Contents

[**1.** **CNN của mô hình OCR** 1](#_Toc132763060)

[**2.** **Attention OCR** 2](#_Toc132763061)

[**3.** **Transformer OCR** 4](#_Toc132763062)

[Encoder 6](#_Toc132763063)

[Self Attention Layer 7](#_Toc132763064)

[Multi Head Attention 8](#_Toc132763065)

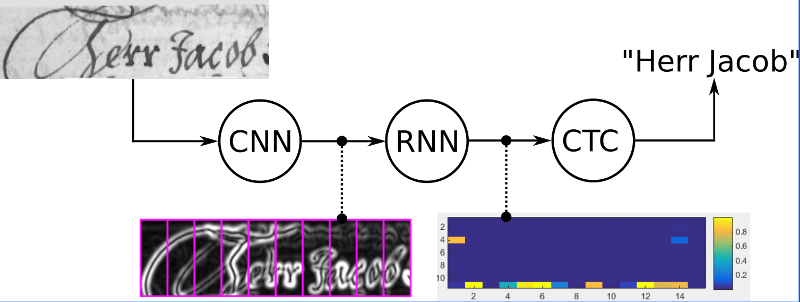
[Decoder 9](#_Toc132763066)

[**4.** **Thư viện VietOCR** 10](#_Toc132763067)

1. **CNN của mô hình OCR**

Mô hình **OCR (Optical Character Recognition)** là một ứng dụng của trí tuệ nhân tạo được sử dụng để nhận dạng các ký tự chữ viết tay hoặc in ấn trong ảnh hoặc văn bản. Trong mô hình OCR, CNN (Convolutional Neural Network) là một trong những kiến trúc mạng thường được sử dụng để thực hiện nhận dạng ký tự.

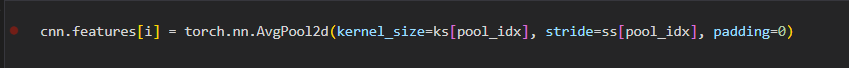
CNN là một kiến trúc mạng nơ-ron được sử dụng phổ biến trong việc xử lý hình ảnh. Trong mô hình OCR, CNN thường được sử dụng để học các đặc trưng của các ký tự bằng cách sử dụng các tầng convolution và pooling. Tầng convolution sẽ giúp mô hình học các đặc trưng của ảnh, còn tầng pooling sẽ giảm kích thước ảnh và giúp mô hình tránh overfitting.



Trong bài toán OCR, mô hình **CNN** được sử dụng nhận đầu vào là một ảnh có kích thước với chiều dài lớn hơn nhiều so với chiều rộng. Vì vậy, việc điều chỉnh tham số stride size của tầng pooling là rất quan trọng để đạt được kết quả tốt. Trong trường hợp này, tầng pooling cuối cùng được chọn có kích thước wxh=2x1. Nếu không thay đổi stride size phù hợp với kích thước ảnh, mô hình sẽ cho kết quả nhận dạng tệ.

Tuy nhiên, việc điều chỉnh tham số pooling size trong các mô hình phức tạp như resnet lại khó hơn do ảnh bị downsampling không chỉ bởi tầng pooling mà còn tại các tầng convolution khác. Trong khi đó, đối với mô hình đơn giản như VGG, việc thay đổi pooling size có thể được thực hiện dễ dàng bằng cách thay đổi stride size của tầng pooling.

Để thực hiện thay đổi stride size của tầng pooling trong mô hình VGG, có thể sử dụng mã sau trong Pytorch:



Trong đó,

* cnn là mô hình VGG
* i là chỉ số của tầng pooling cần thay đổi
* ks[pool\_idx] là kích thước kernel của tầng pooling
* ss[pool\_idx] là stride size của tầng pooling
* padding là tham số padding của tầng pooling.

1. **Attention OCR**

AttentionOCR là sự kết hợp giữa mô hình CNN và mô hình Attention Seq2Seq. Cách hoạt động của mô hình này tương tự như kiến trúc của mô hình seq2seq trong bài toán dịch máy. Với bài toán dịch máy từ tiếng việt sang anh, ta cần encode một chuỗi tiếng việt thành một vector đặc trưng, còn trong mô hình AttentionOCR, thì dữ liệu đầu vào này là một ảnh.

Mô hình Attention OCR bao gồm ba phần chính: encoder, decoder và attention mechanism. Encoder là một mạng tích chập (CNN) được sử dụng để trích xuất các đặc trưng của ảnh đầu vào. Decoder là một mạng dự báo (LSTM) được sử dụng để giải mã và dự đoán các ký tự. Attention mechanism được sử dụng để tập trung vào những phần quan trọng của ảnh để giúp mô hình nhận dạng ký tự chính xác hơn.

Diagram

Description automatically generated

Một ảnh qua mô hình CNN, sẽ cho một feature maps có kích thước channelxheightxwidth, feature maps này sẽ trở thành đầu vào cho mô hình LSTM, tuy nhiên, mô hình LSTM chỉ nhận chỉ nhận đầu vào có kích thước là hiddenxtime\_step. Một cách đơn giản và hợp lý là 2 chiều cuối cùng heightxwidth của feature maps sẽ được duổi thẳng. Feature maps lúc này sẽ có kích thước phù hợp với yêu cầu của mô hình LSTM.

Diagram

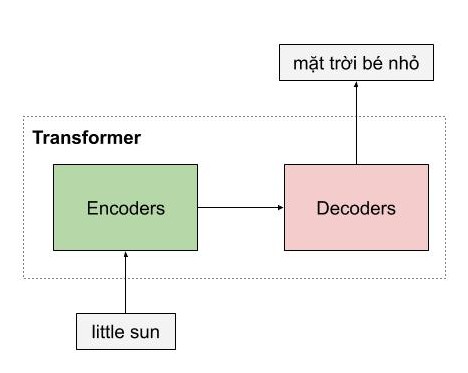
Description automatically generated

Feature maps của mô hình CNN sau khi được flatten thì được truyền vào làm input của mô hình LSTM, tại mỗi thời điểm, mô hình LSTM cần dự đoán từ tiếp theo trong ảnh là gì.

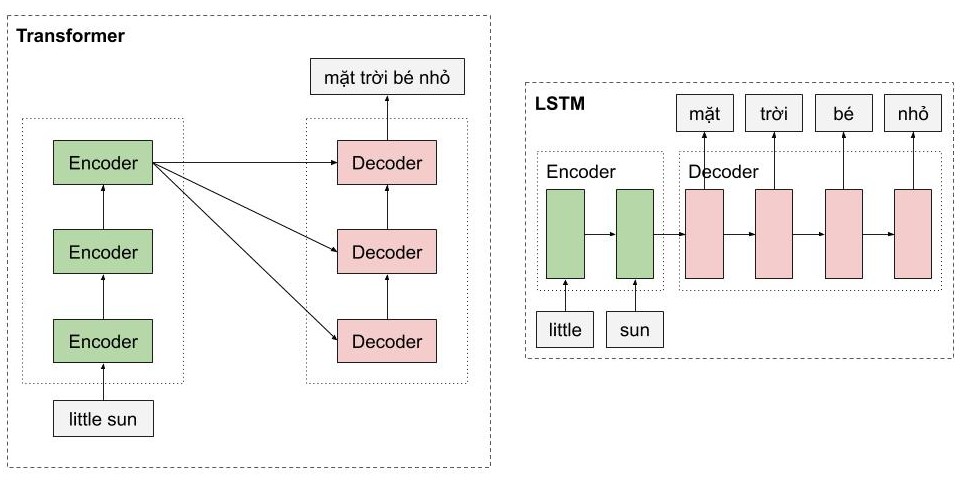
1. **Transformer OCR**

Giống như những mô hình dịch máy khác, kiến trúc tổng quan của mô hình transformer bao gồm 2 phần lớn là encoder và decoder. Encoder dùng để học vector biểu của câu với mong muốn rằng vector này mang thông tin hoàn hảo của câu đó. Decoder thực hiện chức năng chuyển vector biểu diễn kia thành ngôn ngữ đích.

Trong hình dưới, encoder của mô hình transformer nhận một câu tiếng anh, và encode thành một vector biểu diễn ngữ nghĩa của câu *little sun*, sau đó mô hình decoder nhận vector biểu diễn này, và dịch nó thành câu tiếng việt *mặt trời bé nhỏ*



Một trong những ưu điểm của transformer là mô hình này có khả năng xử lý song song cho các từ. Encoders của mô hình transformer là một dạng feedforward neural nets, bao gồm nhiều encoder layer khác, mỗi encoder layer này xử lý đồng thời các từ. Trong khi đó, với mô hình LSTM, thì các từ phải được xử lý tuần tự. Ngoài ra, mô hình Transformer còn xử lý câu đầu vào theo 2 hướng mà không cần phải stack thêm một hình LSTM nữa như trong kiến trúc Bidirectional LSTM.



Diagram

Description automatically generated with medium confidence

***Kiến trúc tổng quan của Encoder và Decoder***

## Encoder

Encoder của mô hình transformer có thể bao gồm nhiều encoder layer tượng tự nhau. Mỗi encoder layer của transformer lại bao gồm 2 thành phần chính là multi head attention và feedforward network, ngoài ra còn có cả skip connection và normalization layer.

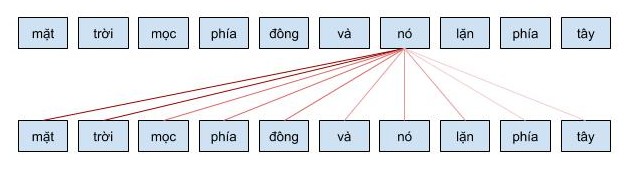
Diagram

Description automatically generated

Encoder đầu tiên sẽ nhận ma trận biểu diễn của các từ đã được cộng với thông tin vị trí thông qua positional encoding. Sau đó, ma trận này sẽ được xử lý bởi Multi Head Attention. Multi Head Attention thật chất là self-attention, nhưng mà để mô hình có thể có chú ý nhiều pattern khác nhau, tác giả đơn giản là sử dụng nhiều self-attention.

### Self Attention Layer

Self Attention cho phép mô hình khi mã hóa một từ có thể sử dụng thông tin của những từ liên quan tới nó. Ví dụ khi từ **nó** được mã hóa, nó sẽ chú ý vào các từ liên quan như là **mặt trời.**

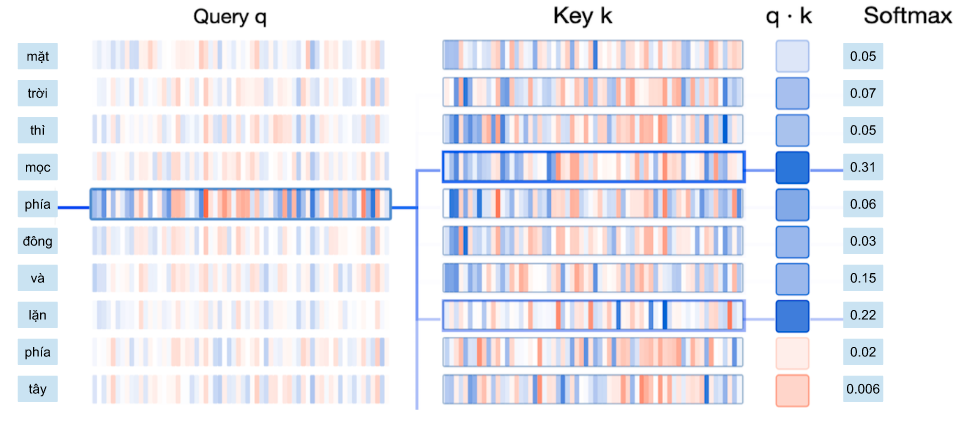


Với một từ cho trước, cơ chế này sẽ cho phép mô hình tìm kiếm trong cách từ còn lại, từ nào “giống” để sau đó thông tin sẽ được mã hóa dựa trên tất cả các từ trên.

Đầu tiên, với môi từ ta cần tạo ra 3 vector: query, key, value vector bằng cách nhân ma trận biểu diễn các từ đầu vào với ma trận học tương ứng.

* query vector: vector dùng để chứa thông tin của từ được tìm kiếm, so sánh. Giống như là câu query của google search.
* key vector: vector dùng để biểu diễn thông tin các từ được so sánh với từ cần tìm kiếm ở trên. Ví dụ, như các trang web mà google sẽ so sánh với từ khóa mà bạn tìm kiếm.
* value vector: vector biểu diễn nội dung, ý nghĩa của các từ. Các bạn có thể tượng tượng, nó như là nội dung trang web được hiển thị cho người dùng sau khi tìm kiếm.

Để tính tương quan, ta đơn giản chỉ cần tính tích vô hướng dựa các vector query và key. Sau đó dùng hàm softmax để chuẩn hóa chỉ số tương quan trong đoạn 0-1, và cuối cùng, tính trung bình cộng có trọng số giữa các vector values sử dụng chỉ số tương quan mới tính được.



### Multi Head Attention

MHA hoạt động bằng cách sử dụng nhiều đầu attention (heads) để học các thông tin khác nhau về mối quan hệ giữa các từ. Mỗi head attention được sử dụng để tính toán trọng số attention cho các từ theo cách khác nhau, sau đó kết hợp các trọng số attention này để tạo ra một ma trận trọng số attention cuối cùng.

Diagram

Description automatically generated

Mỗi đầu attention bao gồm ba lớp fully connected tương ứng với ma trận trọng số query, key và value. Đầu tiên, ma trận trọng số query được nhân với ma trận trọng số key để tính toán độ tương đồng giữa các từ. Sau đó, độ tương đồng được đưa vào hàm softmax để tạo ra ma trận trọng số attention. Cuối cùng, ma trận trọng số attention được nhân với ma trận trọng số value để tạo ra đầu ra cuối cùng.

Tại mỗi đầu attention, một hàm linear layer được sử dụng để trích xuất các tính năng đại diện cho từng head attention. Các đầu attention này được kết hợp lại với nhau bằng cách xếp chồng chúng và truyền qua một lớp fully connected cuối cùng để tạo ra đầu ra của layer MHA.

MHA giúp tăng cường khả năng học của Transformer bằng cách cho phép mô hình học các mối quan hệ khác nhau giữa các từ trong câu đầu vào. Nó cũng giúp giảm độ phức tạp tính toán của Attention bằng cách sử dụng nhiều đầu attention thay vì sử dụng một đầu attention duy nhất.

# Decoder

Decoder thực hiện chức năng giải mã vector của câu nguồn thành câu đích, do đó decoder sẽ nhận thông tin từ encoder là 2 vector key và value. Kiến trúc của decoder rất giống với encoder, ngoại trừ có thêm một multi head attention nằm ở giữa dùng để học mối liên quan giữ từ đang được dịch với các từ được ở câu nguồn.

Diagram

Description automatically generated

1. **Thư viện VietOCR**

Thư viện VietOCR là một thư viện mã nguồn mở được sử dụng để nhận dạng ký tự quang học (OCR) và chuyển đổi ảnh thành văn bản cho ngôn ngữ tiếng Việt. Nó được phát triển bởi một nhóm các nhà phát triển tại Viện Khoa học và Công nghệ Việt Nam.

VietOCR được xây dựng trên nền tảng Java và sử dụng công nghệ nhận dạng OCR Tesseract để thực hiện chức năng nhận dạng ký tự. Nó cũng cung cấp một số tính năng khác như xử lý ảnh trước khi nhận dạng, hỗ trợ định dạng tệp đầu vào và đầu ra khác nhau và có khả năng hoạt động trên nhiều hệ điều hành như Windows, Linux và macOS.

Thư viện VietOCR được tác giả Quốc xây dựng tại [đây](https://github.com/pbcquoc/vietocr) với mục đích hỗ trợ các cộng đồng có thể sử dụng để giải quyết các bài toán liên quan đến OCR trong công nghiệp. Thư viện cung cấp cả 2 kiến trúc AtentionOCR và TransformerOCR. Tuy kiến trúc TransformerOCR hoạt động khá tốt trong NLP.

Tác giả có cung cấp pretrained model được mình huấn luyện trên tập dữ liệu 10m ảnh với kết quả trong bảng sau mà ta sẽ sử dụng trong bài toán. 