

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  
**KHOA CÔNG NGHỆ THÔNG TIN 1**



**BÁO CÁO THỰC TẬP CƠ SỞ**  
**NHÓM 30**

Giảng viên hướng dẫn: GV. Trần Đình Quế

Họ và tên: Nguyễn Thị Tú Anh  
Mã sinh viên: B22DCCN034  
Lớp: D22CQCN10-B

**Hà Nội - 2025**

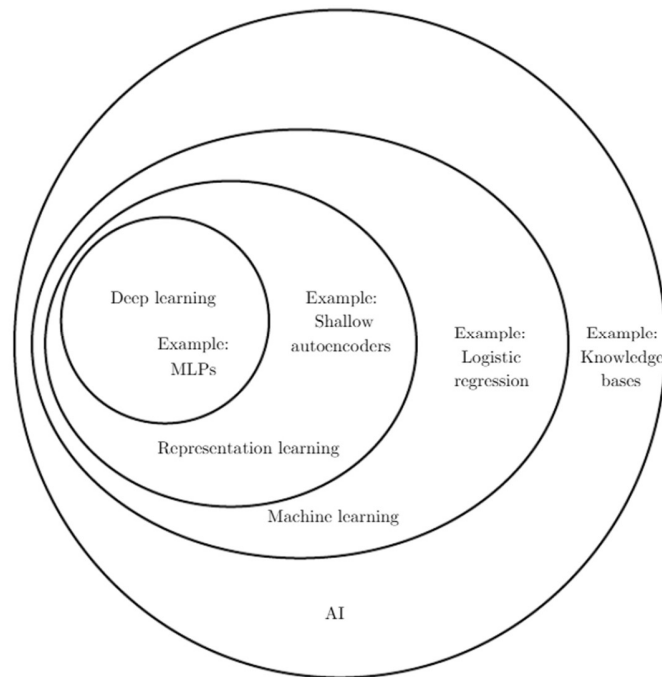
## Mục lục

CHƯƠNG 1: TRÍ TUỆ NHÂN TẠO VÀ CÁC ỨNG DỤNG .....	4
1.1. Khái niệm về trí tuệ nhân tạo .....	4
1.1.1 Trí Tuệ Nhân Tạo (AI).....	4
1.1.2. Học máy (Machine Learning) .....	5
1.1.3. Học sâu (Deep Learning) .....	5
1.2. Lịch sử phát triển của trí tuệ nhân tạo .....	5
1.3. Các ứng dụng của trí tuệ nhân tạo .....	6
1.3.1. Y Tế .....	7
1.3.2. Giao Thông Vận Tải.....	7
1.3.3. Thương Mại Điện Tử.....	8
1.3.4. Giáo Dục .....	9
1.3.5. An Ninh .....	9
1.3.6. Công Nghiệp và Sản Xuất.....	10
1.4. Lợi ích và thách thức của trí tuệ nhân tạo .....	10
1.4.1. Lợi ích .....	10
1.4.2. Thách thức .....	11
1.5. Tương lai của trí tuệ nhân tạo.....	11
1.6. Kết luận .....	12
CHƯƠNG 2: CÁC KỸ THUẬT HỌC SÂU.....	12
2.1. Giới thiệu về học sâu.....	12
2.2. Nền tảng lý thuyết .....	13
2.2.1. Từ học máy đến học sâu.....	13
2.2.2. Mạng nơ-ron nhân tạo .....	14
2.2.3. Quá trình học trong mạng nơ-ron.....	14
2.3. Các kiến trúc học sâu cơ bản.....	15
2.3.1. Mạng nơ-ron tích chập (CNN) .....	15
2.3.2. Mạng nơ-ron hồi quy (RNN).....	16
2.3.3. Mạng trí nhớ dài-ngắn hạn (LSTM).....	17
2.3.4. Mạng nơ-ron cổng hồi quy (GRU).....	18
2.3.5. Bộ mã hóa-giải mã (Autoencoder) .....	19
2.4. Kiến trúc học sâu tiên tiến.....	21
2.4.1. Mạng Transformer.....	21
2.4.2. Mô hình sinh (Generative Models) .....	22
2.4.3. Mạng đối kháng sinh (GAN).....	24
2.4.4. Mạng học khuếch tán (Diffusion Models) .....	26

2.5. Kỹ thuật huấn luyện và tối ưu hóa .....	28
2.5.1. Hàm kích hoạt .....	28
2.5.2. Các thuật toán tối ưu hóa.....	30
2.5.3. Kỹ thuật chính quy hóa (Regularization Techniques) .....	37
2.5.4. Chuẩn hóa dữ liệu (Data Normalization) .....	38
2.6 Ứng dụng của học sâu .....	39
2.6.1. Xử lý ảnh và thị giác máy tính (Image Processing and Computer Vision) .....	39
2.6.2. Xử lý ngôn ngữ tự nhiên (Natural Language Processing).....	39
2.6.3. Hệ thống đề xuất (Recommendation Systems) .....	40
2.6.4. Y học và chăm sóc sức khỏe (Medicine and Healthcare).....	40
2.6.5. Tự động hóa và robotics (Automation and Robotics) .....	41
2.7. Thách thức và hạn chế.....	42
2.7.1. Vấn đề học quá mức và thiếu mức (Overfitting and Underfitting) .....	42
2.7.2. Nhu cầu dữ liệu lớn (Large Data Requirements) .....	43
2.7.3. Chi phí tính toán cao (High Computational Costs) .....	43
2.7.4. Tính giải thích được (Explainability) .....	44
2.7.5. Vấn đề đạo đức và công bằng (Ethics and Fairness).....	45
CHƯƠNG 3: ỨNG DỤNG HỌC SÂU CHO PHÂN LOẠI HÌNH ẢNH SỬ DỤNG CNN VÀ SO SÁNH HIỆU SUẤT TRÊN 3 BỘ DỮ LIỆU KHÁC NHAU .....	46
3.1. Mô tả dự án.....	46
3.2. Ba bộ dữ liệu .....	46
3.3. Huấn luyện và đánh giá.....	47
3.3.1. Huấn luyện và đánh giá một mạng nơ-ron tích chập (CNN) để phân loại ảnh từ tập dữ liệu CIFAR-10 .....	47
3.3.2. Huấn luyện các mô hình học sâu để phân loại chữ số viết tay từ tập dữ liệu FOOD-101. ....	50
3.3.3. Huấn luyện các mô hình học sâu để phân loại chữ số viết tay từ tập dữ liệu FASHION MNIST.....	54
3.4. Kết quả và So sánh .....	56
3.5. Ứng dụng Web.....	57
3.6. Cài đặt và chạy ứng dụng.....	63
3.7. Hướng phát triển tương lai .....	64
TỔNG KẾT .....	64
TÀI LIỆU THAM KHẢO.....	66

# CHƯƠNG 1: TRÍ TUỆ NHÂN TẠO VÀ CÁC ỨNG DỤNG

## 1.1. Khái niệm về trí tuệ nhân tạo



Hình 1.1: Biểu đồ Venn cho thấy học sâu là một loại học biểu diễn, mà ngược lại là một loại học máy, được sử dụng cho nhiều nhưng không phải tất cả các phương pháp tiếp cận AI. Mỗi phần của biểu đồ Venn bao gồm một ví dụ về công nghệ AI

### 1.1.1 Trí Tuệ Nhân Tạo (AI)

Trí tuệ nhân tạo (Artificial Intelligence - AI) là một lĩnh vực quan trọng trong khoa học máy tính, tập trung vào việc phát triển các hệ thống và máy móc có khả năng thực hiện những nhiệm vụ đòi hỏi trí thông minh của con người. Những nhiệm vụ này bao gồm nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, ra quyết định, học hỏi từ dữ liệu và tương tác với môi trường xung quanh. Mục tiêu cốt lõi của AI là tạo ra các hệ thống có thể "suy nghĩ", "học hỏi" và "giải quyết vấn đề" một cách thông minh, bắt chước hoặc thậm chí vượt qua khả năng của con người trong một số lĩnh vực nhất định. Từ khi ra đời, AI đã trải qua nhiều giai đoạn phát triển, từ những ý tưởng cơ bản ban đầu đến những ứng dụng phức tạp trong cuộc sống hiện đại, và hứa hẹn sẽ tiếp tục định hình tương lai của nhân loại.

Ban đầu, AI được xây dựng dựa trên phương pháp AI biểu tượng (symbolic AI), nơi các nhà khoa học lập trình sẵn các quy tắc và logic để máy móc tuân theo. Ví dụ, trong một trò chơi cờ vua, máy tính được lập trình để phản ứng với các tình huống cụ thể, chẳng hạn như "nếu quân hậu bị đe dọa, thì di chuyển sang trái". Cách tiếp cận này tỏ ra hiệu quả với các vấn đề có cấu trúc rõ ràng như cờ vua hoặc các trò chơi logic khác. Tuy nhiên, nó lại gặp khó khăn lớn khi đối mặt với những

nhiệm vụ phức tạp hơn như nhận diện hình ảnh, dịch ngôn ngữ tự nhiên hay hiểu được ngữ cảnh trong giao tiếp hàng ngày. Lý do là vì những nhiệm vụ này không thể được mô tả đầy đủ bằng các quy tắc cố định, mà đòi hỏi sự linh hoạt và khả năng thích nghi với dữ liệu đa dạng.

### 1.1.2. Học máy (Machine Learning)

Học máy là một bước tiến quan trọng trong trí tuệ nhân tạo (AI), giúp vượt qua hạn chế của AI truyền thống dựa trên quy tắc cố định. Thay vì phải lập trình từng bước cụ thể (ví dụ: "Nếu thấy màu xanh thì là biển"), học máy cho phép máy tính tự học từ dữ liệu và cải thiện khả năng của mình mà không cần con người can thiệp chi tiết.

Cách hoạt động rất đơn giản: bạn cung cấp cho máy một lượng lớn dữ liệu đã được gán nhãn. Chẳng hạn, 10.000 bức ảnh về biển, núi, thành phố - rồi máy tự tìm ra quy luật để phân biệt chúng. Ví dụ, nó có thể nhận ra "màu xanh lấp lánh thường là biển" hay "màu xám với hình khối là thành phố".

Học máy bắt đầu phổ biến từ những năm 1990, nhờ máy tính mạnh hơn và dữ liệu ngày càng nhiều. Hiện nay, nó xuất hiện khắp nơi: từ nhận diện khuôn mặt trên điện thoại đến gợi ý phim trên Netflix. Nói một cách dễ hiểu, học máy là quá trình máy tính tự "mò mẫm" tìm ra cách giải quyết vấn đề dựa trên dữ liệu bạn đưa cho nó.

### 1.1.3. Học sâu (Deep Learning)

Trong học máy, một nhánh tiên tiến hơn gọi là học sâu (deep learning) đã xuất hiện, sử dụng các mạng nơ-ron nhân tạo (neural networks) mô phỏng cách hoạt động của não bộ con người. Học sâu đã đưa AI lên một tầm cao mới với những thành tựu ấn tượng trong nhiều lĩnh vực.

Deep Learning tập trung vào việc học các tầng (layers) biểu diễn dữ liệu liên tiếp. Bạn không cần những điều huyền bí đó để hiểu về deep learning, và tốt nhất nên quên đi bất kỳ mối liên hệ giả định nào giữa deep learning và sinh học. Về cơ bản, deep learning là một khuôn khổ toán học để học các biểu diễn từ dữ liệu.

Trong học sâu, thay vì phải thiết kế các đặc trưng (features) theo cách thủ công như trong các phương pháp máy học truyền thống, mô hình tự động trích xuất đặc trưng từ dữ liệu.

### **Tại sao học sâu mạnh mẽ?**

Học sâu có khả năng xử lý khối lượng dữ liệu khổng lồ và tìm ra các mẫu phức tạp mà các phương pháp truyền thống không thể làm được. Điều này nhờ vào cấu trúc nhiều lớp (layers) của mạng nơ-ron, cho phép hệ thống "hiểu" dữ liệu ở nhiều mức độ trừu tượng khác nhau. Học sâu sử dụng mạng nơ-ron nhân tạo (artificial neural networks), có cấu trúc lấy cảm hứng từ cách hoạt động của não bộ, nhưng không thực sự mô phỏng não bộ con người.

## 1.2. Lịch sử phát triển của trí tuệ nhân tạo

Lĩnh vực trí tuệ nhân tạo (AI) bắt đầu hình thành từ những năm 1950, khi Alan Turing đề xuất ý tưởng về máy móc có khả năng tư duy thông qua bài kiểm tra Turing nổi tiếng. Năm 1956, hội

ngộ Dartmouth đánh dấu sự ra đời chính thức của AI như một lĩnh vực nghiên cứu. Trong những thập kỷ tiếp theo, AI trải qua nhiều giai đoạn phát triển quan trọng với những thăng trầm:

#### **- Giai đoạn 1950-1970: Khởi đầu và sự lạc quan ban đầu**

Sau hội nghị Dartmouth, các nhà nghiên cứu tin rằng máy móc có thể mô phỏng trí thông minh của con người. Các dự án như Logic Theorist và General Problem Solver ra đời, đặt nền móng cho AI. Tuy nhiên, do hạn chế về dữ liệu và sức mạnh tính toán, AI gặp nhiều khó khăn và dần mất đi sự chú ý.

#### **- Giai đoạn 1970-1980: Sự phát triển của hệ thống chuyên gia**

Trong giai đoạn này, hệ thống chuyên gia (Expert Systems) được phát triển, cho phép máy tính giải quyết các vấn đề phức tạp trong các lĩnh vực như y học và kỹ thuật dựa trên các quy tắc logic được lập trình sẵn. Dù đạt được một số thành công, các hệ thống này vẫn bị giới hạn bởi khả năng xử lý dữ liệu lớn và học hỏi từ kinh nghiệm.

#### **- Giai đoạn 1980-2000: Sự nổi lên của học máy**

Học máy (Machine Learning) bắt đầu phát triển với các thuật toán như mạng nơ-ron nhân tạo và cây quyết định. Không còn phụ thuộc hoàn toàn vào các quy tắc cố định, máy tính giờ đây có thể tự học từ dữ liệu. Tuy nhiên, do thiếu dữ liệu lớn và công nghệ tính toán mạnh mẽ, học máy chưa thực sự tạo ra bước đột phá.

#### **- Giai đoạn 2000-nay: Sự bùng nổ của dữ liệu lớn và học sâu**

Sự phát triển của dữ liệu lớn (Big Data), điện toán đám mây và sức mạnh tính toán đã tạo điều kiện cho sự ra đời và phát triển của học sâu (Deep Learning). Học sâu cho phép AI xử lý khối lượng dữ liệu khổng lồ, trích xuất đặc trưng tự động và đạt được những thành tựu vượt bậc trong các lĩnh vực như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên và xe tự hành. Những cột mốc như AlphaGo của DeepMind là minh chứng rõ ràng cho sự tiến bộ này.

Tóm lại, lịch sử phát triển của AI là một hành trình dài từ những ý tưởng lý thuyết ban đầu đến các ứng dụng thực tiễn ấn tượng, với những bước tiến không ngừng nhờ sự phát triển của công nghệ và dữ liệu.

### **1.3. Các ứng dụng của trí tuệ nhân tạo**

Trí tuệ nhân tạo (AI) không chỉ là một khái niệm công nghệ tiên tiến mà còn là một công cụ mạnh mẽ, thúc đẩy sự chuyển đổi sâu rộng trong nhiều lĩnh vực từ y tế, giao thông, thương mại điện tử đến giáo dục, an ninh và sản xuất. Với khả năng phân tích dữ liệu khổng lồ, học hỏi từ kinh nghiệm và đưa ra quyết định thông minh, AI đang trở thành một phần không thể thiếu trong cuộc sống hiện đại. Dưới đây là các ứng dụng nổi bật của AI, được trình bày chi tiết với ví dụ cụ thể và những tác động thực tế mà công nghệ này mang lại.

### 1.3.1. Y Tế

AI đang thay đổi cách ngành y tế vận hành, từ việc hỗ trợ chẩn đoán chính xác hơn, phát triển thuốc nhanh hơn đến cá nhân hóa phương pháp điều trị cho từng bệnh nhân.

- **Chẩn đoán bệnh lý:** Các hệ thống AI hiện nay có thể phân tích hình ảnh y tế như X-quang, chụp MRI hoặc CT với độ chính xác ấn tượng, đôi khi vượt qua cả các chuyên gia con người. Chẳng hạn, Google đã phát triển một hệ thống AI có khả năng phát hiện ung thư vú từ ảnh chụp nhũ hoa (mammogram) với độ chính xác cao hơn so với bác sĩ trong một số trường hợp. Công nghệ này giúp phát hiện sớm các dấu hiệu bệnh lý, giảm thiểu sai sót do yếu tố con người và hỗ trợ các bác sĩ ở những khu vực thiếu nguồn lực y tế. Một ví dụ khác là công ty Zebra Medical Vision, sử dụng AI để phát hiện các bệnh về gan, phổi và tim qua hình ảnh y tế, mang lại kết quả nhanh chóng và đáng tin cậy.

- **Phát triển thuốc:** Quá trình phát triển một loại thuốc mới thường mất hàng thập kỷ và tiêu tốn hàng tỷ đô la. AI đang thay đổi điều đó bằng cách dự đoán cách các hợp chất hóa học tương tác với cơ thể con người, từ đó rút ngắn thời gian nghiên cứu. Công ty Insilico Medicine đã sử dụng AI để xác định một hợp chất thuốc tiềm năng trong vòng 46 ngày – một kỷ lục so với quy trình truyền thống kéo dài nhiều năm. Tương tự, DeepMind (một công ty con của Alphabet) đã áp dụng AI để dự đoán cấu trúc protein, mở ra cơ hội phát triển các loại thuốc mới cho các bệnh nan y như Alzheimer hay ung thư.

- **Cá nhân hóa điều trị:** AI cho phép các bác sĩ thiết kế phác đồ điều trị phù hợp với từng cá nhân dựa trên dữ liệu y tế, di truyền và lối sống. Hệ thống IBM Watson là một ví dụ điển hình, khi nó phân tích hàng triệu hồ sơ y tế và tài liệu khoa học để đưa ra gợi ý điều trị tối ưu cho bệnh nhân ung thư. Điều này không chỉ tăng hiệu quả điều trị mà còn giảm thiểu tác dụng phụ, mang lại hy vọng cho những trường hợp bệnh phức tạp.

- **Hỗ trợ chăm sóc sức khỏe từ xa:** Trong bối cảnh đại dịch COVID-19, các chatbot AI như Babylon Health đã giúp hàng triệu người tự đánh giá triệu chứng, nhận tư vấn y tế cơ bản và giảm tải cho hệ thống y tế. Những công cụ này sử dụng ngôn ngữ tự nhiên để giao tiếp với bệnh nhân, cung cấp thông tin đáng tin cậy và kết nối họ với bác sĩ khi cần thiết.

### 1.3.2. Giao Thông Vận Tải

AI đang định hình tương lai của giao thông với các phương tiện tự hành, quản lý giao thông thông minh và tối ưu hóa logistics.

- **Xe tự hành:** Các công ty như Tesla, Waymo và Cruise đang tiên phong trong việc phát triển xe tự hành nhờ AI. Hệ thống AI trên xe Tesla sử dụng dữ liệu từ camera, radar và cảm biến để nhận diện môi trường xung quanh, điều hướng qua các tuyến đường phức tạp, tránh chướng ngại vật và tự động đỗ xe. Waymo, một công ty con của Alphabet, đã triển khai dịch vụ taxi tự hành tại một số thành phố ở Mỹ, cho thấy tiềm năng thay thế tài xế con người trong tương lai gần.

- **Quản lý giao thông đô thị:** AI giúp giảm ùn tắc giao thông tại các thành phố lớn thông qua hệ thống đèn giao thông thông minh. Ví dụ, tại Singapore, AI phân tích dữ liệu từ camera giao thông và cảm biến để điều chỉnh thời gian đèn xanh/đỏ theo thời gian thực, giảm thời gian chờ đợi và cải thiện luồng xe cộ. Tương tự, Los Angeles đã áp dụng AI để đồng bộ hóa hệ thống giao thông, giảm lượng khí thải và tiết kiệm nhiên liệu cho người dân.

- **Tối ưu hóa logistics:** Trong ngành vận tải và giao hàng, AI được sử dụng để tối ưu hóa lộ trình, dự đoán nhu cầu và quản lý kho hàng. Công ty UPS sử dụng hệ thống AI có tên ORION (On-Road Integrated Optimization and Navigation) để tính toán lộ trình giao hàng tối ưu, giúp tiết kiệm hàng triệu lít nhiên liệu mỗi năm. DHL cũng áp dụng AI để dự đoán thời gian giao hàng chính xác hơn, dựa trên các yếu tố như thời tiết, giao thông và lịch sử đơn hàng.

- **An toàn giao thông:** AI còn được tích hợp vào các hệ thống cảnh báo va chạm trên xe hơi, như tính năng phanh khẩn cấp tự động (AEB) của Volvo, giúp giảm thiểu tai nạn giao thông nhờ khả năng phát hiện nguy hiểm trong tích tắc.

### 1.3.3. Thương Mại Điện Tử

AI đang biến đổi cách chúng ta mua sắm trực tuyến với các công nghệ cá nhân hóa, tự động hóa và phân tích dữ liệu.

- **Hệ thống đề xuất sản phẩm:** Amazon và Netflix là hai “gã khổng lồ” sử dụng AI để gợi ý sản phẩm và nội dung phù hợp với từng người dùng. Amazon phân tích lịch sử mua sắm, tìm kiếm và thậm chí cả thời gian người dùng dừng lại trên một sản phẩm để đề xuất những mặt hàng liên quan. Netflix thì dựa vào lịch sử xem phim, đánh giá và sở thích để gợi ý các bộ phim hoặc chương trình truyền hình, giữ chân người dùng trên nền tảng lâu hơn.

- **Chatbot hỗ trợ khách hàng:** Các chatbot AI như Virtual Artist của Sephora cho phép khách hàng thử son môi hoặc trang điểm ảo trên khuôn mặt của họ thông qua camera, đồng thời tư vấn sản phẩm phù hợp. H&M cũng sử dụng chatbot trên ứng dụng Kik để giúp khách hàng tìm kiếm quần áo theo phong cách cá nhân. Những công cụ này hoạt động 24/7, giảm chi phí nhân sự và nâng cao trải nghiệm mua sắm.

- **Phân tích dữ liệu và dự đoán xu hướng:** AI giúp các nhà bán lẻ như Walmart hay Alibaba dự đoán nhu cầu thị trường và điều chỉnh giá cả theo thời gian thực. Chẳng hạn, Walmart sử dụng AI để phân tích dữ liệu từ đối thủ cạnh tranh, thời tiết và hành vi người tiêu dùng, từ đó tối ưu hóa giá sản phẩm và đảm bảo nguồn cung đáp ứng nhu cầu.

- **Quảng cáo thông minh:** AI hỗ trợ các chiến dịch quảng cáo trực tuyến bằng cách nhắm mục tiêu chính xác đến từng nhóm khách hàng. Google Ads và Facebook Ads sử dụng AI để phân tích sở thích, hành vi và nhân khẩu học của người dùng, đảm bảo quảng cáo hiển thị đúng người, đúng thời điểm.



#### 1.3.4. Giáo Dục

AI mang đến những cải tiến đáng kể trong giáo dục, từ học tập cá nhân hóa đến hỗ trợ giáo viên và mở rộng giáo dục từ xa.

- **Học tập cá nhân hóa:** Các nền tảng như Duolingo và Khan Academy sử dụng AI để điều chỉnh bài học theo trình độ và tốc độ học của từng người. Duolingo phân tích cách học viên trả lời câu hỏi để tăng hoặc giảm độ khó, trong khi Khan Academy cung cấp bài giảng và bài tập phù hợp với từng cá nhân, giúp họ tiến bộ nhanh hơn.

- **Hỗ trợ giáo viên:** Các công cụ AI như Grammarly không chỉ sửa lỗi chính tả mà còn đưa ra gợi ý cải thiện văn phong, giúp học sinh nâng cao kỹ năng viết. Turnitin sử dụng AI để phát hiện đạo văn trong bài luận, trong khi các hệ thống như Google Classroom tích hợp AI để theo dõi tiến độ học tập và cung cấp phản hồi tự động.

- **Giáo dục từ xa và thực tế ảo:** Trong thời kỳ đại dịch, AI đã hỗ trợ các nền tảng như Zoom bằng cách tự động ghi chú cuộc họp, dịch ngôn ngữ theo thời gian thực và quản lý lớp học ảo. Ngoài ra, AI kết hợp với thực tế ảo (VR) đang được thử nghiệm để tạo ra các lớp học mô phỏng, như phòng thí nghiệm hóa học ảo, giúp học sinh trải nghiệm thực hành mà không cần đến trường.

#### - Đánh giá thông minh:

AI có thể chấm điểm bài thi trắc nghiệm hoặc thậm chí phân tích bài luận mở, như hệ thống của ETS (Educational Testing Service) dùng trong bài thi TOEFL, giúp giảm tải cho giáo viên và đảm bảo tính công bằng.

#### 1.3.5. An Ninh

AI đóng vai trò quan trọng trong việc bảo vệ an ninh cá nhân, doanh nghiệp và xã hội thông qua các công nghệ giám sát và phát hiện mối đe dọa.

- **Nhận diện khuôn mặt:** Công nghệ nhận diện khuôn mặt của Clearview AI được các cơ quan thực thi pháp luật sử dụng để xác định tội phạm qua hình ảnh hoặc video. Tại các sân bay như Changi (Singapore), AI nhận diện khuôn mặt giúp kiểm soát hành khách nhanh chóng, giảm thời gian xếp hàng và tăng cường an ninh.

- **Phát hiện gian lận:** Các ngân hàng như HSBC và dịch vụ thanh toán như PayPal sử dụng AI để phân tích hàng triệu giao dịch mỗi giây, phát hiện các mẫu bất thường như rút tiền bất hợp pháp hay mua sắm đáng ngờ. Khi phát hiện vấn đề, AI có thể tự động chặn giao dịch và cảnh báo người dùng ngay lập tức.

- **Giám sát thông minh:** Các camera an ninh như Nest Cam của Google tích hợp AI để phân tích video, nhận diện chuyển động bất thường (như người lạ đột nhập) và gửi thông báo đến chủ nhà hoặc cảnh sát. Hệ thống này còn phân biệt được giữa người quen và người lạ nhờ học hỏi từ dữ liệu trước đó.

- **An ninh mạng:** AI được sử dụng để bảo vệ dữ liệu số khỏi các cuộc tấn công mạng. Công ty Darktrace sử dụng AI để phát hiện các mối đe dọa trong mạng nội bộ, như mã độc hoặc hành vi bất thường của nhân viên, trước khi chúng gây thiệt hại nghiêm trọng.

#### 1.3.6. Công Nghiệp và Sản Xuất

AI đang tối ưu hóa quy trình sản xuất, nâng cao hiệu suất và giảm chi phí trong các nhà máy hiện đại.

- **Tự động hóa sản xuất:** Các tập đoàn như Siemens và Foxconn sử dụng robot AI để lắp ráp sản phẩm, kiểm tra chất lượng và đóng gói tự động. Robot AI của Tesla trong dây chuyền sản xuất xe điện có thể thực hiện hàng trăm thao tác chính xác, giảm lỗi và tăng tốc độ sản xuất.

- **Bảo trì dự đoán:** General Electric (GE) áp dụng AI để phân tích dữ liệu từ cảm biến trên máy móc công nghiệp, dự đoán thời điểm cần bảo trì và ngăn ngừa hỏng hóc. Trong ngành hàng không, AI giúp kiểm tra động cơ máy bay của Boeing, phát hiện các vấn đề tiềm ẩn trước khi chúng gây nguy hiểm.

- **Quản lý chuỗi cung ứng:** Các công ty như Procter & Gamble sử dụng AI để dự đoán nhu cầu sản phẩm, tối ưu hóa tồn kho và điều phối vận chuyển. AI phân tích dữ liệu từ thị trường, thời tiết và xu hướng tiêu dùng để đảm bảo hàng hóa luôn sẵn sàng mà không lãng phí tài nguyên.

- **Tiết kiệm năng lượng:** AI còn giúp các nhà máy giảm tiêu thụ năng lượng bằng cách tối ưu hóa hoạt động của máy móc. Ví dụ, Google đã áp dụng AI trong các trung tâm dữ liệu của mình, giảm 40% năng lượng làm mát nhờ điều chỉnh thông minh các hệ thống.

Tóm lại, trí tuệ nhân tạo không chỉ là một công nghệ của tương lai mà đã trở thành hiện thực, tác động mạnh mẽ đến mọi khía cạnh của đời sống và công việc. Từ việc cứu sống bệnh nhân trong y tế, giảm ùn tắc giao thông, nâng cao trải nghiệm mua sắm, cá nhân hóa giáo dục, bảo vệ an ninh đến tối ưu hóa sản xuất, AI đang mở ra một kỷ nguyên mới của sự tiện lợi, hiệu quả và sáng tạo. Với tốc độ phát triển hiện tại, chúng ta có thể kỳ vọng AI sẽ tiếp tục mang lại những thay đổi lớn hơn nữa trong thập kỷ tới, định hình cách con người sống và làm việc trên toàn cầu.

### 1.4. Lợi ích và thách thức của trí tuệ nhân tạo

#### 1.4.1. Lợi ích

- **Tăng hiệu quả và năng suất:** Trí tuệ nhân tạo giúp tự động hóa các công việc lặp đi lặp lại và tiêu tốn nhiều thời gian, chẳng hạn như xử lý dữ liệu, kiểm tra lỗi, quản lý kho hàng hay chăm sóc khách hàng. Nhờ đó, các tổ chức có thể tiết kiệm chi phí, rút ngắn thời gian thực hiện công việc, đồng thời nâng cao năng suất lao động tổng thể.

- **Hỗ trợ ra quyết định chính xác hơn:** AI có khả năng phân tích khối lượng lớn dữ liệu trong thời gian ngắn, phát hiện các xu hướng, mô hình và bất thường mà con người có thể bỏ sót. Điều này giúp các doanh nghiệp, tổ chức y tế, tài chính, giao thông... đưa ra quyết định dựa trên dữ liệu (data-driven), từ đó nâng cao độ chính xác và hiệu quả.

- **Cá nhân hóa trải nghiệm người dùng:** Các hệ thống AI có thể học hành vi, sở thích và nhu cầu của từng người dùng để đưa ra các gợi ý phù hợp. Ví dụ: AI trong giáo dục có thể tạo chương trình học riêng cho từng học sinh; trong thương mại điện tử, AI đề xuất sản phẩm theo sở thích cá nhân; trong giải trí, AI chọn nội dung phù hợp với thói quen người xem.

- **Thúc đẩy đổi mới và công nghệ tiên tiến:** Trí tuệ nhân tạo đóng vai trò then chốt trong sự phát triển của nhiều công nghệ mới, như xe tự hành, robot phẫu thuật chính xác trong y học, hệ thống tài chính thông minh, công nghệ dịch ngôn ngữ theo thời gian thực và nhiều lĩnh vực khác. Nhờ AI, con người có thể giải quyết những vấn đề phức tạp mà trước đây khó tiếp cận.

#### 1.4.2. Thách thức

- **Vấn đề đạo đức và quyền riêng tư:** AI thường yêu cầu thu thập và xử lý một lượng lớn dữ liệu cá nhân, đặt ra mối lo ngại về việc lạm dụng thông tin, theo dõi người dùng và xâm phạm quyền riêng tư. Việc sử dụng AI trong nhận diện khuôn mặt, giám sát hành vi hay phân tích tâm lý cũng gây tranh cãi về đạo đức.

- **Nguy cơ thất nghiệp và bất bình đẳng lao động:** Khi AI và tự động hóa dần thay thế con người trong một số công việc, đặc biệt là những công việc đơn giản, lặp lại, nhiều lao động có thể bị mất việc nếu không được đào tạo lại. Điều này đòi hỏi xã hội phải có chính sách chuyển đổi nghề nghiệp và đào tạo lại kỹ năng để thích nghi với thời đại AI.

- **Độ tin cậy và tính minh bạch:** Các hệ thống AI, dù thông minh, vẫn có thể mắc lỗi, bị sai lệch do dữ liệu huấn luyện không đầy đủ hoặc bị khai thác, tấn công từ bên ngoài. Ngoài ra, nhiều thuật toán AI hoạt động như “hộp đen” (black box), khiến người dùng và chuyên gia khó hiểu được cách chúng đưa ra quyết định, từ đó làm giảm niềm tin và khả năng kiểm soát.

- **Thiên vị và phân biệt đối xử trong thuật toán:** Nếu AI được huấn luyện trên các tập dữ liệu thiên lệch (bias), nó có thể củng cố hoặc tái tạo lại các định kiến xã hội về chủng tộc, giới tính, tầng lớp... Điều này đặc biệt nguy hiểm trong các ứng dụng như tuyển dụng, xét duyệt tín dụng, xét xử hình sự, vì có thể dẫn đến quyết định không công bằng hoặc sai lệch.

#### 1.5. Tương lai của trí tuệ nhân tạo

Trí tuệ nhân tạo (AI) đang đứng trước ngưỡng cửa của một kỷ nguyên mới, nơi công nghệ này không chỉ phát triển vượt bậc mà còn len lỏi vào mọi ngóc ngách của cuộc sống. Tương lai của AI hứa hẹn sẽ được định hình bởi những xu hướng đột phá sau:

- **AI tổng quát:** Các nhà khoa học đang dồn sức để tạo ra những hệ thống AI linh hoạt, có thể xử lý nhiều nhiệm vụ khác nhau chẳng kém gì con người. Hãy tưởng tượng một trợ lý ảo không chỉ nhắc bạn lịch hẹn mà còn soạn thảo báo cáo, điều chỉnh nhiệt độ trong nhà, hay thậm chí sáng tác một bài thơ theo tâm trạng của bạn.

- **AI bền vững:** AI sẽ trở thành “người hùng thầm lặng” trong cuộc chiến vì môi trường. Chẳng hạn, các thuật toán thông minh có thể tối ưu hóa mức tiêu thụ điện trong các thành phố hiện đại,

giảm lãng phí trong chuỗi cung ứng, hay dự đoán chính xác hơn các hiện tượng thời tiết cực đoan do biến đổi khí hậu, góp phần bảo vệ hành tinh xanh.

- **Quy định pháp lý:** Khi AI ngày càng trở nên quyền lực, các chính phủ buộc phải hành động. Họ sẽ xây dựng những khung pháp lý chặt chẽ để đảm bảo AI được sử dụng an toàn và công bằng, từ việc bảo vệ dữ liệu cá nhân trước những “con mắt” tò mò của công nghệ, đến ngăn chặn việc lạm dụng AI trong các hoạt động mờ ám.

- **Tích hợp con người - máy:** Sự kết hợp giữa trí tuệ con người và máy móc đang mở ra chân trời mới. Công nghệ giao diện não-máy có thể giúp người khuyết tật điều khiển cánh tay robot chỉ bằng ý nghĩ, hoặc thậm chí nâng cao khả năng tư duy của chúng ta, biến khoa học viễn tưởng thành hiện thực.

## 1.6. Kết luận

Trí tuệ nhân tạo (AI) không còn là khái niệm xa vời của tương lai - nó đã và đang hiện hữu, thay đổi cách chúng ta sống, làm việc và tương tác với thế giới. Từ việc hỗ trợ bác sĩ chẩn đoán bệnh chính xác hơn, đến điều khiển những chiếc xe tự hành an toàn trên đường phố, hay cá nhân hóa trải nghiệm mua sắm và học tập - AI đã chứng minh sức mạnh của mình trong mọi lĩnh vực, từ y tế, giao thông, thương mại điện tử, giáo dục, an ninh đến công nghiệp và sản xuất. Những ứng dụng này không chỉ nâng cao hiệu quả và độ chính xác, mà còn mở ra những cơ hội chưa từng có, giúp con người giải quyết các vấn đề toàn cầu như sức khỏe, môi trường và an ninh.

Nhưng đây mới chỉ là bước khởi đầu. Tương lai của AI hứa hẹn sẽ còn tiến xa hơn, với những đột phá như AI tổng quát có thể thực hiện đa nhiệm vụ như con người, hay sự tích hợp giữa con người và máy móc qua các công nghệ như giao diện não-máy. Tuy nhiên, để tận dụng tối đa tiềm năng của AI, chúng ta cần hợp tác chặt chẽ với công nghệ này, đồng thời đặt ra những quy định pháp lý nghiêm ngặt để đảm bảo nó được sử dụng một cách an toàn và công bằng.

AI không chỉ là một công cụ - nó là một cuộc cách mạng đang diễn ra, và chúng ta đang đứng ở ngã rẽ lịch sử. Liệu AI sẽ trở thành người bạn đồng hành đắc lực hay một thách thức mà nhân loại phải đối mặt? Câu trả lời nằm trong tay chúng ta, và cách chúng ta định hình tương lai này sẽ quyết định không chỉ sự phát triển của công nghệ, mà còn cả tương lai của chính loài người.

## CHƯƠNG 2: CÁC KỸ THUẬT HỌC SÂU

### 2.1. Giới thiệu về học sâu

Học sâu, hay còn được biết đến với tên gọi học cấu trúc sâu, là một nhánh của học máy, sử dụng các mạng nơ-ron nhân tạo với nhiều lớp để trích xuất các đặc trưng phân cấp từ dữ liệu. Về bản chất, học sâu mô phỏng khả năng của bộ não con người trong việc xử lý thông tin phức tạp và đưa ra quyết định dựa trên các mẫu đã học. Sự phát triển của học sâu bắt nguồn từ nỗ lực cải thiện hiệu quả của các kỹ thuật học máy truyền thống, đặc biệt là trong các tác vụ đòi hỏi sự hiểu biết sâu sắc về dữ liệu. Mặc dù các khái niệm cơ bản về mạng nơ-ron đã được hình thành từ những năm 1960,

sự trỗi dậy mạnh mẽ của học sâu chỉ diễn ra gần đây, nhờ vào sự tiến bộ vượt bậc trong khả năng phân tích dữ liệu lớn và sức mạnh tính toán của phần cứng hiện đại. Sự sẵn có của lượng lớn dữ liệu và các đơn vị xử lý đồ họa (GPU) hiệu năng cao đã tạo điều kiện cho việc huấn luyện các mô hình học sâu phức tạp với hàng tỷ tham số, mở ra một kỷ nguyên mới cho các ứng dụng trí tuệ nhân tạo.

Học sâu có mối quan hệ mật thiết với trí tuệ nhân tạo (AI) và học máy (ML). Học sâu là một tập hợp con của học máy, trong khi học máy lại là một lĩnh vực của trí tuệ nhân tạo. Sự khác biệt chính nằm ở kiến trúc và khả năng học đặc trưng. Học sâu sử dụng các mạng nơ-ron sâu, tức là mạng có nhiều lớp ẩn, để tự động trích xuất các đặc trưng phức tạp từ dữ liệu thô. Ngược lại, học máy truyền thống thường yêu cầu con người thiết kế và lựa chọn các đặc trưng phù hợp trước khi đưa vào mô hình.

So với học máy truyền thống, học sâu mang lại nhiều ưu điểm vượt trội. Khả năng tự động trích xuất đặc trưng là một trong những lợi thế quan trọng nhất. Điều này giúp giảm bớt công sức của con người trong việc tìm kiếm và lựa chọn các đặc trưng phù hợp, đồng thời cho phép mô hình học được những đặc trưng ẩn sâu và phức tạp trong dữ liệu mà các phương pháp thủ công có thể bỏ sót. Học sâu đặc biệt hiệu quả trong việc xử lý dữ liệu phi cấu trúc, chẳng hạn như hình ảnh, văn bản và âm thanh, vốn là những loại dữ liệu phổ biến trong thế giới thực. Bên cạnh đó, các mô hình học sâu có khả năng mở rộng tốt với lượng dữ liệu lớn, tận dụng lợi thế của dữ liệu lớn để cải thiện hiệu suất một cách đáng kể. Nhờ những ưu điểm này, học sâu đã chứng minh được khả năng giải quyết các vấn đề phức tạp với độ chính xác cao, thường vượt trội hơn các phương pháp học máy truyền thống trong nhiều lĩnh vực.

## 2.2. Nền tảng lý thuyết

### 2.2.1. Từ học máy đến học sâu

Học máy là một tập hợp các kỹ thuật cho phép máy tính học từ dữ liệu mà không cần được lập trình một cách tường minh. Các phương pháp học máy truyền thống như hồi quy tuyến tính, máy vector hỗ trợ (SVM), hay rừng ngẫu nhiên (Random Forest) thường đòi hỏi việc trích xuất đặc trưng thủ công - một quá trình yêu cầu chuyên môn trong lĩnh vực cụ thể và tốn nhiều công sức.

Học sâu, một phân nhánh của học máy, sử dụng các mạng nơ-ron nhân tạo nhiều tầng để học các biểu diễn phân cấp của dữ liệu. Sự khác biệt chính giữa học máy truyền thống và học sâu là:

1. **Tự động trích xuất đặc trưng:** Học sâu tự động phát hiện các đặc trưng quan trọng từ dữ liệu thô mà không cần sự can thiệp của con người.
2. **Khả năng mở rộng:** Hiệu suất của các mô hình học sâu thường tăng khi được cung cấp nhiều dữ liệu hơn và quy mô mô hình lớn hơn.
3. **Học biểu diễn phân cấp:** Các mạng học sâu học các biểu diễn ở nhiều mức độ trừu tượng khác nhau, với mỗi lớp biến đổi biểu diễn từ lớp trước thành biểu diễn trừu tượng hơn.

### 2.2.2. Mạng nơ-ron nhân tạo

Mạng nơ-ron nhân tạo, nền tảng của học sâu, được lấy cảm hứng từ cấu trúc của não người. Cấu trúc cơ bản của mạng nơ-ron nhân tạo bao gồm:

1. **Nơ-ron (neuron):** Đơn vị tính toán cơ bản, còn được gọi là “nút” hay “đơn vị”.
2. **Lớp (layer):** Tập hợp các nơ-ron cùng cấp. Một mạng nơ-ron điển hình có:
  - **Lớp đầu vào (input layer):** Nhận dữ liệu thô.
  - **Lớp ẩn (hidden layer):** Xử lý dữ liệu thông qua các phép biến đổi phi tuyến. Mạng học sâu có nhiều lớp ẩn.
  - **Lớp đầu ra (output layer):** Cung cấp kết quả dự đoán cuối cùng.
3. **Trọng số (weights) và độ lệch (biases):** Các tham số có thể điều chỉnh của mạng, xác định cách dữ liệu được biến đổi.
4. **Hàm kích hoạt (activation function):** Hàm phi tuyến áp dụng cho đầu ra của mỗi nơ-ron, giúp mạng học được các mối quan hệ phức tạp trong dữ liệu.

Toán học đằng sau một nơ-ron đơn giản như sau:

- Đầu vào:  $\mathbf{x} = [x_1, x_2, \dots, x_n]$
- Trọng số:  $\mathbf{w} = [w_1, w_2, \dots, w_n]$
- Độ lệch:  $b$
- Đầu ra:  $y = f(\mathbf{w}^T \mathbf{x} + b)$

Trong đó  $f$  là hàm kích hoạt phi tuyến như sigmoid, tanh, hoặc ReLU.

### 2.2.3. Quá trình học trong mạng nơ-ron

Quá trình học trong mạng nơ-ron bao gồm hai bước chính:

1. **Lan truyền xuôi (Forward propagation):**
  - Dữ liệu đầu vào được truyền qua mạng, từ lớp đầu vào đến lớp đầu ra.
  - Tại mỗi lớp, dữ liệu được biến đổi dựa trên trọng số, độ lệch và hàm kích hoạt.
  - Kết quả cuối cùng được so sánh với kết quả mong đợi để tính toán lỗi.
2. **Lan truyền ngược (Backpropagation):**
  - Lỗi được truyền ngược từ lớp đầu ra về lớp đầu vào.
  - Thuật toán tính toán đạo hàm của hàm lỗi đối với từng tham số của mạng.

- Các tham số được cập nhật để giảm thiểu lỗi sử dụng các thuật toán tối ưu hóa như Gradient Descent.

Hàm mất mát (loss function) đo lường sự chênh lệch giữa kết quả dự đoán và kết quả thực tế. Các hàm mất mát phổ biến bao gồm:

- Mean Squared Error (MSE) cho các bài toán hồi quy
- Cross-Entropy cho các bài toán phân loại

## 2.3. Các kiến trúc học sâu cơ bản

### 2.3.1. Mạng nơ-ron tích chập (CNN)

Mạng nơ-ron tích chập (Convolutional Neural Network - CNN) là một kiến trúc được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc lưới như hình ảnh. Được giới thiệu bởi Yann LeCun vào những năm 1990, CNN đã tạo ra đột phá trong lĩnh vực nhận dạng hình ảnh và thị giác máy tính.

#### **Đặc điểm chính của CNN:**

##### **1. Các lớp tích chập (Convolutional layers):**

- Sử dụng bộ lọc (filter) hoặc kernel để trích xuất đặc trưng cục bộ từ dữ liệu đầu vào
- Tham số được chia sẻ giúp giảm số lượng tham số cần học
- Công thức tích chập:  $S(i, j) = \sum_m \sum_n K(m, n) \cdot I(i - m, j - n)$  trong đó  $K$  là kernel và  $I$  là dữ liệu đầu vào

##### **2. Lớp gộp (Pooling layers):**

- Giảm kích thước không gian của biểu diễn, giúp giảm tham số và tính toán
- Các loại phổ biến: max pooling (lấy giá trị lớn nhất) và average pooling (lấy giá trị trung bình)

##### **3. Bất biến tịnh tiến (Translation invariance):**

- CNN có khả năng nhận biết đối tượng bất kể vị trí của nó trong hình ảnh

##### **4. Học phân cấp:**

- Các lớp đầu học các đặc trưng đơn giản như cạnh, góc
- Các lớp sau học các đặc trưng phức tạp hơn như mắt, mũi, và cuối cùng là khuôn mặt hoàn chỉnh

#### **Kiến trúc CNN điển hình:**

- Lớp đầu vào: Hình ảnh (ví dụ:  $224 \times 224 \times 3$  cho RGB)

- Nhiều khối lặp lại của các lớp tích chập + hàm kích hoạt + lớp gộp
- Lớp làm phẳng (flatten): Chuyển đổi dữ liệu đa chiều thành véc-tơ
- Các lớp kết nối đầy đủ (fully connected layers)
- Lớp đầu ra: Sử dụng softmax cho phân loại

#### Các kiến trúc CNN nổi bật:

- LeNet-5: Kiến trúc tiên phong cho nhận dạng chữ số viết tay
- AlexNet: Chiến thắng ImageNet 2012, đánh dấu sự bùng nổ của học sâu
- VGG: Kiến trúc đơn giản nhưng sâu với các lớp tích chập  $3 \times 3$
- ResNet: Giới thiệu các kết nối tắt (skip connections) cho phép đào tạo mạng rất sâu
- Inception/GoogLeNet: Sử dụng các mô-đun Inception với các phép tích chập đa kích thước

#### 2.3.2. Mạng nơ-ron hồi quy (RNN)

Mạng nơ-ron hồi quy (Recurrent Neural Network - RNN) là một họ mạng nơ-ron được thiết kế cho dữ liệu tuần tự như văn bản, âm thanh, hoặc chuỗi thời gian. Khác với mạng nơ-ron truyền thẳng, RNN có các kết nối vòng lặp cho phép thông tin từ các bước thời gian trước được lưu giữ và ảnh hưởng đến quyết định hiện tại.

#### Đặc điểm chính của RNN:

1. **Bộ nhớ ngắn hạn:** RNN có thể lưu trữ thông tin từ các đầu vào trước đó thông qua trạng thái ẩn (hidden state)
2. **Trọng số được chia sẻ:** Cùng một tập tham số được sử dụng tại mỗi bước thời gian, giúp mô hình có thể xử lý chuỗi có độ dài khác nhau
3. **Tính toán tại mỗi bước thời gian  $t$ :**

$$- \text{Trạng thái ẩn: } h_t = f(W_{hh} \cdot h_{t-1} + W_{xh} \cdot x_t + b_h)$$

$$- \text{Đầu ra: } y_t = W_{hy} \cdot h_t + b_y$$

Trong đó:

- $x_t$  là đầu vào tại thời điểm  $t$
- $h_t$  là trạng thái ẩn tại thời điểm  $t$
- $h_{t-1}$  là trạng thái ẩn từ thời điểm trước
- $y_t$  là đầu ra tại thời điểm  $t$
- $W$  là các ma trận trọng số,  $b$  là các véc-tơ độ lệch



- $f$  là hàm kích hoạt phi tuyến (thường là tanh hoặc ReLU)

#### Hạn chế của RNN cơ bản:

- **Vấn đề gradient biến mất/bùng nổ:** Khi lan truyền ngược qua nhiều bước thời gian, gradient có thể trở nên rất nhỏ (biến mất) hoặc rất lớn (bùng nổ)
- **Khó khăn trong việc nắm bắt phụ thuộc dài hạn:** RNN cơ bản thường gặp khó khăn khi phải liên kết thông tin cách xa nhau trong chuỗi

Những hạn chế này đã dẫn đến sự phát triển của các kiến trúc RNN nâng cao như LSTM và GRU.

#### 2.3.3. Mạng trí nhớ dài-ngắn hạn (LSTM)

Mạng trí nhớ dài-ngắn hạn (Long Short-Term Memory - LSTM) được giới thiệu bởi Hochreiter và Schmidhuber vào năm 1997 để giải quyết vấn đề gradient biến mất trong RNN truyền thống. LSTM là một loại RNN đặc biệt có khả năng học các phụ thuộc dài hạn.

#### Kiến trúc của LSTM:

LSTM giới thiệu khái niệm về “tế bào trí nhớ” (memory cell) và các “cổng” (gates) để kiểm soát luồng thông tin:

1. **Tế bào trí nhớ (Cell state)  $C_t$ :** Đóng vai trò như một băng chuyền thông tin dọc theo chuỗi, cho phép thông tin quan trọng được giữ lại trong thời gian dài
2. **Ba cổng chính:**
  - **Cổng quên (Forget gate)  $f_t$ :** Quyết định thông tin nào từ trạng thái tế bào trước đó cần được loại bỏ
  - **Cổng đầu vào (Input gate)  $i_t$ :** Quyết định thông tin mới nào cần được lưu trữ vào trạng thái tế bào
  - **Cổng đầu ra (Output gate)  $o_t$ :** Quyết định phần nào của trạng thái tế bào sẽ được đưa ra làm đầu ra
3. **Các phương trình cập nhật:**
  - Cổng quên:  $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
  - Cổng đầu vào:  $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
  - Giá trị ứng viên:  $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$
  - Cập nhật trạng thái tế bào:  $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$
  - Cổng đầu ra:  $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$
  - Trạng thái ẩn:  $h_t = o_t * \tanh(C_t)$

Trong đó:

- $\sigma$  là hàm sigmoid, có giá trị từ 0 đến 1, xác định mức độ thông tin được cho phép đi qua
- $*$  là phép nhân Hadamard (nhân từng phần tử)
- $[h_{t-1}, x_t]$  là phép nối véc-tơ  $h_{t-1}$  và  $x_t$

#### **Ưu điểm của LSTM:**

- Giải quyết vấn đề gradient biến mất bằng cách sử dụng cơ chế cổng và đường đi không biến đổi thông qua trạng thái tế bào
- Có khả năng nắm bắt phụ thuộc dài hạn
- Hiệu quả trong nhiều tác vụ xử lý chuỗi như dịch máy, nhận dạng giọng nói, và phân tích cảm xúc

#### **Các biến thể của LSTM:**

- LSTM hai chiều (Bidirectional LSTM): Xử lý chuỗi theo cả hai hướng (từ trái sang phải và từ phải sang trái)
- LSTM chồng (Stacked LSTM): Nhiều lớp LSTM được xếp chồng lên nhau để tăng khả năng trừu tượng hóa
- LSTM có cơ chế chú ý (LSTM with attention): Kết hợp cơ chế chú ý để tập trung vào các phần quan trọng của chuỗi đầu vào

#### **2.3.4. Mạng nơ-ron cổng hồi quy (GRU)**

Mạng nơ-ron cổng hồi quy (Gated Recurrent Unit - GRU) được giới thiệu bởi Cho và cộng sự vào năm 2014 như một phiên bản đơn giản hóa của LSTM. GRU giữ lại khả năng học các phụ thuộc dài hạn nhưng với ít tham số hơn, làm cho nó nhanh hơn và dễ huấn luyện hơn trong một số trường hợp.

#### **Kiến trúc của GRU:**

GRU kết hợp cổng quên và cổng đầu vào của LSTM thành một “cổng cập nhật” duy nhất, và hợp nhất trạng thái tế bào và trạng thái ẩn:

1. **Cổng đặt lại (Reset gate)  $r_t$ :** Quyết định mức độ thông tin từ quá khứ được kết hợp với thông tin đầu vào mới
2. **Cổng cập nhật (Update gate)  $z_t$ :** Quyết định mức độ thông tin từ quá khứ được giữ lại
3. **Các phương trình cập nhật:**
  - Cổng đặt lại:  $r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$

- Công cập nhật:  $z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$
- Ứng viên trạng thái ẩn:  $\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t] + b)$
- Trạng thái ẩn cuối cùng:  $h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$

Trong đó:

- $\sigma$  là hàm sigmoid
- $*$  là phép nhân Hadamard
- $[h_{t-1}, x_t]$  là phép nối véc-tơ

#### So sánh GRU và LSTM:

- GRU có ít tham số hơn LSTM (hai cổng so với ba cổng)
- GRU không có trạng thái tế bào riêng biệt
- GRU thường huấn luyện nhanh hơn và yêu cầu ít dữ liệu hơn để khái quát hóa
- LSTM có thể mạnh hơn khi xử lý các chuỗi dài và các bài toán phức tạp
- Không có người chiến thắng rõ ràng; hiệu suất phụ thuộc vào tác vụ và bộ dữ liệu cụ thể

#### Ứng dụng của GRU:

- Xử lý ngôn ngữ tự nhiên
- Nhận dạng giọng nói
- Phân tích chuỗi thời gian
- Dịch máy
- Tạo văn bản

#### 2.3.5. Bộ mã hóa-giải mã (Autoencoder)

Bộ mã hóa-giải mã (Autoencoder) là một kiến trúc mạng nơ-ron học không giám sát được thiết kế để học các biểu diễn nén (mã hóa) của dữ liệu đầu vào, sau đó tái tạo lại dữ liệu đầu vào từ các biểu diễn đã nén này. Mục tiêu chính là học một biểu diễn ẩn (latent representation) hoặc mã hóa của dữ liệu đầu vào, thường có chiều thấp hơn, để nắm bắt các đặc trưng quan trọng nhất.

#### Cấu trúc cơ bản của Autoencoder:

1. **Bộ mã hóa (Encoder):** Chuyển đổi dữ liệu đầu vào  $x$  thành biểu diễn ẩn  $z$ :  $z = f(Wx + b)$
2. **Bộ giải mã (Decoder):** Tái tạo dữ liệu đầu vào từ biểu diễn ẩn:  $\hat{x} = g(W'z + b')$

3. **Hàm mất mát:** Thường là lỗi tái tạo, đo lường sự khác biệt giữa đầu vào và đầu ra:  
$$L(x, \hat{x}) = \|x - \hat{x}\|^2$$

Trong đó:

- $f$  và  $g$  là các hàm kích hoạt phi tuyến (thường là sigmoid, tanh, hoặc ReLU)
- $W$  và  $W'$  là các ma trận trọng số (trong nhiều trường hợp,  $W' = W^T$ )
- $b$  và  $b'$  là các véc-tơ độ lệch

**Các loại Autoencoder:**

1. **Autoencoder cơ bản (Vanilla Autoencoder):**
  - Sử dụng các lớp kết nối đầy đủ (fully connected layers)
  - Học biểu diễn nén tuyến tính hoặc phi tuyến của dữ liệu
2. **Autoencoder chính quy hóa (Regularized Autoencoder):**
  - Thêm các ràng buộc vào quá trình huấn luyện để tránh việc chỉ đơn giản sao chép dữ liệu đầu vào
  - Ví dụ: Sparse Autoencoder, Denoising Autoencoder, Contractive Autoencoder
3. **Autoencoder nhiễu (Denoising Autoencoder):**
  - Được huấn luyện để khôi phục dữ liệu gốc từ phiên bản bị nhiễu
  - Giúp mô hình học biểu diễn mạnh mẽ hơn và tránh việc học đơn thuần
4. **Autoencoder biến phân (Variational Autoencoder - VAE):**
  - Mô hình sinh xác suất học phân phối của dữ liệu
  - Mã hóa đầu vào thành phân phối xác suất trong không gian ẩn thay vì một điểm cố định
  - Cho phép tạo ra dữ liệu mới bằng cách lấy mẫu từ không gian ẩn
5. **Autoencoder tích chập (Convolutional Autoencoder):**
  - Sử dụng các lớp tích chập trong cả bộ mã hóa và bộ giải mã
  - Hiệu quả cho dữ liệu hình ảnh, giữ được thông tin không gian

**Ứng dụng của Autoencoder:**

- Giảm chiều dữ liệu và trích xuất đặc trưng
- Phát hiện bất thường (anomaly detection)

- Khử nhiễu hình ảnh và âm thanh
- Hoàn thiện dữ liệu bị thiếu (data imputation)
- Tạo sinh dữ liệu (với VAE và các mô hình sinh khác)
- Học biểu diễn để sử dụng trong các tác vụ khác

## 2.4. Kiến trúc học sâu tiên tiến

### 2.4.1. Mạng Transformer

Kiến trúc Transformer, được giới thiệu trong bài báo “Attention Is All You Need” bởi Vaswani và cộng sự (2017), đã tạo ra một cuộc cách mạng trong lĩnh vực xử lý ngôn ngữ tự nhiên và sau đó lan sang nhiều lĩnh vực khác. Không giống như RNN và CNN, Transformer hoàn toàn dựa vào cơ chế tự chú ý (self-attention) và không sử dụng tính chất hồi quy.

#### Các thành phần chính của Transformer:

##### 1. Cơ chế chú ý (Attention Mechanism):

- Cho phép mô hình tập trung vào các phần khác nhau của chuỗi đầu vào khi tạo ra đầu ra
- Công thức chú ý:  $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ 
  - $Q$  (Query),  $K$  (Key),  $V$  (Value) là các biến đổi tuyến tính của đầu vào
  - $d_k$  là chiều của véc-tơ khóa

##### 2. Tự chú ý đa đầu (Multi-head Attention):

- Cho phép mô hình đồng thời chú ý đến thông tin từ các vị trí khác nhau từ các góc độ biểu diễn khác nhau
- Kết hợp nhiều “đầu” chú ý song song, mỗi đầu học một kiểu phụ thuộc khác nhau

##### 3. Mã hóa vị trí (Positional Encoding):

- Thêm thông tin về vị trí tương đối của từng token trong chuỗi
- Sử dụng các hàm sin và cos với tần số khác nhau:
  - $PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{model}})$
  - $PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{model}})$

##### 4. Kiến trúc Encoder-Decoder:

- **Encoder:** Biến đổi chuỗi đầu vào thành biểu diễn liên tục ẩn

- Gồm N khối giống nhau, mỗi khối có hai lớp con:
  - Lớp tự chú ý đa đầu
  - Mạng feed-forward vị trí khôn (position-wise feed-forward network)
- Sử dụng các kết nối còn lại (residual connections) và chuẩn hóa lớp
  - **Decoder:** Tạo ra chuỗi đầu ra một token tại một thời điểm
    - Tương tự như encoder nhưng thêm một lớp chú ý đa đầu thứ hai
    - Sử dụng masked attention để ngăn vị trí hiện tại nhìn thấy vị trí tương lai

#### Ưu điểm của Transformer:

- Xử lý song song, cho phép huấn luyện nhanh hơn so với các mô hình tuần tự
- Nắm bắt hiệu quả các phụ thuộc dài hạn
- Khả năng mở rộng tốt với các bộ dữ liệu và kích thước mô hình lớn
- Tính đa năng cao, có thể áp dụng cho nhiều loại dữ liệu và tác vụ khác nhau

#### Các mô hình dựa trên Transformer:

- **BERT** (Bidirectional Encoder Representations from Transformers): Mô hình encoder hai chiều tiền huấn luyện cho NLP
- **GPT** (Generative Pre-trained Transformer): Mô hình decoder chỉ tự hồi quy cho tạo văn bản
- **T5** (Text-to-Text Transfer Transformer): Chuyển đổi mọi tác vụ NLP thành định dạng văn bản-đến-văn bản
- **ViT** (Vision Transformer): Áp dụng Transformer cho xử lý hình ảnh
- **DALL-E, Stable Diffusion:** Áp dụng Transformer cho tạo hình ảnh từ văn bản

Transformer đã trở thành nền tảng cho hầu hết các mô hình ngôn ngữ lớn (LLM) hiện đại và đang được áp dụng ngày càng nhiều trong các lĩnh vực khác như thị giác máy tính, xử lý âm thanh, và khoa học cuộc sống.

#### 2.4.2. Mô hình sinh (Generative Models)

Mô hình sinh là một lớp mô hình học sâu được thiết kế để học phân phối xác suất của dữ liệu và có khả năng tạo ra dữ liệu mới tương tự với dữ liệu huấn luyện. Những mô hình này đã dẫn đến những tiến bộ đáng kể trong việc tạo ra hình ảnh, âm thanh, văn bản và các loại dữ liệu khác với độ chân thực ngày càng cao.

## **Các loại mô hình sinh chính:**

### **1. Mô hình sinh tiềm ẩn (Latent Variable Models):**

- Học một biểu diễn ẩn của dữ liệu
- Ví dụ: Variational Autoencoder (VAE)

### **2. Mô hình dòng (Flow-based Models):**

- Sử dụng các biến đổi có thể đảo ngược để ánh xạ giữa không gian dữ liệu và không gian ẩn
- Ví dụ: Real NVP, Glow, Normalizing Flows

### **3. Mô hình tự hồi quy (Autoregressive Models):**

- Mô hình hóa phân phối xác suất có điều kiện của từng phần tử dữ liệu dựa trên các phần tử trước đó
- Ví dụ: PixelCNN, WaveNet, GPT

### **4. Mô hình đối kháng (Adversarial Models):**

- Sử dụng cơ chế đối kháng giữa hai mạng để tạo ra dữ liệu mới
- Ví dụ: Generative Adversarial Network (GAN)

### **5. Mô hình khuếch tán (Diffusion Models):**

- Dần dần thêm nhiễu vào dữ liệu và học cách đảo ngược quá trình này
- Ví dụ: DDPM (Denoising Diffusion Probabilistic Models), Stable Diffusion

## **Ứng dụng của mô hình sinh:**

- Tổng hợp hình ảnh và thao tác hình ảnh
- Tạo văn bản và ngôn ngữ tự nhiên
- Tổng hợp giọng nói và âm nhạc
- Tạo dữ liệu tăng cường (data augmentation)
- Hoàn thiện dữ liệu bị thiếu
- Thiết kế phân tử trong khoa học dược phẩm
- Khám phá không gian thiết kế trong kiến trúc và kỹ thuật

### 2.4.3. Mạng đối kháng sinh (GAN)

Mạng đối kháng sinh (Generative Adversarial Network - GAN) được giới thiệu bởi Ian Goodfellow và cộng sự vào năm 2014, là một khuôn khổ học sâu đột phá cho các mô hình sinh. GAN bao gồm hai mạng nơ-ron cạnh tranh với nhau trong một quy trình đào tạo đối kháng.

#### Kiến trúc và cơ chế hoạt động của GAN:

##### 1. Hai thành phần chính:

- **Generator (G):** Tạo ra dữ liệu giả từ nhiễu ngẫu nhiên, cố gắng lừa Discriminator
- **Discriminator (D):** Phân biệt giữa dữ liệu thật và dữ liệu giả do Generator tạo ra

##### 2. Quá trình đào tạo:

- G và D được đào tạo đồng thời trong một trò chơi minimax hai người
- G cố gắng tối đa hóa xác suất D phân loại sai
- D cố gắng tối đa hóa khả năng phân biệt chính xác giữa dữ liệu thật và giả

##### 3. Hàm mục tiêu: $\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]$

Trong đó:

- $p_{\text{data}}(x)$  là phân phối của dữ liệu thật
- $p_z(z)$  là phân phối của nhiễu đầu vào cho Generator
- $G(z)$  là dữ liệu giả được tạo từ nhiễu  $z$
- $D(x)$  là xác suất mà D phân loại  $x$  là dữ liệu thật

#### Các biến thể GAN quan trọng:

##### 1. DCGAN (Deep Convolutional GAN):

- Kết hợp CNN với GAN
- Sử dụng các lớp tích chập và chuyển vị tích chập
- Chuẩn hóa theo lô và các hướng dẫn kiến trúc khác để ổn định việc đào tạo

##### 2. CGAN (Conditional GAN):

- Điều kiện hóa cả G và D trên một số thông tin bổ sung
- Cho phép tạo ra đầu ra có điều kiện (ví dụ: hình ảnh của một lớp cụ thể)



### 3. **CycleGAN:**

- Chuyển đổi hình ảnh giữa các miền khác nhau mà không cần dữ liệu được ghép cặp
- Sử dụng ràng buộc chu kỳ nhất quán để đảm bảo ánh xạ một-một

### 4. **StyleGAN:**

- Cho phép kiểm soát phong cách của hình ảnh được tạo ra ở các độ phân giải khác nhau
- Tạo ra hình ảnh chất lượng cao với khả năng điều chỉnh các thuộc tính cụ thể

### 5. **BigGAN:**

- GAN quy mô lớn với nhiều cải tiến trong kiến trúc và phương pháp đào tạo
- Tạo ra hình ảnh chất lượng cao và đa dạng

### **Thách thức trong đào tạo GAN:**

#### 1. **Không ổn định trong đào tạo:**

- Khó đạt được cân bằng giữa G và D
- Vấn đề gradient biến mất và mode collapse

#### 2. **Mode collapse:**

- G tạo ra một tập hợp đầu ra hạn chế, không nắm bắt được toàn bộ phân phối dữ liệu

#### 3. **Các biện pháp khắc phục:**

- Wasserstein GAN (WGAN): Sử dụng khoảng cách Wasserstein để đo lường sự khác biệt giữa phân phối
- Gradient penalty: Giúp ổn định đào tạo
- Spectral normalization: Hạn chế độ lớn của ma trận trọng số
- Progressive growing: Đào tạo dần dần từ độ phân giải thấp đến cao

### **Ứng dụng của GAN:**

- Tổng hợp hình ảnh siêu thực
- Chuyển đổi hình ảnh (image-to-image translation)
- Tăng độ phân giải hình ảnh (super-resolution)
- Phục hồi hình ảnh và loại bỏ nhiễu
- Thiết kế sản phẩm và thời trang

- Tạo dữ liệu đào tạo cho các tác vụ khác
- Tổng hợp dữ liệu y tế (như hình ảnh y tế)

#### 2.4.4. Mạng học khuếch tán (Diffusion Models)

Mô hình khuếch tán (Diffusion Models) là một lớp mô hình sinh tương đối mới đã đạt được những kết quả ấn tượng trong việc tạo ra hình ảnh, âm thanh, và văn bản chất lượng cao. Các mô hình này hoạt động bằng cách học quá trình đảo ngược của một chuỗi Markov khuếch tán, trong đó dữ liệu dần dần bị phá hủy bằng cách thêm nhiễu Gaussian, sau đó được khôi phục bằng cách học cách loại bỏ nhiễu.

##### Nguyên lý hoạt động:

- Quá trình chuyển tiếp tiến (Forward diffusion process):**
  - Dần dần thêm nhiễu Gaussian vào dữ liệu gốc qua một chuỗi các bước
  - Sau đủ bước, dữ liệu gốc biến thành nhiễu Gaussian thuần túy
  - Mỗi bước tuân theo một phương trình khuếch tán:  $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t\mathbf{I})$
  - Trong đó  $\beta_t$  là lịch trình nhiễu (thường tăng dần theo thời gian)
- Quá trình khuếch tán ngược (Reverse diffusion process):**
  - Bắt đầu từ nhiễu thuần túy  $x_T \sim \mathcal{N}(0, \mathbf{I})$
  - Dần dần loại bỏ nhiễu để tạo ra mẫu dữ liệu  $x_0$
  - Mô hình học để dự đoán trung bình của phân phối  $p_\theta(x_{t-1}|x_t)$
  - Thường được tham số hóa để dự đoán nhiễu đã thêm vào hoặc giá trị  $x_0$  trực tiếp

##### Các biến thể chính:

- DDPM (Denoising Diffusion Probabilistic Models):**
  - Mô hình khuếch tán đầu tiên đạt được kết quả cạnh tranh với GAN
  - Sử dụng lịch trình nhiễu tuyến tính và nhiều bước lấy mẫu
- DDIM (Denoising Diffusion Implicit Models):**
  - Cải tiến cho phép lấy mẫu nhanh hơn đáng kể
  - Không phải là một quá trình Markov, cho phép lấy mẫu phi ngẫu nhiên
- Latent Diffusion Models (LDM):**
  - Thực hiện khuếch tán trong không gian ẩn thay vì không gian pixel

- Giảm đáng kể chi phí tính toán
- Cơ sở của Stable Diffusion, một mô hình tạo hình ảnh từ văn bản phổ biến

#### 4. **Guided Diffusion:**

- Sử dụng hướng dẫn phân loại để điều khiển quá trình tạo
- Cải thiện chất lượng mẫu và khả năng điều khiển

#### 5. **Score-based Generative Models:**

- Cách tiếp cận liên quan chặt chẽ tập trung vào gradient của log-likelihood

#### **Ưu điểm của mô hình khuếch tán:**

- Chất lượng mẫu cực kỳ cao, thường vượt qua GAN
- Đa dạng hơn, ít bị mode collapse hơn
- Quá trình đào tạo ổn định hơn
- Có thể điều kiện hóa linh hoạt trên nhiều loại đầu vào (văn bản, hình ảnh, v.v.)
- Khung xác suất rõ ràng

#### **Nhược điểm:**

- Thời gian lấy mẫu chậm (mặc dù đã có nhiều cải tiến)
- Đòi hỏi tài nguyên tính toán lớn
- Thường cần nhiều bước để tạo ra mẫu chất lượng cao

#### **Ứng dụng:**

- Tạo hình ảnh từ văn bản (text-to-image) - như Stable Diffusion, DALL-E 2, Midjourney
- Siêu phân giải (super-resolution)
- Hoàn thiện hình ảnh (image inpainting)
- Tạo âm thanh và âm nhạc
- Thiết kế phân tử và phát triển thuốc
- Tạo video

Mô hình khuếch tán đang nhanh chóng trở thành công nghệ hàng đầu cho việc tạo nội dung chất lượng cao và đa phương thức, với những cải tiến mới xuất hiện thường xuyên.

## 2.5. Kỹ thuật huấn luyện và tối ưu hóa

### 2.5.1. Hàm kích hoạt

Hàm kích hoạt (activation function) là một thành phần quan trọng trong mạng nơ-ron, giới thiệu tính phi tuyến vào mô hình và cho phép mạng học các mối quan hệ phức tạp. Lựa chọn hàm kích hoạt phù hợp có thể ảnh hưởng đáng kể đến hiệu suất và khả năng hội tụ của mạng.

#### Các hàm kích hoạt phổ biến:

##### 1. Sigmoid:

- Công thức:  $\sigma(x) = \frac{1}{1+e^{-x}}$
- Khoảng giá trị:  $(0, 1)$
- Ưu điểm: Muọt, có thể diễn giải như xác suất
- Nhược điểm: Bão hòa gradient ở cả hai đầu, không lấy trung tâm ở 0, vấn đề gradient biến mất

##### 2. Tanh (Hyperbolic Tangent):

- Công thức:  $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- Khoảng giá trị:  $(-1, 1)$
- Ưu điểm: Lấy trung tâm ở 0, muọt
- Nhược điểm: Vẫn bị bão hòa ở cả hai đầu

##### 3. ReLU (Rectified Linear Unit):

- Công thức:  $\text{ReLU}(x) = \max(0, x)$
- Khoảng giá trị:  $[0, +\infty)$
- Ưu điểm: Tính toán đơn giản, giảm vấn đề gradient biến mất, thúc đẩy tính thưa thớt
- Nhược điểm: “Neuron chết” (khi gradient = 0 cho đầu vào âm), không lấy trung tâm ở 0

##### 4. Leaky ReLU:

- Công thức:  $\text{LeakyReLU}(x) = \max(\alpha x, x)$  với  $\alpha$  nhỏ (thường là 0.01)
- Khoảng giá trị:  $(-\infty, +\infty)$
- Ưu điểm: Khắc phục vấn đề neuron chết của ReLU

- Nhược điểm: Tham số  $\alpha$  thêm vào cần được điều chỉnh

#### 5. ELU (Exponential Linear Unit):

- Công thức: 
$$\text{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{if } x \leq 0 \end{cases}$$
- Ưu điểm: Có thể tạo ra đầu ra trung bình gần 0, giúp tăng tốc học tập
- Nhược điểm: Tính toán phức tạp hơn ReLU

#### 6. GELU (Gaussian Error Linear Unit):

- Công thức:  $\text{GELU}(x) = x \cdot \Phi(x)$  với  $\Phi$  là hàm phân phối tích lũy của phân phối chuẩn
- Ưu điểm: Hiệu suất tốt trong các mô hình Transformer hiện đại
- Nhược điểm: Tính toán phức tạp hơn

#### 7. Swish:

- Công thức:  $\text{Swish}(x) = x \cdot \sigma(\beta x)$
- Ưu điểm: Hiệu suất tốt trong các mạng sâu, không bị bão hòa ở vùng dương
- Nhược điểm: Tính toán phức tạp hơn ReLU

#### Lựa chọn hàm kích hoạt:

- **ReLU** thường là lựa chọn mặc định tốt cho hầu hết các lớp ẩn
- **Sigmoid** thường được sử dụng cho lớp đầu ra trong phân loại nhị phân
- **Softmax** cho lớp đầu ra trong phân loại đa lớp
- **Tanh** thường được sử dụng trong RNN và các kiến trúc nhất định
- **GELU** và **Swish** thường được sử dụng trong các mô hình Transformer và mạng rất sâu

#### Tác động của hàm kích hoạt:

- Ảnh hưởng đến khả năng hội tụ và tốc độ hội tụ
- Có thể giúp giảm thiểu vấn đề gradient biến mất/bùng nổ
- Ảnh hưởng đến khả năng của mạng trong việc học các mối quan hệ phức tạp
- Có thể ảnh hưởng đến tính thừa thớt của biểu diễn

### 2.5.2. Các thuật toán tối ưu hóa

Thuật toán tối ưu hóa đóng vai trò quan trọng trong việc huấn luyện mạng học sâu hiệu quả, giúp điều chỉnh trọng số và độ lệch để giảm thiểu hàm mất mát. Lựa chọn tối ưu hóa phù hợp có thể ảnh hưởng đáng kể đến tốc độ hội tụ và chất lượng của mô hình cuối cùng.

#### 1. Gradient Descent (GD)

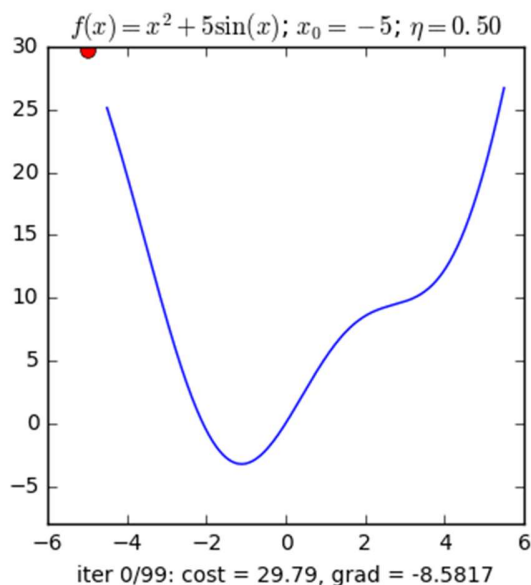
Trong các bài toán tối ưu, chúng ta thường tìm giá trị nhỏ nhất của 1 hàm số nào đó, mà hàm số đạt giá trị nhỏ nhất khi đạo hàm bằng 0. Nhưng đâu phải lúc nào đạo hàm hàm số cũng được, đối với các hàm số nhiều biến thì đạo hàm rất phức tạp, thậm chí là bất khả thi. Nên thay vào đó người ta tìm điểm gần với điểm cực tiểu nhất và xem đó là nghiệm bài toán. Gradient Descent dịch ra tiếng Việt là giảm dần độ dốc, nên hướng tiếp cận ở đây là chọn 1 nghiệm ngẫu nhiên cứ sau mỗi vòng lặp (hay epoch) thì cho nó tiến dần đến điểm cần tìm.

Công thức :  $x_{new} = x_{old} - learningrate.gradient(x)$

Công thức trên được xây dựng để cập nhật lại nghiệm sau mỗi vòng lặp. Dấu '-' trừ ở đây ám chỉ **ngược hướng đạo hàm**.

Ví dụ như đối với hàm  $f(x) = x^2 + 5\sin(x)$  như hình dưới thì  $f'(x) = 2x + 5\cos(x)$  với  $x_{old} = -4$  thì  $f'(-4) < 0 \Rightarrow x_{new} > x_{old}$  nên nghiệm sẽ di chuyển về bên phải tiến gần tới điểm cực tiểu. Ngược lại với  $x_{old} = 4$  thì  $f'(4) > 0 \Rightarrow x_{new} < x_{old}$  nên nghiệm sẽ di chuyển về bên trái tiến gần tới điểm cực tiểu

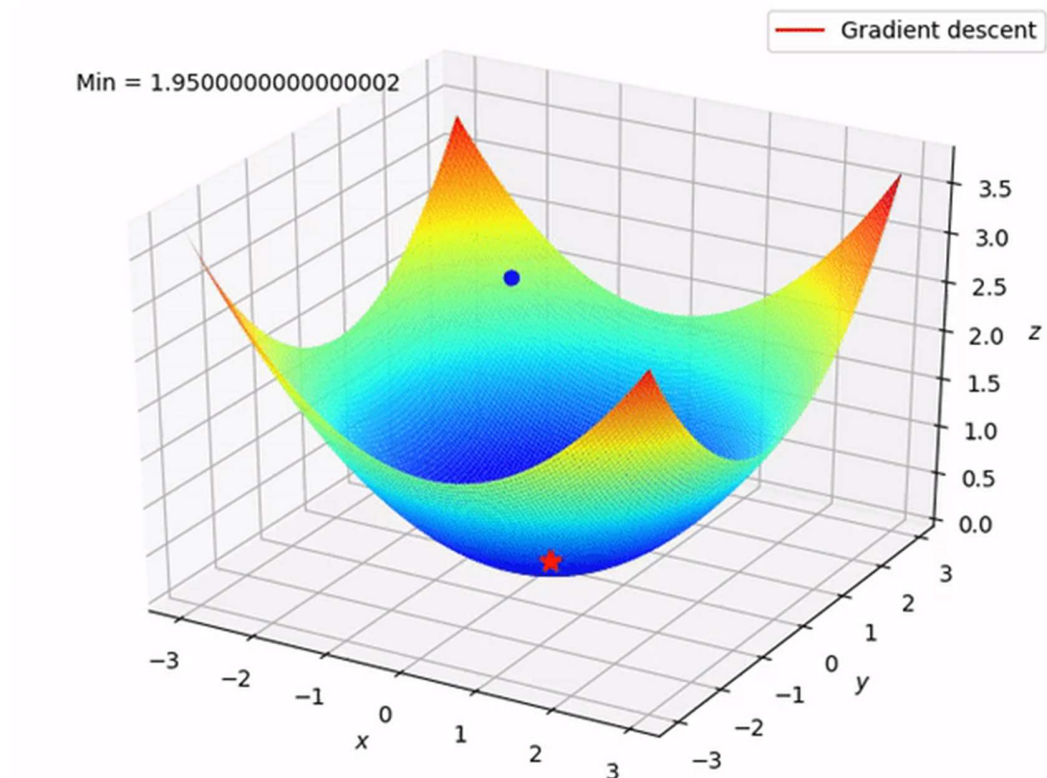
a) Gradient descent cho hàm một biến:



Qua các hình trên ta thấy Gradient descent phụ thuộc vào nhiều yếu tố : như nếu chọn điểm x ban đầu khác nhau sẽ ảnh hưởng đến quá trình hội tụ; hoặc tốc độ học (learning rate) quá lớn hoặc quá

nhỏ cũng ảnh hưởng: nếu tốc độ học quá nhỏ thì tốc độ hội tụ rất chậm ảnh hưởng đến quá trình training, còn tốc độ học quá lớn thì tiến nhanh tới đích sau vài vòng lặp tuy nhiên thuật toán không hội tụ, quanh quẩn quanh đích vì bước nhảy quá lớn.

b) Gradient descent cho hàm nhiều biến:



**Ưu điểm:** Thuật toán gradient descent cơ bản, dễ hiểu. Thuật toán đã giải quyết được vấn đề tối ưu model neural network bằng cách cập nhật trọng số sau mỗi vòng lặp.

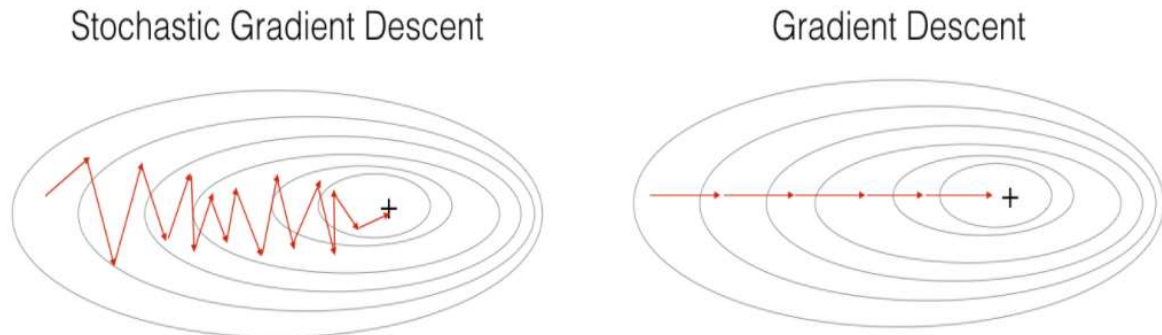
**Nhược điểm:**

- Vì đơn giản nên thuật toán Gradient Descent còn nhiều hạn chế như phụ thuộc vào nghiệm khởi tạo ban đầu và learning rate.
- Ví dụ 1 hàm số có 2 global minimum thì tùy thuộc vào 2 điểm khởi tạo ban đầu sẽ cho ra 2 nghiệm cuối cùng khác nhau.
- Tốc độ học quá lớn sẽ khiến cho thuật toán không hội tụ, quanh quẩn bên đích vì bước nhảy quá lớn; hoặc tốc độ học nhỏ ảnh hưởng đến tốc độ training

## 2. Stochastic Gradient Descent (SGD)

Stochastic là 1 biến thể của Gradient Descent. Thay vì sau mỗi epoch chúng ta sẽ cập nhật trọng số (Weight) 1 lần thì trong mỗi epoch có N điểm dữ liệu chúng ta sẽ cập nhật trọng số N lần. Nhìn vào 1 mặt, SGD sẽ làm giảm đi tốc độ của 1 epoch. Tuy nhiên nhìn theo 1 hướng khác, SGD sẽ hội

tự rất nhanh chỉ sau vài epoch. Công thức SGD cũng tương tự như GD nhưng thực hiện trên từng điểm dữ liệu.



Nhìn vào 2 hình trên, ta thấy SGD có đường đi khá là zig zắc , không mượt như GD. Để hiểu điều đó vì 1 điểm dữ liệu không thể đại diện cho toàn bộ dữ liệu. Đặt câu hỏi tại sao phải dùng SGD thay cho GD mặc dù đường đi của nó khá zig zắc ? Ở đây, GD có hạn chế đối với cơ sở dữ liệu lớn ( vài triệu dữ liệu ) thì việc tính toán đạo hàm trên toàn bộ dữ liệu qua mỗi vòng lặp trở nên cồng kềnh. Bên cạnh đó GD không phù hợp với **online learning**. Online learning là khi dữ liệu cập nhật liên tục (ví dụ như thêm người dùng đăng kí ) thì mỗi lần thêm dữ liệu ta phải tính lại đạo hàm trên toàn bộ dữ liệu => thời gian tính toán lâu, thuật toán không online nữa. Vì thế SGD ra đời để giải quyết vấn đề đó, vì mỗi lần thêm dữ liệu mới vào chỉ cần cập nhật trên 1 điểm dữ liệu đó thôi, phù hợp với online learning.

Một ví dụ minh họa : có 10.000 điểm dữ liệu thì chỉ sau 3 epoch ta đã có được nghiệm tốt, còn với GD ta phải dùng tới 90 epoch để đạt được kết quả đó.

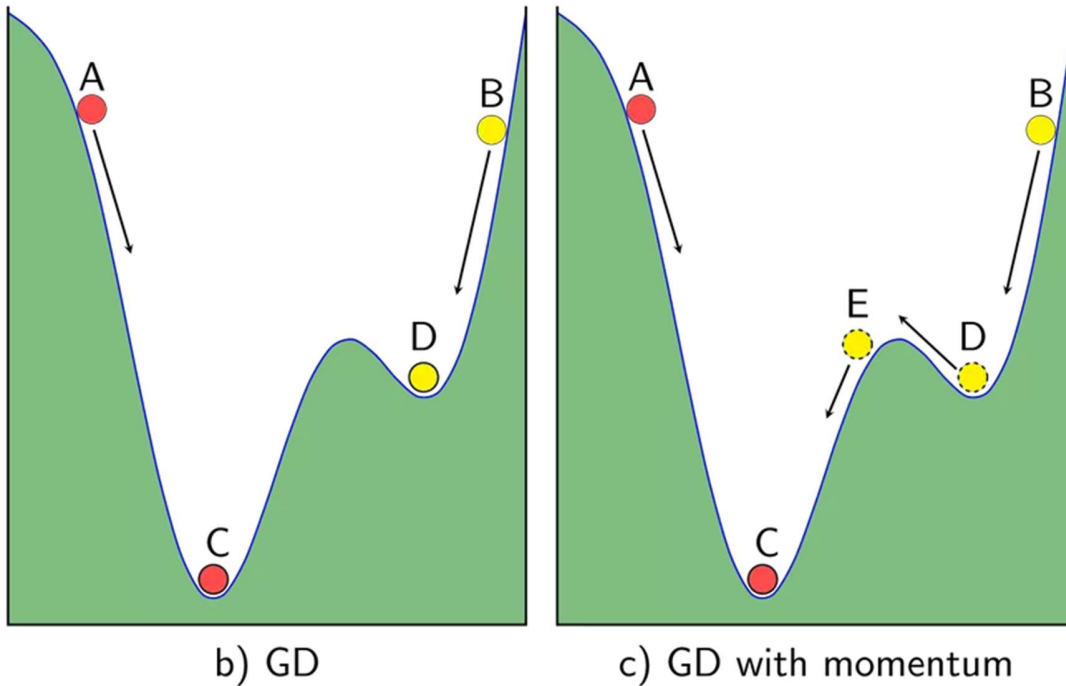
**Ưu điểm:** Thuật toán giải quyết được đối với cơ sở dữ liệu lớn mà GD không làm được. Thuật toán tối ưu này hiện nay vẫn hay được sử dụng.

**Nhược điểm:** Thuật toán vẫn chưa giải quyết được 2 nhược điểm lớn của gradient descent (learning rate, điểm dữ liệu ban đầu). Vì vậy ta phải kết hợp SGD với 1 số thuật toán khác như: Momentum, AdaGrad,..Các thuật toán này sẽ được trình bày ở phần sau.

### 3. Momentum

Để khắc phục các hạn chế trên của thuật toán Gradient Descent người ta dùng gradient descent with momentum. Vậy gradient with momentum là gì ?





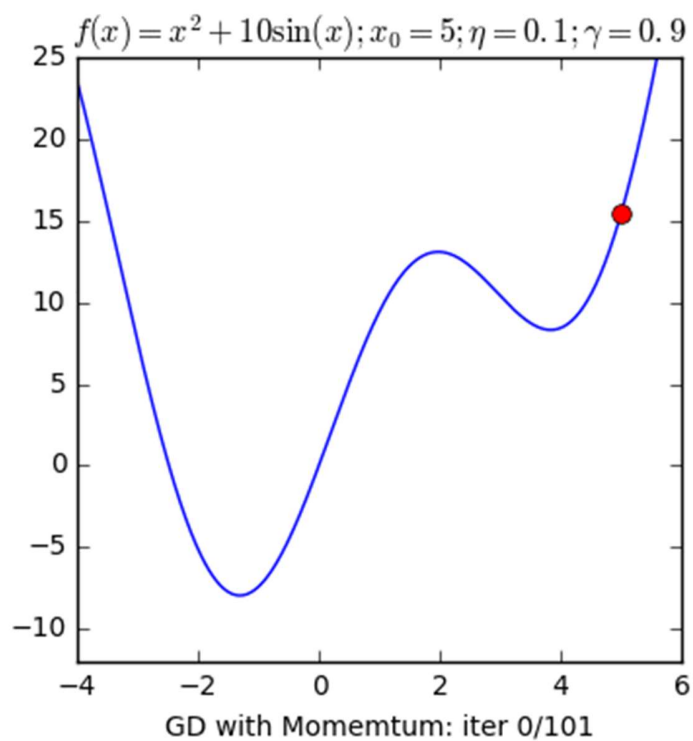
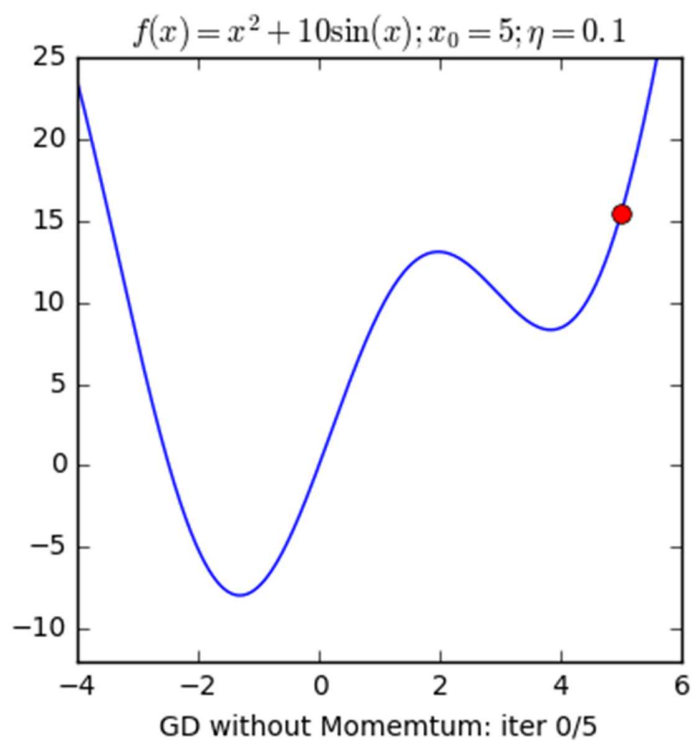
Để giải thích được Gradient with Momentum thì trước tiên ta nên nhìn dưới góc độ vật lý: Như hình b phía trên, nếu ta thả 2 viên bi tại 2 điểm khác nhau A và B thì viên bi A sẽ trượt xuống điểm C còn viên bi B sẽ trượt xuống điểm D, nhưng ta lại không mong muốn viên bi B sẽ dừng ở điểm D (local minimum) mà sẽ tiếp tục lăn tới điểm C (global minimum). Để thực hiện được điều đó ta phải cấp cho viên bi B 1 vận tốc ban đầu đủ lớn để nó có thể vượt qua điểm E tới điểm C. Dựa vào ý tưởng này người ta xây dựng nên thuật toán Momentum (tức là theo đà tiến tới).

Nhìn dưới góc độ toán học, ta có công thức Momentum:

$$x_{new} = x_{old} - (\text{gama} \cdot v + \text{learningrate} \cdot \text{gradient})$$

Trong đó:

- $x_{new}$ : tọa độ mới
- $x_{old}$ : tọa độ cũ
- $\text{gama}$ : parameter, thường  $=0.9$
- $\text{learningrate}$ : tốc độ học
- $\text{gradient}$ : đạo hàm của hàm  $f$



Qua 2 ví dụ minh họa trên của hàm  $f(x) = x^2 + 10\sin(x)$ , ta thấy GD without momentum sẽ hội

tụ sau 5 vòng lặp nhưng không phải là global minimum. Nhưng GD with momentum dù mất nhiều vòng lặp nhưng nghiệm tiến tới global minimum, qua hình ta thấy nó sẽ vượt tốc tiến tới điểm global minimum và dao động qua lại quanh điểm đó trước khi dừng lại.

**Ưu điểm:** Thuật toán tối ưu giải quyết được vấn đề: Gradient Descent không tiến được tới điểm global minimum mà chỉ dừng lại ở local minimum.

**Nhược điểm:** Tuy momentum giúp hòn bi vượt dốc tiến tới điểm đích, tuy nhiên khi tới gần đích, nó vẫn mất khá nhiều thời gian giao động qua lại trước khi dừng hẳn, điều này được giải thích vì viên bi có đà.

#### 4. Adagrad

Không giống như các thuật toán trước đó thì learning rate hầu như giống nhau trong quá trình training (learning rate là hằng số), Adagrad coi learning rate là 1 tham số. Tức là Adagrad sẽ cho learning rate biến thiên sau mỗi thời điểm  $t$ .

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$

**Trong đó:**

- $\eta$  : hằng số
- $g_t$  : gradient tại thời điểm  $t$
- $\epsilon$  : hệ số tránh lỗi ( chia cho mẫu bằng 0
- $G$  : là ma trận chéo mà mỗi phần tử trên đường chéo  $(i,i)$  là bình phương của đạo hàm vector tham
- số tại thời điểm  $t$ .

**Ưu điểm:** Một lợi ích dễ thấy của Adagrad là tránh việc điều chỉnh learning rate bằng tay, chỉ cần để tốc độ học default là 0.01 thì thuật toán sẽ tự động điều chỉnh.

**Nhược điểm:** Yếu điểm của Adagrad là tổng bình phương biến thiên sẽ lớn dần theo thời gian cho đến khi nó làm tốc độ học cực kì nhỏ, làm việc training trở nên đóng băng.

#### 5. RMSprop

RMSprop giải quyết vấn đề tỷ lệ học giảm dần của Adagrad bằng cách chia tỷ lệ học cho trung bình của bình phương gradient.

$$E[g^2]_t = 0,9E[g^2]_{t-1} + 0,1g_t^2$$

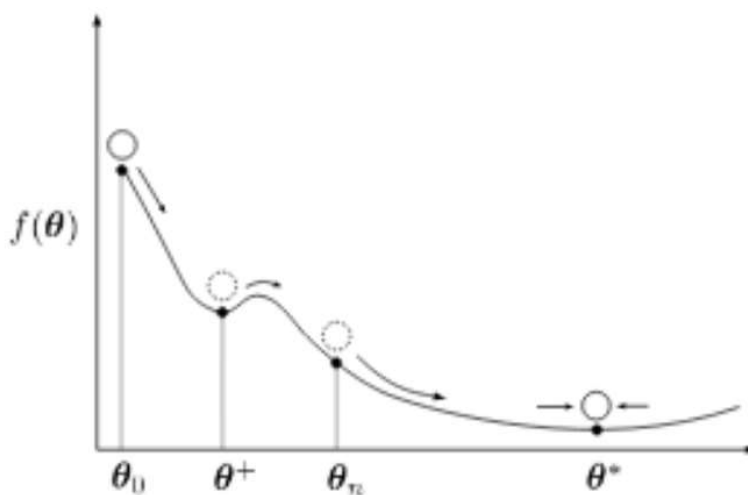
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

**Ưu điểm:** Ưu điểm rõ nhất của RMSprop là giải quyết được vấn đề tốc độ học giảm dần của Adagrad (vấn đề tốc độ học giảm dần theo thời gian sẽ khiến việc training chậm dần, có thể dẫn tới bị đóng băng).

**Nhược điểm:** Thuật toán RMSprop có thể cho kết quả nghiệm chỉ là local minimum chứ không đạt được global minimum như Momentum. Vì vậy người ta sẽ kết hợp cả 2 thuật toán Momentum với RMSprop cho ra 1 thuật toán tối ưu Adam. Chúng ta sẽ trình bày nó trong phần sau.

## 6. Adam

Như đã nói ở trên Adam là sự kết hợp của Momentum và RMSprop. Nếu giải thích theo hiện tượng vật lý thì Momentum giống như 1 quả cầu lao xuống dốc, còn Adam như 1 quả cầu rất nặng có ma sát, vì vậy nó dễ dàng vượt qua local minimum tới global minimum và khi tới global minimum nó không mất nhiều thời gian dao động qua lại quanh đích vì nó có ma sát nên dễ dừng lại hơn.



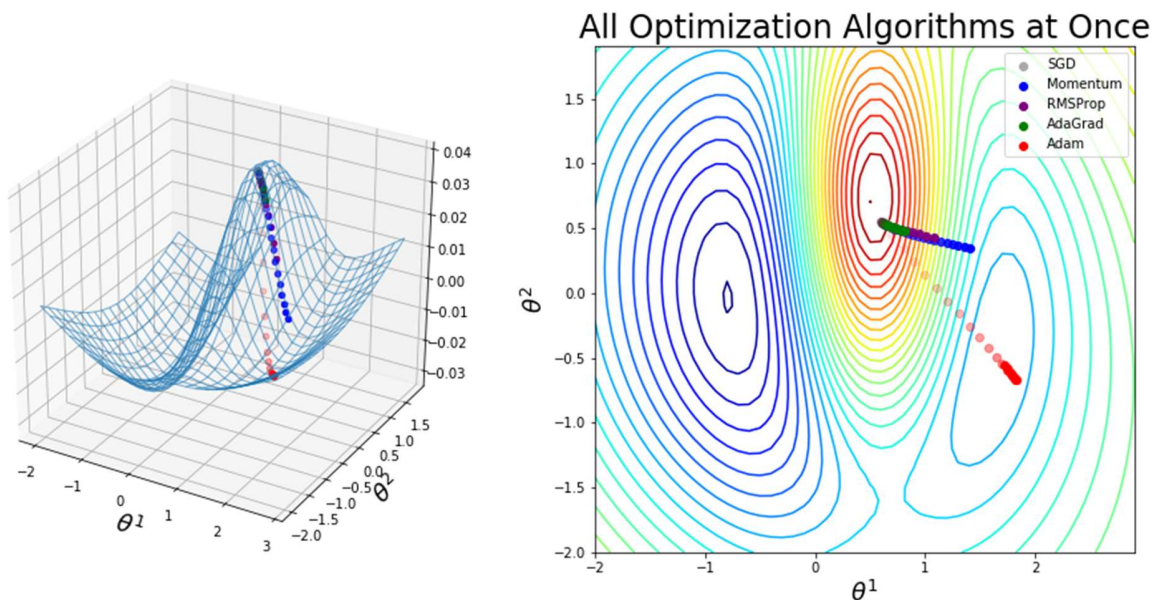
**Figure 2:** Heavy Ball with Friction, where the ball with mass overshoots the local minimum  $\theta^+$  and settles at the flat minimum  $\theta^*$ .

Công thức:

$$\begin{aligned}
\mathbf{g}_n &\leftarrow \nabla f(\boldsymbol{\theta}_{n-1}) \\
\mathbf{m}_n &\leftarrow (\beta_1/(1 - \beta_1^n)) \mathbf{m}_{n-1} + ((1 - \beta_1)/(1 - \beta_1^n)) \mathbf{g}_n \\
\mathbf{v}_n &\leftarrow (\beta_2/(1 - \beta_2^n)) \mathbf{v}_{n-1} + ((1 - \beta_2)/(1 - \beta_2^n)) \mathbf{g}_n \odot \mathbf{g}_n \\
\boldsymbol{\theta}_n &\leftarrow \boldsymbol{\theta}_{n-1} - a \mathbf{m}_n / (\sqrt{\mathbf{v}_n} + \epsilon),
\end{aligned}$$

## Tổng quan

Còn có rất nhiều thuật toán tối ưu như Nesterov (NAG), Adadelta, Nadam,... nhưng mình sẽ không trình bày trong bài này, mình chỉ tập trung vào các optimizers hay được sử dụng. Hiện nay optimizers hay được sử dụng nhất là 'Adam'.



Qua hình trên ta thấy optimizer Adam hoạt động khá tốt, tiến nhanh tới mức tối thiểu hơn các phương pháp khác.

**AdamW:** Biến thể của Adam với cơ chế weight decay tốt hơn, được sử dụng rộng rãi trong các mô hình mới.

### 2.5.3. Kỹ thuật chính quy hóa (Regularization Techniques)

Chính quy hóa giúp tránh tình trạng overfitting, cải thiện khả năng tổng quát hóa của mô hình:

- **L1 Regularization (Lasso):** Thêm giá trị tuyệt đối của trọng số vào hàm mất mát, thúc đẩy tính thưa thớt bằng cách đưa nhiều trọng số về 0.

- **L2 Regularization (Ridge):** Thêm bình phương của trọng số vào hàm mất mát, ngăn trọng số trở nên quá lớn.
- **Dropout:** Tắt ngẫu nhiên một tỷ lệ nơ-ron trong quá trình huấn luyện, buộc mạng phải học các đặc trưng mạnh hơn.
- **Batch Normalization:** Chuẩn hóa dữ liệu đầu vào của mỗi lớp, giúp ổn định và tăng tốc quá trình học.
- **Early Stopping:** Dừng huấn luyện khi hiệu suất trên tập validation không cải thiện sau một số epoch.
- **Data Augmentation:** Tạo dữ liệu huấn luyện mới bằng cách biến đổi dữ liệu hiện có (xoay, lật, thay đổi độ sáng cho ảnh).
- **Weight Decay:** Thường được triển khai như L2 regularization, giảm dần giá trị của trọng số.
- **Label Smoothing:** Làm mềm nhãn one-hot để tránh mô hình quá tự tin.

#### 2.5.4. Chuẩn hóa dữ liệu (Data Normalization)

Chuẩn hóa dữ liệu giúp mô hình hội tụ nhanh hơn và ổn định hơn:

- **Min-Max Scaling:** Chuyển đổi dữ liệu vào khoảng  $[0,1]$  hoặc  $[-1,1]$ .  

$$x_{\text{normalized}} = (x - \min(x)) / (\max(x) - \min(x))$$
- **Z-score Normalization (Standardization):** Biến đổi dữ liệu thành phân phối có trung bình 0 và độ lệch chuẩn 1.  

$$x_{\text{standardized}} = (x - \text{mean}(x)) / \text{std}(x)$$
- **Robust Scaling:** Sử dụng median và IQR thay vì mean và std để giảm ảnh hưởng của outlier.  

$$x_{\text{robust}} = (x - \text{median}(x)) / \text{IQR}(x)$$
- **Log Transformation:** Áp dụng cho dữ liệu có phân phối lệch phải.  

$$x_{\text{log}} = \log(x + c)$$
- **Power Transformation:** Chuyển đổi dữ liệu về dạng gần phân phối chuẩn hơn.  

$$x_{\text{power}} = x^{\lambda} \text{ hoặc Box-Cox transformation}$$

## 2.6 Ứng dụng của học sâu

### 2.6.1. Xử lý ảnh và thị giác máy tính (Image Processing and Computer Vision)

- **Phân loại ảnh (Image Classification):** Xác định đối tượng chính trong ảnh. Mô hình tiêu biểu: ResNet, EfficientNet, Vision Transformer.
- **Phát hiện đối tượng (Object Detection):** Xác định vị trí và phân loại nhiều đối tượng trong ảnh. Mô hình: YOLO, Faster R-CNN, SSD, DETR.
- **Phân đoạn ảnh (Image Segmentation):** Phân loại từng pixel trong ảnh. Mô hình: U-Net, Mask R-CNN, DeepLab.
- **Tạo sinh ảnh (Image Generation):** Tạo ảnh mới từ dữ liệu huấn luyện. Mô hình: GAN, Diffusion Models (DALL-E, Stable Diffusion, Midjourney).
- **Siêu phân giải (Super Resolution):** Cải thiện độ phân giải ảnh. Mô hình: SRGAN, ESRGAN.
- **Nhận dạng khuôn mặt (Face Recognition):** Xác định danh tính từ ảnh khuôn mặt. Kỹ thuật: FaceNet, ArcFace.
- **Ứng dụng thực tế:** Xe tự hành, hệ thống giám sát, chẩn đoán y tế, thực tế ảo/tăng cường.

### 2.6.2. Xử lý ngôn ngữ tự nhiên (Natural Language Processing)

- **Mô hình ngôn ngữ (Language Models):** GPT, LLaMA, Claude, PaLM dùng kiến trúc Transformer để tạo và hiểu văn bản.
- **Phân loại văn bản (Text Classification):** Phân tích cảm xúc, phân loại thể loại văn bản, phát hiện spam.
- **Dịch máy (Machine Translation):** Dịch giữa các ngôn ngữ. Mô hình: Transformer-based NMT.
- **Trả lời câu hỏi (Question Answering):** Trả lời câu hỏi từ ngữ cảnh cho trước. Hệ thống: BERT, RoBERTa, T5.
- **Tóm tắt văn bản (Text Summarization):** Tạo bản tóm tắt ngắn gọn từ văn bản dài. Kỹ thuật: Extractive và Abstractive summarization.
- **Tạo sinh văn bản (Text Generation):** Tạo văn bản mới dựa trên prompt. Mô hình: GPT, BLOOM, LLaMA.
- **Xử lý giọng nói:** Nhận dạng giọng nói (ASR), chuyển văn bản thành giọng nói (TTS). Mô hình: Whisper, WaveNet.
- **Ứng dụng thực tế:** Trợ lý ảo, chatbot, phân tích dữ liệu văn bản, dịch tự động, hệ thống CRM.

### 2.6.3. Hệ thống đề xuất (Recommendation Systems)

- **Lọc cộng tác (Collaborative Filtering):** Đề xuất dựa trên hành vi của người dùng tương tự.
  - User-based: Tìm người dùng tương tự để đề xuất.
  - Item-based: Tìm sản phẩm tương tự với sản phẩm người dùng đã thích.
- **Lọc dựa trên nội dung (Content-based Filtering):** Đề xuất dựa trên đặc điểm của sản phẩm tương tự với sản phẩm người dùng đã thích.
- **Hệ thống kết hợp (Hybrid Systems):** Kết hợp lọc cộng tác và lọc dựa trên nội dung.
- **Hệ thống dựa trên học sâu:**
  - Neural Collaborative Filtering
  - Deep Matrix Factorization
  - Wide & Deep Learning
  - Neural Graph Collaborative Filtering
  - Sequential Recommendation với RNN, LSTM, Transformer
- **Ứng dụng thực tế:** Đề xuất sản phẩm trên thương mại điện tử, nội dung trên dịch vụ phát trực tuyến, bài viết trên mạng xã hội, quảng cáo cá nhân hóa.

### 2.6.4. Y học và chăm sóc sức khỏe (Medicine and Healthcare)

- **Chẩn đoán hình ảnh y tế:** Phát hiện bệnh từ X-quang, CT, MRI, siêu âm.
  - Phát hiện ung thư (vú, phổi, da, não)
  - Phân tích X-quang xương và khớp
  - Phát hiện bệnh võng mạc
- **Dự đoán bệnh và chẩn đoán:** Dự đoán bệnh từ dữ liệu bệnh án, xét nghiệm, triệu chứng.
  - Dự đoán bệnh tim mạch
  - Dự đoán tiểu đường
  - Chẩn đoán bệnh hiếm
- **Phân tích dữ liệu gen:** Dự đoán cấu trúc protein, phát hiện đột biến gen.
  - AlphaFold đột phá trong dự đoán cấu trúc protein
  - Phân tích chuỗi DNA để phát hiện bệnh di truyền



- **Phát triển thuốc:** Tăng tốc quy trình phát triển thuốc mới.
  - Dự đoán hoạt tính dược lý
  - Thiết kế phân tử mới
  - Tối ưu hóa hợp chất dẫn đầu
- **Y học cá nhân hóa:** Điều chỉnh phương pháp điều trị dựa trên đặc điểm cá nhân.
  - Liều lượng thuốc cá nhân hóa
  - Dự đoán phản ứng với thuốc
- **Ứng dụng thực tế:** Hỗ trợ bác sĩ trong chẩn đoán, giám sát từ xa, dự báo dịch bệnh, tối ưu hóa quy trình bệnh viện.

#### 2.6.5. Tự động hóa và robotics (Automation and Robotics)

- **Xe tự hành (Autonomous Vehicles):**
  - Nhận dạng đối tượng và người đi bộ
  - Phát hiện làn đường và biển báo
  - Lập kế hoạch đường đi
  - Dự đoán chuyển động của phương tiện khác
- **Robot công nghiệp:**
  - Nhặt và đặt đối tượng phức tạp
  - Kiểm tra chất lượng tự động
  - Lắp ráp chính xác
  - Điều khiển robot dựa trên thị giác
- **Robot dịch vụ:**
  - Tương tác với con người bằng ngôn ngữ tự nhiên
  - Nhận dạng cử chỉ và biểu cảm
  - Điều hướng trong môi trường phức tạp
  - Hỗ trợ người già và người khuyết tật
- **Drone thông minh:**
  - Bay tự động trong môi trường phức tạp

- Tránh vật cản động
- Theo dõi đối tượng
- Khảo sát và lập bản đồ
- **Tự động hóa trong nông nghiệp:**
  - Phát hiện bệnh cây trồng
  - Thu hoạch tự động
  - Phun thuốc chính xác
  - Giám sát sức khỏe cây trồng và vật nuôi
- **Ứng dụng thực tế:** Nhà máy thông minh, kho hàng tự động, chăm sóc sức khỏe, giám sát an ninh, thăm dò và cứu hộ.

## 2.7. Thách thức và hạn chế

### 2.7.1. Vấn đề học quá mức và thiếu mức (Overfitting and Underfitting)

- **Học quá mức (Overfitting):** Mô hình học “thuộc lòng” dữ liệu huấn luyện, hoạt động tốt trên tập huấn luyện nhưng kém trên dữ liệu mới.
  - Nguyên nhân: Mô hình quá phức tạp so với dữ liệu, thiếu dữ liệu huấn luyện.
  - Giải pháp: Chính quy hóa (L1, L2, Dropout), Data Augmentation, Early Stopping, K-fold cross-validation.
- **Học thiếu mức (Underfitting):** Mô hình quá đơn giản, không nắm bắt được mối quan hệ trong dữ liệu.
  - Nguyên nhân: Mô hình quá đơn giản, hyperparameter không phù hợp, thiếu đặc trưng.
  - Giải pháp: Tăng độ phức tạp mô hình, kỹ thuật Feature Engineering, điều chỉnh hyperparameter.
- **Cân bằng giữa Bias và Variance:** Tìm kiếm điểm cân bằng giữa mô hình quá đơn giản (bias cao) và quá phức tạp (variance cao).
  - Bias-Variance Tradeoff
  - Bias: Lỗi do giả định đơn giản hóa quá mức
  - Variance: Lỗi do mô hình nhạy cảm với biến động nhỏ trong dữ liệu huấn luyện

### 2.7.2. Nhu cầu dữ liệu lớn (Large Data Requirements)

- **Đòi hỏi lượng dữ liệu huấn luyện lớn:** Mô hình học sâu thường cần hàng triệu mẫu dữ liệu để đạt hiệu suất tốt.
- **Thách thức thu thập dữ liệu:**
  - o Chi phí và thời gian thu thập
  - o Dữ liệu nhạy cảm (y tế, tài chính)
  - o Dữ liệu hiếm gặp (bệnh hiếm, sự kiện bất thường)
- **Giải pháp cho vấn đề dữ liệu hạn chế:**
  - o Transfer Learning: Sử dụng mô hình pre-trained trên tập dữ liệu lớn, sau đó fine-tune trên tập dữ liệu nhỏ.
  - o Data Augmentation: Tạo dữ liệu mới từ dữ liệu hiện có.
  - o Self-supervised Learning: Học từ dữ liệu không nhãn.
  - o Few-shot Learning: Học từ số lượng mẫu ít.
  - o Data Synthesis: Tạo dữ liệu tổng hợp (GAN, Diffusion Models).
- **Vấn đề chất lượng dữ liệu:**
  - o Dữ liệu nhiễu hoặc sai nhãn
  - o Dữ liệu thiên lệch
  - o Thiếu đa dạng trong dữ liệu

### 2.7.3. Chi phí tính toán cao (High Computational Costs)

- **Yêu cầu phần cứng:**
  - o GPU/TPU mạnh cho huấn luyện và triển khai
  - o Bộ nhớ lớn cho mô hình và dữ liệu
  - o Hệ thống lưu trữ nhanh
- **Chi phí huấn luyện:**
  - o Thời gian huấn luyện từ nhiều giờ đến nhiều tháng
  - o Chi phí điện năng cao
  - o Chi phí thuê đám mây (cloud) đắt đỏ
  - o Ví dụ: Huấn luyện GPT-3 tốn khoảng 4,6 triệu USD

- **Chi phí suy luận và triển khai:**
  - Chi phí vận hành liên tục
  - Yêu cầu phần cứng chuyên dụng
  - Độ trễ trong ứng dụng thời gian thực
- **Giải pháp giảm chi phí tính toán:**
  - Distillation: Chuyển kiến thức từ mô hình lớn sang mô hình nhỏ
  - Quantization: Giảm độ chính xác của trọng số (16-bit, 8-bit, 4-bit)
  - Pruning: Loại bỏ các kết nối không cần thiết
  - Efficient architectures: MobileNet, EfficientNet
  - Specialized hardware: ASIC, FPGA
  - Model parallelism và pipeline parallelism

#### 2.7.4. Tính giải thích được (Explainability)

- **“Hộp đen” trong học sâu:** Mô hình phức tạp với hàng triệu tham số, khó hiểu quá trình ra quyết định.
- **Hạn chế trong ứng dụng nghiêm ngặt:**
  - Y tế: Bác sĩ cần hiểu lý do chẩn đoán
  - Tài chính: Quyết định tín dụng phải có cơ sở
  - Pháp lý: Quyết định tòa án cần minh bạch
  - Quốc phòng: Hệ thống vũ khí tự động cần giải trình
- **Kỹ thuật XAI (Explainable AI):**
  - Feature Visualization: Hiển thị đặc trưng được mô hình kích hoạt
  - Attention Maps: Hiển thị vùng mô hình tập trung
  - LIME (Local Interpretable Model-agnostic Explanations)
  - SHAP (SHapley Additive exPlanations)
  - Gradient-based attribution methods
  - Concept-based explanations
- **Đánh đổi hiệu suất và khả năng giải thích:**

- Mô hình đơn giản hơn (decision tree, linear models) dễ giải thích nhưng hiệu suất thấp hơn
- Mô hình phức tạp hiệu suất cao nhưng khó giải thích

#### 2.7.5. Vấn đề đạo đức và công bằng (Ethics and Fairness)

##### - **Thiên lệch và phân biệt đối xử:**

- Mô hình phản ánh thiên lệch trong dữ liệu huấn luyện
- Ví dụ: Phân biệt giới tính trong tuyển dụng, phân biệt chủng tộc trong hệ thống tư pháp

##### - **Quyền riêng tư và bảo mật dữ liệu:**

- Thu thập và sử dụng dữ liệu cá nhân
- Nguy cơ rò rỉ thông tin nhạy cảm
- Mô hình đối kháng (Adversarial attacks)

##### - **Tác động xã hội:**

- Mất việc làm do tự động hóa
- Phân phối không đều lợi ích của AI
- Digital divide (khoảng cách số)

##### - **Thao túng và tin giả (deepfakes):**

- Tạo nội dung giả mạo có chất lượng cao
- Lan truyền thông tin sai lệch
- Ảnh hưởng đến bầu cử và ý kiến công chúng

##### - **Quản trị và quy định:**

- Phát triển quy định pháp lý cho AI
- Tiêu chuẩn đạo đức trong phát triển AI
- Trách nhiệm giải trình

##### - **Giải pháp:**

- Fairness-aware machine learning
- Federated learning bảo vệ quyền riêng tư
- Dữ liệu huấn luyện đa dạng và đại diện

- Kiểm toán thuật toán
- Tham vấn các bên liên quan

## **CHƯƠNG 3: ỨNG DỤNG HỌC SÂU CHO PHÂN LOẠI HÌNH ẢNH SỬ DỤNG CNN VÀ SO SÁNH HIỆU SUẤT TRÊN 3 BỘ DỮ LIỆU KHÁC NHAU**

### **3.1. Mô tả dự án**

Dự án này nghiên cứu và triển khai các mô hình học sâu cho bài toán phân loại hình ảnh sử dụng mạng neural tích chập (CNN) trên 3 bộ dữ liệu tiêu chuẩn với độ phức tạp tăng dần: Fashion MNIST, CIFAR-10 và Food-101. Kết quả cho thấy sự tương quan giữa độ phức tạp của dữ liệu, kiến trúc mạng neural, và hiệu suất phân loại. Dự án cũng bao gồm một ứng dụng web trực quan hóa kết quả và cho phép người dùng thử nghiệm các mô hình đã được huấn luyện.

#### **Mục tiêu của dự án**

- Nghiên cứu hiệu suất của các kiến trúc CNN trên ba bộ dữ liệu có độ phức tạp khác nhau
- Đánh giá tác động của kiến trúc mạng và kỹ thuật tiền xử lý đến độ chính xác phân loại
- Phân tích mối quan hệ giữa đặc điểm bộ dữ liệu, số lượng tham số của mô hình và hiệu suất phân loại
- Xây dựng ứng dụng web để trực quan hóa kết quả và cho phép người dùng thử nghiệm các mô hình

### **3.2. Ba bộ dữ liệu**

#### **Fashion MNIST**

- Mô tả: Bộ dữ liệu phân loại quần áo với 10 loại khác nhau
- Kích thước ảnh: 28x28 pixel
- Số kênh màu: 1 (đơn sắc)
- Số lớp: 10
- Số lượng: 60,000 ảnh huấn luyện, 10,000 ảnh kiểm tra
- Độ phức tạp: Thấp

#### **CIFAR-10**

- Mô tả: Bộ dữ liệu phân loại đối tượng với 10 loại đối tượng khác nhau
- Kích thước ảnh: 32x32 pixel
- Số kênh màu: 3 (RGB)

- Số lớp: 10
- Số lượng: 50,000 ảnh huấn luyện, 10,000 ảnh kiểm tra
- Độ phức tạp: Trung bình

### Food-101

- **Mô tả:** Bộ dữ liệu phân loại thực phẩm với 101 loại món ăn khác nhau
- **Kích thước ảnh:** 128x128 pixel (đã được resize từ gốc)
- **Số kênh màu:** 3 (RGB)
- **Số lớp:** 101 lớp
- **Số lượng:** 101,000 ảnh (750 ảnh huấn luyện và 250 ảnh kiểm tra cho mỗi lớp)
- **Độ phức tạp:** Cao

### 3.3. Huấn luyện và đánh giá

3.3.1. Huấn luyện và đánh giá một mạng nơ-ron tích chập (CNN) để phân loại ảnh từ tập dữ liệu CIFAR-10

#### Phương pháp

##### 1. Tải và khám phá dữ liệu

Tập dữ liệu CIFAR-10 được tải bằng hàm `tf.keras.datasets.cifar10.load_data()` từ TensorFlow, trả về các tập huấn luyện và kiểm tra. Tập huấn luyện gồm 50.000 ảnh, mỗi ảnh có kích thước (32, 32, 3), biểu thị chiều cao, chiều rộng và kênh màu (RGB). Tập kiểm tra gồm 10.000 ảnh với cùng kích thước. Nhận được cung cấp dưới dạng số nguyên tương ứng với 10 lớp.

```
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
```

Downloading data from <https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz>  
170498071/170498071 ————— 3s 0us/step

```
[ ] print("x_train shape", x_train.shape)
    print("y_train shape", y_train.shape)
    print("x_test shape", x_test.shape)
    print("y_test shape", y_test.shape)
```

```
x_train shape (50000, 32, 32, 3)
y_train shape (50000, 1)
x_test shape (10000, 32, 32, 3)
y_test shape (10000, 1)
```

## 2. Tiền xử lý dữ liệu

Mặc dù các bước tiền xử lý cụ thể không hiển thị đầy đủ trong mã bị cắt ngắn, nhưng thông thường, các tác vụ phân loại ảnh sẽ chuẩn hóa giá trị pixel. Với CIFAR-10, giá trị pixel dao động từ 0 đến 255, nên chúng thường được chia cho 255 để đưa về khoảng [0, 1]. Quá trình chuẩn hóa này giúp cải thiện sự hội tụ của mạng nơ-ron trong quá trình huấn luyện.

```
x_train, x_test = x_train / 255.0, x_test / 255.0
```

## 3. Kiến trúc mô hình

```
✓ 0s ▶ model = models.Sequential([
    data_augmentation,
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(10, activation='softmax')
])
```

Mã có khả năng định nghĩa một mạng nơ-ron tích chập (CNN) sử dụng API Keras của TensorFlow. CNN là lựa chọn tiêu chuẩn cho phân loại ảnh nhờ khả năng nắm bắt các đặc trưng không gian trong ảnh. Kiến trúc mô hình thường bao gồm:

- **Lớp tích chập:** Trích xuất các đặc trưng từ ảnh.
- **Lớp gộp (pooling):** Giảm kích thước không gian và tải tính toán.
- **Lớp dropout:** Ngăn ngừa hiện tượng quá khớp bằng cách ngắt ngẫu nhiên các nơ-ron trong quá trình huấn luyện.
- **Lớp dày đặc (dense):** Thực hiện phân loại, với lớp cuối cùng sử dụng hàm kích hoạt softmax để xuất xác suất cho 10 lớp.
- **Số lượng tham số:** 237,322
- **Thời gian huấn luyện:** 15 phút

## 4. Biên dịch và huấn luyện mô hình

Sau khi định nghĩa, mô hình được biên dịch với bộ tối ưu hóa, hàm mất mát và các chỉ số đánh giá. Với bài toán phân loại đa lớp như CIFAR-10, các lựa chọn phổ biến bao gồm:



- **Bộ tối ưu hóa:** Adam, tự động điều chỉnh tốc độ học.
- **Hàm mất mát:** Categorical cross-entropy, phù hợp với phân loại đa lớp.
- **Chỉ số:** Độ chính xác (accuracy), đo tỷ lệ ảnh được phân loại đúng.

Mô hình sau đó được huấn luyện trên tập dữ liệu huấn luyện, với một phần được giữ lại làm tập xác thực để theo dõi hiệu suất và phát hiện quá khớp.

#### ✓ Compile the model

```
[8] model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])
```

#### ✓ Train the model

```
history = model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

```
Epoch 1/10
1563/1563 ————— 19s 7ms/step - accuracy: 0.2731 - loss: 1.9389 - val_accuracy: 0.4530 - val_loss: 1.4861
Epoch 2/10
1563/1563 ————— 15s 7ms/step - accuracy: 0.4604 - loss: 1.4977 - val_accuracy: 0.5311 - val_loss: 1.3145
Epoch 3/10
1563/1563 ————— 11s 7ms/step - accuracy: 0.5079 - loss: 1.3783 - val_accuracy: 0.5871 - val_loss: 1.1451
Epoch 4/10
1563/1563 ————— 20s 7ms/step - accuracy: 0.5420 - loss: 1.3038 - val_accuracy: 0.6030 - val_loss: 1.1229
Epoch 5/10
1563/1563 ————— 21s 7ms/step - accuracy: 0.5655 - loss: 1.2352 - val_accuracy: 0.6107 - val_loss: 1.1013
Epoch 6/10
1563/1563 ————— 20s 7ms/step - accuracy: 0.5777 - loss: 1.1993 - val_accuracy: 0.6183 - val_loss: 1.0799
Epoch 7/10
1563/1563 ————— 11s 7ms/step - accuracy: 0.5909 - loss: 1.1699 - val_accuracy: 0.6266 - val_loss: 1.0524
Epoch 8/10
1563/1563 ————— 11s 7ms/step - accuracy: 0.6057 - loss: 1.1414 - val_accuracy: 0.6529 - val_loss: 0.9868
Epoch 9/10
1563/1563 ————— 21s 7ms/step - accuracy: 0.6098 - loss: 1.1251 - val_accuracy: 0.6346 - val_loss: 1.0587
Epoch 10/10
1563/1563 ————— 21s 7ms/step - accuracy: 0.6188 - loss: 1.0968 - val_accuracy: 0.6657 - val_loss: 0.9723
```

## 5. Đánh giá mô hình

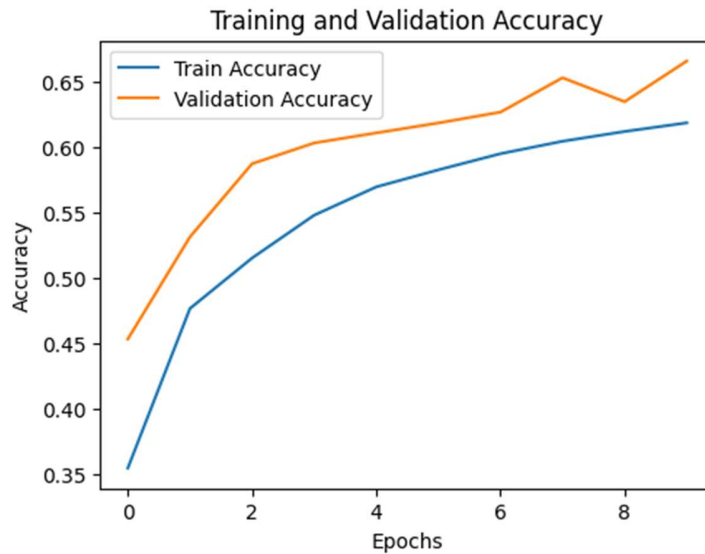
Sau khi huấn luyện, hiệu suất của mô hình được đánh giá trên tập kiểm tra, bao gồm tính toán độ chính xác trên tập kiểm tra. Ngoài ra, mã có thể tạo ma trận nhầm lẫn và báo cáo phân loại để cung cấp thông tin chi tiết về hiệu suất trên từng lớp.

```
test_loss, test_acc = model.evaluate(x_test, y_test)
print(f'Test accuracy: {test_acc:.4f}')
```

```
313/313 ————— 1s 3ms/step - accuracy: 0.6716 - loss: 0.9605
Test accuracy: 0.6657
```

## 6. Trực quan hóa kết quả

```
Text(0.5, 1.0, 'Training and Validation Accuracy')
```



### 3.3.2. Huấn luyện các mô hình học sâu để phân loại chữ số viết tay từ tập dữ liệu FOOD-101.

Bộ dữ liệu Food-101 là một tập hợp lớn các hình ảnh về các món ăn, bao gồm 101 danh mục khác nhau, mỗi danh mục chứa 1.000 hình ảnh. Đây là một bộ dữ liệu phổ biến được sử dụng trong các nhiệm vụ phân loại hình ảnh thuộc lĩnh vực học máy và thị giác máy tính. Food-101 mang tính thách thức cao do sự đa dạng về loại thực phẩm, phong cách chế biến, góc chụp, và chất lượng hình ảnh. Mục tiêu khi huấn luyện mô hình trên bộ dữ liệu này là xây dựng một hệ thống có khả năng nhận diện và phân loại chính xác các món ăn, từ đó ứng dụng vào các lĩnh vực như công nghệ thực phẩm, dịch vụ ăn uống, hoặc hỗ trợ dinh dưỡng.

Bộ dữ liệu này thường được tải thông qua các thư viện như TensorFlow Datasets (TFDS), giúp đơn giản hóa quá trình truy cập và xử lý. Việc hiểu rõ đặc điểm của Food-101 là bước đầu tiên quan trọng để thiết kế một quy trình huấn luyện hiệu quả.

## Phương pháp

### 1. Tải và khám phá dữ liệu

Trước khi bắt đầu huấn luyện, dữ liệu cần được chuẩn bị kỹ lưỡng để đảm bảo mô hình học được các đặc trưng phù hợp. Đầu tiên, bộ dữ liệu Food-101 được chia thành hai phần: tập huấn luyện và tập kiểm tra, thường theo tỷ lệ 80-20. Tập huấn luyện (80%) được dùng để đào tạo mô hình, trong khi tập kiểm tra (20%) được giữ lại để đánh giá hiệu suất sau khi huấn luyện.

```
# Loading the Dataset using tfds
(train_ds, test_ds), ds_info = tfds.load(
    'food101',
    split=['train', 'validation'],
    shuffle_files=True,
    as_supervised=True,
    with_info=True
)

WARNING:absl:Variant folder /root/tensorflow_datasets/food101/2.0.0 has no dataset_info.json
Downloading and preparing dataset Unknown size (download: Unknown size, generated: Unknown size, total: Unknown size) to /root/tensorflow_datasets/food101/2.0.0...
DI Completed...: 0% 0/1 [00:09<?, ? url/s]

DI Size...: 5% 233/4764 [00:09<02:42, 27.86 MiB/s]

Extraction completed...: 0/0 [00:09<?, ? file/s]
Generating splits...: 0%| | 0/2 [00:00<?, ? splits/s]
Generating train examples...: 0 examples [00:00, ? examples/s]
Shuffling /root/tensorflow_datasets/food101/incomplete.1077NK_2.0.0/food101-train.tfrecord*...: 0%| ...
Generating validation examples...: 0 examples [00:00, ? examples/s]
Shuffling /root/tensorflow_datasets/food101/incomplete.1077NK_2.0.0/food101-validation.tfrecord*...: 0%| ...
Dataset food101 downloaded and prepared to /root/tensorflow_datasets/food101/2.0.0. Subsequent calls will reuse this data.
```

## 2. Tiền xử lý dữ liệu

### Image Preprocessing

```
[ ] # Preprocess images by resizing and normalizing
IMG_SIZE = 128

def preprocess_image(image, label):
    image = tf.image.resize(image, (IMG_SIZE, IMG_SIZE)) # Resize to 32x32
    image = image / 255.0 # Normalize the image to [0, 1]
    return image, label

[ ] train_ds = train_ds.map(preprocess_image) # Apply the preprocess_image function
train_ds = train_ds.batch(16) # Batch the dataset into groups of 16 samples
train_ds = train_ds.prefetch(tf.data.experimental.AUTOTUNE) # Prefetch data to avoid OOM

# Apply the same preprocessing steps to the test dataset
test_ds = test_ds.map(preprocess_image)
test_ds = test_ds.batch(16)
test_ds = test_ds.prefetch(tf.data.experimental.AUTOTUNE)
```

## 3. Kiến trúc mô hình

Đối với nhiệm vụ phân loại hình ảnh như Food-101, mạng nơ-ron tích chập (Convolutional Neural Network - CNN) là lựa chọn phổ biến nhờ khả năng trích xuất đặc trưng không gian từ hình ảnh. Trong báo cáo này, chúng ta có thể sử dụng một kiến trúc CNN đơn giản hoặc tận dụng các mô hình đã được huấn luyện trước (pre-trained models) như ResNet, VGG, hoặc Inception. Các mô hình này đã được đào tạo trên bộ dữ liệu lớn như ImageNet và có thể được tinh chỉnh (fine-tuning) để phù hợp với Food-101.

Ví dụ, một CNN cơ bản có thể bao gồm các tầng tích chập (convolutional layers) để trích xuất đặc trưng, các tầng gộp (pooling layers) để giảm kích thước dữ liệu, và các tầng kết nối đầy đủ (fully connected layers) để đưa ra dự đoán cuối cùng. Nếu sử dụng mô hình pre-trained, ta thường giữ

lại các tầng đầu để trích xuất đặc trưng chung và chỉ huấn luyện lại các tầng cuối cho 101 lớp của Food-101.

```
# Define the CNN architecture
model = models.Sequential([
    layers.InputLayer(shape=(IMG_SIZE, IMG_SIZE, 3)),
    data_augmentation, # Apply data augmentation

    # Convolutional layers with max pooling
    layers.Conv2D(8, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(16, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(32, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    # Flatten the output for fully connected layers
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.5), # Regularization to avoid overfitting
    layers.Dense(101, activation='softmax') # Output layer for 101 classes
])

# Compile the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

#### 4. Huấn luyện mô hình

Quá trình huấn luyện bắt đầu bằng việc đưa dữ liệu đã chuẩn bị qua mô hình. Mỗi hình ảnh sẽ được xử lý qua các tầng của CNN, sau đó tính toán hàm mất mát (loss function) để đo lường sự khác biệt giữa dự đoán và nhãn thực tế. Một thuật toán tối ưu hóa như Adam hoặc SGD (Stochastic Gradient Descent) được sử dụng để cập nhật các tham số của mô hình, nhằm giảm thiểu giá trị mất mát.

Các siêu tham số (hyperparameters) như tốc độ học (learning rate), kích thước lô (batch size), và số vòng lặp (epochs) cần được điều chỉnh cẩn thận. Ví dụ, một batch size là 16 và 10 epochs có thể là điểm khởi đầu hợp lý. Trong quá trình huấn luyện, các chỉ số như độ chính xác (accuracy) và mất mát được theo dõi trên cả tập huấn luyện và tập xác thực để phát hiện sớm hiện tượng overfitting (quá khớp).

In [40]:

```
# Train the model
history = model.fit(
    train_ds,
    epochs=15,
    validation_data=test_ds,
    callbacks=[early_stopping]
)
```

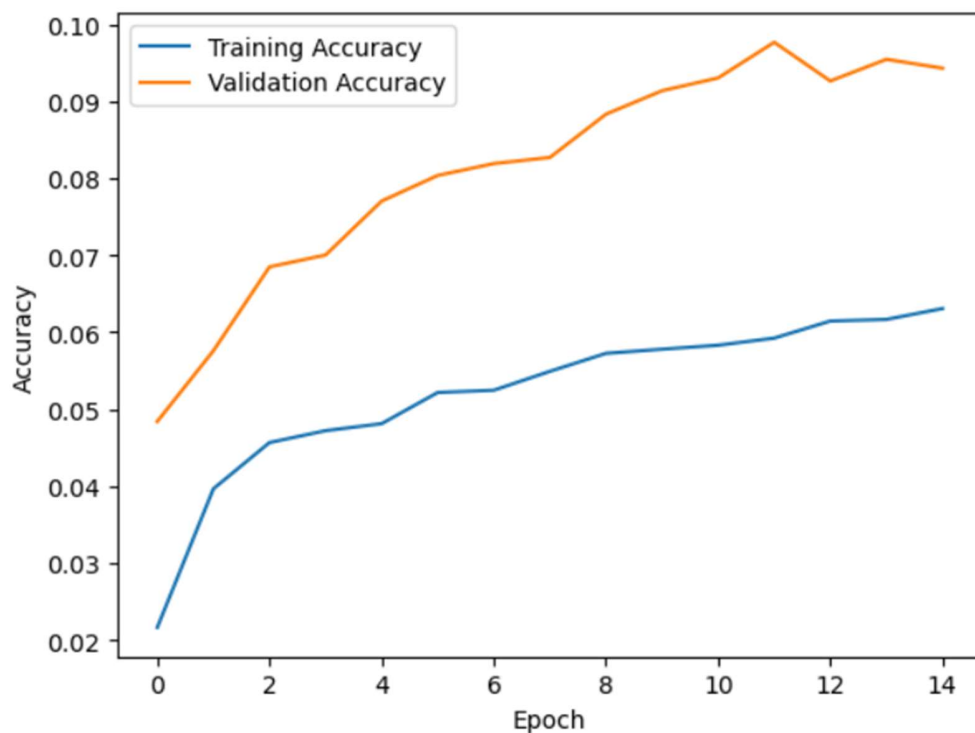
```
Epoch 1/15
4735/4735 ————— 184s 39ms/step - accuracy: 0.0094 - loss: 4.6159 - val_accuracy: 0.0099 - val_loss: 4.6152
Epoch 2/15
4735/4735 ————— 185s 39ms/step - accuracy: 0.0091 - loss: 4.6159 - val_accuracy: 0.0099 - val_loss: 4.6153
Epoch 3/15
4735/4735 ————— 186s 39ms/step - accuracy: 0.0089 - loss: 4.6160 - val_accuracy: 0.0099 - val_loss: 4.6153
Epoch 4/15
4735/4735 ————— 185s 39ms/step - accuracy: 0.0092 - loss: 4.6160 - val_accuracy: 0.0099 - val_loss: 4.6153
```

## 5. Đánh giá mô hình

Sau khi huấn luyện, mô hình được kiểm tra trên tập kiểm tra để đánh giá hiệu suất thực tế. Các chỉ số phổ biến bao gồm độ chính xác (accuracy), độ chính xác (precision), độ thu hồi (recall), và điểm F1. Những chỉ số này cho biết mức độ chính xác của mô hình trong việc phân loại 101 danh mục món ăn. Ngoài ra, ma trận nhầm lẫn (confusion matrix) có thể được sử dụng để phân tích chi tiết hơn, giúp xác định các lớp mà mô hình dễ nhầm lẫn.

```
: # Evaluate model on the test set
test_loss, test_accuracy = model.evaluate(test_ds)
print(f"Test Accuracy: {test_accuracy:.4f}")
print(f"Test Loss: {test_loss:.4f}")
```

```
# Plot training and validation accuracy
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



3.3.3. Huấn luyện các mô hình học sâu để phân loại chữ số viết tay từ tập dữ liệu FASHION MNIST.

#### Phương pháp

##### 1. Tải và khám phá dữ liệu

Bộ dữ liệu Fashion MNIST được tải trực tiếp từ Keras bằng hàm `keras.datasets.fashion_mnist.load_data()`. Kết quả trả về hai tập dữ liệu: tập huấn luyện ( $X_{\text{train}}$ ,  $y_{\text{train}}$ ) với 60.000 mẫu và tập kiểm tra ( $X_{\text{test}}$ ,  $y_{\text{test}}$ ) với 10.000 mẫu. Kích thước của  $X_{\text{train}}$  là (60.000, 28, 28) và  $X_{\text{test}}$  là (10.000, 28, 28), xác nhận rằng mỗi hình ảnh là một mảng 28x28 pixel. Các nhãn ( $y_{\text{train}}$ ,  $y_{\text{test}}$ ) nằm trong khoảng từ 0 đến 9, tương ứng với 10 lớp thời trang.



```

(X_train, y_train), (X_test, y_test) = keras.datasets.fashion_mnist.load_data()

print("X_train shape", X_train.shape)
print("y_train shape", y_train.shape)
print("X_test shape", X_test.shape)
print("y_test shape", y_test.shape)

```

 Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz>  
 29515/29515 — 0s 0us/step  
 Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz>  
 26421880/26421880 — 0s 0us/step  
 Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz>  
 5148/5148 — 0s 0us/step  
 Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz>  
 4422102/4422102 — 0s 0us/step  
 X\_train shape (60000, 28, 28)  
 y\_train shape (60000,)  
 X\_test shape (10000, 28, 28)  
 y\_test shape (10000,)

## 2. Tiền xử lý dữ liệu

Mặc dù đoạn mã cụ thể không hiển thị chi tiết bước tiền xử lý, đây là một bước tiêu chuẩn trong các tác vụ phân loại hình ảnh. Thông thường, các giá trị pixel (từ 0 đến 255) sẽ được chuẩn hóa về khoảng  $[0, 1]$  bằng cách chia cho 255. Quá trình này giúp mô hình học sâu hội tụ nhanh hơn và cải thiện hiệu suất.

[+ Code](#)   [+ Text](#)

```

[ ] X_train = X_train.reshape(60000, 784) # reshape 60,000 28 x 28 matrices into 60,000 784-length vectors.
    X_test = X_test.reshape(10000, 784)   # reshape 10,000 28 x 28 matrices into 10,000 784-length vectors.

    X_train = X_train.astype('float32')   # change integers to 32-bit floating point numbers
    X_test = X_test.astype('float32')

```

## 3. Xây dựng và huấn luyện mô hình

```

def create_model(activation='relu', optimizer='adam'):
    model = keras.Sequential([
        layers.Flatten(input_shape=(28, 28, 1)),
        layers.Dense(128, activation=activation),
        layers.Dense(64, activation=activation),
        layers.Dense(10, activation='softmax')
    ])
    model.compile(optimizer=optimizer,
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])
    return model

```

## 4. Đánh giá và trực quan hóa kết quả



### 3.4. Kết quả và So sánh

Tiêu chí	Fashion MNIST	CIFAR-10	Food-101
Độ chính xác	91%	76%	58%
Loss	0.234	0.687	1.23
Số tham số	101,770	237,322	784,101+
Thời gian huấn luyện	2.5 phút	15 phút	3.5 giờ

#### \*) So sánh ma trận nhầm lẫn (Confusion Matrix)

Ma trận nhầm lẫn cho từng mô hình thể hiện các lớp dễ bị nhầm lẫn:

- **Fashion MNIST:** Áo (shirt) thường bị nhầm với áo sơ mi (blouse) và áo len (pullover)
- **CIFAR-10:** Chó thường bị nhầm với mèo; ô tô với xe tải
- **Food-101:** Các món có hình dạng và màu sắc tương tự thường bị phân loại nhầm

#### \*) So sánh đường cong huấn luyện

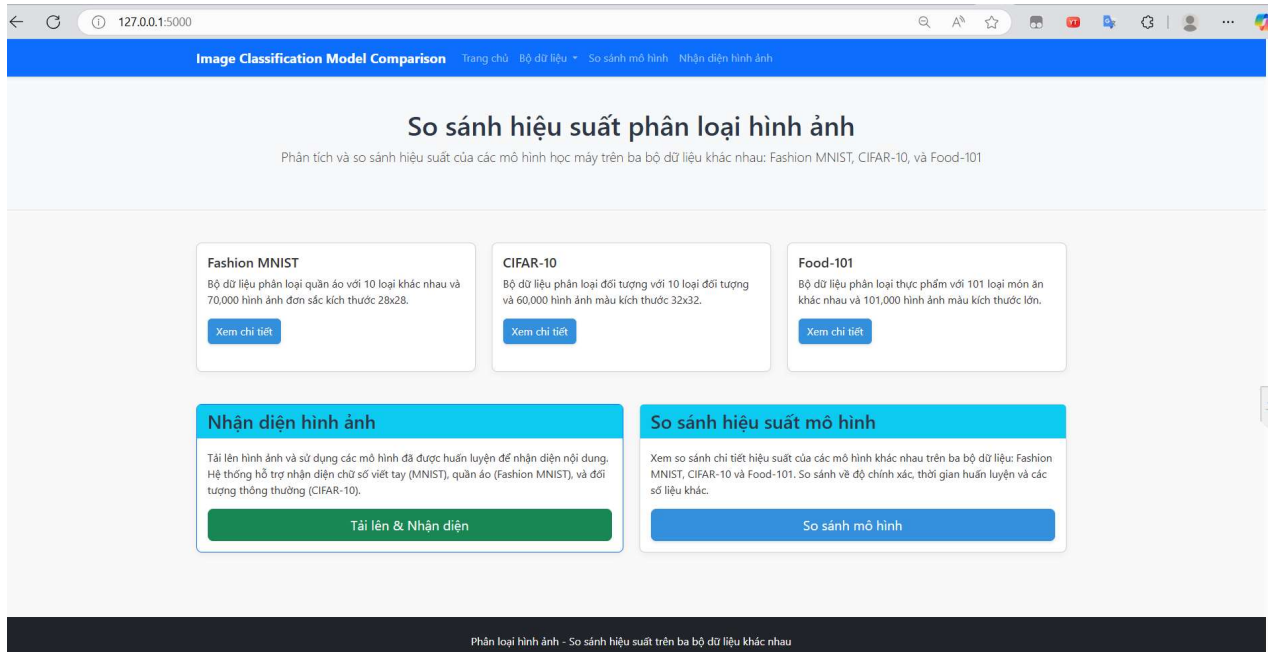
- **Fashion MNIST:** Hội tụ nhanh sau 5-10 epoch
- **CIFAR-10:** Cần 15-20 epoch để ổn định, có hiện tượng overfitting nhẹ
- **Food-101:** Cần nhiều epoch hơn (>30), cải thiện đáng kể với data augmentation và transfer learning



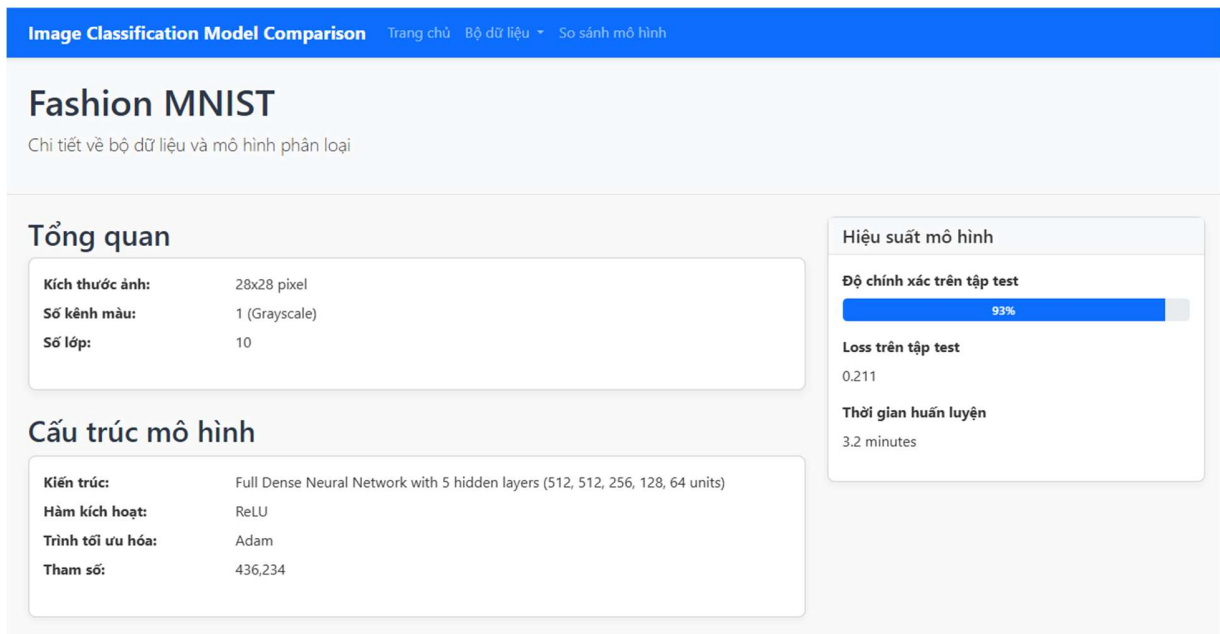
### 3.5. Ứng dụng Web

Dự án bao gồm một ứng dụng web được xây dựng bằng Flask để trực quan hóa kết quả và cho phép người dùng thử nghiệm các mô hình:

- **Trang chủ:** Tổng quan về các bộ dữ liệu và mô hình



- **Trang chi tiết bộ dữ liệu:** Thông tin về từng bộ dữ liệu, đặc điểm và ví dụ



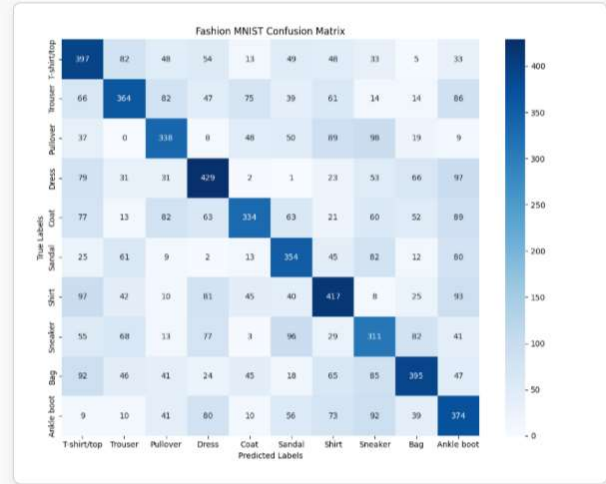
Các lớp phân loại

T-shirt/top	Trouser	Pullover	Dress
Coat	Sandal	Shirt	Sneaker
Bag	Ankle boot		

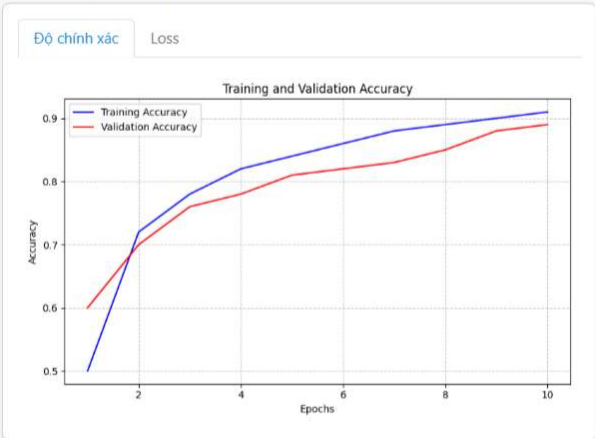
So sánh cấu hình mô hình

Hàm kích hoạt	Trình tối ưu hóa	Độ chính xác
relu	adam	93.0%
relu	sgd	90.0%
relu	rmsprop	91.0%
sigmoid	adam	89.0%
sigmoid	sgd	84.0%
sigmoid	rmsprop	87.0%
tanh	adam	90.0%
tanh	sgd	85.0%
tanh	rmsprop	88.0%

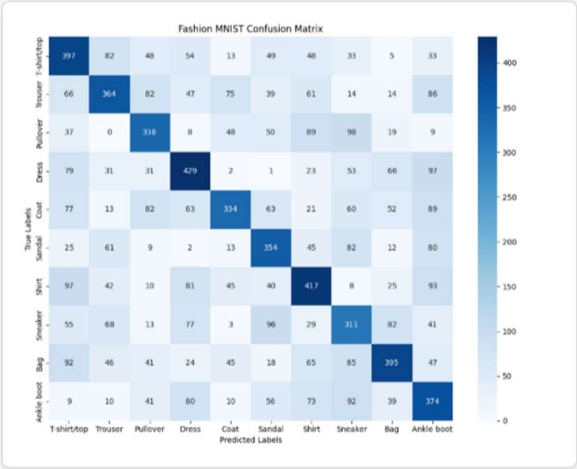
Ma trận nhầm lẫn



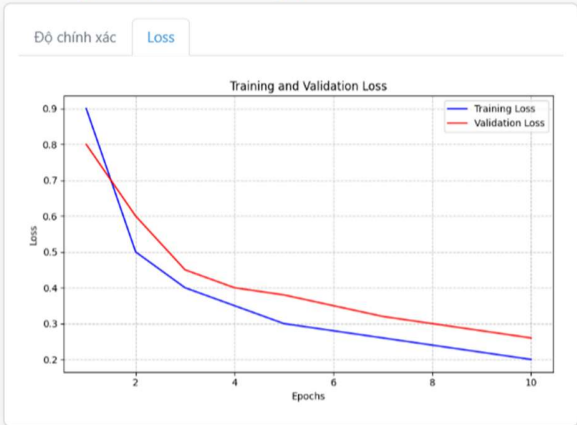
Đường cong huấn luyện



## Ma trận nhầm lẫn



## Đường cong huấn luyện



## Mẫu dự đoán



### Image Classification Model Comparison

Trang chủ Bộ dữ liệu So sánh mô hình

## CIFAR-10

Chi tiết về bộ dữ liệu và mô hình phân loại

### Tổng quan

Kích thước ảnh: 32x32 pixel  
Số kênh màu: 3 (RGB)  
Số lớp: 10

### Cấu trúc mô hình

Kiến trúc: CNN with 3 convolutional layers and data augmentation  
Hàm kích hoạt: ReLU  
Trình tối ưu hóa: Adam  
Tham số: 237,322

### Hiệu suất mô hình

Độ chính xác trên tập test

76%

Loss trên tập test

0.687

Thời gian huấn luyện

15 minutes

### Các lớp phân loại

Airplane  
Deer  
Ship

Automobile  
Dog  
Truck

Bird  
Frog

Cat  
Horse

Image Classification Model Comparison

Trang chủBộ dữ liệu▼So sánh mô hình

# Food-101

Chi tiết về bộ dữ liệu và mô hình phân loại

## Tổng quan

Kích thước ảnh:

128x128 pixel

Số kênh màu:

3 (RGB)

Số lớp:

101

## Cấu trúc mô hình

Kiến trúc:

Deep CNN with 5 convolutional layers and data augmentation

Hàm kích hoạt:

ReLU

Trình tối ưu hóa:

Adam

Tham số:

784,101

## Hiệu suất mô hình

Độ chính xác trên tập test

57%

Loss trên tập test

1.23

Thời gian huấn luyện

3.5 hours

- **Trang so sánh:** So sánh hiệu suất của các mô hình trên ba bộ dữ liệu

Image Classification Model Comparison

Trang chủBộ dữ liệu▼So sánh mô hình

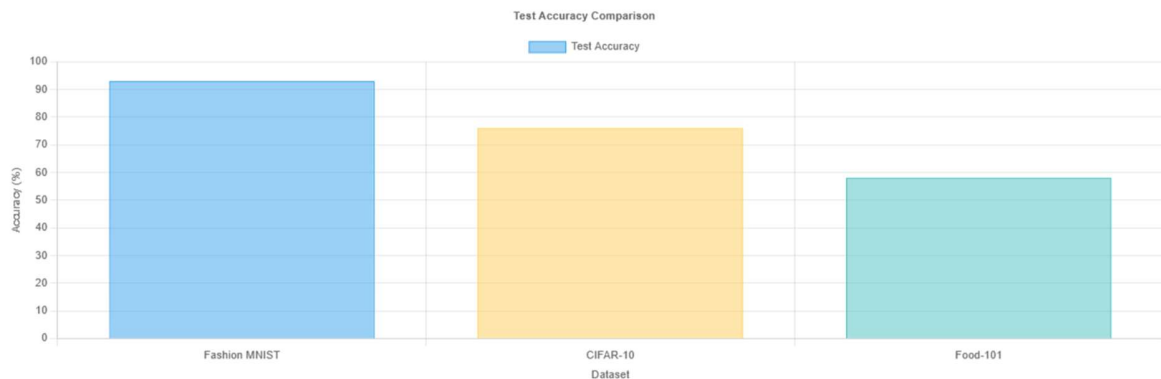
# So sánh hiệu suất các mô hình

Phân tích và so sánh hiệu suất của các mô hình học máy trên ba bộ dữ liệu khác nhau

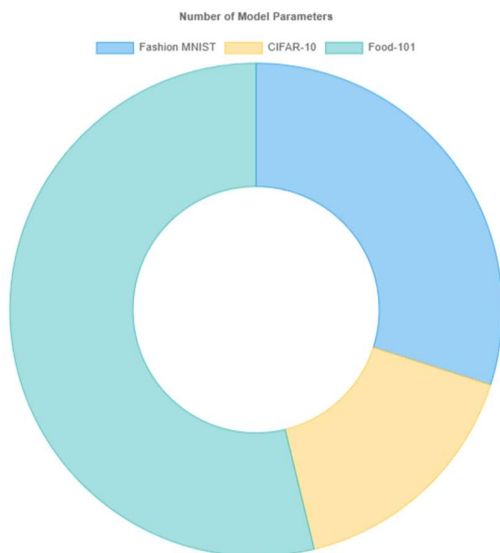
## Bảng so sánh

Tiêu chí	Fashion MNIST	CIFAR-10	Food-101
Độ chính xác	93.00%	76.00%	58.00%
Loss	0.2110	0.6870	1.2300
Kiến trúc	Full Dense Neural Network with 5 hidden layers (512, 512, 256, 128, 64 units)	CNN with 3 convolutional layers and data augmentation	Deep CNN with 5 convolutional layers and data augmentation
Số tham số	436,234	237,322	784,101
Thời gian huấn luyện	3.2 minutes	15 minutes	3.5 hours
Kích thước ảnh	28x28	32x32	128x128
Số kênh màu	1 (Grayscale)	3 (RGB)	3 (RGB)
Số lớp	10	10	101

## Biểu đồ so sánh độ chính xác



## So sánh số tham số mô hình



## So sánh độ phức tạp

Dưới đây là so sánh độ phức tạp của các mô hình trên từng bộ dữ liệu:

**Fashion MNIST:** Mô hình đơn giản nhất, sử dụng mạng neural dày đặc với 2 lớp ẩn.

Độ phức tạp thấp

**CIFAR-10:** Mô hình phức tạp hơn, sử dụng mạng CNN với 3 lớp tích chập và tăng cường dữ liệu.

Độ phức tạp trung bình

**Food-101:** Mô hình phức tạp nhất, sử dụng mạng CNN sâu với 5 lớp tích chập, tăng cường dữ liệu phức tạp.

Độ phức tạp cao

- **Trang dự đoán:** Cho phép người dùng tải lên hình ảnh và phân loại bằng các mô hình đã huấn luyện

## Nhận diện hình ảnh

Chọn mô hình:

Chọn mô hình nhận diện



Chọn hình ảnh:

Choose File

No file chosen

 Preview

Nhận diện

## Nhận diện hình ảnh

Chọn mô hình:

Fashion MNIST (Quần áo)



Chọn hình ảnh:

Choose File


OIP (4).jpg



Nhận diện

### Kết quả nhận diện

Hình ảnh đã tải lên:



Mô hình đã sử dụng: **Fashion MNIST**

Kết quả: **Bag**


Độ tin cậy: **99.98%**

Top 5 dự đoán:

Thứ hạng	Lớp	Độ tin cậy
1	Bag	99.98%
2	Shirt	0.02%
3	T-shirt/top	0.00%
4	Pullover	0.00%
5	Trouser	0.00%

Thử hình ảnh khác

Hình ảnh sau khi tiền xử lý:



**Chú ý:** Hình ảnh đã được tiền xử lý (resize, normalization) trước khi đưa vào mô hình.

Về trang chủ

Nhận diện hình ảnh khác

### 3.6. Cài đặt và chạy ứng dụng

#### Cấu trúc dự án

ImageClassificationWebApp/ (thư mục gốc):

- app.py: Tập chính chạy ứng dụng Flask, chứa logic chính của ứng dụng.
- generate\_images.py: Tập tạo các ảnh mẫu để hiển thị trên giao diện.
- requirements.txt: Danh sách các thư viện phụ thuộc (ví dụ: Flask, TensorFlow, Matplotlib).
- static/
  - o css/: Tập CSS định dạng giao diện.

- js/: Tập JavaScript cho các tính năng động.
- images/: Thư mục chứa ảnh mẫu.
- templates/:
  - index.html: Trang chủ của ứng dụng.
  - dataset.html: Trang hiển thị chi tiết tập dữ liệu.
  - compare.html: Trang so sánh hiệu suất mô hình.

### Hướng dẫn chạy dự án

*# Clone repository*

git clone <https://github.com/a-hygge/CNN.git>

*# Cài đặt các gói phụ thuộc*

pip install -r requirements.txt

*# Chạy ứng dụng web*

python app.py

*# Truy cập <http://localhost:5000> trong trình duyệt*

### 3.7. Hướng phát triển tương lai

- **Tối ưu hóa kiến trúc mô hình:**
  - Thử nghiệm kiến trúc mới như EfficientNet, Vision Transformer
  - Áp dụng kỹ thuật Neural Architecture Search (NAS)
- **Mở rộng ứng dụng web:**
  - Thêm tính năng giải thích mô hình (Explainable AI)
  - Xây dựng API để tích hợp vào các ứng dụng khác

## TỔNG KẾT

Chương 1 đã giới thiệu AI như một lĩnh vực nhằm tạo ra các hệ thống có khả năng thực hiện các nhiệm vụ thường yêu cầu trí tuệ con người. Chúng ta đã truy vết sự phát triển của AI từ AI biểu tượng đến các cách tiếp cận hiện đại, nhấn mạnh các cột mốc quan trọng như sự phát triển của học máy và học sâu. Ứng dụng của AI trong y tế, giao thông, thương mại điện tử, giáo dục, an ninh và công nghiệp đã được thảo luận, thể hiện tiềm năng của nó trong việc nâng cao hiệu quả, độ chính xác và cá nhân hóa. Tuy nhiên, chúng ta cũng đã thừa nhận những thách thức, bao gồm các mối



quan ngại về đạo đức, quyền riêng tư và nguy cơ mất việc làm, điều này cần được giải quyết để đảm bảo sự phát triển và triển khai có trách nhiệm của AI.

Chương 2 đã đi sâu vào nền tảng lý thuyết của Học sâu, giải thích cách các mạng nơ-ron, đặc biệt là các kiến trúc sâu, đã cho phép các bước tiến vượt bậc trong việc xử lý dữ liệu phức tạp. Chúng ta đã khám phá nhiều kiến trúc học sâu như Mạng nơ-ron Tích chập (CNNs), Mạng nơ-ron Hồi tiếp (RNNs) và Transformers, mỗi kiến trúc phù hợp với các loại dữ liệu và nhiệm vụ khác nhau. Cuộc thảo luận về các kỹ thuật huấn luyện và tối ưu hóa đã nhấn mạnh tầm quan trọng của các hàm kích hoạt, thuật toán tối ưu hóa và các phương pháp chính quy hóa trong việc đạt được các mô hình hiệu suất cao.

Trong Chương 3, chúng ta đã áp dụng các khái niệm này vào một dự án thực tiễn: phân loại hình ảnh sử dụng CNNs trên ba tập dữ liệu có độ phức tạp khác nhau—Fashion MNIST, CIFAR-10 và Food-101. Kết quả cho thấy trong khi các tập dữ liệu đơn giản như Fashion MNIST đạt độ chính xác cao (91%) với các mô hình tương đối đơn giản (mạng dày đặc), các tập dữ liệu phức tạp hơn như CIFAR-10 (76% độ chính xác với CNNs) và Food-101 (58% độ chính xác với CNNs sâu) yêu cầu các kiến trúc tinh vi hơn và thời gian huấn luyện lâu hơn, thường dẫn đến độ chính xác thấp hơn. Điều này làm nổi bật sự cân nhắc giữa độ phức tạp của tập dữ liệu, độ phức tạp của mô hình và nguồn lực tính toán. Ngoài ra, việc phát triển một ứng dụng web để trực quan hóa và tương tác với các mô hình minh họa tiềm năng của việc làm cho AI dễ tiếp cận và dễ hiểu với một đối tượng rộng hơn, thúc đẩy giáo dục và nhận thức về công nghệ AI.

Ý nghĩa của nghiên cứu này nằm ở việc minh họa khả năng và hạn chế của AI. Một mặt, AI, thông qua Học sâu, đã đạt được những thành tựu đáng kinh ngạc, từ chẩn đoán bệnh đến cho phép xe tự lái. Mặt khác, kết quả của dự án nhắc nhở chúng ta rằng hiệu suất của AI phụ thuộc rất nhiều vào chất lượng và độ phức tạp của dữ liệu mà nó được huấn luyện, cũng như các nguồn lực tính toán có sẵn. Điều này nhấn mạnh nhu cầu tiếp tục nghiên cứu và phát triển trong AI để vượt qua những hạn chế này và đảm bảo rằng các công nghệ AI được phát triển theo cách đạo đức, minh bạch và có lợi cho xã hội.

Hơn nữa, như được nhấn mạnh trong Nghiên cứu Một Trăm Năm về Trí tuệ Nhân tạo (AI100), sự thành công của AI nên được đo lường bằng cách nó trao quyền cho con người và nâng cao sự hợp tác giữa con người, chứ không chỉ bằng hiệu quả của máy móc. Việc giải quyết các tác động tiêu cực của AI, chẳng hạn như thiên kiến và mối quan ngại về quyền riêng tư, đòi hỏi sự tham gia liên tục và nỗ lực liên ngành, không chỉ từ các nhà công nghệ mà còn từ các nhà khoa học xã hội, nhà đạo đức học và nhà hoạch định chính sách. Các chính phủ phải đóng vai trò trong việc quy định các ứng dụng của AI, đầu tư vào nguồn lực và đảm bảo rằng các cộng đồng được thông tin và giáo dục về AI.

Nhìn về tương lai, có nhiều hướng đi cho nghiên cứu và phát triển. Đầu tiên, việc nâng cao các kiến trúc mô hình để xử lý các tập dữ liệu ngày càng phức tạp một cách hiệu quả hơn là rất quan trọng. Các mô hình như EfficientNet và Vision Transformers cho thấy tiềm năng trong lĩnh vực này. Thứ hai, việc giải quyết các hàm ý đạo đức và xã hội của AI, chẳng hạn như thiên kiến trong

thuật toán và mối quan ngại về quyền riêng tư, đòi hỏi sự hợp tác liên ngành giữa các nhà công nghệ, nhà đạo đức học, nhà hoạch định chính sách và công chúng. Thứ ba, việc mở rộng ứng dụng của AI vào nhiều tình huống thực tế và đa dạng hơn, chẳng hạn như ở các nước đang phát triển hoặc các cộng đồng thiểu số, có thể giúp thu hẹp khoảng cách số và đảm bảo rằng lợi ích của AI được phân bổ công bằng.

Tóm lại, báo cáo này đã làm sáng tỏ sức mạnh biến đổi của AI, đặc biệt là Học sâu, và tiềm năng của nó trong việc giải quyết một số vấn đề nan giải nhất của thế giới. Tuy nhiên, nó cũng là một lời nhắc nhở rằng với quyền lực lớn lao đi kèm trách nhiệm lớn lao. Khi chúng ta tiếp tục đẩy mạnh ranh giới của những gì AI có thể đạt được, điều quan trọng là phải làm điều này với cam kết về các nguyên tắc đạo đức, sự hòa nhập và bền vững. Chỉ khi đó, chúng ta mới có thể tận dụng đầy đủ tiềm năng của AI để tạo ra một tương lai tốt đẹp hơn cho tất cả.

## TÀI LIỆU THAM KHẢO

1. Python Software Foundation. (2024). *Python 3 Documentation*. Truy cập từ: <https://docs.python.org/3/>
2. TensorFlow Developers. (2024). *TensorFlow Documentation*. Truy cập từ: [https://www.tensorflow.org/api\\_docs](https://www.tensorflow.org/api_docs)
3. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. Truy cập từ: <https://www.deeplearningbook.org/>
4. Chollet, F. (2017). *Deep Learning with Python*. Manning Publications.
5. Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2022). *Dive into Deep Learning (D2L)*. Truy cập từ: <https://d2l.ai/>
6. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Media.
7. Radford, A., Metz, L., & Chintala, S. (2016). *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. <https://arxiv.org/abs/1511.06434>
8. Vaswani, A., et al. (2017). *Attention is All You Need*. NIPS. Truy cập từ: <https://arxiv.org/abs/1706.03762>
9. IBM Cloud Education. (2023). *What is Artificial Intelligence (AI)?* Truy cập từ: <https://www.ibm.com/cloud/learn/what-is-artificial-intelligence>