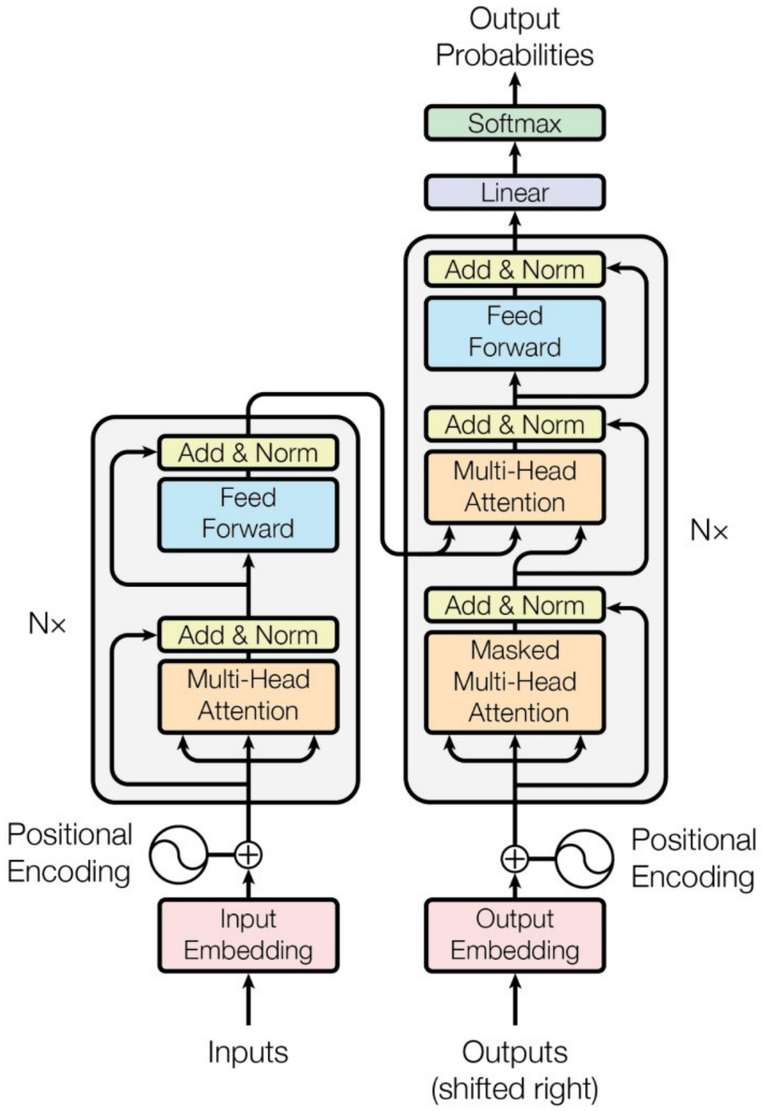


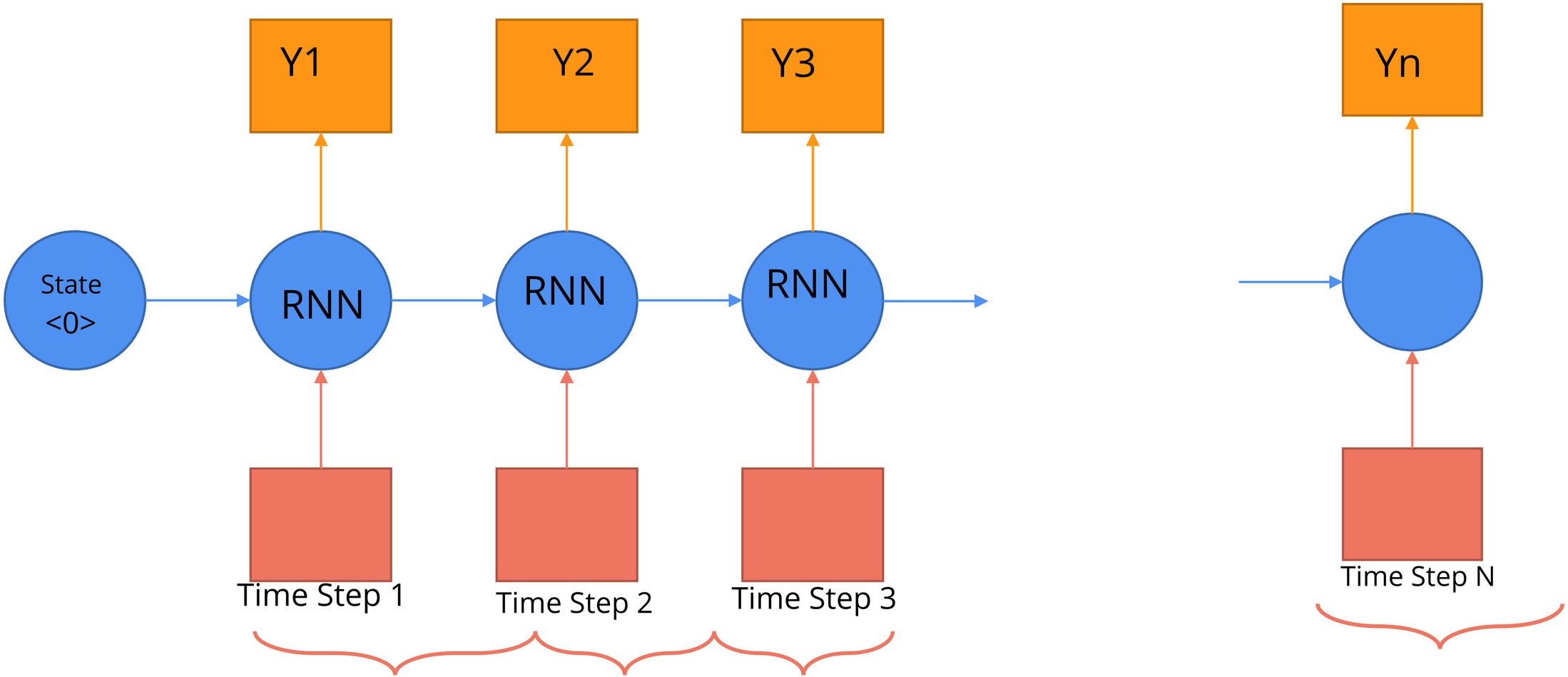
Transformer

Attention is all you need

DngBack

Paper: https://arxiv.org/pdf/1706.03762.pdf

Recurrent Neural Networks (RNN)



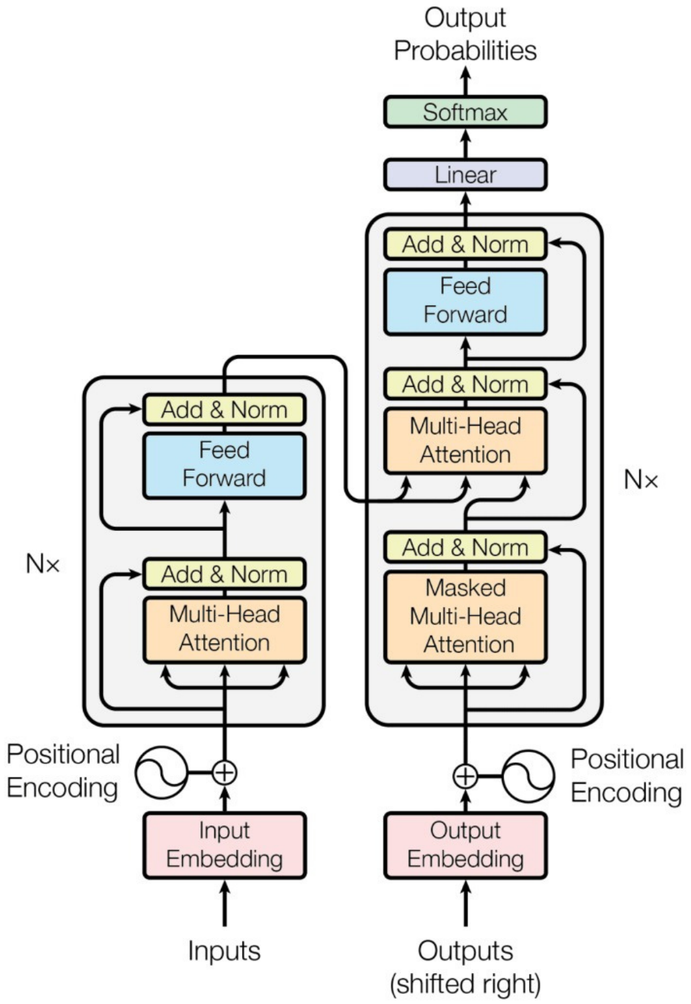
Vấn đề của RNN

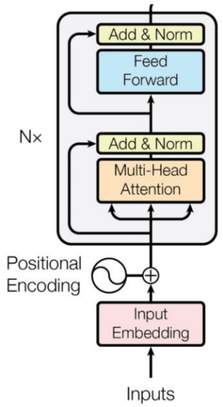
1. Tốc độ tính toán chậm với đoạn câu dài
2. Vấn đề về bùng nổ đạo hàm (Vanishing or exploding gradients)
3. Khó khăn trong việc truy cập thông tin có khoảng cách xa (Difficulty in accessing information from long time ago)

⟶ Có những sự thay đổi và cải tiến

- ⟶ LSTM
- ⟶ Attention
- ⟶ Transformer

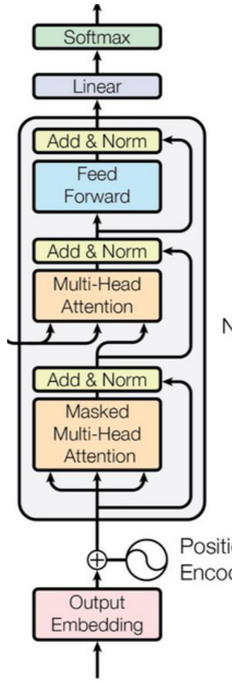
Transformer



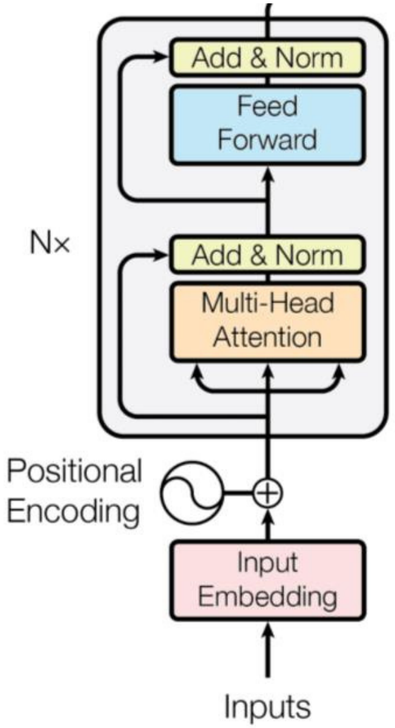


Encoder

Dencoder



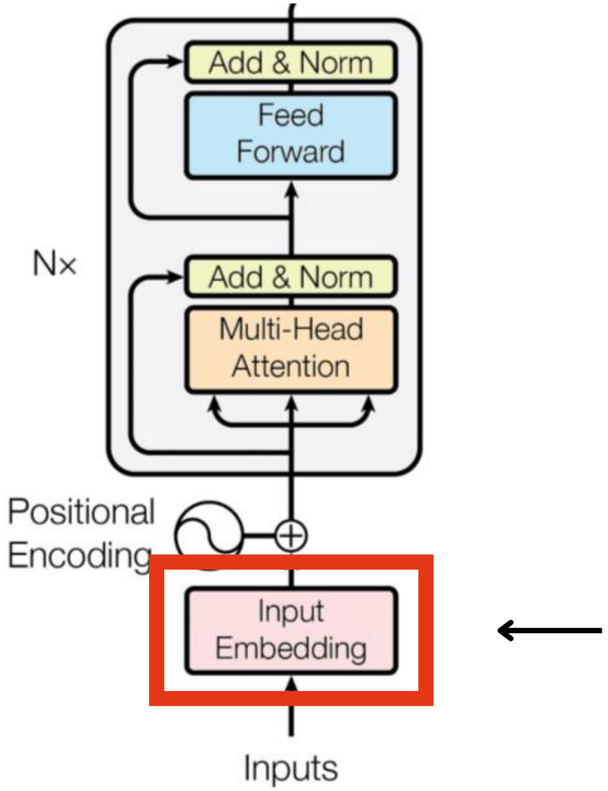
Encoder

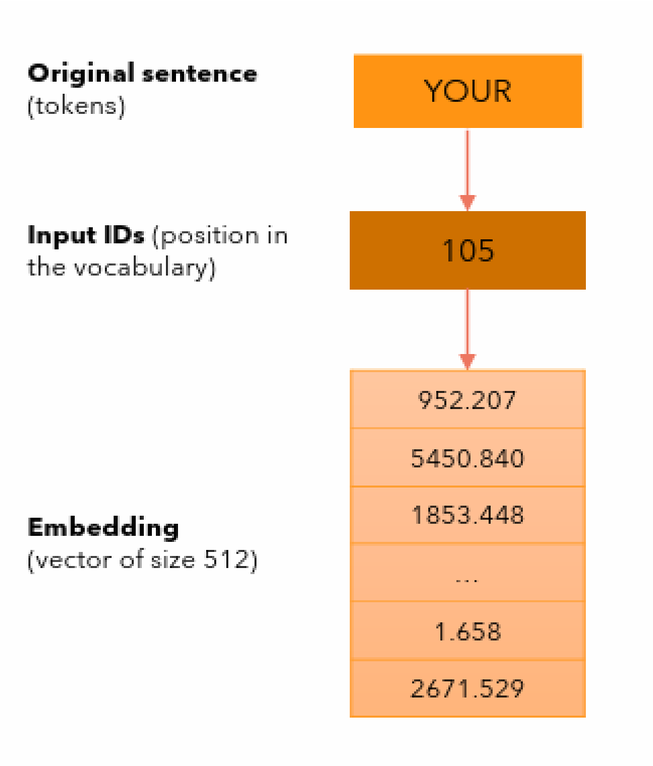


Encoder: The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection [11] around each of the two sub-layers, followed by layer normalization [1]. That is, the output of each sub-layer is $\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$, where $\mathrm{Sublayer}(x)$ is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{\text{model}} = 512$.

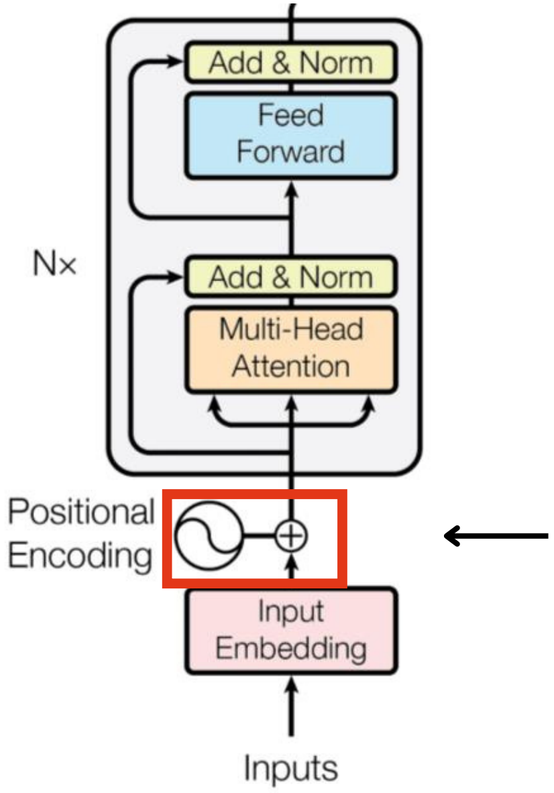
Input Embedding

Quá trình chuyển các từ và cụm từ trong dữ liệu đầu vào thành các vector số thực





Positional Encoding



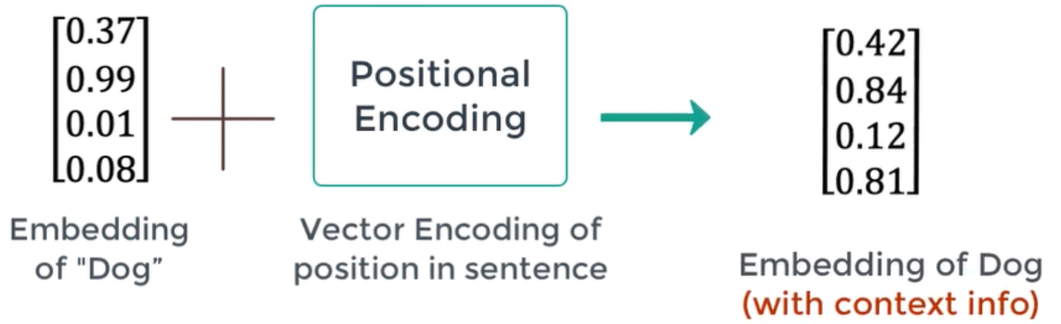
3.5 Positional Encoding

Since our model contains no recurrence and no convolution, in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence. To this end, we add "positional encodings" to the input embeddings at the bottoms of the encoder and decoder stacks. The positional encodings have the same dimension d_{model} as the embeddings, so that the two can be summed. There are many choices of positional encodings, learned and fixed [9].

In this work, we use sine and cosine functions of different frequencies:

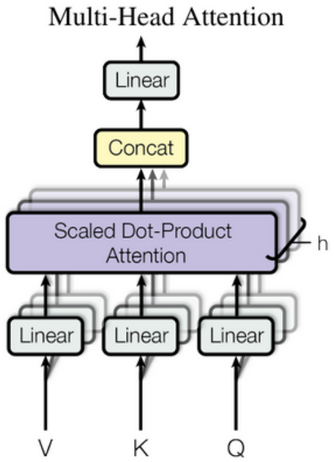
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

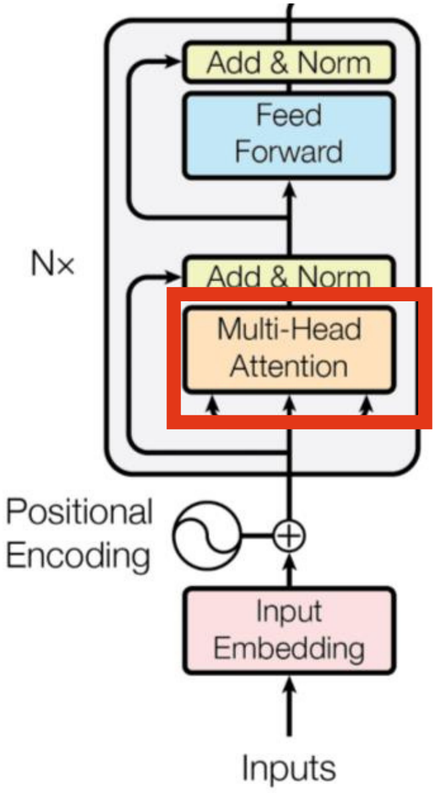
where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.



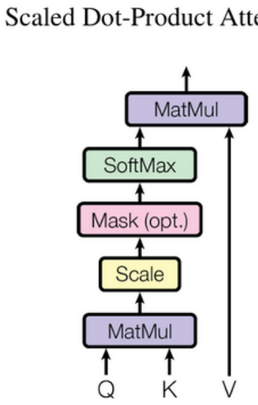
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

MultiHead Attention

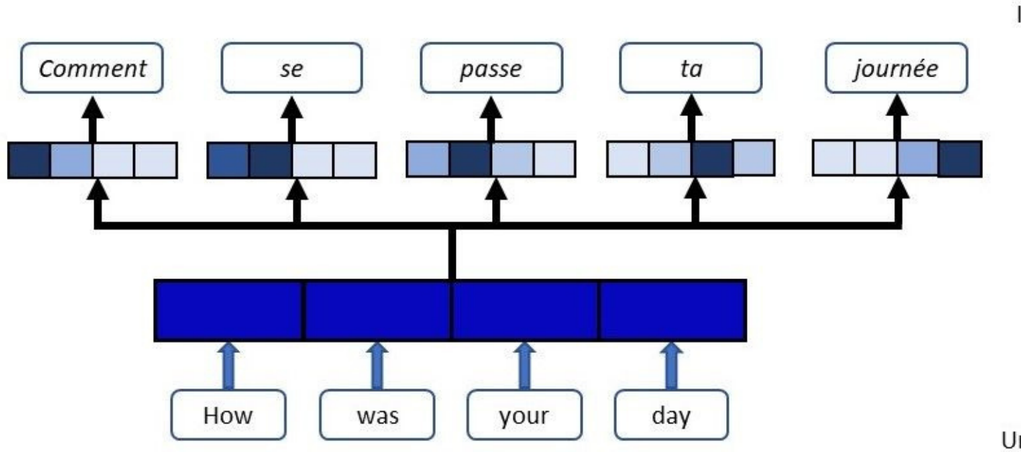




Self Attention



Attention



Self Attention

Each vector receives three representations ("roles")

$W_Q \times$ = **Query**: vector **from** which the attention is looking

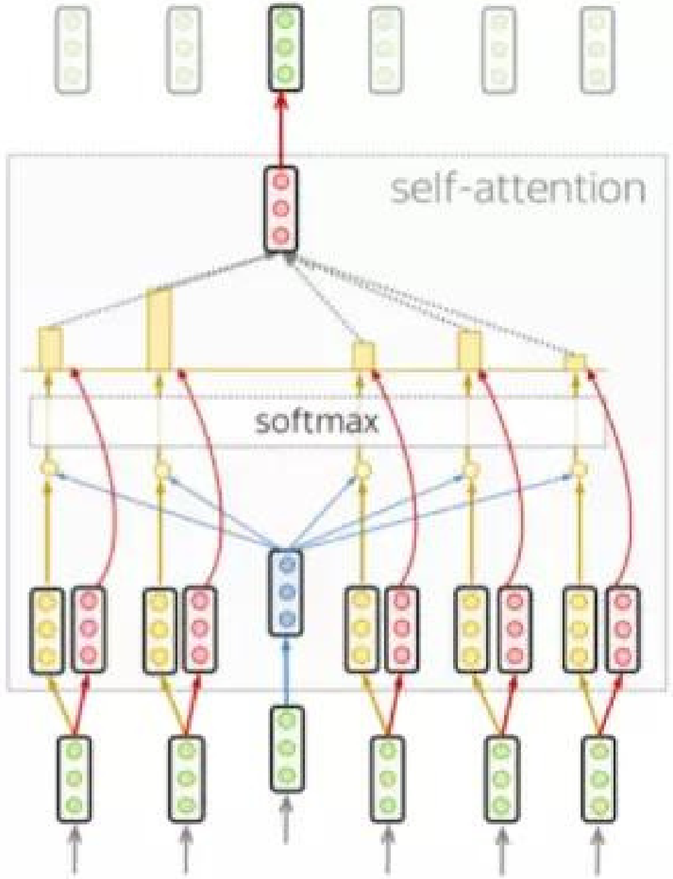
"Hey there, do you have this information?"

$W_K \times$ = **Key**: vector **at** which the query looks to compute weights

"Hi, I have this information - give me a large weight!"

$W_V \times$ = **Value**: their weighted sum is attention output

"Here's the information I have!"

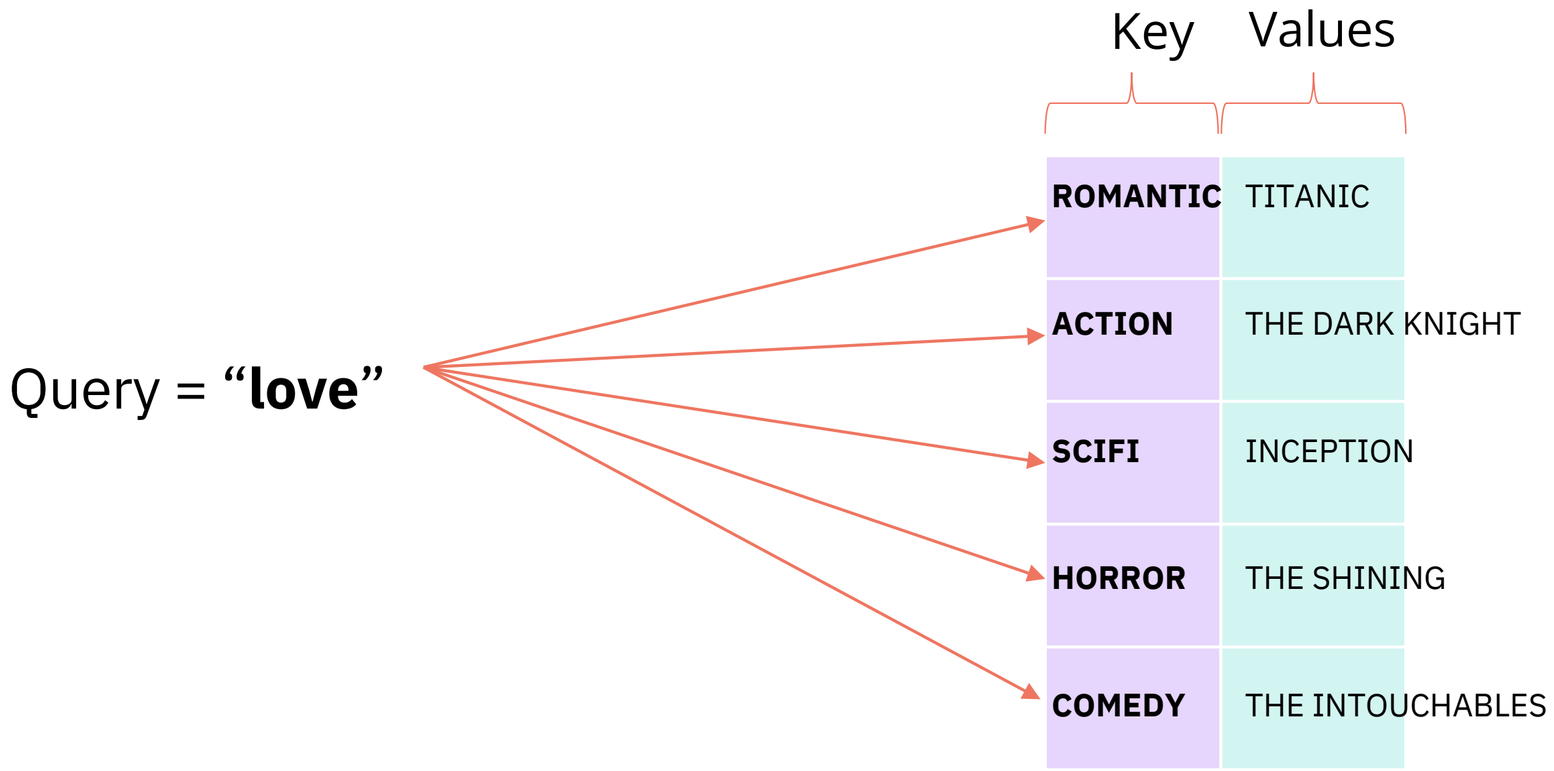


Self Attention

Query ⟶ Đại diện cho thông tin hiện tại mà mô hình đang xử lý

Key ⟶ Đại diện cho thông tin về các thành phần khác trong chuỗi đầu vào

Values ⟶ Đại diện cho thông tin bổ sung của Query



Self Attention

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

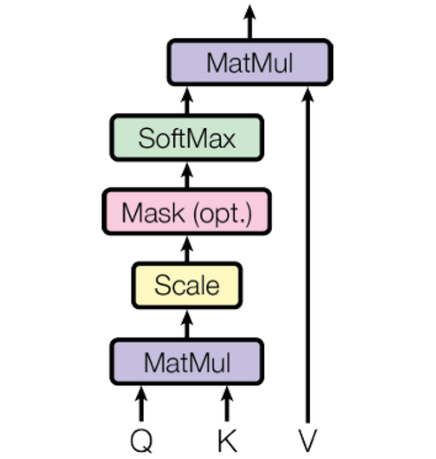
QK^T ⟶ Thể hiện mối tương quan của từ đang xét với các thành phần khác trong câu

$\sqrt{d_k}$ ⟶ Scaling factor

softmax ⟶ Chuẩn hóa trọng số attention về khoảng (0,1)

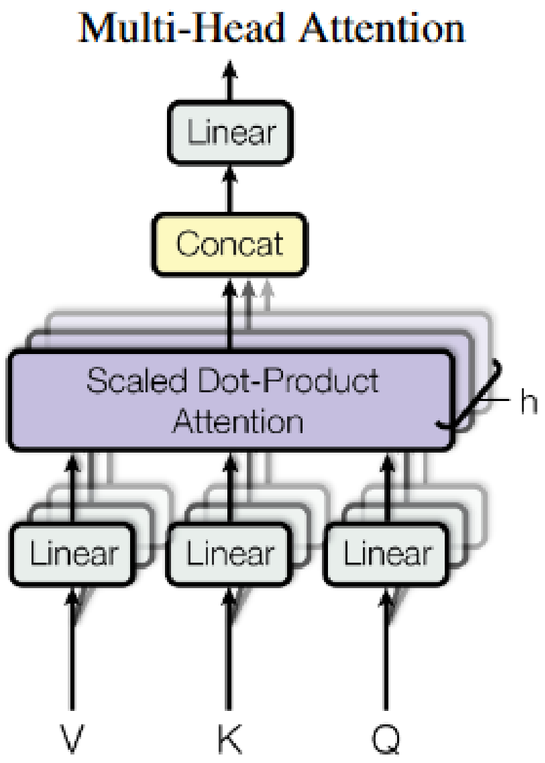
V ⟶ Lọc các giá trị attention quan trọng

⟶ Cho ra một ma trận Attention trong một câu



Multihead Attention

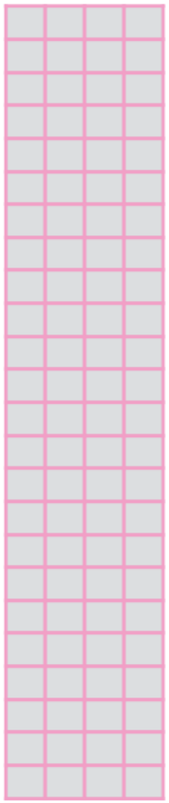
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



1) Concatenate all the attention heads

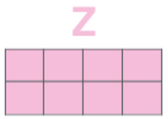
2) Multiply with a weight matrix W^O that was trained jointly with the model

X

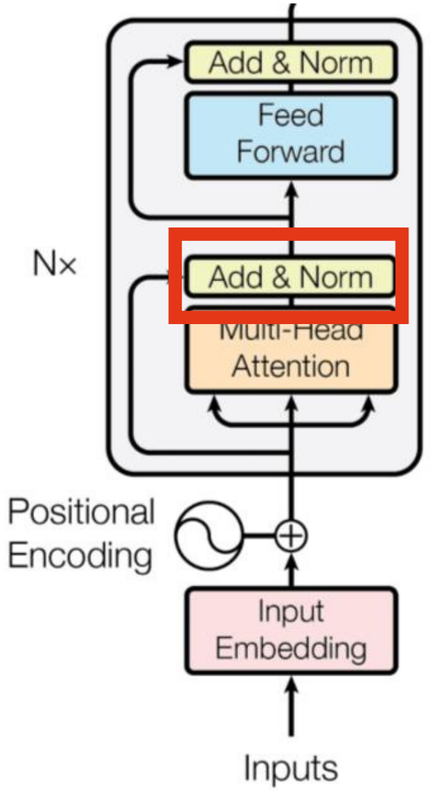


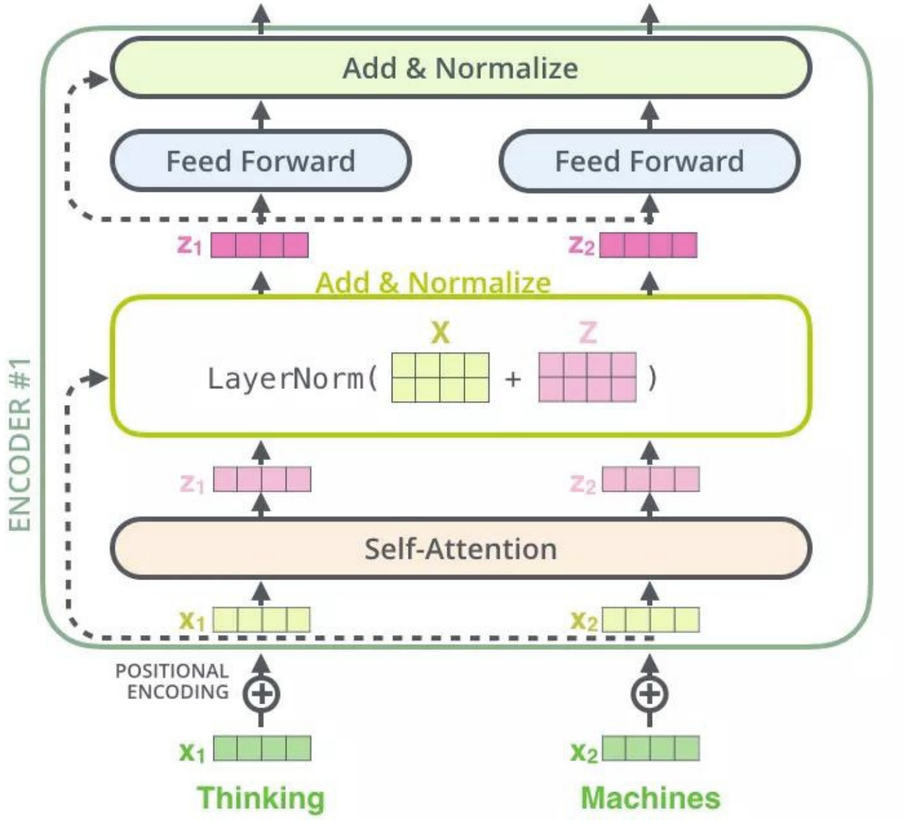
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=



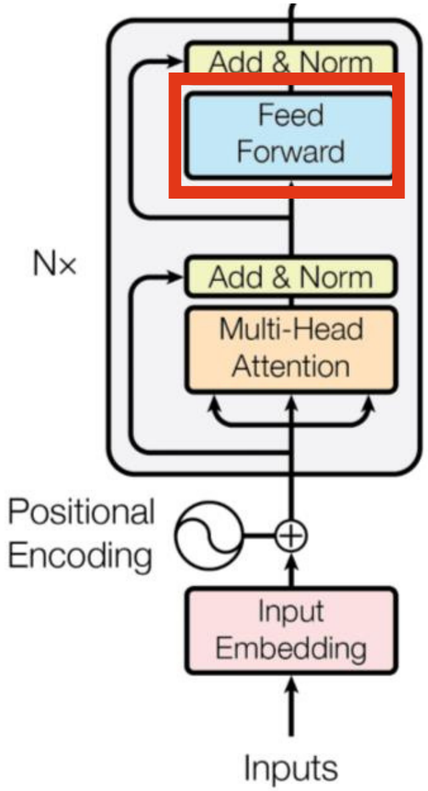
Residuals





- Skip Connection
- Layer Normalzation

Feed Forward

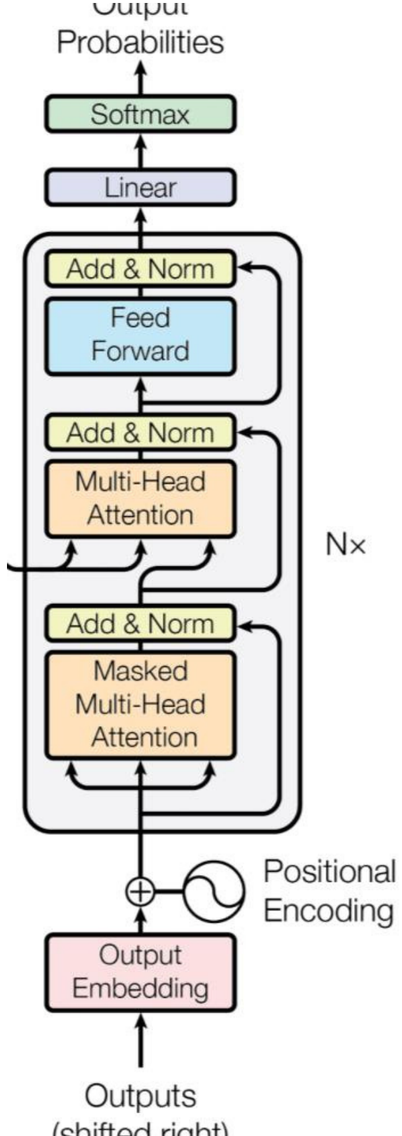


In addition to attention sub-layers, each of the layers in our encoder and decoder contains a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a ReLU activation in between.

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. Another way of describing this is as two convolutions with kernel size 1. The dimensionality of input and output is $d_{\text{model}} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$.

Decoder

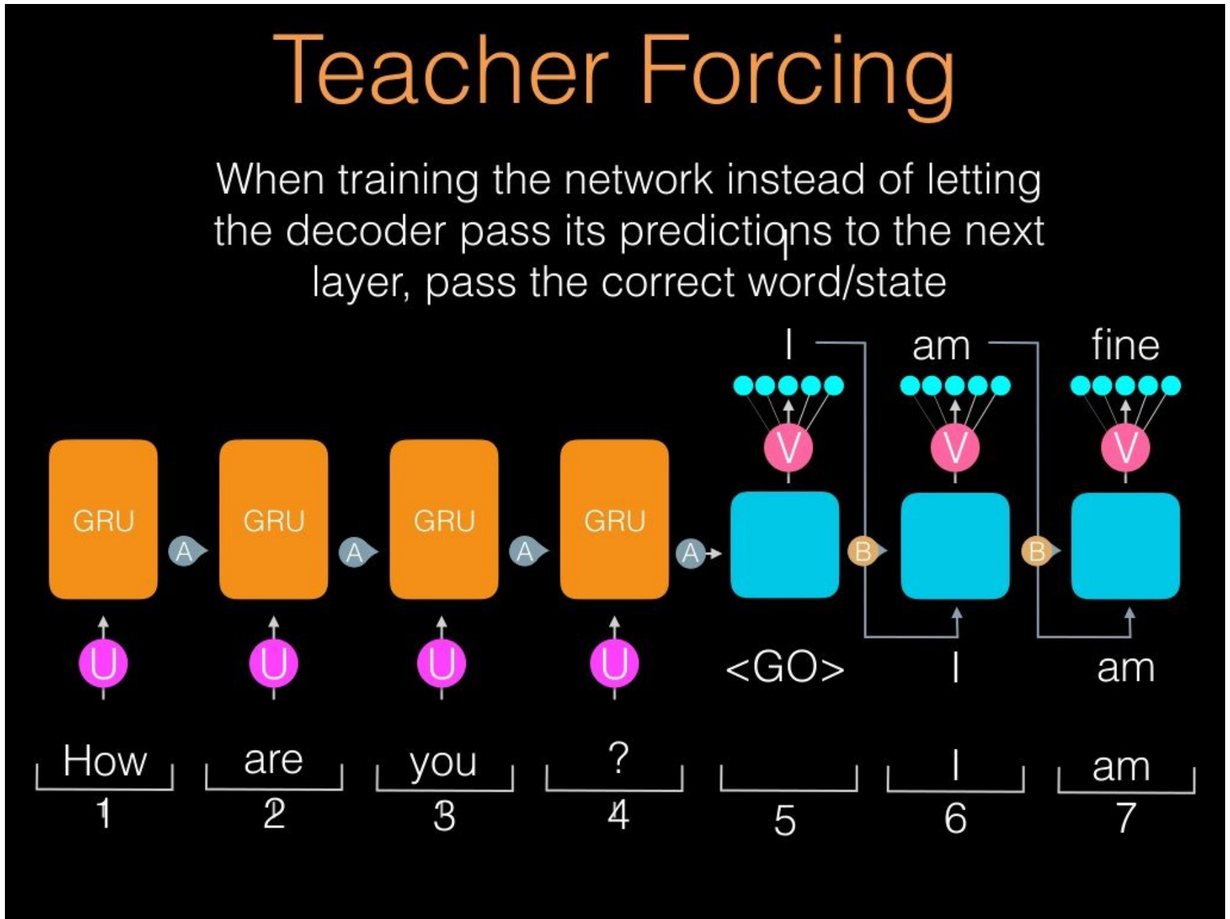


Decoder: The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i.

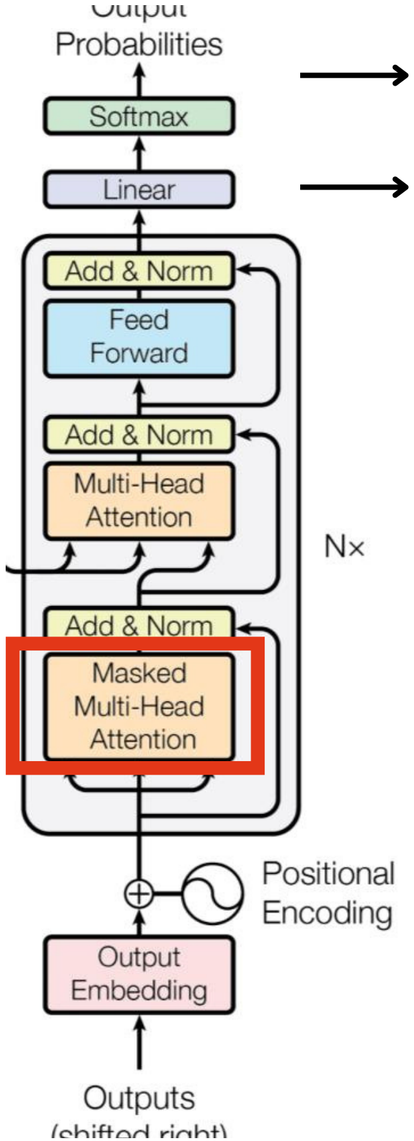
Teacher Forcing

⟶ Kỹ thuật training cho bài toán AutoRegressive

⟶ Sử dụng groundth-truth để dẫn đường cho mô hình học hiệu quả



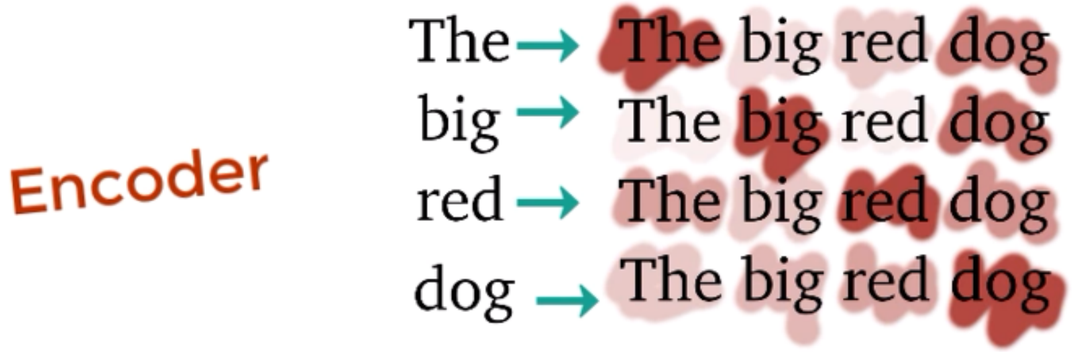
Masked Multihead Attention

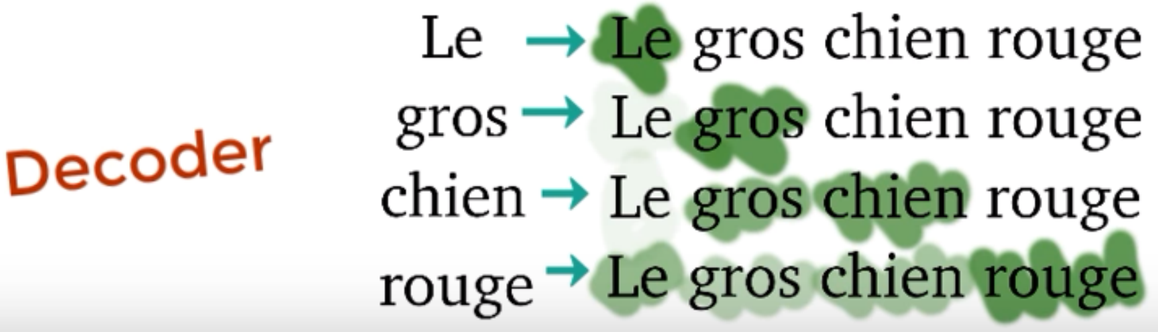


⟶ Sự kết hợp của self-attention và causal masking

⟶ Tập trung vào những token hiện tại và quá khứ

⟶ Idea: Sử dụng một ma trận vuông với những thành phần nằm trên đường chéo chính có giá trị âm vô cùng, các thành phần khác có giá trị là 0





Reference

- https://www.youtube.com/watch?v=bCz4OMemCcA
- https://www.youtube.com/watch?v=TQQlZhbC5ps
- https://viblo.asia/p/transformers-nguoi-may-bien-hinh-bien-doi-the-gioi-nlp-924lJPOXKPM
- http://jalammar.github.io/illustrated-transformer/

Thanks for watching!