S_1: Hello world haha

S_2: How old are you

S_3: Hey you

d_model = 4 , max_len = 4

S_1 embedding :

$$\begin{array}{l} \\ \text{hello} \\ \text{world} \\ \text{haha} \\ \text{<\cdot>} \end{array}
\overset{\xleftrightarrow{\text{d_model}}}{\begin{bmatrix} 1 & 3 & 4 & 1 \\ 3 & 2 & 1 & 0 \\ 4 & 5 & 7 & 10 \\ -1 & -1 & -1 & -1 \end{bmatrix}}$$

S_2: embedding:

$$\begin{array}{l} \text{how} \\ \text{old} \\ \text{are} \\ \text{you} \end{array}
\begin{bmatrix} 1 & 4 & 6 & 1 \\ 3 & 1 & 5 & 4 \\ 1 & 10 & 20 & 10 \\ 1 & 2 & 0 & 1 \end{bmatrix}$$

S_3 embedding :

$$\begin{array}{l} \text{Hey} \\ \text{you} \\ \text{<\cdot>} \\ \text{<\cdot>} \end{array}
\begin{bmatrix} 3 & 1 & 4 & 5 \\ 1 & 2 & 0 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

Positional encoding:

$$PE_{(pos, 2i)} = \sin\left(\frac{pos}{1000^{2i/d\_model}}\right)$$

pos : [0, 1, 2, 3] according to max_len

i : 0 to 3

$PE_{(pos, i=0)}$: [sin(0), sin(0), sin(0), sin(0)

$PE(_{pos, i=1})$ = [cos(·), cos(·), cos(·), cos(·)]

$PE(pos, i=2)$ = [sin(0), sin(0), sin(·), sin(·)]

$$\text{d-model}$$

$$EPE_1 = \begin{array}{l} \text{hello} \\ \text{world} \\ \text{haha} \\ \langle\cdot\rangle \end{array} \begin{bmatrix} 1.1 & 3.2 & 4.1 & 10 \\ 3.2 & 2.1 & 1.2 & 0.1 \\ 4.1 & 5.5 & 7.1 & 10 \\ -1 & -1 & -0 & -0 \end{bmatrix}$$

$$EPE_2 : \begin{array}{l} \text{how} \\ \text{old} \\ \text{are} \\ \text{you} \end{array} \begin{bmatrix} 1 & 4 & 6 & 1 \\ 3 & 1 & 5 & 4 \\ 1 & 10 & 20 & 10 \\ 1 & 2 & 0 & 1 \end{bmatrix}$$

max len

$$EPE_3 : \begin{array}{l} \text{Hey} \\ \text{you} \\ \langle\cdot\rangle \\ \langle\cdot\rangle \end{array} \begin{bmatrix} 3 & 1 & 4 & 5 \\ 1 & 2 & 0 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

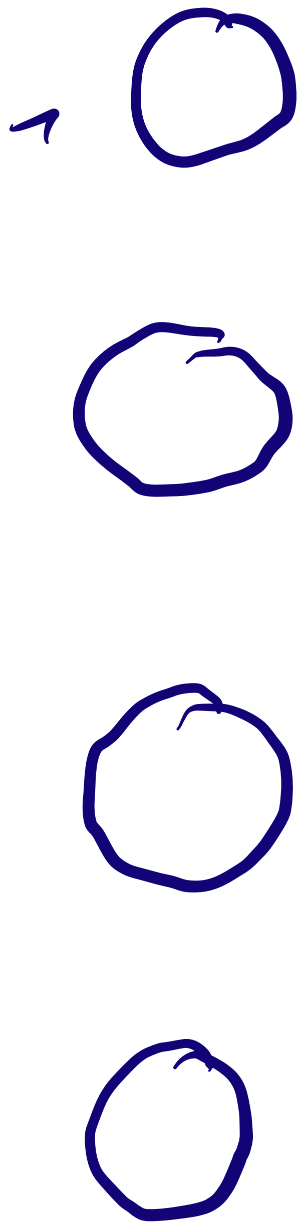
3

data shape: (3, 4, 4) // samples

samples

Feed-Forward :

token embbeding siz n



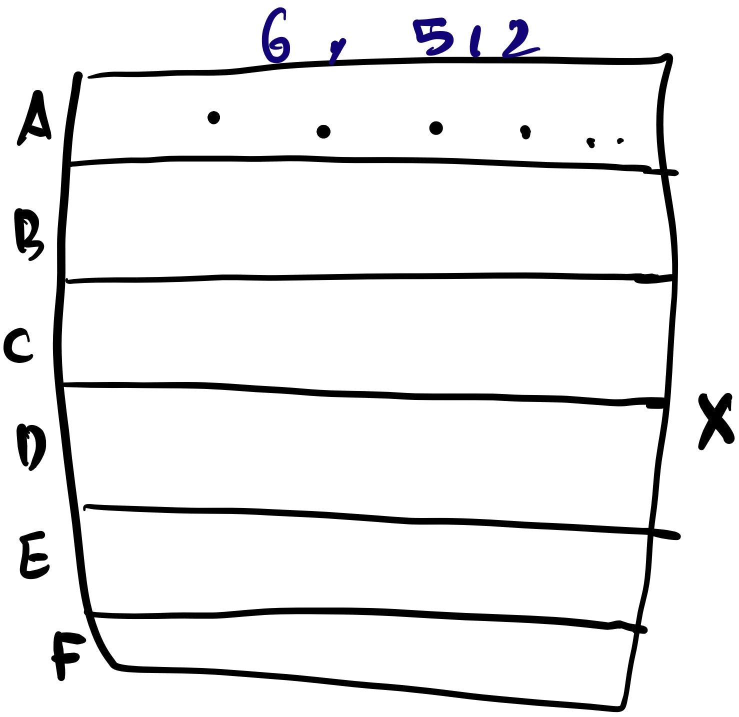
$x \in$

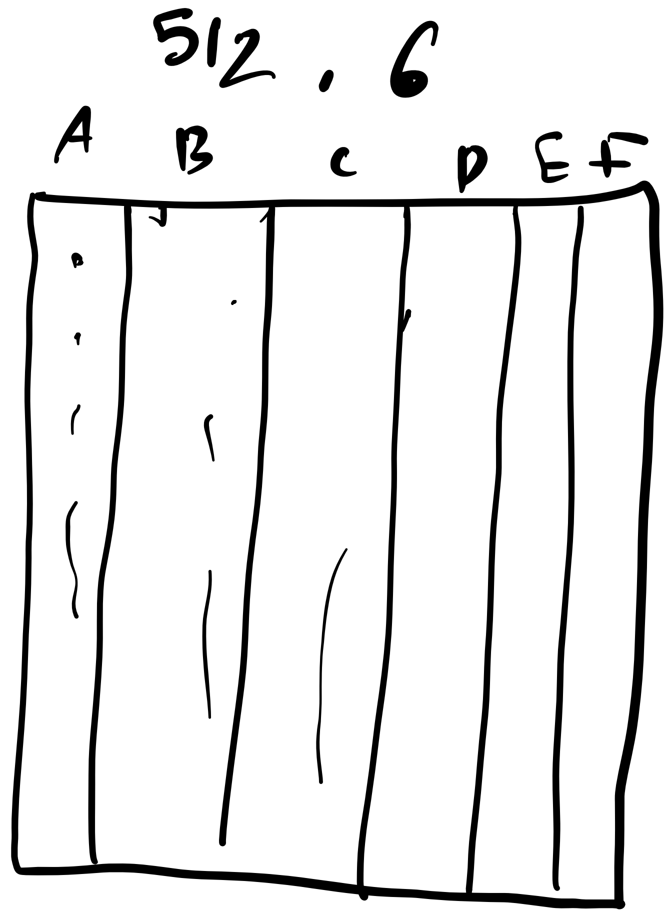
$xW + b$

d-model = 16

d-ff = 64

input matrix
of words





dot product = $\sum_{i=1}^{n} x_i y_i$

6 , 6

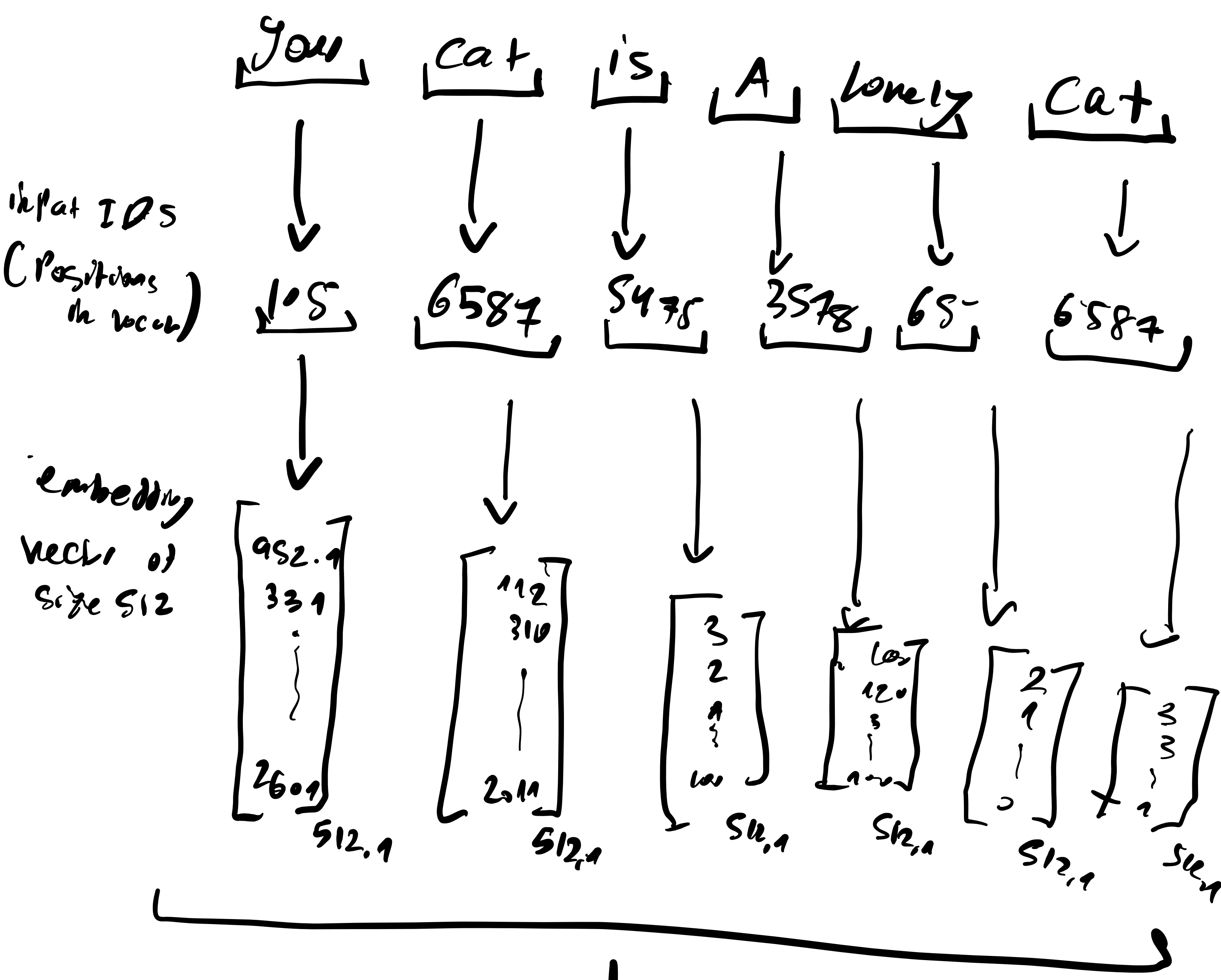
=

A·A	A·B	A·C			A·F
B·A	B·B	B·C			

Encoder:

Start with word embeddings

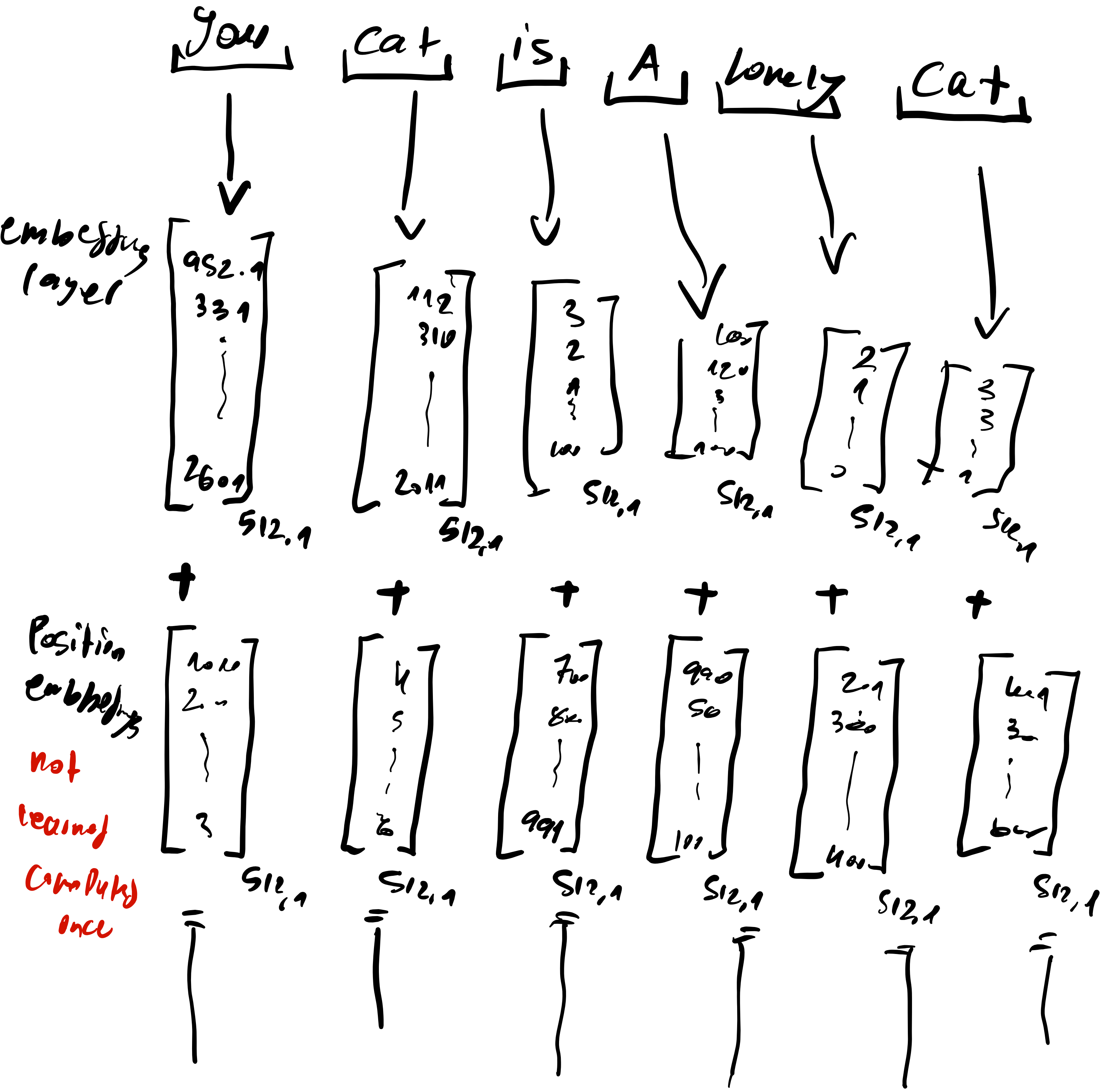
tokenize:

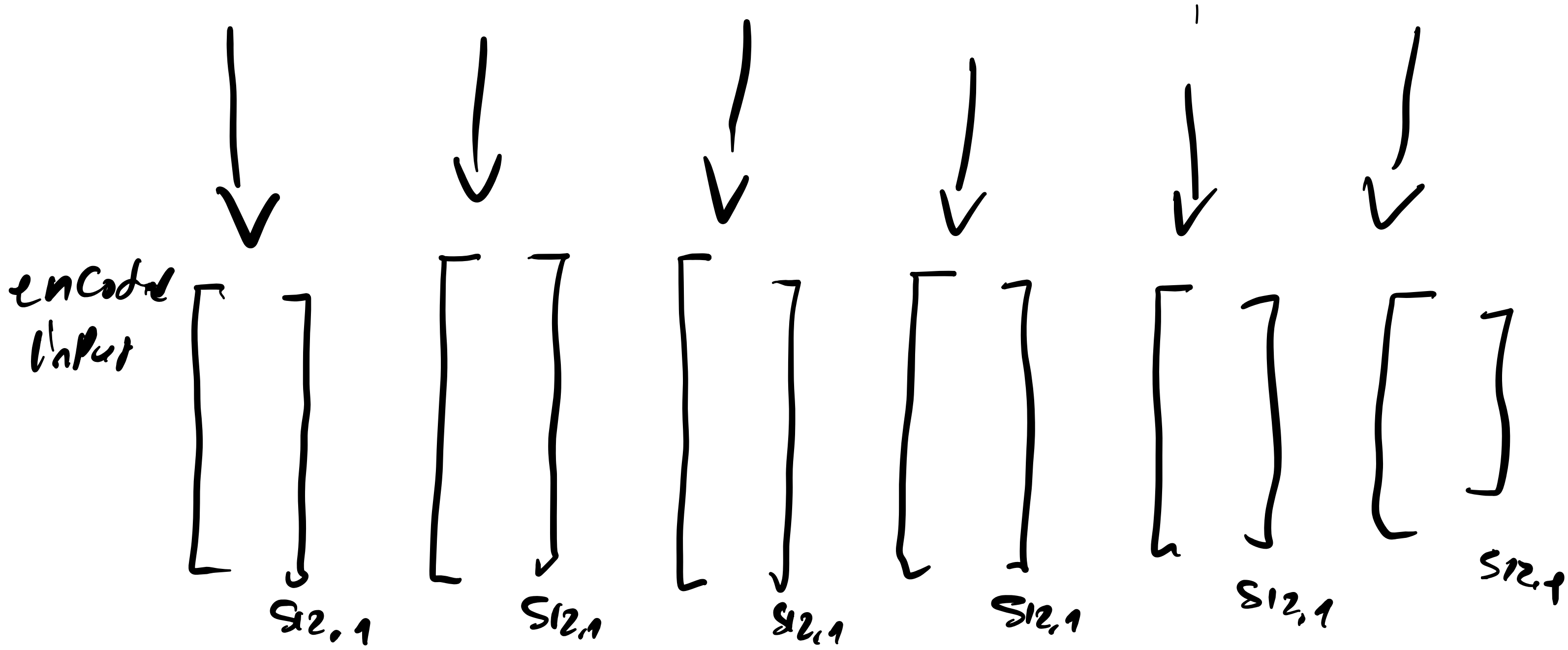


embedding layer

Positional encoding

Carry information about the position of the words





How to calculate positional encodings?

assume input text:

your cat is

$$
\begin{bmatrix} PE(0,0) \\ PE(0,1) \\ PE(0,2) \\ \vdots \\ PE(0,510) \\ PE(0,511) \end{bmatrix}
\quad
\begin{bmatrix} PE(1,0) \\ PE(1,1) \\ PE(1,2) \\ \vdots \\ PE(1,510) \\ PE(1,511) \end{bmatrix}
\quad
\begin{bmatrix} PE(2,0) \\ PE(2,1) \\ PE(2,2) \\ \vdots \\ PE(2,510) \\ PE(2,511) \end{bmatrix}
$$

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d\_model}}}\right) : \text{even}$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d\_model}}}\right) : \text{odd}$$

every other sentence will have the same positional encodings

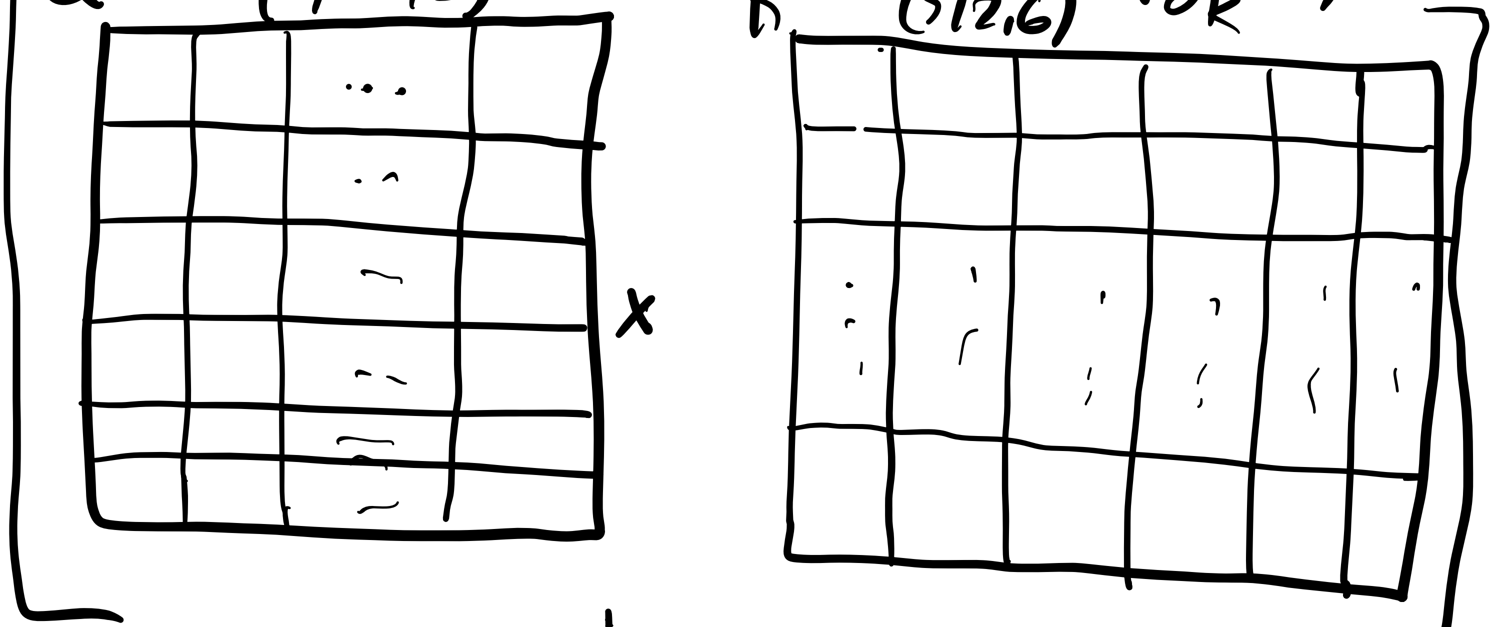
Self_attention

allows to relate words to each other.

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

soft max

Q $(6, 512)$ $\times$ K^T $(512, 6)$



$d_k = 512$

(6x6) matrix

$\sum = 1$

	your	cat	is	a	lovely	cat
your	0.268	0.119	0.134	0.148	0.179	0.152
cat	0.124	0.278	0.201	0.128	0.154	0.115
is	0.147	0.132	0.262	0.097	0.218	0.145
a	0.210	0.128	0.206	0.212	0.119	0.125
lovely	0.146	0.158	0.152	0.143	0.227	0.174
cat	0.195	0.114	0.203	0.203	0.157	0.229

the values represent how intense the relation between words

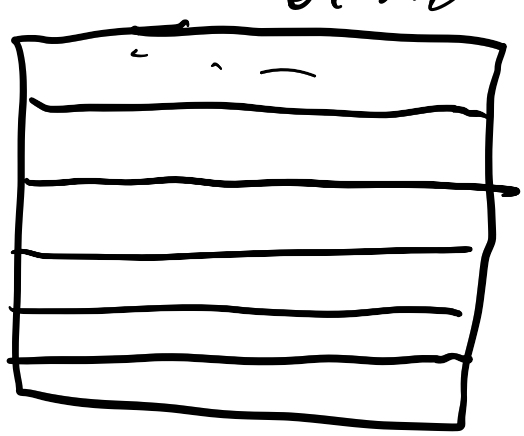
(6x6) matrix

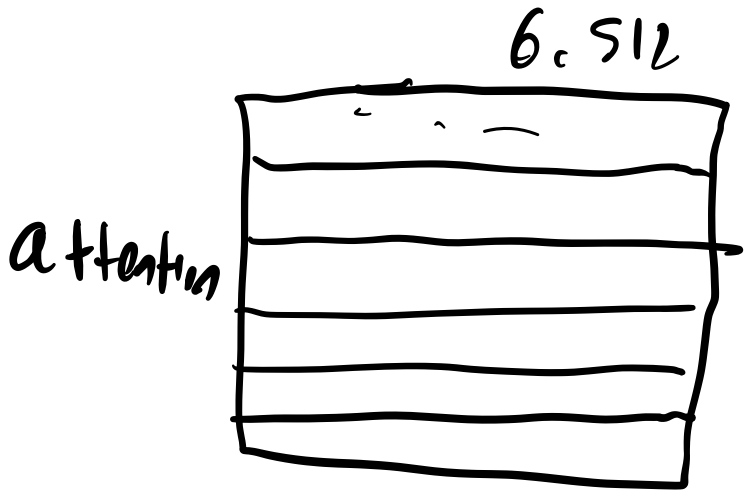
	you	cat	is	a	lovely	cat
you	0.268	0.119	0.134	0.148	0.179	0.152
cat	0.124	0.278	0.21	0.128	0.154	0.115
is	0.147	0.132	0.262	0.097	0.218	0.145
a	0.210	0.128	0.206	0.212	0.119	0.125
lovely	0.146	0.158	0.152	0.143	0.227	0.174
cat	0.195	0.114	0.203	0.203	0.157	0.229

$\Sigma = 1$

X

6.512

V  =



6 c 512
attention

$$\text{embedding} = \left[\begin{array}{l} [[1,2,1],\ [3,1,2],\ [3,1,3]] \\ [[2,1,0],\ [3,1,1],\ [2,1,0]] \\ [[3,0,0],\ [4,1,2],\ [3,3,3]] \\ [[1,0,1],\ [2,3,7],\ [7,1,0]] \end{array} \right]_{4,3,3}$$

$$PE = \left[\begin{array}{l} [[1,0,0],\ [2,1,3],\ [4,1,3]] \end{array} \right]_{1,3,3}$$

Multi-head Attention

d_model = 4
num_heads = 2

$$\text{depth} = \frac{\text{d\_model}}{\text{num\_heads}} = 2$$

assume seq_len = 3

$$q = \left[\begin{bmatrix} [1, 0, 0, 1] \\ [0, 1, 1, 0] \\ [1, 1, 1, 1] \end{bmatrix} \right]_{1 \times 3 \times 4}$$

head for q: will be shape $(\overset{\text{batch\_size}}{1}, \underset{\text{heads}}{2}, \underset{\text{seq\_len}}{3}, \overset{\text{depth}}{2})$

$$\text{q\_heads} = \left[\begin{bmatrix} [1, 0] \\ [0, 1] \\ [1, 1] \end{bmatrix}, \begin{bmatrix} [0, 1] \\ [1, 0] \\ [1, 1] \end{bmatrix} \right]$$

Steps:

(1) embeddings + PE:
shape: (batch_size, seq_len, d_model)

(2) Multi head attention:

q = dense(d_model) // $q \cdot W_q$
v = dense(d_model) // $v \cdot W_v$
k = dense(d_model) // $k \cdot W_k$
shapes:

(batch_size, seq_len, d_model)

- split into heads:

q_heads, k_heads, v_heads: (batch_size, num_heads, seq_len, depth)

- dot-Product attention:

(*) $A = (q\_heads \cdot k\_heads^T) / \sqrt{d_k}$

d_k = shape(k)[-1]

(*) $S = softmax(A)$

(*) $S \cdot v\_heads$