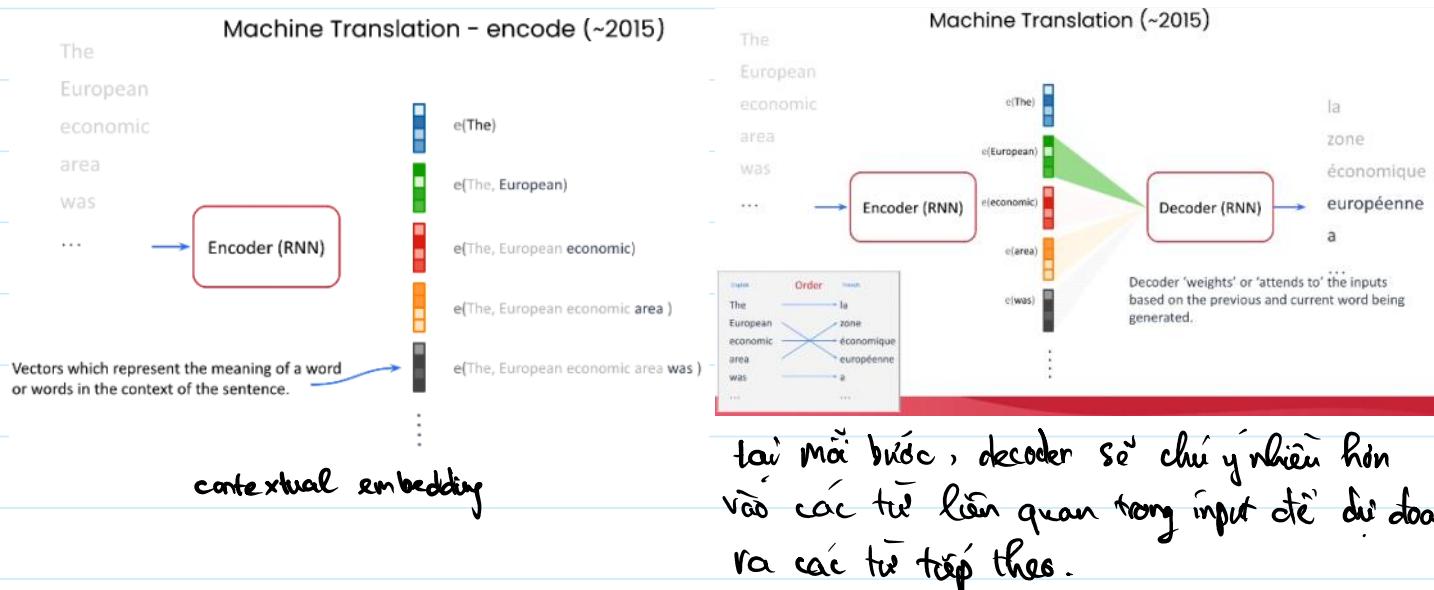


Introduction

Thursday, June 19, 2025 11:31 AM

2014: bài toán machine translation English → French

Phát hiện ra sự hiệu quả của kiến trúc encoder-decoder sử dụng attention mechanism



2017: "Attention is all you need"

Giới thiệu kiến trúc transformer
dạng tổng quát của attention mechanism

Kiến trúc của transformer đã thiết kế để tận dụng tài nguyên của GPU

Attention Is All You Need

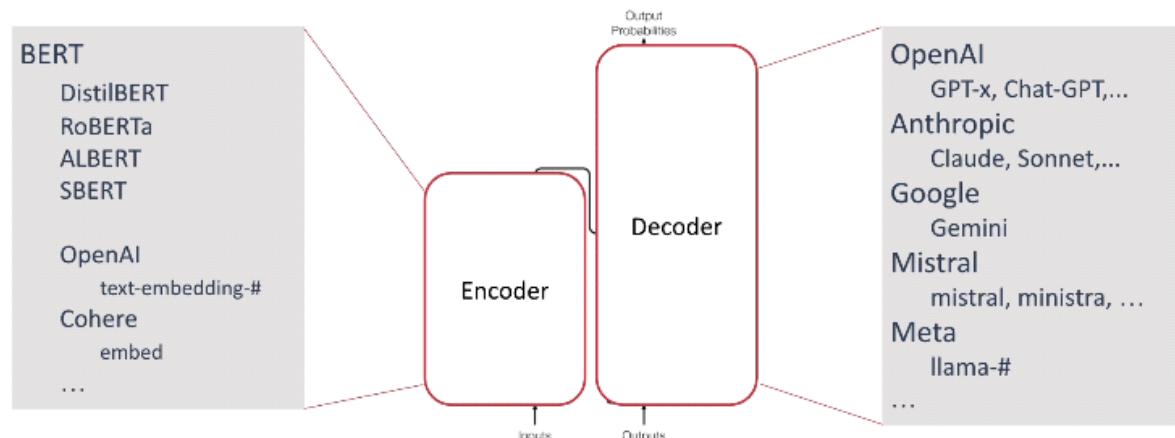
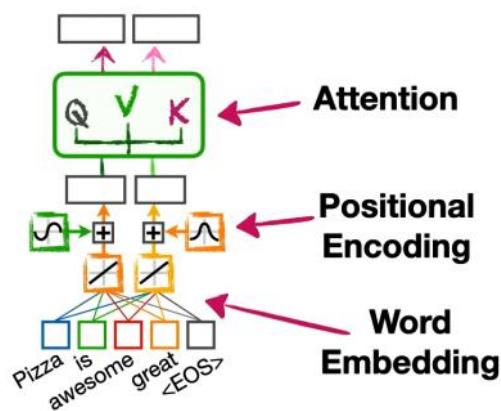


Figure 1: The Transformer - model architecture.

Kiến trúc của transformer là nền tảng cho nhiều kiến
trúc màng cao hơn



3 bộ phận chính của transformers

- Word embedding: chuyển text string thành vector số (tokens)
transformers bao chất là 1 neural network
⇒ chỉ có thể xử lý data dạng số

(Sử dụng các tokenizer trainable)

- Positional encoding: mã hóa thông tin về vị trí của các token trong câu

theo số đếm, positional encoding cũng trả về 1 vector

kết quả này sẽ cộng trực tiếp vào embedded vector làm input cho attention layers

- Attention Layer: liên kết mối quan hệ giữa các từ trong câu

có nhiều loại attention

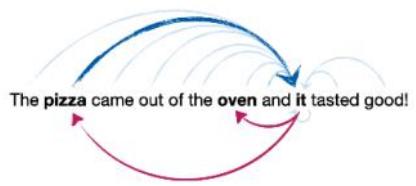
- self attention, masked self-attention
- cross attention
- multi-head attention

SELF - ATTENTION

Danh giá mức độ tương đồng của mỗi từ với tất cả các từ còn lại trong câu (bao gồm cả chính nó)



Điểm tương đồng này sẽ dùng để encode các từ trong câu
có vai trò quan trọng trong việc quyết định cách mà transformer encode



The **pizza** came out of the **oven** and it tasted good!

vD: *tù "it" có ^{điểm} tương đồng với "pizza"*

Ct tổng quát :

$$\text{Attention } (Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

query ↓ key ↓ value

trong đó :

- Q (query) : đại diện cho thông tin mà input đang muốn tìm kiếm

$$Q = X \cdot W_Q \quad \left(\begin{array}{l} X \in \mathbb{R}^{\text{seq-len} \times d_{\text{model}}} \\ W_Q \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}} \end{array} \right)$$

query dc dùng để so khớp với các key

- K (key) : đại diện cho thông tin mỗi từ đang chứa

$$K = X \cdot W_K \quad (\text{tương tự phần trên})$$

key dc dùng để so khớp với query

- V (value) : thông tin thực tế dc rút ra từ mỗi từ

$$V = X \cdot W_V \quad (\text{trong tu})$$

value là thông tin dc truyền đi nếu token tương ứng dc chọn

!1 W_Q, W_K, W_V đều là các trọng số (Learnable parameters)

Các thành phần trong công thức :

1) $\frac{QK^T}{\sqrt{d_k}}$: tính độ tương đồng cho tất cả các cặp từ okre trên ý tưởng của cosine similarity (đo mỗi từ oktay)
 (để đại diện bởi)
 1 vector

$\frac{QK^T}{\sqrt{d_k}} \in \mathbb{R}^{\text{seq-len} \times \text{seq-len}}$: mỗi phần tử tại vị trí (i, j) đại diện cho mức độ chú ý của token i so với token j

2) softmax : dùng để néi các similarity score về $1/p^2$ xác suất

thể hiện % mức độ ảnh hưởng của mỗi từ

phần tử tại vị trí (i, j) thể hiện mức độ phản hồi (%) sự chú ý của token i cho token j

3, Kết quả cuối cùng : $\text{softmax}(\cdot) \cdot V \in \mathbb{R}^{\text{seq-len} \times d_{\text{model}}}$

mỗi hàng thứ i thể hiện lượng thông tin mà token i tổng hợp dc từ các token còn lại

→ ... you can say ...

mỗi hàng thứ i thể hiện bằng thông tin mà token i tổng hợp được từ các token còn lại

Nhận xét :

Như vậy, sau một attention layer, mỗi token sẽ nắm bắt thông tin về bản thân nó, về ngữ cảnh, mối quan hệ của nó với các token khác trong câu...

⇒ token đã được ngữ cảnh hóa (contextual embedding)

Qua mỗi attention layer, các thông tin đó sẽ bắt đầu tăng, khai quật hóa mức độ cao hơn (tăng từ 0%)

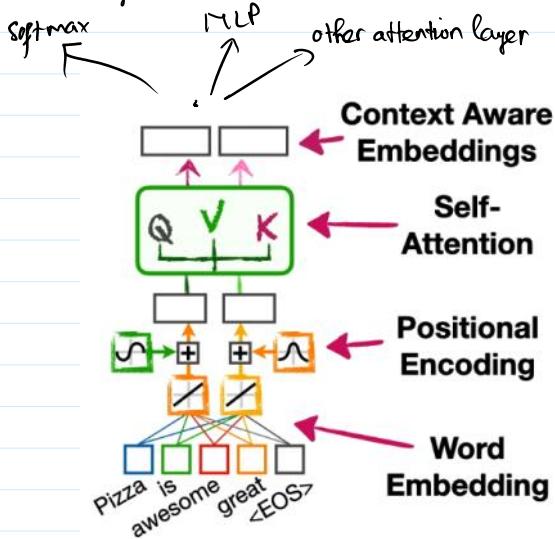
Self-attention vs Masked Self-attention

Friday, June 20, 2025 1:40 PM

Encoder-only transformer:

chỉ sử dụng self-attention
sử dụng cho mục đích trích xuất thông tin
(đặc trưng của kiến trúc encoder)

Sentiment classification



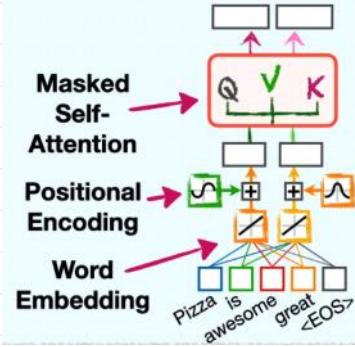
San attention layer, contextual embedding có thể có nhiều đặc trưng và sử dụng với nhiều mục đích đa dạng

Decoder-only transformer:

sử dụng masked self-attention thay cho self-attention
sử dụng cho mục đích sinh (đặc trưng của decoder)
Nhớ Lại: các kiến trúc sinh đều sử dụng tới kỹ thuật masking
(mask R-CNN, UNet cho segmentation)

softmax

Decoder-Only Transformer



Điểm khác biệt căn bản của Self-attention và Masked Self-attention

Self-attention có cái nhìn tổng quát hơn
với mỗi 1 token đang xét, nó quan tâm tới cả token
trước và sau nó



Masked self-attention chỉ quan tâm tới các token trước nó
tây là đặc trưng của bài toán sinh (đặc điểm vào để mà phải
tự hoàn thành rất cần)



Chat GPT là một decoder-only transformer
↓
Generative Pre-trained transformer

The matrix math for calculating masked self-attention

Friday, June 20, 2025 3:19 PM

$$\text{Masked Attention } (Q, K, V, M) = \text{Softmax} \left(\frac{QK^T}{\sqrt{dk}} + M \right) \cdot V$$

Vai trò và các bước tính của Q , K , V vẫn như trong self-attention

Ở đây, ta chỉ duy thêm 1 thành phần mask M

Mục đích: che đi các token mà token đang xét \in set phép nhín thay (chú ý tới)

Scaled Dot Product Similarities			Keys	Queries
write	a	poem		
write	-0.06	0.04	-0.43	
a	-0.28	0.29	-2.10	
poem	0.35	-0.50	2.91	

Mask			Keys	Queries
write	a	poem		
write	0	-inf	-inf	
a	0	0	-inf	
poem	0	0	0	

Masked Scaled Dot Product Similarities			Keys	Queries
write	a	poem		
write	-0.06	-inf	-inf	write
a	-0.28	0.29	-inf	a
poem	0.35	-0.50	2.91	poem

Sau khi qua softmax
các giá trị -inf thành 0
 \Rightarrow chú ý tới chúng

write a poem Keys			write	a	poem
1.00	0.00	0.00			
0.36	0.64	0.00			
0.07	0.03	0.90			

vẫn là ma trận similarity score

nhưng với từ "write", nó chỉ để nhìn thấy chính nó

"a", nhìn để "write và chính nó"

"poem", nhìn để tất cả

\Rightarrow cần che đi bớt các
giá trị trong similarity matrix

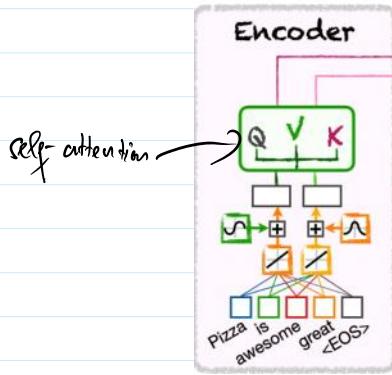
Encoder-only transformer: tạo contextual embedding từ input

Decoder-only transformer: tạo generative input để sinh chuỗi mới

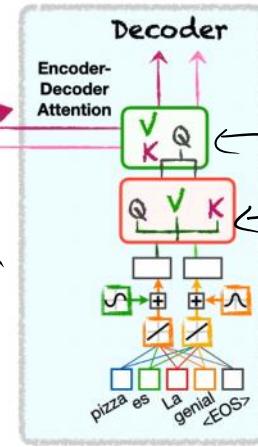
Nguồn gốc của những tên gọi này là từ kiến trúc transformer gốc:



And we have also learned about a 3rd type of attention, **Encoder-Decoder Attention**, which is also called **Cross-Attention**.



contextual embedding của encoder dc cung cấp để tính toán K, V cho decoder
Q dc tính từ đầu ra của decoder



designed
for machine
translation

Lưu ý về data: input text → [encoder] → contextual embedding

cross
attention

<bos> ... → [decoder] → Q(K, V) → new token
append to form new input.

Sau quá trình pt, cái tên → mỗi phần đều có nhiều ý nghĩa
→ tách riêng biệt ⇒ hình thành tên gọi

Sau này, các mô hình đa phương thức (multimodal model) cũng sử dụng kiến trúc encoder-decoder:

Multi-head attention

Friday, June 20, 2025 7:41 PM

Để nắm bắt ngữ cảnh ở phạm vi dài hơn, nhiều khía cạnh hơn \Rightarrow dùng nhiều đầu attention song song

\Rightarrow Multi-head attention

Bản chất là nhiều khía cạnh attention đó, thiết kế song song

Kết quả của các khía cạnh tổng hợp theo nhiều cách:

+ sử dụng 1 fully connected layer để tổng hợp

