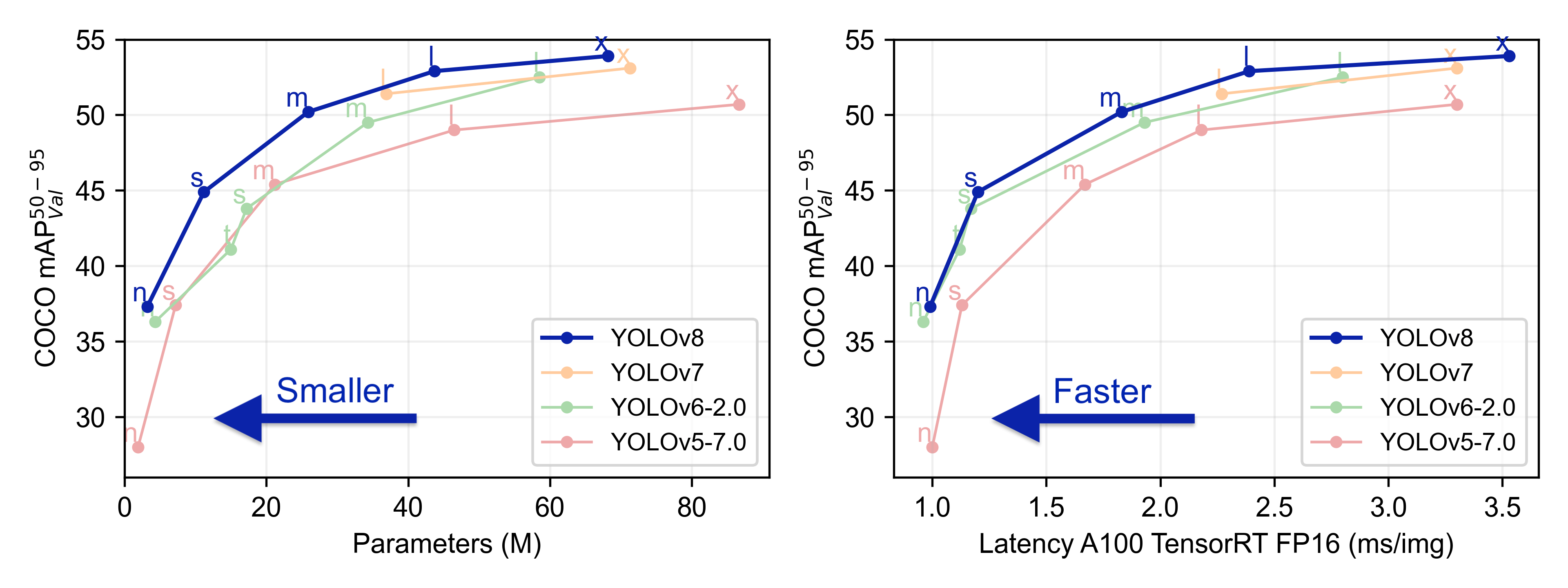


##### **Hình 2. 6** So sánh tốc độ của YOLOv3 với các mô hình khác. “Nguồn: [pjreddie.com](https://pjreddie.com/media/files/papers/YOLOv3.pdf)”

**Tóm tắt lịch sử phát triển**:

YOLO được phát triển bởi Joseph Redmon và Ali Farhadi tại Đại học Washington. Ra mắt vào năm 2015, YOLO nhanh chóng trở nên nổi tiếng nhờ tốc độ và độ chính xác cao. []

* YOLOv2, được phát hành vào năm 2016, đã cải thiện mô hình ban đầu bằng cách kết hợp chuẩn hóa hàng loạt, hộp neo và cụm kích thước.[]
* YOLOv3, ra mắt vào năm 2018, đã nâng cao hơn nữa hiệu suất của mô hình bằng cách sử dụng mạng đường trục hiệu quả hơn, nhiều neo và tổng hợp kim tự tháp không gian.[]
* YOLOv4 được phát hành vào năm 2020, giới thiệu những cải tiến như tăng cường dữ liệu Mosaic, đầu phát hiện không neo mới và chức năng mất mát mới.[]
* YOLOv5 tiếp tục cải thiện hiệu suất của mô hình và thêm các tính năng mới như tối ưu hóa siêu tham số, theo dõi thử nghiệm tích hợp và xuất tự động sang các định dạng xuất phổ biến.[]
* YOLOv6 được Meituan phát hành mã nguồn mở vào năm 2022 và đang được sử dụng trong nhiều robot giao hàng tự động của công ty.[]
* YOLOv7 đã thêm các tác vụ bổ sung như ước tính tư thế trên bộ dữ liệu điểm chính COCO.[]
* YOLOv8 là phiên bản mới nhất của YOLO được phát triển bởi Ultralytics. Là một mô hình tiên tiến, hiện đại (SOTA), YOLOv8 được xây dựng dựa trên sự thành công của các phiên bản trước, giới thiệu các tính năng và cải tiến mới để nâng cao hiệu suất, tính linh hoạt và hiệu quả. YOLOv8 hỗ trợ đầy đủ các tác vụ AI tầm nhìn, bao gồm phát hiện, phân đoạn, ước tính tư thế, theo dõi và phân loại. Tính linh hoạt này cho phép người dùng tận dụng khả năng của YOLOv8 trên các ứng dụng và miền đa dạng.[]



##### **Hình 2. 7** Hiệu suất của YOLOv8. “Nguồn: [github.com](https://github.com/ultralytics/ultralytics)”

**Kiến trúc và cách hoạt động:**



##### **Hình 2. 8** Kiến trúc phát hiện đối tượng của YOLO. “Nguồn: [arxiv.org](https://arxiv.org/pdf/1506.02640.pdf)”

Mạng phát hiện đối tượng của YOLO gồm 24 lớp tích chập theo sau là 2 lớp được kết nối đầy đủ. Các lớp tích chập 1x1 xen kẽ làm giảm không gian tính năng từ các lớp trước đó. []

20 lớp tích chập đầu tiên của mô hình được đào tạo trước trên nhiệm phụ phân loại ImageNet nhờ việc cắm vào một lớp gộp trung bình tạm thời và lớp được kết nối đầy đủ. Sau đó, mô hình được đào tạo trước này được chuyển đổi để thực hiện phát hiện vì nghiên cứu trước đây cho thấy rằng việc thêm các lớp tích chập và được kết nối vào mạng được đào tạo trước sẽ cải thiện hiệu suất. Lớp kết nối đầy đủ cuối cùng của YOLO dự đoán cả xác suất lớp và tọa độ hộp giới hạn.[]

YOLO chia ảnh đầu vào thành lưới S × S. Nếu tâm của một đối tượng rơi vào ô lưới, ô lưới đó có nhiệm vụ phát hiện đối tượng đó. Mỗi ô lưới dự đoán các hộp giới hạn B và điểm tin cậy cho các hộp đó. Những điểm tin cậy này phản ánh mức độ tin tưởng của mô hình rằng hộp chứa một đối tượng và mức độ chính xác mà nó nghĩ rằng hộp dự đoán là như thế nào.[]

YOLO dự đoán nhiều hộp giới hạn trên mỗi ô lưới. Tại thời điểm đào tạo, chỉ một công cụ dự đoán hộp giới hạn đảm nhận nhiệm vụ cho mỗi đối tượng. YOLO chỉ định một công cụ dự đoán "có nhiệm vụ" dự đoán một đối tượng dựa trên dự đoán nào có IOU hiện tại cao nhất với sự thật cơ bản. Điều này dẫn đến sự chuyên môn hóa giữa các bộ dự đoán hộp giới hạn. Mỗi công cụ dự đoán trở nên tốt hơn trong việc dự báo kích thước, tỷ lệ khung hình hoặc lớp đối tượng nhất định, cải thiện điểm recall tổng thể.[]

Một kỹ thuật quan trọng được sử dụng trong các mô hình YOLO là triệt tiêu không tối đa (*non-maximum suppression* - NMS). NMS là một bước xử lý hậu kỳ được sử dụng để cải thiện độ chính xác và hiệu quả của việc phát hiện đối tượng. Trong phát hiện đối tượng, thông thường nhiều hộp giới hạn được tạo cho một đối tượng trong một hình ảnh. Các hộp giới hạn này có thể chồng lên nhau hoặc được đặt ở các vị trí khác nhau, nhưng tất cả chúng đều đại diện cho cùng một đối tượng. NMS được sử dụng để xác định và loại bỏ các hộp giới hạn dư thừa hoặc không chính xác và để xuất ra một hộp giới hạn duy nhất cho mỗi đối tượng trong hình ảnh.[]

**Ưu điểm:**

* Tốc độ tính toán và xử lý của YOLO khá cao, đặc biệt là trong thời gian thực so với hầu hết các phương pháp đào tạo và thuật toán phát hiện đối tượng khác.[]
* Ngoài tốc độ tính toán nhanh, thuật toán YOLO còn quản lý để cung cấp độ chính xác tổng thể cao với việc giảm các lỗi nền thường thấy trong các phương pháp khác.[]
* Kiến trúc của YOLO cho phép mô hình học hỏi và phát triển sự hiểu biết về nhiều đối tượng hiệu quả hơn.[]

**Nhược điểm:**

* Khó có thể phát hiện các đối tượng nhỏ hơn trong hình ảnh hoặc video do tỷ lệ recall chưa cao.[]
* Khó thể phát hiện hai đối tượng cực kỳ gần nhau do giới hạn của các hộp giới hạn.[]

## Bài toán phân loại ảnh (Image Classification)

### Khái niệm

Phân loại ảnh (thường được gọi là *Image Recognition* – Nhận dạng ảnh) là nhiệm vụ thuộc lĩnh vực *Computer vision* (Thị giác máy tính). Đây là một kỹ thuật liên quan đến việc phân tích nội dung của hình ảnh để gắn thẻ hoặc nhãn cho hình ảnh đó. Điều này phân phối các hình ảnh thành các lớp khác nhau. Mục đích của quá trình này là giải thích kỹ thuật số nội dung của hình ảnh cho máy. [[?](https://research.aimultiple.com/image-classification/)]

Phân loại hình ảnh (Image classification) hay Nhận dạng hình ảnh (Image recognition) là một trong những tác vụ của thị giác máy tính, ở đó thuật toán xem xét và dán nhãn cho hình ảnh từ một tập danh mục được xác định và đào tạo trước. Ví dụ, với một tập các hình ảnh, mỗi hình ảnh mô tả một con mèo hoặc một con chó, thuật toán sẽ “quan sát” toàn bộ dữ liệu và dựa trên hình dạng, màu sắc để hình thành giả thuyết liên quan đến nội dung của ảnh. Kết quả thu được là từ tập dữ liệu ban đầu, các hình ảnh chó/mèo đã được phân loại một cách tự động. Thực tế, thị giác góp phần tạo nên 80-85% nhận thức của con người về thế giới. Hàng ngày, mỗi người phải thực hiện phân loại trên bất kỳ dữ liệu hình ảnh nào mà chúng ta bắt gặp. Do đó, mô phỏng nhiệm vụ phân loại với sự trợ giúp của mạng nơ-ron là một trong những ứng dụng đầu tiên của thị giác máy tính mà các nhà nghiên cứu nghĩ đến.

Lĩnh vực thị giác máy tính bao gồm một tập hợp các vấn đề chính như phân loại hình ảnh, bản địa hóa, phân đoạn hình ảnh và phát hiện đối tượng. Trong số đó, phân loại ảnh có thể được coi là vấn đề cơ bản. Nó tạo cơ sở cho các vấn đề về thị giác máy tính khác. [[?](https://viso.ai/computer-vision/image-classification/)]

### Phân loại

Có nhiều thuật toán khác nhau được ứng dụng trong việc phân loại hình ảnh. Các thuật toán này được chia thành hai nhóm chính là **Học có giám sát (supervised learning)** và **Học không giám sát (unsupervised learning**).

**Phân loại có giám sát**: Trong học máy có giám sát, thuật toán được huấn luyện trên một tập hình ảnh đã được dán nhãn. Từ dữ liệu mẫu này, thuật toán có thể trích xuất thông tin, phục vụ phân loại ngay cả những hình ảnh chưa từng nhìn thấy trước đó.

Xuyên suốt quá trình đào tạo, đặc điểm của ma trận hình ảnh sẽ được trích xuất dưới dạng dữ liệu quan trọng để đưa vào xử lý. Các đặc điểm này đại diện cho hình ảnh trong không gian chiều thấp (lower-dimensional feature space) và là cơ sở để thuật toán tiến hành phân loại. Trong quá trình đánh giá, các đặc điểm của ảnh thử nghiệm được thu thập và tái phân loại với sự hỗ trợ của mạng thần kinh nhân tạo. Hệ thống lúc này đã có thể nhận biết các đặc điểm điển hình của mọi lớp hình ảnh mà nó được đào tạo.

Các phương pháp phân loại phổ biến dựa trên học có giám sát bao gồm:

* Support Vector Machines
* Decision Trees
* K Nearest Neighbors

Các mạng nơ-ron thường được sử dụng để phân loại hình ảnh có giám sát bao gồm AlexNet, ResNet, DenseNet và Inception.

Đối với phân loại có giám sát, việc dán nhãn dữ liệu đóng vai trò quan trọng. Độ chính xác của dữ liệu được dán nhãn quyết định phần lớn hiệu suất của mô hình học máy. Các thuật toán phân loại có giám sát có thể được chia thành hai mục nhỏ hơn dựa trên nhãn dữ liệu.

***Phân loại nhãn đơn (Single-label classification)*** là tác vụ phổ biến nhất trong phân loại ảnh có giám sát. Theo đó, mỗi hình ảnh được đại diện bởi một nhãn/chú thích (a single label or annotation). Mô hình xuất ra một giá trị hoặc dự đoán duy nhất cho mỗi hình ảnh mà nó xử lý. Đầu ra từ mô hình là mã hóa One-hot (từng giá trị được biến đổi thành các đặc trưng nhị phân chỉ chứa giá trị 1 hoặc 0). Mã hóa One-hot có độ dài bằng số lớp và giá trị biểu thị xác suất hình ảnh thuộc về lớp này. Hàm Softmax được sử dụng để đảm bảo các xác suất tổng bằng một và xác suất tối đa được chọn làm đầu ra của mô hình. Mặc dù Softmax không có giá trị về mặt dự đoán, nhưng nó giúp ràng buộc đầu ra giữa 1 và 0, nhờ vậy, có thể đánh giá độ tin cậy của mô hình từ điểm Softmax. Một số ví dụ về bộ dữ liệu phân loại nhãn đơn bao gồm MNIST, SVHN, ImageNet, v.v. Phân loại nhãn đơn có thể được xếp vào phân loại đa lớp (Multiclass classification) hoặc phân loại nhị phân (binary classification).

***Phân loại đa nhãn*** là một tác vụ phân loại trong đó mỗi hình ảnh có thể chứa nhiều hơn một nhãn hoặc một số hình ảnh chứa đồng thời tất cả các nhãn. Phân loại đa nhãn xuất hiện phổ biến trong lĩnh vực xử lý hình ảnh y tế, khi một bệnh nhân có thể được chẩn đoán mắc nhiều bệnh dựa trên dữ liệu chụp X-quang.

**Phân loại không giám sát**

Trong học máy không giám sát, thuật toán chỉ sử dụng dữ liệu thô để đào tạo.Các nhãn phân loại thường không xuất hiện trong kiểu học này và mô hình học bằng cách nhận dạng các mẫu trong tập dữ liệu huấn luyện. Giống như phân loại có giám sát, các phương pháp dựa trên không giám sát cũng liên quan đến bước trích xuất đặc điểm với các thông tin chi tiết nhất về hình ảnh. Các đặc điểm này sau đó được xử lý bằng các phương pháp phân cụm tham số (Gaussian Mixture Models) và phi tham số (K-means) hoặc các thuật toán học không giám sát khác. Các thuật toán và kỹ thuật phân loại của thị giác máy tính không chỉ giới hạn trong dữ liệu hình ảnh 2D đơn giản mà còn mở rộng ra dưới dạng Video và ảnh 3D.

### Các hướng giải quyết

## Các mô hình CNN được đào tạo trước (pre-trained CNN model)

### Học chuyển giao (Transfer learning)

#### a. Khái niệm

Học chuyển giao (Transfer learning) là một lĩnh vực con của học máy và trí tuệ nhân tạo nhằm mục đích áp dụng kiến thức thu được từ một tác vụ (tác vụ nguồn – source task) cho một tác vụ khác nhưng tương tự (tác vụ đích – target task).

Ví dụ, kiến thức thu được khi học cách phân loại văn bản Wikipedia có thể được sử dụng để giải quyết các bài toán phân loại văn bản pháp lý. Một ví dụ khác là sử dụng kiến thức từ bài toán phân loại ô tô để nhận biết các loài chim trên bầu trời.

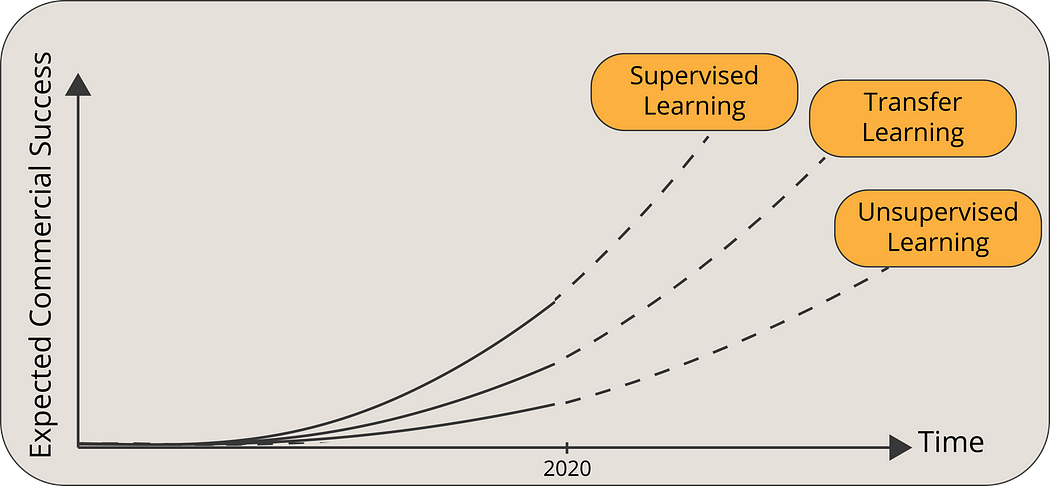
Transfer learning, do đó, có thể hiểu đơn giản là cải tiến việc học trong một tác vụ mới thông qua chuyển giao kiến thức từ một tác vụ liên quan đã được học.

Như vậy, với transfer learning, chúng ta có thể xây dựng các ứng dụng AI trong một khoảng thời gian rất ngắn.

#### b. Quá trình ra đời

Để chứng minh cho sức mạnh của transfer learning, hãy nhìn vào nhận định của Andrew Ng – người đồng sáng lập Google Brain: “Transfer learning will be the next driver of machine learning’s commercial success after supervised learning.” (tạm dịch: Học chuyển giao sẽ là động lực tiếp theo dẫn dắt sự thành công về thương mại của học máy sau học có giám sát).

Lịch sử của Transfer learning bắt đầu từ năm 1993. Với bài báo “Discriminability-Based Transfer between Neural Networks”, Lorien Pratt đã giới thiệu với thế giới về tiềm năng của học chuyển giao. Vào tháng 7 năm 1997, tạp chí Machine Learning đã xuất bản một số đặc biệt cho các bài báo về Transfer learning. Khi lĩnh vực này phát triển, các chủ đề có liên quan như học đa tác vụ (multi-task learning) cũng được đưa vào lĩnh vực học chuyển giao. Learning to Learn là một trong những cuốn sách tiên phong trong lĩnh vực này. Ngày nay, học chuyển giao là một nguồn lực mạnh mẽ để các công ty công nghệ xây dựng những giải pháp AI mới và các nhà nghiên cứu tiếp tục thúc đẩy giới hạn của học máy.



##### **Hình 2. 18** Kỳ vọng của Andrew Ng về thành công thương mại của các lĩnh vực học máy

#### c. Cách hoạt động

Có ba yêu cầu để đạt được học chuyển giao:

1. Mô hình được đào tạo sẵn với mã nguồn mở bởi bên thứ ba

Mô hình được đào tạo trước là mô hình do người khác xây dựng và huấn luyện để giải quyết một vấn đề tương tự. Thực tế, các mô hình được tham khảo thường là sản phẩm của những gã khổng lồ công nghệ hoặc nhóm các nhà khoa học nổi tiếng. Họ thường chọn tập dữ liệu rất lớn làm dữ liệu cơ sở, chẳng hạn như ImageNet hoặc Wikipedia Corpus. Sau đó, họ phát triển một mạng thần kinh lớn (ví dụ: VGG19 có 143.667.240 tham số) để giải quyết một vấn đề cụ thể (ví dụ: phân loại hình ảnh cho VGG19). Tất nhiên, mô hình được đào tạo trước này phải được công khai để ta có thể tái sử dụng chúng.

2. Tái sử dụng mô hình

Sau khi tiếp cận các mô hình được đào tạo trước này, ta sử dụng lại kiến thức mà mô hình đã học, bao gồm các lớp, tính năng, trọng số và hệ số tự do. Có một số cách giúp ta tải mô hình đã được đào tạo vào môi trường của mình, biến nó trở thành một tệp/thư mục chứa thông tin liên quan. Tuy nhiên, nên ưu tiên sử dụng các thư viện học sâu lưu trữ nhiều mô hình được đào tạo trước, chẳng hạn như: TensorFlow Hub, Keras Applications, PyTorch Hub

Chúng ta có thể sử dụng một trong các nguồn trên để tải mô hình đào tạo sẵn. Nó thường đi kèm với tất cả các lớp, trọng số và có thể chỉnh sửa mạng theo ý muốn.

3. Tinh chỉnh cho bài toán

Việc tinh chỉnh mô hình được đào tạo trước nhằm hai mục đích: nâng cao độ chính xác của mô hình và đảm bảo đầu ra đúng định dạng.

Nói chung, trong một mạng thần kinh, trong khi các lớp dưới cùng và giữa thường đại diện cho các tính năng chung, thì các lớp trên cùng thể hiện các tính năng cụ thể của bài toán. Vì bài toán mới khác với bài toán ban đầu, nên ta thường có xu hướng loại bỏ các lớp trên cùng và thêm vào các lớp cụ thể cho bài toán của mình để đạt được độ chính xác cao hơn.

Ví dụ: một mô hình được đào tạo với ImageNet có thể phân loại tới 1000 đối tượng. Nếu ta cố gắng phân loại các chữ số viết tay (ví dụ: phân loại MNIST), cách tốt nhất là nên kết thúc với lớp cuối cùng chỉ có 10 nơ-ron.

Sau khi thêm các lớp tùy chỉnh vào mô hình được đào tạo trước, ta có thể định cấu hình mô hình đó bằng các loss functions và trình tối ưu hóa đặc biệt, đồng thời tiến hành đào tạo bổ sung.

### Mô hình VGG-19

VGG là một mạng thần kinh tích chập (CNN) có độ sâu 19 lớp. Nó được xây dựng và đào tạo bởi Karen Simonyan và Andrew Zisserman tại Đại học Oxford vào năm 2014. VGG-19 được đào tạo bằng cách sử dụng hơn 1 triệu hình ảnh từ cơ sở dữ liệu ImageNet, trong đó có các ảnh màu 224×224 pixel. Đương nhiên, chúng ta có thể nhập vào mô hình các trọng số được huấn luyện bởi ImageNet. Điểm đặc biệt, VGG-19 có thể phân loại tới 1000 đối tượng. Dưới đây là thông tin ngắn gọn về kích thước và hiệu suất của nó:

* Kích thước: 549 MB
* Top-1: Độ chính xác: 71,3%
* Top-5: Độ chính xác: 90,0%
* Số tham số: 143.667.240
* Độ sâu: 19



##### **Hình 2. 19** Mô hình mạng VGG-19

### Mô hình InceptionV3 (GoogLeNet)

Inceptionv3 là một mạng thần kinh tích chập có độ sâu 50 lớp. Nó được xây dựng và đào tạo bởi Google. Phiên bản được đào tạo trước của Inceptionv3 với trọng số ImageNet có thể phân loại tới 1000 đối tượng. Kích thước hình ảnh đầu vào của mạng này là 299×299 pixel, lớn hơn mạng VGG19. Trong khi VGG19 về nhì trong cuộc thi ImageNet năm 2014 thì Inception là mô hình thắng cuộc. Tóm tắt ngắn gọn về các tính năng của Inceptionv3 như sau:

* Kích thước: 92 MB
* Top-1: Độ chính xác: 77,9%
* Top-5: Độ chính xác: 93,7%
* Số tham số: 23.851.784
* Độ sâu: 189



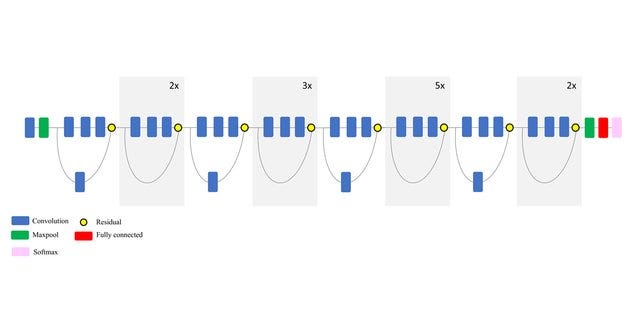
##### **Hình 2. 20** Mô hình mạng Inceptionv3 (GoogLeNet)

### Mô hình ResNet50

ResNet50 là một mạng thần kinh tích chập có độ sâu 50 lớp. Nó được xây dựng và đào tạo bởi Microsoft vào năm 2015. Sau khi chiến thắng cuột thi ImageNet 2015, nó khẳng định xu hướng chung: các mô hình thị giác máy tính ngày càng sâu hơn, với ngày càng ít tham số hơn. Chìa khóa để có thể đào tạo một mạng sâu như vậy là sử dụng *các kết nối bỏ qua* (còn gọi là *kết nối tắt*): tín hiệu cấp vào một lớp cũng được thêm vào đầu ra của một lớp nằm ở vị trí cao hơn trong ngăn xếp. Mô hình này cũng được đào tạo trên hơn 1 triệu hình ảnh từ cơ sở dữ liệu ImageNet. Cũng giống như VGG-19, nó có thể phân loại tới 1000 đối tượng và mạng được đào tạo trên các hình ảnh có màu 224×224 pixel. Dưới đây là thông tin ngắn gọn về kích thước và hiệu suất của nó:

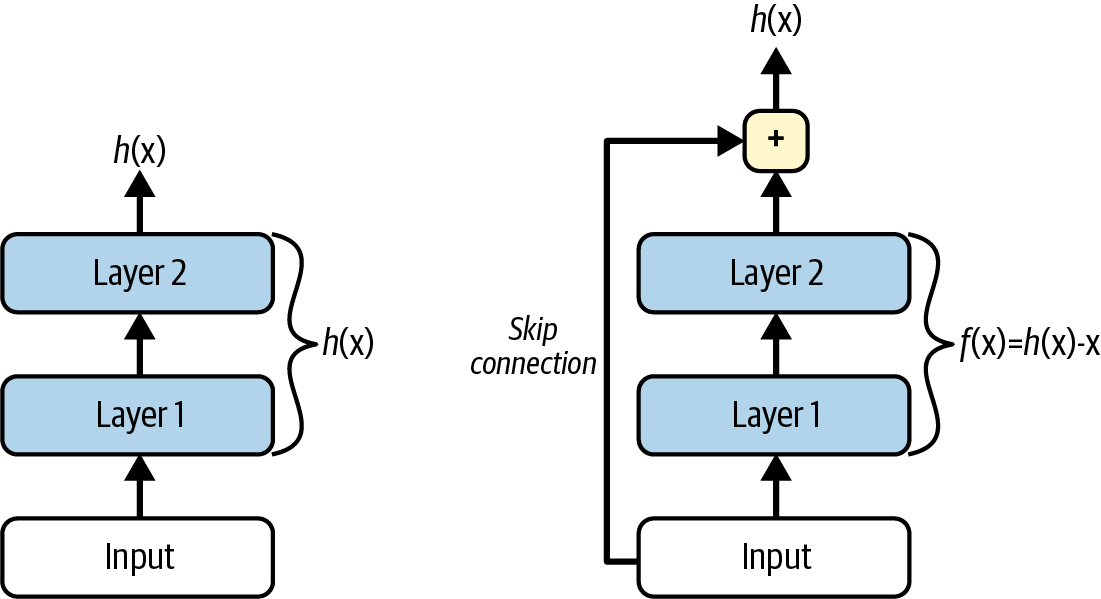
* Kích thước: 98 MB
* Top-1: Độ chính xác: 74,9%
* Top-5: Độ chính xác: 92,1%
* Số tham số: 25.636.712
* Độ sâu: 107

Nếu so sánh ResNet50 với VGG19, ResNet50 thực sự vượt trội so với VGG19 mặc dù nó có độ phức tạp thấp hơn. ResNet50 đã được cải thiện nhiều lần và có nhiều phiên bản mới hơn như ResNet101, ResNet152, ResNet50V2, ResNet101V2, ResNet152V2.



##### **Hình 2. 21** Mô hình mạng ResNet50

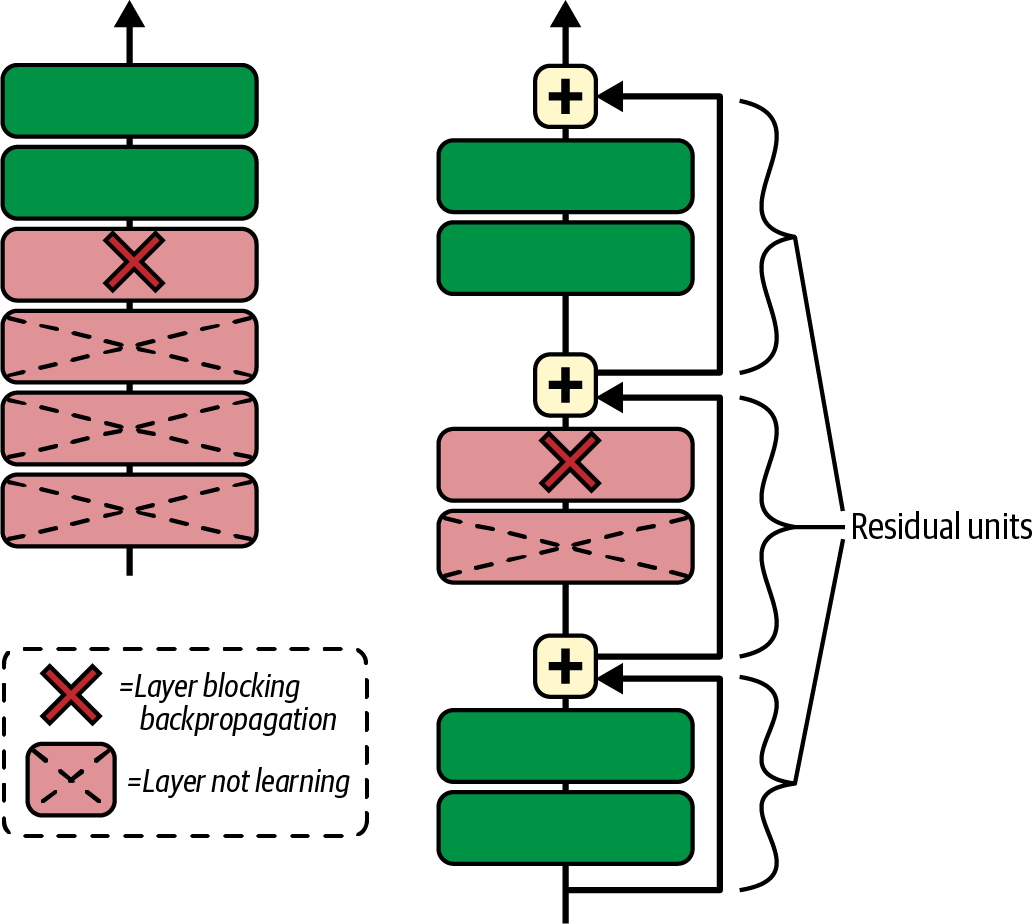
Khi huấn luyện một mạng nơ-ron, mục tiêu là làm cho nó mô hình hóa một hàm mục tiêu *h*(**x**). Nếu thêm đầu vào **x** vào đầu ra của mạng (nghĩa là thêm một kết nối bỏ qua), thì mạng sẽ buộc phải lập mô hình *f*(**x**) = *h*(**x**) – **x** thay vì *h*(**x**). Điều này được gọi là *học dư*.



##### **Hình 2. 22** Học dư

Khi ta khởi tạo một mạng thần kinh thông thường, các trọng số của nó gần bằng không, vì vậy mạng chỉ xuất ra các giá trị gần bằng không. Nếu thêm một kết nối bỏ qua, thì mạng kết quả chỉ xuất ra một bản sao các đầu vào của nó; nói cách khác, ban đầu nó mô hình hóa chức năng nhận dạng. Nếu hàm mục tiêu khá gần với hàm nhận dạng (thường là như vậy), điều này sẽ tăng tốc độ đào tạo đáng kể.

Hơn nữa, nếu ta thêm nhiều kết nối bỏ qua, mạng có thể bắt đầu tiến triển ngay cả khi một số lớp chưa bắt đầu học.

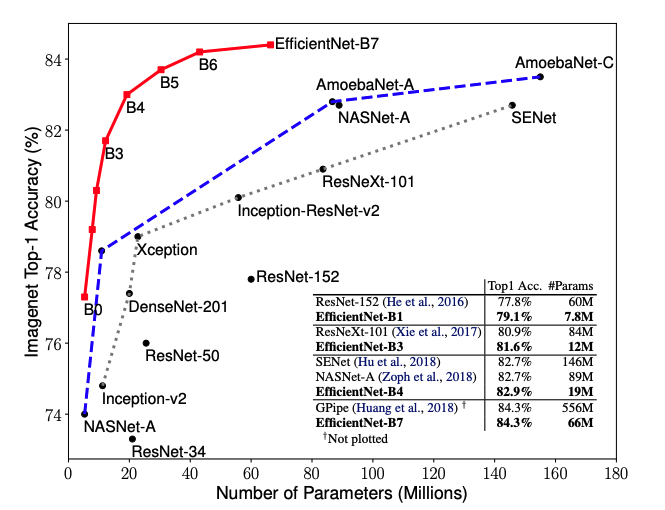


##### **Hình 2. 23** Mạng nơ-ron sâu thông thường (trái) và mạng thặng dư sâu (phải)

Nhờ bỏ qua các kết nối, tín hiệu có thể dễ dàng đi qua toàn bộ mạng. Mạng thặng dư sâu có thể được xem như một chồng các *đơn vị thặng dư* (RUs), trong đó mỗi đơn vị thặng dư là một mạng thần kinh nhỏ có kết nối bỏ qua.

### Mô hình EfficientNet

EfficientNet là mạng thần kinh tích chập tiên tiến nhất đã được Google đào tạo và phát hành ra công chúng với bài báo “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks” vào năm 2019. Có 8 phiên bản thay thế của EfficientNet (B0 đến B7) và ngay cả phiên bản đơn giản nhất - EfficientNetB0, cũng xuất sắc. Với 5,3 triệu tham số, nó đạt hiệu suất chính xác Top-1 77,1%.



##### **Hình 2. 24** Kích thước mô hình EfficientNet so với Độ chính xác ImageNet

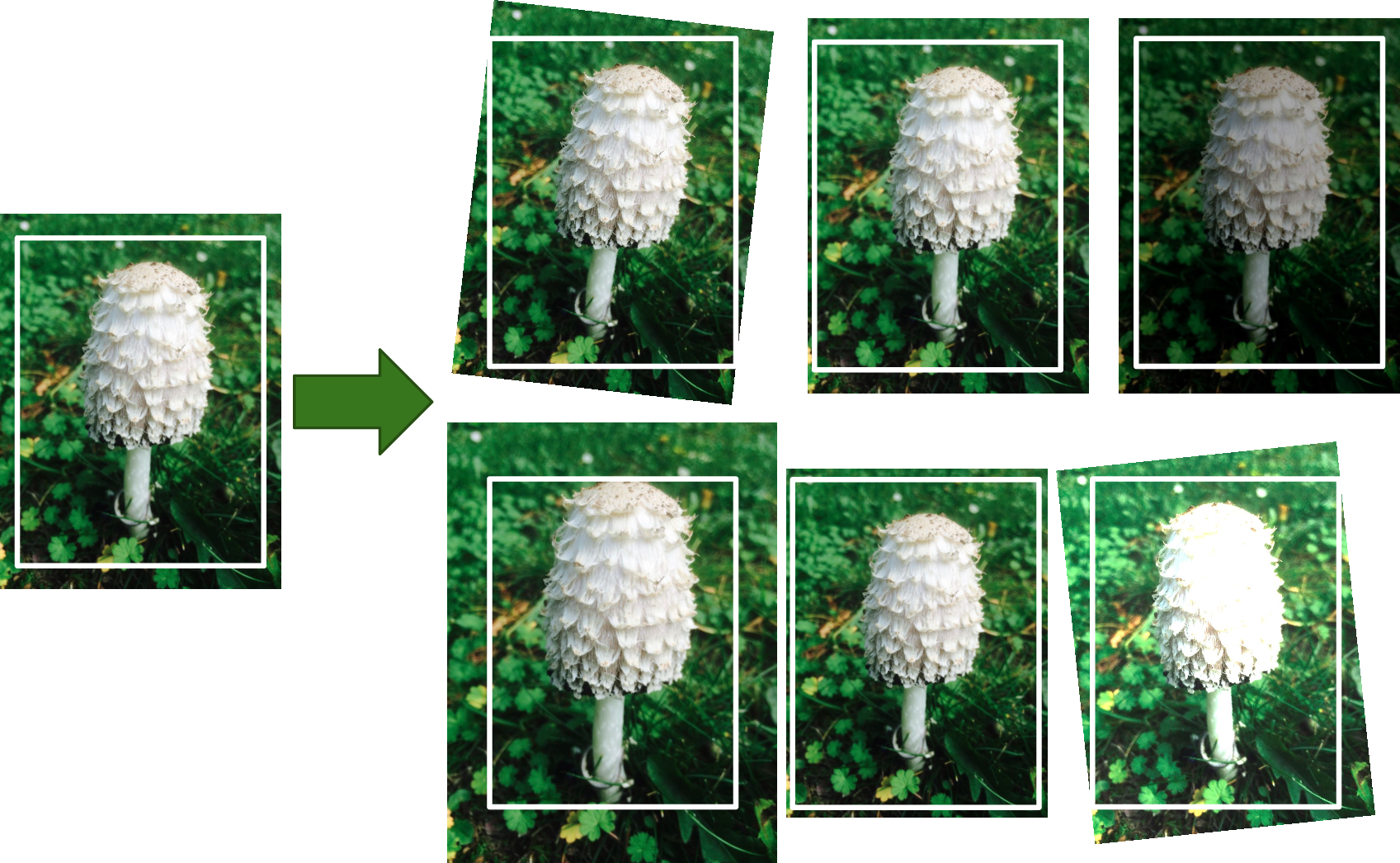
Tóm tắt ngắn gọn về các tính năng của EfficientNetB0 như sau:

* Kích thước: 29 MB
* Top-1: Độ chính xác: 77,1%
* Top-5: Độ chính xác: 93,3%
* Số tham số: ~5.300.000
* Độ sâu: 132

### Tăng cường dữ liệu

Tăng cường dữ liệu làm tăng kích thước của tập huấn luyện một cách giả tạo bằng cách tạo ra nhiều biến thể thực tế của từng trường hợp huấn luyện. Điều này làm giảm quá khớp, làm cho kỹ thuật này trở thành một kỹ thuật chính quy. Các thể hiện được tạo phải càng thực tế càng tốt: lý tưởng nhất là đưa ra một hình ảnh từ tập huấn luyện tăng cường, con người sẽ không thể biết liệu nó có được tăng cường hay không.

Ví dụ: ta có thể dịch, xoay và thay đổi kích thước từng ảnh trong tập huấn luyện một chút với các mức khác nhau và thêm ảnh kết quả vào tập huấn luyện. Điều này buộc mô hình phải thích ứng tốt hơn với những thay đổi về vị trí, hướng và kích thước của các đối tượng trong ảnh. Để tạo ra một mô hình có khả năng thích ứng tốt hơn với các điều kiện ánh sáng khác nhau, ta có thể tạo ra nhiều hình ảnh có độ tương phản khác nhau theo cách tương tự. Nói chung, ta cũng có thể lật ảnh theo chiều ngang (ngoại trừ văn bản và các đối tượng không đối xứng khác). Bằng cách kết hợp các phép biến đổi này, chúng ta có thể tăng đáng kể quy mô tập huấn luyện của mình.



##### **Hình 2. 25** Tạo các thể hiện đào tạo mới từ các thể hiện hiện có

Tăng cường dữ liệu cũng hữu ích khi ta có một tập dữ liệu không cân bằng: có thể sử dụng nó để tạo ra nhiều mẫu hơn cho các lớp ít phổ biến hơn. Đây được gọi là *kỹ thuật lấy mẫu thiểu số tổng hợp*, hay viết tắt là SMOTE.

## Mô hình hồi quy Logistic (Logistic Regression)

### Khái niệm

Hồi quy Logistic là một thuật toán học máy hồi quy (học có giám sát) có thể được sử dụng để phân loại (và ngược lại). Hồi quy logistic (còn gọi là hồi quy logit) thường được sử dụng để ước tính xác suất mà một cá thể thuộc về một lớp cụ thể (ví dụ: xác suất email này là thư rác là bao nhiêu?). Nếu xác suất ước tính lớn hơn một ngưỡng nhất định (thường là 50%), thì mô hình dự đoán rằng thực thể thuộc về lớp đó (được gọi là lớp khẳng định (dương), được gắn nhãn “1”) và nếu không thì mô hình dự đoán rằng nó không thuộc về lớp đó (tức là, nó thuộc về lớp phủ định (âm), được dán nhãn là “0”). Điều này làm cho nó trở thành một bộ phân loại nhị phân.

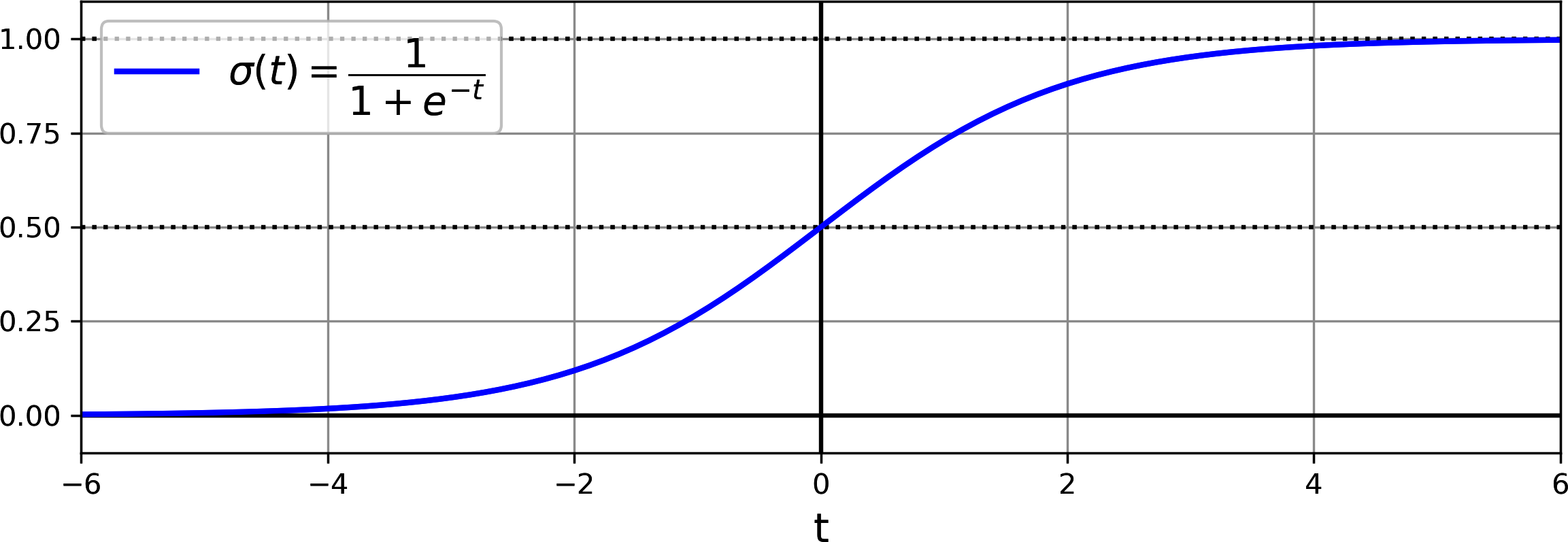
### Nguyên lý hoạt động

Mô hình hồi quy logistic tính toán tổng trọng số của các tính năng đầu vào (cộng với thuật ngữ sai lệch), nó xuất ra logistic của kết quả này (xác suất của kết quả đó).

**Phương trình 2. 1** Mô hình hồi quy logistic xác suất ước tính (dạng véc tơ)

Logistic - được ghi chú là σ(·) - là một *hàm sigmoid* (nghĩa là hình chữ S) xuất ra một số từ 0 đến 1.

**Phương trình 2. 2** Hàm Logistic



##### **Hình 2. 26** Hàm logistic

Khi mô hình hồi quy logistic đã ước tính xác suất p= h θ (x) mà một thể hiện x thuộc về lớp tích cực, nó có thể dễ dàng đưa ra dự đoán ŷ.

**Phương trình 2. 3** Dự đoán mô hình hồi quy logistic sử dụng xác suất ngưỡng 50%

Ta có: σ(t) < 0,5 khi t < 0 và σ(t) ≥ 0,5 khi t ≥ 0, do đó, mô hình hồi quy logistic sử dụng ngưỡng mặc định có xác suất 50% dự đoán 1 nếu **θ⊺x** dương và 0 nếu nó là âm.

Điểm t thường được gọi là *logit*. Cái tên này xuất phát từ thực tế là hàm logit, được định nghĩa là logit(p) = log(p / (1 – p)), là hàm nghịch đảo của hàm logistic, thấy rằng kết quả là t. Logit còn được gọi là *log-odds* vì nó là log của tỷ lệ giữa xác suất ước tính cho lớp dương và xác suất ước tính cho lớp âm.

### Đào tạo và hàm chi phí

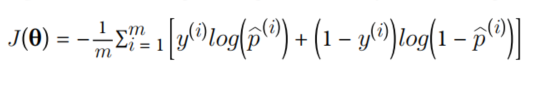
Mục tiêu của đào tạo là thiết lập tham số vector **θ** sao cho mô hình ước tính xác suất cao cho các trường hợp dương (y = 1) và xác suất thấp cho các trường hợp âm (y = 0). Ý tưởng này được nắm bắt bởi hàm chi phí được cho một trường hợp huấn luyện duy nhất **x**.

**Phương trình 2. 4** Hàm chi phí của một trường hợp đào tạo duy nhất

Hàm chi phí này có ý nghĩa vì –log(t) tăng rất lớn khi t tiến đến 0, do đó chi phí sẽ lớn nếu mô hình ước tính xác suất gần bằng 0 cho một trường hợp dương và nó cũng sẽ lớn nếu mô hình ước tính xác suất gần bằng 1 đối với trường hợp âm. Mặt khác, –log(t) gần bằng 0 khi t gần bằng 1, do đó, chi phí sẽ gần bằng 0 nếu xác suất ước tính gần bằng 0 đối với trường hợp âm hoặc gần bằng 1 đối với trường hợp dương, đó chính xác là những gì chúng ta cần.

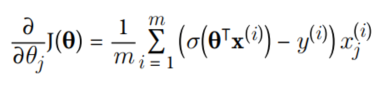
Hàm chi phí trên toàn bộ tập huấn luyện là chi phí trung bình trên tất cả các trường hợp đào tạo. Nó có thể được viết trong một biểu thức duy nhất được gọi là *log loss*.

Phương trình . Hàm chi phí hồi quy logistic (log loss)



Không có phương trình dạng đóng đã biết để tính giá trị của **θ** giúp giảm thiểu hàm chi phí này (không có phương trình tương đương với phương trình Bình thường). Nhưng hàm chi phí này là hàm lồi, do đó, việc giảm độ dốc (hoặc bất kỳ thuật toán tối ưu hóa nào khác) được đảm bảo để tìm ra mức tối thiểu toàn cầu (nếu tốc độ học không quá lớn và ta đợi đủ lâu). Các đạo hàm riêng của hàm chi phí liên quan đến tham số **θ***j* mô hình thứ *j* được cho bởi phương trình dưới.

Phương trình . Đạo hàm riêng hàm chi phí logistic



Đối với mỗi trường hợp, nó tính toán lỗi dự đoán và nhân nó với giá trị tính năng thứ j, sau đó tính giá trị trung bình trên tất cả các trường hợp đào tạo. Sau khi có vectơ độ dốc chứa tất cả các đạo hàm riêng, ta có thể sử dụng nó trong thuật toán giảm dần độ dốc hàng loạt.

### Hồi quy Softmax

Mô hình hồi quy logistic có thể được khái quát hóa để hỗ trợ trực tiếp nhiều lớp mà không cần phải đào tạo và kết hợp nhiều bộ phân loại nhị phân. Điều này được gọi là *hồi quy softmax*, hoặc *hồi quy logistic đa thức*.

Ý tưởng rất đơn giản: khi được cung cấp một thể hiện **x**, mô hình hồi quy softmax trước tiên sẽ tính điểm *sk*(**x**) cho mỗi lớp *k*, sau đó ước tính xác suất của mỗi lớp bằng cách áp dụng *hàm softmax* (còn gọi là *hàm mũ chuẩn hóa*) cho điểm số . Phương trình để tính toán *sk*(**x**) giống như phương trình dự đoán hồi quy tuyến tính.

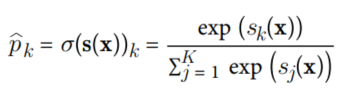
Phương trình . Điểm Softmax cho lớp k



Mỗi lớp có vectơ tham số riêng **θ**(*k*). Tất cả các vectơ này thường được lưu dưới dạng các hàng trong *ma trận tham số* **Θ**.

Khi ta đã tính điểm của mọi lớp cho cá thể **x**, ta có thể ước tính xác suất  rằng cá thể đó thuộc về lớp *k* bằng cách chạy các điểm thông qua hàm softmax. Hàm này tính toán cấp số nhân của mọi điểm số, sau đó chuẩn hóa chúng (chia cho tổng của tất cả các cấp số nhân). Những điểm này thường được gọi là logits hoặc log-odds (mặc dù chúng thực sự là log-odds không chuẩn hóa).

Phương trình . Hàm Softmax



Trong phương trình này:

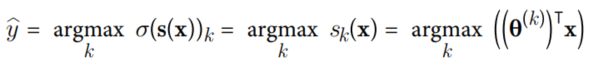
\* K là số lớp.

\* **s(x)** là một véc tơ chứa điểm của mỗi lớp đối với thể hiện **x**.

\* σ(**s(x)**)*k* là xác suất ước tính rằng thực thể x thuộc về lớp k, dựa trên điểm số của từng lớp cho thực thể đó.

Theo mặc định, trình phân loại hồi quy softmax dự đoán lớp có xác suất ước tính cao nhất (đơn giản là lớp có điểm cao nhất).

Phương trình . Dự đoán phân loại hồi quy Softmax



Toán tử *argmax* trả về giá trị của một biến làm cực đại một hàm. Trong phương trình này, nó trả về giá trị của *k* tối đa hóa xác suất ước tính σ(**s(x)**)*k*.

Trình phân loại hồi quy softmax chỉ dự đoán một lớp tại một thời điểm (nghĩa là nó là nhiều lớp, không phải nhiều đầu ra).

**Quá trình đào tạo**

Mục tiêu là có một mô hình ước tính xác suất cao cho lớp mục tiêu (và do đó xác suất thấp cho các lớp khác). Việc tối thiểu hóa hàm chi phí được gọi là *cross entropy*, sẽ dẫn đến mục tiêu này vì nó cản trở mô hình khi ước tính xác suất thấp cho một lớp mục tiêu. Cross entropy thường được sử dụng để đo mức độ phù hợp của một tập hợp các xác suất lớp ước lượng với các lớp mục tiêu.

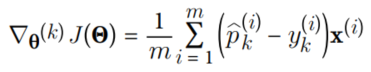
Phương trình . Hàm chi phí Cross entropy



Trong phương trình này, *yk(i)* là xác suất mục tiêu mà thực thể thứ *i* thuộc lớp *k*. Nó bằng 1 hoặc 0, tùy thuộc vào việc thể hiện có thuộc lớp hay không. Khi chỉ có hai lớp (K = 2), hàm chi phí này tương đương với hàm chi phí hồi quy logistic.

Vectơ độ dốc của hàm chi phí này đối với **θ**(*k*) được cho bởi phương trình dưới.

Phương trình . Vectơ độ dốc cross entropy cho lớp k



Ta có thể tính toán vectơ độ dốc cho mọi lớp, sau đó sử dụng độ dốc gốc (hoặc bất kỳ thuật toán tối ưu hóa nào khác) để tìm ma trận tham số **Θ** giúp cực tiểu hóa hàm chi phí.

**Cross Entropy:**

Cross entropy bắt nguồn từ lý thuyết thông tin của Claude Shannon. Giả sử ta muốn truyền thông tin về thời tiết mỗi ngày một cách hiệu quả. Nếu có tám tùy chọn (trời nắng, mưa, v.v.), ta có thể mã hóa từng tùy chọn bằng 3 bit, vì 23 = 8. Tuy nhiên, nếu ta nghĩ rằng hầu như ngày nào trời cũng nắng, thì việc mã hóa sẽ hiệu quả hơn nhiều “ sunny” chỉ trên một bit (0) và bảy tùy chọn khác trên bốn bit (bắt đầu bằng 1). Cross entropy đo số bit trung bình chúng ta thực sự gửi cho mỗi tùy chọn. Nếu giả định của ta về thời tiết là hoàn hảo, thì cross entropy sẽ bằng với entropy của chính thời tiết (nghĩa là tính khó dự đoán nội tại của nó). Nhưng nếu giả định của chúng ta là sai (ví dụ: nếu trời mưa thường xuyên), cross entropy sẽ lớn hơn một lượng gọi là *phân kỳ Kullback–Leibler (KL)*.

Cross entropy giữa hai phân bố xác suất *p* và *q* được định nghĩa là *H(p,q)* = –Σx *p*(x) log *q*(x) (ít nhất là khi các phân phối rời rạc).

## Tensorflow & Keras

## Kết luận

# XÂY DỰNG ỨNG DỤNG

## Tổ chức dữ liệu

### Tổ chức dữ liệu cho mô hình YOLO

Chúng ta sử dụng YOLO để nhận dạng đối tượng hoa, vì vậy chỉ có một lớp duy nhất (“lower”).

Mô hình được xây dựng để nhận dạng … loài hoa, mỗi loài hoa được chuẩn bị 100 ảnh khác nhau liên quan đến các góc cạnh khác nhau của từng loại hoa đó.

Mỗi thư mục của mỗi loài hoa được chia thành 2 thư mục con: train và test tương ứng tỷ lệ (80 : 20).

Vì mô hình YOLO cần thêm file nhãn tương ứng cho từng ảnh, nên trong thư mục chứa ảnh sẽ bao gồm cả những tệp tin chứa nhãn tương ứng của từng đối tượng mà ta vẽ trong ảnh.

Nhãn của từng loài hoa sẽ lấy theo tên của thư mục chứa những tấm ảnh liên quan tới những loại hoa đó.

### Tổ chức dữ liệu cho mô hình CNN

Mô hình CNN là mấu chốt của ứng dụng, chúng ta dùng nó để trích xuất ra các đặc trưng của từng loài hoa, vì vậy nó bao gồm … lớp tương ứng với … loài hoa.

Mỗi loài hoa được chuẩn bị 500 bức ảnh khác nhau liên quan đến các góc cạnh khác nhau của từng loài hoa đó.

Dữ liệu sẽ được chia theo tỷ lệ (80 : 20), sử dụng 80% tổng số ảnh để học (tương đương 400 ảnh) và 20% còn lại để kiểm tra (tương đương 100 ảnh).

## Giao diện ứng dụng

## Những kết quả đạt được

### Kết quả đạt được của mô hình YOLO

### Kết quả đạt được của mô hình CNN

## Kết luận

# KẾT LUẬN VÀ KIẾN NGHỊ

DANH MỤC TÀI LIỆU THAM KHẢO

**Tiếng việt**

1. N.
2. N.
3. N

**Tiếng Anh**

1. R.
2. R.