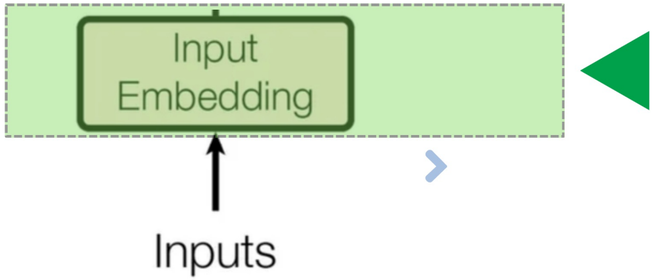
بخش inputs

I really enjoy learning transformers

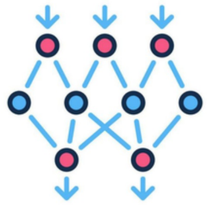
["I", "really", "enjoy", "learning", "transformers"]

بخش inputs



input (ورودی) ⟶ ["I", "really", "enjoy", "learning", "transformers"]

تبدیل کلمات به ایندکس (Tokenization) ⟶ **[0 , 1 , 2 , 3 , 4]**



اعمال Embedding Layer ⟶

(۵ کلمه × ۴ ویژگی)

d_model = 4

[[0.1, 0.3, 0.4, 0.8], ← "I"
[0.9, 0.1, 0.5, 0.2], ← "really"
[0.7, 0.6, 0.1, 0.4], ← "enjoy"
[0.2, 0.8, 0.3, 0.7], ← "learning"
[0.5, 0.4, 0.9, 0.3]] ← "transformers"

⟶

[[0.1, 0.3, 0.4, 0.8],
[0.9, 0.1, 0.5, 0.2],
[0.7, 0.6, 0.1, 0.4],
[0.2, 0.8, 0.3, 0.7],
[0.5, 0.4, 0.9, 0.3]]

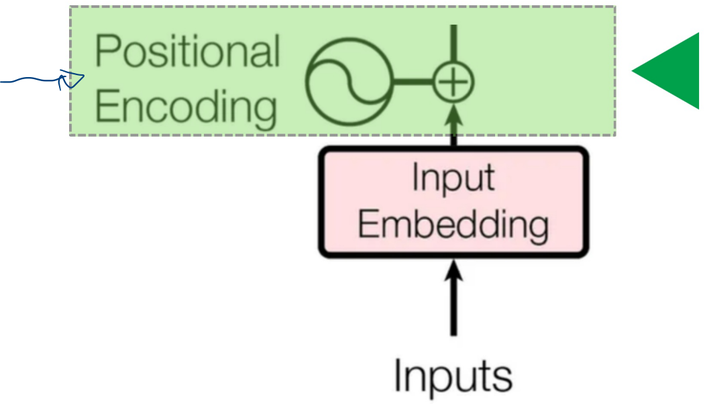
Word Embedding

- One-Hot Encoding
- TF-IDF، Bag-of-Words
- Contextual Embedding
- Trainable Embedding Layer

چرا positional encoding؟

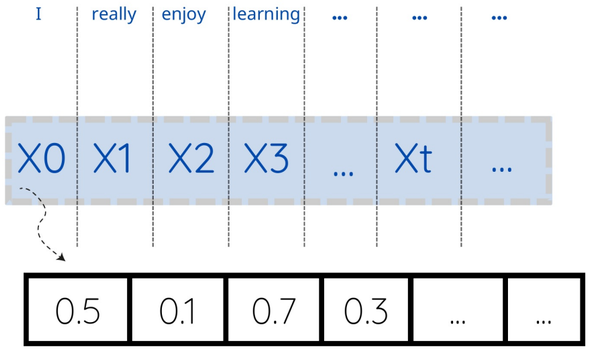
برخلاف RNNها که توکن‌ها را به ترتیب پردازش می‌کنند، Transformer کل جمله را یکجا (موازی) پردازش می‌کند.

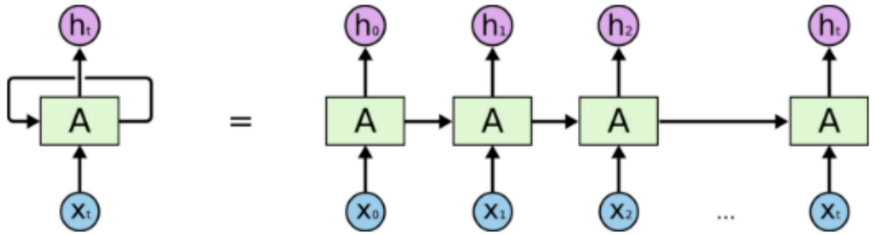
Transformer به‌صورت ذاتی نمی‌داند ترتیب کلمات چیست!



Input: I really enjoy learning transformers

RNN





TRANSFORMERS

[[0.1, 0.3, 0.4, 0.8], ← "I"
[0.9, 0.1, 0.5, 0.2], ← "really"
[0.7, 0.6, 0.1, 0.4], ← "enjoy"
[0.2, 0.8, 0.3, 0.7], ← "learning"
[0.5, 0.4, 0.9, 0.3]] ← "transformers"

[[0.1, 0.3, 0.4, 0.8],
[0.9, 0.1, 0.5, 0.2],
[0.7, 0.6, 0.1, 0.4],
[0.2, 0.8, 0.3, 0.7],
[0.5, 0.4, 0.9, 0.3]]

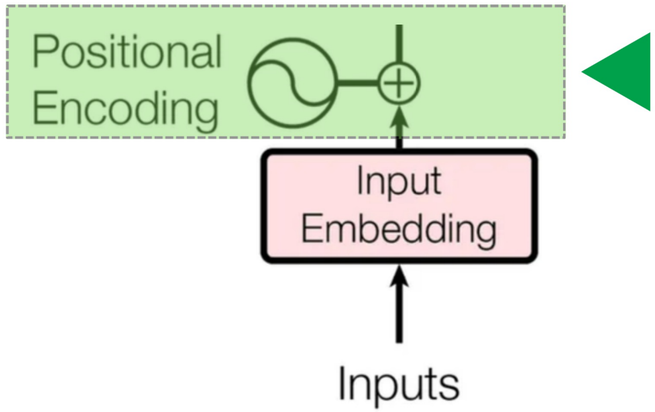
Transformers

[[0.7, 0.8, 0.1, 0.0],
[0.3, 0.3, 0.7, 0.1],
[0.2, 0.4, 0.4, 0.4],
[0.7, 0.7, 0.1, 0.7],
[0.1, 0.1, 0.4, 0.3]]

چگونه positional encoding؟ (sinusoidal positional encoding)

برای اینکه مدل بتواند ترتیب توکن‌ها را تشخیص دهد، به هر توکن یک بردار موقعیتی (Positional Encoding) اضافه می‌شود.

هدف Positional Encoding: دادن اطلاعات «موقعیت» به هر توکن به‌طوری که مدل بفهمد "I" اول جمله است و "transformers" آخر



For each position pos and dimension i:

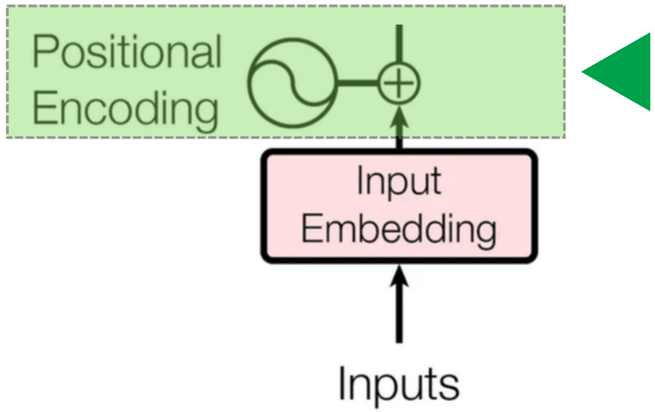
$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

Here, d_{model} is the dimension of the model.

d_model = embedding dimension

چگونه positional encoding ؟



$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

d_model = embedding dimension = 4

i=0 > pos/(10000 ^ (0/4)) = pos / 1

i=1 > pos/(10000 ^ (2/4)) = pos / 10000^0.5 = pos/100

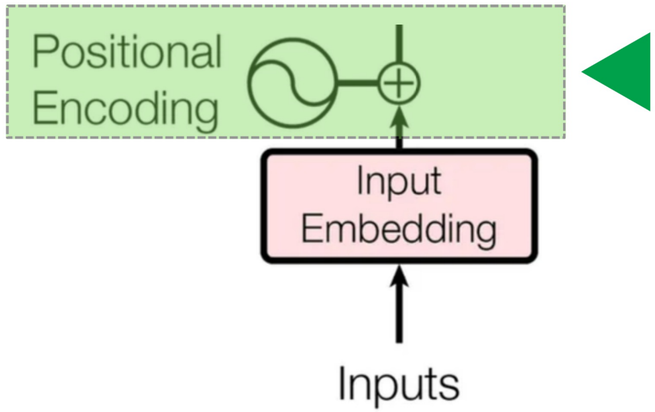
Sequence	Index of token (k)
I	pos = 0
really	pos = 1
enjoy	pos = 2
learning	pos = 3
transformers	pos = 4

i=0	i=1	i=2	i=3
sin(0/10000^(0/4))=0.0	cos(0/10000^(0/4))=1.0	sin(0/10000^(2/4))=0.0	cos(0/10000^(2/4))=1.0
sin(1/10000^(0/4))=0.84	cos(1/10000^(0/4))=0.54	sin(1/10000^(2/4))=0.01	cos(1/10000^(2/4))=0.99
0.90930	-0.41615	0.02000	0.99980
0.14112	-0.98999	0.02999	0.99955
-0.75680	-0.65364	0.03998	0.99920

نحوه ترکیب اطلاعات ترتیبی (ازمرحله ی Positional Encoding) و بردار هر توکن (کلمه) (از مرحله ی Input Embedding)

برای هر توکن (کلمه)، بردار embedding **با بردار** positional encoding **جمع می‌شود.**

$$X = E + PE$$



E

I	0.1	0.3	0.4	0.8
really	0.9	0.1	0.5	0.2
enjoy	0.7	0.6	0.1	0.4
learning	0.2	0.8	0.3	0.7
transformers	0.5	0.4	0.9	0.3

+

PE

0.0	1.0	0.0	1.0
0.84	0.54	0.01	0.99
0	-0.41	0.02	0.99
0.14	-0.98	0.029	0.99
-0.75	-0.65	0.039	0.999

=

X

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

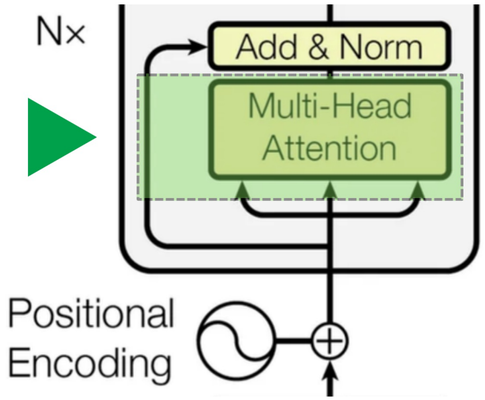
تمرین (نمره اضافه / اختیاری) !

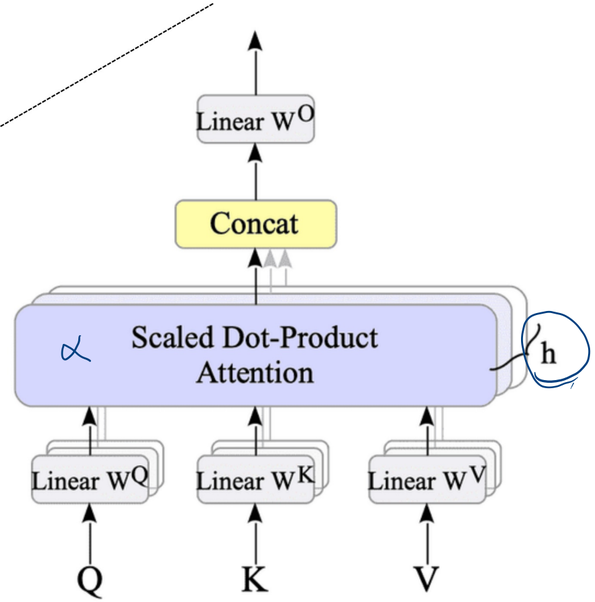
از این ماتریس (خروجی بخش Positional Encoding) مدل چگونه ترتیب کلمات را می فهمد؟

$$\mathbf{X}$$

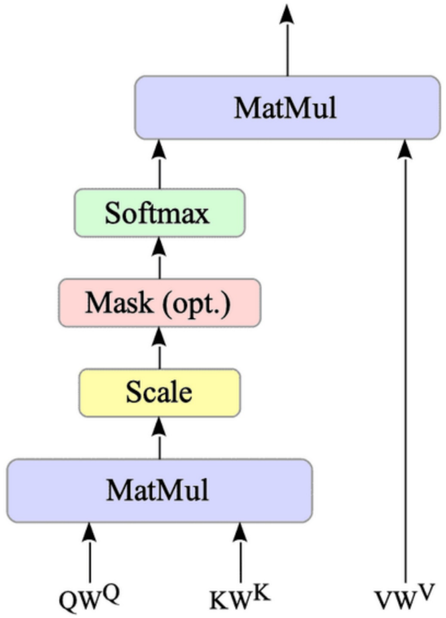
I	0.10	1.30	0.40	1.80
really	1.74	0.64	0.51	1.19
enjoy	1.60	0.18	0.12	1.39
learning	0.34	-0.18	0.32	1.69
transformers	-0.25	-0.25	0.93	1.29

نحوه ورود خروجی Positional Encoding به Multi Head Attention



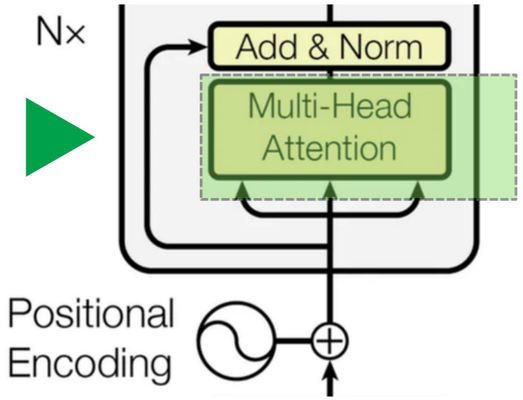


Multi-Head Attention



Scaled Dot-Product Attention

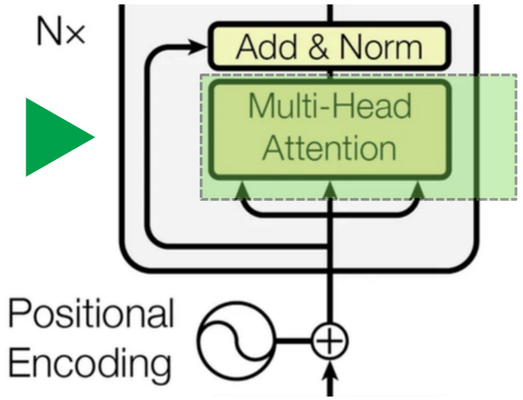
فرمول مکانیزم Self Attention

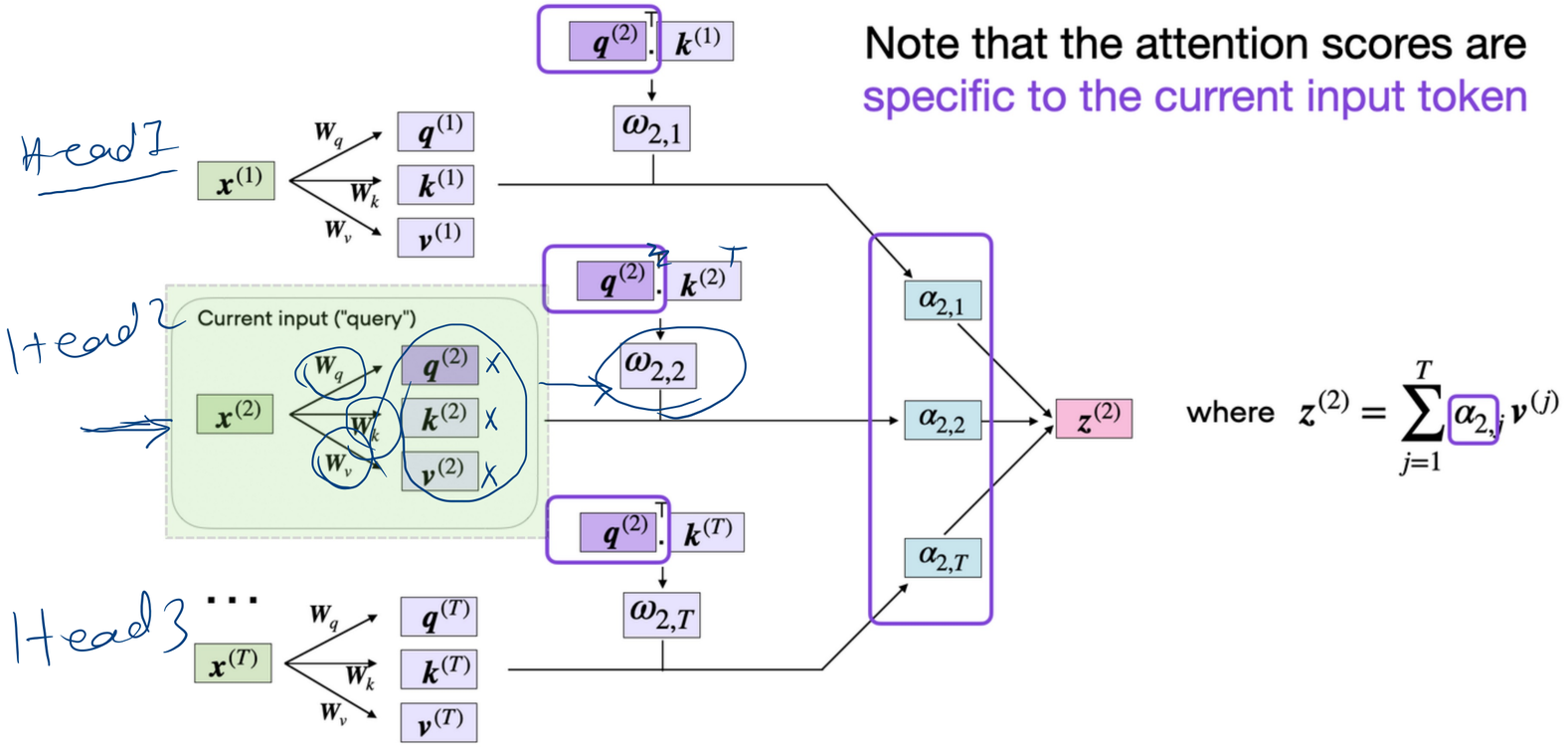


$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

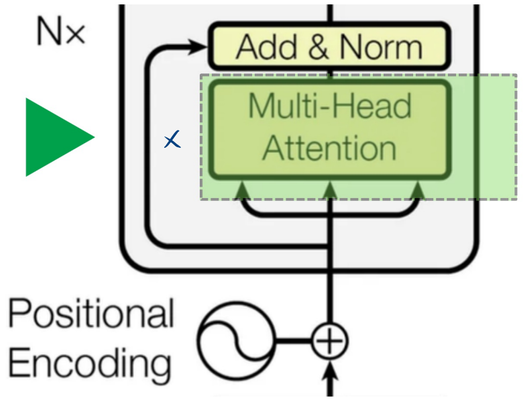
where Q, K and V are Query, Key and Value vectors.

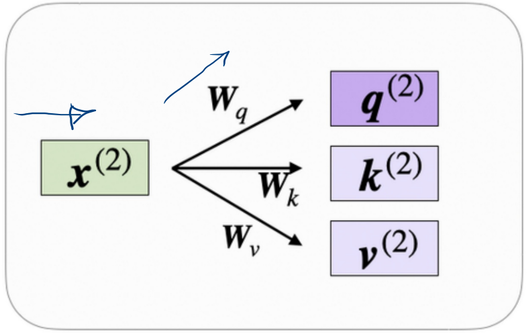
نمایش دیگری از معماری مکانیزم Self Attention





محاسبه بردارهای Query, Key, Value مکانیزم Self Attention





X

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

*

Wq

W1	W2	W3	W4
W5	W6	W7	W8
W9	W10	W11	W12
W13	W14	W15	W16

=

Q

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

X

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

*

Wk

W1	W2	W3	W4
W5	W6	W7	W8
W9	W10	W11	W12
W13	W14	W15	W16

=

K

0.5	2.89	1.45	-0.44
3.96	1.08	1.31	-1.6
4.35	1.67	0.89	-1.0
4.35	1.82	0.22	0.43
4.39	0.43	-0.06	0.72

X

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

(5x4)

*

ماتریس وزن مربوط به Value (Wv)

W1	W2	W3	W4
W5	W6	W7	W8
W9	W10	W11	W12
W13	W14	W15	W16

(4x4)

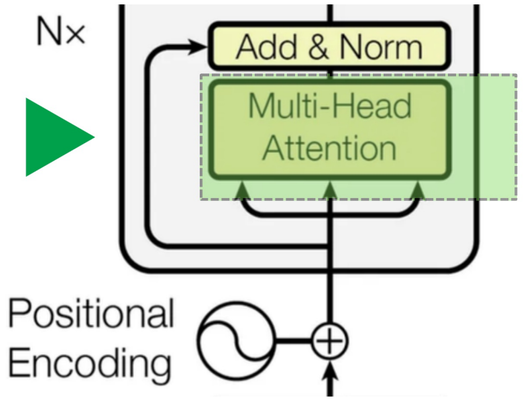
=

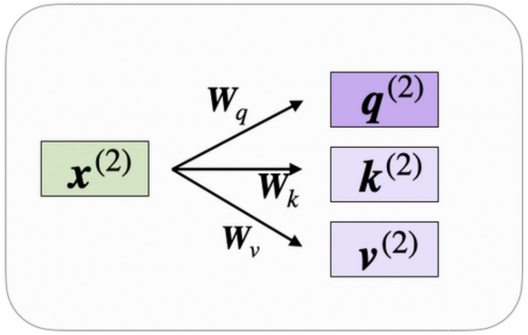
V

0.17	0.01	-3.48	1.8
-1.9	-3.93	-3.23	2.01
-2.04	-3.75	-2.59	1.52
-1.75	-2.12	-2.74	2.08
-1.73	-1.7	-3.04	2.88

(5x4)

نمایش دیگری از معماری مکانیزم Self Attention





Q

0.10	1.30	0.40	1.80
1.74	0.64	0.51	1.19
1.60	0.18	0.12	1.39
0.34	-0.18	0.32	1.69
-0.25	-0.25	0.93	1.29

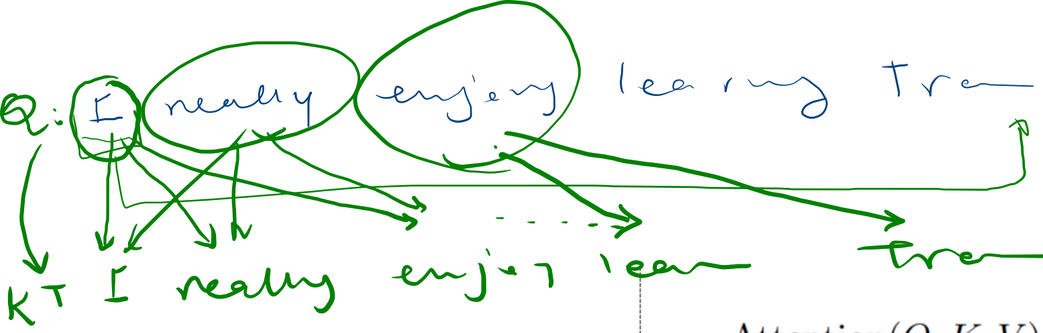
K

0.5	2.89	1.45	-0.44
3.96	1.08	1.31	-1.6
4.35	1.67	0.89	-1.0
4.35	1.82	0.22	0.43
4.39	0.43	-0.06	0.72

V

0.17	0.01	-3.48	1.8
-1.9	-3.93	-3.23	2.01
-2.04	-3.75	-2.59	1.52
-1.75	-2.12	-2.74	2.08
-1.73	-1.7	-3.04	2.88

(5x4)



بخش self attention

محاسبه Attention Scores

توکن جاری (Query) چقدر باید به هر یک از توکن‌های دیگر (Keyها) توجه کند؟

Q_i: بردار "سؤال" توکن i

K_j: بردار "کلید" توکن j

شباهت یا تطابق (میزان توجهی که باید باشد)بین نیاز توکن i و اطلاعات توکن j

مثلا کلمه ی enjoy باید بیشتر به learning توجه کند!

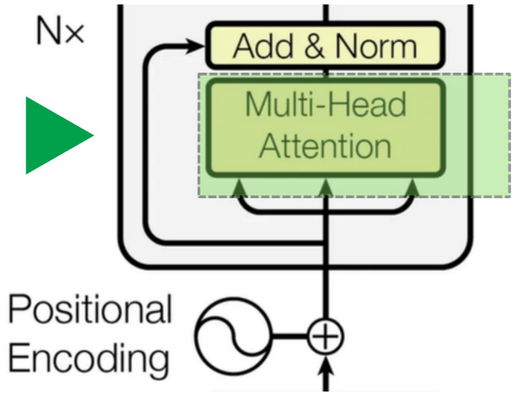
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

K

0.5	2.89	1.45	-0.44
3.96	1.08	1.31	-1.6
4.35	1.67	1.08	-1.0
4.35	1.82	0.22	0.43
4.39	0.43	-0.06	0.72

K^T

0.5	3.96	4.35	4.35	4.39
2.89	1.08	1.67	1.82	0.43
1.45	1.31	1.08	0.22	-0.06
-0.44	-1.6	-1.0	0.43	0.72
I	really	enjoy	learning	transformers



$$\frac{Q.K^T}{\sqrt{d}} =$$

Q

I	0.10	1.30	0.40	1.80
really	1.74	0.64	0.51	1.19
enjoy	1.60	0.18	0.12	1.39
learning	0.34	-0.18	0.32	1.69
transformers	-0.25	-0.25	0.93	1.29

5X4

·

K^T

0.5	3.96	4.35	4.35	4.39
2.89	1.08	1.67	1.82	0.43
1.45	1.31	1.08	0.22	-0.06
-0.44	-1.6	-1.0	0.43	0.72
I	really	enjoy	learning	transformers

4X5

=

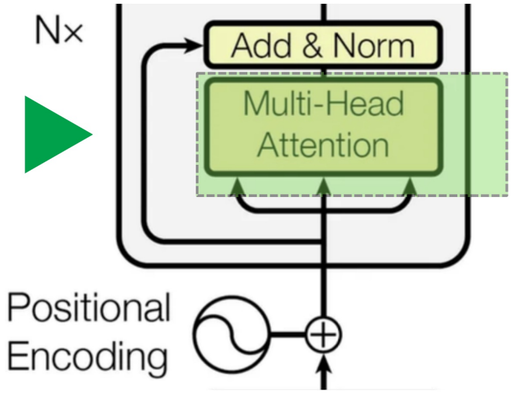
Attention Scores

I	1.07	7.8	8.21	8.28	8.83
really	2.91	8.34	10.33	12.93	13.05
enjoy	2.84	6.94	8.73	11	10.9
learning	1.62	2.49	3.33	4.38	4.09
transformers	0.87	-0.57	-0.04	0.85	0.52
	I	really	enjoy	learning	transformers

محاسبه Attention Scores

توکن جاری (Query) چقدر باید به هر یک از توکن‌های دیگر (Keyها) توجه کند؟

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



$$\text{softmax}(\frac{Q.K^T}{\sqrt{d}}) = \text{softmax}(\begin{matrix} 1.07 & 7.8 & 8.21 & 8.28 & 8.83 \\ 2.91 & 8.34 & 10.33 & 12.93 & 13.05 \\ 2.84 & 6.94 & 8.73 & 11 & 10.9 \\ 1.62 & 2.49 & 3.33 & 4.38 & 4.09 \\ 0.87 & -0.57 & -0.04 & 0.85 & 0.52 \end{matrix}) =$$

Attention Weights

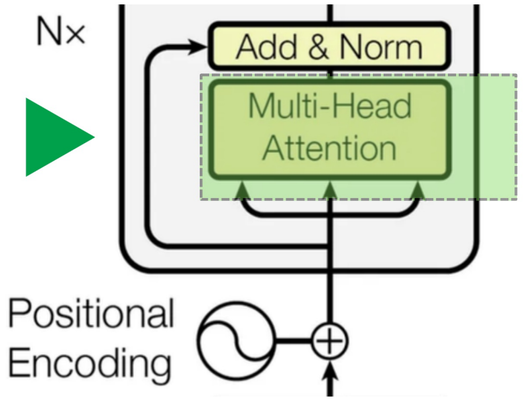
I	0.0002	0.1443	0.2185	0.234	0.403
me	0.0	0.0046	0.0335	0.4509	0.5111
	0.0001	0.0085	0.0508	0.495	0.4456
	0.0274	0.0655	0.1517	0.4332	0.3222
	0.3	0.0716	0.1215	0.2945	0.2124

محاسبه Attention Output

نتیجه نهایی مکانیزم Self-Attention

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

خروجی Attention برای هر توکن، یک بردار جدید است که ترکیبی از اطلاعات بقیه توکن‌هاست، وزن‌دار شده با توجهی که آن توکن به دیگران داشته است.



Attention Weights

$$\text{softmax}(\frac{QK^T}{\sqrt{d_k}})V =$$

0.0002	0.1443	0.2185	0.234	0.403
0.0	0.0046	0.0335	0.4509	0.5111
0.0001	0.0085	0.0508	0.495	0.4456
0.0274	0.0655	0.1517	0.4332	0.3222
0.3	0.0716	0.1215	0.2945	0.2124

x

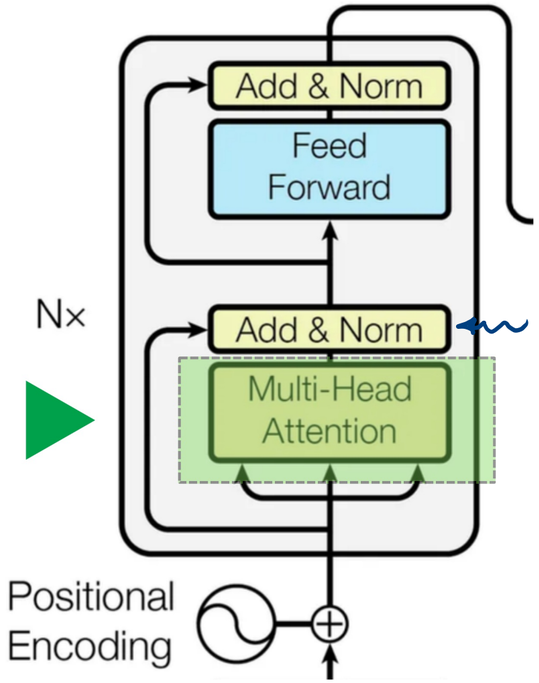
V

0.17	0.01	-3.48	1.8
-1.9	-3.93	-3.23	2.01
-2.04	-3.75	-2.59	1.52
-1.75	-2.12	-2.74	2.08
-1.73	-1.7	-3.04	2.88

=

-1.8281	-2.5672	-2.8994	2.2682
-1.7506	-1.967	-2.8913	2.468
-1.7572	-2.0295	-2.8707	2.4054
-1.7454	-2.2911	-2.866	2.2385
-1.2152	-1.7192	-3.0411	2.092

Multi Head Attention



head1

I really enjoy learning transformers

-1.8281	-2.5672	-2.8994	2.2682
-1.7506	-1.967	-2.8913	2.468
-1.7572	-2.0295	-2.8707	2.4054
-1.7454	-2.2911	-2.866	2.2385
-1.2152	-1.7192	-3.0411	2.092

head 2

Start asking to get answers

...

Multi Head Attention

Concat(head1 ... head100 ...).W0 =

X

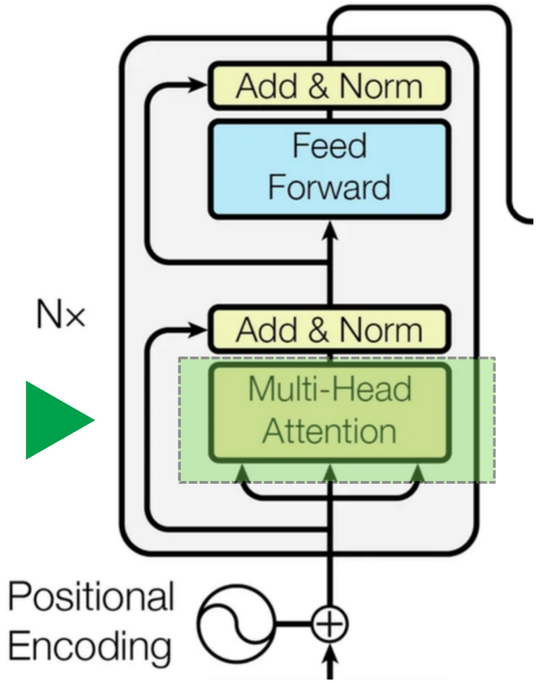
head1 head2 ...

-1.8281	-2.5672	-2.8994	2.2682				
-1.7506	-1.967	-2.8913	2.468				
-1.7572	-2.0295	-2.8707	2.4054				
-1.7454	-2.2911	-2.866	2.2385				
-1.2152	-1.7192	-3.0411	2.092				

... .W0

X

Multi Head Attention



head1

I really enjoy learning transformers

-1.8281	-2.5672	-2.8994	2.2682
-1.7506	-1.967	-2.8913	2.468
-1.7572	-2.0295	-2.8707	2.4054
-1.7454	-2.2911	-2.866	2.2385
-1.2152	-1.7192	-3.0411	2.092

head 2

Start asking to get answers

...

Multi Head Attention

Concat(head1 ... head100 ...).W0 =

-1.8281	-2.5672	-2.8994	2.2682				
-1.7506	-1.967	-2.8913	2.468				
-1.7572	-2.0295	-2.8707	2.4054				
-1.7454	-2.2911	-2.866	2.2385				
-1.2152	-1.7192	-3.0411	2.092				

... **.W0**

مرور مسیر کلی ترنسفورمر