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

TIỂU LUẬN
XỬ LÝ NGÔN NGỮ TỰ NHIÊN

XÂY DỰNG MÔ HÌNH SEQ2SEQ THÊM DẤU TIẾNG VIỆT

Nhóm sinh viên thực hiện:

1. NGUYỄN LUÔN MONG ĐỔ (NT)

2. VÕ ĐẠT VĂN

3. NGUYỄN TIẾN NHẬT

Tên học phần: **XỬ LÝ NGÔN NGỮ TỰ NHIÊN - NHÓM 1**

Giáo viên hướng dẫn: **Giảng viên Đoàn Thị Hồng Phước**

Huế, 12 - 2023

Mục lục

1 GIỚI THIỆU **5**

1.1 Giới thiệu về bài toán 5

1.2 Mục tiêu của bài báo cáo 5

1.3 Cấu trúc của bài báo cáo 6

2 CÁC CÁCH TIẾP CẬN **7**

2.1 Mô hình N-gram 7

2.2 Mô hình RNN 8

2.3 Mô hình Seq2Seq 9

2.4 Mô hình Transformer 10

3 PHƯƠNG PHÁP THỰC HIỆN **12**

3.1 Giới thiệu Sequence to Sequence Learning with Neural Networks . . 12

3.2 Giới thiệu Attention 16

3.2.1 Cách attention làm việc 17

4 XÂY DỰNG MÔ HÌNH SEQUENCE TO SEQUENCE **19**

4.1 Định nghĩa lớp embeddings 19

4.2 Xây dựng lớp encoder 20

4.3 Xây dựng lớp decoder 21

4.4 Thông số mô hình Sequence to Sequence đã xây dựng 22

5 PHƯƠNG PHÁP ĐÁNH GIÁ MÔ HÌNH, KẾT QUẢ **23**

5.1 Giới thiệu về phương pháp đánh giá 23

5.1.1 BLEU Score 23

5.1.2 TER 24

5.2 Kết quả xây dựng mô hình 25
5.2.1 Mô hình không áp dụng Attention 25
5.2.2 Mô hình áp dụng Attention 25

6 KẾT LUẬN **27**

Danh sách hình vẽ

2.1 Minh họa mô hình RNN. 9

3.1 Mô tả cách mà mô hình Seq2Seq đọc một câu đầu vào "ABC"và tạo ra một câu đầu ra "WXYZ". Sau khi tạo ra từ cuối cùng của câu đầu ra, mô hình dừng lại và không tiếp tục dự đoán. Nguồn: [6] 12
3.2 Encoder và Decoder 13
3.3 Cấu trúc của encoder 13
3.4 Cấu trúc của decoder 13
3.5 Nút cổ chai (bottleneck) 14
3.6 Trạng thái ẩn cuối cùng của bộ mã hóa sẽ buộc tất cả thông tin từ toàn bộ câu nguồn đi qua nút cổ chai biểu diễn này, nguồn [3] 15
3.7 Lấy tất cả các trạng thái của encoder 15
3.8 Áp dụng attention 16
3.9 Mô hình seq2seq truyền thống 17
3.10 Sử dụng tất cả các trạng thái của encoder tại bước (timestep) decoder 17
3.11 Công thức attention 17
3.12 Minh họa quá trình tính toán của cơ chế Attention tại một bước giải mã cụ thể, giúp hiểu rõ hơn về cách thức hoạt động của cơ chế Attention [3] 18

4.1 Hình ảnh minh họa mô hình seq2seq 19
4.2 Hình ảnh minh họa thông số mô hình seq2seq 22
4.3 Hình ảnh minh họa thông số mô hình seq2seq 22

5.1 Hình ảnh minh họa kết số lượng câu thỏa mãn ngưỡng Bleu Score khi mô hình không áp dụng attention 25

5.2 Hình ảnh minh họa kết số lượng câu thỏa mãn ngưỡng Bleu Score khi mô hình áp dụng attention 26

Chương 1

GIỚI THIỆU

1.1 Giới thiệu về bài toán

Ngôn ngữ đóng vai trò quan trọng trong việc truyền đạt thông tin và kiến thức.Trong ngôn ngữ tiếng Việt, một bài toán ngôn ngữ nhỏ nhưng mang tầm quan trọng lớn đang thu hút sự chú ý của cộng đồng nghiên cứu - đó là bài toán "thêm dấu tiếng Việt". Với việc công nghệ trí tuệ nhân tạo và internet đang phát triển, người dùng và máy đang tương tác với nhau nhiều hơn thông qua văn bản. Việc cải thiện thời gian đánh máy nhưng vẫn giữ được sự chính xác và rõ ràng trong việc truyền đạt ý nghĩa là một thách thức. Nếu máy có thể dịch được ngôn ngữ tiếng Việt không dấu thành ngôn ngữ tiếng Việt có dấu sẽ giúp vấn đề này được giải quyết nhanh chóng.

Nhận thấy được tiềm năng của bài toán, bài báo cáo này sẽ tiến hành giải quyết bài toán thêm dấu tiếng Việt bằng mô hình sequen to sequen và đánh giá các kết quả thu được từ thực nghiệm.

1.2 Mục tiêu của bài báo cáo

Các mục tiêu chính của bài báo cáo bao gồm:

- Tìm hiểu các cách tiếp cận để xử lý bài toán thêm dấu tiếng Việt;
- Lựa chọn xây dựng mô hình Seq2seq và kết hợp Attention để giải quyết bài toán;
- Nghiên cứu các phương pháp đánh giá độ chính xác của các bài toán dịch máy;
- Đánh giá các kết quả thu được từ mô hình seq2seq vừa được xây dựng;

1.3 Cấu trúc của bài báo cáo

Bài báo cáo gồm các phần như sau:

- Chương 1: Giới thiệu.
- Chương 2: Các cách tiếp cận
- Chương 3: Phương pháp thực hiện
- Chương 4: Xây dựng mô hình Sequence to Sequence
- Chương 5: Phương pháp đánh giá mô hình, kết quả
- Chương 6: Kết luận

Chương 2

CÁC CÁCH TIẾP CẬN

Để giải quyết bài toán thêm dấu tiếng Việt, cần xây dựng một mô hình ngôn ngữ để dịch từ tiếng Việt không dấu sang tiếng Việt có dấu.

Dưới đây là nội dung một số cách tiếp cận:

2.1 Mô hình N-gram

Đây là mô hình ngôn ngữ đơn giản nhất. Mô hình N-gram tổng quát, tương ứng với mô hình Markov bậc n – 1, giải quyết các bài toán ước lượng xác suất của một chuỗi phần tử dựa trên giả thiết Markov: xác suất xuất hiện một phần tử trong một chuỗi chỉ phụ thuộc vào n – 1 phần tử ngay trước đó. Trong mô hình ngôn ngữ N-gram, mỗi N-gram là một chuỗi n từ. Trong ví dụ chuỗi từ tiếng Việt "đấy là một cái mũ", mỗi 1-gram (unigram) là một từ độc lập (ví dụ "đấy", "là", "một"), 2-gram (bigram) là chuỗi 2 từ (ví dụ "đấy là", "là một", "một cái"), 3-gram (trigram) là chuỗi ba từ (ví dụ "đấy là một", "là một cái", "một cái mũ").

Với bài toán ước lượng xác suất P (w | h), các mô hình ngôn ngữ N-gram ước lượng xác suất xuất hiện của một từ chỉ dựa vào n-1 từ trước đó, do vậy chỉ cần đếm trong kho ngữ liệu huấn luyện các chuỗi từ có độ dài tối đa bằng n, không quan tâm tới việc đếm tần suất các chuỗi có độ dài lớn hơn n.

Với bài toán ước lượng xác suất của một chuỗi N từ, xác suất cần tính đưa về bài toán tính xác suất một từ khi biết n – 1 từ trước đó:

$$P(w_1,\ldots,w_n) = P(w_1)\cdot P(w_2|w_1)\cdot P(w_3|w_1,w_2)\cdot\ldots\cdot P(w_n|w_1,\ldots,w_{n-1})$$

Ưu điểm của N-gram:

- N-gram là một phương pháp đơn giản và dễ triển khai, không đòi hỏi nhiều tài nguyên tính toán so với một số phương pháp phức tạp khác.
- N-gram thường được sử dụng hiệu quả trong nhiều ứng dụng như xác định ngôn ngữ, dự đoán từ tiếp theo, và các nhiệm vụ khác trong xử lý ngôn ngữ tự nhiên.

Nhược điểm của N-gram:

- Kích thước của N-gram cần được chọn sao cho đủ lớn để nắm bắt ngữ cảnh quan trọng mà không làm tăng đáng kể về kích thước của mô hình. Tăng kích thước có thể dẫn đến vấn đề về bộ nhớ.
- Nếu một từ mới xuất hiện trong văn bản, N-gram có thể gặp khó khăn trong việc dự đoán nó, đặc biệt là với N nhỏ.
- Không xử lý được sự phụ thuộc xa giữa các từ trong một câu, khiến cho việc dự đoán từ tiếp theo có thể không chính xác.
- N-gram không có khả năng hiểu biết về ý nghĩa của các từ và cấu trúc ngữ pháp, chỉ dựa vào thông tin thống kê từ ngữ cảnh.

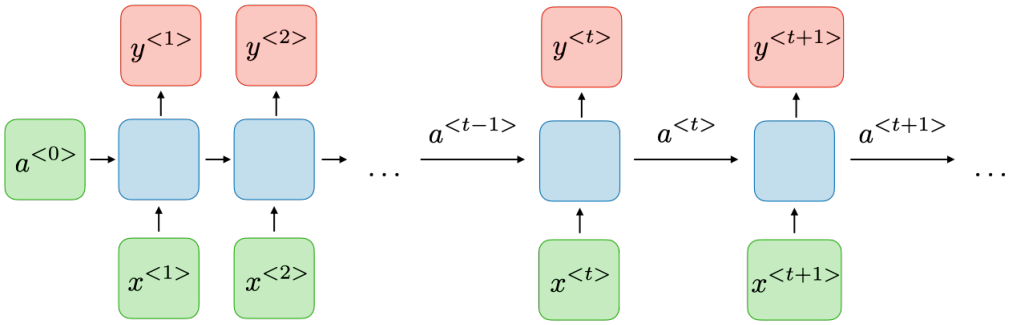
2.2 Mô hình RNN

Mô hình RNN là mô hình mạng nơ-ron sử dụng một bộ nhớ để lưu lại thông tin từ từ những bước tính toán xử lý trước để dựa vào nó có thể đưa ra dự đoán chính xác nhất cho bước dự đoán hiện tại.

Một lớp của mạng neural cho phép đầu ra được sử dụng như đầu vào trong khi có các trạng thái ẩn.

Mô hình RNN gồm có 4 loại:

- **One to one**: mẫu bài toán cho Neural Network (NN) và Convolutional Neural Network (CNN), 1 input và 1 output.
- **One to many**: bài toán có 1 input nhưng nhiều output.
- **Many to one**: bài toán có nhiều input nhưng chỉ có 1 output.
- **Many to many**: bài toán có nhiều input và nhiều output.



Hình 2.1: Minh họa mô hình RNN.

Ưu điểm của RNN:

- Khả năng xử lí đầu vào với bất kì độ dài nào
- Kích cỡ mô hình không tăng theo kích cỡ đầu vào
- Quá trình tính toán sử dụng các thông tin cũ
- Trọng số được chia sẻ trong suốt thời gian

Nhược điểm của RNN:

- Tính toán chậm
- Khó để truy cập các thông tin từ một khoảng thời gian dài trước đây
- Không thể xem xét bất kì đầu vào sau này nào cho trạng thái hiện tại

2.3 Mô hình Seq2Seq

Mô hình Sequence-to-Sequence được đề xuất bởi Sutskever et al. vào năm 2014 và được sử dụng để tạo ra một chuỗi các token của câu trong ngôn ngữ đích $y = \{1, \ldots, Y_m\}$ làm câu bản dịch tương ứng cho một chuỗi các token của câu trong ngôn ngữ nguồn $x = \{x_1, \ldots, n\}$ được cung cấp trước. Mục tiêu của quá trình huấn luyện là tối ưu hóa xác suất có điều kiện

$$P(Y_1, \ldots, Y_m | X_1, \ldots, X_n)$$

với giá trị của m là độ dài của chuỗi đầu ra có thể khác với n là độ dài của chuỗi đầu vào. Mô hình này sử dụng kiến trúc Encoder-Decoder và thông thường thì mạng RNN hoặc những cải tiến như mạng LSTM và GRU sẽ được sử dụng cho cả bộ Encoder và bộ Decoder. Đặc biệt, mạng LSTM được sử dụng để giải quyết các vấn đề phụ thuộc

dài, ghi nhớ và biểu diễn mối quan hệ của các thông tin phụ thuộc vào ngữ cảnh trong câu văn bản.

Ưu điểm của Seq2Seq:

- Mô hình Seq2Seq có khả năng xử lý chuỗi đầu vào và đầu ra có độ dài khác nhau, điều này là một ưu điểm lớn so với các mô hình truyền thống.
- Với khả năng học các biểu diễn phức tạp của chuỗi, mô hình Seq2Seq có thể xử lý dữ liệu chuỗi có ý nghĩa thực tế như văn bản, ngôn ngữ tự nhiên, và mã hóa/ghi lại thông tin.
- Bằng cách sử dụng kiến trúc Encoder-Decoder, mô hình Seq2Seq có khả năng học biểu diễn ngữ cảnh của chuỗi đầu vào, giúp nó hiểu được thông tin liên quan đến chuỗi.

Nhược điểm của mô hình Seq2Seq:

- Seq2Seq có thể gặp khó khăn khi xử lý các chuỗi có độ dài lớn, vì nó phải duy trì trạng thái ẩn qua nhiều bước thời gian, dẫn đến vấn đề vanishing gradient.
- Mô hình Seq2Seq có thể gặp khó khăn khi xử lý thông tin phụ thuộc xa trong chuỗi, đặc biệt khi áp dụng cho các nhiệm vụ yêu cầu hiểu biết ngữ nghĩa sâu sắc.
- Mô hình có thể gặp khó khăn trong việc dự đoán các từ mới mà nó chưa thấy trong quá trình huấn luyện, đặc biệt khi sử dụng mô hình với độ dài ngữ cảnh lớn.

2.4 Mô hình Transformer

Transformer là một mô hình học sâu được thiết kế để phục vụ giải quyết nhiều bài toán trong xử lý ngôn ngữ và tiếng nói, ví dụ như bài toán dịch tự động, bài toán sinh ngôn ngữ, phân loại, nhận dạng thực thể, nhận dạng tiếng nói, chuyển văn bản thành tiếng nói. Tuy nhiên, khác với RNNs, Transformer không xử lý các phần tử trong một chuỗi một cách tuần tự. Nếu dữ liệu đầu vào là một câu ngôn ngữ tự nhiên, Transformer không cần phải xử lý phần đầu câu trước rồi mới tới phần cuối câu. Do tính năng này, Transformer có thể tận dụng khả năng tính toán song song của GPU và giảm thời gian xử lý đáng kể.

Mô hình Transformer giải quyết vấn đề chính của các kiến trúc mạng truyền thống là sự phụ thuộc dài hạn trong dữ liệu bằng cách sử dụng cơ chế chú ý (attention mechanism). Thay vì truyền thông tin qua các lớp nơ-ron theo thứ tự, Transformer cho phép mỗi điểm dữ liệu tương tác với tất cả các điểm khác trong chuỗi đầu vào và đầu ra. Điều này giúp mô hình hiểu được ngữ cảnh toàn cục của dữ liệu và giải quyết các vấn đề phụ thuộc dài hạn.

Ưu điểm của Transformer:

- Transformer cho phép mô hình xử lý thông tin toàn cục và phụ thuộc dài hạn, giúp cải thiện hiệu suất so với các kiến trúc truyền thống, đặc biệt là trong các nhiệm vụ liên quan đến ngôn ngữ tự nhiên.
- Cơ chế chú ý trong Transformer giúp mô hình học được các phụ thuộc xa và không phụ thuộc vào độ dài câu.
- Transformer có thể huấn luyện trên tập dữ liệu lớn mà không gặp vấn đề phụ thuộc dài hạn, giúp nó học được những đặc trưng phức tạp và cải thiện khả năng tổng quát hóa.
- Có thể tiền huấn luyện Transformer trên một lượng lớn dữ liệu, sau đó tinh chỉnh (fine-tune) cho các nhiệm vụ cụ thể, giúp cải thiện hiệu suất của mô hình.

Nhược điểm của Transformer:

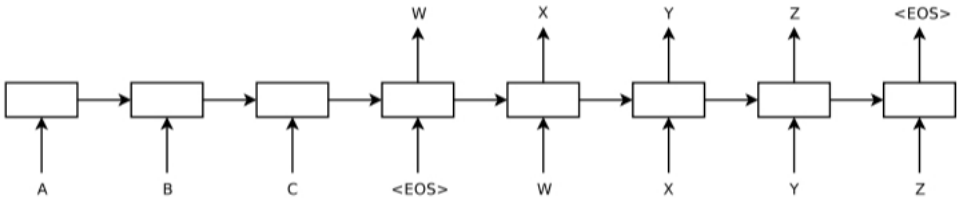
- Transformer yêu cầu một lượng lớn tính toán, đặc biệt là khi xử lý dữ liệu lớn và sử dụng các mô hình lớn. Điều này có thể làm tăng chi phí tính toán và yêu cầu phần cứng mạnh mẽ.
- Các mô hình Transformer thường có số lượng tham số lớn, điều này có thể làm tăng yêu cầu về bộ nhớ và khả năng tính toán.

Chương 3

PHƯƠNG PHÁP THỰC HIỆN

3.1 Giới thiệu Sequence to Sequence Learning with Neural Networks

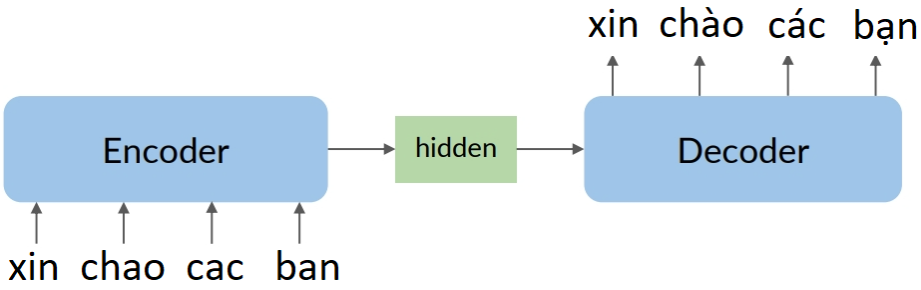
Sequence to Sequence Learning with Neural Networks (seq2seq) là một mô hình học sâu được sử dụng để ánh xạ một chuỗi đầu vào sang một chuỗi đầu ra. Mô hình này thường sử dụng mạng nơ-ron hồi quy (RNN) hoặc mạng nơ-ron dài ngắn hạn (LSTM) [4] để xử lý chuỗi đầu vào và tạo ra chuỗi đầu ra tương ứng [6].
Trong ngữ cảnh của mô hình Seq2Seq, chuỗi đầu vào và chuỗi đầu ra có thể có độ dài khác nhau, và mô hình có khả năng học cách ánh xạ từ một loạt các đầu vào có độ dài và cấu trúc khác nhau sang một loạt các đầu ra tương ứng. Điều này làm cho Seq2Seq trở thành một công cụ mạnh mẽ cho các vấn đề liên quan đến xử lý ngôn ngữ tự nhiên và các tác vụ liên quan đến chuỗi.



Hình 3.1: Mô tả cách mà mô hình Seq2Seq đọc một câu đầu vào "ABC"và tạo ra một câu đầu ra "WXYZ". Sau khi tạo ra từ cuối cùng của câu đầu ra, mô hình dừng lại và không tiếp tục dự đoán. Nguồn: [6]

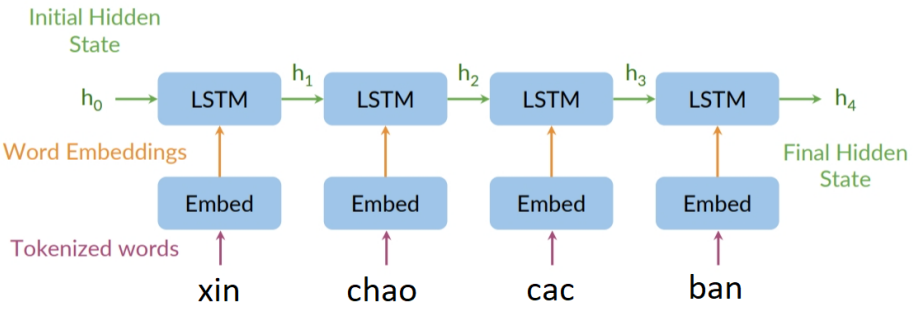
Mô hình seq2seq bao gồm hai phần chính: bộ mã hóa (encoder) và bộ giải mã (decoder). Bộ mã hóa chuyển đổi chuỗi đầu vào thành một vector đại diện cho chuỗi đầu vào, trong khi bộ giải mã sử dụng vector này để tạo ra chuỗi đầu ra. Mô hình này thường được sử dụng trong các tác vụ mà đầu ra có thể có độ dài khác nhau so với

đầu vào, và không tương ứng từng từ một.



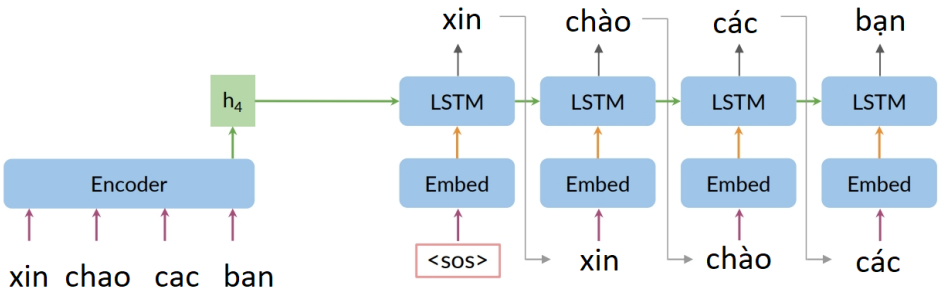
Hình 3.2: Encoder và Decoder

Cụ thể, encoder thường sử dụng một mạng nơ-ron hồi quy (RNN) hoặc mạng nơ-ron dài ngắn hạn (LSTM) để xử lý chuỗi đầu vào. Quá trình này tạo ra một vector đại diện cho chuỗi đầu vào, chứa thông tin quan trọng về chuỗi đó. Vector này thường có kích thước cố định và chứa thông tin tổng hợp từ toàn bộ chuỗi đầu vào.



Hình 3.3: Cấu trúc của encoder

Decoder cũng sử dụng một RNN hoặc LSTM để tạo ra chuỗi đầu ra từ vector đại diện của chuỗi đầu vào. Bộ decoder tạo ra từng từ trong chuỗi đầu ra một cách tuần tự, sử dụng thông tin từ vector đại diện và từ trước đó trong chuỗi đầu ra để dự đoán từ tiếp theo. Quá trình này tiếp tục cho đến khi một token kết thúc câu được sinh ra, hoặc khi đạt đến độ dài tối đa cho chuỗi đầu ra.



Hình 3.4: Cấu trúc của decoder

Mô hình seq2seq có những ưu điểm và nhược điểm sau đây:

Ưu điểm:

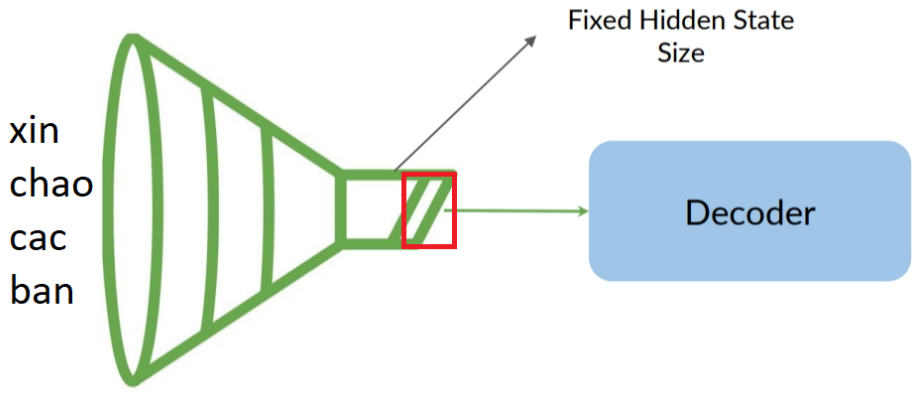
- **Linh hoạt trong xử lý chuỗi đầu vào và đầu ra có độ dài khác nhau:** Mô hình seq2seq có thể xử lý các tác vụ mà đầu ra có thể có độ dài khác nhau so với đầu vào, và không tương ứng từng từ một.
- **Áp dụng rộng rãi:** Mô hình này đã được áp dụng thành công trong nhiều lĩnh vực, bao gồm dịch máy, tóm tắt văn bản và sinh văn bản tự nhiên.
- **Tính tự nhiên trong sinh văn bản:** Seq2seq có khả năng sinh ra văn bản tự nhiên, giúp tạo ra các ứng dụng chatbot và tóm tắt văn bản tự nhiên.

Nhược điểm:

- **Đòi hỏi lượng dữ liệu lớn:** Mô hình seq2seq thường yêu cầu lượng dữ liệu lớn để huấn luyện hiệu quả, đặc biệt là trong các tác vụ phức tạp như dịch máy.
- **Độ phức tạp tính toán:** Mô hình này có thể đòi hỏi tài nguyên tính toán lớn, đặc biệt là khi sử dụng các biến thể phức tạp như LSTM.
- **Đối mặt với vấn đề mô hình hóa chuỗi dài:** Trong một số trường hợp, mô hình seq2seq có thể gặp khó khăn khi xử lý các chuỗi rất dài do vấn đề mô hình hóa.

Phân tích về nhược cuối cùng

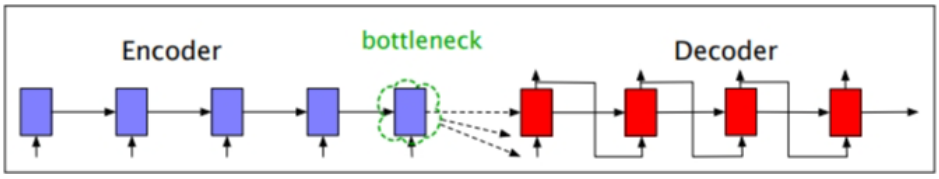
Với nhược điểm cuối cùng là khi đầu vào của mô hình là 1 câu rất dài thì bộ encoder chỉ trả về trạng thái cuối cùng hãy còn được gọi là context vector để làm trạng thái đầu vào của decoder dẫn đến mất mát thông tin của những từ đầu câu.



Hình 3.5: Nút cổ chai (bottleneck)

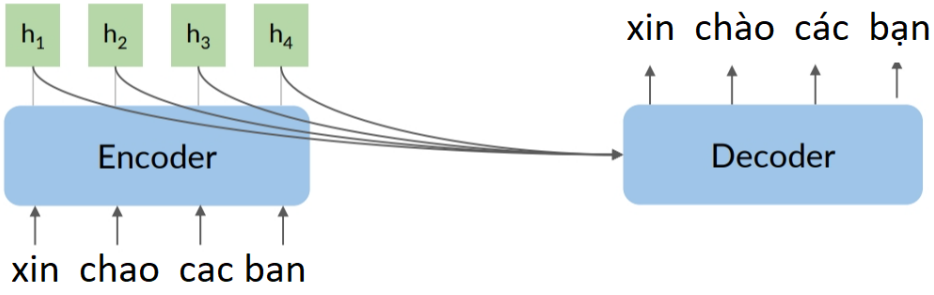
Câu có độ dài thay đổi và bộ nhớ có độ dài cố định khi đó kích thước chuỗi tăng lên, hiệu suất mô hình giảm. cụ thể hơn là về vector ngữ cảnh (context vector) trong mô hình seq2seq. Nó cho thấy rằng vector ngữ cảnh được tạo ra bằng cách sử dụng trạng thái ẩn cuối cùng của bộ mã hóa, tại thời điểm cuối cùng của văn bản nguồn. Vector ngữ cảnh này đóng vai trò quan trọng trong việc giải mã văn bản đích, vì nó là thông tin duy nhất mà bộ giải mã biết về văn bản nguồn.

Tuy nhiên, vector ngữ cảnh có thể trở thành một **bottleneck** trong mô hình seq2seq hình 3.6 , vì nó phải biểu diễn tất cả mọi thứ về ý nghĩa của văn bản nguồn. Điều này có nghĩa là thông tin ở đầu câu, đặc biệt là đối với các câu dài, có thể không được biểu diễn một cách đồng đều trong vector ngữ cảnh. Do đó, việc biểu diễn thông tin trong vector ngữ cảnh có thể không hiệu quả cho các câu dài, và có thể dẫn đến mất mát thông tin hoặc hiệu suất giảm khi giải mã các câu dài.



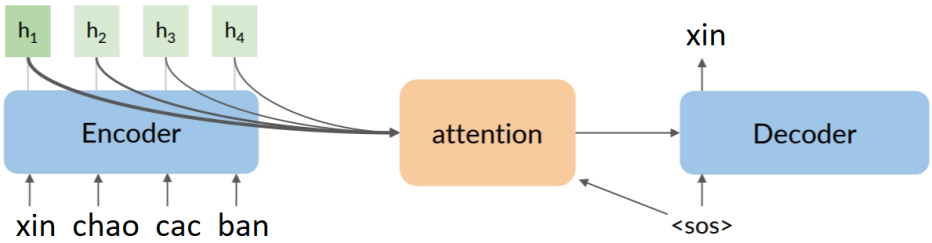
Hình 3.6: Trạng thái ẩn cuối cùng của bộ mã hóa sẽ buộc tất cả thông tin từ toàn bộ câu nguồn đi qua nút cổ chai biểu diễn này, nguồn [3]

Để giải quyết vấn đề này, các phương pháp như Attention Mechanism đã được phát triển để cho phép bộ giải mã truy cập vào các trạng thái ẩn của bộ mã hóa tại các thời điểm khác nhau, thay vì chỉ sử dụng vector ngữ cảnh cuối cùng. thì sử dụng hết tất cả các trạng thái của encoder.



Hình 3.7: Lấy tất cả các trạng thái của encoder

sử dụng attention, tập trung các trạng thái cụ thể ở mỗi bước (timestep).



Hình 3.8: Áp dụng attention

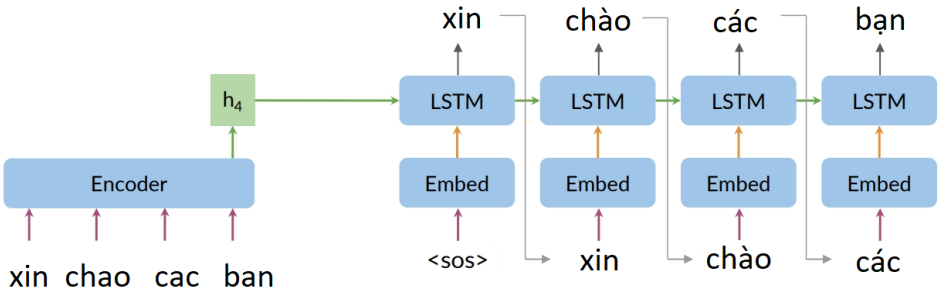
3.2 Giới thiệu Attention

Thuật ngữ "attention"trong xử lý ngôn ngữ tự nhiên (NLP) được giới thiệu đầu tiên trong bài báo nổi tiếng có tựa đề "Neural Machine Translation by Jointly Learning to Align and Translate,"được đăng vào năm 2014 bởi Dzmitry Bahdanau, Kyunghyun Cho và Yoshua Bengio [1]. Trong bài báo này , họ giới thiệu một cơ chế "attention"trong mô hình dịch máy dựa trên mạng nơ-ron học sâu (deep learning). Mô hình này cho phép mô hình tập trung vào các phần khác nhau của câu nguồn khi thực hiện dịch, giúp cải thiện khả năng dịch đa dạng và hiệu suất tổng thể của hệ thống dịch máy.

Attention là một cơ chế quan trọng trong mô hình seq2seq, được sử dụng để giải quyết vấn đề của vector ngữ cảnh trong mô hình. Trong mô hình seq2seq truyền thống, vector ngữ cảnh được tạo ra bằng cách sử dụng trạng thái ẩn cuối cùng của bộ mã hóa, và nó phải biểu diễn tất cả thông tin từ toàn bộ câu nguồn. Điều này có thể dẫn đến việc mất mát thông tin hoặc việc biểu diễn không hiệu quả, đặc biệt là đối với các câu dài hoặc các câu có cấu trúc phức tạp.

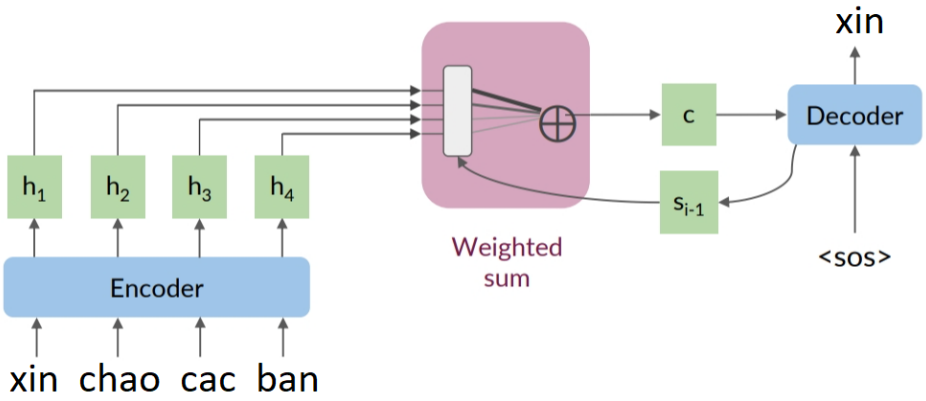
Cơ chế Attention giúp giải quyết vấn đề này bằng cách cho phép bộ giải mã truy cập vào các trạng thái ẩn của bộ mã hóa tại các thời điểm khác nhau, thay vì chỉ sử dụng vector ngữ cảnh cuối cùng. Cụ thể, Attention cho phép bộ giải mã tập trung vào các phần quan trọng của câu nguồn khi giải mã câu đích. Cơ chế Attention tạo ra một trọng số cho mỗi trạng thái ẩn của bộ mã hóa, dựa trên sự tương quan giữa trạng thái ẩn đó và trạng thái ẩn hiện tại của bộ giải mã. Các trọng số này được sử dụng để tính toán một vector ngữ cảnh mới, được tạo ra bằng cách lấy tổng trọng số của các trạng thái ẩn của bộ mã hóa, mỗi trạng thái ẩn được nhân với trọng số tương ứng.

3.2.1 Cách attention làm việc



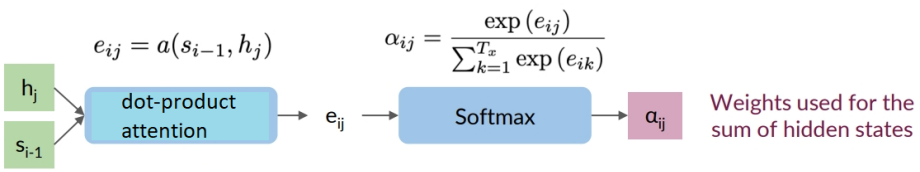
Hình 3.9: Mô hình seq2seq truyền thống

Tính toán một vector trạng thái ẩn cho mỗi từ trong câu nguồn bằng cách sử dụng bộ mã hóa (encoder) RNN hoặc LSTM. Tại mỗi bước giải mã, tính toán một vector trạng thái ẩn cho bộ giải mã (decoder).



Hình 3.10: Sử dụng tất cả các trạng thái của encoder tại bước (timestep) decoder

Tính toán một trọng số cho mỗi vector trạng thái ẩn của encoder, dựa trên sự tương quan giữa vector trạng thái ẩn đó và vector trạng thái ẩn hiện tại của decoder.



Hình 3.11: Công thức attention

$$\text{score}(\mathbf{s}_{i-1}, \mathbf{h}_j) = \mathbf{s}_{i-1} \cdot \mathbf{h}_j$$

Trong đó:

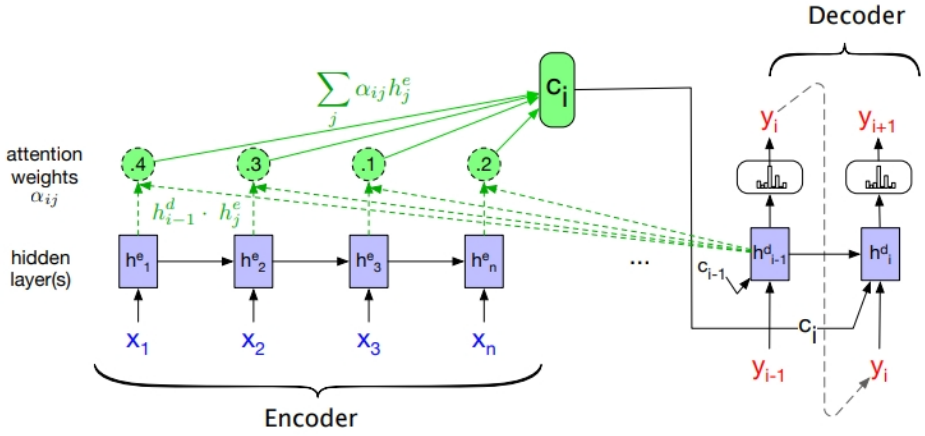
- $\mathbf{s}_{i-1}$: Vector trạng thái ẩn của mô hình ở bước thời gian $i-1$ trong decoder.
- $\mathbf{h}_j$: Vector trạng thái ẩn của mô hình trong encoder tại timestep thứ j

Sử dụng các trọng số này để tính toán một vector ngữ cảnh mới, được tạo ra bằng cách lấy tổng trọng số của các vector trạng thái ẩn của bộ mã hóa, mỗi vector trạng thái ẩn được nhân với trọng số tương ứng.

$$c_i = \sum_j \alpha_{ij} h_j \tag{3.1}$$

- c_i: Context vector của mô hình tại timestep i của decoder.
- α_{ij}: Hệ số trọng số (attention weight) tại timestep thứ i của decoder và timestep thứ j của encoder.
- h_j: Vector trạng thái ẩn của mô hình tại bước thời gian j của encoder

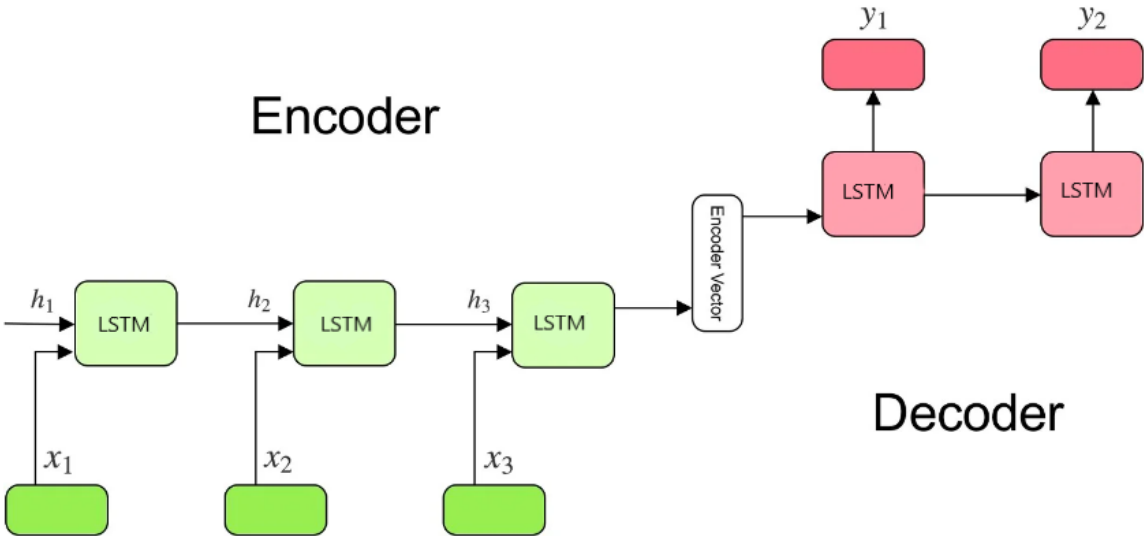
Sử dụng vector ngữ cảnh mới này để tính toán một vector trạng thái ẩn mới cho bộ giải mã. Sử dụng vector trạng thái ẩn mới này để tính toán xác suất của từ tiếp theo trong câu đích. Lặp lại các bước trên cho đến khi kết thúc câu đích.



Hình 3.12: Minh họa quá trình tính toán của cơ chế Attention tại một bước giải mã cụ thể, giúp hiểu rõ hơn về cách thức hoạt động của cơ chế Attention [3]

Chương 4

XÂY DỰNG MÔ HÌNH SEQUENCE TO SEQUENCE



Hình 4.1: Hình ảnh minh họa mô hình seq2seq

4.1 Định nghĩa lớp embeddings

Sử dụng mô hình word embedding: FastText.

Các thông số của mô hình:

- **VOCAB SIZE:** 7099,
- **embeddings_dim:** 300,
- **dropout:** 0.05,
- **max_length:** 10

Minh họa đoạn mã xây dựng lớp Embeddings:

```
# Define Embedding Layer
embedding_layer_inputs = Embedding(VOCAB_SIZE, embeddings_dim,
                                   input_length=maxlen_inputs,
                                   weights=[embedding_matrix],
                                   trainable=False)
# Define Embedding Layer
embedding_layer_outputs=Embedding(VOCAB_SIZE,embeddings_dim
                                   ,input_length=maxlen_outputs
                                   ,weights = [embedding_matrix]
                                   ,trainable=False)
```

4.2 Xây dựng lớp encoder

Minh họa đoạn mã xây dựng lớp Encoder:

```
# Bước 2: Xây dựng Encoder
encoder_inputs = Input(shape = (maxlen_inputs, ))
encoder_embedding = embedding_layer_inputs(encoder_inputs)
encoder_outputs, state_h, state_c = LSTM(300,
                                dropout=0.05,
                                return_state=True)(encoder_embedding)
encoder_states = [state_h, state_c]
```

4.3 Xây dựng lớp decoder

Minh họa đoạn mã xây dựng lớp Decoder:

```
decoder_inputs = Input(shape=(maxlen_inputs, ))
decoder_embedding = embedding_layer_outputs(decoder_inputs)

attention_layer = Attention(use_scale=True)
attention_sequence = attention_layer([decoder_embedding,
                                      encoder_outputs],
                                      return_attention_scores=False)
normalization = keras.layers.LayerNormalization()
normalized_attention_sequence = normalization(attention_sequence)
combined_sequence = keras.layers.Add()([decoder_embedding,
                                      normalized_attention_sequence])

decoder_lstm = LSTM(300, return_state=True,
                return_sequences=True,dropout=0.05)
decoder_outputs , state_hd , state_cd = decoder_lstm(combined_sequence,
                                          initial_state=encoder_states)
# Định nghĩa lớp Dense để kết quả đầu ra
decoder_dense = Dense(VOCAB_SIZE, activation='softmax')
output = decoder_dense(decoder_outputs)
```

4.4 Thông số mô hình Sequence to Sequence đã xây dựng

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 10)]	0	[]
input_2 (InputLayer)	[(None, 10)]	0	[]
embedding (Embedding)	(None, 10, 300)	2129700	['input_1[0][0]']
embedding_1 (Embedding)	(None, 10, 300)	2129700	['input_2[0][0]']
lstm (LSTM)	[(None, 300), (None, 300), (None, 300)]	721200	['embedding[0][0]']

Hình 4.2: Hình ảnh minh họa thông số mô hình seq2seq

attention (Attention)	(None, 10, 300)	1	['embedding_1[0][0]', 'lstm[0][0]']
layer_normalization (Layer Normalization)	(None, 10, 300)	600	['attention[0][0]']
add (Add)	(None, 10, 300)	0	['embedding_1[0][0]', 'layer_normalization[0][0]']
lstm_1 (LSTM)	[(None, 10, 300), (None, 300), (None, 300)]	721200	['add[0][0]', 'lstm[0][1]', 'lstm[0][2]']
dense (Dense)	(None, 10, 7099)	2136799	['lstm_1[0][0]']

Total params: 7839200 (29.90 MB)
Trainable params: 3579800 (13.66 MB)
Non-trainable params: 4259400 (16.25 MB)

Hình 4.3: Hình ảnh minh họa thông số mô hình seq2seq

Chương 5

PHƯƠNG PHÁP ĐÁNH GIÁ MÔ HÌNH, KẾT QUẢ

5.1 Giới thiệu về phương pháp đánh giá

5.1.1 BLEU Score

BLEU là viết tắt của Bilingual Evaluation Understudy, là phương pháp đánh giá một bản dịch dựa trên các bản dịch tham khảo, được giới thiệu trong paper BLEU: a Method for Automatic Evaluation of Machine Translation. BLEU được thiết kế để sử dụng trong dịch máy (Machine Translation).

Cách tính của BLEU là đếm số n-gram khớp nhau giữa câu mẫu (R) và câu được đánh giá (C) sau đó chia cho số token của C. Việc chọn n phụ thuộc vào ngôn ngữ, nhiệm vụ và mục tiêu cụ thể.

Công thức cụ thể như sau [5]:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{c}{r}\right) & \text{if } c \leq r \end{cases}$$

Trong đó:

BP là một trọng số dài hạn (brevity penalty), giả định rằng bản dịch ngắn hơn so với bản dịch tham chiếu sẽ có chất lượng thấp hơn. BP đảm bảo rằng các hệ số BLEU của các bản dịch được đánh giá dựa trên độ dài so sánh với bản dịch tham chiếu.

c là chiều dài của bản dịch được đánh giá.

r là chiều dài của bản dịch tham chiếu (reference translation) ngắn nhất.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log p_n\right)$$

Trong đó:

N là số lượng từ trong bản dịch.

p_n là precision cho các n-grams, được tính bằng cách đếm số lần xuất hiện của các n-grams trong bản dịch và chia cho tổng số lần xuất hiện của n-grams trong các bản dịch tham chiếu.

Trong bài báo cáo này, các kết quả thu được sẽ được đánh giá dựa trên BLEU score.

5.1.2 TER

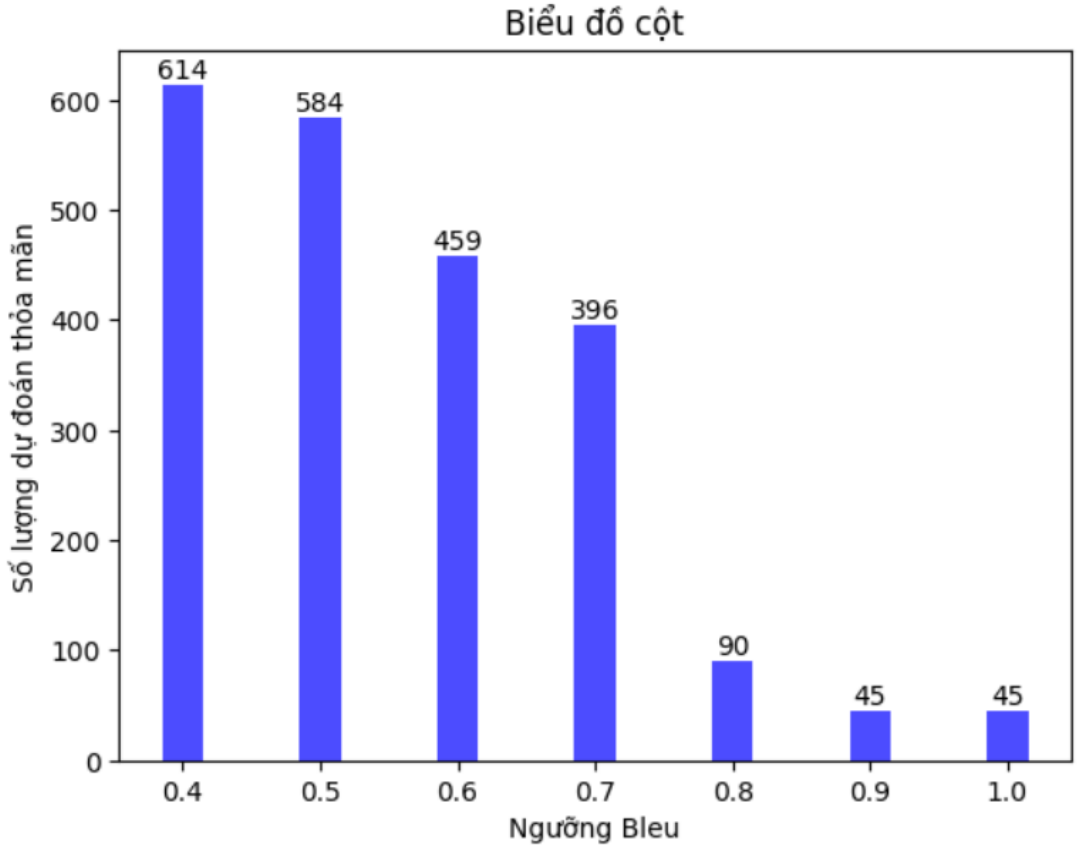
Ngoài BLEU score thường được sử dụng để đánh giá độ chính xác của mô hình dịch máy, TER là độ đo cũng được sử dụng với mục đích tương tự.

TER là viết tắt của Translation Edit Rate, là đại lượng đo lường mức độ chỉnh sửa mà con người phải thực hiện để thay đổi đầu ra của mô hình sao cho nó khớp chính xác với bản dịch thực tế. Công thức [2]:

$$TER = \frac{\text{Number of edits}}{\text{Average number of reference words}}$$

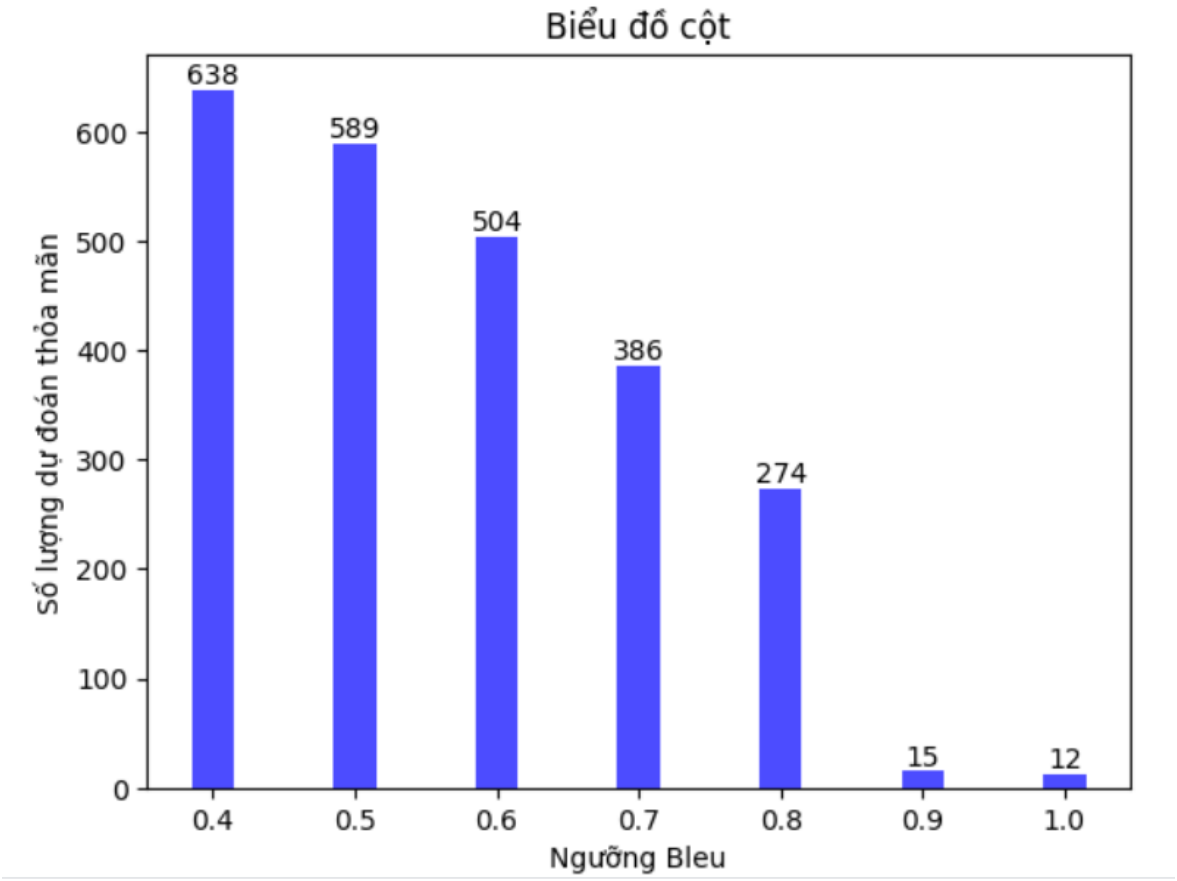
5.2 Kết quả xây dựng mô hình

5.2.1 Mô hình không áp dụng Attention



Hình 5.1: Hình ảnh minh họa kết số lượng câu thỏa mãn ngưỡng Bleu Score khi mô hình không áp dụng attention

5.2.2 Mô hình áp dụng Attention



Hình 5.2: Hình ảnh minh họa kết số lượng câu thỏa mãn ngưỡng Bleu Score khi mô hình áp dụng attention

Nhận xét:

Dựa vào Hình 5.1 và Hình 5.2, kết quả cho thấy rằng mô hình áp dụng cơ chế Attention cho kết quả tốt hơn mô hình khi không áp dụng cơ chế Attention.

Cụ thể, tại giá trị ngưỡng Bleu Score từ 0.4 đến 0.8, mô hình áp dụng cơ chế Attentention có số lượng từ dự đoán lớn hơn số lượng từ dự đoán đúng của mô hình không áp dụng cơ chế Attention.

Tuy nhiên với ngưỡng Bleu Scrore cao 0.9 và 1, mô hình không áp dụng cơ chế Attention lại cho kết quả tốt hơn mô hình áp dụng cơ chế Attention

Chương 6

KẾT LUẬN

Từ quá trình xây dựng mô hình seq2seq, chúng tôi nhận thấy rằng các phương pháp này đã mang lại những kết quả chính xác hơn trong việc thêm dấu tiếng Việt so với các mô hình truyền thống.

Qua quá trình nghiên cứu, chúng tôi đã tiếp cận, nắm được các nguyên lý cơ bản của mô hình Seq2Seq và cơ chế Attention, cũng như cách chúng hoạt động và ứng dụng của chúng.

Tuy kết quả thu được là tích cực, nhưng mô hình vẫn còn nhiều hạn chế: độ chính xác của các câu dịch chưa cao, chưa thể áp dụng vào thực tiễn. Mô hình chỉ xây dựng dựa trên độ dài tối đa của một câu là 10 từ, trong tương lai dự định chúng tôi sẽ phát triển mô hình để có thể dịch được các câu có độ dài lớn hơn.

Mô hình Transformer có thể cho độ chính xác của các câu dịch máy cao hơn so với các mô hình truyền thống nên trong tương lai, chúng tôi dự định sẽ xây dựng mô hình Transformer để đối chiếu kết quả với mô hình seq2seq đã xây dựng.

Tài liệu tham khảo

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. Published on 2014/09/01.

[2] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of bleu in machine translation research. *Journal of Machine Translation*, 2022. In press.

[3] Dan Jurafsky and James H. Martin. Speech and language processing (3rd ed.). `https://web.stanford.edu/~jurafsky/slp3/9.pdf`, 2022. Chapter 9: Finite State Transducers.

[4] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *arXiv preprint arXiv:1808.03314*, page 43, 2018. Published on 2018/08/09.

[5] Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. `https://aclanthology.org/2006.amta-papers.25/`, 2006.

[6] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, page 9, 2014. Published on 2014/09/10.