BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC BÁCH KHOA HÀ NỘI

~~~~~~*~~~~~~

BÁO CÁO BÀI TẬP LỚN
MÔ HÌNH PHIÊN DỊCH NGÔN NGỮ

Thành viên nhóm 8	:	**Dương Văn Giới<20215041>**
		Nguyễn Thành Đạt<20215028>
		Nguyễn Gia Tùng Dương<20215023>
		Phan Trung Đức<20215038>
		Trần Huy Hoàng<20210386>
Lớp	:	**147728 - Nhập môn AI**
Giáo viên hướng dẫn	:	**TS. Trần Thế Hùng**
~~~~~~

PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

Họ và tên	MSSV	Nhiệm vụ
Dương Văn Giới	20215041	Nghiên cứu và lựa chọn mô hình Thực hiện train mô hình Cài đặt, viết báo cáo
Nguyễn Gia Tùng Dương	20215023	Xây dựng Attention + MultiHeadAttention Thực hiện train mô hình Cài đặt, viết báo cáo
Nguyễn Thành Đạt	20215028	Xây dựng Embedding + Positional Encoding Cài đặt, viết báo cáo
Phan Trung Đức	20215041	Xây dựng Optimizer + Label Smoothing Cài đặt, viết báo cáo
Trần Huy Hoàng	20210386	Thu thập data + Data preprocessing Cài đặt, viết báo cáo

NHẬN XÉT CỦA GIẢNG VIÊN:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Mục lục

1. Tổng quan về dự án

1.1. Dự án

Trong thời đại toàn cầu hóa và kết nối hiện nay, giao tiếp và hiểu biết giữa các ngôn ngữ khác nhau trở nên vô cùng quan trọng. Dịch ngôn ngữ không chỉ giúp xóa bỏ rào cản ngôn ngữ mà còn tạo điều kiện thuận lợi cho học tập, làm việc và giao lưu văn hóa. Nhận thấy nhu cầu ngày càng cao về dịch thuật chính xác và nhanh chóng, chúng tôi quyết định phát triển một ứng dụng dịch ngôn ngữ dựa trên trí tuệ nhân tạo (AI) để dịch từ tiếng Anh sang tiếng Việt. Ứng dụng này sẽ mở ra cơ hội mới, thúc đẩy sự kết nối và hiểu biết giữa các cộng đồng ngôn ngữ khác nhau, đồng thời hỗ trợ mạnh mẽ cho các hoạt động giáo dục và kinh doanh trong môi trường quốc tế.

1.2. Mục tiêu

Dự án của chúng tôi hướng tới việc phát triển một công cụ dịch thuật thông minh, cung cấp bản dịch chất lượng cao và dễ sử dụng cho mọi đối tượng. Với đầu vào là một câu tiếng Anh, hệ thống sẽ tự động tạo ra bản dịch tiếng Việt với độ chính xác tối ưu.

1.3. Công nghệ được áp dụng

Ban đầu dự án hướng tới việc xây dựng mô hình seq2seq để dịch từ tiếng Anh sang tiếng Việt. Tuy nhiên, do những hạn chế cố hữu của mô hình seq2seq, chúng tôi đã chuyển hướng sang sử dụng mô hình Transformer, một kiến trúc tiên tiến hơn đã chứng minh được hiệu quả vượt trội trong các tác vụ xử lý ngôn ngữ tự nhiên, đặc biệt là trong lĩnh vực dịch máy.

2. Phương pháp: mô hình Transformer

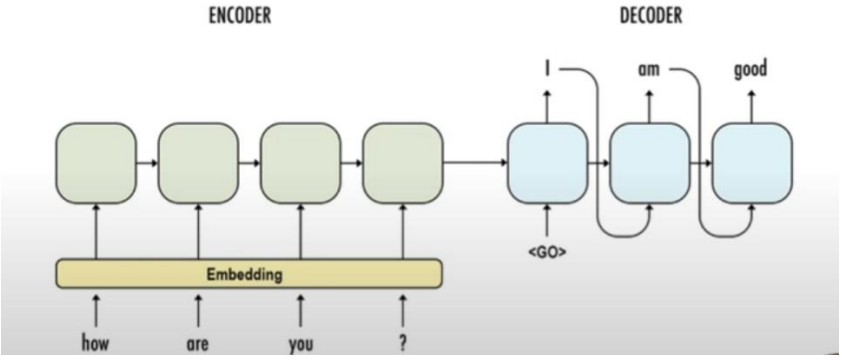
2.1. Tổng Quan Mô Hình

2.1.1. Giới thiệu

Mô hình Transformer là mô hình nổi tiếng, nó là cơ sở của những mô hình phổ biến hiện nay như GPT (như ChatGPT của OpenAI, được áp dụng Transformer để tạo ra văn bản tự nhiên và tự động); BERT (một trong những mô hình dịch ngôn ngữ và hiểu ngôn ngữ tự nhiên phát triển bởi Google, để tạo ra công cụ tìm kiếm của họ). Ý tưởng chủ đạo của Transformer vẫn là áp dụng cơ chế Attention nhưng ở mức phức tạp hơn. Vậy vì sao có thể nói "Attention is all you need" mà không phải là Recurrent Neural Network (RNN) hay Long Short Term Memory (LSTM)?

2.1.1.1. Cơ chế Attention

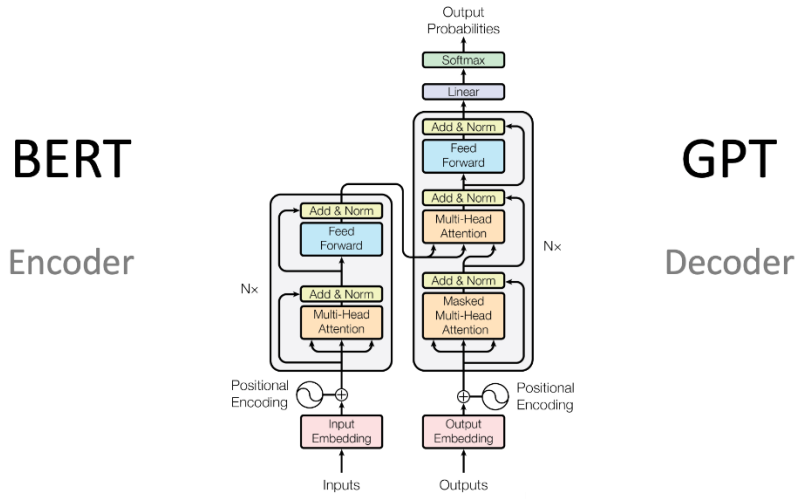
- Quá trình hình thành cơ chế Attention có thể được minh họa bằng một ví dụ trực quan như sau: giả sử chúng ta được giao nhiệm vụ dịch một đoạn văn bản bằng ngôn ngữ không quen thuộc, nhưng may mắn thay ta có sự hỗ trợ của một cuốn từ điển. Nếu áp dụng phương pháp:
  - LSTM/RNN: Chúng ta sẽ đọc tuần tự từng từ, tra cứu những từ chưa biết, và lặp lại quá trình này cho đến hết đoạn văn. Điều này tiềm ẩn nguy cơ phải tra cứu toàn bộ các từ trong văn bản, gây tốn kém thời gian và công sức.



  - Attention: Chúng ta sẽ nhanh chóng đọc lướt qua toàn bộ đoạn văn, tập trung vào những từ có vẻ mang tính then chốt. Khi gặp từ mới, ta chỉ cần tra cứu ý nghĩa của từ đó mà không cần quan tâm đến những từ khác. Nhờ đó, có thể nắm bắt được nội dung chính của câu chứa từ đó mà không cần phải tra cứu toàn bộ văn bản.

⇨ Rõ ràng, Attention thể hiện sự vượt trội về hiệu quả so với các phương pháp trước đó. Tuy nhiên, để xác định được đâu là những từ mang tính

trọng yếu, mô hình cần phải trải qua quá trình huấn luyện và học hỏi. Cụ thể như sau:



- Trong Attention, mỗi từ hoặc Token được biểu diễn dưới dạng một Vector thông qua Input Enbedding, các vector lại được tương tác với Positional Encoding để đính vào một vector vị trí của nó so với các vector còn lại (vì trong Attention, các từ được đưa vào mạng đồng thời, không tuần tự như RNN, LSTM). Cơ chế Self-Attention cho phép mô hình tính toán trọng số cho mỗi từ dựa trên mối quan hệ của nó với tất cả các từ khác trong câu (mối quan hệ giữa các từ được tính toán bằng cách đưa các từ vào một bộ Self-Attention). Quá trình này giúp mô hình hiểu được ngữ cảnh của từ đó trong câu.
- Cụ thể trong Self-Attention, để tính toán trọng số attention cho mỗi từ, cần thực hiện một số phép tính toán. Trong quá trình này, mỗi từ được so sánh với tất cả các từ khác trong câu thông qua các phép nhân ma trận và hàm tương tự. Kết quả là một vector trọng số attention cho mỗi từ, thể hiện mức độ quan trọng của từ đó đối với từ khác trong câu.
- Sau khi tính toán trọng số attention cho mỗi từ, mô hình sử dụng chúng để tính toán tổ hợp có trọng số của các vector biểu diễn từ (hay còn gọi là "context vector"). Quá trình này cho phép mô hình tập trung vào các từ quan trọng và tổ chức lại thông tin theo cách hữu ích cho các nhiệm vụ như dịch ngôn ngữ, tóm tắt văn bản, và nhiều hơn nữa.

2.1.1.2. Hạn chế của RNN/LSTM

- Khả năng xử lí dài hạn kém do gặp vấn đề biến mất Gradient (Gradient là đạo hàm có thể đo lường sự sai lệch giữa đầu ra phán đoán và đầu ra thực). Ví dụ, với câu “Cô ấy không chỉ là một người xinh đẹp mà còn là một người thông minh và tài năng”, mô hình RNN/LSTM có thể gặp

khó khăn trong việc ghi nhớ thông tin 'xinh đẹp' ở đầu câu khi dự đoán các từ ở cuối câu, do đó có thể dịch sai thành “Cô ấy không chỉ là một người thông minh và tài năng mà còn là một người thông minh và tài năng”.

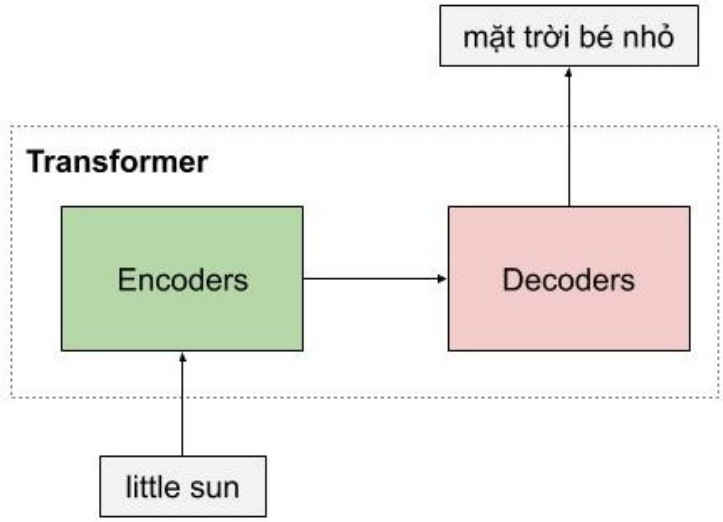
- Vì tính toán theo thứ tự tuần tự từ bước đầu tiên đến bước cuối cùng nên hiệu suất tính toán khá thấp khi xử lí chuỗi dài, và dẫn đến quá trình học và phán đoán của mô hình chậm chạp.

2.2. Giải thích Mô Hình

2.2.1. Phân Tích Kiến Trúc Mô Hình Transformer

Để dễ đàng hiểu cách mô hình hoạt động, nhóm sẽ trình bày kiến trúc mô hình ở mức cao trước, sau đó sẽ đi vào chi tiết từng phần và công thức toán học. Tương tự với các mô hình dịch máy khác, mô hình Transformer có kiến trúc tổng quan gồm 2 phần chính: Encoder và Decoder. Trong đó, Encoder có nhiệm vụ học vector biểu diễn của của câu với mục tiêu vector này chứa đầy đủ thông tin của câu đó. Còn Decoder có nhiệm vụ chuyển vector biểu diễn này thành ngôn ngữ đích.

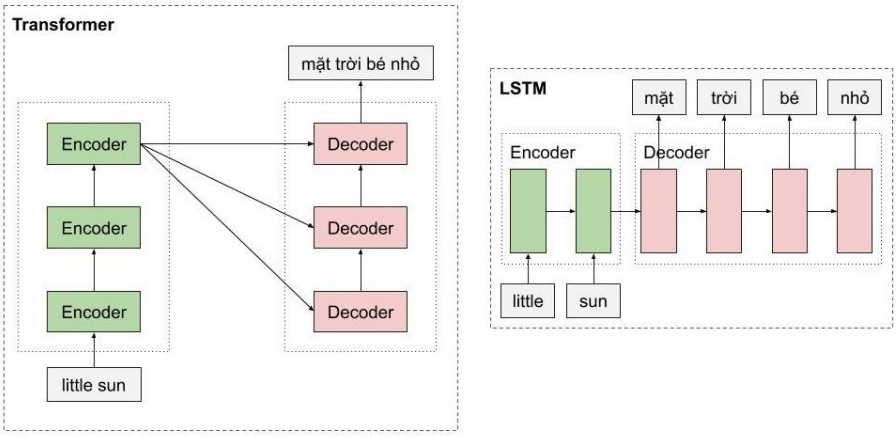
Ví dụ:



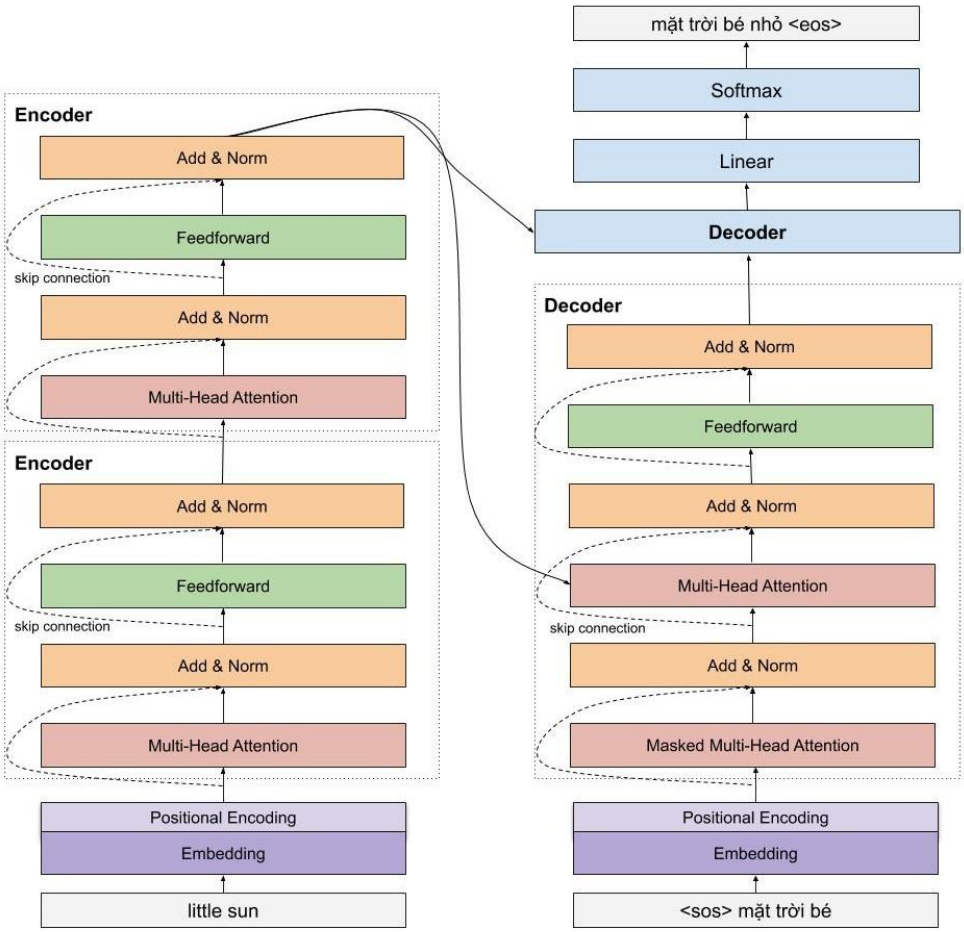
Trong ví dụ trên, Encoder của mô hình Transformer nhận một câu tiếng Anh và mã hóa thành một vector biểu diễn ngữ nghĩa của câu “little sun”. Sau đó, Decoder nhận vector biểu diễn này và dịch nó thành câu tiếng Việt “mặt trời bé nhỏ”.

Ưu điểm của mô hình Transformer là khả năng xử lý song song cho các từ. Encoder của mô hình này là là một dạng của feedforward neurals nets, gồm các

encoder layer khác. Mỗi encoder layer sẽ xử lý đồng thời các từ. Trong lúc đó, đối với mô hình LSTM thì các từ phải được xử lý một cách tuần tự. Bên cạnh đó, mô hình Transformer còn xử lý câu đầu vào theo 2 hướng mà không cần stack thêm một mô hình LSTM nữa như trong kiến trúc Bidirectional LSTM



Tiếp theo nhóm sẽ trình bày chi tiết các nội dung quan trọng như: sinusoidal position encoding, multi head attention của encoder, còn của decoder thì kiến trúc tương tự với của encoder.



2.2.2. Tokenization

Tokenization là quá trình chia nhỏ một đoạn văn bản thành các đơn vị nhỏ hơn gọi là token. Các token này có thể là từ, cụm từ, ký tự hoặc các đơn vị phụ thuộc vào phương pháp tokenization được sử dụng.

Mục đích của Tokenization:

- Chuẩn hóa đầu vào: Tokenization giúp chuyển đổi văn bản thô thành dạng mà mô hình máy tính có thể hiểu và xử lý được.
- Giảm kích thước từ vựng: Bằng cách chia văn bản thành các token, chúng ta có thể giảm số lượng từ vựng duy nhất mà mô hình cần phải học, giúp quá trình huấn luyện hiệu quả hơn.
- Xử lý các từ chưa biết: Tokenization có thể giúp xử lý các từ chưa biết (out-of-vocabulary words) bằng cách chia chúng thành các đơn vị nhỏ hơn đã biết.

Trong project này, chúng tôi lựa chọn word tokenize sử dụng thư viện spaCy

2.2.3. Embedding Layer với Position Encoding

Trước khi đi sâu vào mô hình encoder, nhóm sẽ trình bày về cơ chế Position Encoding. Đây là cơ chế này giúp mô hình transformer nhận biết vị trí của các từ trong câu.

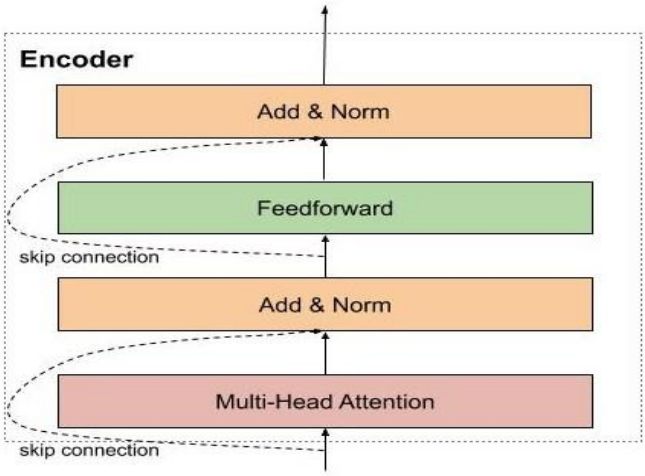
Đầu tiên, các từ sẽ được biểu diễn bằng một vector thông qua ma trận word embedding có số dòng bằng với kích thước của tập từ vựng. Các từ trong câu được tra cứu trong ma trận này và xếp thành các dòng của một ma trận 2 chiều chứa ngữ nghĩa của từng từ. Tuy nhiên, do transformer xử lý các từ song song nên việc chỉ sử dụng word embedding không thể giúp mô hình biết được vị trí của các từ trong câu. Vì vậy, cần một cơ chế để thêm thông tin vị trí các từ vào vector đầu vào. Lúc này, positional encoding được sử dụng để giải quyết vấn đề này.

Positional encoding trong mô hình transformer được thiết kế sử dụng các hàm sin và cosin có chu kỳ khác nhau để biểu diễn vị trí các từ. Cách làm này giúp mô hình không chỉ biết vị trí tương đối của các từ mà còn duy trì được tính chất tuần hoàn, giúp mô hình dễ dàng dự đoán các câu có độ dài vượt quá tập huấn luyện. Việc kết hợp thông tin ngữ nghĩa từ word embedding và thông tin vị trí từ positional encoding giúp mô hình transformer hiểu rõ hơn về cấu trúc và ý nghĩa của câu, đồng thời cải thiện hiệu suất dịch máy và các tác vụ xử lý ngôn ngữ tự nhiên khác.

2.2.4. Encoder

Cấu trúc của một bộ mã hóa trong mô hình Transformer là một phần quan trọng, với nhiều encoder layer tương tự nhau. Mỗi encoder layer bao gồm hai thành phần chính là Multi Head Attention và mạng feedforward. Ngoài ra, còn có skip connection và normalization layer.

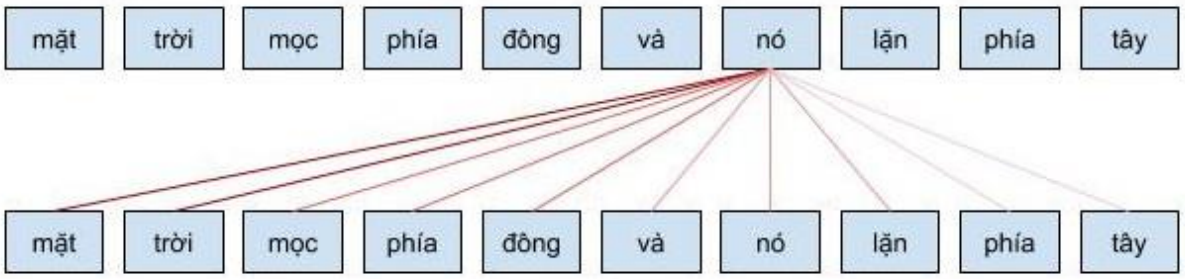
Trong hai thành phần chính này, Multi Head Attention là một phần mới được giới thiệu trong mô hình Transformer, tạo ra sự khác biệt giữa nó và các mô hình khác như LSTM.



Encoder đầu tiên nhận đầu vào là ma trận biểu diễn của các từ kết hợp với thông tin vị trí qua việc mã hóa vị trí. Sau đó, ma trận này được chuyển qua Multi Head Attention. Cơ chế Multi Head Attention có bản chất là self-attention, nhưng để mô hình có thể chú ý đến nhiều mẫu khác nhau, chúng ta sử dụng nhiều self-attention.

2.2.5. Self Attention Layer

Trong quá trình self-attention, một từ được mã hóa có thể sử dụng thông tin từ các từ liên quan đến nó. Điều này có ý nghĩa tương tự như cơ chế attention . Cơ chế này có thể được xem như một quá trình tìm kiếm, cho phép mô hình tìm kiếm các từ tương tự để mã hóa thông tin dựa trên tất cả các từ.

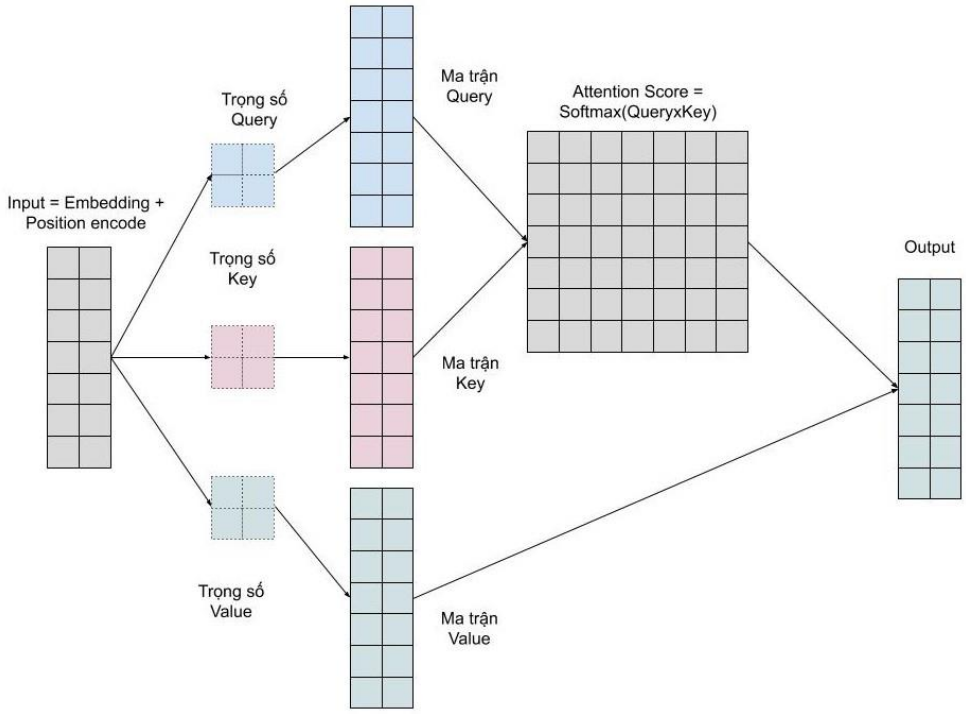


Đầu tiên, với môi từ chúng ta cần tạo ra 3 vector: query, key, value vector bằng cách nhân ma trận biểu diễn các từ đầu vào với ma trận học tương ứng.

- query vector là vector dùng để chứa thông tin của từ được tìm kiếm, so sánh.
- key vector là vector dùng để biểu diễn thông tin các từ được so sánh với từ cần tìm kiếm ở trên.
- value vector là vector biểu diễn nội dung, ý nghĩa của các từ. Để tính tương quan, chúng 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.

Cụ thể, quá trình tính toán attention vector gồm 3 bước như sau:

- Đầu tiên, tính ma trận query, key, value bằng cách khởi tạo 3 ma trận trọng số query, key, vector. Sau đó nhân input với các ma trận trọng số này để tạo thành 3 ma trận tương ứng.
- Tiếp theo, ta tính trọng số attention bằng cách nhân hai ma trận key và query đã tính ở bước trước để so sánh giữa các từ và chuẩn hóa về đoạn [0-1] bằng hàm softmax
- Cuối cùng, ta tính output bằng cách nhân trọng số attention với ma trận value, biểu diễn thông tin của mỗi từ bằng trung bình có trọng số của các giá trị.

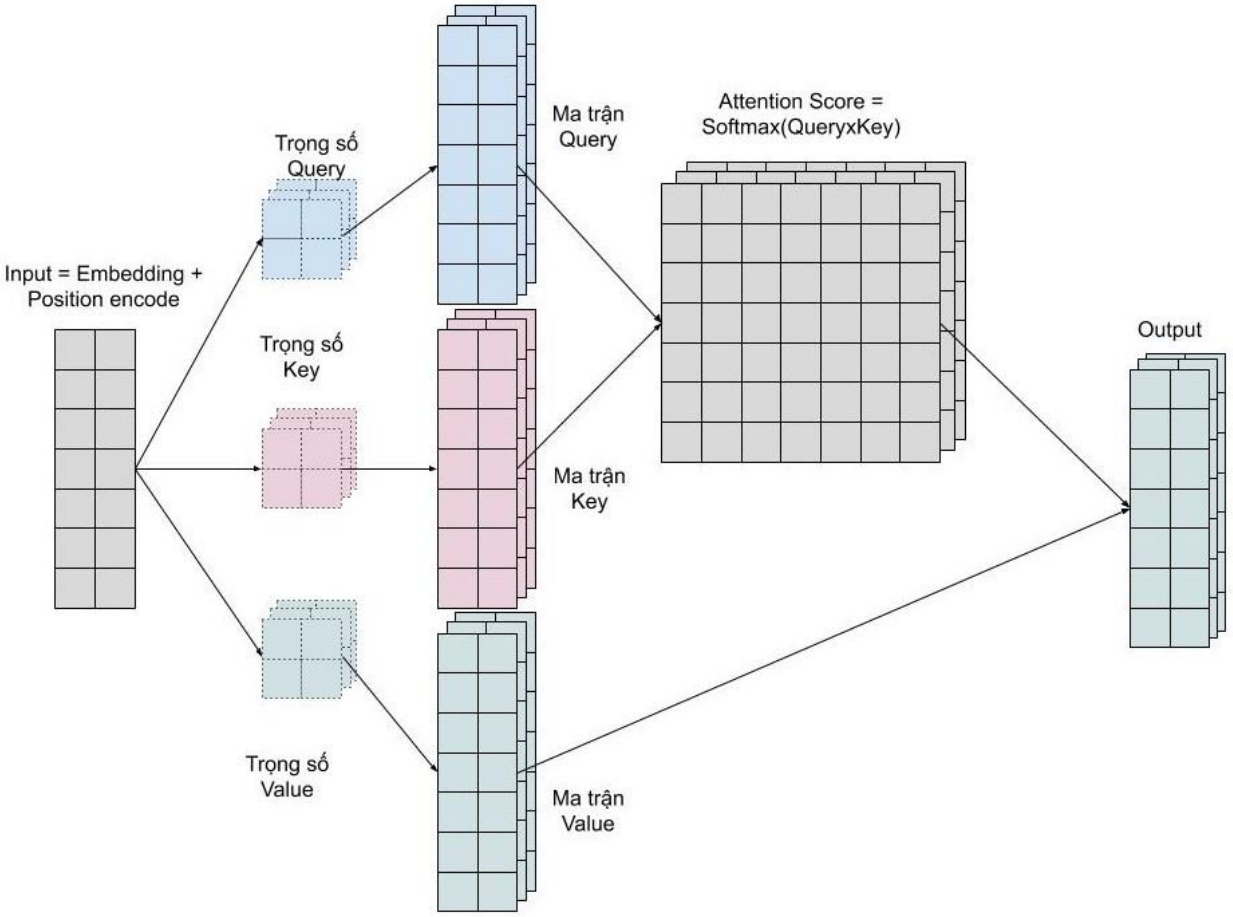


2.2.6. Multi Head Attention

Chúng ta mong muốn mô hình có thể học được nhiều kiểu mối quan hệ giữa các từ với nhau. Với mỗi self-attention, chúng ta học được một kiểu mẫu, do đó để mở rộng khả năng này, chúng ta chỉ cần thêm nhiều self-attention hơn. Tức là chúng ta cần nhiều ma trận query, key, và value hơn. Giờ đây, ma trận trọng số key, query, value sẽ có thêm một chiều depth nữa.

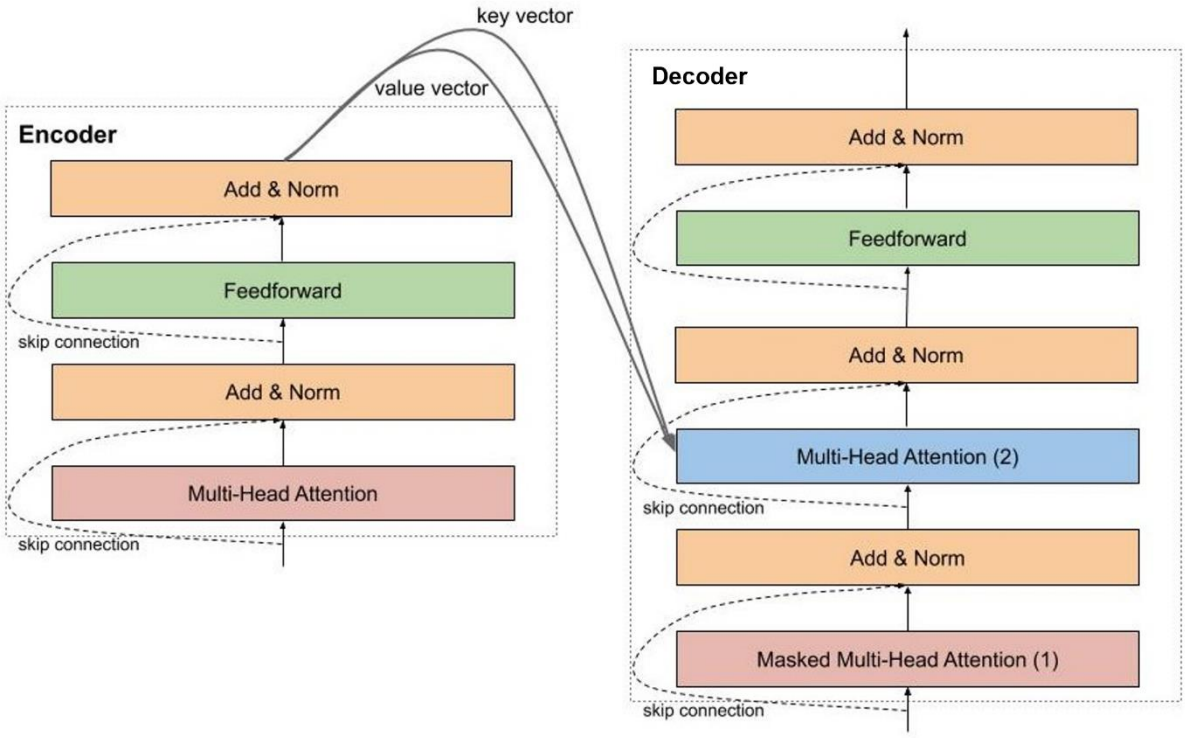
Multi head attention cho phép mô hình đồng thời chú ý đến các pattern dễ quan sát như sau:

- Chú ý đến từ đứng trước một từ
- Chú ý đến từ đứng sau một từ
- Chú ý đến những từ liên quan đến một từ



2.2.7. Decoder

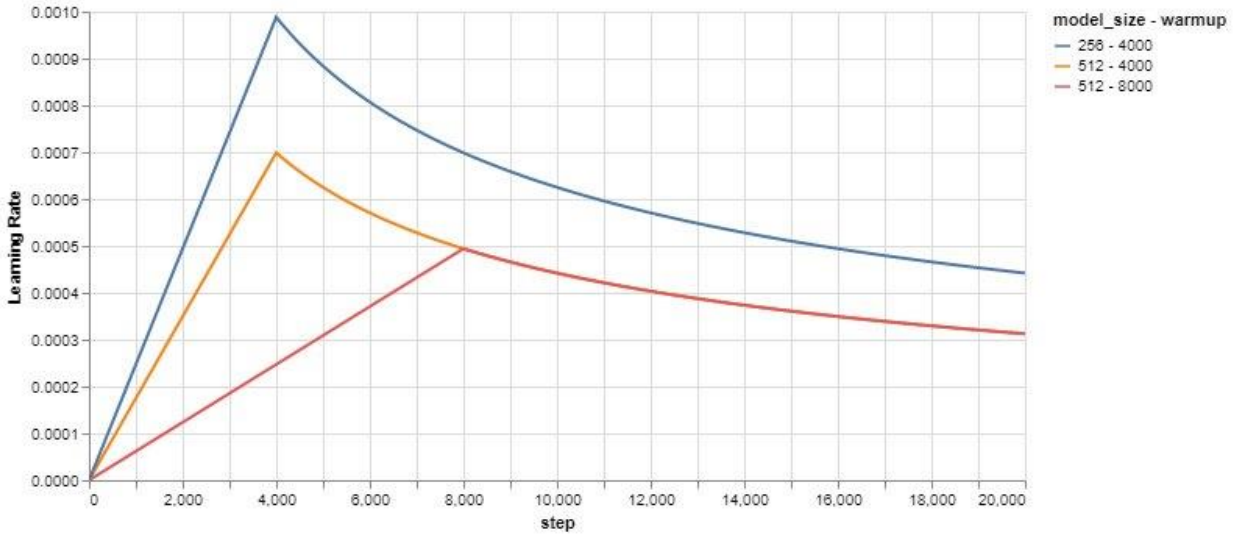
Decoder có nhiệm vụ chuyển đổi vector của câu nguồn thành câu đích, vì vậy nó sẽ nhận thông tin từ encoder dưới dạng hai vector key và value. Cấu trúc của decoder gần giống với encoder, nhưng có thêm một multi head attention ở giữa để học mối quan hệ giữa từ đang được dịch với các từ trong câu nguồn.



2.2.8. Xử lý Overfitting:

Với mô hình nhiều đầu vào thì việc xảy ra Overfitting sẽ thường xuyên hơn, và để hạn chế hiện tượng đó, dự án này được sử dụng 2 phương pháp đó là Optimizer và Label Smoothing.

2.2.8.1. Optimizer:



Optimizer được áp dụng thuật toán Adam (Adaptive Moment Estimation) với tỷ lệ học thay đổi theo số step trong quá trình huấn luyện mô hình Transformer.

Hiểu biết qua một chút về thuật toán Adam: việc tìm điểm cực đại hay cực tiêu của một hàm số được thông qua việc đạo hàm nó bằng 0 tại điểm đó, nhưng không phải hàm số nào ta cũng có thể đạo hàm được, và trong tính toán khoa học, có một số phương thức tìm điểm cực thông qua một số bước lặp với độ dài bước được tự chọn sao cho phù hợp như: phương pháp Fibonacci, phương pháp Newton hay phương pháp Lát cắt vàng,… Đại khái thì thuật toán sẽ chọn một điểm M bất kì trên hàm số, chọn một khoảng x bắt kì trên trục hoàng, chia khoảng x đó ra thành các khoảng vừa đủ dx, và ta được một số điểm x1, x2, …. xn trong khoảng x đó. Sau đó sẽ lặp một hàm số để điểm M cứ sau mỗi bước lắp, M sẽ tiến lại gần điểm cực hơn. Và khi đủ một điều kiện nào đó (ví dụ như không thể tìm được điểm nào có giá trị tung lớn hoặc nhỏ hơn giá trị tung của M thì khi đó M là điểm cực hàm số). Thế nhưng những phương pháp này có nhược điểm là sẽ chỉ tìm được điểm cực cục bộ của hàm trong khoảng x đã chọn, tức là trong khoảng x, có thể điểm M đó là điểm cực nhưng ngoài khoảng x có thể có điểm khác mới thực sự là điểm cực của hàm số xét trên toàn bộ miền giá trị hàm số. Và thuật toán Adam khắc phục được điều này.

Quay trở lại phương pháp Optimizer:

- Giai đoạn khởi động (warm-up): Tỷ lệ học tập tăng tuyến tính từ một giá trị rất gần 0, trong mô hình của chúng em thì bắt đầu từ 0,0001 đến một giá trị cực đại trong một số bước nhất định (warm_up_step). Điều này giúp mô hình ổn định hơn trong giai đoạn đầu.
- Giai đoạn giảm dần: Sau giai đoạn khởi động, tỷ lệ học tập giảm dần theo một hàm số mũ, giúp mô hình hội tụ tốt hơn về cuối quá trình huấn luyện. Khi có thể hội tụ tốt hơn thì việc học của mô hình sẽ tiết kiệm thời gian và tài nguyên. Nó cũng góp phần giảm Overfitting tạo vì khi một mô hình hội tụ về điểm mong đợi nhanh hơn, thì việc xào đi xào lại một dữ liệu đào tạo sẽ ít đi, dẫn đến mô hình không bị ám ảnh bởi dữ liệu đó, linh hoạt hơn trong dự đoán một dữ liệu thử.
- Mô hình có kích thước lớn hơn (d_model = 512) thường yêu cầu tỷ lệ học tập khởi đầu cao hơn so với mô hình nhỏ hơn (d_model = 256).

2.2.8.2. Label smoothing:

Trong các bài toán phân loại, mô hình thường được huấn luyện để dự đoán xác suất thuộc về từng lớp. Để làm điều này, nhãn của dữ liệu thường được mã hóa

một-hot (one-hot encoding), nghĩa là vector nhãn có giá trị 1 ở vị trí của lớp đúng và 0 ở tất cả các vị trí khác.

Ví dụ, với ba lớp, nhãn của một mẫu có thể là:

- [1, 0, 0] cho lớp 1
- [0, 1, 0] cho lớp 2
- [0, 0, 1] cho lớp 3

Label smoothing thay đổi cách mã hóa này bằng cách không sử dụng giá trị 1 và 0 tuyệt đối, mà thay vào đó sử dụng các giá trị hơi dịch chuyển. Ví dụ, với tham số smoothing là ϵ, nhãn có thể được thay đổi thành:

- $[1 - \epsilon + \epsilon /N, \epsilon /N, \epsilon /N]$ cho lớp 1
- $[\epsilon/N, 1 - \epsilon + \epsilon/N, \epsilon/N]$ cho lớp 2
- $[\epsilon/N, \epsilon/N, 1 - \epsilon + \epsilon/N]$ cho lớp 3

Trong đó, N là số lượng lớp và ϵ là tham số smoothing. Giá trị ϵ thường nhỏ, ví dụ 0.1.

Việc sử dụng Label smoothing sẽ làm:

- Giảm Overconfidence (Quá tự tin): Mô hình thường có xu hướng trở nên quá tự tin vào dự đoán của mình, đặc biệt khi nhãn được mã hóa một-hot. Label smoothing giúp giảm bớt sự tự tin này, làm cho mô hình trở nên "khiêm tốn" hơn.
- Cải thiện Khả năng Tổng quát hóa: Bằng cách không sử dụng các giá trị 1 và 0 tuyệt đối, mô hình không học quá mức vào các chi tiết cụ thể của dữ liệu huấn luyện, giúp cải thiện khả năng áp dụng mô hình lên dữ liệu mới.
- Giảm Overfitting: Bằng cách làm mờ nhãn, mô hình được khuyến khích học các đặc trưng chung của dữ liệu thay vì các nhiễu cụ thể, do đó giúp giảm overfitting.

3. Thu thập và tiền xử lý dữ liệu

Để đạt được hiệu suất dịch thuật gần với trình độ con người, các mô hình MT hiện đại cần được đào tạo trên các tập dữ liệu song song chất lượng cao và quy mô lớn.

Tuy nhiên, tiếng Việt, mặc dù là một trong những ngôn ngữ được sử dụng rộng rãi nhất trên thế giới, lại được coi là một ngôn ngữ tài nguyên thấp trong nghiên cứu

MT. Điều này là do thiếu các tập dữ liệu song song công khai có sẵn với số lượng đủ lớn hoặc chất lượng cao. Các tập dữ liệu hiện có thường không có sẵn công khai, quy mô nhỏ hoặc chứa các cặp dịch chất lượng thấp, bao gồm cả các cặp dịch có nghĩa khác nhau (tức là có sự sai lệch).

Để xây dựng hệ thống dịch máy chất lượng cao, chúng tôi sử dụng tập dữ liệu song song PhoMT được cung cấp bởi VinAI Research. Đây là một tập dữ liệu lớn và chất lượng cao cho cặp ngôn ngữ Việt-Anh, bao gồm 3,015,869 cặp câu song ngữ được thu thập từ nhiều nguồn khác nhau như tin tức, blog, bài nói chuyện, wiki, hướng dẫn và phụ đề phim.

Bảng dưới đây thể hiện số liệu thống kê chi tiết của tập dữ liệu PhoMT:

Domain	Total		Training			Validation			Test		
	#doc	#pair	#pair	#en/s	#vi/s	#pair	#en/s	#vi/s	#pair	#en/s	#vi/s
News	2559	41504	40990	24.4	32.0	257	22.3	30.3	257	26.8	34.5
Blogspot	1071	93956	92545	25.0	34.6	597	26.4	37.8	814	23.7	31.5
TED-Talks	3123	320802	316808	19.8	23.8	1994	20.0	24.6	2000	22.0	27.9
MediaWiki	38969	496799	490505	26.0	32.8	3024	25.3	32.3	3270	27.0	33.7
WikiHow	6616	513837	507379	18.9	22.4	3212	17.9	21.5	3246	17.5	21.5
OpenSub	3312	1548971	1529772	9.7	11.1	9635	9.5	10.7	9564	10.0	11.4
All	55650	3015869	2977999	15.7	19.0	18719	15.3	18.7	19151	16.2	19.8

Tập dữ liệu được chia thành ba tập con: tập huấn luyện (training), tập xác thực (validation) và tập kiểm tra (testing). Tập huấn luyện được sử dụng để huấn luyện mô hình dịch máy, tập phát triển được sử dụng để tinh chỉnh mô hình và tập kiểm tra được sử dụng để đánh giá hiệu suất của mô hình. Việc chia tập dữ liệu theo tỷ lệ đảm bảo tính khách quan và độ tin cậy trong quá trình đánh giá mô hình.

Với sự đa dạng về nguồn dữ liệu và số lượng lớn cặp câu song ngữ, PhoMT là một nguồn tài nguyên quý giá để xây dựng và đánh giá các mô hình dịch máy tiếng Việt-Anh. Trong project này, chúng tôi đã sử dụng 130,000 cặp câu từ tập huấn luyện và 1,200 cặp câu từ tập xác thực để huấn luyện mô hình, sau đó các cặp từ tập kiểm tra được chọn ngẫu nhiên để đánh giá mô hình đã được huấn luyện.

Tuy nhiên, do sự chênh lệch về số lượng cặp câu giữa các nguồn là khá lớn (từ 41,504 cặp câu của nguồn News đến 1,548,971 cặp câu của nguồn OpenSub), có thể dẫn đến việc số lượng từ vựng không thể bao quát hết được tất cả các chủ đề và lĩnh vực. Để giải quyết vấn đề này, chúng tôi đã chọn lọc các câu từ nhiều nguồn khác nhau trong tập huấn luyện, nhằm đảm bảo lượng từ vựng có thể bao quát được một cách đa dạng và toàn diện các chủ đề. Rất may là các dữ liệu thuộc cùng một nguồn được đặt theo các cụm, do đó việc tìm kiếm một khoảng dữ liệu chứa các câu đáp ứng tiêu chí trở nên dễ dàng hơn.

Trước khi huấn luyện mô hình Transformer, chúng tôi thực hiện một số bước tiền xử lý dữ liệu quan trọng nhằm đảm bảo chất lượng và hiệu quả của quá trình huấn luyện.

1. Tách từ và chuẩn hóa: Chúng tôi sử dụng thư viện Spacy để tách từ cho cả dữ liệu tiếng Anh và tiếng Việt. Các ký tự đặc biệt, khoảng trắng thừa được loại bỏ và tất cả các chữ cái được chuyển thành chữ thường.
2. Lọc dữ liệu theo độ dài câu: Để đảm bảo hiệu suất huấn luyện và tránh các vấn đề liên quan đến bộ nhớ, chúng tôi chỉ giữ lại các cặp câu có độ dài (tính theo số từ) nhỏ hơn một ngưỡng nhất định (max_strlen = 160).
3. Tạo từ điển: Chúng tôi xây dựng từ điển cho cả tiếng Anh (SRC) và tiếng Việt (TRG) dựa trên tập dữ liệu huấn luyện. Mỗi từ được gán một chỉ số duy nhất trong từ điển.
4. Tạo bộ dữ liệu và bộ lặp: Dữ liệu được chuyển đổi thành định dạng phù hợp để huấn luyện mô hình Transformer. Chúng tôi tạo các bộ lặp (iterator) để cung cấp dữ liệu theo từng batch trong quá trình huấn luyện.
5. Xử lý từ vựng: Các từ không có trong từ điển (out-of-vocabulary words) được thay thế bằng token <unk>. Các câu được thêm token <sos> ở đầu và token <eos> ở cuối để đánh dấu bắt đầu và kết thúc câu.
6. Thêm padding: Để đảm bảo các câu trong cùng một batch có cùng độ dài, chúng tôi thêm các token <pad> vào cuối các câu ngắn hơn.
7. Tạo mask: Chúng tôi tạo các mask để giúp mô hình tập trung vào các phần dữ liệu quan trọng và bỏ qua các phần padding.

Các bước tiền xử lý này giúp chuẩn bị dữ liệu một cách hiệu quả, đảm bảo tính nhất quán và sẵn sàng cho việc huấn luyện mô hình Transformer.

4. Training và thực hiện dịch

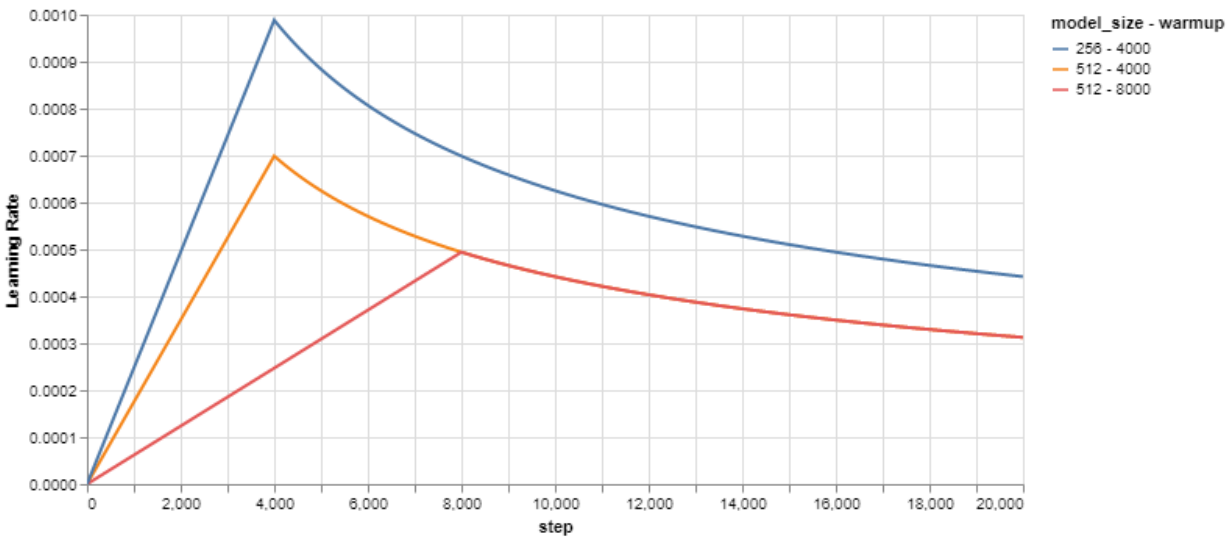
4.1. Thực nghiệm

- Training data: 130,000 cặp câu
- Validation data: 1,200 cặp câu
- Batch size: 1500
- Tỷ lệ Dropout: 0.1
- Learning rate: bắt đầu từ 0.0001
- Số Epoch: 30

4.2. Kịch bản

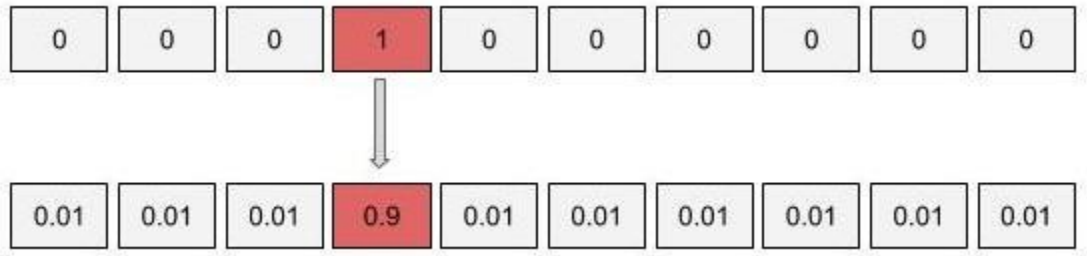
- Sử dụng mô hình Transformer với các tham số được thiết lập trong biến opt.
- Dữ liệu huấn luyện được lọc từ tập dữ liệu Phomt Vietnamese-English Machine Translation.
    - ⇨ Dữ liệu gốc được cắt thành các đoạn nhỏ hơn 160 từ.
    - ⇨ Sử dụng spaCy để phân tách từ cho cả tiếng Anh và tiếng Việt.
    - ⇨ Chỉ sử dụng một phần của tập dữ liệu huấn luyện (train_src_data[200000:330000] và train_trg_data[200000:330000]).
    - ⇨ Tạo các bộ lặp (iterator) để xử lý dữ liệu huấn luyện và dữ liệu xác thực.
- Optimizer sử dụng Adam với learning rate thay đổi theo số step trong quá trình huấn luyện mô hình Transformer, dựa trên công thức được đề xuất trong bài báo "Attention is All You Need":

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5})$$



- Giai đoạn khởi động (warm-up): Tỷ lệ học tập tăng tuyến tính từ 0 đến một giá trị cực đại trong một số bước nhất định (warm_up_step). Điều này giúp mô hình ổn định hơn trong giai đoạn đầu.
- Giai đoạn giảm dần: Sau giai đoạn khởi động, tỷ lệ học tập giảm dần theo một hàm số mũ, giúp mô hình hội tụ tốt hơn về cuối quá trình huấn luyện.
- Ảnh hưởng của d_model: Mô hình có kích thước lớn hơn (d_model = 512) thường yêu cầu tỷ lệ học tập khởi đầu cao hơn so với mô hình nhỏ hơn (d_model = 256).

- Sử dụng Label Smoothing để giảm hiện tượng overfit của mô hình. Khi mô hình quá tự tin vào dự đoán của mình (xác suất 100%), nó có thể học thuộc lòng dữ liệu huấn luyện và không tổng quát hóa tốt cho dữ liệu mới. Bằng cách phân bổ một phần nhỏ xác suất cho các lớp sai, label smoothing giúp mô hình học được các đặc trưng chung của các lớp, từ đó tổng quát hóa tốt hơn cho dữ liệu mới.



- Sau mỗi epoch, tính toán loss trên tập dữ liệu xác thực và tính điểm BLEU để đánh giá hiệu suất mô hình.

Điểm BLEU (Bilingual Evaluation Understudy) là một phương pháp đánh giá chất lượng của bản dịch máy so với một hoặc nhiều bản dịch tham chiếu (thường là do con người thực hiện). Điểm BLEU được tính toán dựa trên mức độ trùng khớp của các n-gram (chuỗi từ có độ dài n) giữa bản dịch máy và bản dịch tham chiếu.

4.3. Kết quả

Out[34]:
```
30.512013401303328
```

In [35]:
```
sentence='My family was not poor , and myself , I had never experienced hunger .'
trans_sent = translate_sentence(sentence, model, SRC, TRG, opt['device'], opt['k'], opt['max_s
trlen'])
trans_sent
```

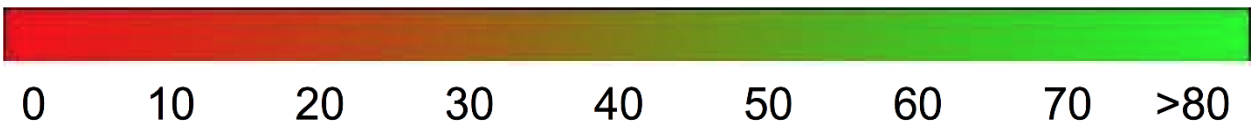
Out[35]:
```
'gia đình tôi không nghèo, và bản thân tôi, tôi chưa bao giờ trải qua nạn đói.'
```

- Điểm BLEU : 30.512013401303328**.**
- Kết quả dịch thử cho câu tiếng Anh "My family was not poor, and myself, I had never experienced hunger." là: "gia đình tôi không nghèo, và bản thân tôi, tôi chưa bao giờ trải qua nạn đói."

4.4. Phân tích kết quả

Có thể tham khảo biểu diễn dải màu sau để hiểu cách diễn giải thang điểm BLEU:

- < 10 (Đỏ): Bản dịch gần như vô dụng.
- 10-19 (Cam): Khó nắm bắt được ý chính.
- 20-29 (Vàng): Ý chính rõ ràng nhưng có nhiều lỗi ngữ pháp.
- 30-40 (Vàng xanh): Có thể hiểu được, bản dịch tốt.
- 40-50 (Xanh nhạt): Bản dịch chất lượng cao.
- 50-60 (Xanh lá): Chất lượng rất cao, đầy đủ và trôi chảy.
- > 60 (Xanh đậm): Chất lượng thường tốt hơn cả người dịch.



Phân tích kết quả dịch thử:

⇨ Điểm BLEU (30.512013401303328) khá cao.

- ⇨ Câu dịch đã dịch tương đối chính xác về mặt nghĩa.
- ⇨ Cụm từ "hunger" được dịch thành "nạn đói" thay vì "cảm giác đói" có thể cải thiện tính tự nhiên của bản dịch.

4.5. Kết luận

- ⇨ Chạy lại đoạn code: Cần chạy lại đoạn code để có kết quả BLEU đầy đủ và đánh giá chính xác hiệu suất của mô hình.
- ⇨ Cải thiện mô hình: Kết quả dịch thử cho thấy mô hình có khả năng dịch cơ bản, nhưng cần tinh chỉnh thêm để cải thiện độ chính xác và tự nhiên của bản dịch.

4.6. Kết quả trên tập kiểm tra

[49]:

```
import random

data_dir = "/kaggle/input/phomt-vietnamese-english-machine-translation/PhoMT/detokenization/test"
vi_file = os.path.join(data_dir, "test.vi")
en_file = os.path.join(data_dir, "test.en")

with open(vi_file, "r", encoding="utf-8") as f:
    vi_lines = f.readlines()

with open(en_file, "r", encoding="utf-8") as f:
    en_lines = f.readlines()

random_pairs = random.sample(list(zip(vi_lines, en_lines)), 10)

for i, (vi_sentence, en_sentence) in enumerate(random_pairs):
    print(f"Câu {i+1}:")
    print(f"- Tiếng Anh: {en_sentence}")
    trans_sentence =  translate_sentence(en_sentence, model, SRC, TRG, opt['device'], opt['k'], op
    print(f"- Dịch: {trans_sentence}")
    print(f"- Bản dịch mẫu: {vi_sentence}")
    print("------------------")
```

```
Câu 1:
- Tiếng Anh: Go to the library and read a variety of books.

- Dịch: đến thư viện và đọc nhiều cuốn sách khác nhau.
- Bản dịch mẫu: Tới thư viện và đọc nhiều sách.

------------------
Câu 2:
- Tiếng Anh: I'll be back in one moment.

- Dịch: tôi sẽ quay lại một lúc.
- Bản dịch mẫu: Em sẽ quay lại ngay.

------------------
```

5. Tài liệu tham khảo

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008). https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
- Alexander Rush. 2018. The Annotated Transformer. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 52–60, Melbourne, Australia. Association for Computational Linguistics. https://aclanthology.org/W18-2509
- Doan, L., Nguyen, L., Tran, N. L., Hoang, T., & Nguyen, D. Q. (2021). PHOMT: a High-Quality and Large-Scale benchmark dataset for Vietnamese-English machine translation. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. https://doi.org/10.18653/v1/2021.emnlp-main.369
- Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). spaCy: Industrial-strength Natural Language Processing in Python. Zenodo. https://doi.org/10.5281/zenodo.1212303
- Viet - Trung Tran. (2021). Viet - Trung Tran / vi_spacy. GitLab. https://gitlab.com/trungtv/vi_spacy
- Post, M. (2018, October). A call for clarity in reporting BLEU scores. In Proceedings of the Third Conference on Machine Translation: Research Papers (pp. 186-191). Association for Computational Linguistics. https://www.aclweb.org/anthology/W18-6319