CSE: **Faculty of Computer Science and Engineering**
Thuyloi University

DỊCH MÁY

Machine Translation

TS. Nguyễn Thị Kim Ngân

Dịch máy

- Dịch tự động một đoạn văn bản từ ngôn ngữ này sang ngôn ngữ khác
- Bộ dữ liệu dịch máy có ít nhất hai ngôn ngữ, ngôn ngữ nguồn và ngôn ngữ đích
- Mỗi câu trong ngôn ngữ nguồn được ánh xạ tới bản dịch tương ứng trong ngôn ngữ đích

Các vấn đề

- Xử lý sự giống nhau và khác nhau giữa các ngôn ngữ
  - Hình vị: số âm tiết/từ:
    - Ngôn ngữ đơn âm tiết (tiếng Việt, Trung Quốc:1 tiếng/từ
    - Ngô ngữ đa âm tiết (Siberian Yupik), 1 từ = cả 1 câu
- Mức độ phân chia âm tiết

Các vấn đề

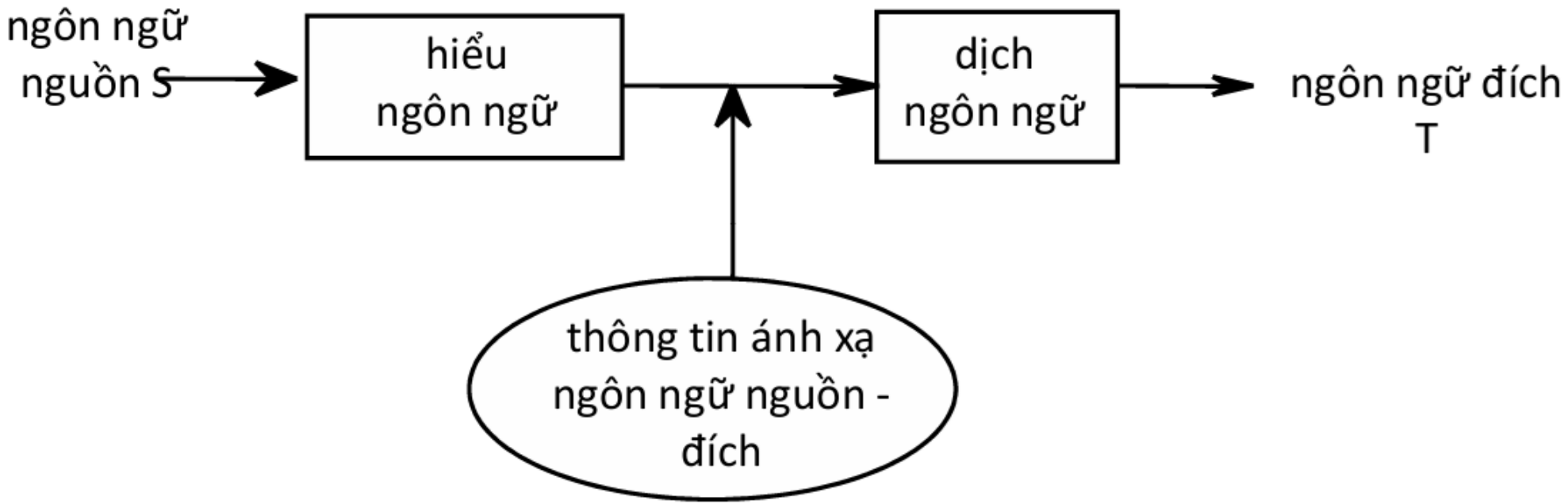
- Cú pháp: trật tự từ trong câu
  - The (affix1) red (affix2) flag (head)
  - Lá cờ (head) đỏ (affix2) ấy (affix1)
- Các nét riêng biệt

English	brother	Vietnamese	anh em
English	wall	German	wand (inside) mauer(outside)
German	berg	English	hill mountain

Các vấn đề

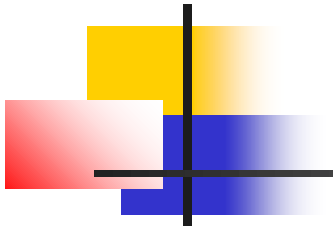
- Không gian khái niệm
  - Khoảng trống từ vựng:
    - Trong tiếng Nhật không có từ nào nghĩa privacy (sự riêng tư); tiếng Anh không có từ ứng với yakoko (long hiếu thảo)

Ba khối chính trong dịch máy



Phương pháp dịch máy

- Sử dụng thông tin cú pháp
- Chuỗi sang chuỗi (sequence to sequence)
- Cơ chế tập trung (attention mechanism)

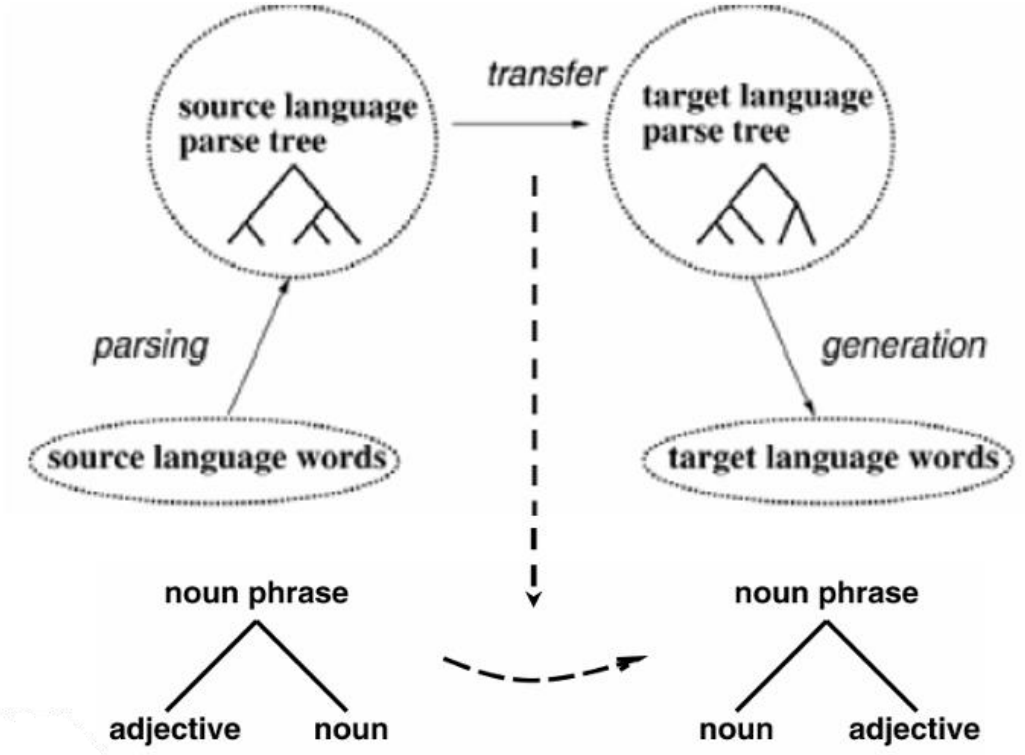


SỬ DỤNG THÔNG TIN CÚ PHÁP

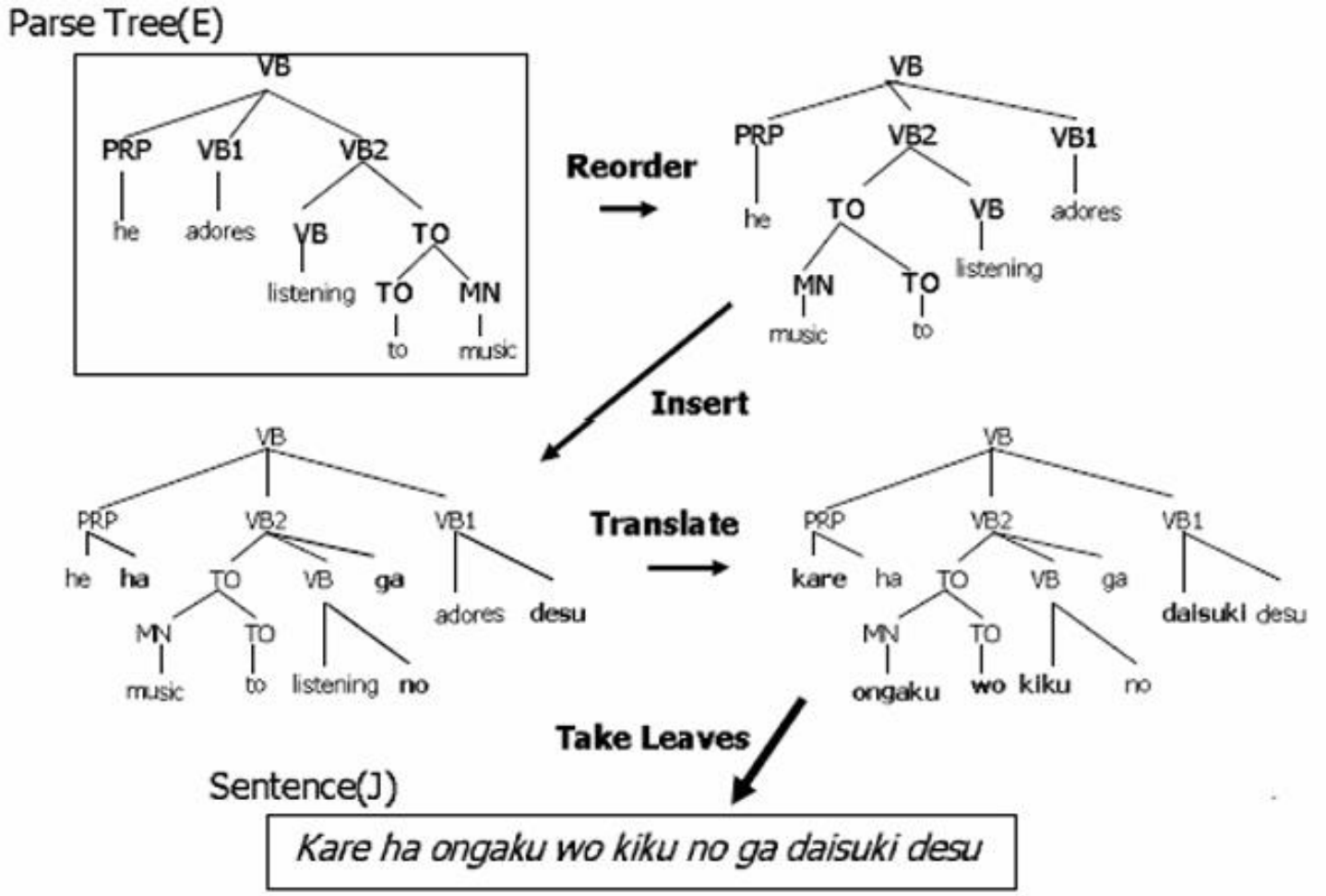
Dịch máy sử dụng thông tin cú pháp

- Cần thông tin ngữ pháp
- Cần các ràng buộc khi sắp lại câu
- Khi chèn các từ chức năng vào câu, cần đặt ở vị trí chính xác
- Khi dịch từ cần sử dụng từ có cùng từ loại với nó

Sơ đồ chuyển đổi



Cây cú pháp (Anh) -> câu (Nhật)



CHUỖI SANG CHUỖI

Sequence to Sequence

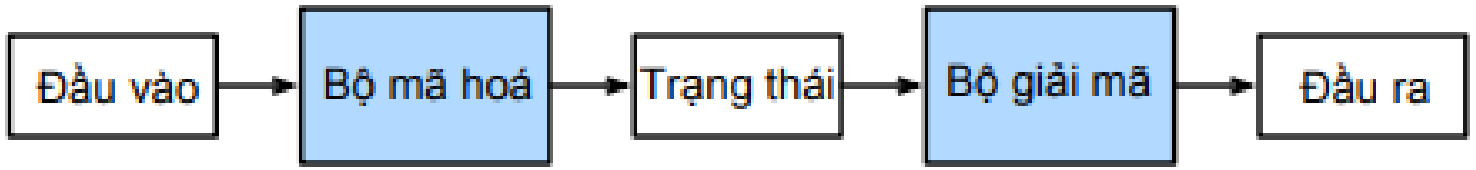
Dịch máy

Các bước tiền xử lý dữ liệu dịch máy gồm:

- Đọc và tiền xử lý dữ liệu từ tệp nguồn
- Token hóa: Mỗi token là một từ hoặc một dấu câu
- Bộ từ vựng: Mỗi ngôn ngữ cần xây dựng một bộ từ vựng
- Nạp dữ liệu: Mỗi câu là một chuỗi có độ dài num_steps. Nếu một câu dài hơn num_steps, ta sẽ cắt bớt độ dài của nó, ngược lại nếu một câu ngắn hơn num_steps, thì ta sẽ đệm thêm token <pad>. Bằng cách này, ta có thể chuyển bất cứ câu nào về một độ dài cố định

Kiến trúc Mã hóa –Giải mã

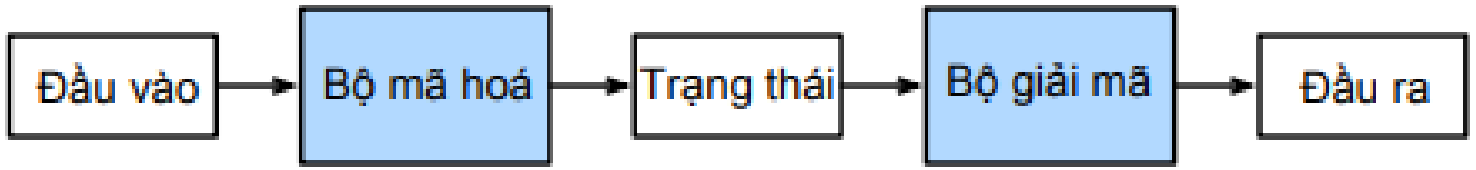
- Kiến trúc mã hoá - giải mã (encoder-decoder architecture) là một khuôn mẫu thiết kế mạng nơ-ron
- Kiến trúc này có 2 phần: bộ mã hoá và bộ giải mã
  - Bộ mã hoá đóng vai trò mã hoá đầu vào thành trạng thái chứa vài tensor
  - Bộ giải mã nhận đầu ra của Bộ mã hóa để giải mã sinh ra đầu ra



Kiến trúc Mã hóa –Giải mã

Trong dịch máy, bộ mã hóa biên đổi một câu nguồn (“Hello world”) thành một vector chứa thông tin ngữ nghĩa của câu đó. Bộ giải mã sử dụng trạng thái này để dịch câu sang ngôn ngữ đích (“Bonjour le monde”)

- Bộ mã hóa: Một mạng nơ ron thông thường, nhận đầu vào (một câu nguồn) và trả về đầu ra
- Bộ giải mã: Nhận 2 đầu vào (câu đích và trạng thái). Nó trả về đầu ra là một câu đích

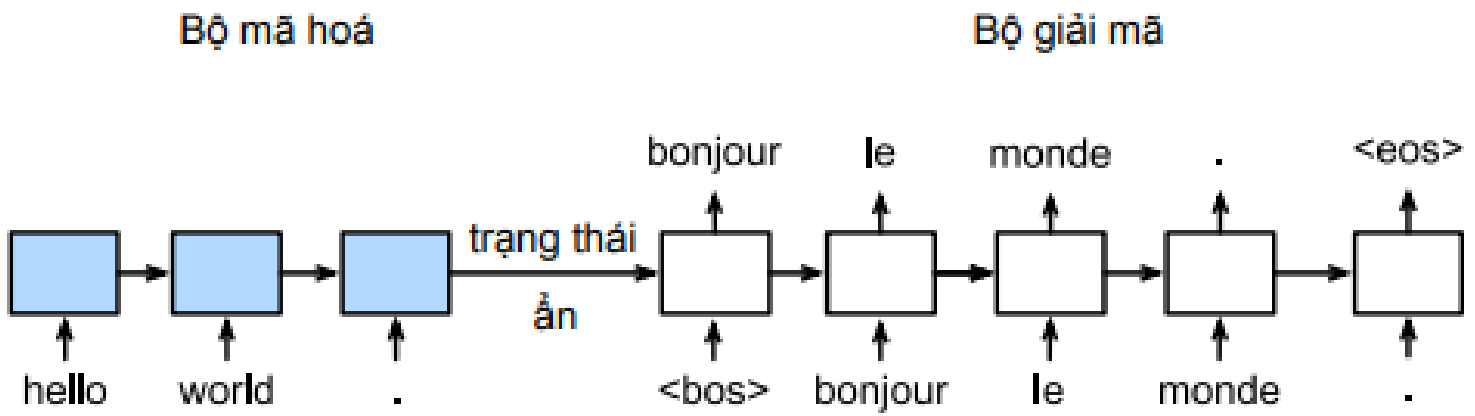


Kiến trúc Mã hóa –Giải mã

- Mô hình mã hoá - giải mã bao gồm một bộ mã hoá và một bộ giải mã
- Phương thức truyền xuôi cho quá trình huấn luyện: nhận cả đầu vào bộ mã hoá và đầu vào bộ giải mã cùng các đối số bổ sung không bắt buộc
- Mô hình tính đầu ra của bộ mã hoá để khởi tạo trạng thái bộ giải mã, sau đó trả về đầu ra của bộ giải mã

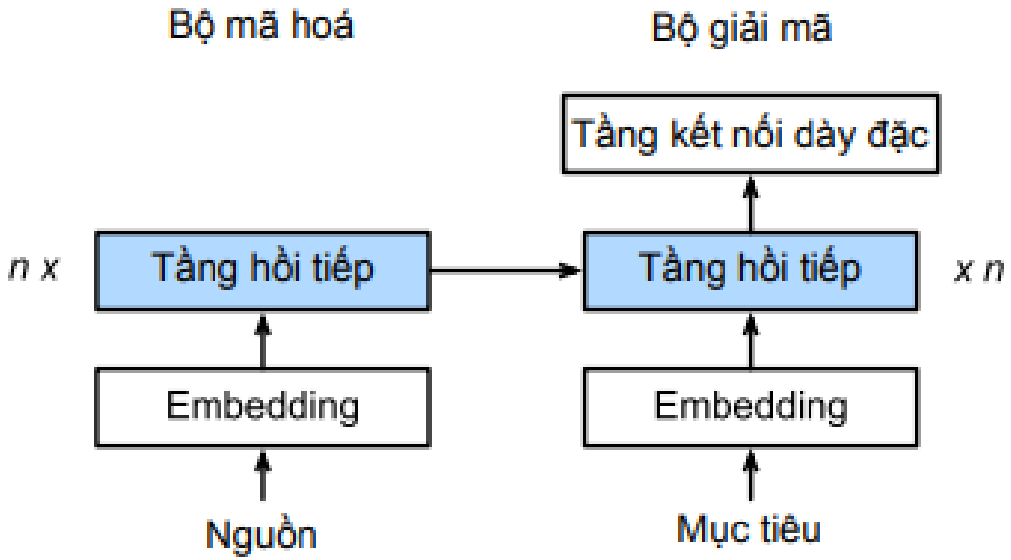
Chuỗi sang chuỗi

- Mô hình chuỗi sang chuỗi (Sequence to Sequence – seq2seq) dựa trên kiến trúc mã hóa - giải mã để sinh ra chuỗi đầu ra từ chuỗi đầu vào như minh họa trong

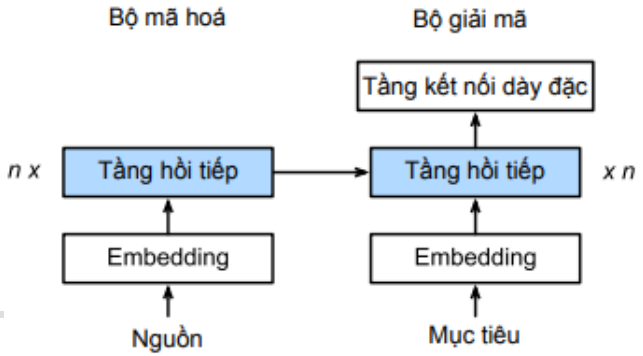


Chuỗi sang chuỗi

- Bộ mã hóa và bộ giải mã sử dụng mạng nơ-ron hồi tiếp (RNN, GRU, LSTM) để xử lý các chuỗi đầu vào với độ dài khác nhau
- Trạng thái ẩn của bộ giải mã được khởi tạo trực tiếp từ trạng thái ẩn của bộ mã hóa, giúp truyền thông tin từ bộ mã hóa tới bộ giải mã



Bộ mã hóa



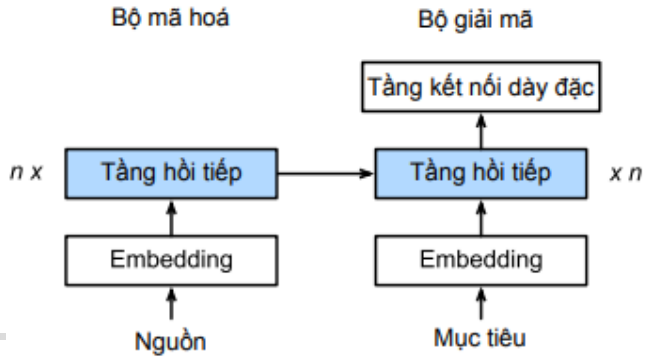
- Mã hóa thông tin của các chuỗi đầu vào với độ dài khác nhau thành một vector ngữ cảnh c bằng các tầng RNN
- Giả sử có một chuỗi đầu vào $x_1,...,x_T$, trong đó x_t là từ thứ t. Tại bước thời gian t, mô hình RNN sẽ có hai vector đầu vào:
  - vector đặc trưng $\mathbf{x_t}$ của x_t
  - trạng thái ẩn của bước thời gian trước đó h_{t-1}

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

- Bộ mã hóa nắm bắt thông tin của tất cả các trạng thái ẩn và mã hóa chúng thành vector ngữ cảnh **c** bằng hàm q:

$$\mathbf{c} = q(\mathbf{h_1}, \ldots, \mathbf{h_T})$$

Bộ giải mã



- Giả sử đầu ra của tập huấn luyện là $y_1, y_2, \ldots, y_{T'}$. Tại mỗi bước thời gian t', xác suất có điều kiện của đầu ra $y_{t'}$ phụ thuộc vào:
  - đầu ra trước đó $y_1, y_2, \ldots, y_{t'-1}$
  - vector ngữ cảnh c

$$P(y_{t'} \mid y_1, y_2, \ldots, y_{t'-1}, c)$$

- Có thể sử dụng một mạng RNN khác trong bộ giải mã. Tại mỗi bước thời gian t', bộ giải mã cập nhật trạng thái ẩn của nó $s_{t'}$ thông qua 3 đầu vào:
  - Vector đặc trưng $y_{t'-1}$ của $y_{t'-1}$
  - Vector ngữ cảnh c
  - Trạng thái ẩn tại bước thời gian trước đó $s_{t'-1}$

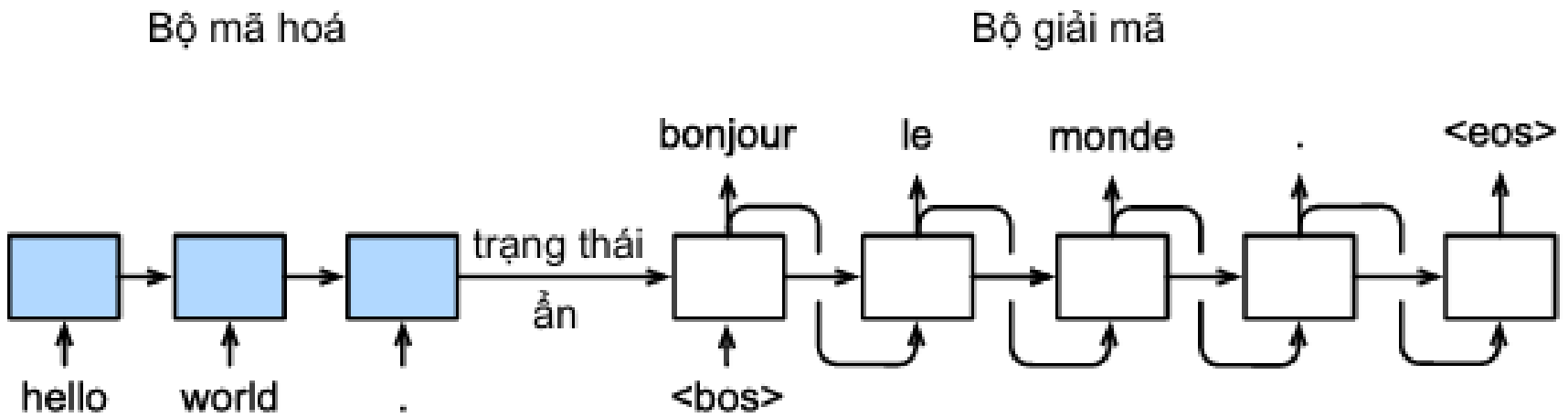
$$s_{t'} = g(y_{t'-1}, c, s_{t'-1})$$

Hàm mất mát

- Tại mỗi bước thời gian, bộ giải mã tạo ra một vector điểm tin cậy có kích thước bằng bộ từ vựng để dự đoán các từ
- Sử dụng softmax để tính xác suất và sau đó sử dụng hàm mất mát entropy chéo để tính mất mát

Dự đoán

- Sử dụng phương pháp *tìm kiếm tham lam (greedy search)* để tạo chuỗi đầu ra
- Token đầu vào cho các bước thời gian sau là token được dự đoán từ bước thời gian trước đó



Beam Search

- Thêm ký hiệu kết thúc câu “<eos>” vào sau mỗi câu
- Gọi kích thước của bộ từ điển đầu ra Y (chứa tất cả các từ có thể xuất hiện ở chuỗi đầu ra, bao gồm cả “<eos>”) là |Y|, và chiều dài tối đa của chuỗi đầu ra là T’. Có tổng cộng $O(|Y|^T)$ chuỗi đầu ra có thể được sinh ra. Tất cả những chuỗi con nằm phía sau “<eos>” trong chuỗi đầu ra sẽ bị lược bỏ
- Ký hiệu c là vector ngữ cảnh mã hóa thông tin của tất cả trạng thái ẩn từ đầu vào

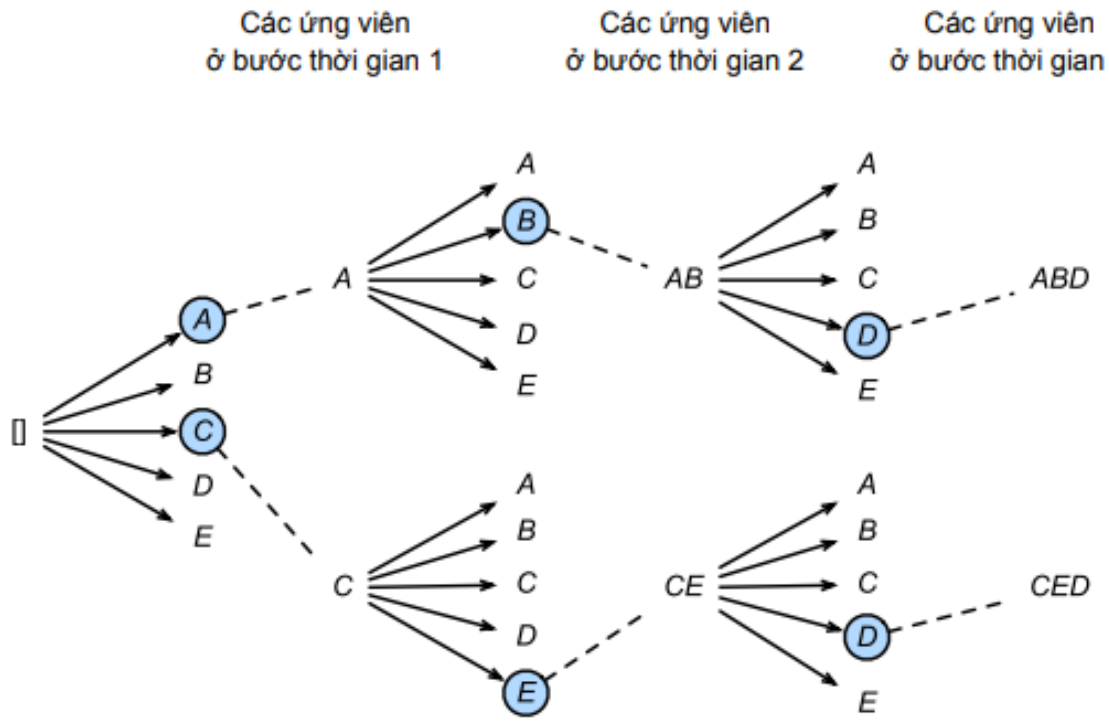
Beam Search

Tìm kiếm chùm (beam search) là một thuật toán cải tiến dựa trên tìm kiếm tham lam. Nó có một siêu tham số k gọi là kích thước chùm (beam size)

- Tại bước thời gian 1: Chọn k từ có xác suất có điều kiện cao nhất để bắt đầu k chuỗi đầu ra ứng viên
- Tại các bước thời gian tiếp theo: Dựa trên k chuỗi đầu ra ứng viên từ bước thời gian trước đó, ta tính và chọn k chuỗi có xác suất có điều kiện cao nhất trong tổng số k|Y| khả năng. Đây sẽ là các chuỗi đầu ra ứng viên cho bước thời gian đó
- Cuối cùng: Lọc ra các chuỗi có chứa “<eos>” từ các chuỗi đầu ra ứng viên tại mỗi bước thời gian và loại bỏ tất cả các chuỗi sau ký tự đó để thu được tập các chuỗi đầu ra ứng viên cuối cùng

Beam Search

Minh họa quá trình tìm kiếm chùm. Kích thước chùm bằng 2 và độ dài tối đa của chuỗi đầu ra bằng 3. Các chuỗi đầu ra ứng viên là A, C, AB, CE, ABD, và CED



Code

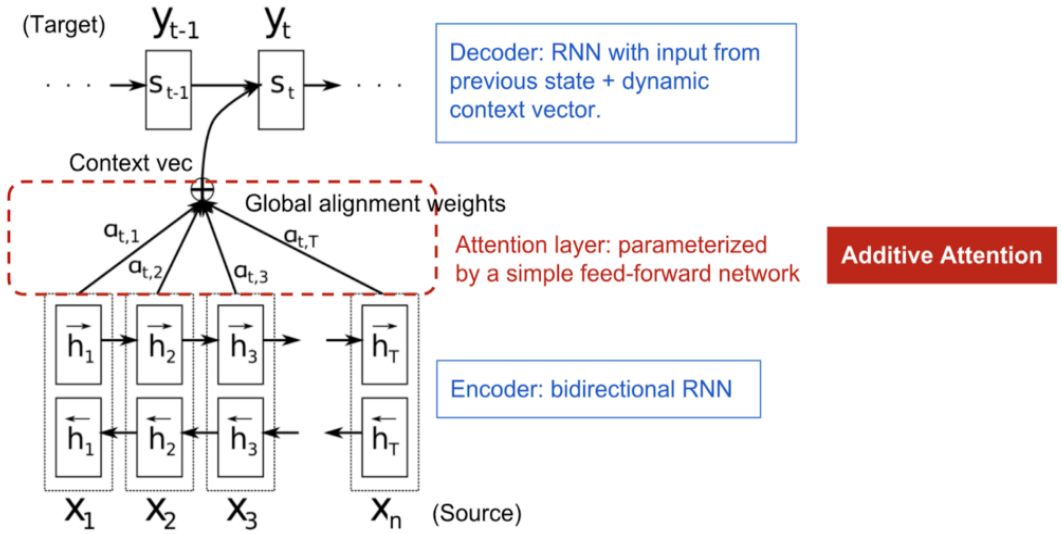
Sử dụng thư viện Tensorflow

https://www.tensorflow.org/addons/tutorials/networks_seq2seq_nmt

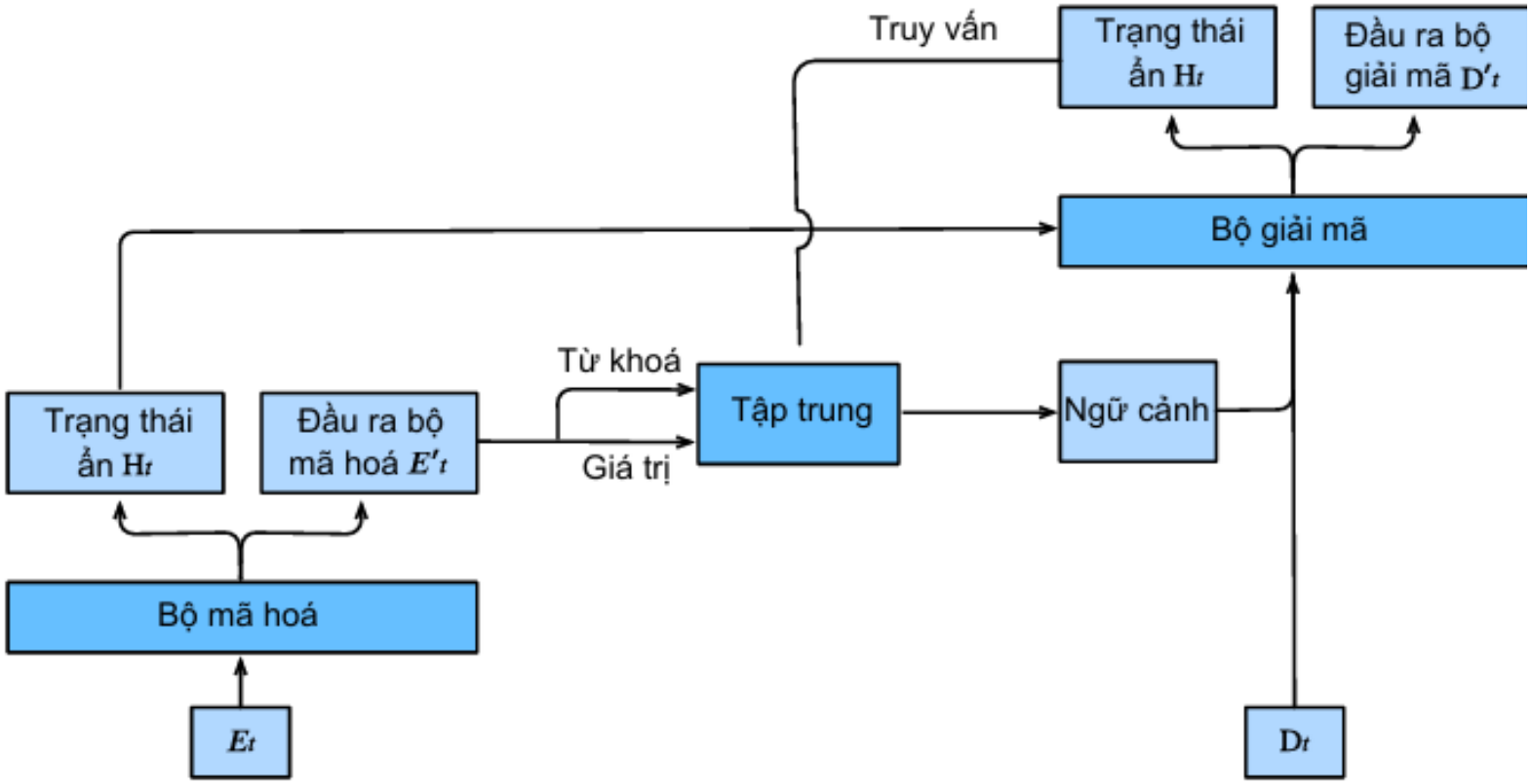
CƠ CHẾ TẬP TRUNG
Attention Mechanism

Cơ chế Attention

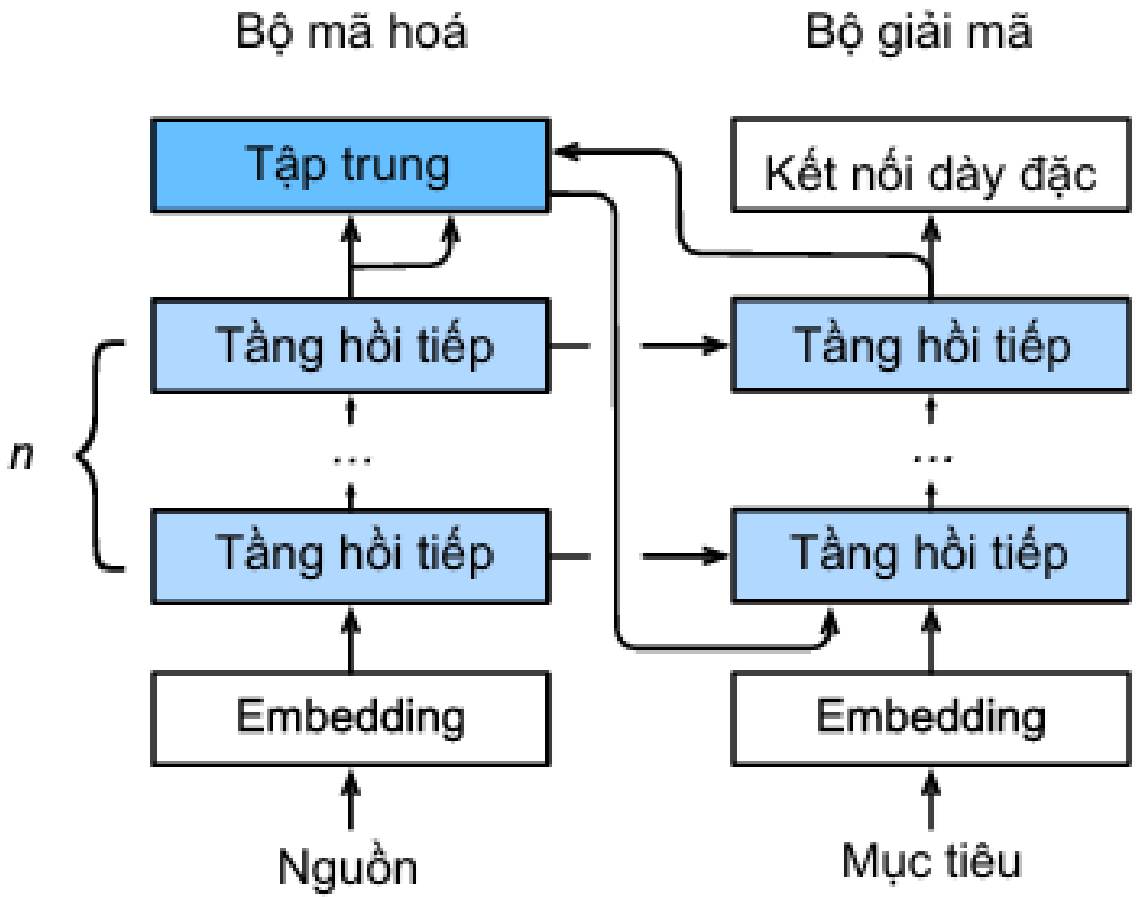
Giả sử chuỗi dữ liệu đầu vào là $(x_1, x_2, x_3, .., x_n)$, tầm quan trọng của mỗi thông tin đầu vào có thể là khác nhau khi dự đoán chuỗi đầu ra. Mục đích của cơ chế Attention là tìm ra bộ trọng số để đánh giá tầm quan trọng của x_1, x_2, x_3 , …, x_n khi dự đoán chuỗi đầu ra



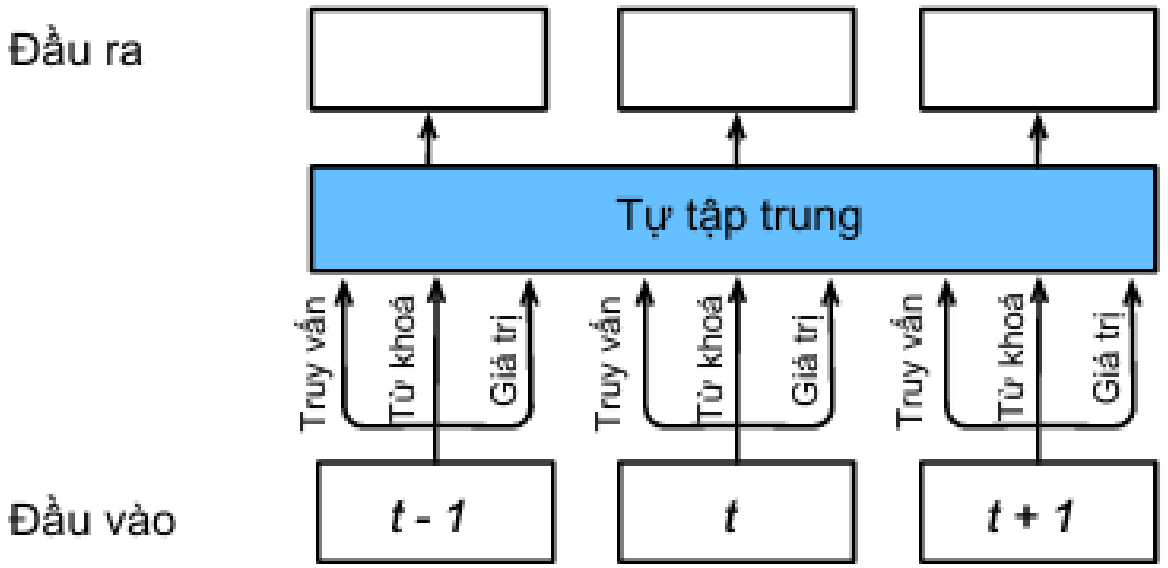
Cơ chế tập trung



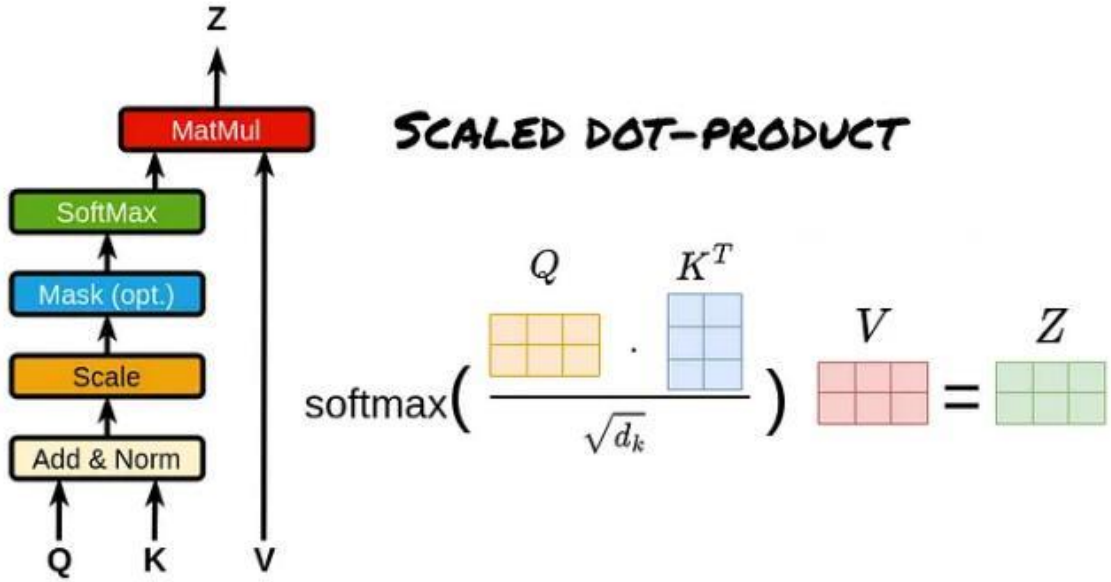
Cơ chế tập trung trong mô hình chuỗi sang chuỗi



Self-attention



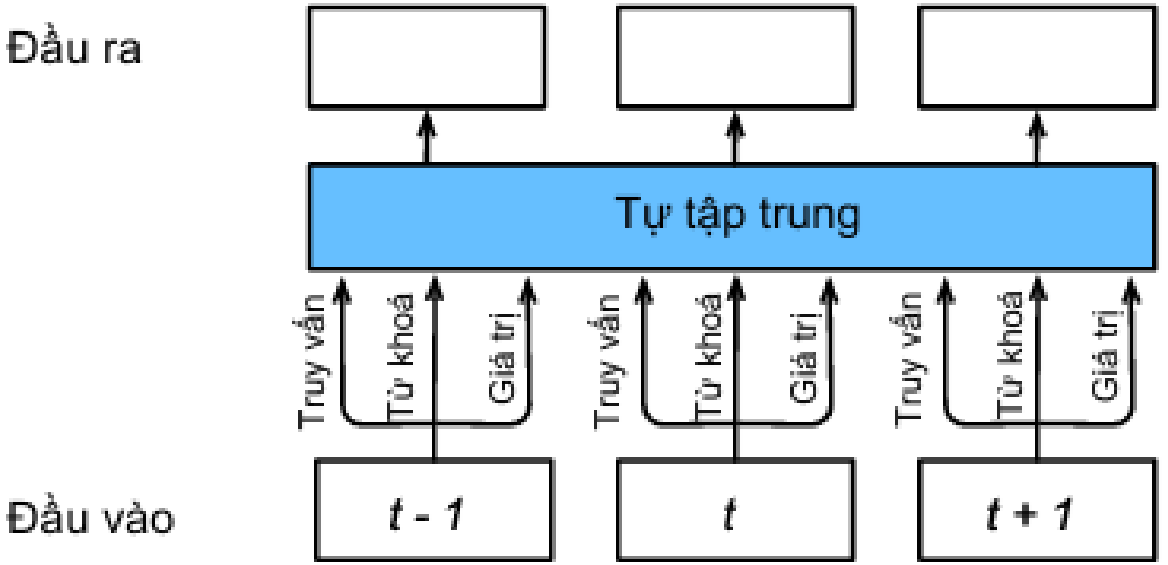
Scaled Dot-Product Attention

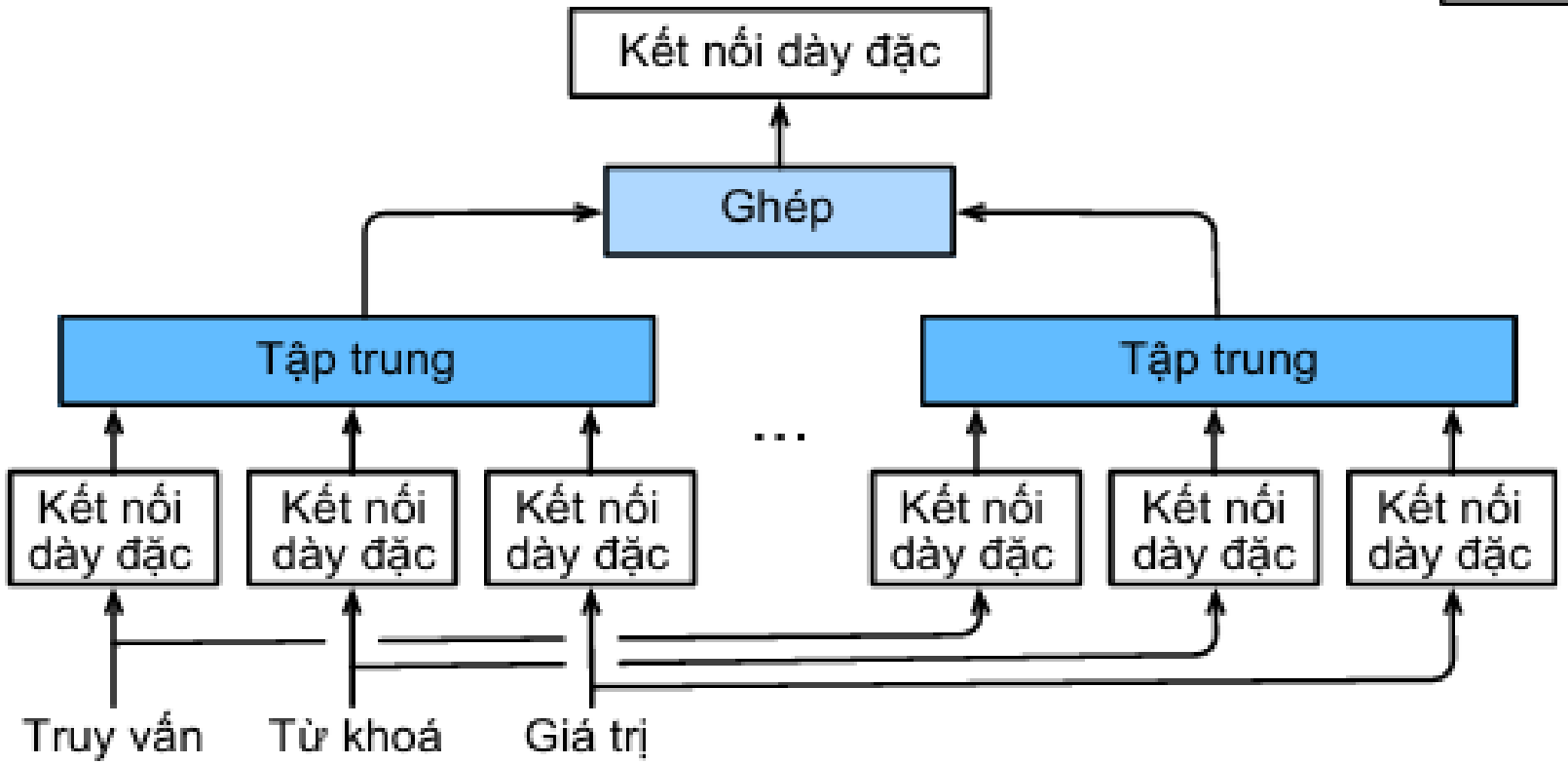


Ma trận đầu ra của tầng attention:

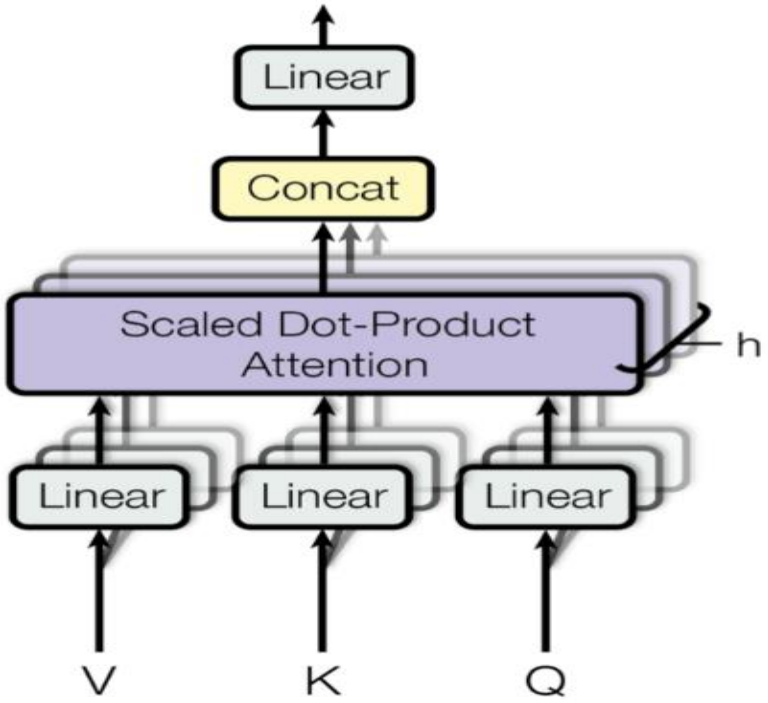
$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Multi-head Attention





Multi-Head Attention

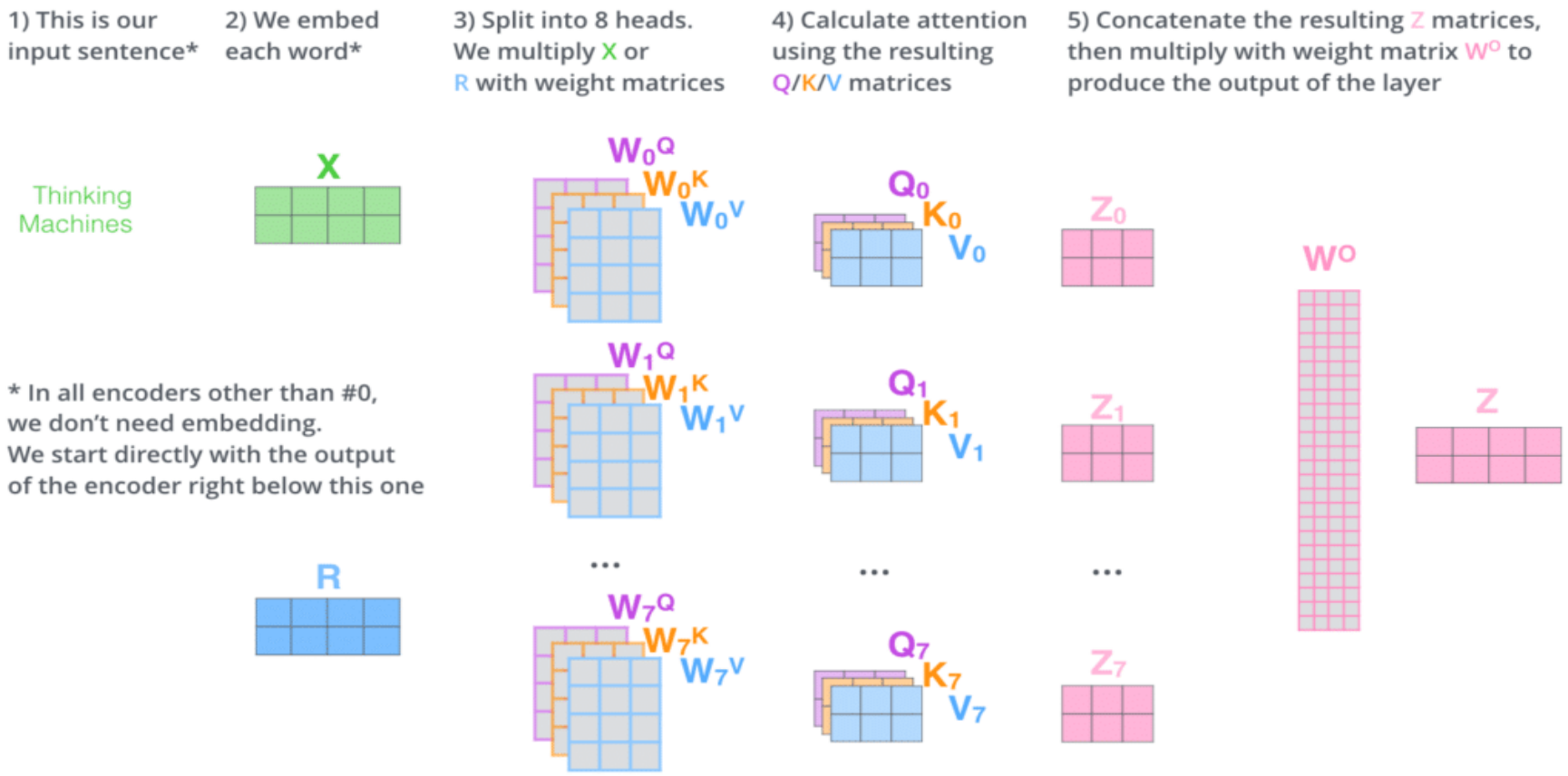


Ma trận đầu ra:

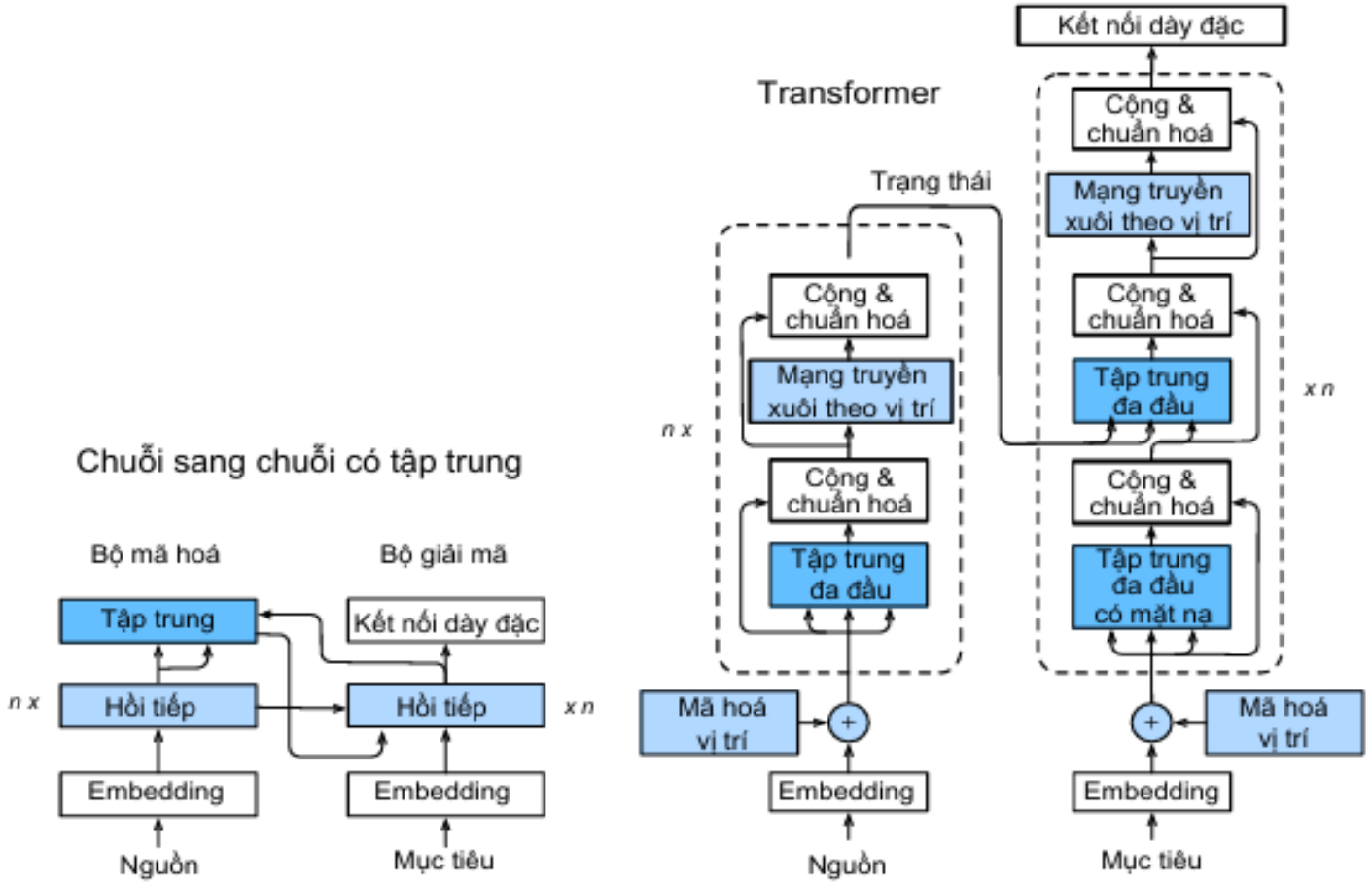
$$Head_i = Attention_i(x) = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})V$$

$$MuhtiHead(Q,K,V) = ConCat(Head_1, Head_2, Head_3, \ldots\ldots, Head_h)W^0$$

Multi-Head Attention

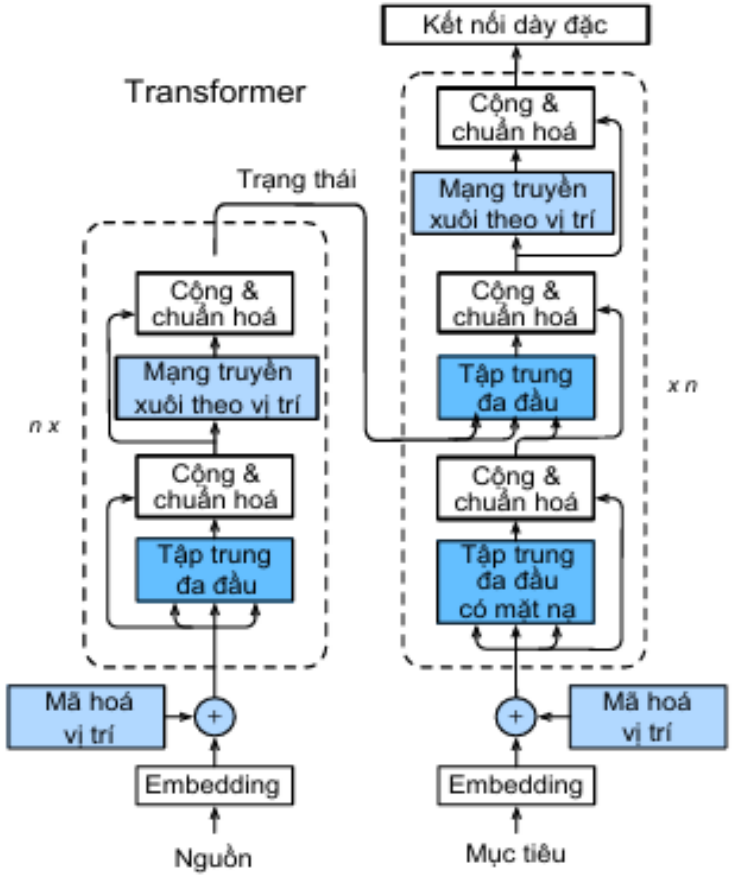


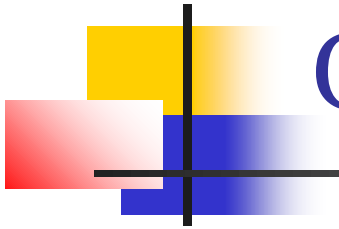
Kiến trúc Transformer



Kiến trúc Transformer

- Khối Transformer: Một tầng hồi tiếp trong seq2seq được thay bằng Khối Transformer
  - Bộ mã hóa gồm: một tầng *tập trung đa đầu vào* và một *mạng truyền xuôi theo vị trí*
  - Bộ giải mã gồm: Một tầng tập trung đa đầu khác để nhận vào trạng thái bộ mã hóa
- Cộng và chuẩn hóa: Đầu vào và đầu ra của tầng tập trung đa đầu hoặc mạng truyền xuôi theo vị trí được xử lý bởi 2 tầng 'cộng và chuẩn hóa' bao gồm cấu trúc phần dư và tầng chuẩn hóa theo tầng (layer normalization)
- Biểu diễn vị trí: được sử dụng để thêm thôn tin vị trí vào từng phần tử trong choỗi





Code

https://www.tensorflow.org/text/tutorials/nmt_with_attention