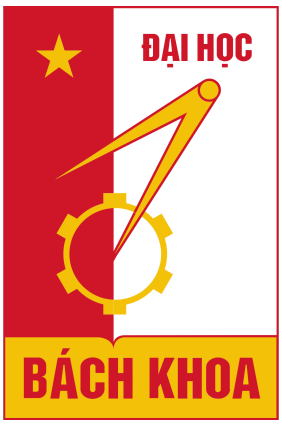
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



Báo cáo BTL Nhập môn Trí tuệ nhân tạo

Nhận diện chữ viết tay

Giảng viên hướng dẫn: Thầy Trần Thế Hùng

Phạm Hải Nam Anh
Trần Đức Hoàng Nam
Bùi Thị Khánh Huyền
Hà Tuấn Hoàng
Đào Sỹ Phúc

Ngày 16 tháng 6 năm 2024

1 Giới thiệu bài toán

Nhận diện chữ viết tay là một trong những lĩnh vực nghiên cứu quan trọng trong ngành trí tuệ nhân tạo và xử lý hình ảnh. Bài toán này nhằm mục đích chuyển đổi văn bản viết tay thành dạng văn bản số mà máy tính có thể hiểu và xử lý được. Đây là một bài toán thách thức do tính đa dạng và phức tạp của chữ viết tay, bao gồm các biến thể về hình dạng, kích thước, và phong cách viết của từng cá nhân.

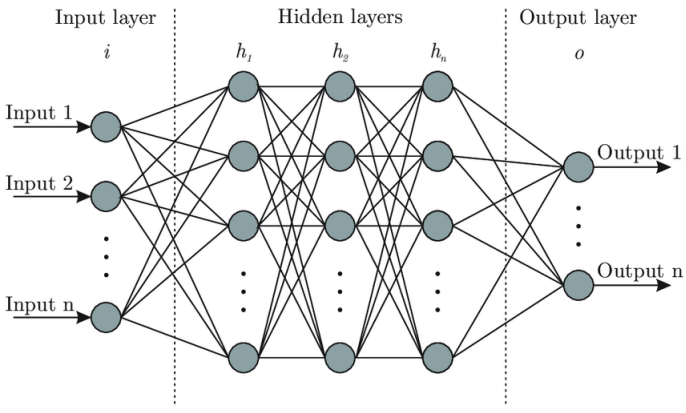
Bài toán nhận diện chữ viết tay có ứng dụng rộng rãi trong nhiều lĩnh vực như tự động hóa nhập liệu, phân loại tài liệu, và hỗ trợ người khuyết tật. Với sự phát triển của công nghệ học sâu (Deep Learning), các mô hình mạng nơ-ron tích chập (Convolutional Neural Network - CNN), mạng nơ-ron hồi quy (Recurrent Neural Network - RNN), và đặc biệt là mạng LSTM (Long Short-Term Memory) đã cho thấy hiệu quả cao trong việc nhận diện chữ viết tay.

Trong báo cáo này, chúng em sẽ trình bày về phương pháp sử dụng mạng CRNN (Convolutional Recurrent Neural Network) kết hợp với hàm mất mát CTC (Connectionist Temporal Classification) và mạng LSTM để giải quyết bài toán nhận diện chữ viết tay. Sự kết hợp này giúp mô hình có thể học và nhận diện các chuỗi ký tự trong văn bản viết tay một cách chính xác và hiệu quả.

2 Cơ Sở Lý Thuyết

2.1 Artifical Neural Network - ANN

Mạng Neural Nhân Tạo (ANNs) là một nhánh của trí tuệ nhân tạo và học máy, được lấy cảm hứng từ cấu trúc và cách thức hoạt động của não người. ANNs được thiết kế để nhận dạng mẫu và giải quyết các vấn đề phức tạp thông qua việc học từ dữ liệu.



Hình 1: Kiến Trúc Một Mạng Thần Kinh Nhân Tạo

Cấu trúc của ANNs:

- **Neurons (Tế bào thần kinh)**: Các đơn vị cơ bản của một ANN, tương tự như tế bào thần kinh sinh học. Mỗi neuron nhận đầu vào, xử lý nó và tạo ra đầu ra.
- **Layers (Các lớp)**: Các neuron được tổ chức thành các lớp:
  - **Input Layer (Lớp đầu vào)**: Lớp đầu tiên, nhận dữ liệu đầu vào.
  - **Hidden Layers (Các lớp ẩn)**: Các lớp trung gian giữa lớp đầu vào và lớp đầu ra, nơi các tính toán được thực hiện. Có thể có nhiều lớp ẩn trong một mạng neural sâu.
  - **Output Layer (Lớp đầu ra)**: Lớp cuối cùng tạo ra đầu ra của mạng.

2.2 RGB và Grayscale

2.2.1 RGB

RGB là một mô hình màu phổ biến trong lĩnh vực deep learning, đặc biệt là trong các ứng dụng liên quan đến hình ảnh và thị giác máy tính. Đặc điểm của RGB trong Deep Learning:

- **Cấu trúc Dữ liệu**: Trong một hình ảnh RGB, mỗi điểm ảnh (pixel) được biểu diễn bằng ba giá trị tương ứng với ba kênh màu: đỏ, xanh lục, và xanh lam. Điều này tạo ra một tensor 3D với các chiều [chiều cao, chiều rộng, kênh màu].
- **Ứng dụng**:
  - **Nhận diện hình ảnh**: Mô hình học sâu như Convolutional Neural Networks (CNNs) sử dụng hình ảnh RGB để nhận diện đối tượng, phân loại hình ảnh, và phát hiện đối tượng.
  - **Thị giác máy tính**: Các ứng dụng như xe tự lái, nhận diện khuôn mặt, và theo dõi chuyển động đều sử dụng hình ảnh RGB để phân tích và xử lý.

2.2.2 Grayscale

Grayscale, hay thang độ xám, là một mô hình màu đơn giản hơn so với RGB, chỉ chứa thông tin về độ sáng của điểm ảnh. Đặc điểm của Grayscale trong Deep Learning:

- **Cấu trúc Dữ liệu**: Một hình ảnh grayscale được biểu diễn bằng một giá trị duy nhất cho mỗi điểm ảnh, tạo ra một tensor 2D với các chiều [chiều cao, chiều rộng].
- **Ứng dụng**:

– **Xử lý ảnh y tế**: Hình ảnh từ các thiết bị y tế như X-quang, MRI thường ở dạng grayscale, giúp các mô hình deep learning phân tích và chẩn đoán bệnh.
– **Nhận diện chữ viết tay**: Hình ảnh chữ viết tay thường được chuyển sang grayscale để giảm phức tạp và tập trung vào hình dạng của các ký tự.

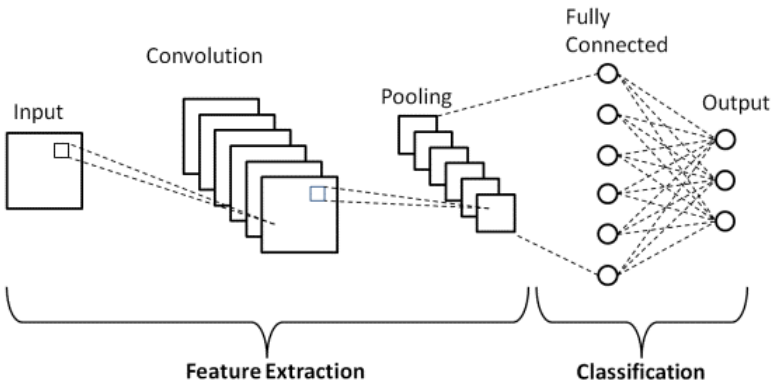
2.3 Mạng CNN

Convolutional Neural Network (CNN hoặc ConvNet) là một loại mạng nơ-ron nhân tạo đặc biệt hiệu quả trong việc xử lý dữ liệu có cấu trúc lưới, như hình ảnh và video. CNN đã trở thành một công cụ quan trọng trong lĩnh vực học sâu (deep learning), đặc biệt là trong nhận dạng hình ảnh, phân loại ảnh và các ứng dụng liên quan đến thị giác máy tính.

Cấu trúc của CNN: CNN bao gồm một loạt các lớp đặc biệt, mỗi lớp có chức năng cụ thể trong việc trích xuất và xử lý các đặc điểm từ dữ liệu đầu vào: Lớp tích chập, lớp gộp, lớp kết nối đầy đủ

2.3.1 Lớp tích chập (Convolution Layer):

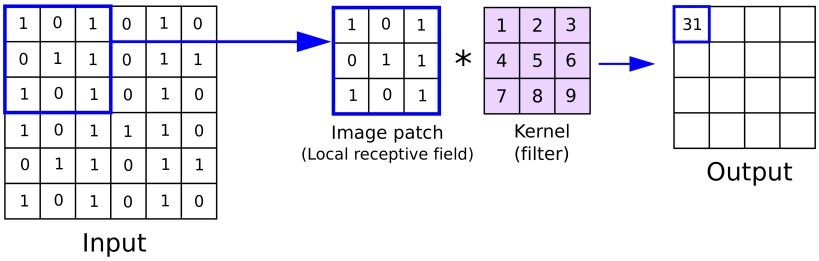
Lớp tích chập là thành phần cơ bản và quan trọng nhất trong mạng nơ-ron tích chập (CNN). Nhiệm vụ chính của lớp này là trích xuất các đặc trưng từ dữ liệu đầu vào, thường là hình ảnh. Quá trình tích chập sử dụng các bộ lọc (filters) để phát hiện các đặc điểm như cạnh, góc, và các mẫu phức tạp hơn trong hình ảnh.



Hình 2: Kiến Trúc Một Mạng Tích Chập

Phép Tích Chập (Convolution Operation): Phép tích chập là quá trình áp dụng một bộ lọc nhỏ (kernel) lên toàn bộ dữ liệu đầu vào để tạo ra một bản đồ đặc trưng (feature map). Mỗi bộ lọc sẽ trích xuất một đặc điểm cụ thể từ dữ liệu đầu vào.

- **Kernel (Bộ lọc):** Một ma trận nhỏ với các giá trị trọng số, có thể là 3x3, 5x5, hoặc kích thước khác.
- **Stride (Bước nhảy):** Số lượng pixel di chuyển của bộ lọc trên dữ liệu đầu vào. Stride lớn hơn sẽ làm giảm kích thước của feature map.
- **Padding (Đệm):** uá trình thêm các giá trị (thường là 0) xung quanh biên của dữ liệu đầu vào để kiểm soát kích thước của đầu ra sau tích chập. Padding có thể là "same" (kích thước đầu ra bằng kích thước đầu vào) hoặc "valid" (không thêm padding).



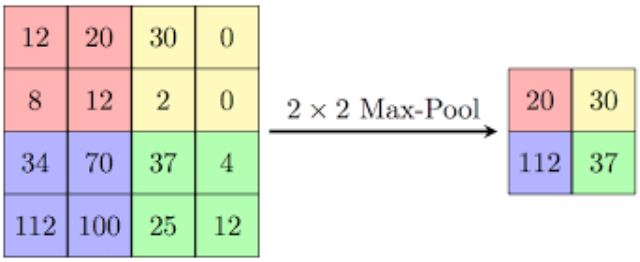
Hình 3: Convolutional Layer

2.3.2 Lớp gộp (Pooling Layer):

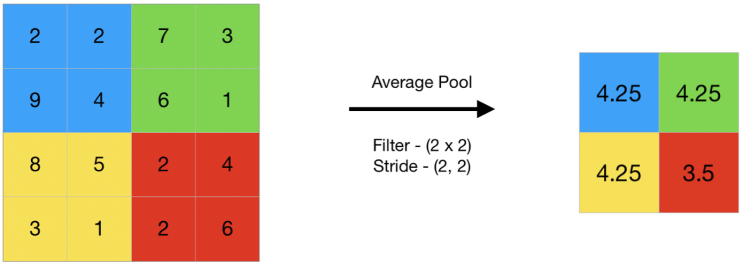
Phép gộp là quá trình giảm kích thước của các bản đồ đặc trưng (feature maps) từ lớp tích chập để giảm số lượng tham số và tính toán, cũng như để kiểm soát hiện tượng overfitting. Có hai loại phép gộp phổ biến là Max Pooling và Average Pooling.

- **Max Pooling:** Max pooling lấy giá trị lớn nhất từ mỗi vùng nhỏ của feature map. Điều này giúp giữ lại các đặc điểm quan trọng nhất trong khi giảm kích thước.
  - **Kích thước cửa sổ (Window size):** Kích thước của vùng được chọn để lấy giá trị cực đại, thường là 2x2 hoặc 3x3.
  - **Stride:** Tương tự như trong lớp tích chập, quyết định số lượng pixel di chuyển của cửa sổ trên feature map.
- **Average Pooling:** Average pooling lấy giá trị trung bình của mỗi vùng nhỏ trong feature map. Điều này giúp làm mịn các đặc trưng và giảm kích thước của feature map.
  - **Kích thước cửa sổ (Window size):** Kích thước của vùng được chọn để lấy giá trị cực đại, thường là 2x2 hoặc 3x3.

– **Stride:** Tương tự như trong lớp tích chập, quyết định số lượng pixel di chuyển của cửa sổ trên feature map.



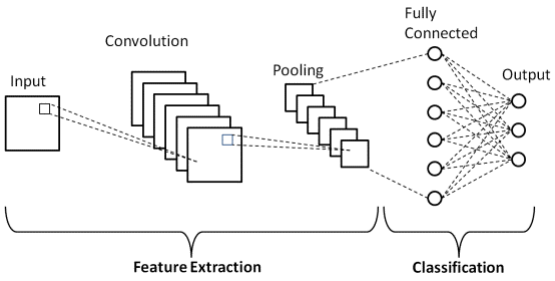
Hình 4: Max Pooling



Hình 5: Average Pooling

2.3.3 Lớp kết nối đầy đủ (Fully Connected Layer)

Lớp này giống như các lớp trong mạng nơ-ron truyền thống, nơi mọi nút trong lớp được kết nối với mọi nút trong lớp tiếp theo. Nó thường được sử dụng ở phần cuối của mạng để dự đoán các lớp hoặc nhãn.



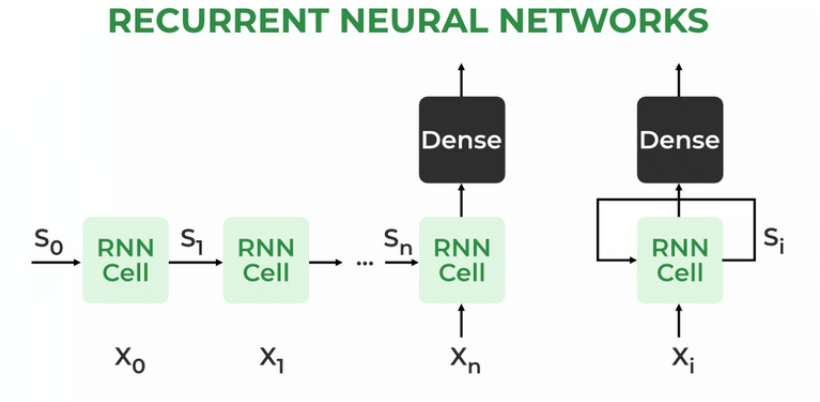
Hình 6: Lớp kết nối đầy đủ - Fully Connected

2.4 Mạng RNN

Recurrent Neural Network (RNN) là một loại mạng nơ-ron nhân tạo được thiết kế để xử lý dữ liệu tuần tự, nơi các kết quả trước đó có thể ảnh hưởng đến các kết quả sau này. RNN rất hiệu quả trong các tác vụ liên quan đến chuỗi thời gian tuần tự.

2.4.1 RNN truyền thống

RNN truyền thống có khả năng lưu trữ thông tin từ các bước trước đó trong chuỗi dữ liệu và sử dụng thông tin này để ảnh hưởng đến kết quả hiện tại. Điều này đạt được bằng cách sử dụng các kết nối lặp lại, cho phép thông tin được "nhớ" và chuyển tiếp qua các bước trong chuỗi.



Hình 7: Kiến trúc một mạng RNN truyền thống

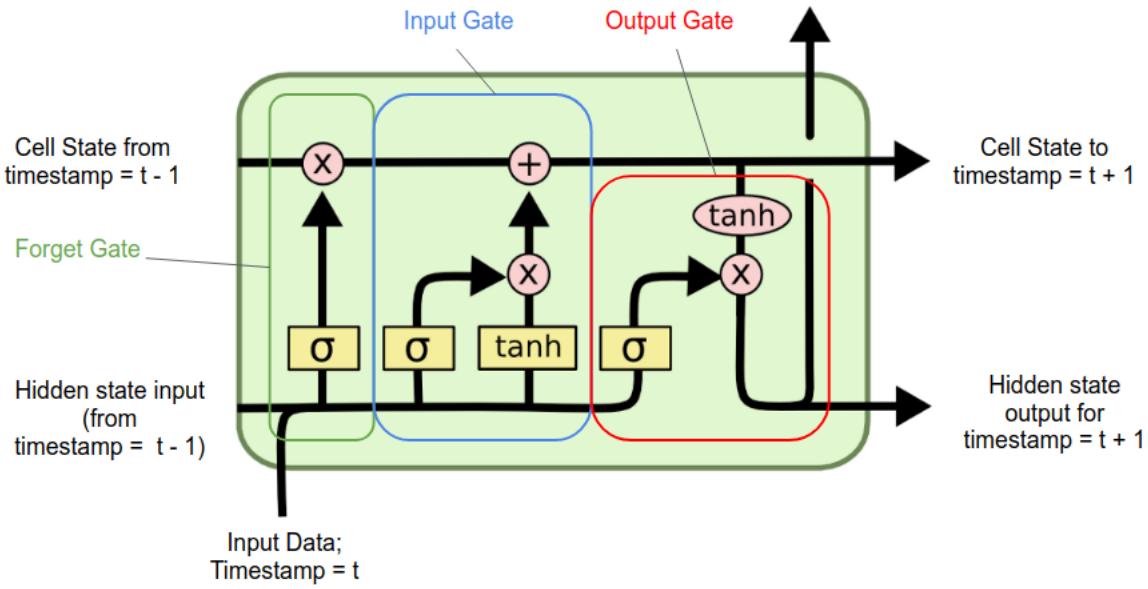
Các thành phần chính của RNN truyền thống:

- **Đầu vào (Input):** Chuỗi dữ liệu đầu vào, thường được biểu diễn dưới dạng vector.
- **Trạng thái ẩn (Hidden State):** Một vector lưu trữ thông tin từ các bước trước đó trong chuỗi. Trạng thái ẩn được cập nhật sau mỗi bước thời gian dựa trên đầu vào hiện tại và trạng thái ẩn trước đó.
- **Đầu ra (Output):** Kết quả của RNN tại mỗi bước thời gian, có thể được sử dụng trực tiếp hoặc chuyển qua các lớp khác để xử lý thêm.

Nhược điểm của RNN truyền thống: RNN truyền thống gặp khó khăn trong việc xử lý các chuỗi dài do hiện tượng tiêu biến gradient (vanishing gradient), làm giảm khả năng học các mối quan hệ dài hạn trong dữ liệu.

2.4.2 LSTM (Long Short-Term Memory)

LSTM là một biến thể của RNN được thiết kế để khắc phục vấn đề vanishing gradient bằng cách duy trì thông tin dài hạn từ các bước trước đó trong chuỗi, ngay cả khi chúng cách xa nhau thông qua bộ nhớ dài hạn. LSTM sử dụng các cổng đặc biệt để kiểm soát dòng thông tin qua chuỗi dữ liệu, giúp mạng "nhớ" và "quên" thông tin một cách chọn lọc.



Hình 8: Cấu trúc một cell LSTM

Các cổng trong LSTM hoạt động cùng nhau để chọn lọc và duy trì thông tin quan trọng, trong khi loại bỏ những thông tin không cần thiết.
Các thành phần chính của LSTM:

- **Cell State (Trạng thái cell):** Lưu trữ thông tin dài hạn qua chuỗi thời gian.
- **Các cổng:**
  - **Cổng quên (Forget Gate):** Quyết định thông tin nào sẽ bị quên từ trạng thái cell.
  - **Cổng nhập (Input Gate):** Quyết định thông tin nào sẽ được lưu trữ vào trạng thái cell.

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

- i_t: Đầu ra của cổng nhập tại thời điểm t. - W_i, b_i: Trọng số và bias của cổng nhập.

– **Cổng đầu ra (Output Gate):** Quyết định thông tin nào sẽ được xuất ra từ trạng thái ẩn.

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

- o_t: Đầu ra của cổng đầu ra tại thời điểm t. - W_o, b_o: Trọng số và bias của cổng đầu ra.

- **Cập nhật trạng thái cell:**

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

- C_t: Trạng thái cell tại thời điểm t. - $\tilde{C}_t$: Trạng thái cell mới được tạo từ đầu vào hiện tại.

- **Cập nhật trạng thái ẩn:**

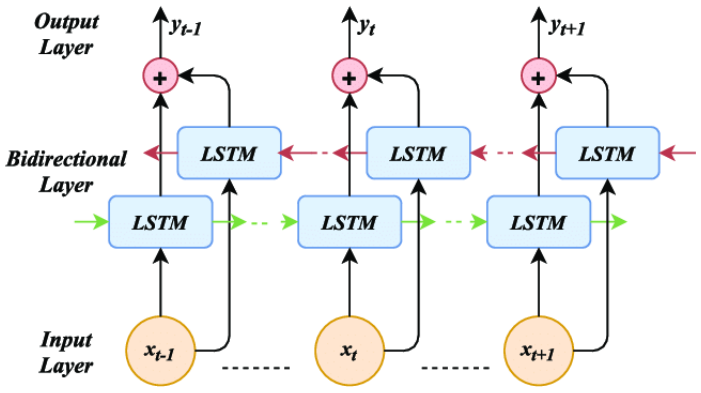
$$h_t = o_t \cdot \tanh(C_t)$$

- h_t: Trạng thái ẩn tại thời điểm t. - tanh: Hàm kích hoạt tanh.

Ưu điểm của LSTM: LSTM có khả năng lưu trữ và duy trì thông tin dài hạn tốt hơn so với RNN truyền thống, làm cho chúng trở nên lý tưởng cho các tác vụ liên quan đến chuỗi thời gian dài và các mối quan hệ phức tạp trong dữ liệu.

2.4.3 Bidirectional LSTM (BiLSTM)

Bidirectional LSTM là một biến thể của LSTM được thiết kế để cải thiện khả năng nắm bắt ngữ cảnh trong chuỗi dữ liệu bằng cách xử lý thông tin theo cả hai hướng: từ quá khứ đến tương lai (forward) và từ tương lai đến quá khứ (backward). Điều này đặc biệt hữu ích trong các tác vụ mà ngữ cảnh xung quanh một từ hoặc một phần của chuỗi có ý nghĩa quan trọng, chẳng hạn như trong xử lý ngôn ngữ tự nhiên (NLP).



Hình 9: Cấu trúc của Bidirectional LSTM

Các thành phần chính của BiLSTM:

- **LSTM Forward:** Một lớp LSTM xử lý chuỗi dữ liệu theo thứ tự từ đầu đến cuối.
- **LSTM Backward:** Một lớp LSTM khác xử lý chuỗi dữ liệu theo thứ tự ngược lại, từ cuối đến đầu.
- **Kết hợp Đầu ra:** Đầu ra từ cả hai lớp LSTM (forward và backward) được kết hợp lại để tạo ra một vector đặc trưng phong phú hơn.

Nguyên lý hoạt động: Bidirectional LSTM hoạt động bằng cách chạy hai mạng LSTM riêng biệt trên chuỗi dữ liệu đầu vào. Một mạng chạy theo hướng tiến (forward), từ đầu đến cuối chuỗi, trong khi mạng kia chạy theo hướng lùi (backward), từ cuối về đầu. Kết quả đầu ra của mỗi thời điểm trong chuỗi là sự kết hợp của cả hai hướng này, cung cấp thông tin từ cả quá khứ và tương lai của chuỗi dữ liệu.

Ưu điểm của BiLSTM:

- **Ngữ cảnh Đầy đủ:** BiLSTM có khả năng nắm bắt ngữ cảnh từ cả hai hướng, giúp hiểu rõ hơn về ngữ cảnh xung quanh mỗi phần tử trong chuỗi dữ liệu.
- **Hiệu suất Cao hơn:** Trong nhiều tác vụ, BiLSTM thường cho kết quả tốt hơn so với LSTM đơn hướng vì nó tận dụng thông tin từ cả trước và sau của chuỗi dữ liệu.

3 Tập dữ liệu sử dụng

3.1 Nguồn gốc dữ liệu

Tập dữ liệu sử dụng trong dự án này được lấy từ trang web Kaggle:

- Handwriting Recognition

3.2 Cấu trúc dữ liệu

Tập dữ liệu này bao gồm các thư mục chứa hình hình ảnh chữ viết tay và các tệp .csv (dạng bảng) tương ứng. Cụ thể như sau:

- **Hình ảnh**: Các hình ảnh được lưu trữ ở định dạng JPEG, đặt tên theo cú pháp thư mục_số thứ tự.jpg, chứa chữ viết tay là tên người.
- **File .csv**: Mỗi tệp gồm 2 cột là 'FILENAME' và 'IDENTITY'. Cột 'FILENAME' chứa tên hình ảnh và cột 'IDENTITY' chứa nhãn viết tay tương ứng có trong hình ảnh.

3.3 Số lượng và chất lượng dữ liệu

Tập dữ liệu này bao gồm hơn bốn trăm nghìn tên viết tay được thu thập thông qua các dự án từ thiện. Cụ thể:

- **Tổng số lượng dữ liệu**: Tập dữ liệu này có tổng cộng 413,823 hình ảnh chữ viết tay, bao gồm 206,799 tên riêng và 207,024 họ.
- **Phân chia dữ liệu**: Dữ liệu được chia thành ba tập chính:
  - Tập train: 331,059 ví dụ
  - Tập test: 41,382 ví dụ
  - Tập validate: 41,382 ví dụ

  Một ví dụ bao gồm cả hình ảnh và nhãn của nó

3.4 Tiền xử lý dữ liệu

Việc phân tích và tiền xử lý dữ liệu giúp thu được những đặc trưng của bộ dữ liệu, đảm bảo rằng mô hình được train trên một tập dữ liệu sạch và cân bằng, từ đó cải thiện độ chính xác và hiệu quả của hệ thống nhận diện chữ viết tay. Đầu tiên, phân tích tập dữ liệu, ta có những thống kê như sau:

- **Những ví dụ có nhãn trong cột "IDENTITY" mang giá trị rỗng**:
  - Tập Train: 565
  - Tập Validation: 78
- **Số giá trị riêng biệt của nhãn "IDENTITY" trong tập Train**: 100539
- **Số ví dụ không thể đọc (mang nhãn "UNREADABLE")**: 102
- **Số ví dụ rỗng (mang nhãn "EMPTY")**: 1796
- **Số ví dụ được đánh nhãn bằng chữ in thường**: 17

Sau đó, ta sẽ tiến hành loại bỏ những ví dụ có nhãn không thể đọc, rỗng, và sửa nhãn những ví dụ in thường thành in hoa. Việc quy về nhãn nhãn in thường giúp giảm kích cỡ "từ điển" những kí tự cần dự đoán.
Để đảm bảo các hình ảnh có kích thước cố định và được chuẩn hóa, giúp mô hình học sâu có thể tiếp nhận và xử lý dữ liệu một cách hiệu quả, ta tiến hành

- Chuyển đổi hình ảnh sang mảng NumPy giúp dễ dàng thao tác trên hình ảnh.
- Cắt hình ảnh nếu kích thước lớn hơn giới hạn đảm bảo tất cả các hình ảnh có kích thước phù hợp.
- Xoay hình ảnh để phù hợp với định dạng đầu vào của model.

- Chuyển đổi sang tensor PyTorch và chuẩn hóa giá trị pixel về [0, 1] để chuẩn bị hình ảnh nạp vào model.

Đồng thời, ta cần mã hóa nhãn để dễ dàng sử dụng các nhãn trong quá trình huấn luyện mô hình.

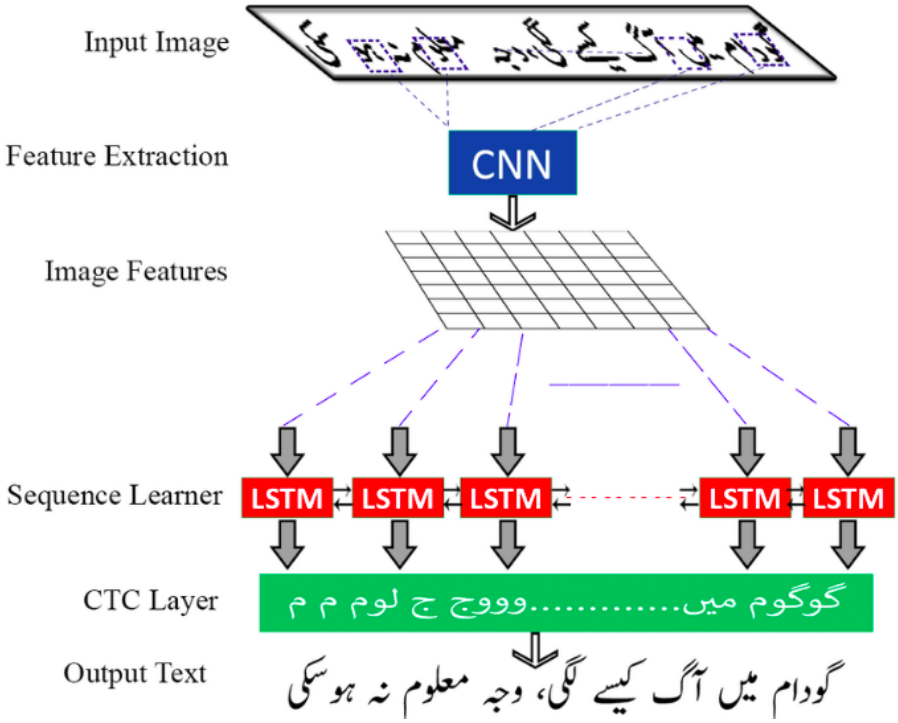
- **Từ điển mã hóa**: `alphabet = u"ABCDEFGHIJKLMNOPQRSTUVWXYZ-'' "`
- **Thực hiện**: Chuyển đổi mỗi ký tự thành index của dãy alphabet

Với các chuỗi nhãn có độ dài khác nhau, để đảm bảo rằng các nhãn có cùng độ dài, ta tiến hành đệm (pad)

- Tìm độ dài tối đa của nhãn trong lô dữ liệu để xác định kích thước đệm cần thiết.
- Thêm các ký tự <BLANK> còn thiếu để các nhãn có cùng độ dài

Cuối cùng, ta sẽ tạo các lô dữ liệu một cách ngẫu nhiên và quản lý việc nạp dữ liệu vào mô hình trong quá trình huấn luyện.

4 Phương pháp



Hình 10: Kiến trúc CRNN + CTC

Để giải quyết bài toán Nhận Dạng Chữ Viết Tay, nhóm chúng em quyết định chọn kiến trúc CRNN (Convolutional Recurrent Neural Networt) kết hợp với CTC (Connectionist Temporal Classification). Kiến trúc CRNN (Convolutional Recurrent Neural Network) kết hợp với CTC (Connectionist Temporal Classification) là một giải pháp mạnh mẽ và tiên tiến trong lĩnh vực xử lý chuỗi và nhận dạng hình ảnh, đặc biệt là trong các bài toán nhận dạng văn bản từ hình ảnh. Sự kết hợp này tận dụng ưu điểm của cả CNN (Convolutional Neural Network) và RNN (Recurrent Neural Network) cùng với thuật toán CTC để xử lý các chuỗi có độ dài không cố định mà không cần gán nhãn chi tiết cho từng khung hình.

Kiến trúc CRNN + CTC bao gồm ba thành phần chính: lớp tích chập (convolutional layers), lớp hồi quy (recurrent layers), và lớp chuyển đổi (transcription layer).

1. **Lớp Tích Chập (Convolutional Layers):**

- **Chức năng:** Tự động trích xuất các đặc trưng từ mỗi hình ảnh đầu vào, giống như cách một mô hình CNN hoạt động.
- **Cách hoạt động:**
  - **Convolutional Layer:** Các lớp convolutional có nhiệm vụ phát hiện các đặc trưng cơ bản như cạnh, góc, và các mẫu phức tạp hơn trong hình ảnh.
  - **Pooling Layer:** Các lớp pooling (thường là max-pooling) giảm kích thước không gian của các đặc trưng, giúp giảm số lượng tham số và tính toán, đồng thời giữ lại các thông tin quan trọng.
  - **Flattening:** Các bản đồ đặc trưng từ các lớp tích chập và pooling được làm phẳng thành một chuỗi các đặc trưng tuần tự, tương ứng với các khung thời gian.

2. **Lớp Hồi Quy (Recurrent Layers):**
   - **Chức năng:** Dự đoán phân phối nhãn cho mỗi khung hình của chuỗi đặc trưng được xuất ra từ các lớp tích chập.
   - **Cách hoạt động:**
     - **Recurrent Layer:** Sử dụng các mạng hồi quy như LSTM (Long Short-Term Memory) hoặc GRU (Gated Recurrent Unit) để xử lý chuỗi đặc trưng và nắm bắt thông tin theo thứ tự thời gian.
     - **Bidirectional RNN:** Để tận dụng thông tin từ cả quá khứ và tương lai, mô hình thường sử dụng LSTM hai chiều, giúp tăng độ chính xác trong dự đoán.

3. **Lớp Chuyển Đổi (Transcription Layer):**
   - **Chức năng:** Dịch các dự đoán cho từng khung hình từ lớp hồi quy thành chuỗi nhãn cuối cùng.
   - **Cách hoạt động:**
     - **CTC (Connectionist Temporal Classification):** CTC là một thuật toán giúp mô hình có thể học từ dữ liệu có nhãn không cố định mà không cần gán nhãn chi tiết cho từng khung hình. CTC cho phép mô hình dự đoán chuỗi đầu ra có độ dài thay đổi và chèn các ký tự "blank" (trống) để xử lý các khoảng trắng và sự không đồng bộ giữa chuỗi đầu vào và chuỗi đầu ra.

4.1 Kiến trúc mạng CRNN

Convolutional Recurrent Neural Network (CRNN) là một mô hình học sâu phức tạp, đặc biệt phù hợp cho các tác vụ liên quan đến dữ liệu tuần tự, chẳng hạn như nhận dạng văn bản viết tay. CRNN tận dụng các điểm mạnh của **Convolutional Neural Network (CNN)** và **Recurrent Neural Network (RNN)** để xử lý hiệu quả các phụ thuộc về không gian và thời gian vốn có trong các chuỗi viết tay. Kiến trúc bắt đầu bằng một loạt các lớp tích chập, có

khả năng nắm bắt các mô hình cục bộ và phân cấp không gian trong dữ liệu hình ảnh một cách thành thạo. Các lớp này trích xuất các tính năng cấp cao từ hình ảnh đầu vào, chuyển đổi hiệu quả dữ liệu pixel thô thành cách trình bày trừu tượng và nhiều thông tin hơn. Theo sau các lớp tích chập, các tính năng được trích xuất sau đó sẽ được xử lý bởi các lớp lặp lại, thường sử dụng các đơn vị **Long Short-Term Memory (LSTM)**, vượt trội trong việc mô hình hóa các phụ thuộc tầm xa và trình tự thời gian. Quá trình xử lý tuần tự này cho phép CRNN duy trì và sử dụng thông tin theo ngữ cảnh trong toàn bộ chiều dài của chuỗi đầu vào, điều này rất quan trọng để diễn giải chính xác văn bản viết tay có nhiều kiểu dáng, khoảng cách và căn chỉnh khác nhau. Giai đoạn cuối cùng của kiến trúc CRNN thường bao gồm lớp phiên mã, chẳng hạn như **Connectionist Temporal Classification (CTC)**, giúp dịch đầu ra tuần tự của RNN thành văn bản có thể đọc được bằng cách căn chỉnh các chuỗi ký tự được dự đoán với hình ảnh đầu vào. Mô hình có thể huấn luyện từ đầu đến cuối này không chỉ nâng cao độ mạnh mẽ và độ chính xác của hệ thống nhận dạng mà còn đơn giản hóa quá trình huấn luyện bằng cách loại bỏ nhu cầu về các ký tự được phân đoạn trước. Về bản chất, CRNN thể hiện một cách tiếp cận mạnh mẽ và linh hoạt để nhận dạng văn bản viết tay, kết hợp khả năng trích xuất đặc điểm không gian của CNN với khả năng mô hình hóa trình tự của RNN, từ đó giải quyết hiệu quả sự phức tạp của các đầu vào viết tay có độ dài thay đổi và phụ thuộc vào ngữ cảnh. Sự tích hợp các thành phần tích chập và lặp lại trong một khuôn khổ thống nhất này khiến CRNN trở thành nền tảng trong lĩnh vực nhận dạng ký tự quang học (OCR), thúc đẩy những tiến bộ trong cả nghiên cứu học thuật và ứng dụng thực tế.

Các Ưu Điểm của CRNN:

1. **Học Từ Nhãn Chuỗi:** Có thể học trực tiếp từ các nhãn chuỗi (ví dụ: từ), không cần các chú thích chi tiết (ví dụ: ký tự).

2. **Đặc Trưng Thông Tin Từ Dữ Liệu Hình Ảnh:** Giữ nguyên khả năng của DCNN trong việc học các đặc trưng thông tin từ dữ liệu hình ảnh, không cần các đặc trưng được tạo thủ công hay các bước tiền xử lý như nhị phân hóa/segmentation.

3. **Sản Xuất Chuỗi Nhãn:** Có khả năng tạo ra một chuỗi các nhãn như RNN.

4. **Không Bị Giới Hạn Bởi Độ Dài:** Không bị giới hạn bởi độ dài của các đối tượng dạng chuỗi, chỉ yêu cầu chuẩn hóa chiều cao trong cả giai đoạn huấn luyện và thử nghiệm.

5. **Hiệu Suất Cao:** Đạt hiệu suất cao hoặc cạnh tranh với các phương pháp trước đó trên các văn bản cảnh (nhận dạng từ).

6. **Tiết Kiệm Bộ Nhớ:** Chứa ít tham số hơn so với mô hình DCNN tiêu chuẩn, tiết kiệm không gian lưu trữ.

4.2 CTC Loss

Connectionist Temporal Classification (CTC) là một thuật toán được thiết kế để giải quyết vấn đề về sự không tương thích về độ dài giữa chuỗi đầu vào và chuỗi đầu ra trong các mô hình nhận diện chuỗi, đặc biệt hữu ích trong bài toán nhận diện chữ viết tay.

Trong bài toán này, chuỗi đầu vào là ảnh chụp chữ viết tay, thường có độ dài khác nhau và được biểu diễn dưới dạng ma trận pixel, trong khi chuỗi đầu ra là văn bản tương ứng, thường có độ dài ngắn hơn và có định dạng khác biệt so với ảnh đầu vào. Điều này dẫn đến vấn đề là mô hình có thể nhận diện sai và lặp lại các ký tự khi chiều dài của chúng trong ảnh chụp quá lớn.

CTC giải quyết vấn đề này bằng cách thêm các ký tự khoảng trống (blank) vào chuỗi đầu ra thu được từ mạng CRNN (Convolutional Recurrent Neural Network). Cụ thể, trước khi dự đoán các ký tự đầu ra, hình ảnh đầu vào được chia thành nhiều khung nhỏ theo chiều ngang. Mỗi khung chứa một phần của hình ảnh và có thể bao gồm khoảng trắng hoặc một số ký tự.

Mạng RNN (Recurrent Neural Network) sau đó sẽ xử lý từng khung hình và đưa ra dự đoán xác suất của từng ký tự hoặc khoảng trắng cho mỗi khung. Chuỗi kết quả từ RNN là một chuỗi các khung hình với xác suất dự đoán cho mỗi ký tự mà nó chứa.

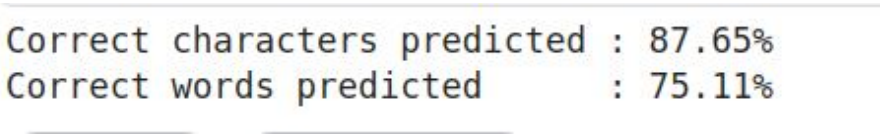
CTC hoạt động như một hàm mất mát (Loss function) để tính toán sự khác biệt giữa chuỗi ký tự đã dự đoán và chuỗi ký tự mục tiêu. Nó thực hiện điều này bằng cách xem xét tất cả các chuỗi khả dĩ có thể ánh xạ từ chuỗi đầu vào sang chuỗi đầu ra mục tiêu, và sau đó tối ưu hóa xác suất của chuỗi mục tiêu thực sự trong không gian tất cả các chuỗi khả dĩ.

Để cụ thể hơn, CTC sử dụng một kỹ thuật gọi là "labeling with blanks" (gán nhãn với khoảng trắng). Trong đó, các ký tự dự đoán có thể bao gồm các khoảng trắng ở giữa, giúp mô hình có khả năng bỏ qua những phần không liên quan trong chuỗi đầu vào. Ví dụ, nếu chuỗi mục tiêu là "HELLO" và mạng RNN dự đoán "H-EE-LL-O" (trong đó "-" là khoảng trắng), CTC sẽ gán nhãn cho chuỗi này và loại bỏ các khoảng trắng để so sánh với chuỗi mục tiêu.

Việc sử dụng CTC làm hàm mất mát giúp mô hình CRNN học cách căn chỉnh chính xác giữa đầu vào và đầu ra mà không cần phải có một sự căn chỉnh thủ công trước đó. Điều này đặc biệt quan trọng trong các bài toán nhận diện chữ viết tay, nơi mà mỗi mẫu chữ có thể có hình dạng và kích thước khác nhau.

5 Đánh giá

Sau khi train model và test lại với bộ test, ta có kết quả như sau:



Hình 11: Kết quả độ chính xác

Kết quả test cho thấy model hiện tại đang hoạt động tương đối tốt. Đối với những test có độ dài quá lớn, model đang chưa thể hiện độ chính xác cao.

6 Kết luận

Trong báo cáo này, chúng em đã trình bày về bài toán nhận diện chữ viết tay và cách tiếp cận sử dụng mô hình Convolutional Recurrent Neural Network (CRNN) kết hợp với hàm mất mát Connectionist Temporal Classification (CTC). Quá trình này bao gồm từ việc thu thập và tiền xử lý dữ liệu, thiết kế và huấn luyện mô hình, cho đến đánh giá kết quả đạt được.

Kết quả thử nghiệm cho thấy mô hình CRNN kết hợp với CTC đã đạt được hiệu suất cao trong việc nhận diện chữ viết tay, đặc biệt là trong việc xử lý các chuỗi ký tự có độ dài thay đổi và không yêu cầu phân đoạn trước các ký tự. Mô hình đã thể hiện khả năng mạnh mẽ trong việc nhận diện chính xác văn bản viết tay và có thể áp dụng trong nhiều ứng dụng thực tiễn như tự động hóa nhập liệu, phân loại tài liệu, và hỗ trợ người khuyết tật.

Tuy nhiên, cũng cần lưu ý rằng mô hình hiện tại vẫn chưa đạt độ chính xác cao đối với các chuỗi ký tự quá dài và phức tạp. Để cải thiện hiệu suất trong tương lai, có thể thực hiện một số biện pháp như sau:

1. **Tăng cường dữ liệu huấn luyện**: Thu thập thêm dữ liệu chữ viết tay đa dạng để mô hình có thể học và tổng quát hóa tốt hơn.
2. **Tối ưu hóa kiến trúc mô hình**: Thử nghiệm với các kiến trúc mạng khác nhau hoặc các kỹ thuật học sâu tiên tiến hơn để cải thiện hiệu suất.
3. **Điều chỉnh tham số**: Tinh chỉnh các tham số của mô hình để đạt được hiệu suất tối ưu.
4. **Sử dụng kỹ thuật học tăng cường**: Áp dụng các kỹ thuật như học tăng cường để cải thiện khả năng dự đoán của mô hình trong các tình huống khó.

Cuối cùng, nhận diện chữ viết tay là một lĩnh vực đầy thách thức và tiềm năng trong ngành trí tuệ nhân tạo và xử lý hình ảnh. Chúng em hy vọng rằng báo cáo này sẽ đóng góp vào việc phát triển và ứng dụng các công nghệ nhận diện chữ viết tay trong thực tiễn, giúp nâng cao hiệu quả và tự động hóa các quy trình xử lý văn bản.

Chúng em xin chân thành cảm ơn sự hướng dẫn và hỗ trợ từ các thầy trong quá trình thực hiện dự án này.