TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI PHÂN HIỆU TẠI TP.HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN HỌC
HỌC MÁY CƠ BẢN

Tìm hiểu các thuật toán: GAN, LSTM, RNN, CNN, ANN

Họ tên: Võ Thành Hoàng Phúc
MSSV: 6351071056
Lớp: CNTT-K63

Nội dung chính

1. Thuật toán GAN - Generative Adversarial Network

2. Thuật toán LSTM - Long Short-Term Memory

3. Thuật toán RNN - Recurrent Neural Network

4. Thuật toán CNN - Convolutional Neural Network

5. Thuật toán ANN - Artificial Neural Network

1. Thuật toán GAN - Generative Adversarial Network

1.1. Giới thiệu

1.2. Cấu trúc của mạng GAN

- Generator (trình tạo)
- Discriminator (trình so sánh)

1.3. Cơ chế hoạt động của GAN

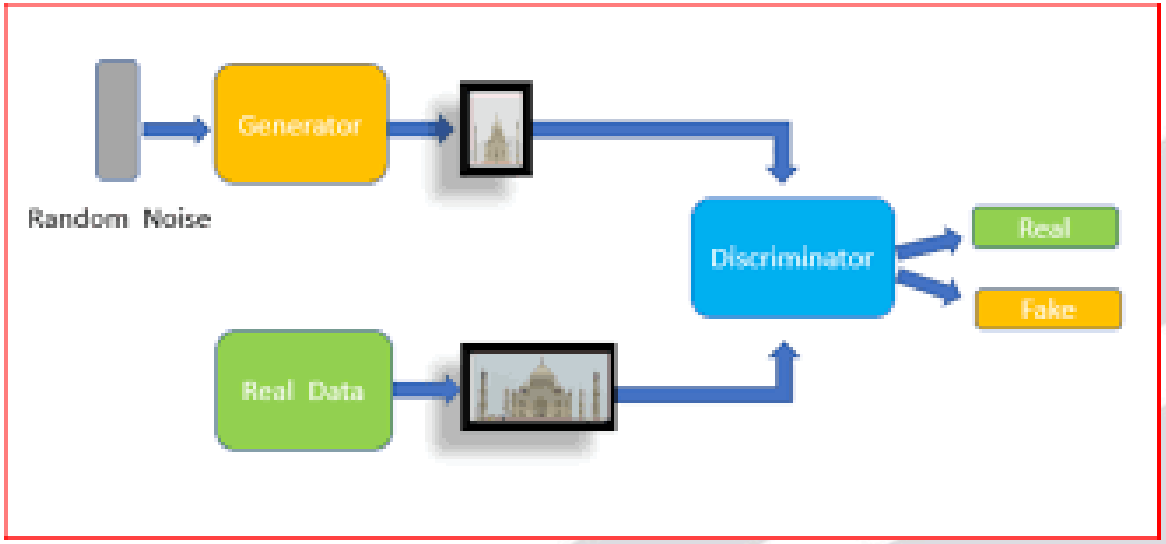
1.4. Huấn luyện, kết quả và đánh giá thuật toán

1.5. Ưu và nhược điểm

1.1 Giới thiệu

- GAN (Generative Adversarial Network) – Mạng đối sinh – là một mô hình học sâu (Deep Learning)

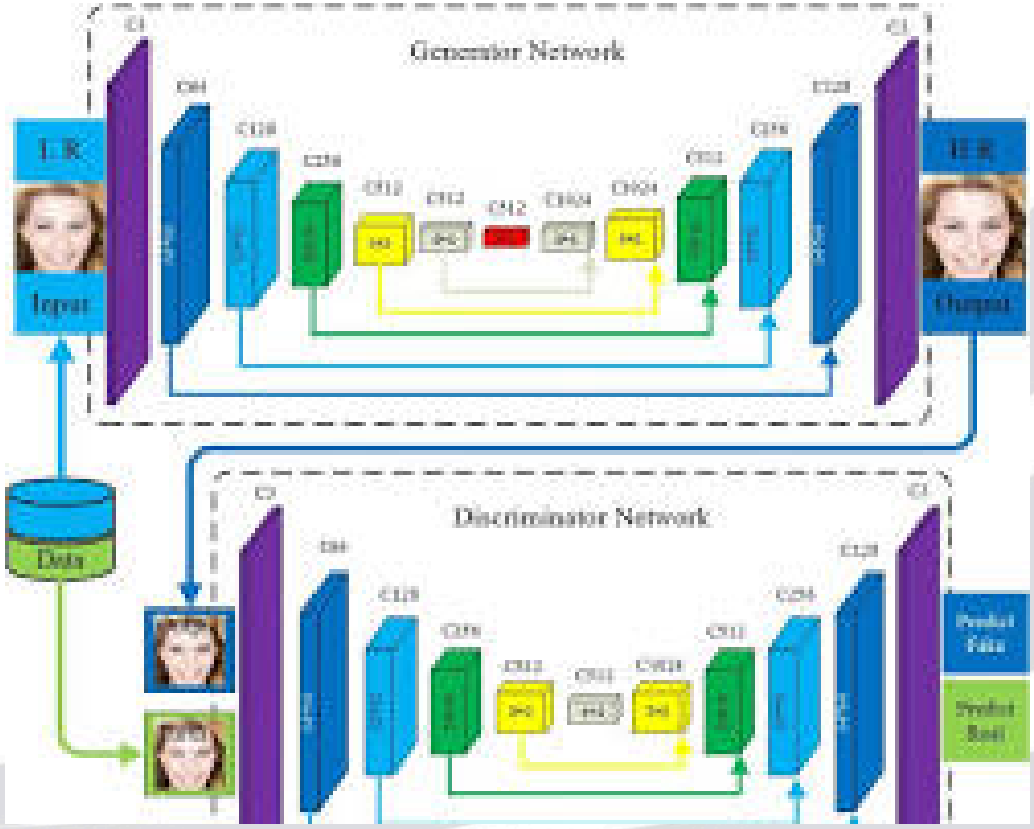
- Mục tiêu chính của GAN là tạo ra dữ liệu mới có phân phối tương tự dữ liệu thật



1.2 Cấu trúc của mạng GAN

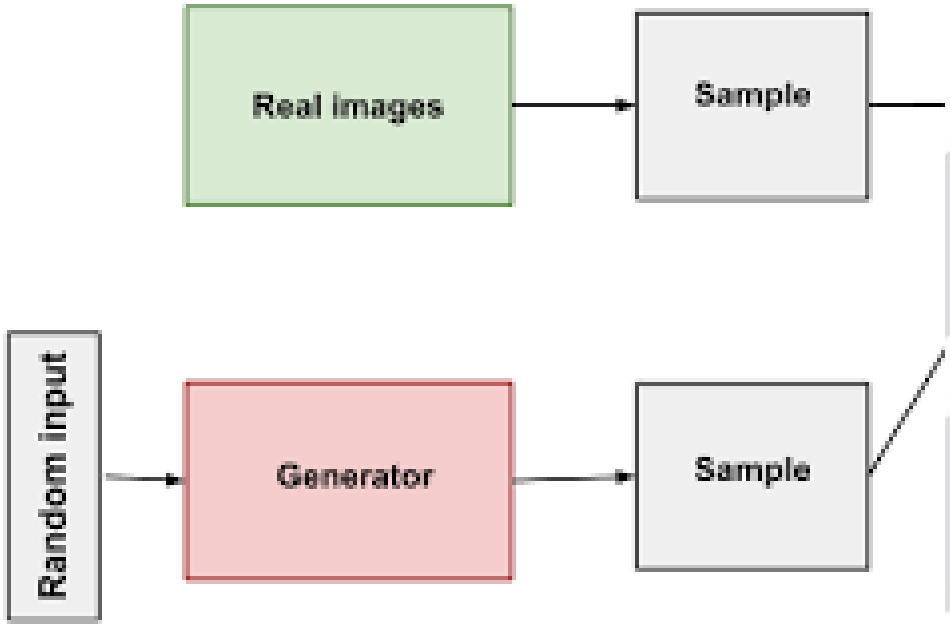
Generator (Trình tạo)

- Mục đích: Tạo ra dữ liệu giả (ví dụ: hình ảnh, âm thanh) càng giống với dữ liệu thật càng tốt.
- Đầu vào: Một vector nhiễu ngẫu nhiên (thường là nhiễu Gaussian).
- Đầu ra: Dữ liệu giả.
- Mục tiêu: "Đánh lừa" Discriminator tin rằng dữ liệu mình tạo ra là thật.



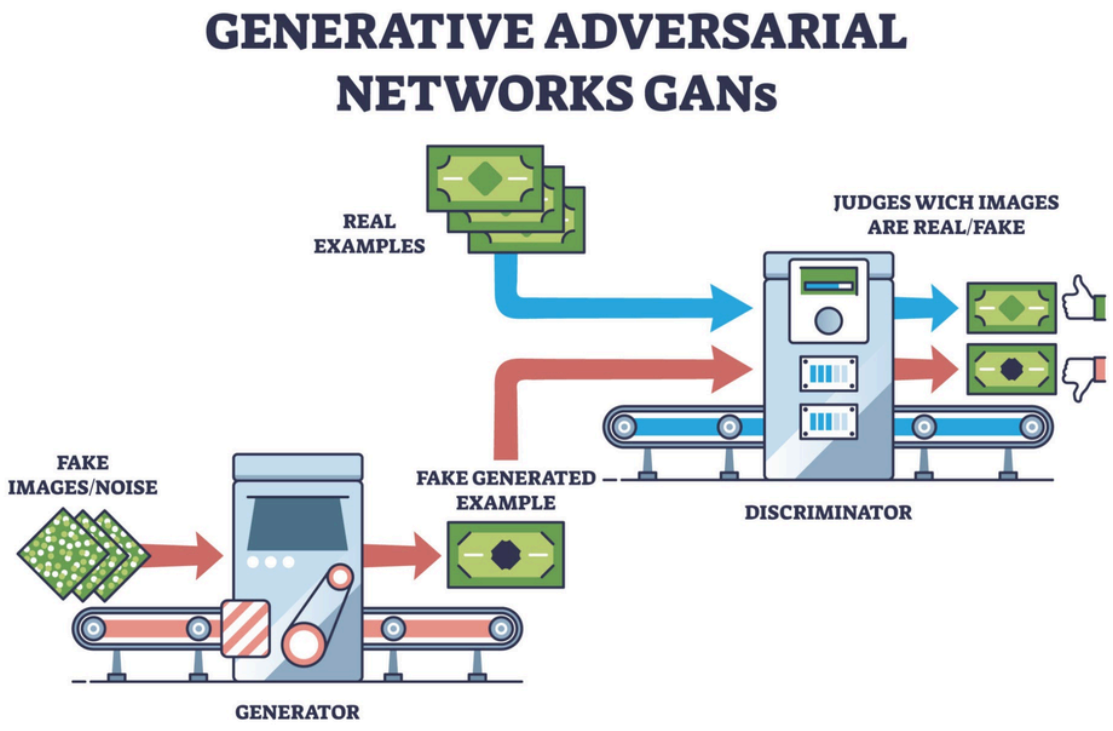
Discriminator (Trình so sánh/Phân biệt)

- Mục đích: Phân biệt dữ liệu thật (từ tập huấn luyện) và dữ liệu giả (từ Generator).
- Đầu vào: Dữ liệu thật hoặc dữ liệu giả.
- Đầu ra: Xác suất (một giá trị từ 0 đến 1) cho biết dữ liệu đầu vào là thật (gần 1) hay giả (gần 0).
- Mục tiêu: Phân loại chính xác nhất có thể.



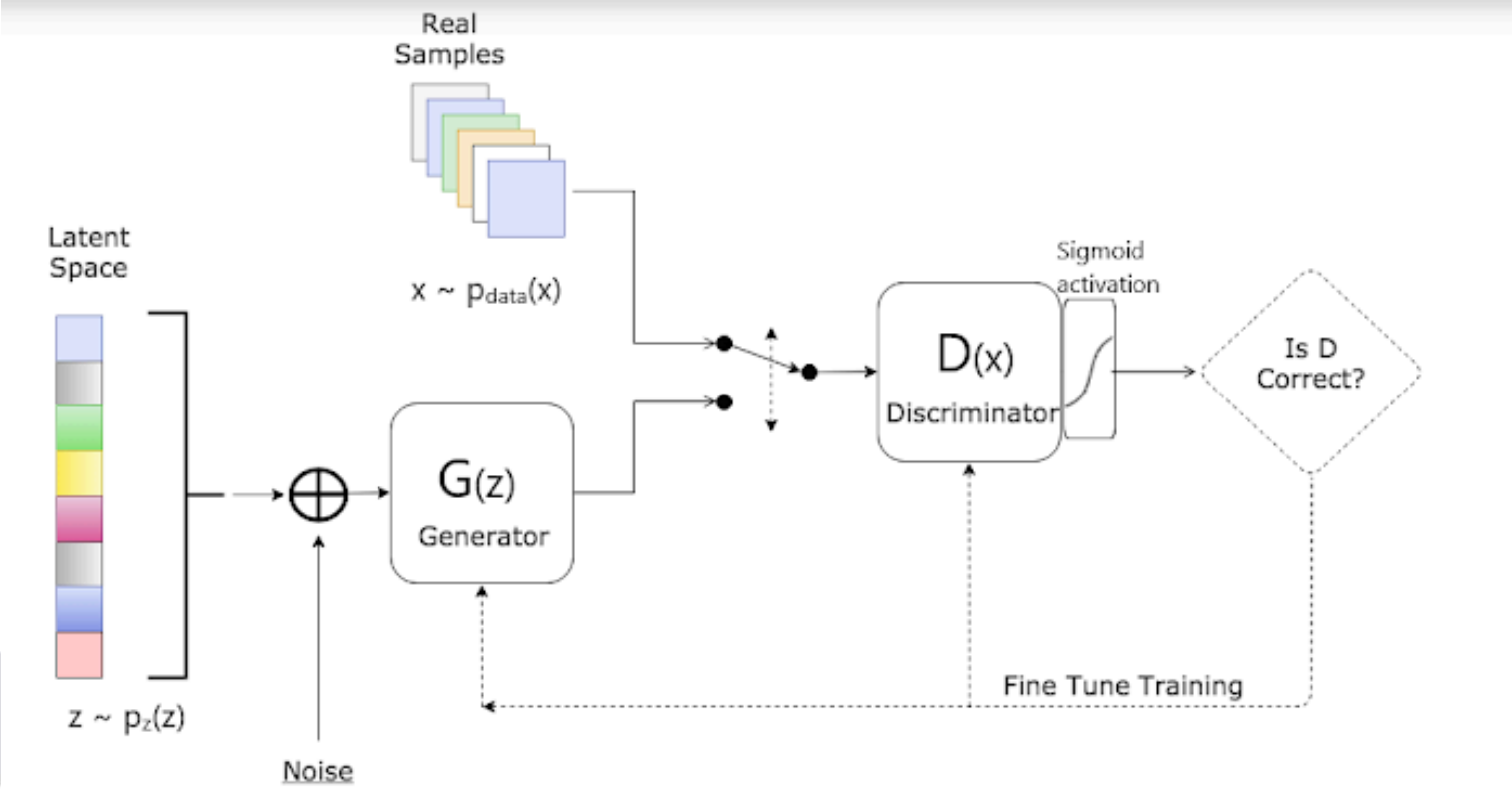
1.3 Cơ chế hoạt động của GAN

GAN hoạt động dựa trên một cơ chế học đối kháng (Adversarial Learning), trong đó bộ sinh và bộ phân biệt liên tục cạnh tranh với nhau để cải thiện chất lượng dữ liệu sinh ra.



1.3 Cơ chế hoạt động của GAN

- Bộ sinh cố gắng tạo ra dữ liệu giả nhưng trông giống dữ liệu thật nhất có thể.
- Bộ phân biệt cố gắng xác định dữ liệu nào là thật và dữ liệu nào là giả.



Huấn luyện Mô hình GAN:

Quy trình Huấn luyện:

1. Generator Loss: Dựa trên Binary Crossentropy với nhãn 1 (ảnh thật).
2. Discriminator Loss: Dựa trên Binary Crossentropy với nhãn 1 (ảnh thật) và 0 (ảnh giả).
3. Tối ưu hóa:
    - Adam optimizer với learning rate = 1e-4 cho cả Generator và Discriminator.

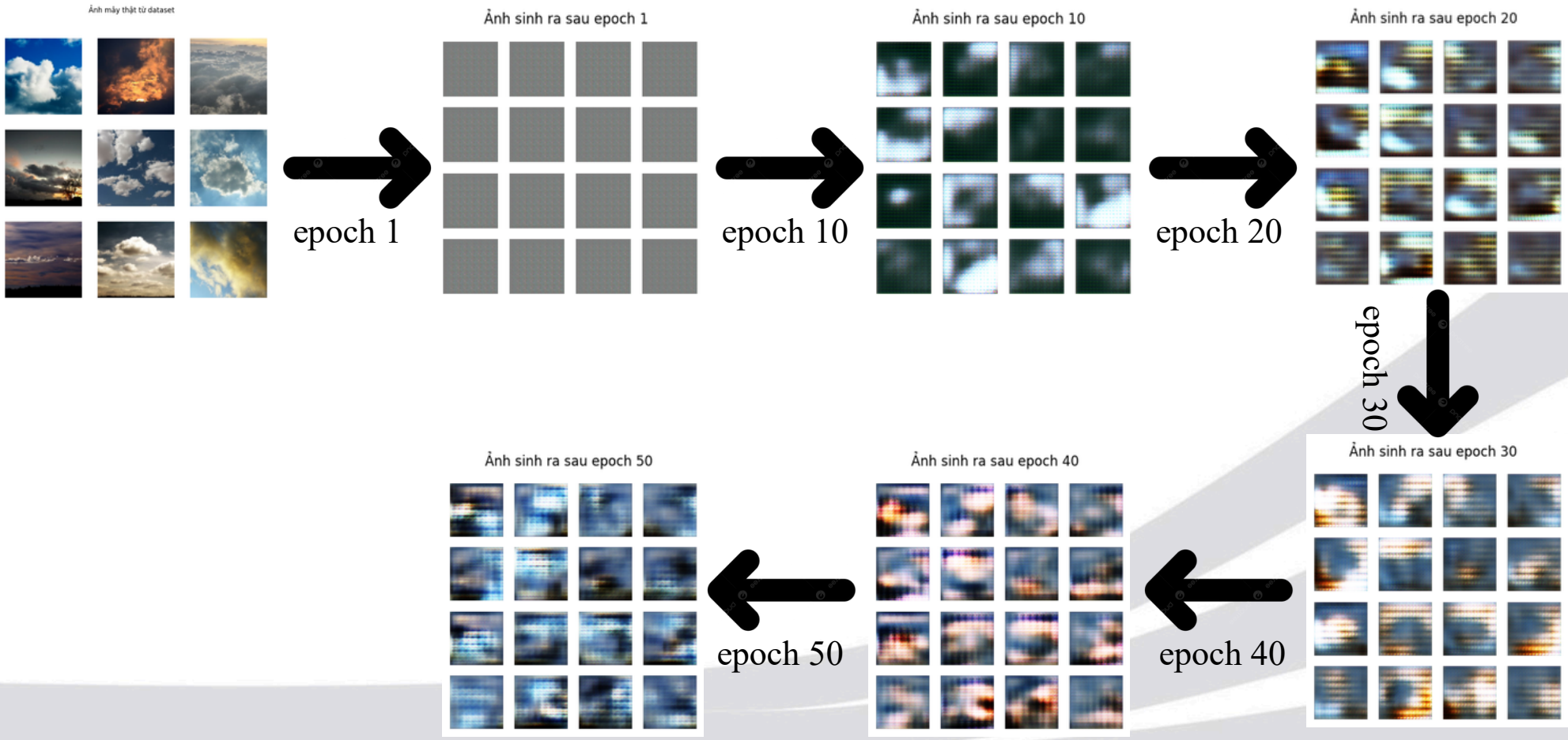
Số Epochs: 50

Kết quả Huấn luyện:

- Generator Loss: Mô hình giảm dần mất mát khi Generator học cách tạo ảnh chân thực hơn.
- Discriminator Loss: Mô hình giảm dần mất mát khi Discriminator học cách phân biệt ảnh thật và ảnh giả.

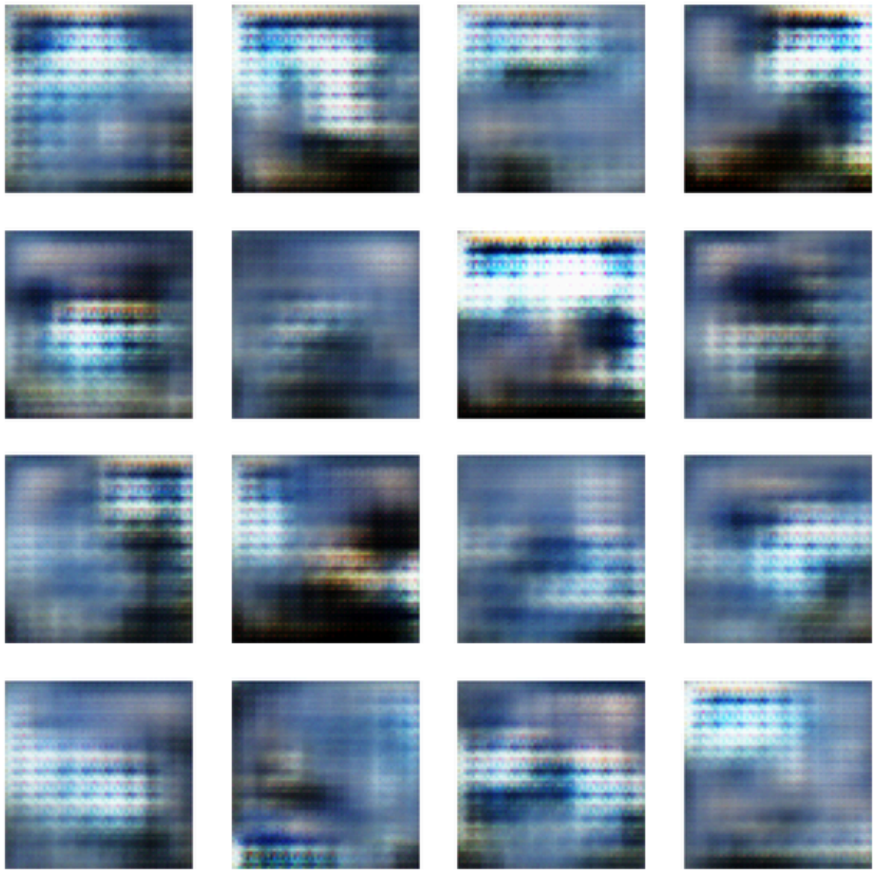
1.4 Huấn luyện, kết quả và đánh giá thuật toán

Mô phỏng Kết Quả Huấn Luyện:



Kết quả Sinh Ảnh Mây:

Ảnh mây nhân tạo do GAN sinh ra



1.4 Huấn luyện, kết quả và đánh giá thuật toán

Đánh giá mô hình:

- Generator dần tạo ra ảnh giả chân thực.
- Discriminator học cách phân biệt ảnh thật và ảnh giả.
- Độ chính xác và chất lượng ảnh sẽ cải thiện sau nhiều epoch.

1.5 Ưu điểm, nhược điểm

Ưu điểm:

- Tạo Dữ liệu Siêu Thực: Sinh ra hình ảnh, âm thanh, video có độ chân thực cao, khó phân biệt với dữ liệu thật.
- Học Không Giám Sát: Có khả năng học được phân phối dữ liệu phức tạp mà không cần nhãn (unlabeled data).
- Đa dạng Ứng dụng: Được sử dụng rộng rãi trong phục hồi/cải thiện ảnh (siêu phân giải), tạo dữ liệu tổng hợp (synthetic data) và sáng tạo nội dung.

Nhược điểm:

- Huấn luyện Không Ổn định: Quá trình huấn luyện rất khó hội tụ và cân bằng giữa Generator và Discriminator.
- Sụp đổ Chế độ (Mode Collapse): Generator chỉ tạo ra một số lượng nhỏ các mẫu lặp đi lặp lại thay vì toàn bộ sự đa dạng của dữ liệu.
- Khó Đánh giá: Rất khó để định lượng chất lượng đầu ra một cách khách quan, không có tiêu chí đánh giá chuẩn hóa rõ ràng.
- Yêu cầu Tài nguyên Lớn: Đòi hỏi tài nguyên tính toán (GPU) và thời gian huấn luyện đáng kể.

2. Thuật toán LSTM - Long Short-Term Memory

2.1. Giới thiệu

2.2. Cấu trúc của LSTM

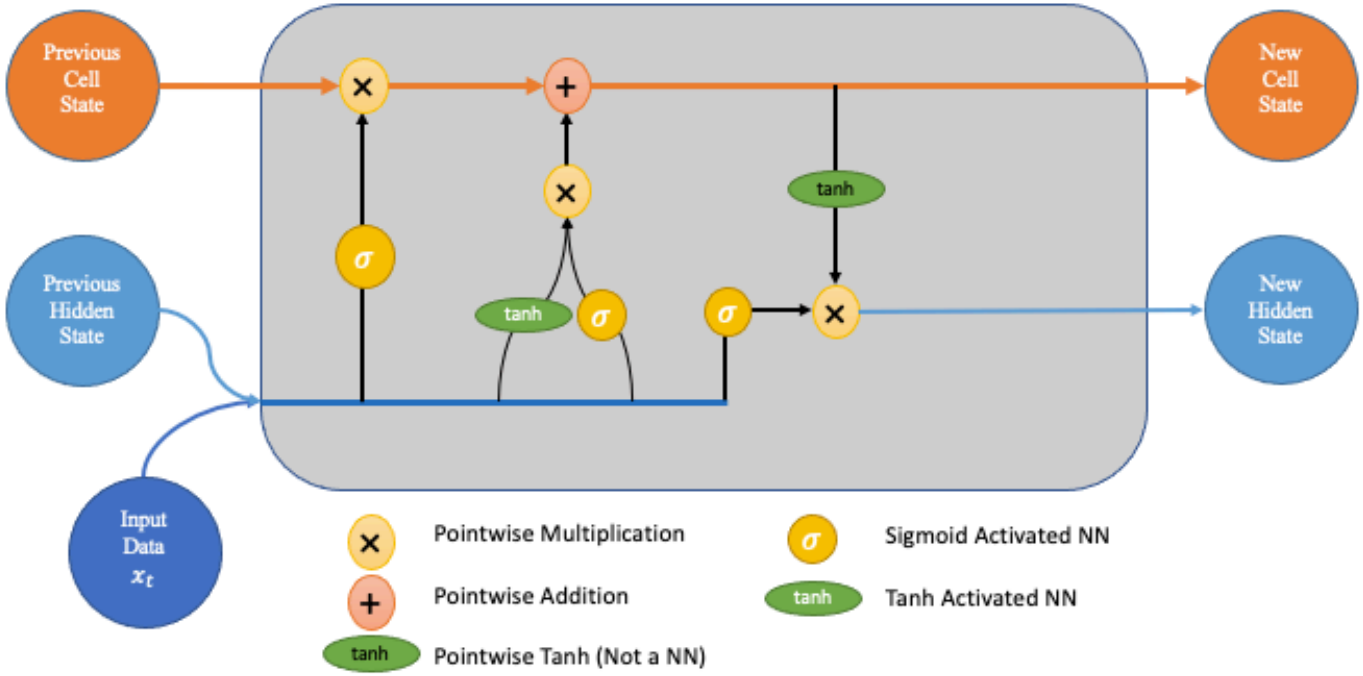
2.3. Cơ chế hoạt động của LSTM

2.4. Huấn luyện, kết quả và đánh giá thuật toán

2.5. Ưu và nhược điểm

2.1 Giới thiệu thuật toán LSTM - Long Short-Term Memory

Long Short-Term Memory (LSTM) là một cải tiến đột phá của Mạng nơ-ron hồi quy (Recurrent Neural Network - RNN), được thiết kế để giải quyết vấn đề lớn nhất của RNN truyền thống: Khả năng nhớ phụ thuộc dài hạn (Long-Term Dependency)



2.2. Cấu trúc của LSTM

Cấu trúc Cell LSTM: Bộ nhớ và Cổng

Mỗi cell LSTM xử lý đầu vào hiện tại (x_t), trạng thái ẩn bước trước (h_{t-1}), và trạng thái ô bước trước (C_{t-1}).

1. **Ô Trạng thái (C_t)**: Là **trạng thái bộ nhớ dài hạn** được cập nhật qua thời gian.
2. **Cổng Quên (f_t)**: Quyết định thông tin nào trong C_{t-1} cần được **bỏ đi** hay giữ lại.
3. **Cổng Đầu vào (i_t)**: Quyết định thông tin **mới** nào cần được thêm vào bộ nhớ.
4. **Ứng viên Bộ nhớ Mới ($\tilde{C}_t$)**: Sinh ra các giá trị ứng viên mới cho bộ nhớ.
5. **Cổng Đầu ra (o_t)**: Quyết định phần nào của bộ nhớ đã cập nhật (C_t) được **xuất ra** làm đầu ra.
6. **Trạng thái Ẩn (h_t)**: Là **đầu ra** tại thời điểm hiện tại (Hidden State), được truyền cho bước kế tiếp.

Các công thức cốt lõi của LSTM

1. Cổng Quên (Forget Gate - f_t)

$$\mathbf{f_t} = \sigma(\mathbf{W_f} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_f})$$

- f_t: Vector của Cổng Quên. Xác định mức độ giữ lại thông tin cũ trong C_{t-1} (từ 0 đến 1).
- W_f, b_f: Trọng số và Bias riêng cho Cổng Quên.

Các công thức cốt lõi của LSTM

2. Cổng Đầu vào (Input Gate - i_t)

$$\mathbf{i_t} = \sigma(\mathbf{W_i} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_i})$$

- i_t: Vector của Cổng Đầu vào. Xác định phần nào của thông tin mới sẽ được cập nhật (từ 0 đến 1).
- W_i, b_i: Trọng số và Bias riêng cho Cổng Đầu vào.

Các công thức cốt lõi của LSTM

3. Ứng viên Bộ nhớ Mới (Candidate Memory - $\tilde{C}_t$)

$$\tilde{\mathbf{C}}_\mathbf{t} = \tanh(\mathbf{W}_\mathbf{C} \cdot [\mathbf{h}_{\mathbf{t-1}}, \mathbf{x}_\mathbf{t}] + \mathbf{b}_\mathbf{C})$$

- $\tilde{C}_t$: Vector giá trị ứng viên mới, chứa thông tin tiềm năng để thêm vào C_t.
- W_C, b_C: Trọng số và Bias dùng để tính toán ứng viên bộ nhớ.

Các công thức cốt lõi của LSTM

4. Cập nhật Trạng thái Bộ nhớ (Cell State - C_t)

$$\mathbf{C_t} = \mathbf{f_t} * \mathbf{C_{t-1}} + \mathbf{i_t} * \tilde{\mathbf{C}}_\mathbf{t}$$

- $f_t * C_{t-1}$: Phần **bộ nhớ cũ** được **giữ lại** (sau khi lọc qua Cổng Quên).
- $i_t * \tilde{C}_t$: Phần **thông tin mới** được **thêm vào** (sau khi lọc qua Cổng Đầu vào).

Các công thức cốt lõi của LSTM

5. Cổng Đầu ra (Output Gate - o_t)

$$\mathbf{o_t} = \sigma(\mathbf{W_o} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_o})$$

- o_t: Vector của Cổng Đầu ra. Quyết định phần nào của bộ nhớ đã cập nhật (C_t) sẽ được xuất ra.
- W_o, b_o: Trọng số và Bias riêng cho Cổng Đầu ra.

Các công thức cốt lõi của LSTM

6. Tính Đầu ra Ẩn (Hidden State - h_t)

$$\mathbf{h_t} = \mathbf{o_t} * \tanh(\mathbf{C_t})$$

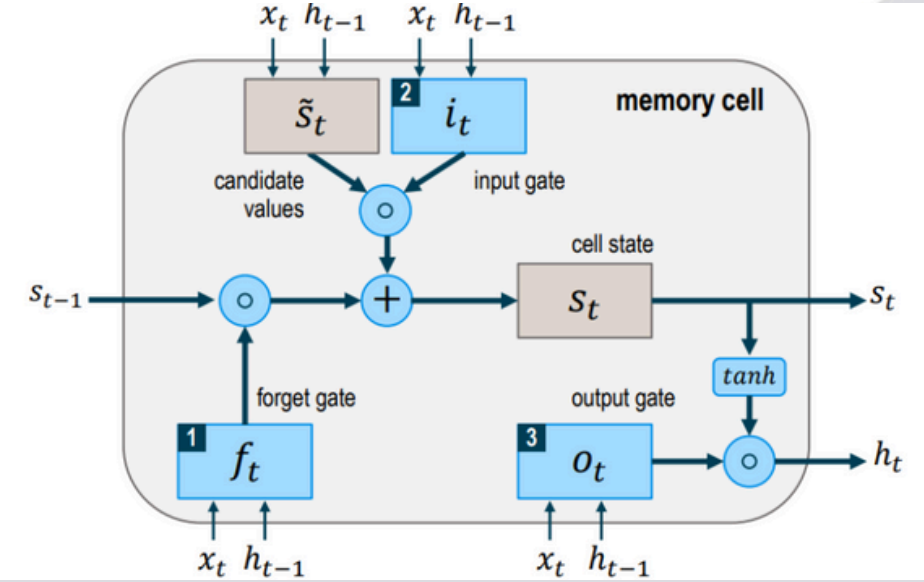
- $\tanh(C_t)$: Chuẩn hóa Trạng thái Ô (C_t) về khoảng $[-1, 1]$.
- $o_t * \tanh(C_t)$: Sử dụng o_t để **lọc** giá trị đã chuẩn hóa của C_t, tạo ra đầu ra ngắn hạn (h_t) cho bước hiện tại.

2. 3 Cơ chế hoạt động của LSTM

Bước 1: Quyết định Quên (Forget Gate - f_t)

$$\mathbf{f_t} = \sigma(\mathbf{W_f} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_f})$$

- Mạng nơ-ron Sigmoid này xác định giá trị từ 0 đến 1 cho từng bit thông tin trong C_{t-1}.
- Giá trị f_t **gần 0** nghĩa là **quên** thông tin đó; **gần 1** nghĩa là **giữ lại** thông tin đó.



2. 3 Cơ chế hoạt động của LSTM

Bước 2: Quyết định Thêm Mới (Input Gate - i_t và $\tilde{C}_t$)

$$\mathbf{i_t} = \sigma(\mathbf{W_i} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_i})$$

$$\tilde{\mathbf{C}}_\mathbf{t} = \tanh(\mathbf{W_C} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_C})$$

- i_t (Sigmoid) hoạt động như một bộ lọc: xác định phần nào của thông tin mới là **quan trọng**.
- $\tilde{C}_t$ (Tanh) tạo ra vector các giá trị ứng viên **mới** tiềm năng để thêm vào bộ nhớ.

2. 3 Cơ chế hoạt động của LSTM

Bước 3: Cập nhật Bộ nhớ (Cell State - C_t)

$$\mathbf{C_t = f_t * C_{t-1} + i_t * \tilde{C}_t}$$

- Đây là bước cốt lõi: Bộ nhớ mới (C_t) được tạo ra bằng cách kết hợp:
  1. Nhân **bộ nhớ cũ (C_{t-1}) với Cổng Quên (f_t)** (giữ lại phần cần thiết).
  2. Cộng với **thông tin mới ($\tilde{C}_t$) đã lọc qua Cổng Đầu vào (i_t)** (thêm vào phần quan trọng).

2. 3 Cơ chế hoạt động của LSTM

Bước 4: Tính Đầu ra (Output Gate - o_t và h_t)

$$\mathbf{o_t} = \sigma(\mathbf{W_o} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_o})$$

$$\mathbf{h_t} = \mathbf{o_t} * \tanh(\mathbf{C_t})$$

- o_t (Sigmoid) quyết định phần nào của C_t sẽ được sử dụng để tạo ra đầu ra.
- h_t (Trạng thái Ẩn) được tạo ra bằng cách lọc trạng thái bộ nhớ đã chuẩn hóa ($\tanh(C_t)$) qua Cổng Đầu ra (o_t). h_t này là đầu ra thực tế và được chuyển sang bước $t + 1$.

Cấu trúc mô hình:

Mô hình dự báo được xây dựng theo kiến trúc LSTM Multi-step với mục tiêu dự đoán số lượt thuê xe trong 24 giờ kế tiếp dựa trên dữ liệu lịch sử.

Kiến trúc cụ thể của mô hình như sau:

Thành phần	Loại lớp	Kích thước đầu ra	Số tham số
Lớp 1	LSTM (64 units, return_sequences=True)	(None, 24, 64)	18,688
Lớp 2	Dropout (tỷ lệ 0.2)	(None, 24, 64)	0
Lớp 3	LSTM (32 units, return_sequences=False)	(None, 32)	12,416
Lớp 4	Dense (24 neurons – dự báo 24 bước kế tiếp)	(None, 24)	792

Cấu trúc mô hình:

Tổng số tham số huấn luyện: 31.896

Mô hình được biên dịch với:

- Hàm mất mát (Loss): Mean Squared Error (MSE)
- Trình tối ưu hóa (Optimizer): Adam
- Cơ chế dừng sớm (EarlyStopping): theo dõi val_loss, patience=10, và khôi phục trọng số tốt nhất (restore_best_weights=True).

Quá trình huấn luyện:

Mô hình được huấn luyện trong tối đa 100 epoch, với batch_size = 32.

Dữ liệu được chia thành tập huấn luyện và tập kiểm định theo tỷ lệ 7/3.

Kết quả log huấn luyện cho thấy:

- Epoch 1: loss = 0.0206, val_loss = 0.0240
- Epoch 8: loss = 0.0051, val_loss = 0.0092
- Epoch 20: loss = 0.0037, val_loss = 0.0091
- Mô hình dừng ở Epoch 27 do hội tụ (early stopping được kích hoạt).

Điều này cho thấy mô hình học dần ổn định, giảm được cả training loss và validation loss, đồng thời tránh được hiện tượng overfitting nhờ Dropout và EarlyStopping.

2.4 Huấn luyện, kết quả và đánh giá thuật toán

Kết quả dự đoán:

Sau khi huấn luyện, mô hình được dùng để dự đoán chuỗi 24 giờ tiếp theo trên tập kiểm thử.

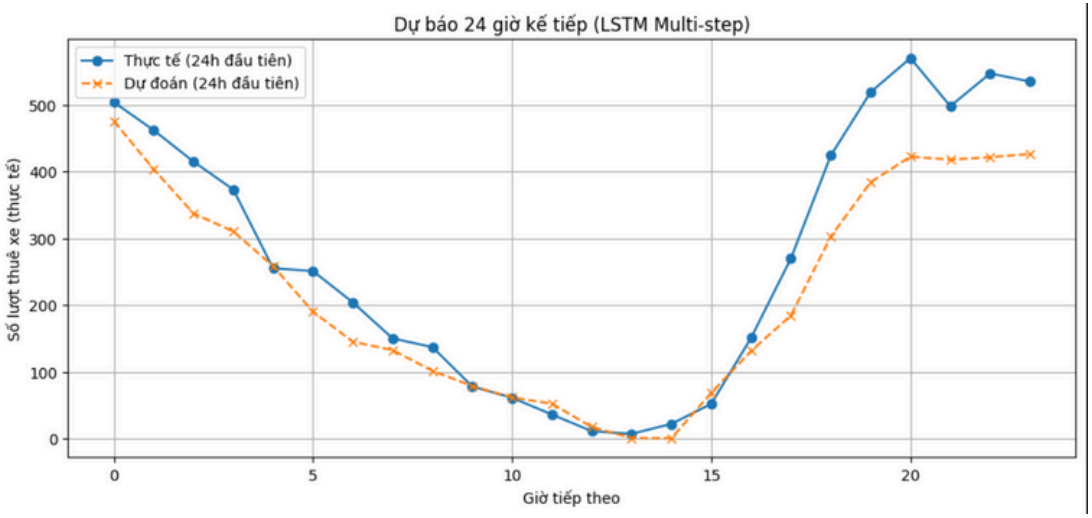
Kết quả đánh giá:

Chỉ số đánh giá	Giá trị
MSE (Mean Squared	854.133
R^2 (Hệ số xác định)	8.243

- Mô hình đạt $R^2 = 0.82$, cho thấy 82% biến thiên của dữ liệu thực tế được giải thích bởi mô hình dự báo.
- Đây là một kết quả khá tốt đối với bài toán dự báo chuỗi thời gian có tính biến động cao.

2.4 Huấn luyện, kết quả và đánh giá thuật toán

Đánh giá trực quan:



- Mô hình bắt được xu hướng tổng thể của dữ liệu (giảm dần buổi sáng – tăng mạnh vào buổi tối).
- Tuy nhiên, biên độ dự đoán thấp hơn thực tế ở các giai đoạn tăng nhanh (đặc biệt sau 17h).
- Sai số lớn hơn xuất hiện ở các vùng có biến động mạnh, nhưng dự đoán vẫn giữ được hình dạng đặc trưng của chu kỳ.

2.5 Ưu điểm, nhược điểm

Ưu điểm:

- Mô hình LSTM học được quy luật chu kỳ 24h của dữ liệu thời gian.
- Độ chính xác cao ($R^2 > 0.8$).
- Ổn định và tránh overfitting nhờ Dropout và EarlyStopping.

Nhược điểm:

- Dự đoán kém chính xác ở các điểm cực trị (tăng/giảm đột ngột).
- Cần tinh chỉnh thêm tham số (số đơn vị LSTM, learning rate) hoặc kết hợp Attention / CNN-LSTM để cải thiện biên độ dự đoán.

3. Thuật toán RNN - Recurrent Neural Network

3.1. Giới thiệu

3.2. Cấu trúc của thuật toán

3.3. Cơ chế hoạt động của RNN

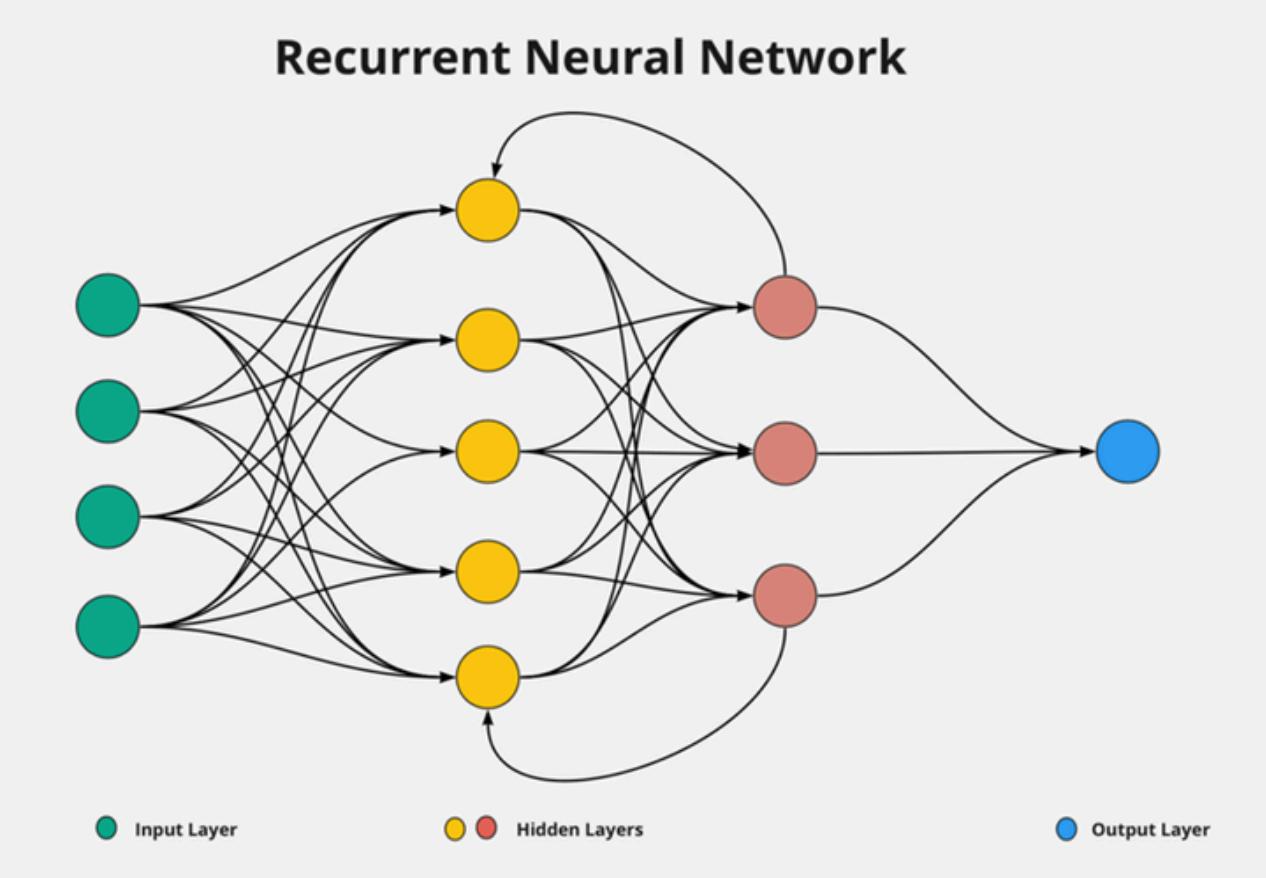
3.4. Phân loại

3.5. Huấn luyện, kết quả và đánh giá thuật toán

3.6. Ưu và nhược điểm

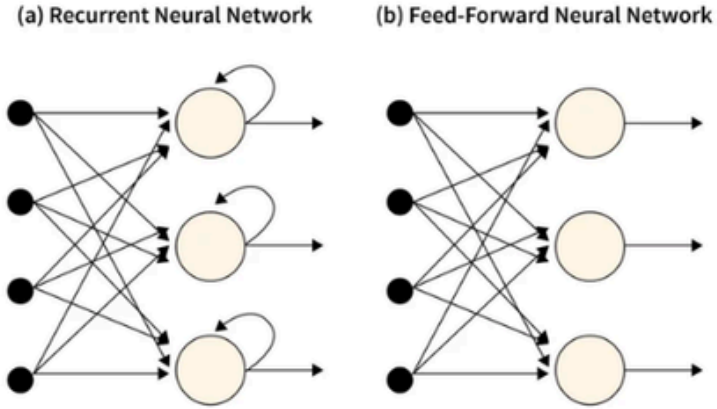
3.1. Giới thiệu

Mạng nơ-ron hồi quy (Recurrent Neural Network – RNN) là một loại mô hình học sâu được thiết kế và huấn luyện để thực hiện việc xử lý và biến đổi dữ liệu đầu vào có dạng tuần tự thành một dạng dữ liệu đầu ra tuần tự cụ thể.



3.2. Cấu trúc của thuật toán

- **Cấu trúc Phát triển (Unrolled):** Mạng được mở rộng theo thời gian để minh họa dòng dữ liệu tuần tự.
- **Đầu vào tại t:**
  - x_t: Đầu vào hiện tại (thông tin mới).
  - h_{t-1}: Trạng thái Ẩn trước (bộ nhớ/ngữ cảnh cũ).



3.2. Cấu trúc của thuật toán

- **Trạng thái Ẩn (h_t):**
  - Là **Trạng thái Bộ nhớ mới** tại bước t.
  - Được tính bằng cách kết hợp thông tin mới (x_t) và bộ nhớ cũ (h_{t-1}).
  - $$\mathbf{h_t} = \tanh(\mathbf{W_h} \cdot \mathbf{h_{t-1}} + \mathbf{W_x} \cdot \mathbf{x_t} + \mathbf{b})$$
  - W_h và W_x là các **Ma trận Trọng số được chia sẻ** qua tất cả các bước.
- **Đầu ra (y_t):** Kết quả dự đoán tại thời điểm hiện tại, được tính toán từ h_t.

3.3. Cơ chế hoạt động của RNN

1. Dòng Dữ liệu Tuần tự (Sequential Data Flow)

- **Đầu vào:** RNN xử lý dữ liệu theo từng bước thời gian t, từ đầu đến cuối chuỗi (ví dụ: từ đầu tiên, từ thứ hai, v.v.).
- **Hồi quy:** Tại mỗi bước, mạng sử dụng lại **cùng một tập hợp Ma trận Trọng số** (W_x, W_h) để tính toán. Đây là đặc điểm tiết kiệm tham số nhất của RNN.

2. Cơ chế Bộ nhớ: Trạng thái Ẩn (h_t)

- **Bộ nhớ Ngắn hạn:** Trạng thái Ẩn (h_t) đóng vai trò là **bộ nhớ** của mạng, chứa thông tin tổng hợp (ngữ cảnh) từ các đầu vào $x_1, x_2, \ldots, x_t$ đã được xử lý.
- **Cập nhật:** Tại mỗi bước t, h_t được tính toán lại dựa trên **Đầu vào hiện tại (x_t)** và **Bộ nhớ cũ (h_{t-1})**.

3.3. Cơ chế hoạt động của RNN

3. Công thức Cập nhật Trạng thái Ẩn (Tính toán Cốt lõi)

Trạng thái Ẩn mới (h_t) là kết quả của việc kết hợp tuyến tính giữa đầu vào mới và bộ nhớ cũ, sau đó đi qua hàm kích hoạt $\tanh$:

$$\mathbf{h_t} = \tanh(\mathbf{W_h} \cdot \mathbf{h_{t-1}} + \mathbf{W_x} \cdot \mathbf{x_t} + \mathbf{b})$$

- **Tác động của** x_t: Thông tin mới x_t được nhân với ma trận trọng số W_x.
- **Tác động của** h_{t-1}: Bộ nhớ cũ h_{t-1} được nhân với ma trận trọng số W_h.
- **Hàm** $\tanh$: Nén kết quả về $[-1, 1]$ để giữ cho đạo hàm (gradients) không quá lớn hoặc quá nhỏ.

3.3. Cơ chế hoạt động của RNN

4. Tính toán Đầu ra (y_t)

- **Đầu ra:** Đầu ra dự đoán (y_t) tại thời điểm t được tính toán dựa trên trạng thái ẩn hiện tại (h_t) thông qua một phép biến đổi tuyến tính khác:

$$\mathbf{y_t} = \mathbf{W_y} \cdot \mathbf{h_t} + \mathbf{b_y}$$

- W_y là ma trận trọng số đầu ra.
- y_t thường được đưa qua hàm **Softmax** (cho bài toán phân loại) hoặc các hàm khác (tùy thuộc vào nhiệm vụ cụ thể).

3.3. Cơ chế hoạt động của RNN

5. Quá trình Huấn luyện (Backpropagation Through Time - BPTT)

- RNN được huấn luyện bằng thuật toán **Lan truyền ngược theo thời gian (BPTT)**.
- BPTT về cơ bản là Lan truyền ngược tiêu chuẩn được áp dụng cho cấu trúc RNN đã mở rộng, tính toán gradient qua tất cả các bước thời gian.
- **Hạn chế:** Khi chuỗi quá dài, BPTT dẫn đến vấn đề **Vanishing Gradient** (đạo hàm biến mất), làm cho mạng không thể học được các phụ thuộc dài hạn. (Đây là lý do chính LSTM ra đời).

3.4. Phân loại

1. One-to-One (1:1)

- Đặc điểm: Một đầu vào duy nhất tạo ra một đầu ra duy nhất.
- Mô tả: Đây thực chất là một mạng nơ-ron truyền thẳng (Feedforward Neural Network) tiêu chuẩn. Không có sự hồi quy.
- Ứng dụng: Phân loại ảnh (Image Classification).

2. One-to-Many (1:N)

- Đặc điểm: Một đầu vào duy nhất tạo ra một chuỗi (sequence) các đầu ra.
- Mô tả: Đầu vào duy nhất được đưa vào mạng, và mạng bắt đầu tạo ra một chuỗi các đầu ra liên tiếp, sử dụng h_t làm ngữ cảnh.
- Ứng dụng:
- Tạo phụ đề ảnh (Image Captioning): Đầu vào là một bức ảnh, đầu ra là một câu mô tả ảnh.
- Tạo nhạc: Đầu vào là một nốt nhạc/thể loại, đầu ra là một chuỗi các nốt nhạc tạo thành bài hát.

3. Many-to-One (N:1)

- Đặc điểm: Một chuỗi các đầu vào tạo ra một đầu ra duy nhất.
- Mô tả: Mạng xử lý toàn bộ chuỗi đầu vào, và chỉ có trạng thái ẩn cuối cùng (h_N) được sử dụng để tạo ra đầu ra dự đoán cuối cùng.
- Ứng dụng:
- Phân tích Cảm xúc (Sentiment Analysis): Đầu vào là một câu/đoạn văn, đầu ra là cảm xúc chung (Tích cực/Tiêu cực/Trung tính).
- Phân loại Văn bản: Đầu vào là tài liệu, đầu ra là chủ đề chính.

4. Many-to-Many (N:N) - Đồng bộ (Synchronous)

- Đặc điểm: Một chuỗi đầu vào tạo ra một chuỗi đầu ra có cùng độ dài, và đầu ra được tạo ra ở mỗi bước thời gian.
- Mô tả: Đầu ra y_t được tạo ra ngay sau khi xử lý đầu vào x_t.
- Ứng dụng:
- Nhận dạng Thực thể Tên (NER): Đầu vào là một từ, đầu ra là nhãn của từ đó (Ví dụ: Từ "Việt Nam" là "Địa điểm").
- Nhận dạng Giọng nói (Speech Recognition): Đầu vào là tín hiệu âm thanh, đầu ra là chuỗi văn bản.

5. Many-to-Many (N:M) - Không đồng bộ (Encoder-Decoder)

- Đặc điểm: Một chuỗi đầu vào tạo ra một chuỗi đầu ra với độ dài khác nhau.
- Mô tả: Cấu trúc gồm hai phần:

1. Encoder (Bộ Mã hóa): Xử lý chuỗi đầu vào ($X_1, ..., X_N$) và tổng hợp thông tin vào một vector ngữ cảnh C (là h_N cuối cùng).
2. Decoder (Bộ Giải mã): Nhận vector ngữ cảnh C và tạo ra chuỗi đầu ra ($Y_1, ..., Y_M$) với độ dài M.

- Ứng dụng: Dịch máy (Machine Translation): Đầu vào là câu tiếng Anh, đầu ra là câu tiếng Việt (độ dài thường khác nhau).

3.5. Huấn luyện, kết quả và đánh giá thuật toán

Cấu trúc mô hình:

Mô hình được xây dựng dựa trên mạng nơ-ron hồi tiếp đơn giản (SimpleRNN) với kiến trúc như sau:

Thành phần	Loại lớp	Kích thước đầu ra	Số tham số
Lớp 1	SimpleRNN (64 units)	(None, 64)	4,8
Lớp 2	Dropout (tỷ lệ 0.2)	(None, 64)	0
Lớp 3	Dense (32 neurons, ReLU)	(None, 32)	2,08
Lớp 4	Dense (24 neurons – đầu ra dự báo 24 giờ kế tiếp)	(None, 24)	792

3.5. Huấn luyện, kết quả và đánh giá thuật toán

Cấu trúc mô hình:

- Tổng số tham số: 7.672
- Hàm mất mát (Loss): Mean Squared Error (MSE)
- Bộ tối ưu (Optimizer): Adam
- Số epoch: 30
- Kích thước batch: 32
- Tỷ lệ validation: 10%

Quá trình huấn luyện:

Trong quá trình huấn luyện, hàm mất mát giảm đều qua các epoch, thể hiện mô hình học được mối quan hệ giữa chuỗi dữ liệu thời gian và giá trị cần dự đoán.

- Ở epoch 1, loss = 0.0294 và val_loss = 0.0282
- Sau 10 epoch, val_loss giảm còn ≈ 0.0128
- Đến epoch 30, mô hình hội tụ với loss = 0.0042, val_loss = 0.0100

Điều này cho thấy mô hình hội tụ nhanh, không có hiện tượng overfitting đáng kể nhờ việc sử dụng Dropout(0.2).

3.5. Huấn luyện, kết quả và đánh giá thuật toán

Kết quả dự đoán:

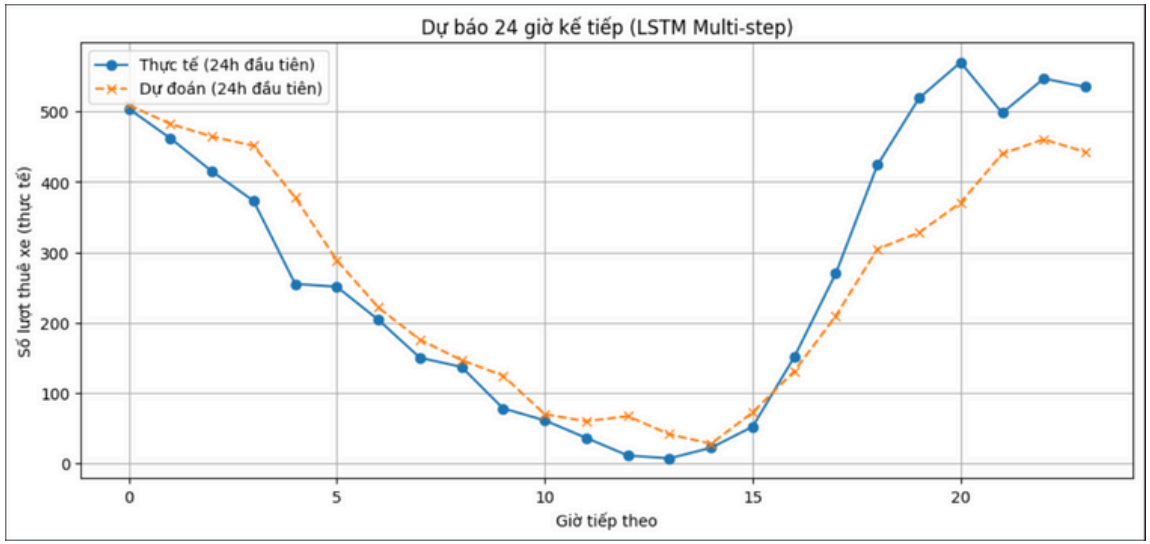
Kết quả đánh giá được tính bằng các chỉ số:

Chỉ số đánh giá	Giá trị
MSE (Mean Squared Error)	11,536.58
R^2 (Hệ số xác định)	7.627

Kết quả cho thấy mô hình RNN có khả năng dự báo tốt, với $R^2 = 0.76$, tức là khoảng 76% độ biến thiên của dữ liệu thực tế được mô hình giải thích. Mặc dù thấp hơn mô hình LSTM ($R^2 = 0.82$), song đây vẫn là hiệu quả chấp nhận được đối với mạng hồi tiếp không có cơ chế ghi nhớ dài hạn.

3.5. Huấn luyện, kết quả và đánh giá thuật toán

Đánh giá trực quan:



- Mô hình RNN nắm bắt được xu hướng tổng thể của chuỗi dữ liệu: giảm dần vào buổi sáng, chạm đáy giữa trưa, rồi tăng mạnh vào buổi tối.
- Tuy nhiên, đường dự đoán (màu cam) có xu hướng mượt hơn và biên độ nhỏ hơn thực tế, đặc biệt ở các giai đoạn biến động mạnh (ví dụ: tăng nhanh sau 17h).
- Điều này phản ánh hạn chế của RNN cổ điển, vốn khó ghi nhớ các quan hệ dài hạn trong chuỗi thời gian phức tạp.

3.6 Ưu và nhược điểm

Ưu điểm:

- Xử lý dữ liệu chuỗi thời gian hiệu quả
- Cấu trúc đơn giản, dễ triển khai
- Phù hợp với dữ liệu có chu kỳ ngắn
- Tiết kiệm tài nguyên tính toán

Nhược điểm:

- Hiện tượng mất và bùng nổ gradient
- Khó nắm bắt phụ thuộc dài hạn
- Tính song song kém
- Độ ổn định kém với chuỗi có biến động mạnh

4. Thuật toán CNN - Convolutional Neural Network

4.1. Giới thiệu

4.2. Các lớp cơ bản của mạng CNN

4.3. Cấu trúc của mạng CNN

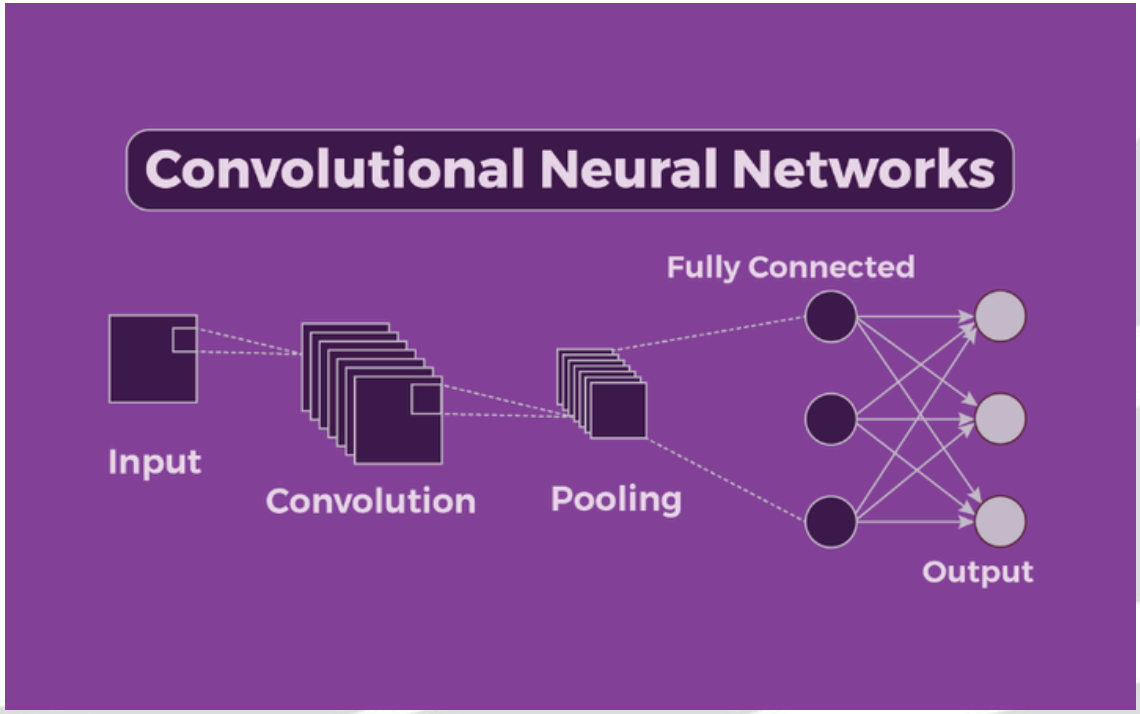
4.4. Huấn luyện, kết quả và đánh giá thuật toán

4.5. Ưu và nhược điểm

4.1. Giới thiệu

- CNN là một loại mạng nơ-ron chuyên dùng cho xử lý và phân loại hình ảnh, video, nhận dạng đối tượng.

- CNN sử dụng các lớp tích chập để trích xuất đặc trưng của hình ảnh, giúp tự động nhận diện các mẫu mà không cần phải can thiệp thủ công.

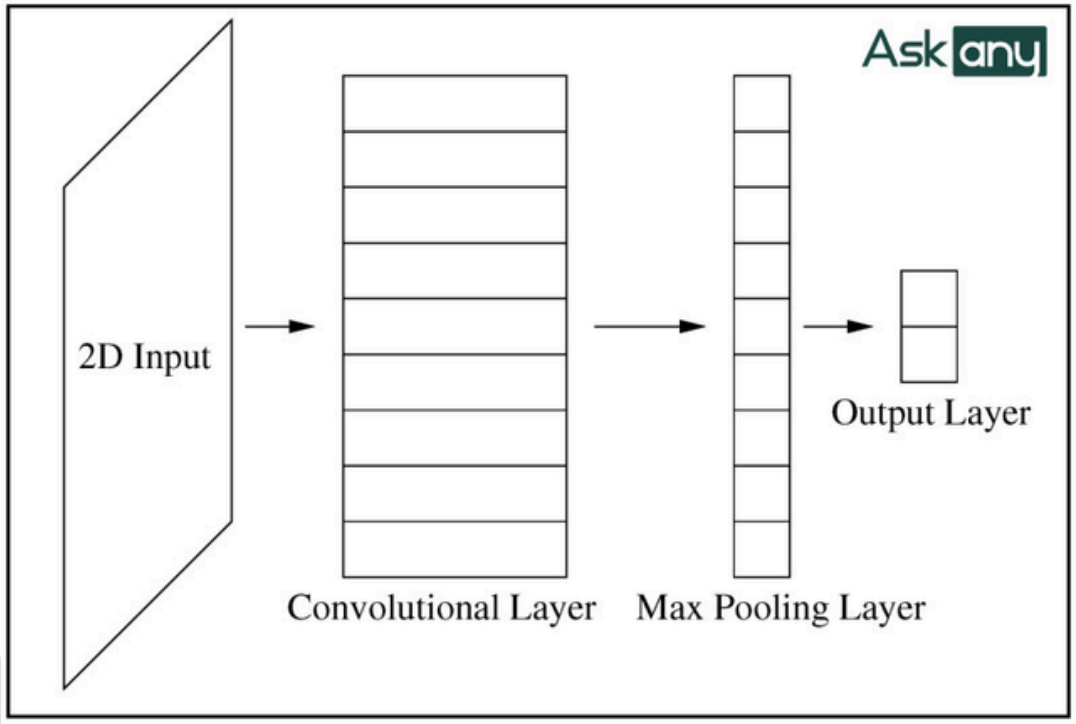


4.2 Các lớp cơ bản của mạng CNN

Convolutional layer

Đây là lớp quan trọng nhất của CNN, thực hiện phần lớn các phép tính toán. Các yếu tố then chốt của lớp này bao gồm:

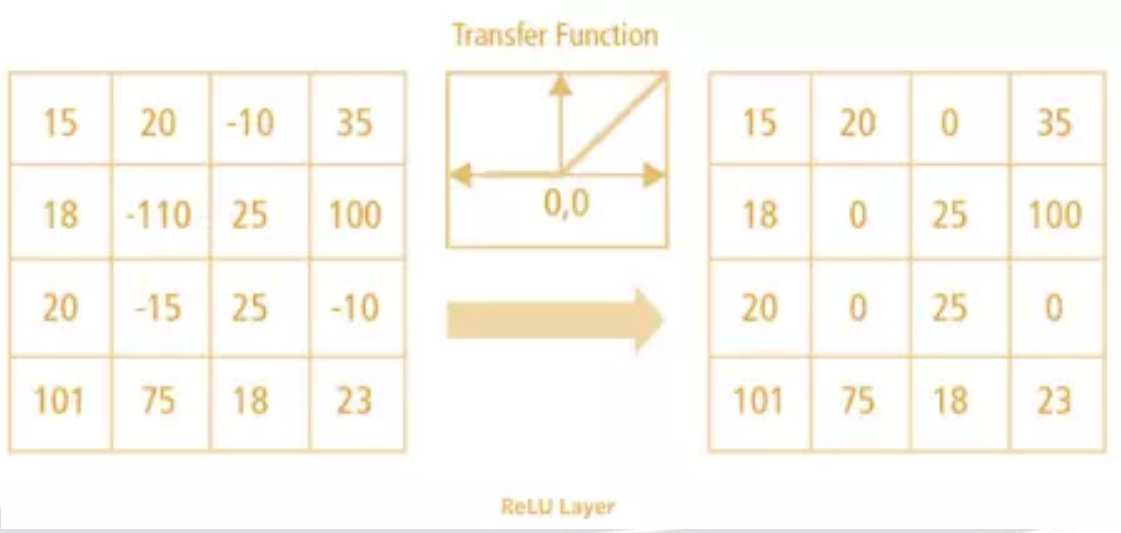
- Stride (bước trượt)
- Padding (phần đệm)
- Filter map (bản đồ bộ lọc)
- Feature map (bản đồ đặc trưng).



4.2 Các lớp cơ bản của mạng CNN

Relu layer

- Đảm nhiệm vai trò mô phỏng hoạt động của các nơ-ron thần kinh bằng cách truyền tín hiệu qua axon.
- Relu layer được ứng dụng rộng rãi trong quá trình huấn luyện mạng nơ-ron nhằm giúp mô hình học nhanh và chính xác hơn.

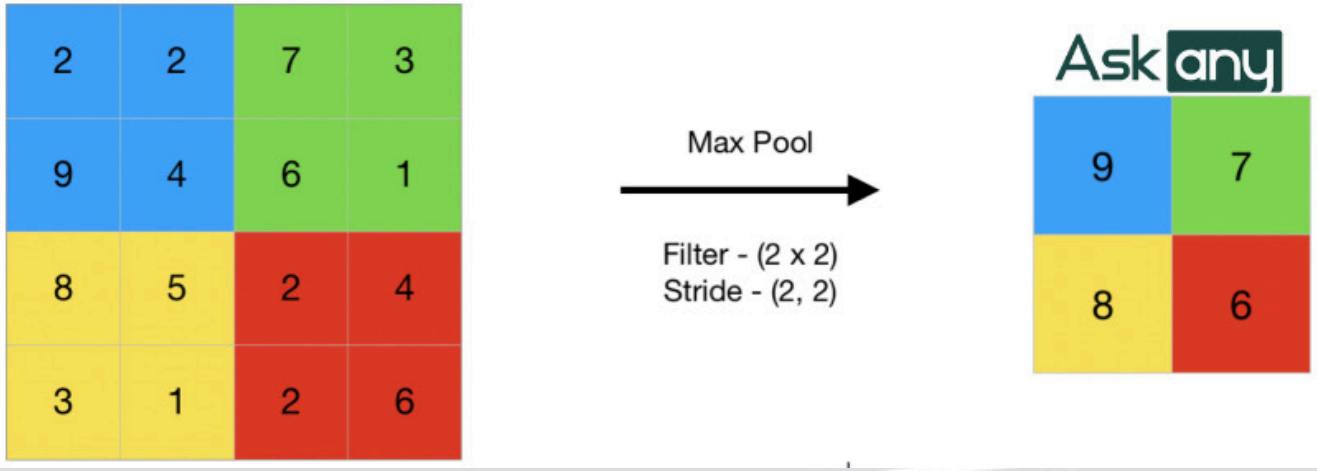


4.2 Các lớp cơ bản của mạng CNN

Pooling layer

Pooling layer mang đến lợi thế giảm kích thước đầu vào khi dữ liệu quá lớn, được đặt giữa các lớp convolutional layer. Max pooling và average pooling là hai loại pooling layer phổ biến nhất. Trong đó:

- Max pooling: Lấy giá trị lớn nhất trong vùng dữ liệu.
- Average pooling: Tính giá trị trung bình.



4.2 Các lớp cơ bản của mạng CNN

Fully connected layer

- Sau các bước Convolution và Pooling, chúng ta thu được thông tin của ảnh ở dạng đã được xử lý. Lớp kết nối đầy đủ (fully connected layer) có nhiệm vụ kết nối tất cả các nơ-ron từ lớp trước đến tất cả các nơ-ron ở lớp tiếp theo.
- Lớp kết nối đóng vai trò như một bộ phân loại (classifier), sử dụng các đặc trưng đã được trích xuất để đưa ra dự đoán cuối cùng. Nếu lớp này nhận được data về ảnh thì chúng sẽ chuyển data đó thành 1 mục chưa phân loại.

4.3. Cấu trúc của mạng CNN

- Local Receptive Field: Lọc những vùng ảnh có giá trị sử dụng cao nhất.
- Shared Weights and Bias: Giảm thiểu lượng tham số, mỗi feature map nhận diện các đặc trưng khác nhau của ảnh.
- Pooling Layer: Tối ưu hoá thông tin đầu ra nhờ loại bỏ các thông tin không cần thiết sau khi tính toán qua các lớp, giúp người dùng nhận kết quả phù hợp nhất với yêu cầu.

4.4. Huấn luyện, kết quả và đánh giá thuật toán

Tiền xử lý dữ liệu:

- Kích thước ảnh: 28x28 pixel, ảnh đơn sắc (grayscale).
- Chuẩn hóa ảnh: Dữ liệu pixel đã được chuẩn hóa từ [0, 255] thành [0, 1] để mô hình học tốt hơn.
- Reshape: Thêm một chiều vào dữ liệu ảnh (28x28 -> 28x28x1) để phù hợp với đầu vào của mạng CNN.

4.4. Huấn luyện, kết quả và đánh giá thuật toán

Mô hình CNN:

Lớp	Loại Lớp	Chi tiết
Input Layer	-	28x28 ảnh với 1 kênh (grayscale)
Lớp 1	Conv2D + MaxPooling2D	Conv2D (32 filters, 3x3 kernel, ReLU), MaxPooling2D (2x2)
Lớp 2	Conv2D + MaxPooling2D	Conv2D (64 filters, 3x3 kernel, ReLU), MaxPooling2D (2x2)
Lớp fully connected	Dense	Dense (128 neurons, ReLU)
Lớp đầu ra	Dense (Softmax)	Dense (10 neurons, softmax) để phân loại 10 loại sản phẩm

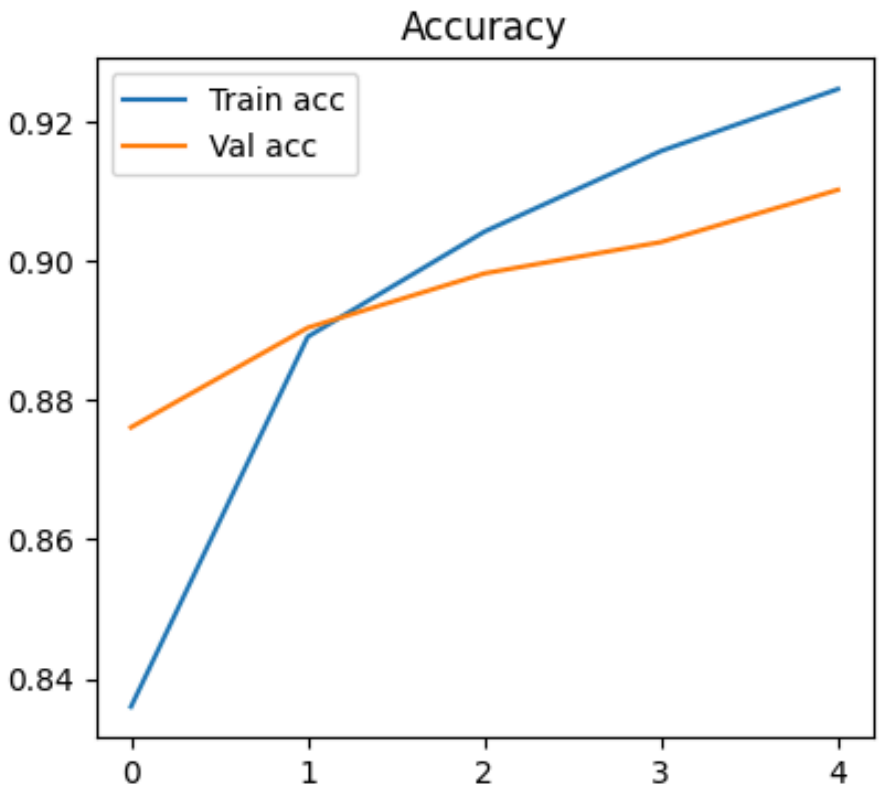
4.4. Huấn luyện, kết quả và đánh giá thuật toán

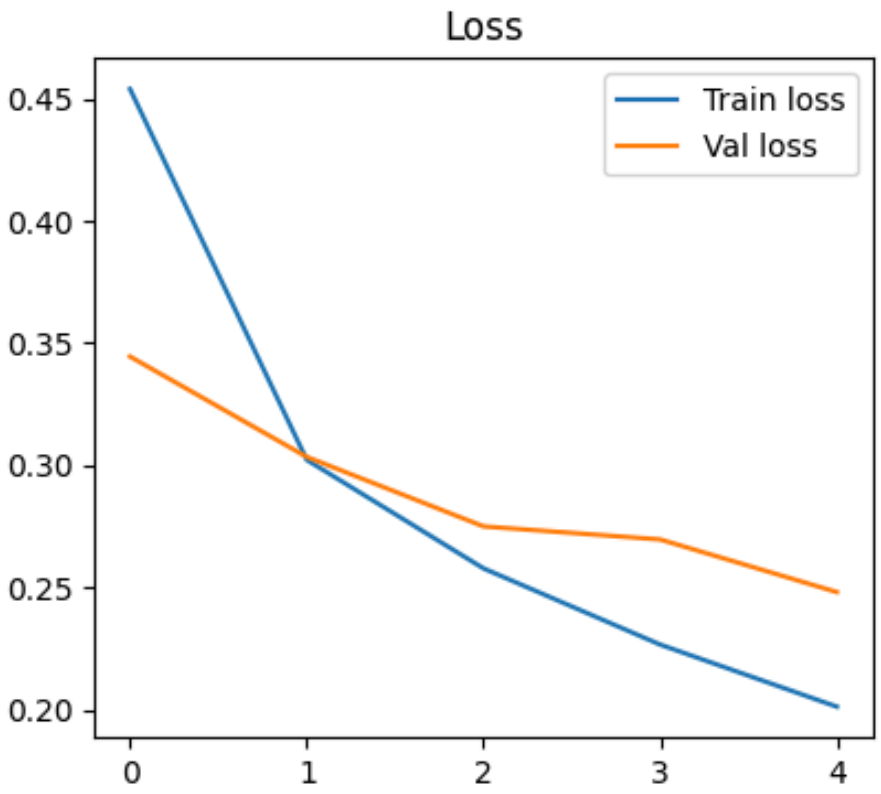
Kết quả Huấn luyện:

- Đồ thị Accuracy:
  - Độ chính xác trên tập huấn luyện và tập kiểm tra theo từng epoch.
  - Sự tăng trưởng trong độ chính xác trên tập kiểm tra trong quá trình huấn luyện cho thấy mô hình đang học tốt.
- Đồ thị Loss:
  - Giảm loss qua các epoch, cho thấy mô hình đang tối ưu hóa tốt.

4.4. Huấn luyện, kết quả và đánh giá thuật toán

Biểu đồ Mô phỏng Kết Quả Huấn Luyện:





4.4. Huấn luyện, kết quả và đánh giá thuật toán

Đánh Giá Mô Hình:

- Độ chính xác (Accuracy): Mô hình đạt 91.02% trên tập test.
- Chất lượng mô hình: Đạt kết quả khá tốt trong việc phân loại các loại trang phục thời trang từ Fashion MNIST.

4.5. Ưu và nhược điểm

Ưu điểm:

- Tự động trích xuất đặc trưng, không cần kỹ thuật thủ công.
- Hiệu quả cao trong nhận dạng hình ảnh và video.
- Có khả năng tổng quát tốt khi huấn luyện trên dữ liệu lớn.

Nhược điểm:

- Cần lượng lớn dữ liệu có nhãn để huấn luyện.
- Đòi hỏi tài nguyên tính toán lớn và bộ nhớ cao trong quá trình huấn luyện.
- Dễ bị overfitting (quá khớp) nếu dữ liệu không đủ lớn hoặc không có kỹ thuật điều chuẩn (regularization) phù hợp.

5. Thuật toán ANN - Artificial Neural Network

5.1. Giới thiệu

5.2. Cấu trúc của ANN

5.3. Nguyên lý hoạt động

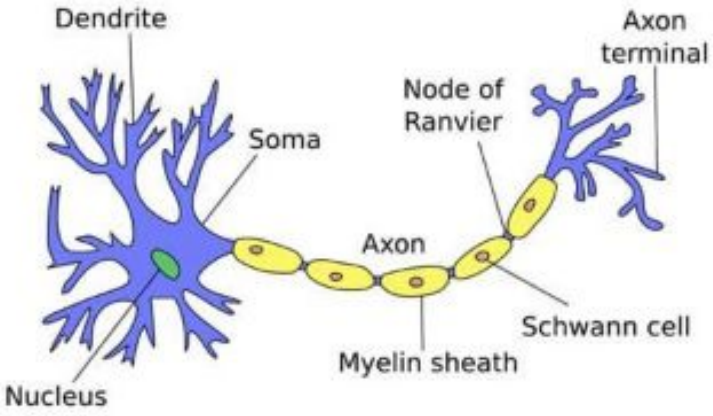
5.4. Huấn luyện, kết quả và đánh giá thuật toán

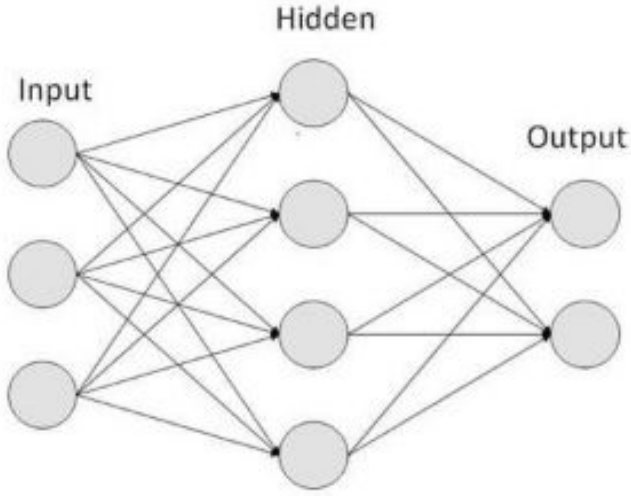
5.5. Ưu và nhược điểm

5.1. Giới thiệu

- Mạng nơ-ron nhân tạo là một mô hình tính toán được lấy cảm hứng từ cấu trúc và chức năng của bộ não con người. ANN được thiết kế để thực hiện:

- Nhận diện các mẫu phức tạp.
- Thực hiện các nhiệm vụ như phân loại, dự đoán, và xử lý dữ liệu.





5.2. Cấu trúc của ANN

Lớp đầu vào (Input Layer)

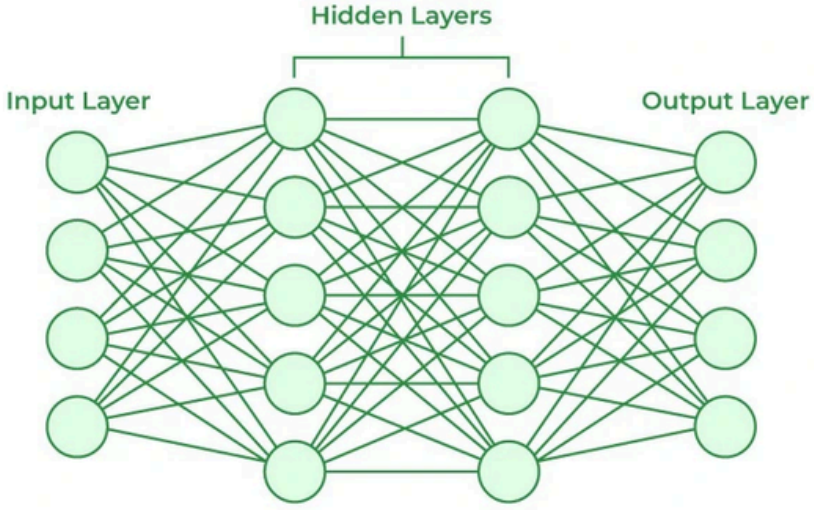
- Nhận thông tin từ bên ngoài vào mạng.
- Số lượng nơ-ron phụ thuộc vào số đặc trưng (features) của dữ liệu.

Lớp ẩn (Hidden Layer)

- Thực hiện các tính toán phức tạp để tìm đặc trưng ẩn trong dữ liệu.
- Có thể có nhiều lớp ẩn khác nhau.

Lớp đầu ra (Output Layer)

- Đưa ra kết quả của mạng.
- Số lượng nơ-ron phụ thuộc vào bài toán:
  – Phân loại nhị phân: 1 nơ-ron.
  – Phân loại đa lớp: Số nơ-ron tương ứng với số lớp.

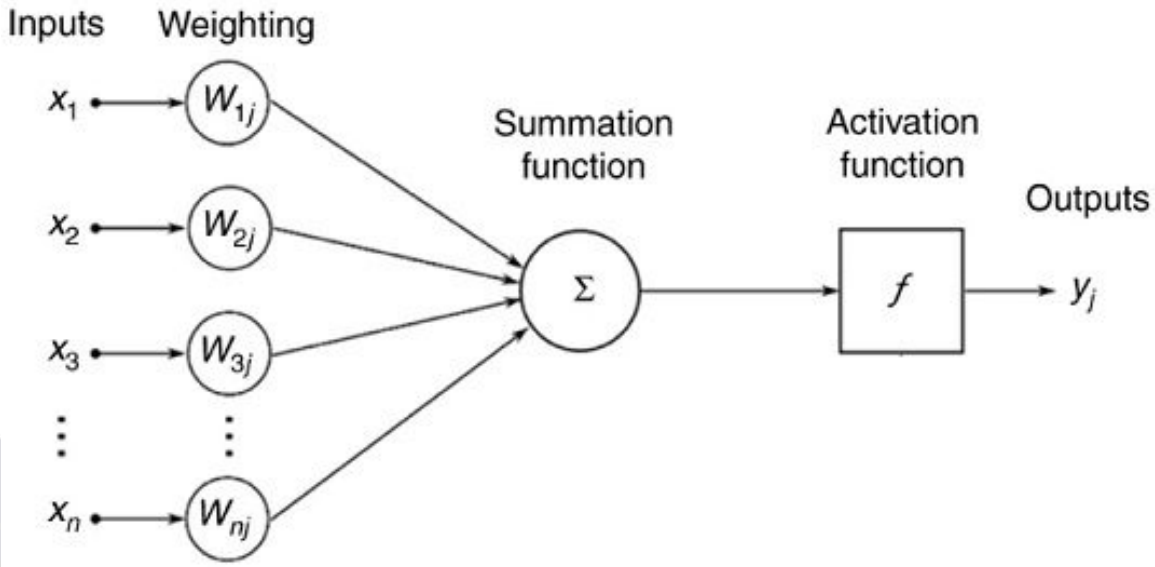


5.3. Nguyên lý hoạt động

Công thức tính toán trong nơ-ron

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

- x_i : Giá trị đầu vào.
- w_i : Trọng số của kết nối.
- b : Giá trị thiên lệch.
- f : Hàm kích hoạt.
- y : Đầu ra của nơ-ron.



5.3. Nguyên lý hoạt động

Hàm kích hoạt

Hàm Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	• Chuyển đổi giá trị đầu vào thành giá trị từ 0 đến 1. • Phù hợp cho bài toán phân loại.
Hàm ReLU	$f(x) = \max(0, x)$	• Giúp mạng học nhanh hơn, đặc biệt trong mô hình lớn.
Hàm Tanh	$f(x) = \frac{2}{1+e^{-2x}} - 1$	• Tương tự Sigmoid, nhưng đầu ra từ -1 đến 1. • Hữu ích cho bài toán liên quan giá trị âm.

5.3. Nguyên lý hoạt động

Quá trình lan truyền tiến (Feedforward)

- Dữ liệu đầu vào truyền từ lớp đầu vào qua lớp ẩn đến lớp đầu ra.
- Mỗi nơ-ron tính toán đầu ra và truyền tiếp.

5.3. Nguyên lý hoạt động

Quá trình lan truyền tiến (Feedforward)

Ví dụ: với mạng đơn giản (một lớp đầu vào, một lớp ẩn, một lớp đầu ra)

Đầu vào:

x_1, x_2 : Các giá trị đầu vào từ lớp đầu vào.

Các trọng số kết nối giữa lớp đầu vào và lớp ẩn:

w_{11}, w_{12} : Trọng số từ x_1, x_2 đến nơ-ron đầu tiên trong lớp ẩn.

w_{21}, w_{22} : Trọng số từ x_1, x_2 đến nơ-ron thứ hai trong lớp ẩn.

$$z_1 = w_{11}x_1 + w_{12}x_2 + b_1$$
$$z_2 = w_{21}x_1 + w_{22}x_2 + b_2$$

Đầu ra lớp ẩn:

$$a_1 = \sigma(z_1)$$

$$a_2 = \sigma(z_2)$$

5.3. Nguyên lý hoạt động

Quá trình lan truyền ngược (Backpropagation)
Cập nhật trọng số dựa trên lỗi giữa đầu ra dự đoán và thực tế.

Sử dụng hàm mất mát, ví dụ: Mean Squared Error (MSE).

$$L = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- y_i : Giá trị thực tế của đầu ra.
- $\hat{y}_i$: Giá trị dự đoán của mạng.

Cập nhật trọng số bằng gradient descent:

$$w_i = w_i - \eta \frac{\partial L}{\partial w_i}$$

- η : Tốc độ học (learning rate).
- $\frac{\partial L}{\partial w_i}$: Đạo hàm của hàm mất mát đối với trọng số.

5.4. Huấn luyện, kết quả và đánh giá thuật toán

Tiền xử lý dữ liệu:

Chuẩn hóa dữ liệu doanh thu về phạm vi [0, 1] sử dụng MinMaxScaler để mô hình học tốt hơn.

Dữ liệu huấn luyện và kiểm tra:

- Dữ liệu huấn luyện: 20 tháng đầu (train).
- Dữ liệu kiểm tra: 4 tháng cuối (test).

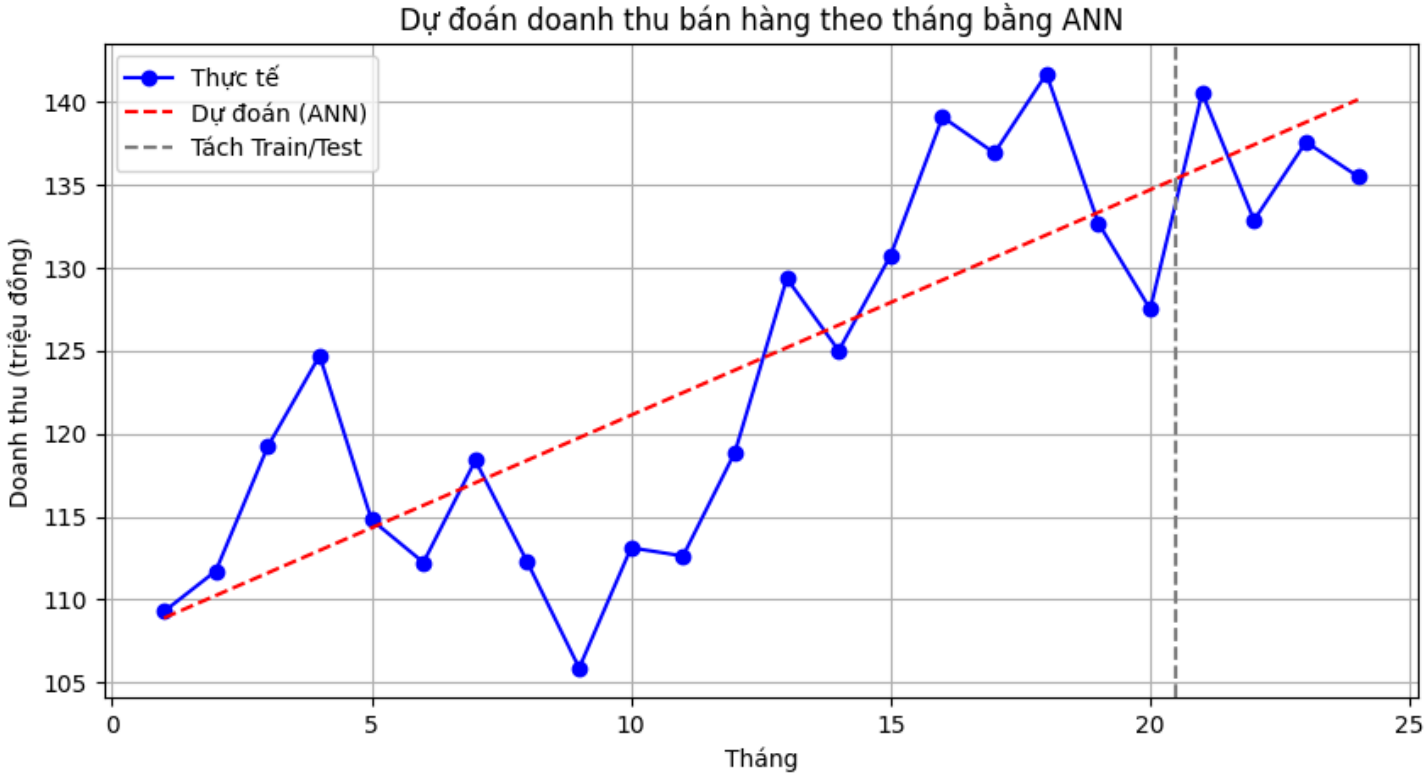
5.4. Huấn luyện, kết quả và đánh giá thuật toán

Mô hình ANN:

Input Layer	Tháng
Hidden Layer 1	16 neurons, activation = ReLU
Hidden Layer 2	8 neurons, activation = ReLU
Output Layer	1 neuron, activation = Linear

5.4. Huấn luyện, kết quả và đánh giá thuật toán

Kết quả Dự đoán Doanh Thu:



5.5. Ưu và nhược điểm

Ưu điểm:

- Linh hoạt và có thể áp dụng cho nhiều bài toán khác nhau.
- Dễ dàng mở rộng để xây dựng các mô hình phức tạp.

Nhược điểm:

- Cần dữ liệu lớn và tài nguyên tính toán mạnh.
- Quá trình huấn luyện có thể mất thời gian và khó tối ưu hóa.

THANK YOU FOR LISTENING