Review of lecture 04.

D. Encoder-Decoder model

RNN: $x_1 \to h_1$, $x_2 \to h_2$, ... $x_n \to h_n$,

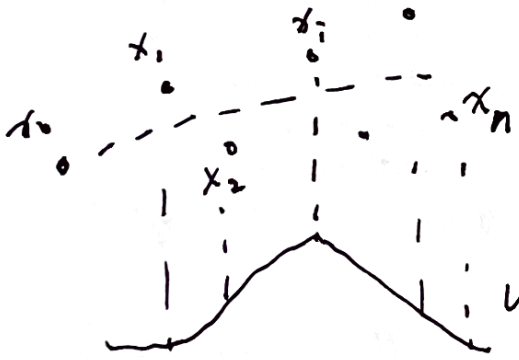
$\tilde{c} = h_n \to$ decoder.

RNN + Attention: $x_1 \to h_1$, $x_2 \to h_2$, $\to h_n$

$w_1 h_1 + h_2 w_2 + \dots h_n \cdot w_n \to \tilde{c}$.

new context: $[\tilde{c}, h_{n+1}]$ concatinate. → output.

② Intuition of Attention: Re-weighting : get more and better context.



$x = [x_0, x_1, \dots, x_n]$.

$W_i = [W_{0i}, W_{1i}, \dots, W_{ni}]$

$y = [y_0, y_1, \dots, y_n]$.

$y = W \cdot x$, $W = [w_0, w_1, \dots w_n]$.

A good W should smooth the noise of x to get a good y.

③. Noa can be annoying but she is a great cat.

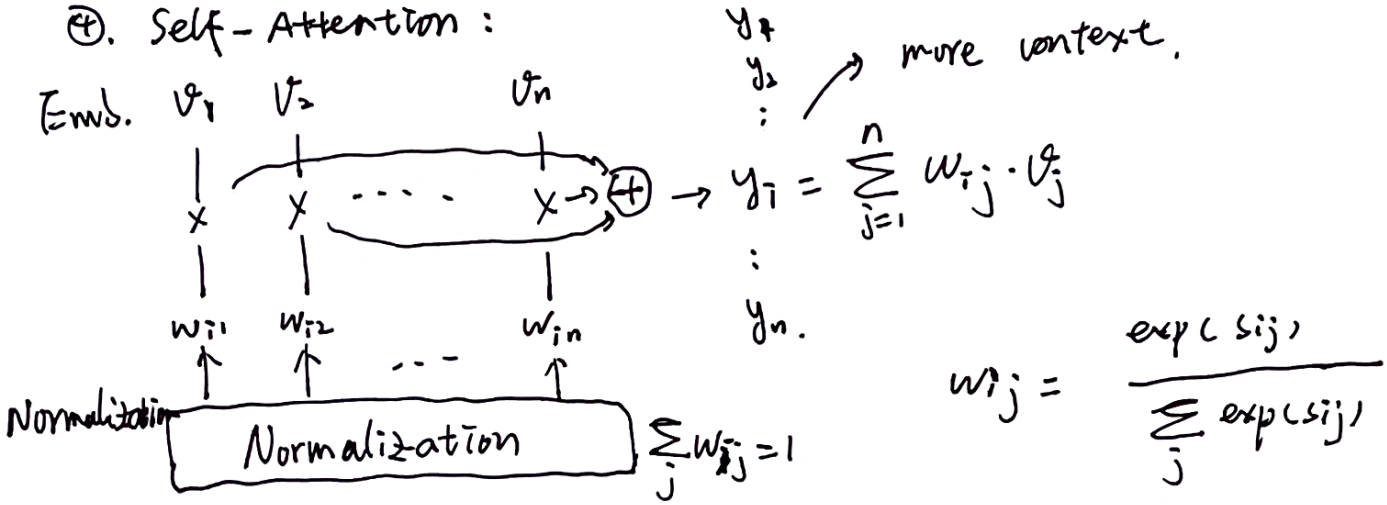
	v_1	v_2	- - -		v_{10}
Re-weighting	w_1	w_2			.. w_{10}.
	↓	↓			↓
	y_1	y_2			y_{10}.

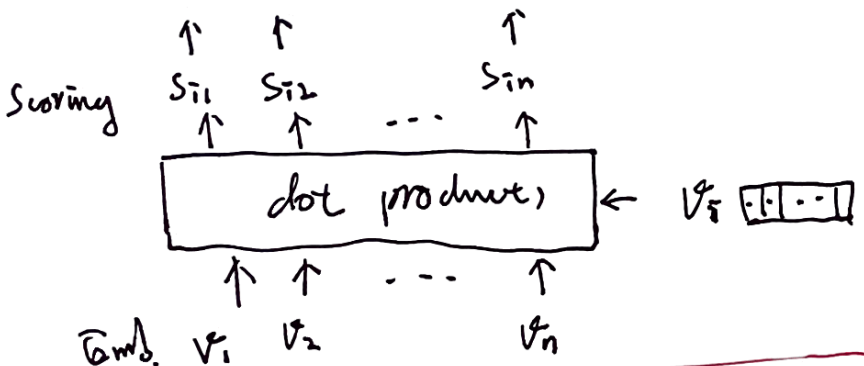
$y_{1:10}$: is more and better context conveyed with $v_{1:10}$.

$$y_i = w_{i1} \cdot v_1 + w_{i2} \cdot v_2 + \cdots w_{in} \cdot v_n.$$

How to design w_{ij}? How to introduce model weights?

⑪. Self-Attention:

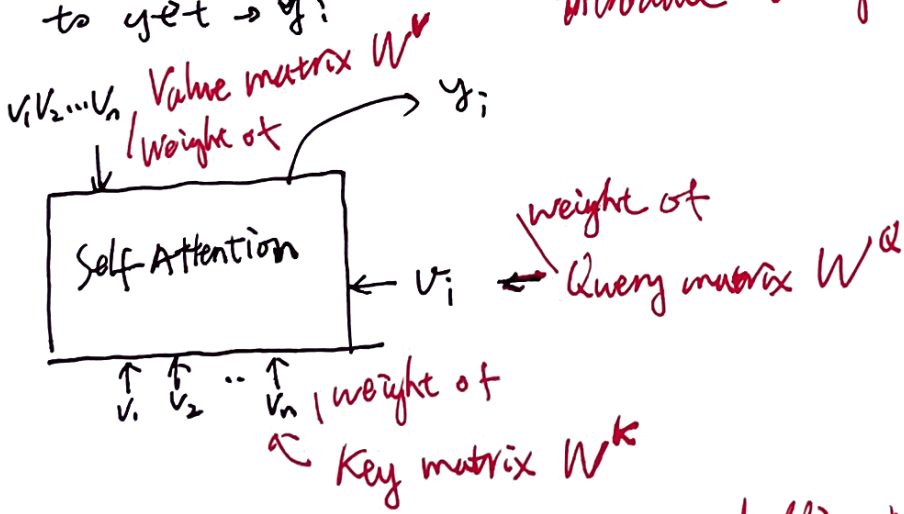




---

⑫-1. to get y_i:

introduce weights.



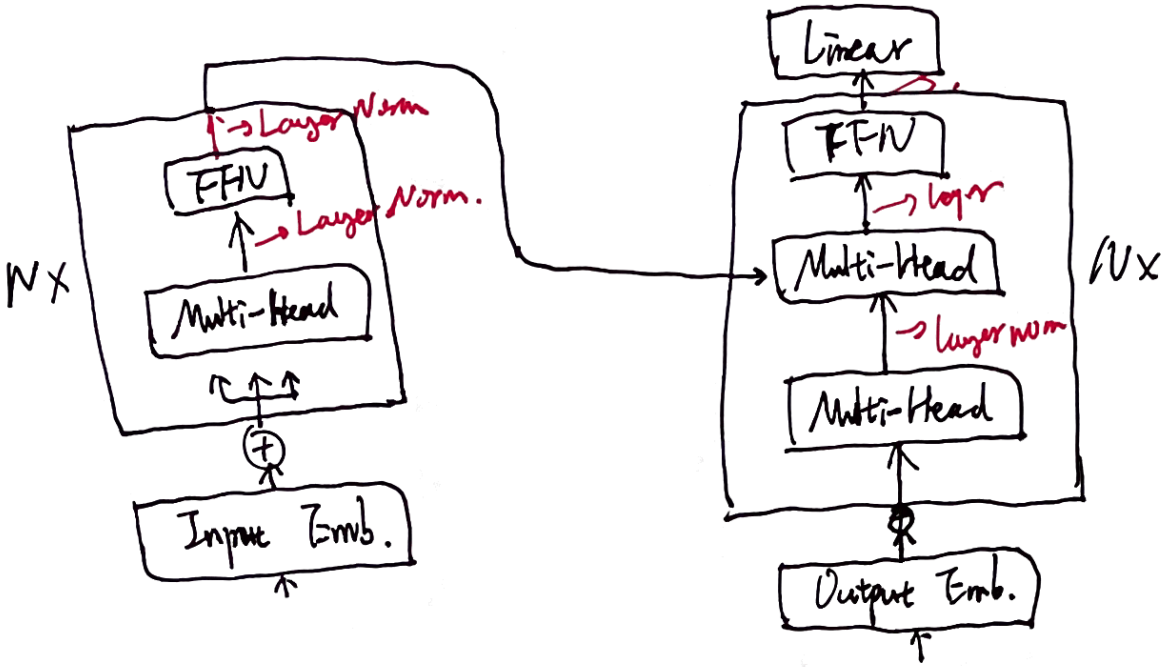
Let $X = [v_1\ v_2 \cdots v_n]$ is the input embedding matrix.

Query matrix: $Q = X \cdot W^Q$

Key matrix: $K = X \cdot W^K$

Value matrix: $V = X \cdot W^V$.

- Each col is a query/key/value vector.
- Time complexity: $O(n^2 \cdot d)$.



Transformer

Model configeration:

N: number of layers

d_{model}: dimension of output layers

d_{ff}: inner layer dimension of FFN

h: # heads

d_k: queries and keys

d_v: values

Total model size:

Total = A1 + A2

A1 = Input Emb. + Output Emb. + Linear

A2 = N x Encoder + N x Decoder.

A1: all three weight matrix share same weight. (see 3.4 Embeddings and Softmax).

$W^{emb} \in \mathbb{R}^{|V| \times d_{model}}$

A2 = N x Encoder + N x Decoder,

Encoder = Multihead + FFN., Decoder = 2 x Multihead + FFN.

Multihead = $(|W_i^Q| + |W_i^K| + |W_i^V|) \cdot h + |W^O|$ + 2 · layernorm

$W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$.

$W^O \in \mathbb{R}^{hxd_v \times d_{model}}$

$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$

$W_1 \in \mathbb{R}^{d_{model} \times d_{ff}}$, $b_1 \in \mathbb{R}^{d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff} \times d_{model}}$, $b_2 \in \mathbb{R}^{d_{model}}$.

En-De: $|V| = 37000$,

$N = 6$, $d_{model} = 512$, $d_{ff} = 2048$, $h = 8$, $d_k = 64$, $d_v = 64$

Total = A1 + A2. $\quad h \times d_v = 8 \times 64 = 512$

$A1 = |V| \times d_{model} = 37000 \times 512$

A2 = N × Encoder + N × Decoder

$Encoder = h \cdot (|W_i^Q| + |W_i^K| + |W_i^V|) + |W^O| + FFN$

$= 8 \cdot (512 \times 64 + 512 \times 64 + 512 \times 64) + 512 \times 512 + FFN$

$= 4 \cdot 512 \times 512 + FFN$

$FFN = 2 \cdot d_{model} \times d_{ff} + d_{ff} + d_{model}$

$= 2 \cdot 512 \times 2048 + 2048 + 512$

$= 2 \cdot 1024^2 + 2 \cdot 1024 + 512$

$Encoder = 1024^2 + 2 \cdot 1024^2 + 2 \cdot 1024 + 512$

$Decoder = 2 \cdot 1024^2 + 2 \cdot 1024^2 + 2 \cdot 1024 + 512$

$Total = 37000 \times 512 + 6 \cdot (1024^2 \times 3 + 2 \cdot 1024 + 512)$

$+ 6 \cdot (1024^2 \times 4 + 2 \cdot 1024 + 512)$

$37000 \approx 18.066406025 \times 1024 \times 2$

$18 + 18 + 24 \approx 60.$

$\approx 37000 \times 512 + 42 \times 1024^2 + 30 \times 1024$

$= 63,014,912 \quad 63M.$

Use the following:

$T_{total} = |V| \cdot d_{model} + N \cdot [B_1 + B_2]$

$B_1 = (2 \cdot d_{model} \times d_k + d_{model} \cdot d_v) \cdot h + h \cdot d_v \cdot d_{model} + 2 d_{model} \cdot d_{ff} + d_{model} + d_{ff}$

$= (2 \cdot d_{model} \cdot d_k + d_{model} \cdot d_v) \cdot 2h + 2h \cdot d_v d_{model} + 2 \cdot d_{model} \cdot d_{ff} + d_{model} + d_{ff}$

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

For each q_i: $q_i \cdot K^T$. → we only need to consider small subset of closest keys. 32, 64.

Locality sensitive hashing:

Idea of random projection:  → false.

Given n points $P \subset \mathbb{R}^d$, for a given $q \in \mathbb{R}^d$, find the point $p \in P$ that is closest to q. $O(n^\rho)$. $\rho < 1$.

Linformer:

$X \cdot W_i^Q = Q$,

$\Pr(\|\tilde{P}W^T - PW^T\| \le \varepsilon \cdot \|P \cdot W^T\|) \ge 1 - O(1)$.

$\text{rank}(P) = \theta(\log(n))$.

Remark: ① layer normalization:

$$y = r \cdot \frac{x - E(x)}{\sqrt{\text{Var}(x) + \varepsilon}} + \beta.$$

r: weights.

β: bias.

$r \in \mathbb{R}^{d_{\text{model}}}$, $\beta \in \mathbb{R}^{d_{\text{model}}}$

Encoder = $N \times 2 \times d_{\text{model}}$

Decoder = $N \times 3 \times d_{\text{model}}$.

Time complexity: $O(n^2 \cdot d + n \cdot d^2)$

$w_1, w_2, \dots, w_n$:

weighted sum of all words.