1 Convolutional Neural Networks CNN

3 Auto-Encoder AE, VAE

7 RNN-LSTM

1 3 7

- Tổng quan (nó là gì)

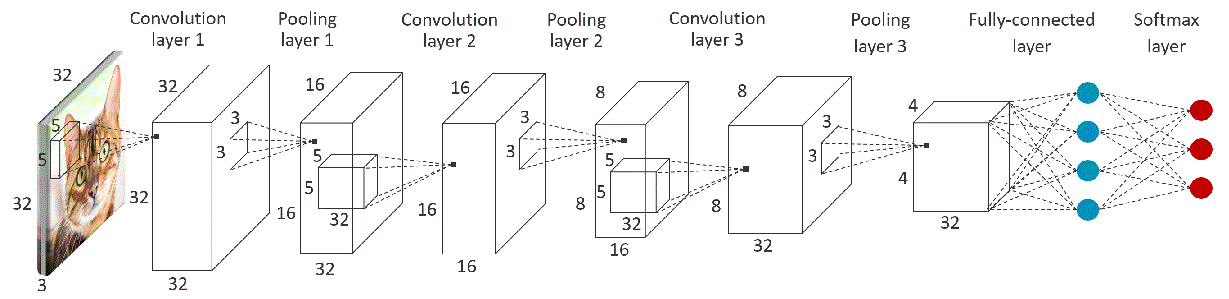
- Có những loại nào, ứng dụng gì

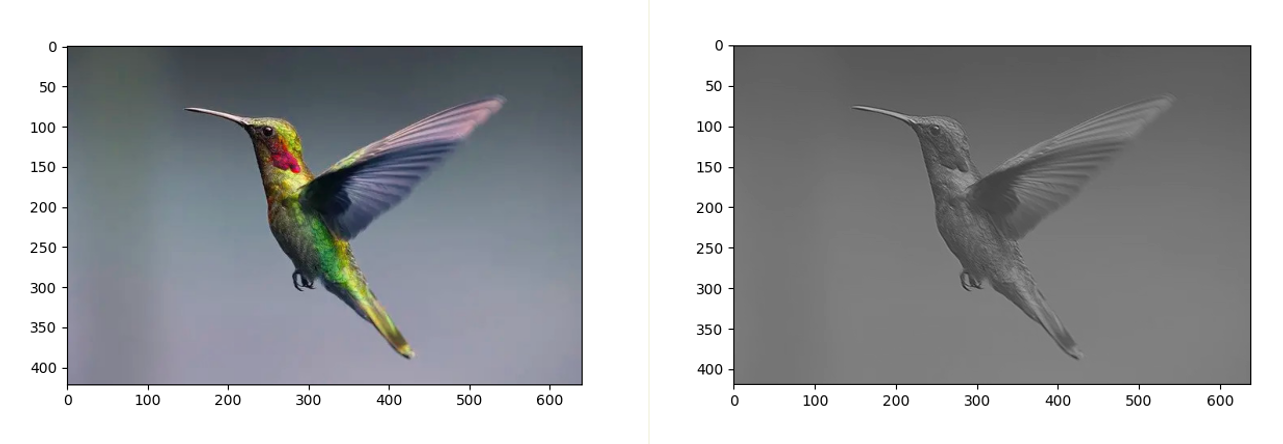
- So sánh, đánh giá

Tổng quan: CNN là một loại mạng thần kinh thường được sử dụng trong Computer Vision để xử lý hình ảnh. Chúng sử dụng các lớp tích chập để trích xuất thông tin từ hình ảnh và thường được sử dụng cho nhiệm vụ như nhận diện vật thể, phân loại ảnh, phát hiện đường biên, và nhiều ứng dụng khác.

**Hàm phi tuyến - ReLU**

* ReLU viết tắt của Rectified Linear Unit, là 1 hàm phi tuyến. Với đầu ra là: ƒ (x) = max (0, x).
* Tại sao ReLU lại quan trọng: ReLU giới thiệu tính phi tuyến trong ConvNet. Vì dữ liệu trong thế giới mà chúng ta tìm hiểu là các giá trị tuyến tính không âm
* Có 1 số hà phi tuyến khác như tanh, sigmoid cũng có thể được sử dụng thay cho ReLU. Hầu hết người ta thường dùng ReLU vì nó có hiệu suất tốt.





ảnh bình thường sau khi qua kernel

Trích xuất những đặt điểm nổi bật của bức hình

Dưới input -> kernel -> feater map

Stride để giảm kích thước ma trận ngõ vào, để dễ dàng hơn trong việc huấn luyện

Paddling giữ lại kích thước của input. Khi các kernel quét qua thì feature map vẫn giữ nguyên kích thước ban đầu.

Stride giảm kích thước và padding giữ nguyên

| **Thuộc tính** | **Stripe** | **Padding** |
| --- | --- | --- |
| Định nghĩa | Stripe là khoảng cách giữa các vị trí mà kernel di chuyển qua ảnh hoặc feature map trong quá trình tích chập. | Padding là việc thêm các giá trị vào xung quanh biên của ảnh hoặc feature map trước khi áp dụng phép tích chập. |
| Loại | Stripe có thể là một giá trị nguyên dương và có thể là một chiều hoặc hai chiều (stripe theo chiều ngang và dọc). | Padding có hai loại chính: zero-padding (thêm giá trị 0) và valid-padding (không thêm giá trị). |
| Mục tiêu | Điều chỉnh cách kernel di chuyển qua feature map. | Duy trì kích thước của feature map sau phép tích chập. |
| Kích thước | Stripe có thể là 1 hoặc lớn hơn 1. Nếu stripe là 1, kernel di chuyển một pixel mỗi lần; nếu stripe lớn hơn 1, kernel bỏ qua nhiều pixel trước khi di chuyển. | Padding có thể được xác định theo chiều ngang và chiều dọc riêng lẻ. |
| Thông tin biên | Stripe không duy trì thông tin ở biên của feature map và có thể gây mất mát thông tin ở biên. | Padding duy trì thông tin ở biên của feature map và giữ lại thông tin ở biên. |
| Ứng dụng | Sử dụng để kiểm soát thông tin nhanh chóng giảm dần qua các lớp tích chập. | Sử dụng để duy trì thông tin biên và hình dáng của feature map. |

Fully connected layer

Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương đối các đặc điểm của ảnh (ví dụ mắt, mũi, khung mặt,…) thì tensor của output của layer cuối cùng, kích thước H\*W\*D, sẽ được chuyển về 1 vector kích thước (H\*W\*D)

Sau đó ta dùng các fully connected layer để kết hợp các đặc điểm của ảnh để ra được output của model.

### Loại của CNN:

**Convolutional Neural Network (CNN):** Kiến trúc cơ bản của CNN với các lớp tích chập, pooling và lớp kết nối đầy đủ. Được sử dụng rộng rãi trong xử lý ảnh và thị giác máy tính.

1. **LeNet:** Được phát triển bởi Yann LeCun và được sử dụng cho nhận dạng ký tự viết tay. LeNet là một trong những kiến trúc ban đầu của CNN.
2. **AlexNet:** Đây là một kiến trúc CNN đặc biệt với năm lớp tích chập và bốn lớp kết nối đầy đủ, đã đem lại sự thăng tiến lớn cho việc nhận dạng hình ảnh.
3. **VGGNet:** Kiến trúc với nhiều lớp tích chập và được sử dụng để giành chiến thắng trong các cuộc thi nhận dạng hình ảnh.
4. **GoogLeNet (Inception):** Kiến trúc mạng lưới sâu với các lớp tích chập song song (inception modules) để tăng cường hiệu suất.
5. **ResNet (Residual Network):** Kiến trúc dựa trên các khối "residual" để giảm tình trạng vanishing gradient và cho phép xây dựng các mạng rất sâu.

### Ứng dụng của CNN:

1. **Nhận dạng hình ảnh:** CNN được sử dụng rộng rãi cho các ứng dụng như nhận dạng khuôn mặt, phân loại đối tượng, nhận dạng biển số xe, dò tìm sản phẩm phổ biến trong cửa hàng, v.v.
2. **Xử lý video:** CNN cũng được sử dụng để theo dõi đối tượng trong video và phát hiện hành vi không bình thường.
3. **Xử lý văn bản trong hình ảnh:** CNN có thể được sử dụng để trích xuất văn bản từ hình ảnh hoặc biểu đồ, giúp tự động hóa công việc nhập liệu dựa trên hình ảnh.
4. **Xử lý âm thanh và tín hiệu:** CNN cũng có thể được áp dụng cho xử lý âm thanh và tín hiệu thời gian thực, như trong ứng dụng nhận dạng giọng nói và dự đoán tín hiệu thời tiết.
5. **Y tế:** CNN có thể được sử dụng để phát hiện và phân loại bệnh từ hình ảnh y tế, như phát hiện ung thư, viêm gan, và nhiều ứng dụng khác trong ngành y tế.
6. Ứng dụng: CNNs có ứng dụng rộng rãi trong Computer Vision, từ xe tự hành, phát hiện gương mặt, đến ứng dụng y tế như xử lý ảnh chụp cắt lớp và phân loại bệnh.

Autoencoder

Tổng quan: AE và VAE có thể được sử dụng trong Computer Vision để nén và biểu diễn dữ liệu hình ảnh. Chúng có thể giúp giảm chiều dữ liệu và tạo ra biểu diễn không gian ẩn cho hình ảnh.

Kiến trúc của Autoencoder bao gồm 3 phần chính: Encoder, Bottleneck, Decoder… Cụ thể như sau:

* **Encoder (bộ mã hóa):** Một module có chức năng nén dữ liệu đầu vào của bộ kiểm tra xác thực thành một biểu diễn được mã hóa. Thông thường nó sẽ nhỏ hơn một vài bậc so với dữ liệu đầu vào.
* [**Bottleneck**](https://bizflycloud.vn/tin-tuc/bottleneck-la-gi-20181119094602437.htm)**:** Một module chứa các biểu diễn tri thức đã được nén hay còn gọi là output của Encoder, đây là phần quan trọng nhất trong mạng vì nó mang đặc trưng của dữ liệu đầu vào, có thể sử dụng để lấy đặc trưng của ảnh, tái tạo hình ảnh…
* **Decoder (bộ giải mã):** Module hỗ trợ mạng giải nén các biểu diễn tri thức và tái tạo lại cấu trúc dữ liệu từ dạng mã hóa của nó. Mô hình học dựa theo việc so sánh đầu ra của Decoder với đầu vào bạn đầu của nó.
* **Encoder**

Encoder (bộ mã hóa) là một tập hợp bao gồm các khối tích hợp theo sau là các modules dùng để nén đầu vào của mô hình thành một phần nhỏ gọn được gọi là bottleneck.

Bottleneck là bộ giải mã bao gồm một loạt các upsampling modules hỗ trợ đưa tính năng nén trở lại dưới dạng hình ảnh. Trong trường hợp mã tự động đơn giải, đầu ra được mong đợi giống với đầu vào.

Ngoài ra, đối với các mã tự động mã hóa biến thể thì đầu ra sẽ cho ra một hình ảnh hoàn toàn mới được hình thành với thông tin mà mô hình đã được cung cấp làm dữ liệu đầu vào.

* **Bottleneck**

Đây là phần quan trọng nhất trong mạng neural và cũng là phần nhỏ nhất. Nó giúp hạn chế luồng thông tin đến bộ giải mã từ bộ mã hóa, vì thế, bottleneck chỉ cho phép thông tin quan trọng nhất đi qua.

Bottleneck được thiết kế theo cách mà thông tin tối đa mà một hình ảnh sở hữu. Nó giúp thiết lập biểu diễn tri thức của dữ liệu đầu vào.

Do đó, cấu trúc của bộ mã hóa-giải mã (encoder-decoder) sẽ giúp bạn trích xuất nhiều nhất thông tin từ một hình ảnh dưới dạng dữ liệu và thiết lập các mối tương quan hữu ích giữa những đầu vào khác nhau trong mạng.

Bottleneck như một biểu diễn nén của đầu vào, nó ngăn cản mạng neural ghi nhớ dữ liệu đầu vào và trang bị quá nhiều trên dữ liệu. Đặc biệt, bottleneck càng nhỏ thì nguy cơ quá tải càng thấp.

* **Decoder**

Decoder (bộ giải mã) là một tập hợp bao gồm các khối lấy mẫu và tích tụ để tái tạo lại đầu ra của bottleneck.

Do đầu vào cho bộ giải mã là một biểu diễn tri thức đã được nén nên bộ giải mã đóng vai trò như một "bộ giải nén" và thiết lập lại hình ảnh từ các thuộc tính tiềm ẩn của nó.

**Hyper-parameter**

1. **Code size (Kích thước mã):** Đây là số lượng đơn vị ẩn trong lớp mã hóa của autoencoder. Code size quyết định kích thước của biểu diễn nén của dữ liệu. Nó ảnh hưởng đến mức độ nén và khả năng trích xuất đặc trưng của mô hình.
2. **Number of layers (Số lớp):** Số lớp trong mô hình autoencoder. Mô hình có thể có một hoặc nhiều lớp ẩn. Số lớp này ảnh hưởng đến độ phức tạp của mô hình và khả năng học đặc trưng.
3. **Number of nodes per layer (Số đơn vị ẩn cho mỗi lớp):** Đây là số lượng đơn vị (neurons) trong mỗi lớp ẩn của autoencoder. Số đơn vị này quyết định khả năng học và biểu diễn của mô hình. Một số đơn vị nhiều hơn có thể làm cho mô hình mạnh mẽ hơn, nhưng cũng có thể gây overfitting.
4. **Reconstruction Loss (Hàm mất mát tái tạo):** Hàm mất mát quyết định cách mô hình tính toán sai số giữa dữ liệu đầu vào và đầu ra của autoencoder. Các loại hàm mất mát phổ biến bao gồm Mean Squared Error (MSE) cho dữ liệu số và Binary Cross-Entropy cho dữ liệu nhị phân.

Có **4 hyperparameters** cần quan tâm trước khi training một mô hình Autoencoder:

* **Code size**: kích thước của Bottleneck là 1 hyperparameter rất quan trọng được sử dụng mà chúng ta cần lưu ý. Kích thước Bottleneck quyết định lượng thông tin được nén. “Nhiều quá không tốt mà ít quá cũng không ổn”
* **Number of layers**: giống với hầu hết các mạng neural, một hyperparameter quan trọng để điều chỉnh độ sâu của encoder và decoder trong Autoencoder, càng sâu thì mô hình càng phức tạp, nhưng càng nông thì mô hình chạy càng nhanh, càng light weights
* **Number of nodes per layer**: Số lượng nodes trên 1 layer quyết định số weights ra sẽ sử dụng trên từng layer. Thông thường, số lượng nút này giảm dần theo mỗi lớp tiếp theo bởi đầu vào của lớp này là đầu ra của lớp trước đó, và đầu vào này thì dần trở nên nhỏ hơn trên các lớp
* **Reconstruction Loss**: Loss function là một thức không thể thiếu trong mạng neural. Hàm loss này sẽ phụ thuộc vào kiểu input và oupt của mô hình chúng ta muốn đáp ứng. Ví dụ với việc xử lý ảnh, các hàm loss thông thường được ưa chuộng là Mean Square Error (MSE) và L1 Loss. Còn với một số trường hợp ảnh nhị phân (MNIST), chúng ta có thể sử dụng Binary Cross Entropy sẽ tốt hơn.

### Loại của Autoencoder:

1. **Vanilla Autoencoder:** Đây là kiểu cơ bản của autoencoder với một lớp mã hóa và một lớp giải mã. Nó được sử dụng để giảm chiều dữ liệu và trích xuất đặc trưng.
2. **Sparse Autoencoder:** Autoencoder thưa (sparse autoencoder) được thiết kế để có ít đơn vị ẩn hoạt động. Nó có thể được sử dụng để học các biểu diễn đặc trưng sâu và thưa.
3. **Denoising Autoencoder:** Autoencoder làm sạch dữ liệu (denoising autoencoder) thêm nhiễu vào dữ liệu đầu vào và học để loại bỏ nhiễu. Nó được sử dụng cho tiền xử lý dữ liệu và phân loại.
4. **Variational Autoencoder (VAE):** VAE là một kiểu autoencoder sử dụng cho khả năng sinh dữ liệu mới. Nó có thể tạo ra dữ liệu mới trong một không gian ẩn.
5. **Contractive Autoencoder:** Loại autoencoder này bao gồm một thành phần biểu diễn học sâu và biểu diễn thuộc tính. Nó được sử dụng để học biểu diễn bền vững của dữ liệu.

### Ứng dụng của Autoencoder:

1. **Trích xuất đặc trưng:** Autoencoder được sử dụng để trích xuất đặc trưng quan trọng từ dữ liệu, chẳng hạn như ảnh hoặc văn bản.
2. **Giảm chiều dữ liệu:** Autoencoder được sử dụng để giảm chiều dữ liệu, giúp tăng tốc quá trình huấn luyện và giảm nhiễu.
3. **Tiền xử lý dữ liệu:** Đặc biệt là denoising autoencoder, autoencoder được sử dụng để làm sạch dữ liệu bằng cách loại bỏ nhiễu.
4. **Nén dữ liệu:** Trong truyền thông và lưu trữ, autoencoder có thể được sử dụng để nén dữ liệu mà không mất mất thông tin quan trọng.
5. Ứng dụng: AE và VAE có ứng dụng trong nhiều tác vụ của Computer Vision, như nén hình ảnh, tạo hình ảnh gốc từ biểu diễn không gian ẩn, và tạo ra biến thể của hình ảnh.

**RNN-LSTM trong Computer Vision:**

LSTM (Long Short-Term Memory) khắc phục vấn đề biến mất gradient trong mạng RNN và có khả năng xử lý chuỗi dài hơn.

RNN (Recurrent Neural Network) đơn giản hơn và thích hợp cho các tác vụ xử lý chuỗi ngắn.

LSTM thường được ưa chuộng trong nhiều ứng dụng thực tế, trong khi RNN thường được sử dụng cho các tác vụ đơn giản hơn.

Tổng quan: RNN-LSTM có thể được sử dụng trong Computer Vision cho việc xử lý chuỗi hình ảnh hoặc video. Chúng có khả năng lưu trạng thái và giúp máy tính hiểu ngữ cảnh trong dữ liệu thời gian.

Ứng dụng: RNN-LSTM có ứng dụng trong việc xử lý video, nhận diện hoạt động, và theo dõi đối tượng trong thời gian thực.

So sánh và đánh giá:

CNN thường là mô hình chính trong Computer Vision, chủ yếu được sử dụng cho xử lý hình ảnh và ứng dụng như phân loại và nhận diện vật thể.

AE và VAE có thể được sử dụng để nén và biểu diễn hình ảnh, tạo ra biểu diễn không gian ẩn. Chúng thường được sử dụng trong các ứng dụng đòi hỏi giảm chiều dữ liệu hoặc tạo ra biến thể của dữ liệu.

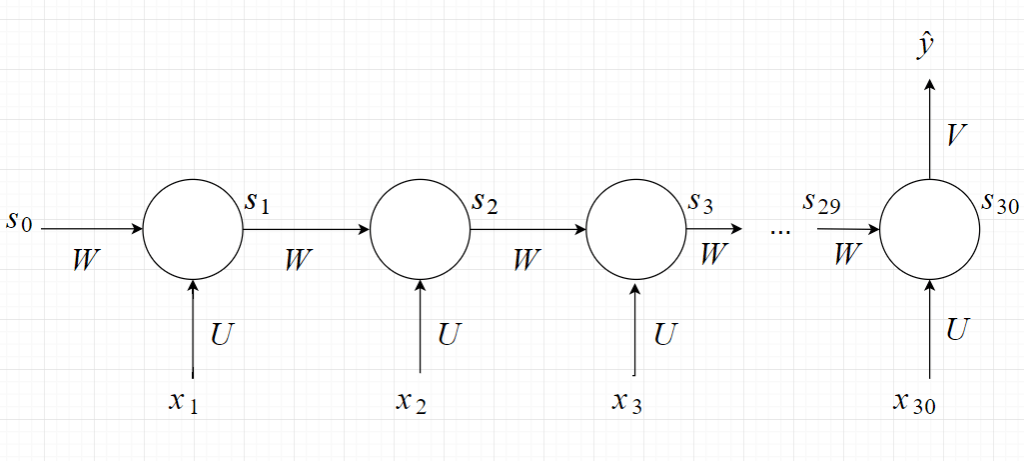
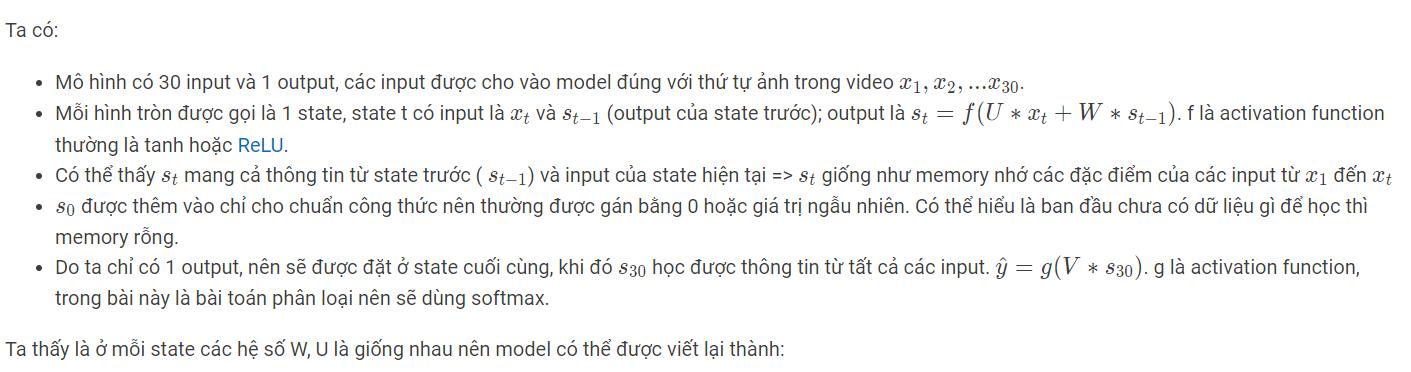
RNN-LSTM thường được sử dụng khi cần xử lý dữ liệu chuỗi thời gian hoặc trong các tình huống đòi hỏi lưu trạng thái, như việc theo dõi đối tượng trong video.

Mô hình RNN

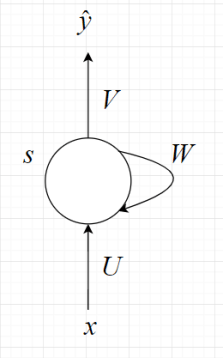
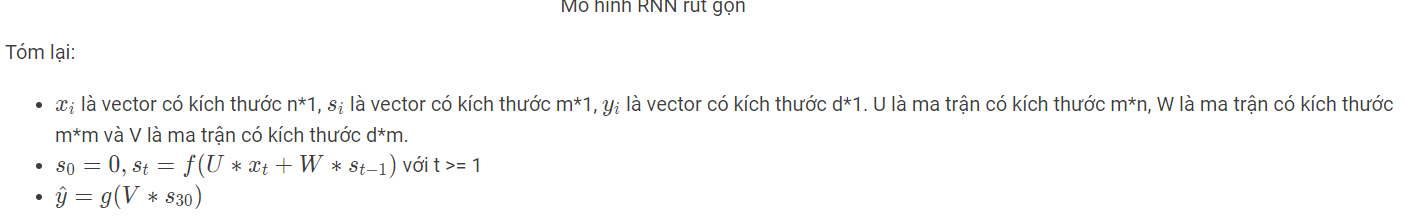
Bài toán: Nhận diện hành động trong video 30s. Đây là dạng bài toán many to one trong RNN, tức nhiều input và 1 output.

Input ta sẽ tách video thành 30 ảnh ở mỗi giây. Các ảnh sẽ được cho qua model CNN để lấy ra các feature ([feature extraction](https://nttuan8.com/bai-9-transfer-learning-va-data-augmentation/#Feature_extractor)) thành các vector có kích thước n\*1. Vector tương ứng với ảnh ở giây thứ i là xi

Output là vector có kích thước d\*1, softmax function được sử dụng như trong [bài phân loại ảnh](https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/#Xay_dung_model).

Mô hình RNN cho bài toán.

Ta có



Biến thể của RNN:

**Vanilla RNN**: RNN cơ bản với các ô nhớ đơn giản, thường gặp vấn đề biến mất gradient khi xử lý chuỗi dài.

**Gated Recurrent Unit (GRU):** Tương tự như LSTM nhưng có mô trình đơn giản hơn, giữ lại một phần nhớ trạng thái cũ hơn và không có cổng quên riêng biệt.

Biến thể của LSTM:

**Bidirectional LSTM**: Sử dụng cả hai hướng chuỗi (trái sang phải và phải sang trái) để tăng khả năng hiểu và dự đoán chuỗi.

**Stacked LSTM:** Nối nhiều tầng LSTM lại với nhau để tăng khả năng học các mức độ phức tạp của dữ liệu.

**Peephole LSTM**: Mở rộng ý tưởng của LSTM bằng cách cho phép cổng quên và cổng đầu vào "nhìn thấy" trực tiếp trạng thái ẩn trước đó.

**Coupled LSTM (C-LSTM):** Kết hợp các cổng đầu vào và cổng quên để giảm độ phức tạp của mô hình.

1. **Nhận Diện và Theo Dõi Đối Tượng (Object Detection and Tracking):**
   * LSTM và RNN có thể được sử dụng để theo dõi và nhận diện đối tượng trong video. Các mô hình như YOLO (You Only Look Once) kết hợp RNN để theo dõi các đối tượng qua các khung hình.
2. **Nhận Diện Hành Động (Action Recognition):**
3. LSTM và RNN có thể phân loại **Nhận Diện và Theo Dõi Đối Tượng (Object Detection and Tracking):**
   * LSTM và RNN có thể được sử dụng để theo dõi và nhận diện đối tượng trong video. Các mô hình như YOLO (You Only Look Once) kết hợp RNN để theo dõi các đối tượng qua các khung hình.
4. **Nhận Diện Hành Động (Action Recognition):**
   * LSTM và RNN có thể phân loại và nhận diện các hành động trong video, như nhận diện động tác trong thể thao hoặc theo dõi hành động của đối tượng.
5. **Dự Đoán Hành Động Tiếp Theo (Next Action Prediction):**
   * LSTM có thể được sử dụng để dự đoán hành động tiếp theo của đối tượng trong video, giúp mô hình hiểu và dự đoán luồng hành động.
6. **Phân Loại Video và Phân Tích Phát Hiện Đột Ngột (Video Classification and Anomaly Detection):**
   * LSTM có thể được sử dụng để phân loại nội dung video, cũng như phát hiện các sự kiện không bình thường hoặc đột ngột.
7. **Tổng Hợp Mô Hình Dựa Trên Thời Gian (Temporal Modeling):**
   * LSTM có thể được tích hợp vào mô hình để mô phỏng thông tin không gian và thời gian đồng thời trong quá trình xử lý hình ảnh, đặc biệt hữu ích trong các tình huống đòi hỏi việc theo dõi đối tượng qua thời gian.
8. **Tăng Cường Học (Reinforcement Learning):**
   * LSTM và RNN có thể được sử dụng trong các ứng dụng tăng cường học, nơi mô hình cần theo dõi các hành động và trạng thái qua thời gian.
9. **Nhận Diện Vùng Quan Trọng (Attention Mechanisms):**
   * Mô hình sử dụng LSTM cũng có thể sử dụng cơ chế chú ý để nhấn mạnh vùng quan trọng trong hình ảnh hoặc video.
10. **Nhận Diện Bối Cảnh (Scene Understanding):**
    * LSTM có thể giúp trong việc hiểu bối cảnh trong các tác vụ như tự động lái xe, nơi mô hình cần hiểu và dự đoán hành vi của các đối tượng xung quanh qua thời gian.
    * và nhận diện các hành động trong video, như nhận diện động tác trong thể thao hoặc theo dõi hành động của đối tượng.
11. **Dự Đoán Hành Động Tiếp Theo (Next Action Prediction):**
    * LSTM có thể được sử dụng để dự đoán hành động tiếp theo của đối tượng trong video, giúp mô hình hiểu và dự đoán luồng hành động.
12. **Phân Loại Video và Phân Tích Phát Hiện Đột Ngột (Video Classification and Anomaly Detection):**
    * LSTM có thể được sử dụng để phân loại nội dung video, cũng như phát hiện các sự kiện không bình thường hoặc đột ngột.
13. **Tổng Hợp Mô Hình Dựa Trên Thời Gian (Temporal Modeling):**
    * LSTM có thể được tích hợp vào mô hình để mô phỏng thông tin không gian và thời gian đồng thời trong quá trình xử lý hình ảnh, đặc biệt hữu ích trong các tình huống đòi hỏi việc theo dõi đối tượng qua thời gian.
14. **Tăng Cường Học (Reinforcement Learning):**
    * LSTM và RNN có thể được sử dụng trong các ứng dụng tăng cường học, nơi mô hình cần theo dõi các hành động và trạng thái qua thời gian.
15. **Nhận Diện Vùng Quan Trọng (Attention Mechanisms):**
    * Mô hình sử dụng LSTM cũng có thể sử dụng cơ chế chú ý để nhấn mạnh vùng quan trọng trong hình ảnh hoặc video.
16. **Nhận Diện Bối Cảnh (Scene Understanding):**
    * LSTM có thể giúp trong việc hiểu bối cảnh trong các tác vụ như tự động lái xe, nơi mô hình cần hiểu và dự đoán hành vi của các đối tượng xung quanh qua thời gian.