

ĐẠI HỌC HUẾ  
TRƯỜNG ĐẠI HỌC KHOA HỌC  
KHOA CÔNG NGHỆ THÔNG TIN

\*\*\*



KHÓA LUẬN  
TỐT NGHIỆP ĐẠI HỌC

Đề tài:

TÌM HIỂU VỀ NHẬN DIỆN VĂN BẢN SỬ  
DỤNG CRNN VÀ CTC LOSS

Sinh viên thực hiện: **NGUYỄN NGỌC QUANG HUY**

Khóa: **K44-HỆ CHÍNH QUY**

Huế, 06 - 2024

ĐẠI HỌC HUẾ  
TRƯỜNG ĐẠI HỌC KHOA HỌC  
KHOA CÔNG NGHỆ THÔNG TIN  
\*\*\*



**KHÓA LUẬN  
TỐT NGHIỆP ĐẠI HỌC  
NGÀNH CÔNG NGHỆ THÔNG TIN**

Đề tài:

**TÌM HIỂU VỀ NHẬN DIỆN VĂN BẢN SỬ  
DỤNG CRNN VÀ CTC LOSS**

Sinh viên thực hiện: **NGUYỄN NGỌC QUANG HUY**

Khóa: **K44-HỆ CHÍNH QUY**

Giáo viên hướng dẫn: **TS. Nguyễn Ngọc Thủy**

Huế, 06 - 2024

# LỜI CAM ĐOAN

Tôi xin cam đoan đồ án này là công trình nghiên cứu riêng của tôi, không sao chép ở bất kỳ công trình khoa học nào trước đây. Những phần có sử dụng tài liệu trong đồ án sẽ ghi rõ tên tài liệu trong phần tài liệu tham khảo. Các kết quả nêu trong đồ án có nguồn gốc rõ ràng và được trích dẫn đầy đủ.

Tôi xin hoàn toàn chịu trách nhiệm về đồ án này.

Thừa Thiên Huế, ngày 1 tháng 6 năm 2024

**Sinh Viên**

**Nguyễn Ngọc Quang Huy**

# LỜI CẢM ƠN

Trong suốt quá trình học tập và nghiên cứu để hoàn thành khóa luận này, tôi đã nhận được sự giúp đỡ và ủng hộ nhiệt tình từ nhiều người. Để bày tỏ lòng biết ơn sâu sắc, tôi xin gửi lời cảm ơn chân thành đến các thầy cô giảng viên trong khoa Công Nghệ Thông Tin của Trường Đại Học Khoa Học Huế đã truyền đạt cho tôi những kiến thức quý báu và những kinh nghiệm thực tiễn trong suốt thời gian học tập tại trường. Đặc biệt, tôi xin gửi lời cảm ơn sâu sắc đến thầy Nguyễn Ngọc Thủy, người đã tận tình hướng dẫn, động viên và góp ý cho tôi trong quá trình thực hiện khóa luận này. Sự nhiệt huyết và tận tâm của thầy đã giúp tôi vượt qua nhiều khó khăn và hoàn thành tốt nhiệm vụ.

Tôi xin gửi lời cảm ơn đến Ban Giám Hiệu và toàn thể các cán bộ, nhân viên của Trường Đại Học Khoa Học Huế đã tạo điều kiện thuận lợi về cơ sở vật chất, môi trường học tập và các hoạt động hỗ trợ sinh viên. Sự quan tâm và hỗ trợ từ nhà trường là nguồn động lực lớn lao để tôi có thể hoàn thành khóa luận này.

Không thể thiếu lời cảm ơn đến các bạn học cùng lớp và những người bạn thân thiết đã luôn bên cạnh, chia sẻ kiến thức, kinh nghiệm và động viên tôi trong suốt thời gian qua. Những khoảnh khắc cùng nhau học tập, thảo luận và trao đổi đã giúp tôi học hỏi thêm nhiều điều và cảm thấy không đơn độc trong quá trình thực hiện khóa luận.

Tôi muốn gửi lời cảm ơn đến gia đình và người thân đã nuôi dưỡng, luôn bên cạnh, thấu hiểu và tin tưởng, giúp tôi yên tâm học tập và làm việc. Cuối cùng, tôi muốn gửi lời cảm ơn đến bản thân vì đã luôn cố gắng và không bỏ cuộc trong suốt khoảng thời gian qua. Dù có những lúc tôi gặp khó khăn và thất bại, nhưng bản thân luôn đứng lên và tiếp tục chiến đấu. Tôi biết rằng không có gì là dễ dàng, và việc vượt qua những trở ngại đó đã khiến tôi trưởng thành hơn và mạnh mẽ hơn. Bản thân tôi đã cố gắng hết sức trong quá trình thực hiện đề tài này nhưng chắc chắn sẽ không tránh khỏi những thiếu sót. Kính mong quý thầy cô và các bạn góp ý, chỉ bảo.

Xin chân thành cảm ơn!

# DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Từ viết tắt	Điễn giải	Dịch nghĩa
OCR	Optical Character Recognition	Nhận diện kí tự quang học
NLP	Natural Language Processing	Xử lý ngôn ngữ tự nhiên
DL	Deep learning	Học sâu
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy
LSTM	Long short-term memory	Mạng bộ nhớ dài-ngắn
CRNN	Convolutional Recurrent Neural Network	Mạng nơ-ron hồi quy tích chập
CER	Character error rate	Tỷ lệ lỗi ký tự
WER	Word error rate	Tỷ lệ lỗi từ
SER	Sequence error rate	Tỷ lệ lỗi chuỗi từ
CPU	Central Processing Unit	Đơn vị xử lý trung tâm
GPU	Graphics Processing Unit	Đơn vị xử lý đồ họa

# Danh sách hình vẽ

1.1	Mô tả việc phát hiện và nhận diện văn bản. Nguồn: Zhu et al	11
1.2	Ví dụ về đầu vào . . . . .	12
1.3	Ví dụ về đầu ra của bài toán . . . . .	13
1.4	Ví dụ về vấn đề thường gặp trong bài toán . . . . .	14
1.5	Stroke Width Transform phát hiện văn bản trong hình ảnh.Nguồn: Matas et al [9] . . . . .	15
1.6	Ví dụ về khả năng phát hiện chữ nghiêng của PSENet.Nguồn: Wang et al [20] . . . . .	16
1.7	Tổng quan về mô hình CRNN sử dụng Connectionist Temporal Classification. Nguồn: Shi et al [13] . . . . .	17
1.8	Trái: Tâm hình ảnh thử nghiệm ILSVRC-2010 và năm nhãn được mô hình AlexNet coi là có khả năng xảy ra nhất. Nhãn chính xác được ghi dưới mỗi hình ảnh và xác suất được gán cho nhãn chính xác cũng được hiển thị bằng thanh màu đỏ (nếu nó nằm trong top 5); Phải: Năm hình ảnh thử nghiệm ILSVRC-2010 ở cột đầu tiên. Các cột còn lại hiển thị sáu ảnh huấn luyện tạo ra các vectơ đặc trưng ở lớp ẩn cuối cùng có khoảng cách euclidean nhỏ nhất tới vectơ đặc trưng cho ảnh thử nghiệm (Hình ảnh tham khảo ở Alex 2012 [1]). . . . .	19
1.9	Ví dụ về phát hiện đối tượng bằng cách sử dụng đề xuất RPN trong bài kiểm tra PASCAL VOC 2007 (Hình ảnh tham khảo từ Shaoqing et al [12]). . . . .	19
1.10	Ví vụ về một mạng CNN điển hình. Nguồn: Shaoqing et al [12]	20
1.11	.Minh họa bộ lọc tích chập trong mô hình trích xuất bản đồ đặc trưng (feature map) từ ảnh đầu vào . . . . .	21
1.12	Ví dụ về lớp chập với nhiều bộ lọc để cho ra bản đồ đặc trưng hai chiều . . . . .	21

1.13	Ảnh minh họa giá trị đầu ra thay đổi khi thay đổi bước nhảy (stride). . . . .	22
1.14	Ảnh minh họa giá trị đầu ra thay đổi khi thay đổi bước nhảy (stride). . . . .	23
1.15	Ảnh tổng kết lại công thức tính đầu ra của lớp chập cùng với các tham số. . . . .	23
1.16	Ví dụ về lớp gộp (pooling layer). . . . .	24
1.17	Ví dụ về lớp gộp (pooling layer). . . . .	25
1.18	Hình minh họa đầu vào của lớp kết nối đầy đủ có 3072 đơn vị, lớp kết nối đầy đủ có 10 đơn vị thì sẽ có $10 \times 3072$ trọng số cần được học. . . . .	26
1.19	Cách thức hoạt động của mạng RNN . . . . .	27
1.20	Lớp kích hoạt trong RNN . . . . .	27
1.21	Kiến trúc của mạng RNN . . . . .	28
1.22	Mô hình RNN được sử dụng theo các loại bài toán . . . . .	28
1.23	Giải pháp cho vấn đề vanishing gradient . . . . .	29
1.24	Mô hình Bi-directional RNN . . . . .	30
1.25	Mô hình Long Short-term Memory (Nguồn: <a href="https://colah.github.io">https://colah.github.io</a> )	30
1.26	Cell state . . . . .	31
1.27	Cổng Forget. Nguồn: <a href="https://colah.github.io">https://colah.github.io</a> . . . . .	32
1.28	Cổng Tạo ra các giá trị ứng viên vào Cell state. Nguồn: <a href="https://colah.github.io">https://colah.github.io</a> . . . . .	33
1.29	Cập nhập Cell state. Nguồn: <a href="https://colah.github.io">https://colah.github.io</a> . . . . .	33
1.30	Đầu ra của Cell state. Nguồn: <a href="https://colah.github.io">https://colah.github.io</a> . . . . .	34
1.31	Hình ảnh mô tả mạng BiLSTM. . . . .	35
2.1	Mô tả quy trình giải quyết bài toán nhận diện văn bản Tiếng Việt . . . . .	37
2.2	Ví dụ về dataset Tiếng Việt . . . . .	37
2.3	Ví dụ về dataset Tiếng Việt được sinh ra . . . . .	38
2.4	Xử lý ảnh trước khi đưa vào mô hình . . . . .	39
2.5	Phân bố độ dài văn bản có trong hình ảnh . . . . .	39
2.6	Kiến trúc mạng CNN được sử dụng . . . . .	40
2.7	Biểu đồ nhiệt thể hiện khu vực ảnh hưởng đến đầu ra của mô hình . . . . .	40
2.8	Vị trí của Bidirectional LSTM trong mô hình . . . . .	41

2.9	Ví dụ về cách CTC hoạt động.Nguồn: <a href="https://distill.pub/2017/ctc/">https://distill.pub/2017/ctc/</a> . . . . .	42
2.10	Minh họa quá trình hoạt động của CTC . . . . .	43
2.11	Kết quả dự đoán cùng với nhãn thật . . . . .	45
2.12	Kết quả nhận dạng văn bản trong trường hợp dữ liệu thực tế .	46

# Mục lục

<b>Mở đầu</b>	<b>8</b>
<b>1 Tổng quan về bài toán nhận diện văn bản và kiến thức cơ sở</b>	<b>10</b>
1.1 Giới thiệu bài toán nhận diện văn bản . . . . .	10
1.1.1 Phát biểu bài toán . . . . .	12
1.1.2 Đầu vào đầu ra của bài toán . . . . .	12
1.1.3 Những vấn đề thường gặp . . . . .	13
1.1.4 Phương pháp phát hiện và nhận diện văn bản . . . .	14
1.1.5 Phương pháp phát hiện và nhận diện văn bản . . . .	16
1.2 Các kiến thức cơ sở . . . . .	18
1.2.1 Convolutional Neural Network . . . . .	18
1.2.2 Recurrent Neural Network . . . . .	27
1.2.3 Long Short-term Memory . . . . .	30
<b>2 Nhận diện văn bản sử dụng CRNN và CTC Loss</b>	<b>36</b>
2.1 Mô hình nhận diện văn bản với CRNN và CTC loss . . . . .	36
2.1.1 Mô hình chung . . . . .	36
2.1.2 Các bước thực hiện . . . . .	36
2.2 Thực nghiệm nhận diện văn bản . . . . .	37
2.2.1 Mô tả dữ liệu và tiền xử lý . . . . .	37
2.2.2 Xây dựng mô hình . . . . .	39
2.2.3 Hậu xử lý kết quả . . . . .	41
2.2.4 Kết quả thực nghiệm . . . . .	43
<b>KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN</b>	<b>47</b>
Kết luận . . . . .	47
Hướng phát triển . . . . .	48

# Mở đầu

Trong môi trường văn phòng thế kỷ 21, khối lượng tài liệu giấy vẫn rất lớn mặc dù chúng ta đã trải qua nhiều bước tiến lớn trong cuộc cách mạng kỹ thuật số. Việc cần phải chuyển đổi tài liệu từ giấy vật lý sang dữ liệu số là một vấn đề tất yếu cần phải được giải đáp. Đây không chỉ là vấn đề lưu trữ và quản lý dữ liệu, mà còn là việc tối ưu hóa quy trình làm việc, giảm thiểu rủi ro mất mát tài liệu, và tăng cường khả năng truy xuất thông tin một cách nhanh chóng và chính xác.

Để đáp ứng nhu cầu này, các công cụ Nhận diện kí tự quang học (Optical Character Recognition - OCR) đã được phát triển và ứng dụng rộng rãi. OCR là công nghệ nhận diện và chuyển đổi văn bản trong hình ảnh, tài liệu quét hoặc bất kỳ định dạng nào khác chứa văn bản, thành dữ liệu số có thể chỉnh sửa và tìm kiếm được. Với sự trợ giúp của OCR, việc chuyển đổi hàng loạt tài liệu giấy thành dữ liệu số trở nên đơn giản và hiệu quả hơn bao giờ hết. Các công cụ OCR hiện đại không chỉ có khả năng nhận diện văn bản với độ chính xác cao, mà còn có thể phân loại và tổ chức thông tin theo các tiêu chí khác nhau, giúp việc quản lý dữ liệu trở nên dễ dàng và hiệu quả hơn.

Bên cạnh việc nhận diện văn bản, các công nghệ OCR tiên tiến còn có khả năng xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) để hiểu ngữ cảnh và ý nghĩa của văn bản. Điều này mở ra nhiều cơ hội mới trong việc tự động hóa quy trình làm việc, từ việc tự động phân loại email, phân tích nội dung hợp đồng, đến việc trích xuất thông tin quan trọng từ các báo cáo tài chính. Hơn nữa, OCR còn đóng vai trò quan trọng trong việc hỗ trợ các ứng dụng trí tuệ nhân tạo khác, như chatbot và trợ lý ảo, bằng cách cung cấp dữ liệu đầu vào chính xác và đáng tin cậy.

Với những tiến bộ vượt bậc trong công nghệ OCR và Thị giác Máy tính,

chúng ta đang tiến gần hơn đến một tương lai nơi mà mọi thông tin đều có thể được số hóa, truy xuất và phân tích một cách tự động và thông minh. Điều này không chỉ mang lại lợi ích lớn cho doanh nghiệp và tổ chức, mà còn góp phần thúc đẩy sự phát triển của xã hội số, nơi mà dữ liệu và thông tin trở thành tài sản quý giá nhất.

# Chương 1

## Tổng quan về bài toán nhận diện văn bản và kiến thức cơ sở

### 1.1 Giới thiệu bài toán nhận diện văn bản

Nhận diện văn bản là việc chuyển đổi văn bản từ một hay nhiều ảnh trong tài liệu văn bản sang dữ liệu văn bản số được lưu trữ trong máy tính. Là quá trình xử lý và nhận dạng các ký tự, số và các ký tự đặc biệt trong các hình ảnh chứa văn bản. Nhận diện văn bản gồm hai quá trình chính đó là phát hiện văn bản(Text Detection) và nhận diện văn bản(Text Recognition) với việc phát hiện văn bản bao gồm xác định vị trí của các ký tự hoặc từ trong một hình ảnh.

Mục tiêu của phát hiện văn bản là xác định vùng chứa văn bản trong hình ảnh và tạo ra một khu vực bao quanh văn bản. Trong khi đó nhận diện văn bản sẽ lấy những vùng có văn bản đã phát hiện được sau đó sẽ nhận diện ký tự hoặc từ có trong đó



Hình 1.1: Mô tả việc phát hiện và nhận diện văn bản. Nguồn: Zhu et al [24]

Việc áp dụng nhận diện văn bản hiện nay đang trở nên rất phổ biến, đặc biệt là trong lĩnh vực kinh tế với việc số hóa các bản giao dịch hay dịch văn bản giữa các ngôn ngữ với nhau, sau đây là một vài ứng dụng chính của nhận diện văn bản

- Quản lý văn bản: OCR cho phép chuyển đổi các tài liệu giấy thành dạng số, giúp quản lý tài liệu dễ dàng hơn. Các tài liệu như hóa đơn, biên lai, giấy tờ tùy thân và các văn bản khác có thể được quét và chuyển đổi thành dữ liệu số để lưu trữ, tra cứu và chia sẻ một cách hiệu quả. [15]
- Nhận diện chữ viết tay: OCR không chỉ nhận diện văn bản in sẵn mà còn có thể nhận diện chữ viết tay. Điều này có thể hữu ích trong các ứng dụng như ghi chú, đọc ghi chú tay hoặc xử lý các biên bản ghi chép tay.
- Quét thẻ căn cước, thẻ nhận dạng: OCR được sử dụng để tự động nhận dạng thông tin trên thẻ căn cước, thẻ nhận dạng, thẻ sinh viên, giấy tờ nhận dạng, giúp tiết kiệm thời gian và giảm sai sót trong quá trình nhập dữ liệu. [6]
- Dịch thuật tự động: Các ứng dụng dịch thuật như Google Translate sử dụng OCR để nhận diện và dịch các đoạn văn bản từ hình ảnh hoặc tài liệu đã quét. [25]

Bài toán nhận diện văn bản nhằm mục đích chuyển đổi các tài liệu được lưu trữ dưới dạng hình ảnh thành các đối tượng có thể xử lý được trong máy tính như văn bản dạng thuần túy hoặc văn bản mã hóa. Để thực hiện việc này, các hình ảnh hay file PDF sẽ được xử lý bằng các thuật toán OCR để chuyển đổi thành các đối tượng văn bản. Sau đó, các đối tượng này sẽ được mã hóa và

phân loại để trở thành các đối tượng văn bản có thể xử lý được.

### 1.1.1 Phát biểu bài toán

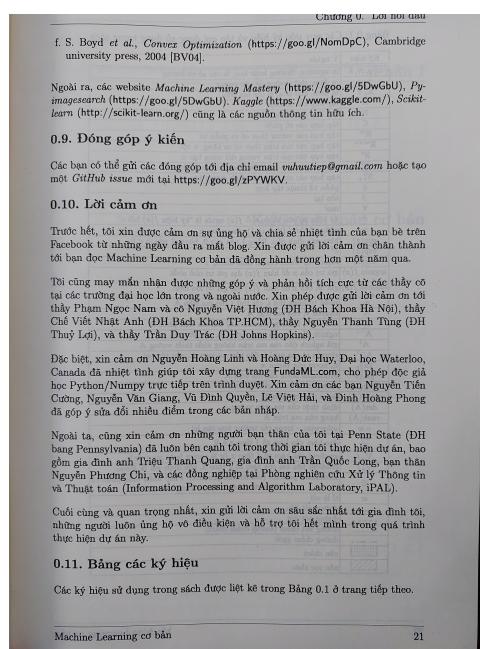
Quy trình giải quyết bài toán được đề ra được minh họa ở hình 2.1, với những tiêu chuẩn cần phải được đáp ứng:

- Phát hiện được hết toàn bộ khu vực có văn bản trong ảnh và trích xuất ra các vùng ảnh chứa văn bản
- Từ CRNN huấn luyện được, dự đoán chính xác văn bản có trong vùng ảnh đã lấy ra theo từng chuỗi
- Lưu trữ văn bản đã dự đoán được dưới dạng văn bản số trong tệp tin văn bản như Notepad, Microsoft Word,...

### 1.1.2 Đầu vào đầu ra của bài toán

#### 1.1.2.1 Đầu vào của bài toán

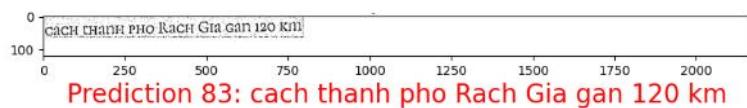
Đầu vào của bài toán là một hình ảnh có chứa văn bản dưới dạng giấy in, được thu thập bằng các thiết bị như máy ảnh, camera của điện thoại thông minh, máy scan,... và đáp ứng những điều kiện như không bị quá nhiễu, mờ, ánh sáng trong ảnh ở mức phù hợp cho việc nhận dạng văn bản



Hình 1.2: Ví dụ về đầu vào

### 1.1.2.2 Đầu ra của bài toán

Sau khi có ảnh đầu vào của bài toán, đầu ra sẽ là văn bản thuần túy, có thể được lưu vào định dạng văn bản như Microsoft Word, NotePad, LibreOffice,...

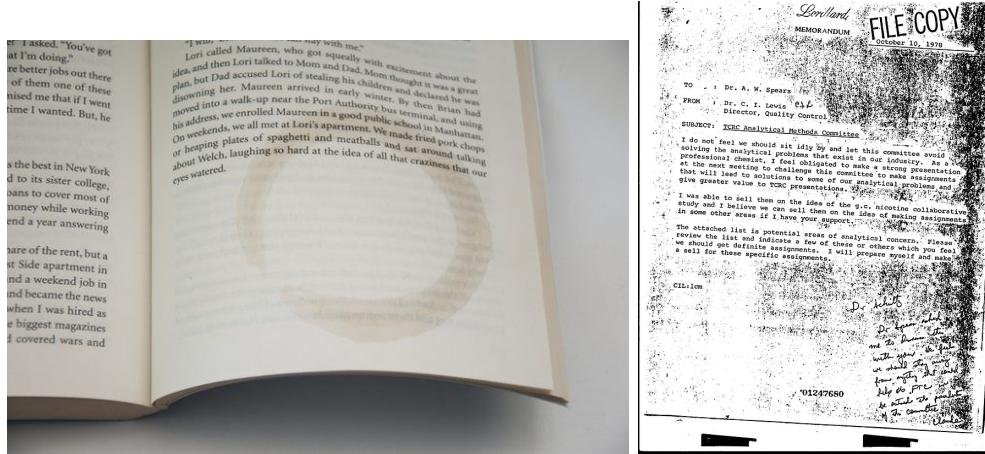


Hình 1.3: Ví dụ về đầu ra của bài toán

### 1.1.3 Những vấn đề thường gặp

Bài toán nhận diện văn bản đặt ra nhiều yêu cầu thách thức với các nhà nghiên cứu về mặt xử lý ảnh chịu ảnh hưởng bởi các yếu tố như ảnh nhiễu, ảnh mờ, sự xuất hiện của chữ viết tay hoặc chữ in ở góc nghiên khác với thông thường,... làm ảnh hưởng đến độ chính xác của mô hình nhận dạng văn bản. Sau đây là một vài vấn đề thường gặp của bài toán nhận diện văn bản

- Độ phức tạp của kí tự có trong ảnh: Văn bản in có trong ảnh có thể chứa nhiều font chữ với nhiều kích thước và màu sắc khác nhau, chữ viết tay có thể thuộc về nhiều người với cách viết khác nhau
- Chất liệu giấy: Giấy có thể có nhiều màu sắc, bị ô màu hoặc có vết bẩn. Giấy bóng phản chiếu lại ánh sáng nhiều hơn so với giấy thông thường hay giấy nhám làm ảnh hưởng đến việc phát hiện văn bản
- Lỗi từ việc thu thập dữ liệu: Ảnh chụp văn bản có thể chịu ảnh hưởng của các yếu tố như nhiễu, mờ, độ phân giải thấp, chịu ảnh hưởng bởi ánh sáng từ môi trường bên ngoài hay do thiết bị chụp ảnh. Văn bản trong ảnh cũng có thể bị nghiêng do người chụp ảnh
- Thời gian xử lý: Việc xử lý các tài liệu văn bản có thể đòi hỏi nhiều thời gian và tài nguyên tính toán. Điều này đặc biệt quan trọng đối với các công việc yêu cầu xử lý khối dữ liệu lớn cần độ chính xác cao



(a) Ảnh bị ô bởi vết cà phê

(b) Ảnh bị nhiễu và nghiêng

Hình 1.4: Ví dụ về vấn đề thường gặp trong bài toán

## 1.1.4 Phương pháp phát hiện và nhận diện văn bản

### 1.1.4.1 Phương pháp phát hiện

Phương pháp phát hiện văn bản truyền thống thường chú trọng vào việc sử dụng các đặc trưng được tạo một cách thủ công để phân biệt phần có văn bản và phần không có văn bản trong hình ảnh, được chia thành hai phương thức chính là cửa sổ trượt(Sliding Window-SW) và thành phần liên kết(Connected Component-CC)

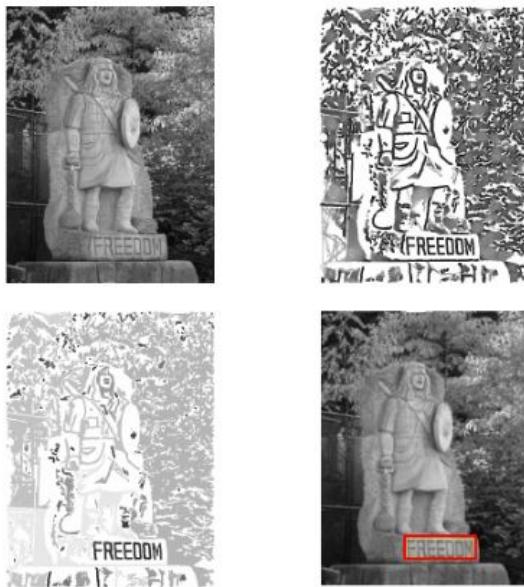
### 1.1.4.2 Cửa sổ trượt(Sliding Window-SW)

Cửa sổ trượt là phương pháp sử dụng một khung hình có kích thước cố định và di chuyển nó để quét qua tất cả các vùng của hình ảnh, từ đó nhận dạng các ký tự.

Các ký tự được nhận dạng từ khung cửa sổ qua việc trích xuất thông tin và sử dụng một classifier đã được huấn luyện từ trước để xác định văn bản có trong khung cửa sổ trượt hay không. Ngoài việc sử dụng classifier có từ trước, nhiều nghiên cứu như của Wang et al [19], Jaderberg et al [7] đã áp dụng thêm mô hình CNN để phát hiện ra các vùng ứng viên tiềm năng có thể chứa văn bản

#### 1.1.4.3 Thành phần liên kết (Connected Component-CC)

Thành phần liên kết là phương pháp trích xuất những thành phần (thông thường là các pixel) liên kết với nhau và sau đó lọc ra những phần không có chứa văn bản. So sánh với phương pháp dùng cửa sổ trượt phương pháp này cho ra được kết quả hiệu quả hơn. Hai mô hình tiêu biểu của phương pháp này là Stroke Width Transform [3] và Maximally Stable Extremal Regions [9]



Hình 1.5: Stroke Width Transform phát hiện văn bản trong hình ảnh. Nguồn: Matas et al [9]

### Các phương pháp deep learning

Trong những năm gần đây, lĩnh vực nhận diện kí tự quang học (Optical Character Recognition - OCR) có nhiều bước nhảy vọt với sự ra đời của nhiều mô hình deeplearning như EAST [23], SAST [18], PSENet [20] và phiên bản nâng cấp của nó Pixel Aggregation Network (PAN) [21],... đã đưa ra nhiều phương pháp giúp cải thiện độ chính xác và tốc độ xử lý hình ảnh với độ chính xác cao. Những mô hình này có thể phát hiện được văn bản ở nhiều góc cảnh khác nhau, điều mà nhiều mô hình trước đó còn gặp nhiều khó khăn để đạt được.



Hình 1.6: Ví dụ về khả năng phát hiện chữ nghiêng của PSENet. Nguồn: Wang et al [20]

### 1.1.5 Phương pháp phát hiện và nhận diện văn bản

Cũng giống như phương pháp phát hiện văn bản, phương pháp nhận diện văn bản cũng có bước nhay vọt nhờ việc áp dụng deep learning thay vì sử dụng các đặc trưng thủ công theo kiểu cũ. Hiện nay phương pháp nhận diện văn bản có thể được chia thành 3 hướng chính: phân loại theo ký tự(character classification base), phân loại theo từ(word classification base), theo chuỗi(sequence base),

#### 1.1.5.1 Phân loại theo ký tự

Phương pháp phân loại theo ký tự sẽ tìm từng ký tự riêng biệt trong ảnh và lần lượt nhận diện chúng theo một cách riêng lẻ. Những quy luật heuristic phức tạp hoặc mô hình ngôn ngữ là những thứ không thể thiếu cho phương pháp này vì khả năng xâu chuỗi các ký từ thành từ của chúng, do khả năng ký tự khi nhận dạng bị mất hoặc bị cắt mất một phần

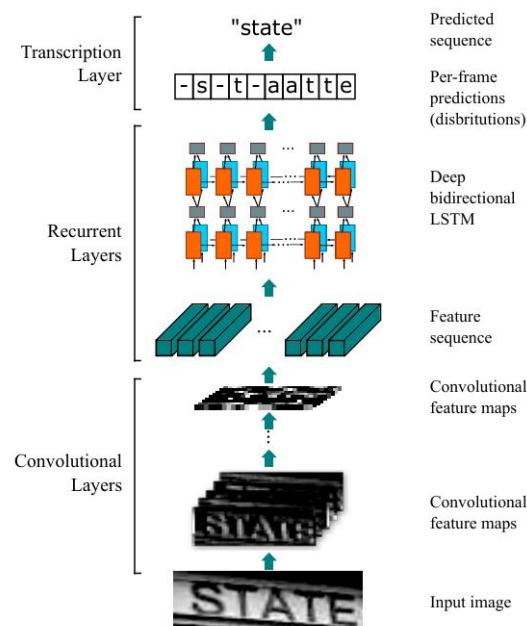
Bissacco et al [2] đã sử dụng mô hình deep learning được huấn luyện trên các đặc trưng HOG để phân loại văn bản, công trình có kết hợp sử dụng hai lớp mô hình ngôn ngữ N-Gram để tăng hiệu năng nhận diện văn bản. Lee et al [8] đề ra việc sử dụng mạng hồi quy (Recurrent Neural Network-RNN) kết hợp với mô hình attention làm loại bỏ đi nhu cầu sử dụng N-Gram, giúp cho mô hình có thể chọn lọc đặc trưng một cách linh hoạt

### 1.1.5.2 Phân loại theo từ

Phương pháp phân loại theo từ thực chất là một công việc phân loại đa lớp với số lượng nhãn lớp lớn(Tiếng Việt có hơn 36000 từ, Tiếng Anh có đến 90000 từ). Nhờ việc áp dụng CNN với khả năng tính toán của nó mà công việc này mới phần nào có thể giải quyết được. Tuy nhiên, đối với các ảnh mà có từ dài thì độ nhận diện từ có thể bị ảnh hưởng lớn. Hơn nữa, do phương pháp này dựa phần nhiều vào từ điển đã được xác định từ trước nên những từ nằm ngoài phạm vi đã xác định có thể không nhận diện được.

### 1.1.5.3 Phân loại theo chuỗi

Trong công trình được đề xuất bởi Shi et al [13], nhóm nghiên cứu đã đề ra một mô hình hồi quy nhân chập cho việc nhận dạng chuỗi từ dựa trên ảnh. Đầu tiên một mô hình CNN tiêu chuẩn sẽ được dùng để trích xuất các chuỗi đặc trưng ra từ ảnh đầu vào. Sau đó đặc trưng sẽ được đưa vào một mạng lưới Long-short term memory hai chiều(Bidirectional LSTM) kết nối với mạng nhân chập để đưa ra được nhãn cho từng khung hình của đặc trưng. Phân loại thời gian của mạng Kết nối(Connectionist Temporal Classification-CTC) được áp dụng để tìm ra được chuỗi nhãn có khả năng xuất hiện cao nhất trên dự đoán của mỗi khung hình.



Hình 1.7: Tổng quan về mô hình CRNN sử dụng Connectionist Temporal Classification. Nguồn: Shi et al [13]

Mô hình được sử dụng trong bài báo cáo này sẽ dựa trên công trình của Shi et al [13]

## 1.2 Các kiến thức cơ sở

### 1.2.1 Convolutional Neural Network

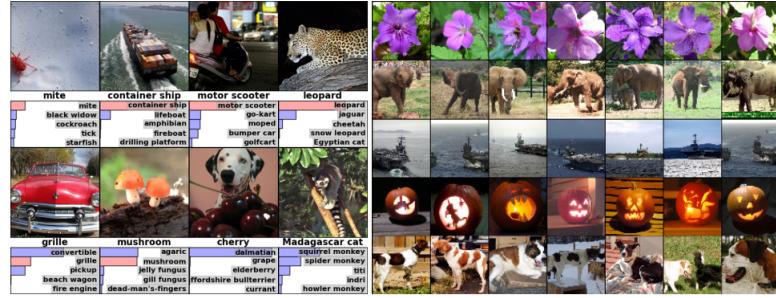
#### 1.2.1.1 Giới thiệu chung

Convolutional Neural Network (CNN hoặc ConvNets) là một loại kiến trúc mạng nơ-ron được thiết kế đặc biệt để xử lý và phân loại dữ liệu có cấu trúc lưới, như hình ảnh và video.

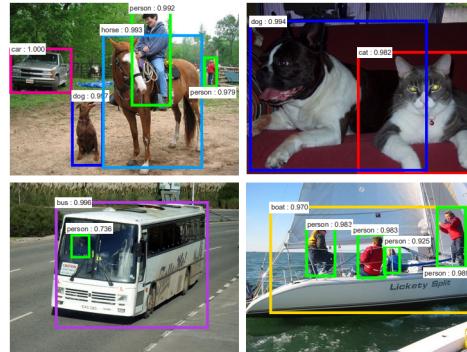
Năm 1998, Yann LeCun đã đưa ra ví dụ đầu tiên về việc áp dụng phương pháp lan truyền ngược (backpropagation) và học tập dựa trên độ dốc (gradient-based learning) để huấn luyện các mạng thần kinh tích chập hoạt động rất tốt trong việc nhận dạng tài liệu [22]. Đặc biệt là họ đã làm tốt công việc nhận dạng các chữ số của mã zip. Do đó, CNN được sử dụng khá rộng rãi để nhận dạng mã zip trong dịch vụ bưu chính. Nhưng, nó vẫn chưa thể mở rộng quy mô sang những dữ liệu phức tạp và khó khăn hơn.

Những kết quả thành công đầu tiên bằng cách sử dụng mạng lưới thần kinh, điều thực sự khơi dậy cơn sốt sử dụng những loại mạng CNN này một cách thực sự rộng rãi là vào khoảng năm 2012, là bài báo mang tính bước ngoặc của Alex Krizhevsky trong phòng thí nghiệm của Geoff Hinton, giới thiệu kiến trúc mạng thần kinh tích chập đầu tiên có thể thực hiện được và thu được kết quả thực sự mạnh mẽ về phân loại ImageNet [1], mô hình này được gọi là AlexNet. Ngoài mô hình LeNet [22] và mô hình AlexNet [1], còn có các mô hình CNN khác như ZFNetzeiler2013, GoogleNet [16], VGGNet [14] và ResNet [5]. Những mô hình này phổ biến vì chúng cung cấp hiệu suất cao và đáng kể trên các nền tảng khác nhau tiêu chuẩn phân loại hình ảnh.

Ngày nay, mạng CNN được sử dụng ở khắp nơi, ví dụ như phân loại và truy xuất ảnh như hình 1.8, trích xuất và nhận dạng đối tượng như hình 1.9, ...



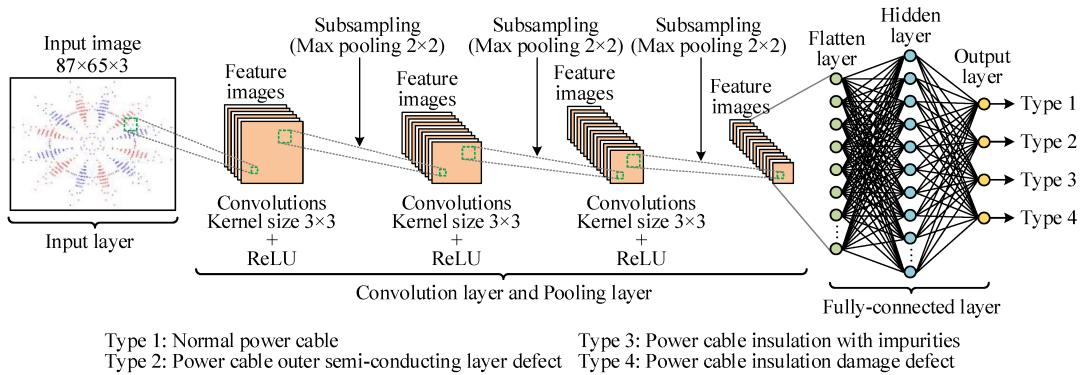
Hình 1.8: Trái: Tám hình ảnh thử nghiệm ILSVRC-2010 và năm nhãn được mô hình AlexNet coi là có khả năng xảy ra nhất. Nhãn chính xác được ghi dưới mỗi hình ảnh và xác suất được gán cho nhãn chính xác cũng được hiển thị bằng thanh màu đỏ (nếu nó nằm trong top 5); Phải: Năm hình ảnh thử nghiệm ILSVRC-2010 ở cột đầu tiên. Các cột còn lại hiển thị sáu ảnh huấn luyện tạo ra các vectơ đặc trưng ở lớp ẩn cuối cùng có khoảng cách euclidean nhỏ nhất tới vectơ đặc trưng cho ảnh thử nghiệm (Hình ảnh tham khảo ở Alex 2012 [1]).



Hình 1.9: Ví dụ về phát hiện đối tượng bằng cách sử dụng đề xuất RPN trong bài kiểm tra PASCAL VOC 2007 (Hình ảnh tham khảo từ Shaoqing et al [12]).

### 1.2.1.2 Kiến trúc mô hình

CNN sở hữu khả năng học các tính năng tự động từ dữ liệu đầu vào và nó loại bỏ việc trích xuất tính năng thủ công. CNN yêu cầu bộ dữ liệu được dán nhãn khổng lồ để đào tạo vì nó là phương pháp học có giám sát và về cơ bản nó được lấy cảm hứng từ vỏ não thị giác của động vật.



Hình 1.10: Ví dụ về một mạng CNN điển hình. Nguồn: Shaoqing et al [12]

Dựa vào hình 1.10, có thể thấy được cơ bản kiến trúc của CNN, lớp chập (convolutional layer) bao gồm tập hợp các bộ lọc (filter) đã được áp dụng trên hình ảnh đầu vào và sau khi dữ liệu được tạo được truyền qua đến lớp gộp (polling layer). Bản đồ đặc trưng (feature map) được tạo ở mỗi lớp chập, nó có được thông qua tính toán tích chập giữa các mảng cục bộ và vectơ trọng số thường được gọi là bộ lọc. Bản đồ đặc trưng là nhóm các tổng trọng số cục bộ. Để nâng cao hiệu quả đào tạo, các bộ lọc được áp dụng nhiều lần. Quá trình này giúp giảm số lượng tham số trong quá trình học.

Việc lấy mẫu con (subsampling) tối đa hoặc trung bình của các vùng không chồng chéo trong bản đồ đặc trưng được thực hiện tại mỗi lớp gộp (polling layer). Cuối cùng, các lớp kết nối đầy đủ (fully connected layer) đang hoạt động như một mạng lưới thần kinh thông thường. Giai đoạn cuối cùng, CNN vẫn có hàm mất mát (ví dụ: SVM/Softmax) trên lớp kết nối đầy đủ.

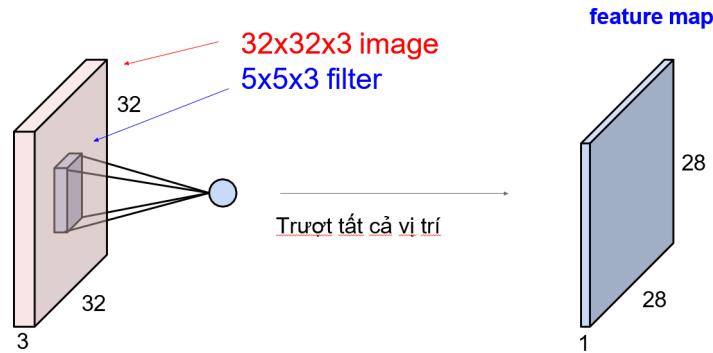
Như vậy, kiến trúc CNN thông thường được xây dựng bởi các lớp bao gồm: lớp chập (convolutional layer), lớp gộp (pooling layer), lớp kích hoạt (activation layer), lớp kết nối đầy đủ (fully connected layer).

### 1.2.1.3 Lớp Chập (Convolutional Layer)

Lớp chập là cốt lõi của Mạng tích chập, thực hiện hầu hết các công việc tính toán nặng nhọc trong mô hình.

Các tham số của lớp chập bao gồm một tập hợp các bộ lọc có thể học được. Mọi bộ lọc đều có kích thước nhỏ về mặt không gian (dọc theo chiều rộng và chiều cao), nhưng kéo dài đến toàn bộ chiều sâu của đầu vào. Ví dụ: bộ lọc thông thường trên lớp đầu tiên của CNN có thể có kích thước [5x5x3]

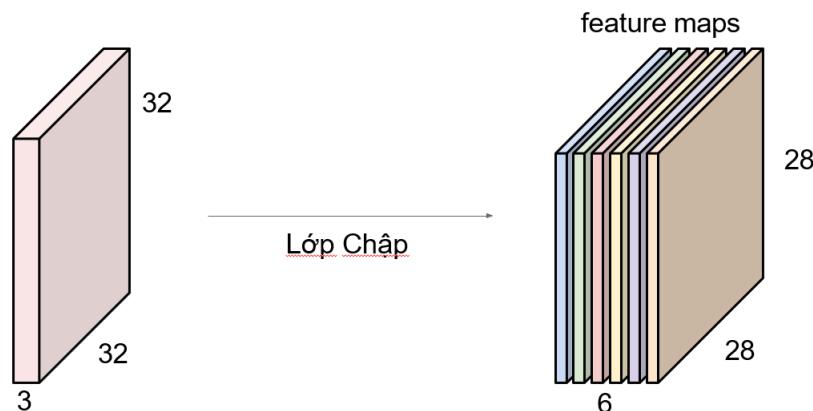
(tức là chiều rộng và chiều cao 5 pixel và 3 vì hình ảnh có độ sâu 3, tức là các kênh màu).



Hình 1.11: Minh họa bộ lọc tích chập trong mô hình trích xuất bản đồ đặc trưng (feature map) từ ảnh đầu vào

Trong quá trình chuyển tiếp, CNN thực hiện tích chập từng bộ lọc theo chiều rộng và chiều cao của đầu vào và tính toán tích số chấm (dot product) giữa các mục của bộ lọc và đầu vào ở bất kỳ vị trí nào (Như hình 1.11). Khi trượt bộ lọc theo chiều rộng và chiều cao của đầu vào, CNN sẽ tạo ra bản đồ đặc trưng (feature map) 2 chiều cung cấp phản hồi của bộ lọc đó ở mọi vị trí.

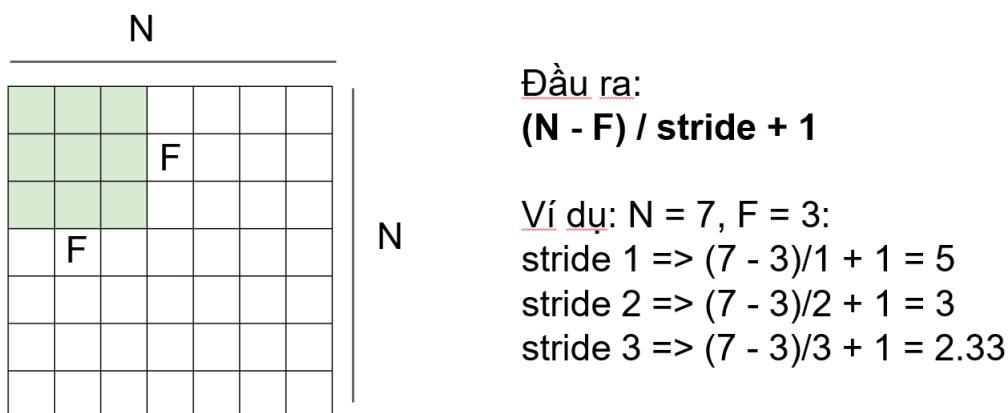
Theo trực quan, CNN sẽ tìm hiểu các bộ lọc kích hoạt khi chúng nhìn thấy một số loại tính năng trực quan, chẳng hạn như cạnh của một số hướng hoặc một đốm màu nào đó trên lớp đầu tiên. Bây giờ, ta sẽ có toàn bộ bộ lọc trong mỗi lớp CONV (ví dụ: 6 bộ lọc) và mỗi bộ lọc sẽ tạo ra một bản đồ đặc trưng 2 chiều riêng biệt. Ta sẽ xếp chồng các bản đồ đặc trưng này dọc theo chiều sâu và tạo ra khối lượng đầu ra (Như hình 1.12).



Hình 1.12: Ví dụ về lớp chập với nhiều bộ lọc để cho ra bản đồ đặc trưng hai chiều

Bây giờ, do vẫn chưa rõ về việc có bao nhiêu nơ-ron trong khối lượng đầu ra hoặc cách chúng được sắp xếp, ta có ba siêu tham số kiểm soát kích thước của âm lượng đầu ra: độ sâu (depth), bước nhảy (stride) và khoảng đệm 0 (zero-padding).

- Đầu tiên, độ sâu của đầu ra là một siêu tham số: nó tương ứng với số lượng bộ lọc mà ta muốn sử dụng, mỗi bộ lọc sẽ tìm kiếm thứ gì đó khác nhau trong đầu vào. Ví dụ: nếu Lớp chập đầu tiên lấy hình ảnh làm đầu vào, thì các nơ-ron khác nhau đọc theo chiều sâu có thể kích hoạt khi có các cạnh định hướng khác nhau hoặc các đốm màu. Điều này cũng có thể giải thích theo hình 1.12, ta có thể thay đổi số lượng bộ lọc để thay đổi giá trị đầu ra.
- Thứ hai, ta phải chỉ định bước nhảy (stride) mà ta trượt bộ lọc. Khi bước nhảy là 1 thì bộ lọc di chuyển qua từng pixel một. Khi bước nhảy là 2 (hoặc hiếm gặp là 3 hoặc nhiều hơn) thì các bộ lọc sẽ nhảy 2 pixel mỗi trượt chúng. Điều này sẽ tạo ra khối lượng đầu ra nhỏ hơn về mặt không gian. Hình 1.13 minh họa giá trị đầu ra khi thay đổi bước nhảy (stride), trong đó  $N = 7$ ,  $F = 3$  và với bước nhảy là 1, ta nhận được đầu ra là  $5 \times 5$ ; bước nhảy là 2, ta nhận được đầu ra là  $3 \times 3$ ; bước nhảy là 3, có lỗi không khớp xảy ra.



Hình 1.13: Ảnh minh họa giá trị đầu ra thay đổi khi thay đổi bước nhảy (stride).

- Đôi khi sẽ rất thuận tiện khi đệm đầu vào bằng các số 0 xung quanh đường viền. Kích thước của phần đệm số 0 (zero-padding) này là một siêu tham số. Tính năng hay của phần đệm bằng 0 là nó sẽ cho phép ta

kiểm soát kích thước khung gian của âm lượng đầu ra. Theo ví dụ hình 1.14, ta có một đầu vào  $[7 \times 7]$ , với bước nhảy là 1; bộ lọc  $[3 \times 3]$ ; đệm 1 lớp 0 xung quanh; ta được đầu ra sẽ là  $[7 \times 7]$ . Thông thường, giá trị kích thước của phần đệm số 0 sẽ được tính dựa vào F (kích thước bộ lọc), dựa vào công thức  $(F - 1)/2$ . Ví dụ, với  $F = 3$  thì sẽ đệm  $(F - 1) = (3 - 1)/2 = 1$  lớp 0, với  $F = 3$  thì sẽ đệm  $(F - 1) = (5 - 1)/2 = 2$  lớp 0.

0	0	0	0	0	0			
0								
0								
0								
0								

Hình 1.14: Ảnh minh họa giá trị đầu ra thay đổi khi thay đổi bước nhảy (stride).

Như có thể thấy trong CNN, việc xác định tham số cho mô hình CNN một cách thích hợp sao cho tất cả các kích thước đều "hoàn thiện" có thể thực sự là một vấn đề khó giải.

Tổng kết lại, đầu ra sau khi thông qua lớp chập có thể tính theo công thức hình 1.15:

**Bước 1:** Nhận đầu vào  $W_1 \times H_1 \times D_1$

**Bước 2:** Nhập 4 tham số:

- Số bộ lọc K,
- Phạm vi bộ lọc F,
- Bước nhảy S,
- Kích thước của bộ đệm 0.

**Bước 3:** Tính đầu ra  $W_2 \times H_2 \times D_2$ , với:

- $W_2 = (W_1 - F + 2P)/S + 1$ ,
- $H_2 = (H_1 - F + 2P)/S + 1$ ,
- $D_2 = K$

Hình 1.15: Ảnh tổng kết lại công thức tính đầu ra của lớp chập cùng với các tham số.

Ví dụ, có một ảnh đầu vào  $[32 \times 32 \times 3]$ , 10 bộ lọc  $[5 \times 5]$ , bước tiến bằng 1, kích cỡ đệm 0 bằng 2, như vậy, ta có:  $W1 = 32; H1 = 32; D1 = 3; K = 10; F = 5; S = 1; P = 2$ . Áp dụng công thức hình 1.15:

$$W2 = (W1 - F + 2P)/S + 1 = (32 - 5 + 2 * 2)/1 + 1 = 32$$

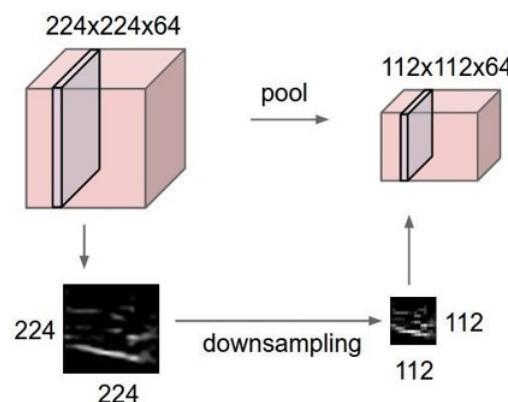
$$H2 = (H1 - F + 2P)/S + 1 = (32 - 5 + 2 * 2)/1 + 1 = 32$$

$$D2 = K = 10$$

Như vậy, đầu ra sẽ là  $[32 \times 32 \times 10]$ .

Bên cạnh đó, một số bài báo sử dụng tích chập  $1 \times 1$ , như Network in Network [10] đã nghiên cứu lần đầu tiên. Một số người lúc đầu bối rối khi thấy các tích chập  $[1 \times 1]$ , thông thường các tín hiệu là 2 chiều nên các tích chập  $[1 \times 1]$  không có ý nghĩa (nó chỉ là tỷ lệ theo điểm). Tuy nhiên, trong CNN, điều này không xảy ra vì cần phải nhớ rằng CNN hoạt động trên các khối 3 chiều và các bộ lọc luôn mở rộng đến toàn bộ độ sâu của khối đầu vào. Ví dụ: nếu đầu vào là  $[32 \times 32 \times 3]$  thì việc thực hiện tích chập  $[1 \times 1]$  sẽ thực hiện các tích điểm 3 chiều một cách hiệu quả (vì độ sâu đầu vào là 3 kênh).

#### 1.2.1.4 Lớp Gộp (Pooling Layer)



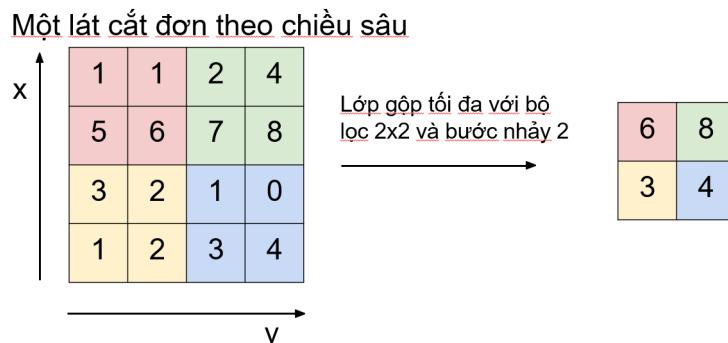
Hình 1.16: Ví dụ về lớp gộp (pooling layer).

Người ta thường luôn chèn một lớp gộp vào giữa các lớp chập liên tiếp trong CNN. Chức năng của nó là giảm dần kích thước không gian của biểu diễn để giảm số lượng tham số và tính toán trong mạng và do đó cũng kiểm

soát việc overfitting.

Dựa vào hình 1.16, lớp gộp nhận thông tin đầu vào của ảnh [224x224x64] và tiến hành giảm kích thước xuống, cuối cùng nhận được [112x112x64] (độ sâu không thay đổi vì chỉ gộp lại theo chiều rộng (width) và chiều cao (height)).

Phổ biến nhất là gộp tối đa (Max Pooling), theo ví dụ như hình 1.17. Các thí nghiệm thường làm cho lớp gộp không có bất kỳ sự trùng lặp nào vì về cơ bản là chỉ muốn lấy mẫu xuống, từ đó sẽ hợp lý hơn khi nhìn vào khu vực (region) này và chỉ lấy một giá trị để đại diện cho vùng này và sau đó chỉ cần nhìn vào vùng tiếp theo. Ngoài sử dụng gộp tối đa (max pooling), một số khác cũng nảy lên ý tưởng gộp trung bình (average pooling), tuy nhiên, cách này không phổ biến và không hiệu quả. Trong bài toán trích xuất đặc trưng ảnh hay nhận diện đối tượng, dù là ánh sáng hay khía cạnh nào đó của hình ảnh mà bạn đang tìm kiếm, các thí nghiệm luôn muốn nhắm đến với giá trị cao nhằm tìm thấy giá trị đại diện cho vùng đó, từ đó, giữ lại đặc trưng quan trọng cho vùng đó.



Hình 1.17: Ví dụ về lớp gộp (pooling layer).

### 1.2.1.5 Lớp Kích Hoạt (Activation Layer)

Lớp kích hoạt trong CNN thường được sử dụng để thêm tính phi tuyến tính vào mô hình. Mục tiêu của lớp này là giúp mô hình học được các biểu diễn phức tạp hơn từ dữ liệu đầu vào.

Lớp kích hoạt được áp dụng cho mỗi giá trị đầu ra của một neuron trong lớp trước đó và thường được thiết kế để giữ lại các giá trị dương và loại bỏ các giá trị âm (hoặc ngược lại). Hàm kích thích phổ biến nhất là Rectified Linear Unit (ReLU).

Công thức của hàm ReLU được mô tả như sau:

$$f(x) = \max(0, x)$$

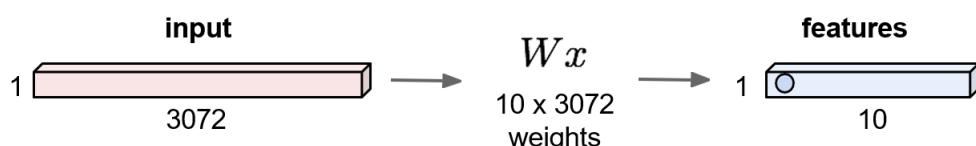
Trong đó, x là giá trị đầu vào; hàm max là hàm lấy giá trị lớn nhất giữa 0 và x.

ReLU giúp mô hình học được các đặc trưng phi tuyến tính, làm cho nó linh hoạt hơn trong việc biểu diễn các mối quan hệ phức tạp trong dữ liệu. Ngoài ra, ReLU còn giúp giảm ván đề mất mát đạo hàm bằng cách tránh giá trị đạo hàm tiến gần 0 ở các giá trị dương. Vì ReLU làm cho một số đơn vị trong mạng không hoạt động (output là 0), nó có thể làm tăng tốc quá trình huấn luyện bằng cách giảm số lượng tham số cần được cập nhật.

### 1.2.1.6 Lớp Kết Nối Đầy Đủ (Fully Connected Layer)

Lớp kết nối đầy đủ (Fully-Connected Layer hay còn gọi là Dense Layer) đóng vai trò quan trọng trong việc kết hợp thông tin từ các đặc trưng đã được trích xuất từ các lớp trước đó để tạo ra dự đoán hoặc phân loại cuối cùng.

Trong lớp kết nối đầy đủ, mỗi nơ-ron kết nối với mọi nơ-ron trong lớp trước đó. Đầu vào của lớp này là một vector một chiều của các giá trị từ các đơn vị trước đó. Nếu lớp trước có n đơn vị, và lớp kết nối đầy đủ có m đơn vị, thì sẽ có  $n \times m$  trọng số (weights) cần được học (như hình 1.18).



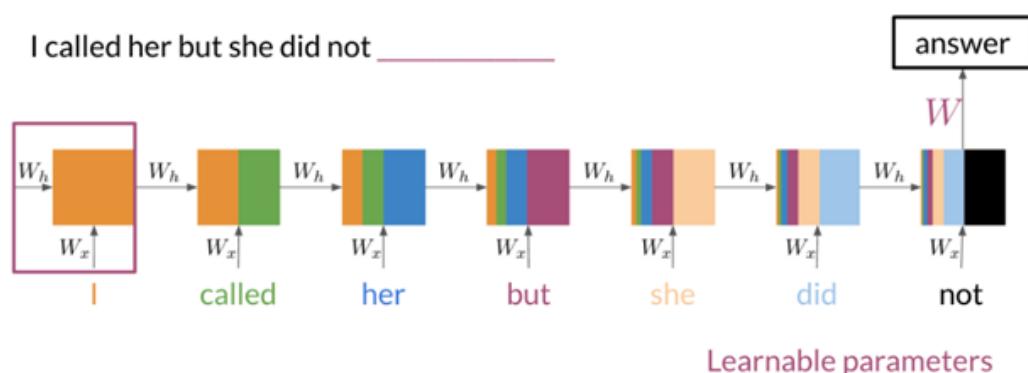
Hình 1.18: Hình minh họa đầu vào của lớp kết nối đầy đủ có 3072 đơn vị, lớp kết nối đầy đủ có 10 đơn vị thì sẽ có  $10 \times 3072$  trọng số cần được học.

## 1.2.2 Recurrent Neural Network

### 1.2.2.1 Giới thiệu mô hình

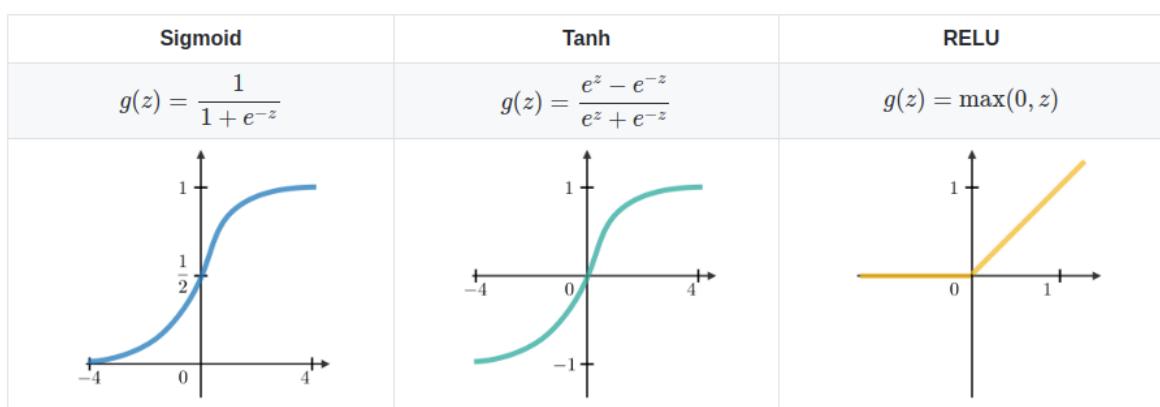
Recurrent Neural Network (RNN) được thiết kế để xử lý thông tin dạng chuỗi, với thứ tự thành phần của thông tin đầu vào được nhấn mạnh. Trong việc xử lý ngôn ngữ tự nhiên, RNN có thể nắm bắt ngữ nghĩa, ngữ cảnh và mối quan hệ giữa cá từ một cách hiệu quả nhờ việc có những liên kết được hình thành và lưu giữ thông tin từ đầu vào ban đầu đến các thành phần thông tin sau này (Hình 1.19).

Trong RNN, có một tham số học sẽ lưu lại thông tin về các tính toán về chuỗi thông tin được gọi là trạng thái ẩn (hidden state).



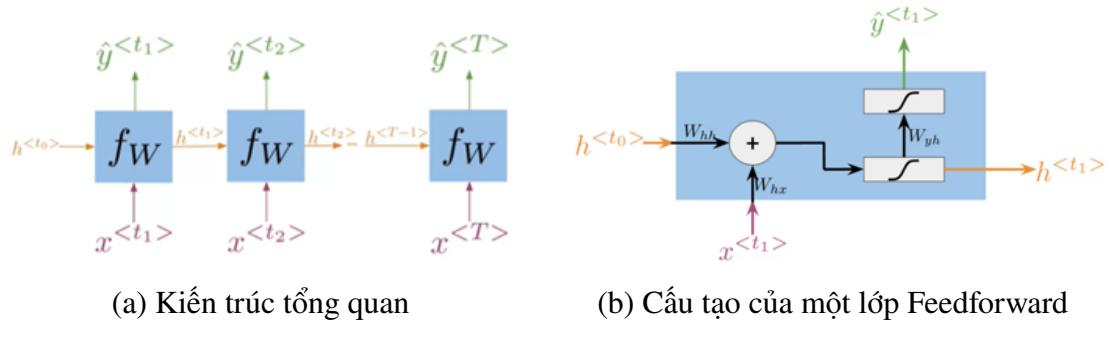
Hình 1.19: Cách thức hoạt động của mạng RNN

Hidden state và những trọng số đầu ra ở các bước của RNN được quyết định bởi lớp kích hoạt (Hình 1.20) để đưa dữ liệu vào chuỗi sau.



Hình 1.20: Lớp kích hoạt trong RNN

Kiến trúc mạng RNN cho phép đầu ra từ những chuỗi thông tin trước trở thành đầu vào của những chuỗi thông tin sau, và có cấu trúc như hình 1.21:

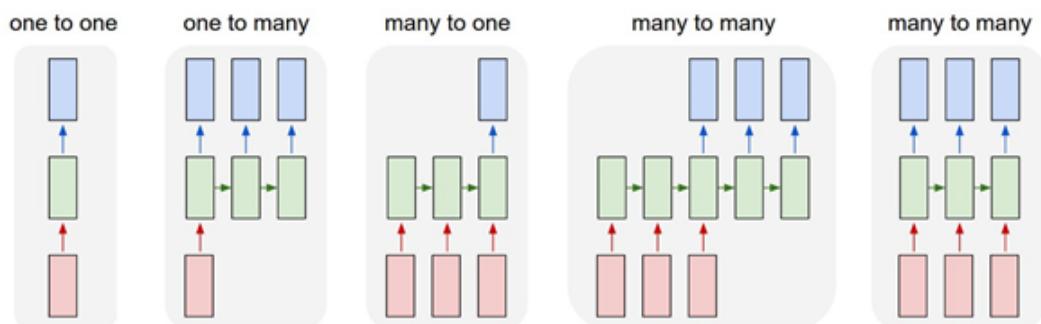


Hình 1.21: Kiến trúc của mạng RNN

Trong đó:

- $h^{<t>} = g(W_h[h^{<t-1>}, x^{<t>}] + b_h$  : Hidden state của mô hình RNN tại mỗi bước
- $y^{<t>} = g(W_{hh}h^{<t-1>} + W_{hx}x^{<t>} + b_h)$  : Đầu ra của mô hình RNN với  $x^{<t>}$  là đầu vào

Bài toán sử dụng RNN sẽ được phân loại dựa trên cách  $y^{<t>}$  được đưa ra khỏi mô hình từ đầu vào là  $x^{<t>}$ , bao gồm các loại bài toán như hình 1.22.



Hình 1.22: Mô hình RNN được sử dụng theo các loại bài toán

### **Ưu điểm của RNN:**

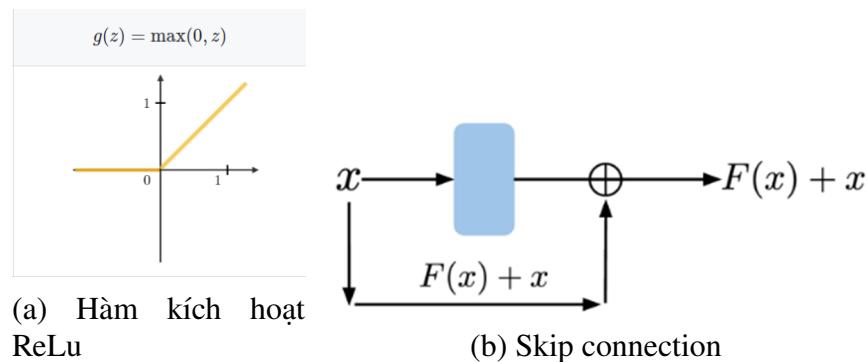
- Giúp nắm bắt được các phụ thuộc từ về mặt ngữ pháp và ngữ nghĩa trong một khoảng ngắn.
- Tốn ít tài nguyên bộ nhớ hơn mô hình N-gram.

### **Nhược điểm của RNN:**

- Gặp khó khăn với các chuỗi thông tin có độ dài lớn.
- Đạo hàm bị triệt tiêu (vanishing gradient), exploding gradient (đạo hàm bùng nổ) do các trọng số trở nên nhỏ đi khi đi qua nhiều lớp, đến khi không còn giá trị sử dụng được hoặc tăng quá mức kiểm soát khi có quá nhiều dữ liệu.

### Giải pháp cho vấn đề Vanishing Gradient (Hình 1.23):

- RNN với hàm kích hoạt ReLu.
- Gradient clipping.
- Skip connection.

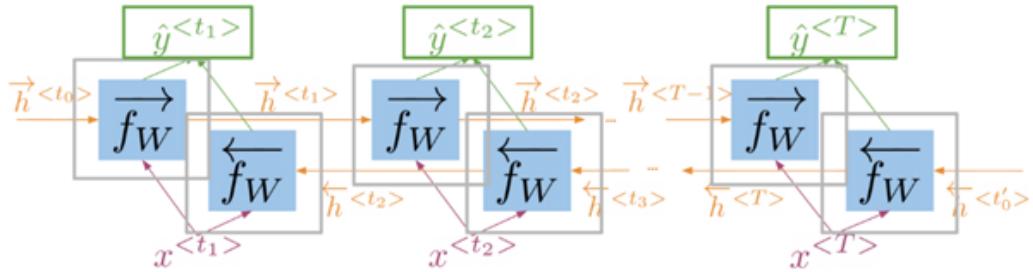


Hình 1.23: Giải pháp cho vấn đề vanishing gradient

#### 1.2.2.2 Bi-directional RNN

Bi-directional RNN là mạng RNN nhưng thay vì chỉ di chuyển một chiều dữ liệu từ trái sang phải, thì lúc này Bi-directional RNN có hai luồng dữ liệu đi theo hai hướng ngược chiều nhau (Hình 1.24).

Bi-directional RNN rất quan trọng, vì nó cho ta biết ngữ cảnh tiếp theo sau từ cần phải dự đoán trong mô hình xử lý ngôn ngữ tự nhiên.



Hình 1.24: Mô hình Bi-directional RNN

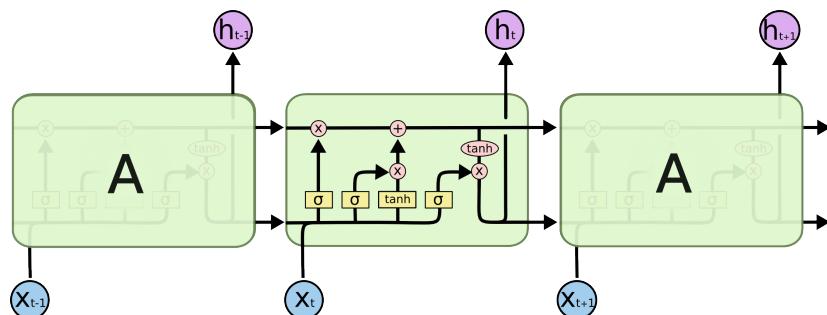
Để có thể dự đoán được  $y^{<t>}$ , ta cần hidden state của cả hai luồng và kết hợp chúng lại để tạo ra một hidden state mới và tiến hành như RNN thông thường.

### 1.2.3 Long Short-term Memory

#### 1.2.3.1 Giới thiệu mô hình

Mạng Long Short-term Memory (LSTM) là một biến thể của RNN có khả năng học hỏi những phụ thuộc theo thứ tự trong vấn đề dự đoán chuỗi dữ liệu

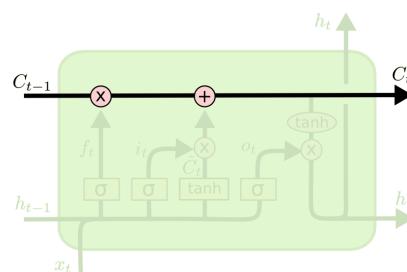
LSTM giải quyết hai vấn đề của RNN là vanishing gradient và exploding gradient bằng cách chọn lựa ra những thông tin cần thiết ở mỗi bước của mô hình và quên những thông tin còn lại. Lúc mô hình có thể lưu lại thông tin ở xa thông tin ở đầu chuỗi (Hình 1.25).



Hình 1.25: Mô hình Long Short-term Memory (Nguồn: <https://colah.github.io>)

LSTM gồm một cell state và một hidden state với ba cổng:

- Cổng Input i: cho biết bao nhiêu thông tin được input vào tại một thời điểm nhất định.
- Cổng Forget f: cho biết bao nhiêu thông tin cần quên tại một thời điểm.
- Cổng Output o: cho biết bao nhiêu thông tin truyền qua tại một thời điểm.
- Cell state: Giúp mô hình giữ lại thông tin quan trọng và quên đi thông tin không quan trọng (Hình 1.26).



Hình 1.26: Cell state

LSTM có khả năng thêm hoặc bớt đi thông tin ở Cell state thông qua các cổng Input, Output và Forget. Cổng (Gate) là một cách để lựa chọn thông tin đi qua mạng LSTM. Chúng được cấu thành từ lớp output sigmoid và một phép toán nhân. Lớp output sigmoid cho ra một số thuộc về giá trị 0 hoặc 1, quyết mỗi phần được duyệt qua mạng neural hay không.

Những tham số trong LSTM được tính như sau (Tham khảo [11]):

- Cổng Input:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \quad (1.1)$$

- Cổng Forget:

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \quad (1.2)$$

- Cổng Cell:

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \quad (1.3)$$

- Cổng Output:

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \quad (1.4)$$

- Cell State:

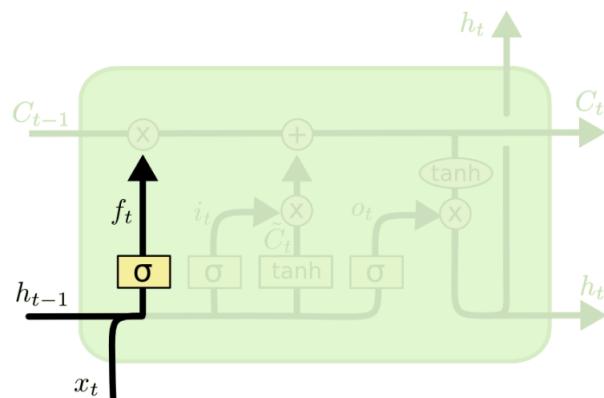
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (1.5)$$

- Hidden State:

$$h_t = o_t \odot \tanh(c_t) \quad (1.6)$$

### 1.2.3.2 Cách thức hoạt động

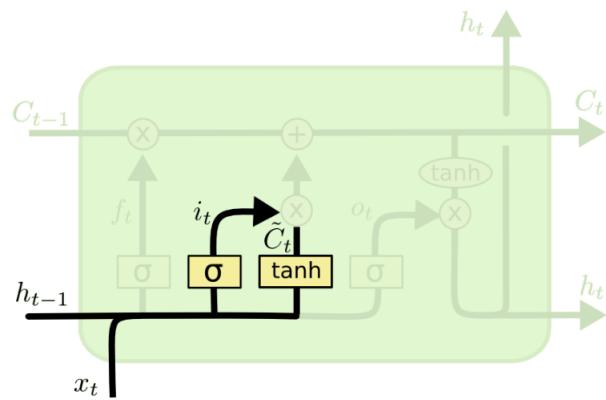
Trong mô hình LSTM, bước đầu tiên phải làm là xác định thông tin nào trong Cell state cần được quên đi bằng công thức 1.1.



Hình 1.27: Cổng Forget. Nguồn: <https://colah.github.io>

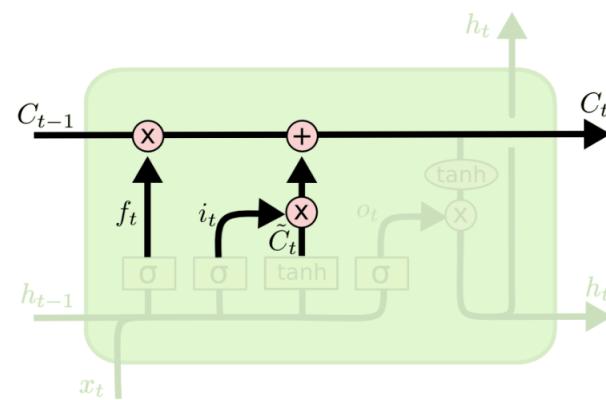
Bước tiếp theo là quyết định thông tin mới được đưa vào Cell state và gồm hai phần như sau:

- Một lớp sigmoid gọi là cổng Input sẽ quyết định giá trị nào sẽ được cập nhập theo công thức 1.1.
- Lớp tanh sẽ tạo ra một vector các giá trị ứng viên để đưa vào Cell state là  $g_t$  theo công thức 1.2.



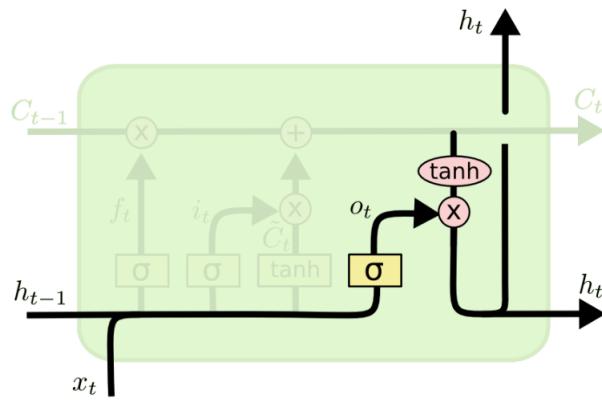
Hình 1.28: Cổng Tạo ra các giá trị ứng viên vào Cell state. Nguồn: <https://colah.github.io>

Cập nhập Cell state cũ là  $C_{t-1}$  và Cell state mới  $C_t$  bằng cách nhân giá trị cũ với  $f_t$  và cộng với giá trị mới nhân cho cổng Input theo công thức 1.4.



Hình 1.29: Cập nhập Cell state. Nguồn: <https://colah.github.io>

Bước cuối cùng là đầu ra của mô hình thông qua Cell state, lúc này sẽ đi qua lớp sigmoid để quyết định phần nào trong Cell state được đưa ra theo công thức 1.4. Sau đó Cell state sẽ đi qua lớp tanh để chuyển giá trị về khoảng (-1, 1) và nhân nó với đầu ra của cổng sigmoid theo công thức 1.3.

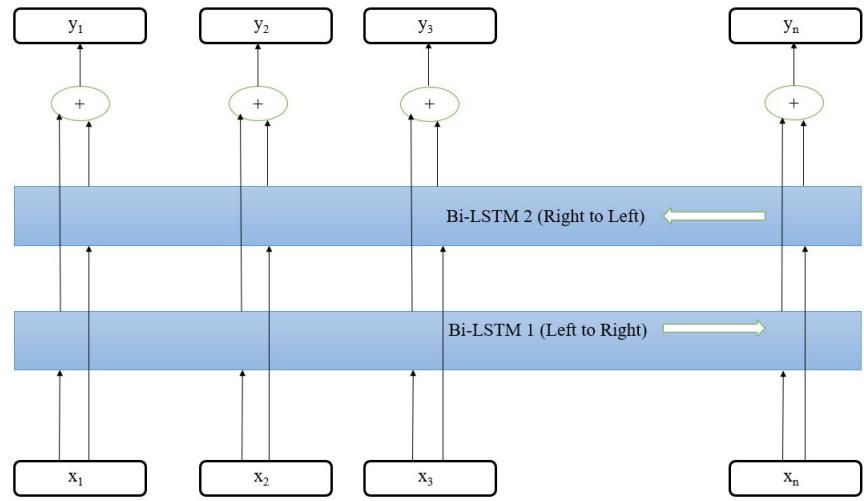


Hình 1.30: Đầu ra của Cell state. Nguồn: <https://colah.github.io>

### 1.2.3.3 Bidirectional-LSTM

Ta có thể sử dụng mạng nơ ron hồi quy theo hai chiều ngược nhau để xử lý Một đơn vị RNN sẽ làm như thường lệ, tức là ta sẽ dùng nó để học các tín hiệu đầu vào từ thời điểm ban đầu tới thời điểm kết thúc (đi xuôi). Bên cạnh đó, đơn vị RNN còn lại, ta sẽ đọc theo thứ tự thời điểm từ kết thúc trở lại ban đầu (đi ngược). Sau khi có cả hai kết quả, chúng sẽ được gom lại thành một để có thể dự đoán. Với ý tưởng như vậy, tại một thời điểm bất kỳ, mạng sẽ có được các thông tin trước và sau thời điểm  $t$  hiện tại.

Do bản chất LSTM là cải tiến của RNN, cho nên ta có thể áp dụng nó và biến nó thành mạng nơ ron dài ngắn song song (BiLSTM). Mỗi LSTM sẽ vẫn có khả năng quên thông tin cũ (cổng quên), lọc thông tin mới (cổng đầu vào), hoặc giữ bớt kết quả (cổng đầu ra) như bình thường. Chính vì vậy, các thông tin từ quá khứ tới tương lai của mạng BiLSTM đều có thể tự học để tự điều chỉnh. Dẫn tới việc với các bài toán mà ta cần biết nhiều hơn về ngữ cảnh hiện tại của nó, thì mạng BiLSTM cho kết quả tốt hơn. Hình 1.31 dưới đây mô tả mạng BiLSTM:



Hình 1.31: Hình ảnh mô tả mạng BiLSTM.

# Chương 2

## Nhận diện văn bản sử dụng CRNN và CTC Loss

### 2.1 Mô hình nhận diện văn bản với CRNN và CTC loss

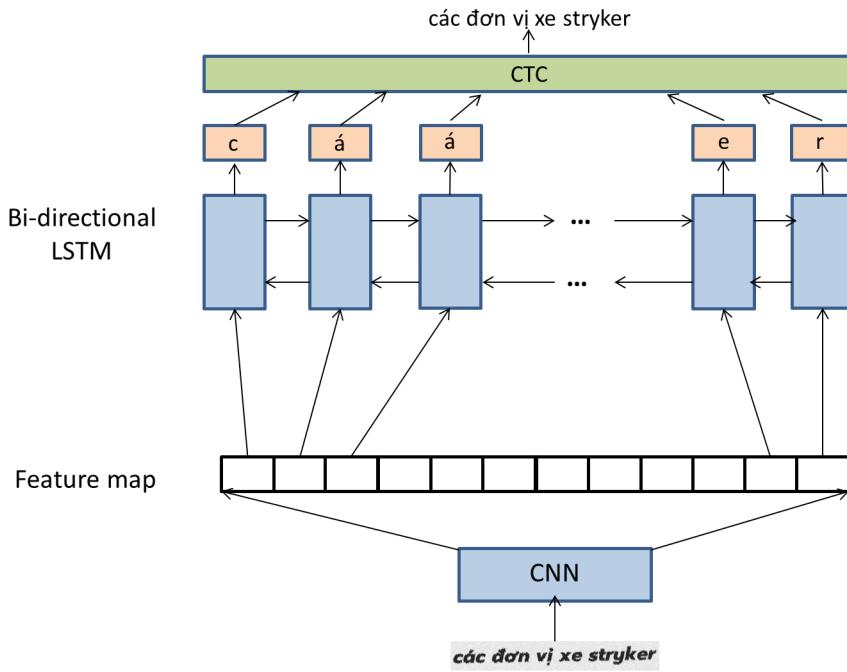
#### 2.1.1 Mô hình chung

Mô hình được sử dụng được gọi là Convolutional Recurrent Neural Network kết hợp với CTC loss, bao gồm 3 thành phần chính sau:

- Mạng nơ-ron tích chập CNN
- Mạng nơ-ron hồi quy RNN(trong mô hình này sẽ sử dụng mạng LSTM hai chiều)
- Lớp mã hóa và giải mã văn bản sử dụng hàm CTC loss

#### 2.1.2 Các bước thực hiện

Đầu vào của mô hình sẽ những ảnh chứa các vùng có văn bản, được đưa vào mạng CNN nhằm mục đích lấy ra được bản đồ đặc trưng của văn bản có trong ảnh. Từ đó, những bản đồ đặc trưng được cho vào mạng Bi-LSTM để nhận diện ký tự trong bản đồ đặc trưng theo từng chuỗi. Kết quả ở bước này là được các ký tự trong ảnh. Sử dụng hàm CTC loss để loại bỏ các ký tự rỗng hoặc sáp nhập các ký tự trùng nhau thành một

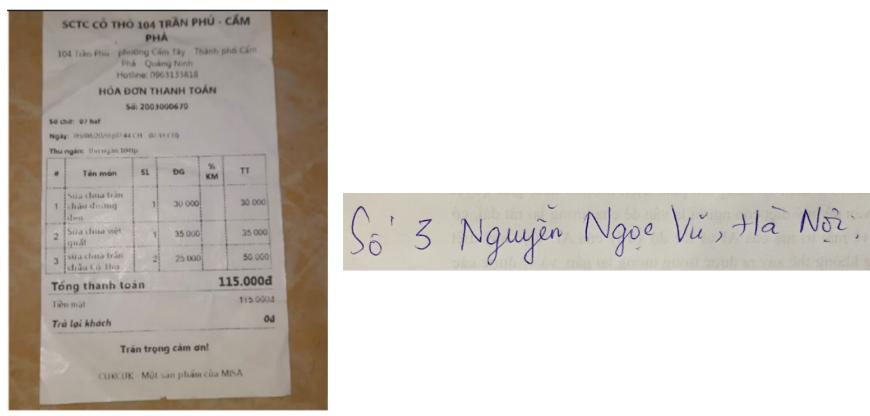


Hình 2.1: Mô tả quy trình giải quyết bài toán nhận diện văn bản Tiếng Việt

## 2.2 Thực nghiệm nhận diện văn bản

### 2.2.1 Mô tả dữ liệu và tiền xử lý

Dữ liệu huấn luyện OCR Tiếng Việt hiện nay có hai nguồn chính có thể sử dụng để xây dựng mô hình OCR: dữ liệu về hóa đơn bán hàng được tạo ra từ nghiên cứu của Xuân Sơn et al MC-OCR [17] và dữ liệu chữ viết tay trong cuộc thi Cinnamon-AI Marathon 2018 Challenge 1



Hình 2.2: Ví dụ về dataset Tiếng Việt

Do yêu cầu đặt ra của bài toán cần phải phát hiện và nhận diện chính xác

văn bản được chụp ảnh , cần phải có một tập dữ liệu phong phú chứa đầy đủ các ký tự và từ trong tiếng Việt ở dạng chữ in với nhiều font chữ khác nhau, điều mà hai tập dữ liệu đã nói ở trên không đáp ứng được

Với các lý do đã nêu ra ở trên, việc xây dựng mô hình trong khóa luận này sẽ sử dụng tập dữ liệu ảnh được tạo sinh một cách ngẫu nhiên từ một từ file văn bản chứa văn bản tiếng Anh và tiếng Việt với việc các đặc trưng sau được ngẫu nhiên:

- Font chữ
- Độ nghiêng của chữ
- Độ đậm nhạt của chữ
- Độ mờ của văn bản
- Độ biến dạng của văn bản

Tất cả văn bản sẽ được đặt trên một phông nền xám có một độ nhiễu nhất định để giả lập lại môi trường chữ in trên giấy



Hình 2.3: Ví dụ về dataset Tiếng Việt được sinh ra

Tập dữ liệu ảnh bao gồm 1624 ảnh, được chia thành 1433 ảnh ở tập huấn luyện và 191 ảnh ở tập kiểm thử. Tất cả ảnh đều có chiều cao cố định là 64 pixel, với độ dài tối đa của văn bản có trong ảnh là 182 ký tự

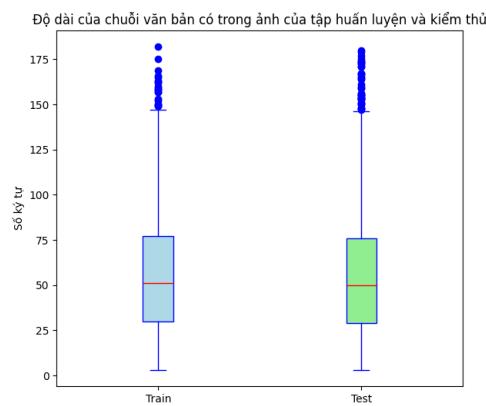
Để có thể đưa ảnh vào mô hình huấn luyện thì ảnh cần phải có kích thước là 2167x118, yêu cầu việc phải xử lý ảnh để có được chiều ảnh mong muốn. Đồng thời công đoạn này cũng bao gồm việc áp dụng Gaussian Blur cùng

Adapted Threshold để giảm độ nhiễu của ảnh, làm mịn các đường viền và chuẩn hóa ảnh đầu vào



Hình 2.4: Xử lý ảnh trước khi đưa vào mô hình

Độ dài tối đa của chuỗi văn bản trong tập huấn luyện là 180, chỉ kém độ dài tối đa của tập huấn luyện 2 ký tự. Độ dài tối thiểu của cả hai tập đều là 3, với số ký tự trung vị là 51 đối với tập huấn luyện, 50 với tập kiểm thử

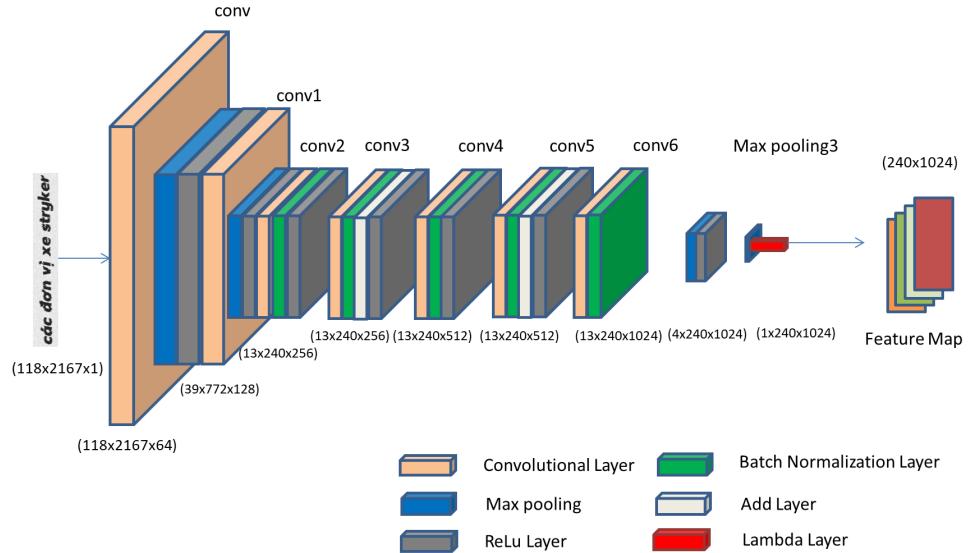


Hình 2.5: Phân bố độ dài văn bản có trong hình ảnh

## 2.2.2 Xây dựng mô hình

### 2.2.2.1 Convolutional Neural Network

Một mạng CNN tiêu chuẩn sẽ được sử dụng để trích xuất feature map từ ảnh chứa văn bản, với ảnh đầu vào đã được padding đến kích cỡ là (118x2167). Kết quả ra được sẽ là các feature map với kích thước (240x1024) (240 là số timestep trong mô hình)



Hình 2.6: Kiến trúc mạng CNN được sử dụng

Với hình minh họa 2.7, ta có thể thấy được cách thức mà mô hình sử dụng để trích xuất đặc trưng từ ảnh đầu vào với vùng được mô hình chú trọng sẽ được làm nổi bật lên trong biểu đồ nhiệt 2.7b. Vùng chứa những đặc trưng cần trích xuất phục vụ cho việc huấn luyện mô hình được nhấn mạnh bởi gam màu sáng, để lại những vùng ít được chú ý hơn

Các Khoản Trợ Cấp Cho Tù Nhân Đã Tăng Trong Giai Đoạn Từ Tháng 6 Năm 2002 Đến Tháng 6 Năm 2004 Lên \$9.6 Triệu Mỗi Tháng

(a) Hình ảnh đầu vào

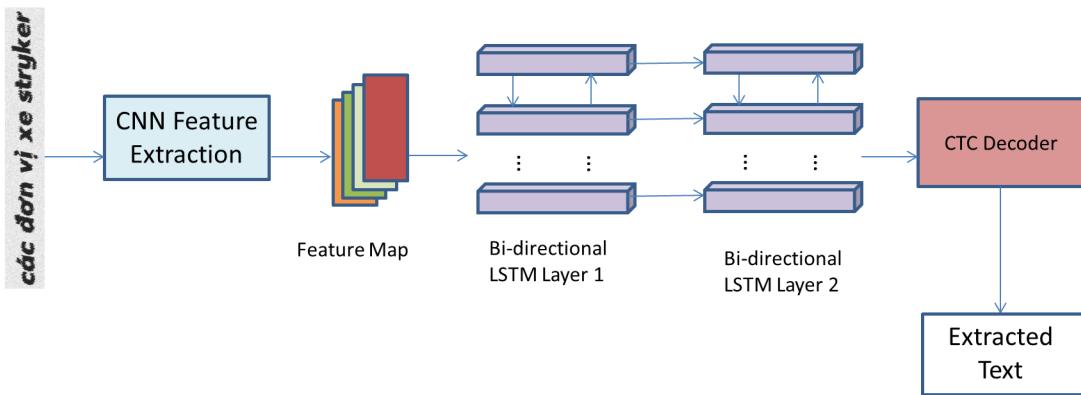


(b) Biểu đồ nhiệt của ảnh đầu vào tại lớp nhân chập(convolutional) đầu tiên

Hình 2.7: Biểu đồ nhiệt thể hiện khu vực ảnh hưởng đến đầu ra của mô hình

### 2.2.2.2 Bidirectional Long Short-Term Memory

Với feature map đã có, ta đưa chúng vào Bidirectional LSTM để có được dự đoán phân bố của từng khung hình(feature map), những dự đoán này sẽ đi qua CTC Decoder để đưa ra được văn bản cuối cùng



Hình 2.8: Ví trí của Bidirectional LSTM trong mô hình

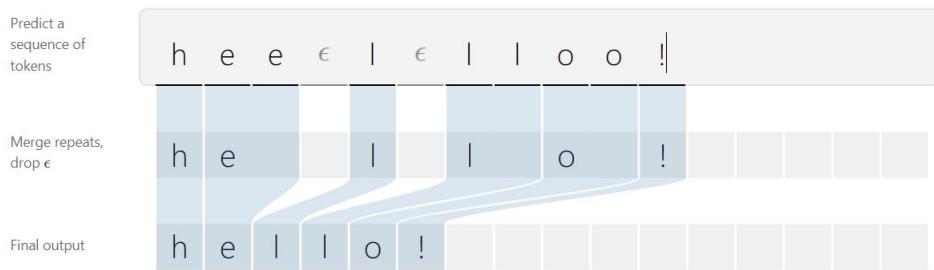
### 2.2.3 Hậu xử lý kết quả

#### 2.2.3.1 Giới thiệu

Connectionist Temporal Classification(từ đây về sau sẽ được gọi là CTC) được đề xuất trong nghiên cứu của Graves et al [4] là một kỹ thuật trong Machine Learning và Neural Networks, thường được sử dụng trong các bài toán nhận dạng và phân loại chuỗi dữ liệu, như nhận dạng ký tự trong chuỗi văn bản, nhận dạng giọng nói, hoặc nhận dạng từ vựng trong âm nhạc.

Chức năng chính của CTC là giúp mô hình có khả năng học được từ các dữ liệu đầu vào có độ dài không cố định, mà không cần thông tin về các điểm dừng (alignment) giữa dữ liệu đầu vào và đầu ra. Điều này làm cho CTC rất hữu ích trong các bài toán mà đầu ra có thể có độ dài biến đổi và không phải luôn tương ứng một cách chính xác với các phần của dữ liệu đầu vào.

Sau khi dùng mạng CNN để trích xuất đặc trưng và RNN để lấy thông tin chuỗi, cuối cùng ta thu được một ma trận và cần phải mapping nó để có được đầu ra. CTC hoạt động bằng cách tính toán xác suất của các chuỗi ký tự đầu ra( $Y = y_1, y_2, \dots, y_U$ ) có thể phù hợp với dữ liệu đầu vào( $X = x_1, x_2, \dots, x_T$ ). Sau đó, thông qua thuật toán tối ưu hóa, mô hình sẽ học cách điều chỉnh các tham số để tối ưu hóa xác suất của các chuỗi đầu ra đúng



Hình 2.9: Ví dụ về cách CTC hoạt động.Nguồn: <https://distill.pub/2017/ctc/>

### 2.2.3.2 Encoding văn bản

Với mỗi timestep layer thì sẽ tương ứng với đầu ra là một ký tự, tuy nhiên do việc CNN cắt ảnh chứa văn bản thành nhiều feature map khác nhau mà 2 hoặc nhiều timestep có thể cùng chứa một ký tự giống nhau. CTC giải quyết vấn đề này bằng cách gộp tất cả các ký tự trùng nhau thành một. Ví dụ như hee-l-loo!->hello!(Hình 2.9)

Vấn đề đặt ra ở đây là có nhiều từ có nhiều ký từ giống nhau, để tiếp tục xử lý vấn đề này thì CTC sử dụng một ký tự giả, gọi là ký tự rỗng và ký hiệu là “-”. Trong khi mã hóa văn bản, nếu gặp 2 ký tự trùng nhau, CTC sẽ chèn thêm ký tự rỗng vào giữa chúng. Ví dụ: meet -> mm-ee-ee-t, mmm-e-ee-tt. Kết quả nhận được là chuỗi các ký tự có độ dài giống nhau(Aligment). Trong quá trình decoding văn bản, nếu gặp ký tự rỗng này thì CTC hiểu rằng phải giữ lại cả 2 ký tự 2 bên ký hiệu đó.

### 2.2.3.3 CTC Loss

Để huấn luyện CRNN, ta tính hàm mất mát giữa hình ảnh và nhãn thật của nó. Thông qua ma trận đầu ra chứa xác suất mỗi ký tự tại từng timestep của CRNN, tất cả xác suất của các alignment hợp lệ sẽ được cộng lại

Công thức tính CTC Loss như sau:

$$L_{CTC} = -\log \left( \sum_{\pi} p(\pi | Y) \right)$$

Trong đó:

- $\pi$ : Đại diện cho một alignment hợp lệ giữa chuỗi đầu ra mở rộng và

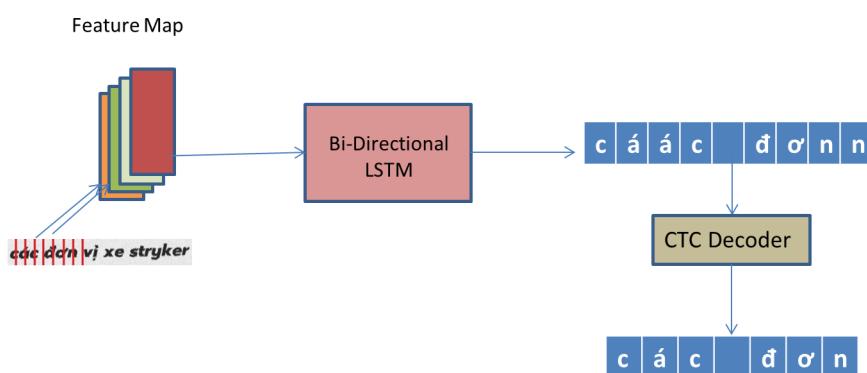
chuỗi đầu ra mong muốn

- $p(\pi|Y)$ : Xác suất của alignment  $\pi$  trong chuỗi đầu ra  $Y$
- $Y$  là chuỗi đầu ra từ mạng CRNN

#### 2.2.3.4 Decoding văn bản

Một khi CRNN đã được huấn luyện xong, kết quả mong muốn nhận được là đầu ra của những ảnh chứa văn bản chưa từng gặp bao giờ sử dụng thuật toán tìm đường đi ngắn nhất gồm hai bước chính:

1. Tính toán đường đi tốt nhất dựa trên việc ký tự nào có khả năng xuất hiện nhất với mỗi timestep
2. Loại bỏ đi ký tự rỗng và ký tự lặp để nhận được văn bản cuối cùng



Hình 2.10: Minh họa quá trình hoạt động của CTC

### 2.2.4 Kết quả thực nghiệm

#### 2.2.4.1 Phương thức đánh giá

Để đánh giá độ chính xác của mô hình khi nhận diện văn bản, ta dùng khoảng cách Levenshtein để tính tỉ lệ lỗi xuất hiện trong từng ký tự, từ và chuỗi từ giữa đầu ra được dự đoán và nhãn gốc

Khoảng cách Levenshtein là một số liệu được sử dụng để đo mức độ khác biệt giữa hai chuỗi văn bản. Nó còn được gọi là Edit distance(khoảng cách chỉnh sửa), vì nó đo số lần chỉnh sửa tối thiểu (chèn, xóa và thay thế) cần thiết để chuyển đổi một chuỗi thành một chuỗi khác.

#### **2.2.4.2 Công thức tính tỉ lệ lỗi ký tự**

$$CER = \frac{S + D + I}{N}$$

Trong đó:

- S: Số lần một ký tự được thay thế bởi ký tự khác
- D: Số lần một ký tự bị xóa đi
- I: Số lần một ký tự được thêm vào
- N: Tổng số ký tự có trong nhãn gốc

#### **2.2.4.3 Công thức tính tỉ lệ lỗi từ**

$$WER = \frac{S_w + D_w + I_w}{N_w}$$

Trong đó:

- S: Số lần một từ được thay thế bởi từ khác
- D: Số lần một từ bị xóa đi
- I: Số lần một từ được thêm vào
- N: Tổng số từ có trong nhãn gốc

#### **2.2.4.4 Công thức tính tỉ lệ lỗi chuỗi từ**

$$SER = \frac{S_f}{N}$$

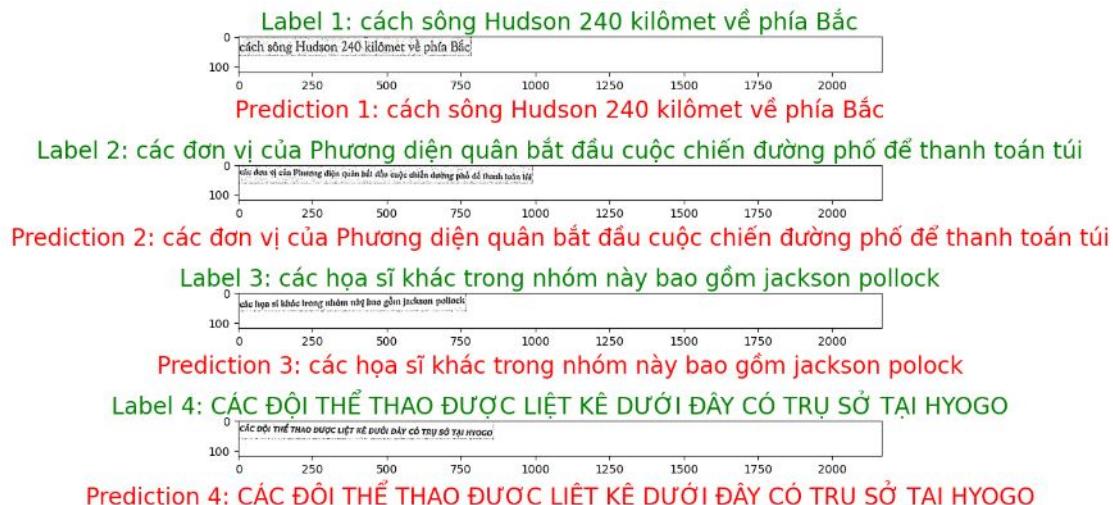
Trong đó:

- $S_f$ : Số lần chuỗi từ có sai so với nhãn thật hoặc có lỗi chính tả
- N: Tổng số chuỗi có trong dữ liệu

#### **2.2.4.5 Độ chính xác trên tập đánh giá**

Kết quả khi đưa dữ liệu ảnh từ tập kiểm thử vào mô hình và đánh giá độ chính xác như sau:

- CER: 0.0263
- WER: 0.0803
- SER: 0.2857



Hình 2.11: Kết quả dự đoán cùng với nhãn thật

#### 2.2.4.6 Phân tích và thảo luận

Mô hình có kết quả dự đoán tốt ngay cả khi gấp phải ảnh chứa văn bản có độ nghiêng lớn( $15^\circ$ ) hay văn bản dài (180 ký tự ở trường hợp văn bản có độ dài lớn nhất ở tập kiểm thử)

Thời gian dự đoán mỗi bức ảnh chứa văn bản là 1 giây mỗi ảnh, trên máy có cấu hình trung bình không có GPU riêng biệt hỗ trợ xử lý CUDA(Intel Core i5-10210U, 24GB RAM DDR4 2667 MHz). Mô hình khi dự đoán không chiếm nhiều bộ nhớ hay tài nguyên xử lý của CPU

Từ được nhận diện so với nhãn thật thường có đầy đủ dấu câu, chấm phẩy theo đúng ngữ pháp tiếng Việt. Điểm yếu của mô hình là khi gấp phải ký tự đặc biệt như dấu ngoặc đơn, dấu nháy, dấu cộng trừ,... thì mô hình không thể nhận diện được chính xác, thường hay không nhận được hoặc nhận dạng sang ký tự khác

Tuy nhiên, khi áp dụng mô hình vào dữ liệu thật lấy từ những vùng ảnh chứa văn bản được phát hiện bởi Teseract, mô hình không thể nhận diện được bất kỳ văn bản nào. Kết quả nhận được lúc này là những ký tự văn bản ngẫu nhiên rời rạc, không liên quan đến văn bản thực tế có trong ảnh

```

Image: Cropped_Image_13.png, Predicted Text: ['']
1/1 1s 1s/step
Image: Cropped_Image_14.png, Predicted Text: ['I ']
1/1 1s 1s/step
Image: Cropped_Image_15.png, Predicted Text: ['BB SS 9+']
1/1 1s 1s/step
Image: Cropped_Image_16.png, Predicted Text: ['"actu naa ncr rr rnar nar rrrrr "rr rara Tr Tr"atTmamTrmrmanumTrmrarrrmmmmam']
1/1 1s 1s/step
Image: Cropped_Image_17.png, Predicted Text: ['c +']
1/1 1s 1s/step
Image: Cropped_Image_18.png, Predicted Text: ['BC g s']
1/1 1s 1s/step
Image: Cropped_Image_19.png, Predicted Text: ['BN']
1/1 1s 980ms/step
Image: Cropped_Image_2.png, Predicted Text: ['cacaAVd "c" x ""1']
1/1 1s 960ms/step
Image: Cropped_Image_20.png, Predicted Text: ['C']
1/1 1s 973ms/step
Image: Cropped_Image_21.png, Predicted Text: ['Bu']
1/1 1s 1s/step
Image: Cropped_Image_22.png, Predicted Text: ['CI']
1/1 1s 1s/step
Image: Cropped_Image_23.png, Predicted Text: ['C']
1/1 1s 1s/step
Image: Cropped_Image_24.png, Predicted Text: [' BUD E"nnSnnSnnSnnSnnSnnSnnSnnnnnnnn3']
1/1 1s 1s/step
Image: Cropped_Image_25.png, Predicted Text: ['H']
1/1 1s 1s/step
Image: Cropped_Image_26.png, Predicted Text: [' B ']

```

Hình 2.12: Kết quả nhận dạng văn bản trong trường hợp dữ liệu thực tế

Việc này xảy ra do dữ liệu sử dụng cho việc huấn luyện có các vấn đề như sau;

- Dữ liệu tạo sinh ngẫu nhiên từ máy tính: dữ liệu có thể có các đặc trưng khác với ảnh trong điều kiện thực tế khiến cho mô hình chỉ được huấn luyện để nhận dạng văn bản có các đặc trưng giả lập này
- Dữ liệu chưa đủ lớn: Với số lượng ảnh hiện có trong tập dữ liệu với 1433 ảnh thì mô hình đã bị quá khớp dữ liệu. Do thiếu dữ liệu huấn luyện, mô hình không thể học được đầy đủ các đặc điểm cần thiết để nhận diện văn bản chính xác, dẫn đến độ chính xác và hiệu suất tổng thể kém khi áp dụng vào thực tế

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Mô hình CRNN kết hợp với CTC loss đã chứng minh khả năng mạnh mẽ trong việc nhận diện văn bản từ hình ảnh, đồng thời mang lại độ chính xác cao và hiệu quả xử lý vượt trội. Qua các chỉ số đánh giá, mô hình đã cho thấy hiệu suất ấn tượng trong việc dự đoán các văn bản có độ phức tạp khác nhau, từ những đoạn văn ngắn đến các văn bản dài, cũng như khả năng xử lý tốt ngay cả khi ảnh chứa văn bản bị nghiêng.

Điểm nổi bật của mô hình này là khả năng hoạt động hiệu quả trên các hệ thống có cấu hình phần cứng trung bình mà không cần đến GPU chuyên dụng. Điều này mở ra nhiều cơ hội ứng dụng trong thực tế, khi mà việc triển khai mô hình không đòi hỏi đầu tư quá nhiều vào phần cứng, giúp tiết kiệm chi phí và tăng tính khả thi cho các dự án quy mô nhỏ và vừa.

Mô hình cũng thể hiện khả năng tái hiện chính xác cấu trúc ngữ pháp và dấu câu trong văn bản tiếng Việt, đáp ứng tốt các yêu cầu khắt khe về ngữ pháp. Tuy nhiên, vẫn còn một số hạn chế cần được cải thiện, đặc biệt là trong việc nhận diện các ký tự đặc biệt như dấu ngoặc đơn, dấu nháy, dấu cộng và dấu trừ. Đây là những thách thức cần được giải quyết trong các nghiên cứu và phát triển tiếp theo để nâng cao hơn nữa độ chính xác và phạm vi ứng dụng của mô hình.

Nhìn chung, mô hình CRNN kết hợp với CTC loss đã chứng tỏ tính ưu việt trong việc xử lý nhận diện văn bản từ hình ảnh, với độ chính xác cao và khả năng hoạt động ổn định trên nhiều loại phần cứng. Với những kết quả đạt

được, mô hình này có tiềm năng lớn để được áp dụng rộng rãi trong các lĩnh vực như số hóa tài liệu, nhận diện văn bản trong các hệ thống giám sát, và nhiều ứng dụng khác liên quan đến xử lý ngôn ngữ tự nhiên và thị giác máy tính.

Trong tương lai, việc cải tiến mô hình để nâng cao khả năng nhận diện các ký tự đặc biệt và tối ưu hóa thời gian xử lý sẽ là những hướng phát triển quan trọng. Sự kết hợp của CRNN và CTC loss không chỉ đáp ứng tốt các yêu cầu hiện tại mà còn hứa hẹn mở ra nhiều tiềm năng ứng dụng mới, đáp ứng nhu cầu ngày càng cao của thị trường trong lĩnh vực nhận diện và xử lý văn bản từ hình ảnh.

## Hướng phát triển

Để nâng cao hơn nữa hiệu suất của mô hình CRNN kết hợp với CTC loss, có một số hướng cải thiện quan trọng cần xem xét:

Trước hết, việc mở rộng tập huấn luyện là cần thiết. Hiện tại, mô hình được huấn luyện với 1433 ảnh, con số này còn khá hạn chế. Việc tăng số lượng ảnh huấn luyện lên nhiều lần, bao gồm hàng chục nghìn hoặc thậm chí hàng trăm nghìn ảnh, sẽ giúp mô hình học được nhiều mẫu đa dạng hơn và cải thiện khả năng tổng quát hóa. Đặc biệt, cần bổ sung nhiều ảnh chứa các ký tự đặc biệt như dấu ngoặc đơn, dấu nháy, dấu cộng, dấu trừ, nhằm giúp mô hình nhận diện và xử lý tốt hơn các ký tự này.

Thứ hai, thêm ảnh thực tế vào tập huấn luyện sẽ làm phong phú thêm dữ liệu đầu vào. Thay vì chỉ sử dụng các ảnh được tạo ra trong môi trường kiểm soát, việc thêm các ảnh chụp từ thực tế, bao gồm các tài liệu giấy, biển báo, văn bản viết tay, và các loại văn bản khác sẽ giúp mô hình đối phó tốt hơn với những tình huống thực tế và cải thiện độ chính xác khi áp dụng vào các ứng dụng đời sống.

Cuối cùng, điều chỉnh kích cỡ ảnh đầu vào và padding phù hợp cũng là một hướng đi quan trọng. Nghiên cứu và thử nghiệm các kích cỡ ảnh đầu vào khác nhau có thể giúp mô hình xử lý tốt hơn các chi tiết nhỏ và các ký tự đặc biệt. Tối ưu hóa kích cỡ ảnh đầu vào sẽ giúp cải thiện khả năng nhận diện của mô hình. Bên cạnh đó, việc nghiên cứu các kỹ thuật padding thích hợp để duy trì tỷ lệ khung hình gốc của ảnh mà không làm biến dạng nội dung cũng

rất quan trọng. Padding hợp lý sẽ giúp mô hình nhận diện chính xác hơn mà không bị ảnh hưởng bởi các khoảng trống thêm vào ảnh.

Bằng cách thực hiện những cải tiến trên, mô hình CRNN kết hợp với CTC loss sẽ có thể nâng cao hiệu suất, độ chính xác và khả năng áp dụng vào các tình huống thực tế đa dạng hơn. Các bước này không chỉ cải thiện kết quả nhận diện hiện tại mà còn mở ra nhiều tiềm năng phát triển trong tương lai cho các ứng dụng liên quan đến nhận diện và xử lý văn bản từ hình ảnh.

# Tài liệu tham khảo

- [1] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. 2012.
- [2] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. Photoocr: Reading text in uncontrolled conditions. pages 785–792, 12 2013.
- [3] Boris Epshtain, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. pages 2963 – 2970, 07 2010.
- [4] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. volume 2006, pages 369–376, 01 2006.
- [5] Kaiming He. Deep residual learning for image recognition. 2016.
- [6] Duc Hoai, Huu-Thanh Duong, and Vinh Truong Hoang. Text recognition for vietnamese identity card based on deep features network. *International Journal on Document Analysis and Recognition (IJDAR)*, 24, 06 2021.
- [7] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. 09 2014.
- [8] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild, 2016.
- [9] Jiri Matas, Ondrej Chum, Martin Urban, and Tomas Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22:761–767, 09 2004.

- [10] Shuicheng Yan Min Lin, Qiang Chen. Network in network. 2014.
- [11] Ha-Thanh Nguyen, Tran Dang, and Le Nguyen. *Deep Learning Approach for Vietnamese Consonant Misspell Correction*, pages 497–504. 07 2020.
- [12] Ross Girshick Shaoqing Ren, Kaiming He and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. 2016.
- [13] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, 07 2015.
- [14] Karen Simonyan and Andrew Zisserman. Deep convolutional networks for large-scale image recognition. 2014.
- [15] R. Smith. An overview of the tesseract ocr engine. volume 2, pages 629 – 633, 10 2007.
- [16] Christian Szegedy. Going deeper with convolutions. 2015.
- [17] Xuan-Son Vu, Quang-Anh Bui, Nhu-Van Nguyen, Thi Nguyen, and Thanh Vu. Mc-ocr challenge: Mobile-captured image document recognition for vietnamese receipts. pages 1–6, 08 2021.
- [18] Pengfei Wang, Chengquan Zhang, Fei Qi, Zuming Huang, Mengyi En, Junyu Han, Jingtuo Liu, Errui Ding, and Guangming Shi. A single-shot arbitrarily-shaped text detector based on context attended multi-task learning. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM ’19. ACM, October 2019.
- [19] Tao Wang, David J. Wu, Adam Coates, and Andrew Y. Ng. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3304–3308, 2012.
- [20] Wenhai Wang, Enze Xie, Xiang Li, Wenbo Hou, Tong Lu, Gang Yu, and Shuai Shao. Shape robust text detection with progressive scale ex-

pansion network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9336–9345, 2019.

- [21] Wenhai Wang, Enze Xie, Xiaoge Song, Yuhang Zang, Wenjia Wang, Tong Lu, Gang Yu, and Chunhua Shen. Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8440–8449, 2019.
- [22] Yoshua Bengio Yann LeCun, Leon Bottou and Patrick Haffner. Gradientbased learning applied to document recognition. 1998.
- [23] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector, 2017.
- [24] Yingying Zhu, Cong Yao, and Xiang Bai. Scene text detection and recognition: recent advances and future trends. *Frontiers of Computer Science (print)*, 10, 06 2015.
- [25] Mohamad Zulkifli, Paridah Daud, and Normaiza Mohamad. *Multi Language Recognition Translator App Design Using Optical Character Recognition (OCR) and Convolutional Neural Network (CNN)*, pages 103–116. 04 2023.