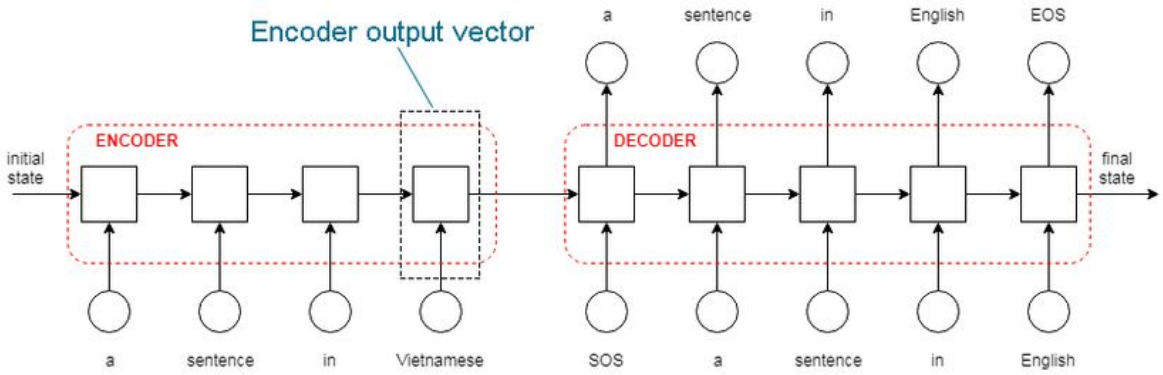
1.2 Cơ chế Attention

1.2.1 Tổng quan về cơ chế Attention

Trong mô hình sequence to sequence, chuỗi đầu vào được Encoder thành một vector ngữ cảnh có độ dài cố định đại diện cho lượng thông tin đầu vào. Sau đó, vector này được sử dụng làm trạng thái ban đầu của Decoder để sinh chuỗi đầu ra theo từng timestep. Cách hoạt động này yêu cầu đầu ra trạng thái cuối cùng của Encoder cần phải chứa tất cả các thông tin về câu nguồn. Dẫn đến việc vector context bị quá tải về mặt thông tin nếu như chuỗi nguồn dài.



Hình 1.5: Cơ chế Encoder-Decoder

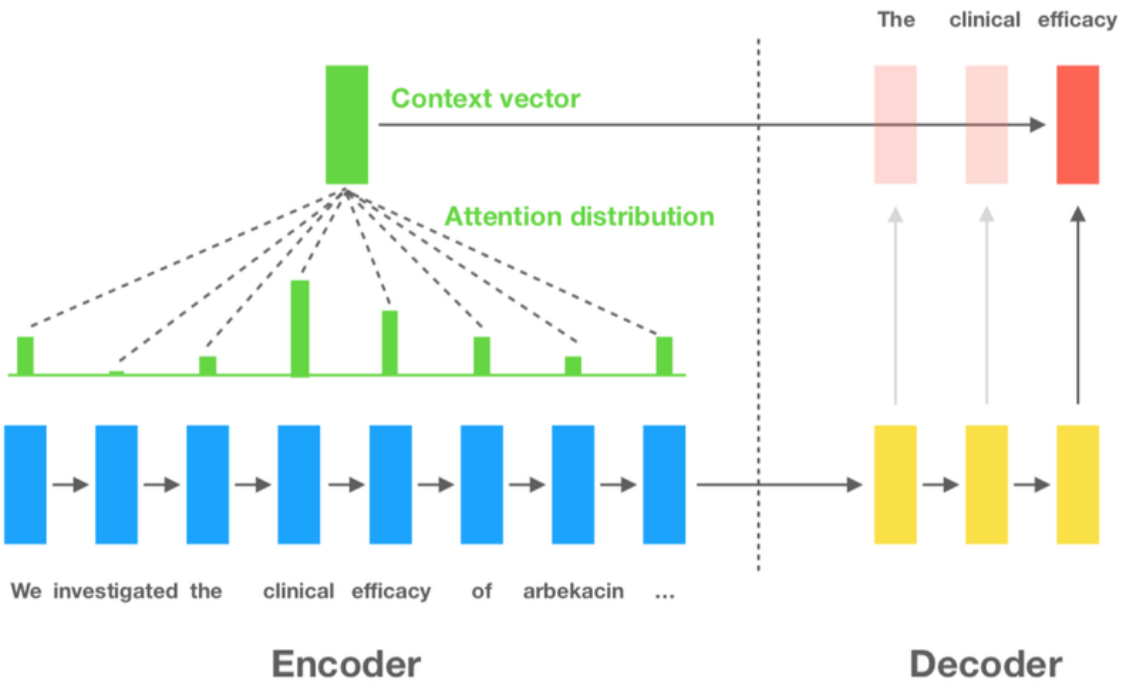
Cơ chế Attention được sinh ra dựa trên nhu cầu ghi nhớ các câu dài, một trong những vấn đề cơ bản của bài toán dịch máy. Attention cho phép mô hình tập trung vào một hoặc một vài ngữ cảnh địa phương trong câu, đó cũng là nguồn gốc tên gọi Attention. Thay vì để Encoder nén toàn bộ thông tin vào một vector trạng thái đầu ra cuối cùng, ta cho phép Decoder quan sát toàn bộ đầu ra của Encoder.

Ví dụ 2.

Giả sử cần dịch một câu *This is a book* sang tiếng Việt là *Đây là một quyển sách*. Ta có thể thấy sự tương ứng giữa từ ngữ ở 2 câu: *This* - *Đây* và *book* - *quyển sách*. Như vậy, việc dịch ra từ *quyển sách* thì từ *book* có ý nghĩa quan trọng hơn là các từ *this* hay *a* .... Và đó cũng chính là ý nghĩa cốt lõi của Attention.

Thay vì sử dụng context vector duy nhất một lần tại đầu vào của Decoder, ta sử dụng mỗi context vector riêng biệt cho từng phần để dự đoán ra từ kế tiếp, mỗi context vector được tổng hợp có trọng số từ các trạng thái ẩn trong Encoder (Hình 1.6).

Cách thức tổng hợp để ra được vector context (hay vector attention) có nhiều cách, trong phần này, chúng ta tìm hiểu về cơ chế tổng hợp attention của Bahdanau gọi là **Align and Jointly model** hay **Additive Attention** hay **Soft-Attention**.



Hình 1.6: Cơ chế Encoder-Decoder với Attention

Mô hình toán học của cơ chế Encoder-Decoder

Trong mô hình Encoder-Decoder, Encoder đọc một câu đầu vào gồm một chuỗi các vector $x = (x_1, \ldots, x_T)$ thành vector context c. Khi sử dụng RNN thì ta có:

$$h_t = f(x_t, h_{t-1})$$

và

$$c = q(\{h1, \ldots, h_T\})$$

Trong đó:

- $h_t \in \mathbb{R}^n$: trạng thái ẩn của tại bước t.
- c: Vector sinh bởi chuỗi trạng thái ẩn.
- f, q: Các hàm phi tuyến.

Decoder dự đoán y_t dựa vào context vector c và tất cả các từ trước nó $\{y_1, \ldots, y_{t-1}\}$. Nói cách khác, Decoder định nghĩa một xác xuất qua phép dịch y như sau:

$$p(y) = \prod_{t=1}^{T} p(y_t | \{y_1, \ldots, y_{t-1}\}, c)$$

Với: $y = (y_1, \ldots, y_T)$.
Khi sử dụng RNN cho Decoder, ta có:

$$p(y_t|\{y_1, \ldots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c)$$

Trong đó:

- g là hàm phi tuyến.
- s_t là trạng thái ẩn của RNN.

Mô hình toán học Encoder-Decoder với cơ chế Attention

Trong kiến trúc mới với Attention, xác suất để tạo ra đầu ra y_i tại Decoder biết các đầu tra trước đó $y_1, \ldots, y_{i-1}$ và câu nguồn x là:

$$p(y_i|\{y_1, \ldots, y_{i-1}\}, x) = g(y_{i-1}, s_i, c_i)$$

Với s_i là trạng thái ẩn của RNN tại bước thứ i: $s_i = f(s_{i-1}, y_{i-1}, c_i)$. Khác với mô hình Encoder-Decoder thông thường, tại mỗi đầu ra y_i, chúng ta sử dụng vector ngữ cảnh c_i. Context vector c_i là vector phụ thuộc vào bộ $h_1, \ldots, h_T$:

$$c = \sum_{j=1}^{T} \alpha_{ij} h_j$$

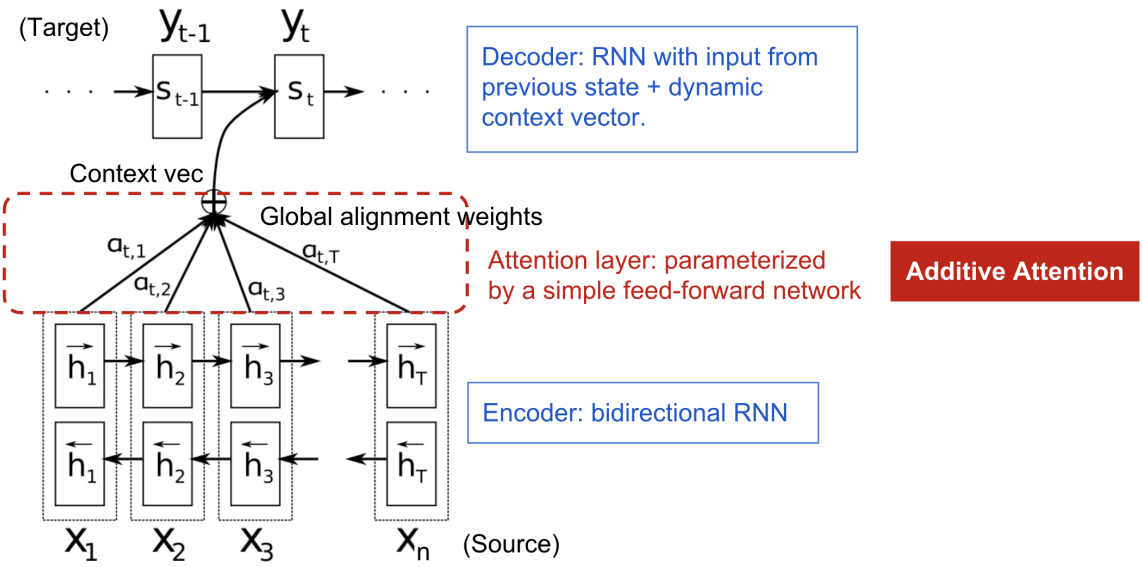
Trong đó:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T} exp(e_{ik})}$$

biểu thị trọng số cho từ mục tiêu y_i được dịch từ từ nguồn x_j.Tại đây:

$$e_{ij} = a(a_{i-1}, h_j)$$

được gọi là **alignment model** với mục đích đánh giá mức độ tương quan của từ tại vị trí j của Encoder với từ đầu ra tại vị trí i của Decoder bằng việc gán trọng số vào α_{ij}. e_{ij} biểu thị tầm quan trọng của h_j với trạng thái ẩn trước đó s_{i-1} trong việc quyết định trạng thái tiếp theo s_i và tạo ra y_i.

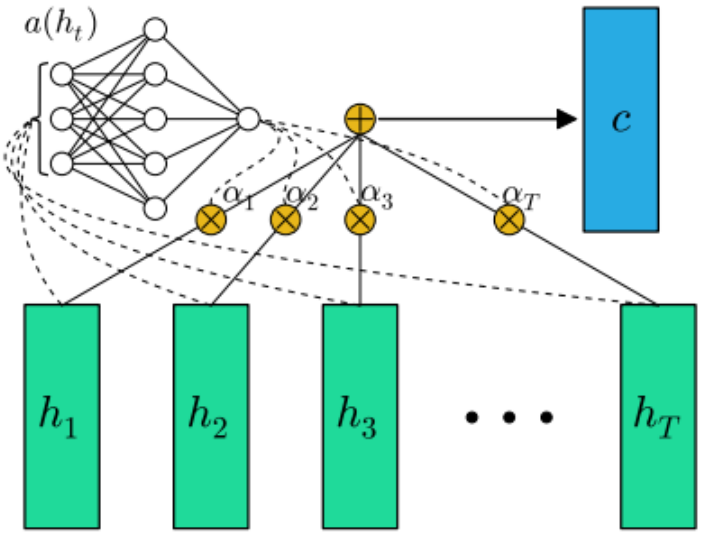


Hình 1.7: Hoạt động của Attention

Trong paper của tác giả Bahdanau, **alignment model** a được chọn với công thức như sau:

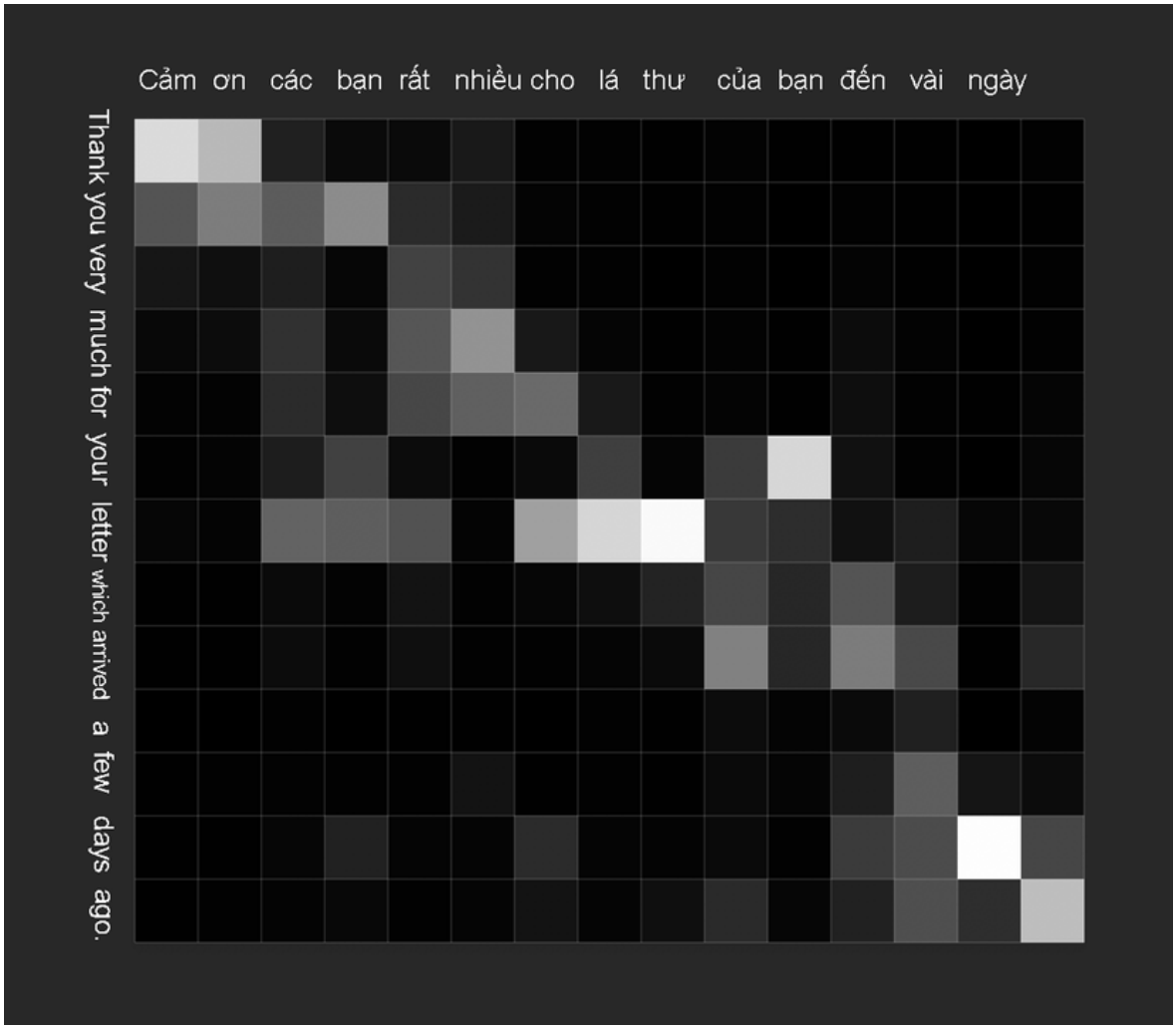
$$\text{score}(s_{i-1}, h_j) = a(s_{i-1}, h_j) = v_a^T \tanh(W_a s_{i-1} + U_a h_j)$$

thực chất là một multi-layer perceptron với W_a, U_a, v_a là các ma trận trọng số.



Hình 1.8: Cơ chế Attention Feed forward

Ma trận bất đối xứng (confusion matrix) được tạo ra bởi **alignment score** thể hiện mức độ tương quan giữa câu nguồn (source) và câu đích (target).



Hình 1.9: Ma trận bất đối xứng thể hiện mức độ tương quan của câu nguồn và câu đích

Nói cách khác, thực chất cơ chế Attention giúp mô hình tập trung vào phần quan trọng trên dữ liệu nguồn , bằng việc tạo ra một **alignment model a** tính các **alignment score** α_{ij} để thay đổi lại các trọng số của trạng thái ẩn của Encoder. Trong hình 1.9, các ô càng màu sáng thì biểu thị mức độ tương quan càng cao giữa từ x_j (source) và từ y_i (target).

1.2.2 Các biến thể của cơ chế Attention

Có nhiều cách tính **alignment score** như sau:

1. Content-base Attention [3]:

$$score(s_{i-1}, h_j) = \text{consine}[s_{i-1}, h_j]$$

2. Multiplicative Attention hay General Attention [4]:

$$score(s_{i-1}, h_j) = s_{i-1}^T W_a h_j$$

3. Dot Product [4]:

$$score(s_{i-1}, h_j) = s_{i-1}^T h_j$$

Ngoài ra có thể tham khảo thêm một số biến thể khác như Hard Attention [5], Hierarchical Attention [6].