[1] **Hybrid convolutional neural network model** là một mô hình mạng neural tích chập kết hợp các phương pháp học sâu và học máy truyền thống để tăng hiệu quả phân loại và phát hiện của hệ thống.

Mô hình này bao gồm các tầng tích chập, các tầng liên kết đầy đủ và các tầng kết nối ngược (deconvolution) để trích xuất đặc trưng của dữ liệu và tạo ra các dự đoán về lớp hoặc nhãn của dữ liệu.

Nó kết hợp cả các thuật toán học máy truyền thống như hồi quy logistic, SVM, k-nearest neighbors và các thuật toán học sâu như mạng neural tích chập và mạng neural cơ sở để tạo ra một mô hình phân loại chính xác hơn và nhạy hơn đối với các tình huống khác nhau.

Trong bài viết được đề cập, mô hình lai này được sử dụng để phát hiện các cuộc tấn công trong mạng IoT.

[2] **IoT module** là một thiết bị hoặc một bộ phận của hệ thống IoT, được sử dụng để kết nối các thiết bị đo, cảm biến hoặc các thiết bị khác với mạng IoT và cho phép chúng truyền thông với nhau và với các trung tâm điều khiển.

Module IoT thường có các tính năng như kết nối mạng không dây, tích hợp các cảm biến và bộ xử lý, hỗ trợ các giao thức truyền thông và các tính năng bảo mật.

Các module IoT có thể được sử dụng trong nhiều ứng dụng khác nhau như trong các thiết bị đeo tay thông minh, các hệ thống tự động hóa nhà thông minh, các thiết bị y tế thông minh, và các hệ thống giám sát công nghiệp.

[3] "**Tradeoff between the IoT environment and data in the system**" có nghĩa là cân bằng giữa môi trường IoT và dữ liệu trong hệ thống.

Khi **triển khai IoT**, cần **đảm bảo** môi trường IoT được **hoạt động một cách an toàn, ổn định và hiệu quả**, đồng thời **đảm bảo dữ liệu được bảo mật**, truy cập được và sử dụng một cách hiệu quả.

**Khả năng thích ứng (adaptability)** trong IoT là **khả năng** của các **thiết bị IoT** để **thích nghi** và thay đổi trong môi trường động, bao gồm **khả năng tự động cấu hình, tái cấu trúc và phát hiện lỗi**. Điều này cho phép **các thiết bị IoT hoạt động tốt** trong các **điều kiện thay đổi** và đa dạng và tăng cường tính linh hoạt của hệ thống..

[4] Một ví dụ về quá trình xác minh trong IoT khác nhau từ hệ thống này sang hệ thống khác là quá trình **xác minh thông qua mã PIN** (Personal Identification Number) trên một thiết bị IoT, ví dụ như một hệ thống an ninh nhà thông minh.

Một hệ thống nhà thông minh sẽ yêu cầu người dùng **nhập mã PIN** để **mở khóa cửa hoặc tắt hệ thống báo động.**

Tuy nhiên, trong một hệ thống khác như **hệ thống y tế thông minh**, xác minh sẽ được thực hiện bằng cách **sử dụng các thông tin y tế** của người dùng để đảm bảo rằng chỉ có các bác sĩ hoặc nhân viên y tế được phép truy cập và xem thông tin của bệnh nhân.

Do đó, **quá trình xác minh** trong IoT sẽ **phụ thuộc vào mục đích** và **tính chất của hệ thống IoT được sử dụng.**

[5] Ví dụ về việc **phân loại đối tượng** trong IoT có thể là trong hệ thống quản lý giao thông thông minh, các đối tượng có thể được phân loại **là ô tô, xe máy, xe đạp, người đi bộ, v.v.**

Các loại đối tượng này **cần được phân loại** để đảm bảo rằng các thiết bị IoT **có thể thu thập dữ liệu chính xác** về số lượng, vị trí và tốc độ di chuyển của từng đối tượng.

Việc phân loại đối tượng được **thực hiện ở giai đoạn ban đầu** của quá trình IoT giúp **cải thiện khả năng hoạt động** của hệ thống và **tăng tính đáng tin cậy của dữ liệu thu thập được.**

**[6] Quy trình ủy quyền (authorization process) trong một mạng IoT** là quá trình **xác định các quyền truy cập** cho người dùng **đối với dữ liệu** **trong** môi trường **IoT**.

Quá trình này **cần được xác định một cách chính xác** dựa trên đánh giá mức độ tin cậy của các thiết bị IoT và độ bảo mật của thông tin được truyền.

**Khi** quy trình ủy quyền **được định nghĩa tốt**, người dùng sẽ chỉ **có thể truy cập** và **sử dụng dữ liệu mà họ có quyền truy cập**. Quy trình này đóng một **vai trò quan trọng** trong việc **bảo vệ mạng IoT khỏi các cuộc tấn công** và **đảm bảo tính riêng tư** và **bảo mật của dữ liệu**.

[7] **Data-oriented security mechanism** là một cơ chế bảo mật **tập trung** vào **bảo vệ dữ liệu** trong hệ thống và **ngăn chặn truy cập trái phép** đến nguồn dữ liệu từ các kẻ tấn công.

Cơ chế này tập trung vào bảo vệ **tính bảo mật, toàn vẹn** và **sẵn sàng của dữ liệu**, bao gồm các hoạt động **như mã hóa, chứng thực, kiểm soát truy cập** và **giám sát dữ liệu**.

**Data-oriented security mechanism** là một trong những phương pháp bảo mật **quan trọng** trong môi trường **IoT**, nơi **lượng dữ liệu lớn** và **đa dạng** tạo ra những thách thức riêng biệt đối với việc bảo mật.

[8] "**Large volume data streams**" (luồng dữ liệu lớn) là thuật ngữ chỉ các **lượng dữ liệu lớn** và **liên tục** được tạo ra **từ các nguồn khác nhau** và được **truyền tải** trong **thời gian thực** qua mạng.

Ví dụ về luồng dữ liệu lớn bao gồm các trang web phương tiện xã hội như **Twitter** hoặc **Facebook**, các **trò chơi trực tuyến**, các **cảm biến Io**T và các **hệ thống giám sát mạng.** Để xử lý các luồng dữ liệu lớn, các hệ thống phải có khả năng **xử lý và phân tích dữ liệu nhanh chóng và hiệu quả** để **phát hiện các sự cố hoặc mối đe dọa đến bảo mật**

**"Các chữ ký của các cuộc tấn công đã biết" (known attack signatures)** là một **tập hợp các mẫu**, **đặc điểm** hoặc **dấu hiệu** **của các cuộc tấn công đã được biế**t đến và được **lưu trữ trong cơ sở dữ liệu** của hệ thống phát hiện xâm nhập (IDS).

Khi IDS phát hiện ra các dấu hiệu hoặc mẫu trong dữ liệu mạng hoặc hệ thống mà tương tự với các chữ ký của các cuộc tấn công đã biết, nó sẽ phát hiện và báo cáo lại cho quản trị viên để xử lý.

Việc sử dụng các chữ ký của các cuộc tấn công đã biết giúp IDS phát hiện các cuộc tấn công mới một cách hiệu quả và nhanh chóng hơn bằng cách so sánh các hoạt động trên mạng với các mẫu đã biết trước đó.

Tuy nhiên, việc sử dụng các chữ ký này cũng có nhược điểm là nó không thể phát hiện các cuộc tấn công mới mà không có sự tương tự với các chữ ký đã biết trước.

[9] **Low power networks** là **mạng không dây** được thiết kế để **tiết kiệm năng lượng** và **kéo dài tuổi thọ pin của các thiết bị mạng**. Các thiết bị trong mạng này thường được **thiết kế để sử dụng ít năng lượng hơn**, **có thể hoạt động** trong một **khoảng thời gian dài** mà không cần thay pin, và **sử dụng các giao thức tiết kiệm năng lượng** để giảm thiểu lượng năng lượng tiêu thụ trong quá trình truyền tải dữ liệu.

Các ví dụ về low power networks bao gồm các mạng cảm biến không dây (Wireless Sensor Networks - WSN), mạng khối thiết bị thông minh (Internet of Things - IoT) và mạng không dây cá nhân (Wireless Personal Area Networks - WPAN).

[10] Mô hình CNN (**Convolutional Neural Network**) là một trong những kiểu mô hình được sử dụng nhiều trong học sâu (**deep learning**) để xử lý các tác vụ **nhận dạng ảnh, âm thanh, văn bản và video**.

Mô hình này sử dụng (**convolutional layers**) để học các đặc trưng (**features**) của **dữ liệu đầu** vào thông qua việc t**rích xuất các thông tin quan trọng** từ các vùng nhỏ của ảnh hay video.

Sau đó, các lớp **pooling** được **sử dụng để giảm kích thước của dữ liệu đầu ra**, giúp **giảm độ phức tạp tính toán** và **cải thiện khả năng tổng quát hóa của mô hình**.

Cuối cùng, các lớp kết nối đầy đủ (**fully connected layer**s) được sử dụng để **kết hợp các đặc trưng** đã được học để **phân loại dữ liệu đầu vào**. Mô hình CNN đã được áp dụng thành công trong nhiều lĩnh vực như xử lý ảnh, nhận dạng giọng nói và xử lý văn bản tự nhiên.

[11] **LSTM** (Long Short-Term Memory) là một loại mô hình mạng nơ-ron dựa trên mạng RNN (Recurrent Neural Network), được thiết kế để xử lý các chuỗi dữ liệu có thể dài hoặc ngắn. LSTM có khả năng giữ lại thông tin quan trọng và loại bỏ thông tin không quan trọng trong quá trình xử lý chuỗi, giúp cho việc dự đoán hoặc phân loại trên dữ liệu chuỗi trở nên hiệu quả hơn. LSTM có thể được sử dụng cho nhiều nhiệm vụ khác nhau, chẳng hạn như: xử lý ngôn ngữ tự nhiên, dự đoán chuỗi thời gian, phân tích dữ liệu chuỗi thị giác, và nhiều hơn nữa.

[12] **Convolutional matrix** (ma trận tích chập) là một phương pháp xử lý ảnh trong mạng deep learning.

Nó **được sử dụng** để **giảm số lượng tham số** và tính toán cần thiết **để huấn luyện mạng**.

Convolutional matrix **được sử dụng** để **tìm kiếm các đặc trưng của ảnh** bằng cách áp dụng một ma trận (hay kernel) có kích thước nhỏ lên toàn bộ ảnh. Kết quả của phép tích chập này sẽ tạo ra một ảnh mới, với các đặc trưng được tách ra rõ ràng hơn.

Các **convolutional matrix khác nhau** có thể được sử dụng để **tìm kiếm các đặc trưng khác nhau của ảnh**, chẳng hạn như cạnh, góc, hay các đặc trưng phức tạp hơn như vật thể.

[13] **Filter matrices** (còn được gọi là kernel hoặc filter) trong mạng CNN (Convolutional Neural Network) là một ma trận **dùng để phát hiện** và **trích xuất các đặc trưng (features) của dữ liệu đầu vào**.

Quá trình tích chập (convolution) đ**ược thực hiện bằng cách trượt filter qua các vùng của dữ liệu đầu vào để tạo ra các feature maps**, từ đó **sử dụng các phép biến đổi tiếp theo để rút trích thông tin hữu ích**.

Filter matrices thường được **khởi tạo ngẫu nhiên ban đầu** và **được cập nhật trong quá trình huấn luyện mạng** để tối ưu hóa quá trình trích xuất đặc trưng.

[14] "**Set of neurons for each labels**" là một thuật ngữ trong mạng neuron, đề cập đến việc sử dụng một nhóm các neuron để đại diện cho một tập hợp các nhãn khác nhau.

Khi huấn luyện một mô hình mạng neuron, **mỗi nhãn** sẽ có **một nhóm các neuron** tương ứng **để đưa ra dự đoán**.

Ví dụ, trong một mô hình phân loại hình ảnh với **10 lớp khác nhau**, sẽ có một tập hợp các neuron ở lớp đầu ra đại diện cho từng lớp khác nhau, tức là có **10 nhóm neuron** ở lớp đầu ra. Khi một hình ảnh được đưa vào mô hình, mỗi nhóm neuron sẽ đưa ra một giá trị tương ứng, thể hiện độ chắc chắn của mô hình về xác suất hình ảnh đó thuộc về lớp nào.

[15] **Hidden layer** (tầng ẩn) là một tầng trong kiến trúc của mạng neural, **nằm giữa tầng đầu vào và tầng đầu ra**.

Tầng ẩn nhận đầu vào từ tầng đầu vào, **xử lý thông tin và tạo ra đầu ra cho tầng đầu ra**. Chức năng của tầng ẩn là **trích xuất các đặc trưng từ dữ liệu đầu vào**, giúp mô hình học được mối quan hệ phức tạp giữa các đặc trưng và **đưa ra kết quả dự đoán** chính xác hơn. Một mạng neural có thể có nhiều tầng ẩn, tùy thuộc vào độ phức tạp của mô hình và độ phức tạp của dữ liệu đầu vào

[16] "**Shared matrices weight depth**" trong đoạn văn được đề cập đến là một **tham số được sử dụng để tính tích của ma trận tích chập và ma trận bộ lọc trong mạng CNN**.

Nó được định nghĩa là **độ sâu trọng số** của các ma trận chia sẻ được sử dụng để tính toán ma trận tích chập trong các tầng ẩn của mạng. Thông qua việc chia sẻ trọng số này giữa các tầng, mô hình có thể học được các tính năng cục bộ của ảnh và tránh overfitting.

[17] **Tích chập (Convolution)** là một phép toán trong **xử lý tín hiệu** và xử lý **ảnh**, được sử dụng để **trích xuất thông tin từ các tín hiệu đầu vào**. Trong lĩnh vực trí tuệ nhân tạo, tích chập là một **phép toán quan trọng** trong mạng neuron nhân tạo sử dụng **để phân tích hình ảnh, âm thanh, văn bản và các loại dữ liệu khác.**

Trong xử lý ảnh, phép tích chập thực hiện việc "**trượt**" **một** mặt nạ (**kernel) trên toàn bộ bức ảnh đầu vào**, **tính toán tích chập giữa** **kernel** và **từng vùng của ảnh** để **tạo ra một bức ảnh mới với các đặc trưng được trích xuất từ ảnh gốc**. Các đặc trưng này có thể là **cạnh, góc, đường cong, điểm nổi bật,** và các đặc trưng khác phù hợp với mục đích của bài toán.

**Kernel** được **trượt** (đi qua) trên bức ảnh **từng pixel một**, tính toán giá trị tích chập tại mỗi vị trí trên ảnh đầu vào. Khi kernel được trượt trên ảnh, **mỗi vị trí trên ảnh** sẽ được **gán giá trị mới bằng tích chập của kernel với phần ảnh tương ứng**.

**Kết quả của phép tích chập** sẽ **tạo ra một bức ảnh mới**, nơi mà **mỗi điểm ảnh trên bức ảnh mới** tương ứng **với kết quả tích chập của kernel với vùng ảnh tương** ứng trên bức ảnh gốc.

**Quá trình trượt** kernel trên bức ảnh này **cho phép trích xuất các đặc trưng của ảnh, như cạnh, góc, đường cong, điểm nổi bật và các đặc trưng khá**c. Tùy thuộc vào kích thước và giá trị của kernel, kết quả của phép tích chập có thể giúp chúng ta giảm nhiễu, tăng cường độ tương phản và giúp cho quá trình nhận dạng ảnh trở nên hiệu quả hơn.

[19] **Kích thước cửa sổ trượt** là kích thước của cửa sổ được trượt qua toàn bộ ảnh đầu vào

Được thực hiện bằng cách **chọn một khu vực trên ảnh đầu vào** có **kích thước cố định**, rồi sau đó **trượt** **trên đó** với một bước nhảy nhất định để **tạo ra các khung hình con (sub-image)** của ảnh

Kích thước cửa sổ trượt có **ảnh hưởng trực tiếp đến độ phân giải** và **khả năng phát hiện các đối tượng trong ảnh**.

Khi kích thước **filter càng lớn**, **độ phân giải** của các **khung hình con tạo ra** **sẽ giảm** và **khả năng phát hiện** các đối tượng **có kích thước nhỏ sẽ giảm đi**.

Tuy nhiên, khi **kích thước cửa sổ trượt càng nhỏ**, số lượng **khung hình con tạo ra cũng càng nhiều**, dẫn đến **tăng độ phức tạp tính toán**.

Do đó, kích thước cửa sổ trượt thường **được lựa chọn sao cho phù hợp với kích thước và số lượng đối tượng cần phát hiện trong ảnh**.

[20] **kích thước cửa sổ bộ lọc** được sử dụng trong các **thuật toán xử lý ảnh** là các **ma trận nhỏ được xác định trước kích thước**

**kích thước** của các bộ lọc cần sử dụng trong một bài toán cụ thể thường **được chọn và điều chỉnh dựa trên mục đích và độ phức tạp của bài toán**.

[18] **Shared weight matrices depth** là một khái niệm liên quan đến kiến trúc của mạng nơ-ron tích chập (CNN). Trong các mạng CNN, **mỗi lớp tích chập** sử dụng một bộ **trọng số** để **thực hiện việc tích** chập trên ảnh đầu vào và **tạo ra các đặc trưng mới.**

Khi **số lượng** lớp **tích chập** trong mạng **tăng** lên, **số lượng tham số cần tối ưu cũng tăng the**o, dẫn đến **khó khăn trong việc huấn luyệ**n và khả năng bị **overfitting**.

**Để giảm số lượng tham số cần tối ưu** trong mạng CNN, ta có thể sử dụng kỹ thuật **shared weight matrices depth**.

Theo cách này, **các lớp tích chập** trong mạng **chia sẻ cùng một bộ trọng số** để **thực hiện việc tích chập trên ảnh đầu vào.**

Khi áp dụng kỹ thuật này, **mỗi lớp tích chập chỉ sử dụng một phần của bộ trọng số được chia sẻ** và **phần còn lại** **được sử dụng bởi các lớp tích chập khác** có cùng shared weight matrices depth.

**Ví dụ**, nếu **shared weight matrices dep**th **là** 2, **các lớp tích chập** trong mạng CNN sẽ **chia sẻ cùng một bộ trọng số ở 2 lớp liên tiếp của mạng**.

Với cách thiết kế này, **số lượng tham số cần tối ưu** trong mạng sẽ **giảm đi** đáng kể, giúp **tăng tốc độ tính toán và giảm khả năng overfitting**. Tuy nhiên, việc thiết kế shared weight matrices depth cần được cân nhắc kỹ lưỡng để đảm bảo độ chính xác và hiệu suất của mạng.

**Ví dụ**, nếu chúng ta có một lớp tích chập với 32 bộ lọc và kích thước mỗi bộ lọc là 3x3, thì số lượng tham số sẽ là 32 x 3 x 3 = 288. Nếu chúng ta có thêm một lớp tích chập khác với số lượng bộ lọc và kích thước tương tự, nhưng ta sử dụng shared weight matrices depth để chia sẻ cùng một bộ trọng số giữa 2 lớp tích chập này, thì số lượng tham số cần tối ưu chỉ là 32 x 3 x 3 = 288, thay vì 2 x 288 = 576.

[21] **Resulting matrix size** là **kích thước ma trận kết quả** thu được **sau khi thực hiện** phép **tích chập** (convolution) giữa **ma trận đầu** **vào** (input matrix) và **ma trận trọng số** (weight matrix).

**Ma trận trọng số (weight matrix)** là một ma trận số được sử dụng trong các mô hình máy học để **biểu diễn mối quan hệ** giữa các **đặc trưng của dữ liệu đầu vào** **và đầu ra** tương ứng.

Trong các mô hình **neural netwo**rk, ma trận trọng số được **sử dụng để tính toán trọng số và lỗi**, cũng như để **thực hiện các phép toán trên dữ liệu đầu vào để tạo ra dữ liệu đầu ra**. Trong **quá trình huấn luyện mô hình**, các g**iá trị trong ma trận trọng số được** cập nhật dựa trên **độ lỗi** và các **phương pháp tối ưu hóa** để tìm ra các giá trị tối ưu nhất cho mô hình.

[22] **Sliding step size** (kích thước bước trượt) là kích thước **khoảng cách giữa các vị trí khung hình con (sub-image) trên ảnh** khi thực hiện sliding window (cửa sổ trượt) trong các mô hình xử lý ảnh.

**Khi thực hiện sliding window**, **cửa sổ trượt** **sẽ được di chuyển qua từng vị trí trên ảnh với một khoảng cách bằng sliding step size**, tạo ra các khung hình con (sub-image) tương ứng để xử lý.

Giá trị sliding step size sẽ **ảnh hưởng đến kích thước của các khung hình con được tạo ra và ảnh hưởng đến hiệu quả và tốc độ xử lý của mô hình.**

**[23] Nội dung trên toàn mạng (global content)** là t**hông tin tổng quát về toàn bộ dữ liệu trong mạng** hoặc bức ảnh.

Nó được sử dụng để **thể hiện những đặc trưng chung của dữ liệu** và **giúp** cho **mô hình** có thể **nhận diện và phân loại các đối tượng khác nhau trong mạng**.

Trong mô hình xử lý ngôn ngữ tự nhiên, nội dung trên toàn mạng **thường được biểu diễn dưới dạng một ma trận có kích thước nhỏ hơn so với dữ liệu đầu vào ban đầu**, bằng cách áp dụng các kỹ thuật như **pooling** hoặc **convolution**. Nó cũng được **sử dụng trong các mô hình xử lý ảnh** để giúp mô hình có thể **nhận biết** và **phân loại** các **đối tượng trên toàn bộ bức ảnh**.

[24] **Hàm mất mát (loss function)** là một hàm số **đo lường sự khác biệt giữa kết quả dự đoán của mô hình với giá trị đích** (ground truth) mong đợi. Hàm mất mát thường **được sử dụng để tối ưu các tham số trong mô hình**.

**Trong quá trình huấn luyện**, mô hình sẽ **điều chỉnh** các **tham số** của mình **để giảm thiểu giá trị của hàm mất mát**, từ đó **cải thiện khả năng dự đoán của mô hình** trên tập dữ liệu huấn luyện và trên các tập dữ liệu mới.

Các loại hàm mất mát phổ biến trong machine learning bao gồm: mean squared error (MSE), mean absolute error (MAE), cross-entropy loss, và hinge loss. Các loại hàm mất mát này được sử dụng tùy theo loại bài toán và mục đích của mô hình.

[25] **Trọng số của đầu vào (input weights)** là các **tham số trong mạng nơ-ron** sử dụng để **biến đổi các đặc trưng đầu vào** thành **các đặc trưng ở các lớp tiếp theo**. **Trong quá trình huấn luyện**, các **trọng số của đầu vào** sẽ được **điều chỉnh để tối ưu hoá kết quả dự đoán của mô hình**

**[26] Compactness** (tính gọn) **đo** lường **độ phân tán của các điểm dữ liệu** trong một nhóm. Các đặc trưng **có tính gọn cao** sẽ có **các điểm dữ liệu gần nhau hơn**, tạo ra một **phân bố đồng đều** và **giảm thiểu số lượng các đặc trưng lặp lại**. Compactness giúp **giảm thiểu sự phụ thuộc của các đặc trưng vào các đặc trưng khác**, từ đó **giảm thiểu sự quá khớp** và **tăng khả năng tổng quát hóa của mô hình**.

**Descriptiveness** (tính mô tả) đo lường **khả năng của các đặc trưng** để **phân biệt** các **điểm dữ liệu khác nhau**. Các đặc trưng **có tính mô tả cao** sẽ có **giá trị khác biệt lớn** **giữa** **các điểm dữ liệu** **khác nhau** và tạo ra một phân bố phân tán. Tính mô tả **giúp tăng độ chính xác và độ tin cậy của mô hình**.

Tổng quát lại, **các đặc trưng tốt** là các đặc trưng **có tính gọn cao** và **tính mô tả cao**, **giúp** **trích xuất được các thông tin quan trọng từ dữ liệu** và **tăng tính tổng quát hóa của mô hình.**

**[27] Feature matrix** là một ma trận **được tạo ra** bằng cách **sắp xếp các đặc trưng của các mẫu dữ liệu** trong các hàng và **các thuộc tính** của **các đặc trưng** trong các cột.

Feature matrix là **đầu vào cho các thuật toán học máy** và **được sử dụng để huấn luyện** và **kiểm tra các mô hình học máy**

**[28]** Trong mạng neuron, **output layer** là **lớp cuối cùng** của mạng được **sử dụng để tính toán đầu ra** của mô hình cho một đầu vào cụ thể.

**Khi các đầu vào được đưa vào mạng**, chúng **được xử lý thông qua** các **lớp trung gian để trích xuất đặc trưng** và cuối cùng đến **output layer để tính toán đầu ra**.

**Ví dụ**, trong **bài toán phân loại hình ảnh**, **output layer** có thể được thiết kế với **số lượng neuron** bằng **với số lượng lớp hình ảnh** cần **phân loại**.

**Mỗi neuron** trong output layer sẽ **đại diện cho một lớp cụ thể**, và **giá trị đầu ra** của neuron đó sẽ cho biết **xác suất của hình ảnh đó thuộc về lớp tương ứng**.

[28] **Lớp phân loại được xác định của hình ảnh đầu** vào là **nhãn** hoặc **lớp** mà **hình ảnh đó được phân loại vào dựa trên các đặc trưng của nó.**

**Ví dụ**, nếu chúng ta đang **xây dựng một mô hình phân loại hình ảnh** để phân loại ảnh là **mèo** hoặc **chó**, thì lớp phân loại của hình ảnh đầu vào sẽ là "**mèo**" hoặc "**chó**".

[29] **Activation function** (hàm kích hoạt) là **một hàm được áp dụng trên output** của một neuron **để đưa ra output của layer** đó.

Cụ thể, nó **tính toán tổng trọng số đầu vào** của neuron và **áp dụng một phép tính toán phi tuyến để đưa ra output của neuron đó**.

Điều này **cho phép mô hình học** **các quan hệ phi tuyến** **trong dữ liệu** và là một phần quan trọng trong **thiết kế của các mạng neural.** Các hàm kích hoạt phổ biến bao gồm Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax, vv.

[30] **Fully connected layer** (hay còn gọi là **dense layer**) là một loại layer trong mạng neuron nhân tạo, trong đó **mỗi neuron** của layer này **được kết nối với tất cả các neuron của layer trước đó** (nếu có).

Nó được gọi là "**fully connected**" vì các neuron ở layer này có kết nối với tất cả các neuron ở layer trước đó.

Các trọng số của kết nối giữa các **neuron trong fully connected** layer được học trong **quá trình huấn luyện mô hình** để **tối ưu hóa hàm mất mát**.

Fully connected layer thường được sử dụng trong các bài toán phân loại và dự đoán.

[31] **Label function** là một **hàm** số **trong machine learning** được **sử dụng để gán một nhãn (label)** cho **một mẫu dữ liệu**.

Nó là một **phần quan trọng** của các **thuật toán supervised learning**, **nơi dữ liệu huấn luyện đã được gán nhãn trước đó** và **mục tiêu của mô hình** là **học cách dự đoán đúng nhãn** cho **các dữ liệu mới dựa trên những gì đã học** được từ dữ liệu huấn luyện.

[32] Trong Machine Learning, **weight function** (hàm trọng số) là một hàm số **được sử dụng** để **tính toán các trọng số** (**weights**) **cho các kết nối giữa các neuron** trong mạng neural.

Các trọng số này **đóng vai trò quan trọng** trong việc **học và dự đoán đầu ra** của mô hình.

**Trọng số càng chính xác** thì **kết quả dự đoán của mô hình sẽ càng tốt.**

**[33] "time series"** được sử dụng để **mô tả dữ liệu được thu thập từ các thiết bị IoT** theo thời gian.

Cụ thể, các thông số như **nhiệt độ, độ ẩm, áp suất, độ rung** **của** các **thiết bị** **IoT** được đo lường và ghi lại liên tục theo thời gian để tạo thành chuỗi dữ liệu thời gian.

Chuỗi dữ liệu thời gian này đư**ợc sử dụng để phát hiện các hành vi xâm nhập bằng cách phân tích** và **so sánh các mẫu dữ liệu theo thời gian**.

[34] **Kỹ thuật tăng cường dữ liệu** (data augmentation) là phương pháp **tạo ra thêm dữ liệu huấn luyện mới** bằng cách **biến đổi dữ liệu huấn luyện ban đầu** mà **không làm thay đổi nhãn** tương ứng. Việc tăng cường dữ liệu giúp tăng tính đa dạng của dữ liệu huấn luyện và **giảm hiện tượng** quá khớp (**overfitting**) của mô hình. Các kỹ thuật tăng cường dữ liệu phổ biến trong Deep Learning bao gồm:

1.Xoay ảnh: xoay ảnh một góc nhỏ để tạo ra dữ liệu mới.

2.Dịch ảnh: dịch ảnh một khoảng cách nhỏ để tạo ra dữ liệu mới.

3. Phóng to và thu nhỏ ảnh: thay đổi kích thước ảnh để tạo ra dữ liệu mới.

4.Lật ảnh: lật ảnh theo trục dọc hoặc trục ngang để tạo ra dữ liệu mới.

5.Thêm nhiễu vào ảnh: thêm nhiễu Gauss hoặc Salt-and-Pepper vào ảnh để tạo ra dữ liệu mới. Cắt ảnh: cắt một phần của ảnh để tạo ra dữ liệu mới. Các kỹ thuật tăng cường dữ liệu có thể được áp dụng đối với nhiều loại dữ liệu, chẳng hạn như ảnh, văn bản, âm thanh và video

[35] **Tăng độ đa dạng của dữ liệu đầu vào** (data augmentation) là kỹ thuật **tạo ra các phiên bản mới của dữ liệu** **huấn luyện** bằng cách **thay đổi** hoặc **biến đổi** các thông số như **độ sáng, độ nghiêng, độ xoay, độ phóng to/thu nhỏ, vị trí, v.v.** của các ảnh hoặc dữ liệu khác.

[36] **Siêu tham số** (**hyperparameters**) của mô hình là các **tham số** **được học thông qua quá trình tối ưu hóa trong quá trình huấn luyện**. Chúng ta phải **tự thiết lập** và **điều chỉnh các giá trị** của siêu tham số này trước khi bắt đầu huấn luyện **để cải thiện hiệu suất của mô hình**

**VD:** tốc độ học (learning rate), số lượng epoch (number of epochs), số lượng layer, số lượng hidden unit, hệ số độ lớn giảm trọng số (weight decay), dropout, kernel size (đối với CNN) và các siêu tham số khác tùy thuộc vào loại mô hình**.**

**[37]** Các phương pháp tiền xử lý dữ liệu mới bao gồm:

1.**Augmentation data** (Tăng cường dữ liệu): Sử dụng các kỹ thuật như xoay, lật, tăng giảm độ sáng để tạo thêm dữ liệu mới từ dữ liệu hiện có.

2.**Data cleaning** (Làm sạch dữ liệu): Loại bỏ dữ liệu không chính xác, không đầy đủ hoặc không cần thiết khỏi tập dữ liệu để tăng độ chính xác cho mô hình.

3.**Feature engineering** (Kỹ thuật xây dựng đặc trưng): Tạo ra các đặc trưng mới từ dữ liệu đầu vào nhằm tăng độ chính xác và giảm kích thước của tập dữ liệu.

4.**Dimensionality reduction** (Giảm chiều dữ liệu): Giảm số lượng các biến đầu vào để tăng tốc độ xử lý và giảm độ phức tạp cho mô hình.

5.**Data normalization** (Chuẩn hóa dữ liệu): Đưa các giá trị trong tập dữ liệu về cùng một phạm vi để giảm thiểu ảnh hưởng của các giá trị ngoại lệ (outliers) trên mô hình. (độ F -> Độ C , m/s -> km/h,…)

6.**Data encoding** (Mã hóa dữ liệu): Chuyển đổi dữ liệu từ dạng văn bản hoặc chuỗi thành dạng số để có thể áp dụng vào các thuật toán học máy.

7.**Imputation** (Điền giá trị thiếu): Điền các giá trị bị thiếu trong tập dữ liệu để giảm thiểu ảnh hưởng của các giá trị thiếu trên mô hình

[38] **Kỹ thuật Ensemble** là một phương pháp **kết hợp các mô hình dự đoán khác nhau** để **tạo ra** một **mô hình dự đoán tốt hơn**. Kỹ thuật này **tận dụng sự khác biệt giữa các mô hình** để tạo ra một mô hình dự đoán tốt hơn, **giảm thiểu các sai sót** của một mô hình duy nhất.

Các phương pháp Ensemble phổ biến bao gồm:

1. **Voting** (Bầu chọn): **kết hợp dự đoán** của các mô hình để **đưa ra quyết định cuối cùng**. Có thể sử dụng phương pháp đa số hoặc trọng số để bầu chọn.
2. **Bagging** (Bootstrap Aggregating): **xây dựng nhiều mô hình dự đoán khác nhau** trên các tập dữ liệu con được lấy mẫu ngẫu nhiên từ tập dữ liệu gốc.
   1. **Kết quả dự đoán cuối cùng** được tính bằng cách **lấy trung bình** hoặc bầu chọn của **các dự đoán đó**.
3. **Boosting**: **xây** dựng các mô hình dự đoán theo chuỗi, mỗi **mô hình sau** sẽ cố gắng **cải thiện sai số của mô hình trước** đó.
   1. **Kết quả dự đoán** cuối cùng được tính bằng cách **tổng hợp dự đoán** của tất cả các mô hình.

(mỗi mô hình có trọng số khác nhau , dự đoán dựa vào tất cả trọng số để có output)

1. **Stacking**: xây dựng **các mô hình dự đoán khác nhau** và sử dụng **kết quả dự đoán** **của** các **mô hình này** làm **đầu vào cho một mô hình cuối cùng**. Mô hình cuối cùng sẽ học cách **kết hợp các dự đoán** này để **đưa ra kết quả cuối cùng.**

Kỹ thuật Ensemble được sử dụng rộng rãi trong nhiều lĩnh vực như machine learning, computer vision, natural language processing, v.v. để cải thiện độ chính xác và độ ổn định của các mô hình dự đoán.

**[38] Bias** và **variance** là hai khái niệm quan trọng trong machine learning để đánh giá và tối ưu hóa hiệu suất của mô hình.

- **Bias**: là **sai lệch** giữa giá trị **dự đoán** trung bình **của mô hình** và **giá trị thực tế**. Nó cho biết mô hình có độ chính xác cao hay thấp trên toàn bộ tập dữ liệu. Một mô hình với bias cao thường có kết quả dự đoán rất khác so với giá trị thực tế.

-**Variance**: là **độ phân tán** của **các dự đoán** từ mô hình. Nó **cho biết mô hình** có **khả năng tổng quát hóa cao hay thấp** trên các tập dữ liệu khác nhau.

Một mô hình với **variance cao** thường có **kết quả dự đoán rất khác nhau trên các tập dữ liệu khác nhau.**

* **Một mô hình hoàn hảo** là mô hình có **bias thấp** và **variance thấp**. Tuy nhiên, thường không thể có được mô hình hoàn hảo như vậy, nên ta phải tìm cách cân bằng giữa hai yếu tố này. Nếu ta giảm bias, thì có thể tăng variance và ngược lại.