Introduction to Deep Learning

Lecture 19
Transformers and LLMs

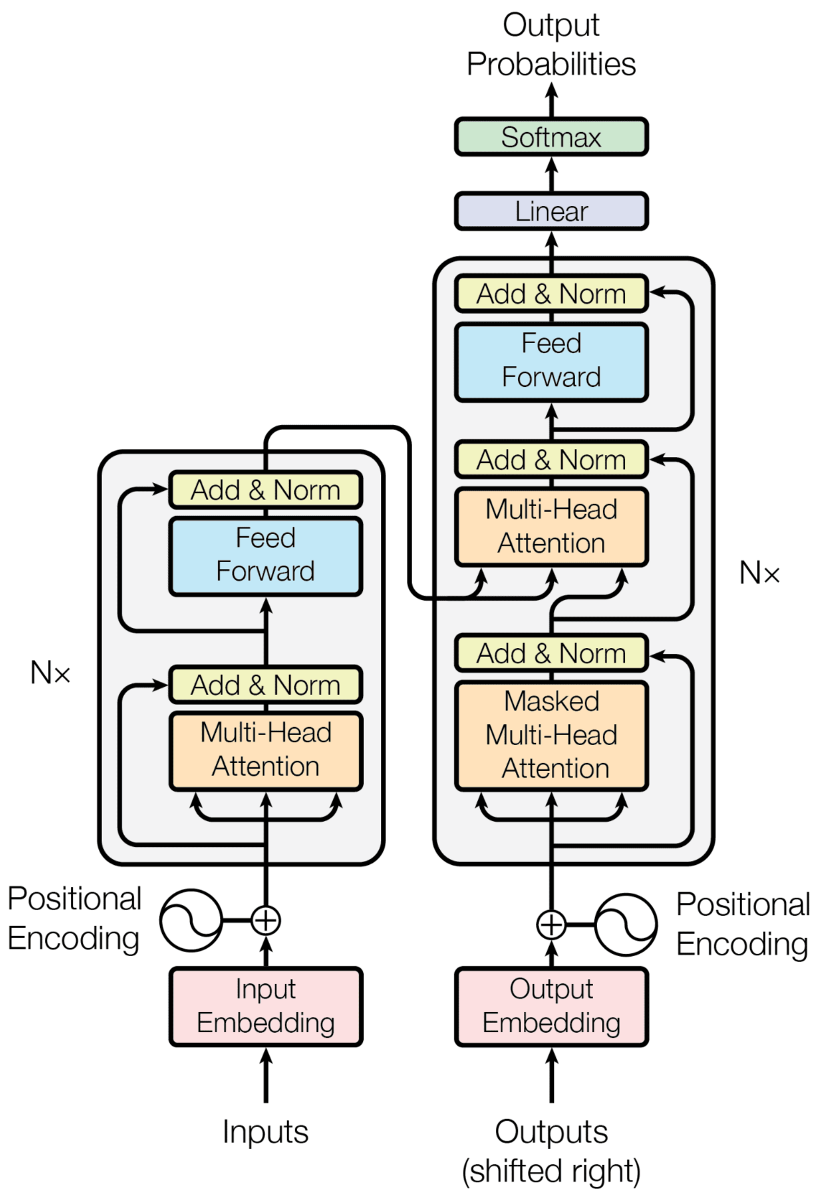
Shikhar Agnihotri

Liangze Li

11-785, Fall 2023

Part 1

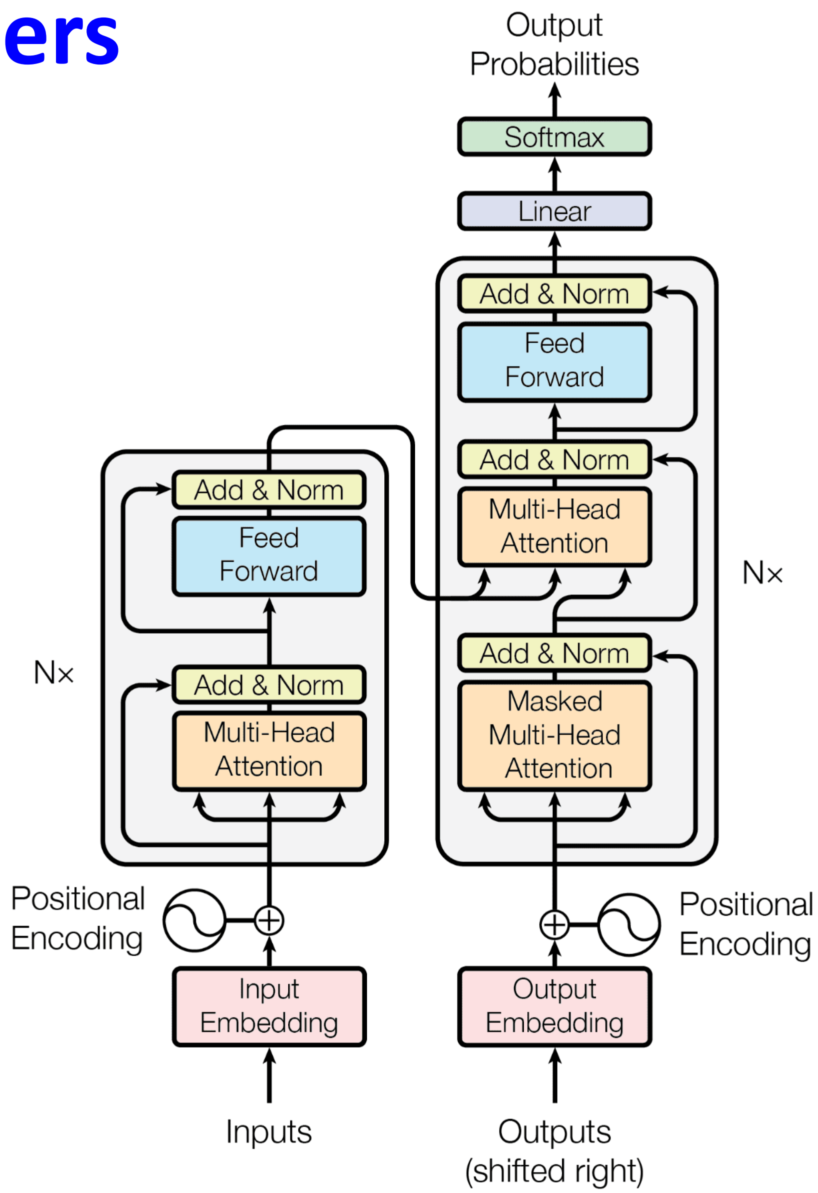
Transformers



Transformers

- Tokenizaton
- Input Embeddings
- Position Encodings
- Residuals
- Query
- Key
- Value
- Add & Norm
- Encoder
- Decoder

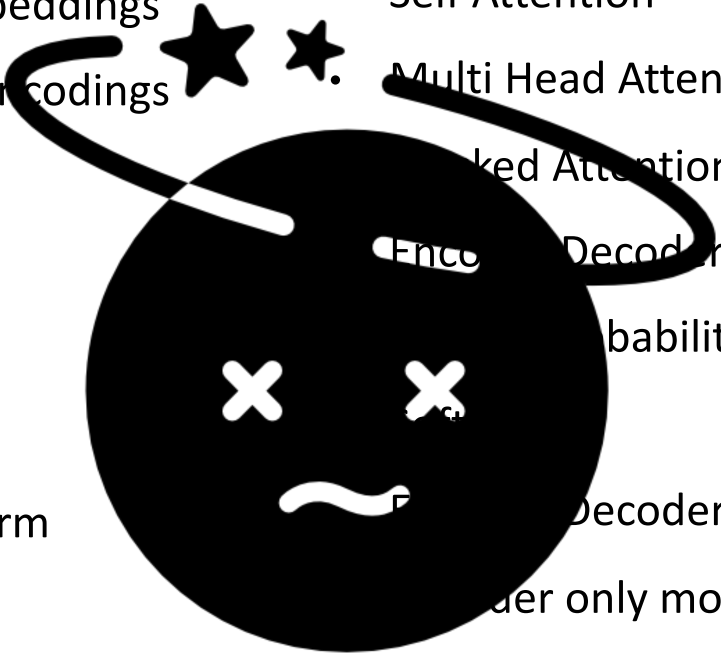
- Attention
- Self Attention
- Multi Head Attention
- Masked Attention
- Encoder Decoder Attention
- Output Probabilities / Logits
- Softmax
- Encoder-Decoder models
- Decoder only models

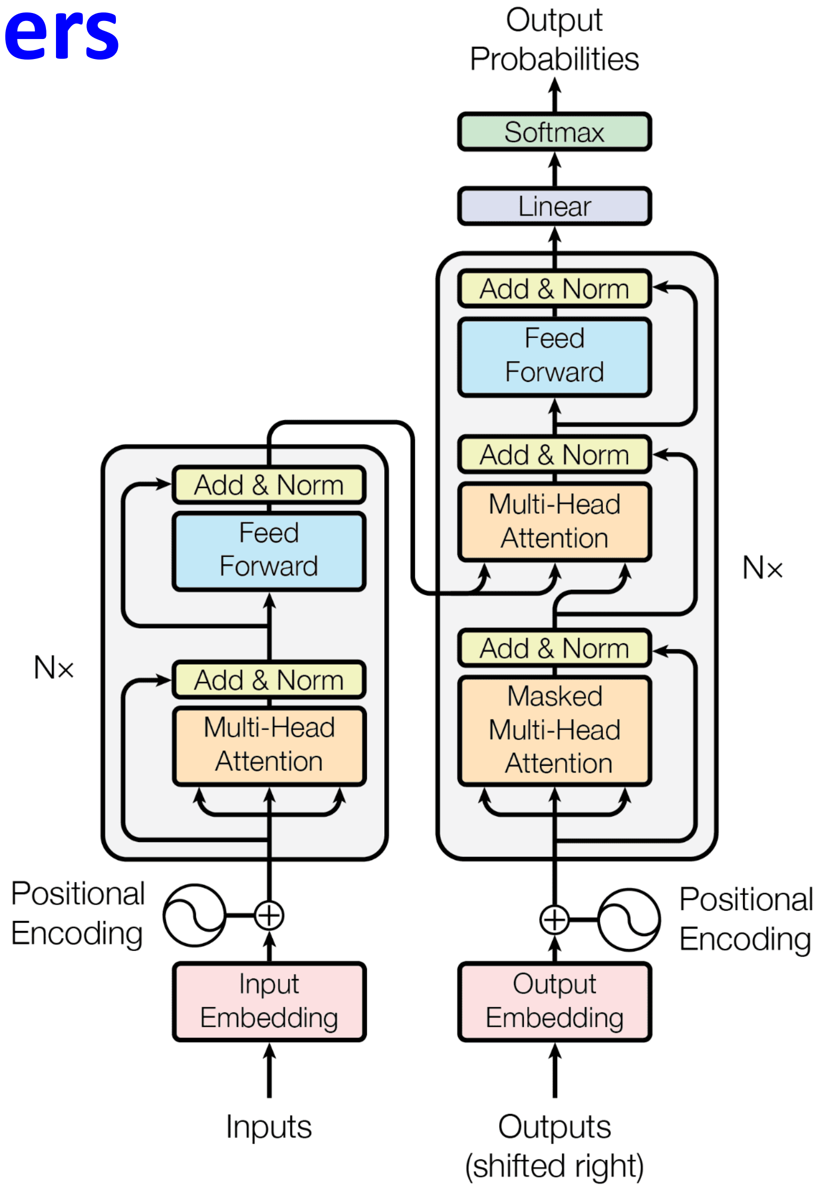


Transformers

- Tokenizaton
- Input Embeddings
- Position Er codings
- Residuals
- Query
- Key
- Value
- Add & Norm
- Encoder
- Decoder

- Attention
- Self Attention
- Multi Head Attention
- ked Attention
- Enc Decod r Attention
- babilities / Logits
- Decoder models
- der only models





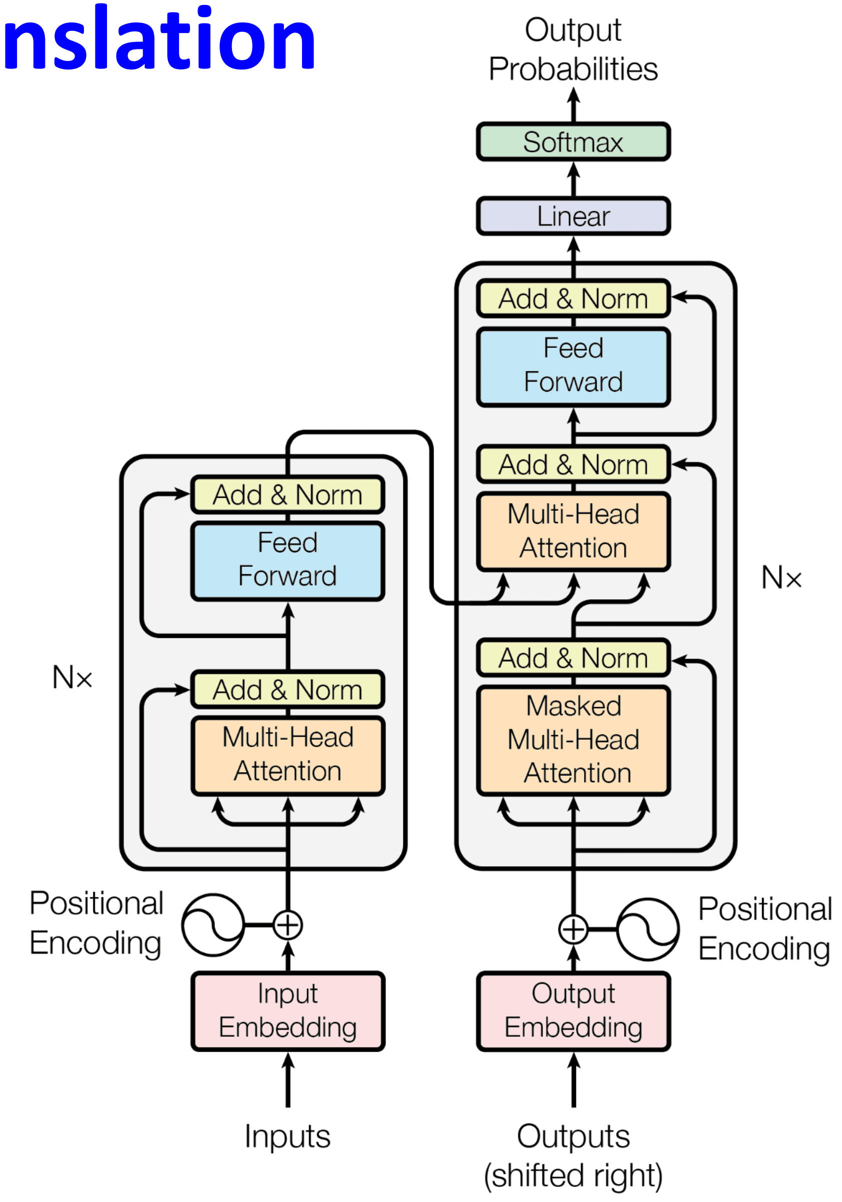
Machine Translation

Targets

Ich have einen apfel gegessen

Inputs

I ate an apple

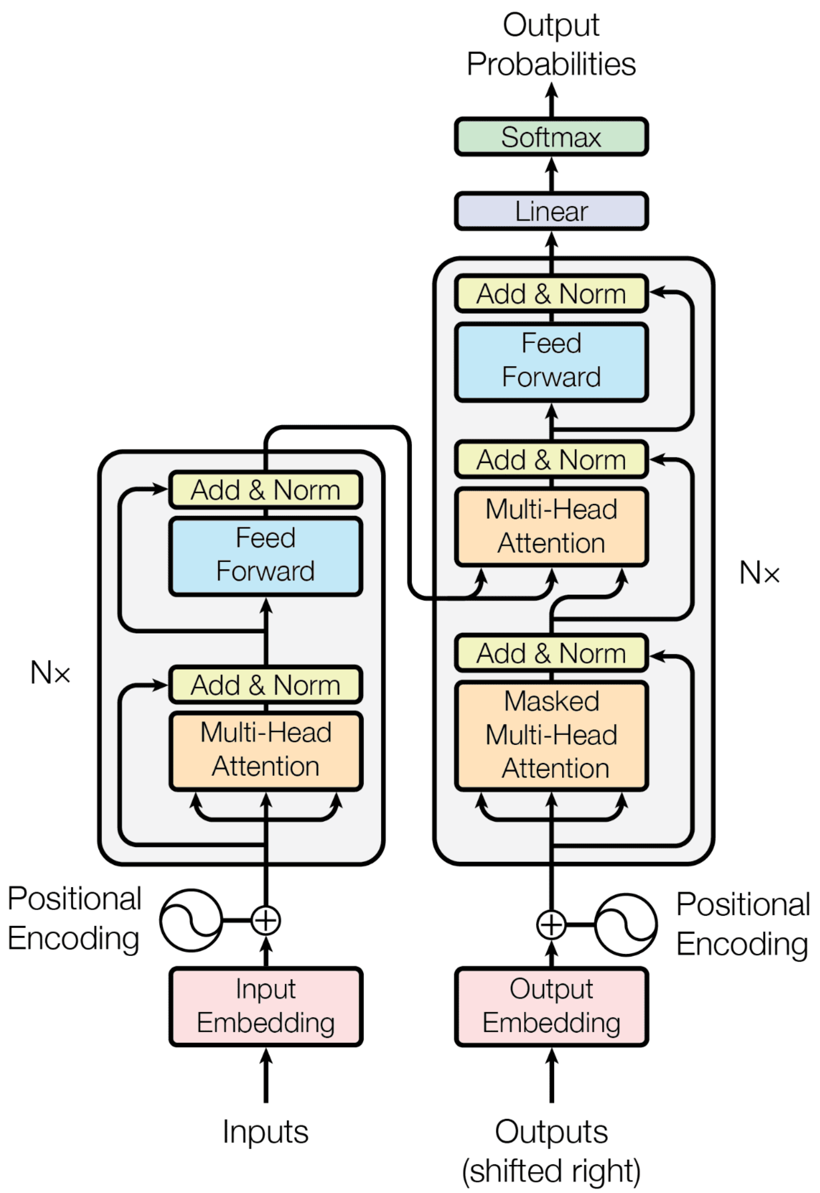


Inputs

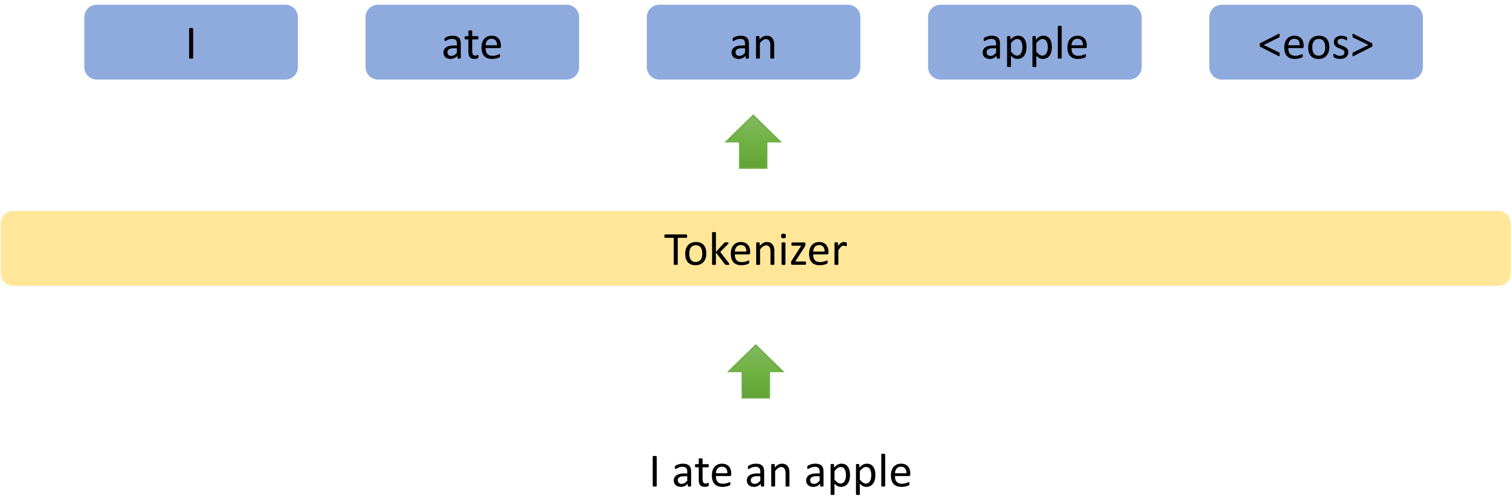
Processing Inputs

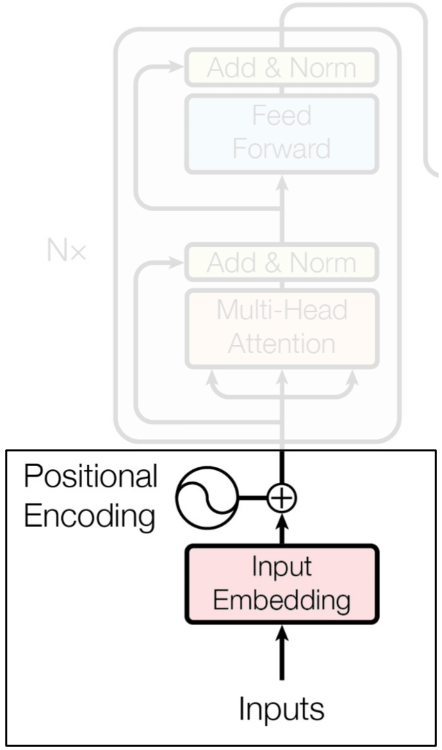
Inputs

I ate an apple



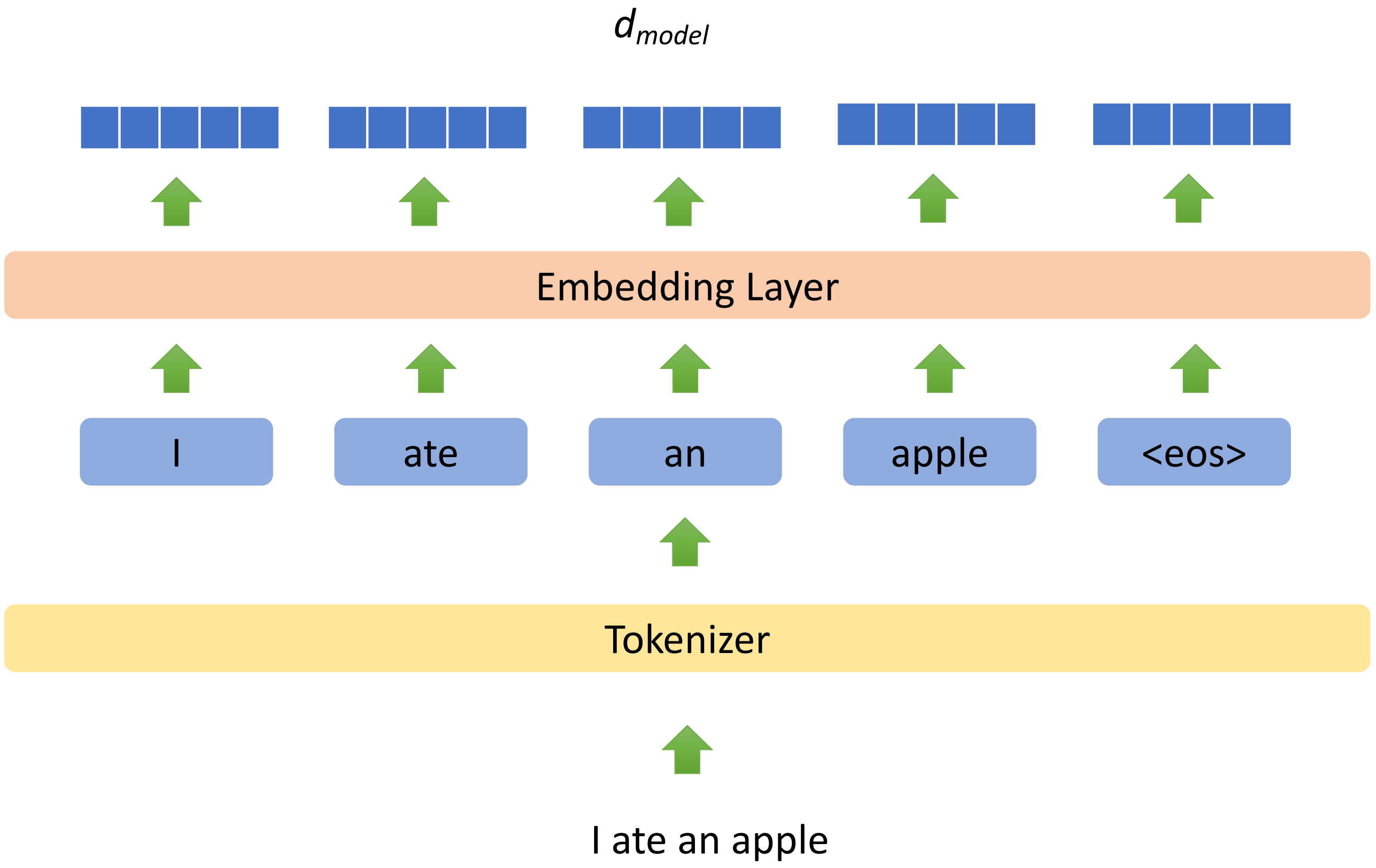
Inputs

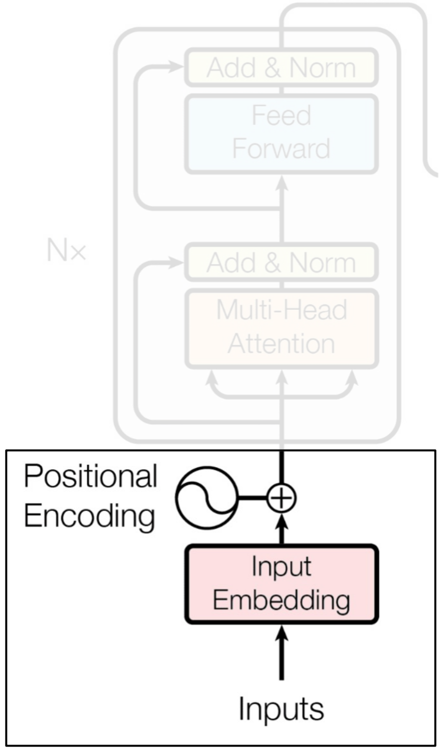




Generate Input Emebeddings

Inputs

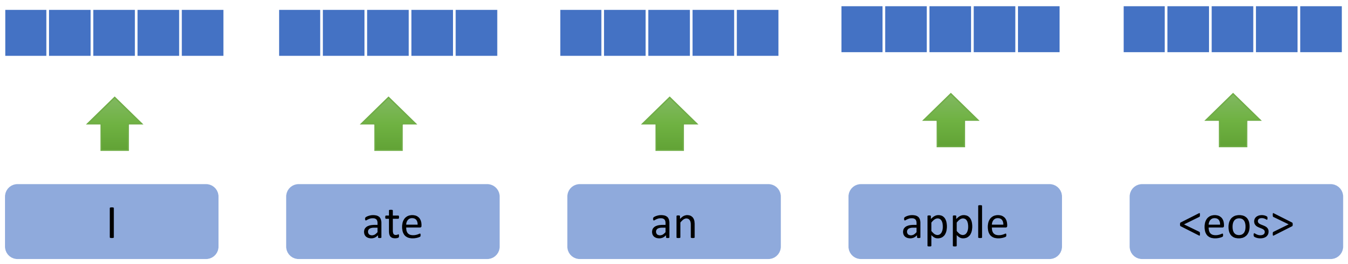


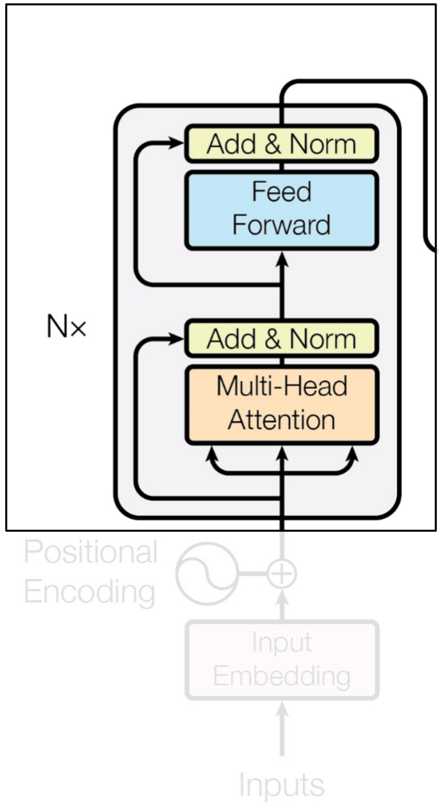


Generate Input Emebeddings

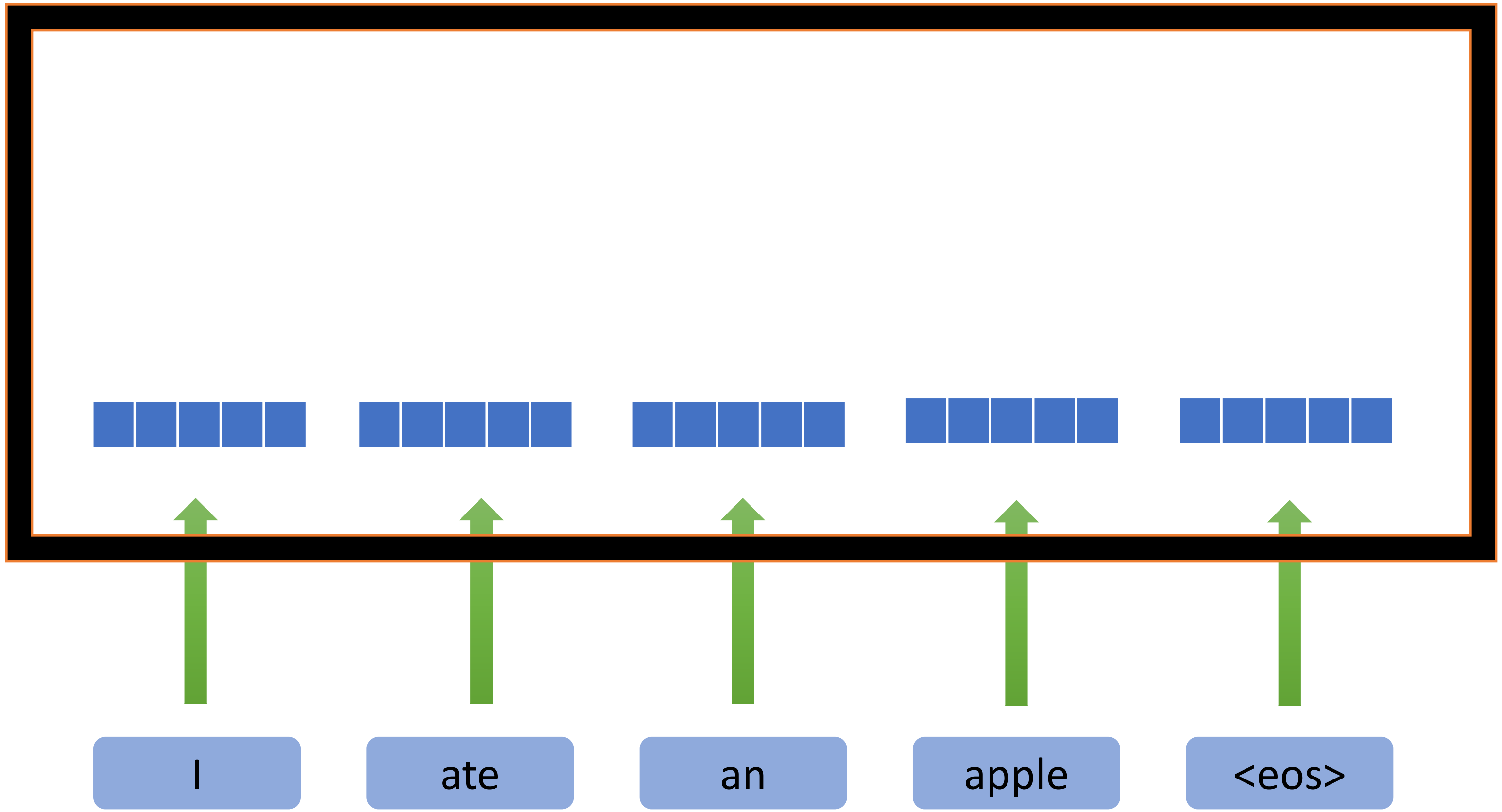
Encoder

WHERE IS THE CONTEXT ?

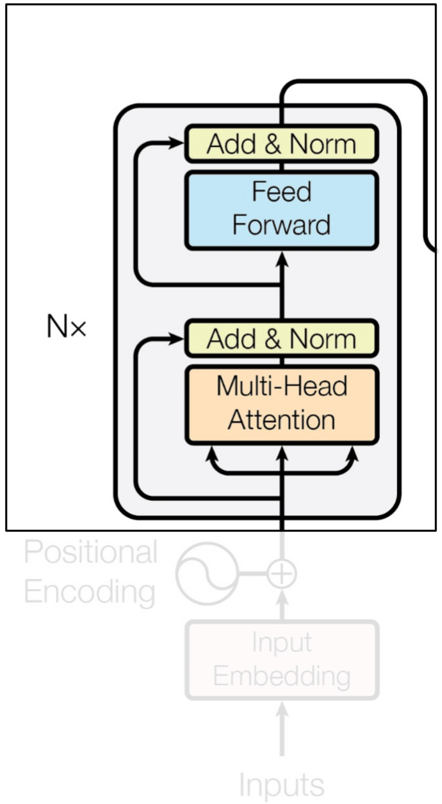




Encoder

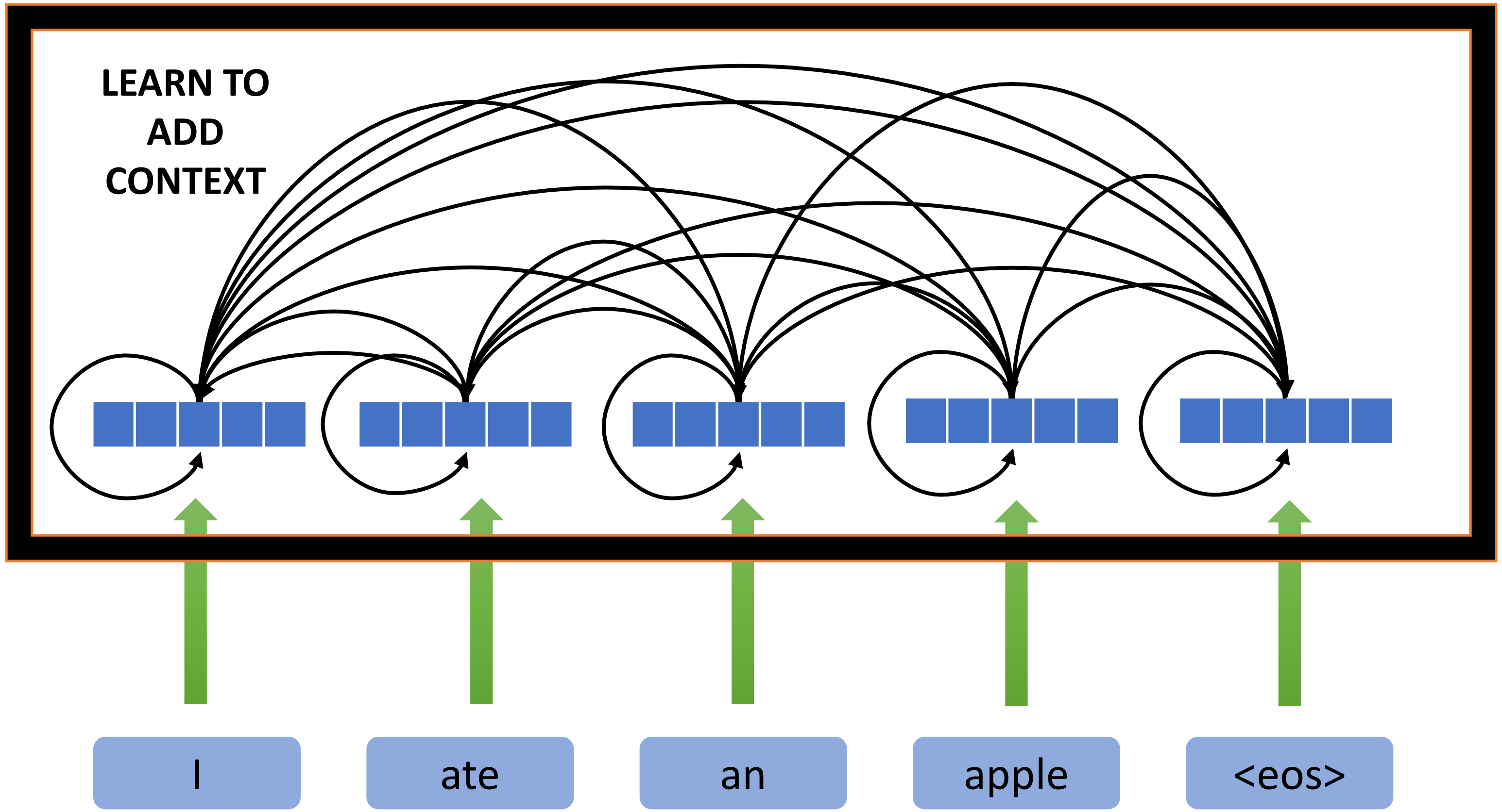


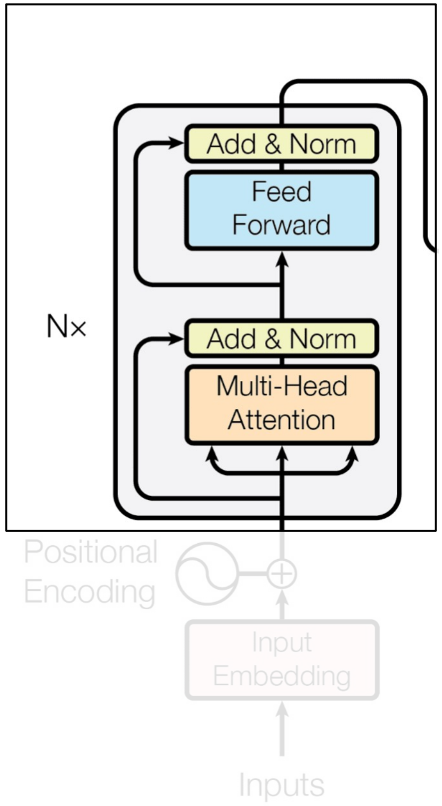
BLACK BOX
OF SORTS



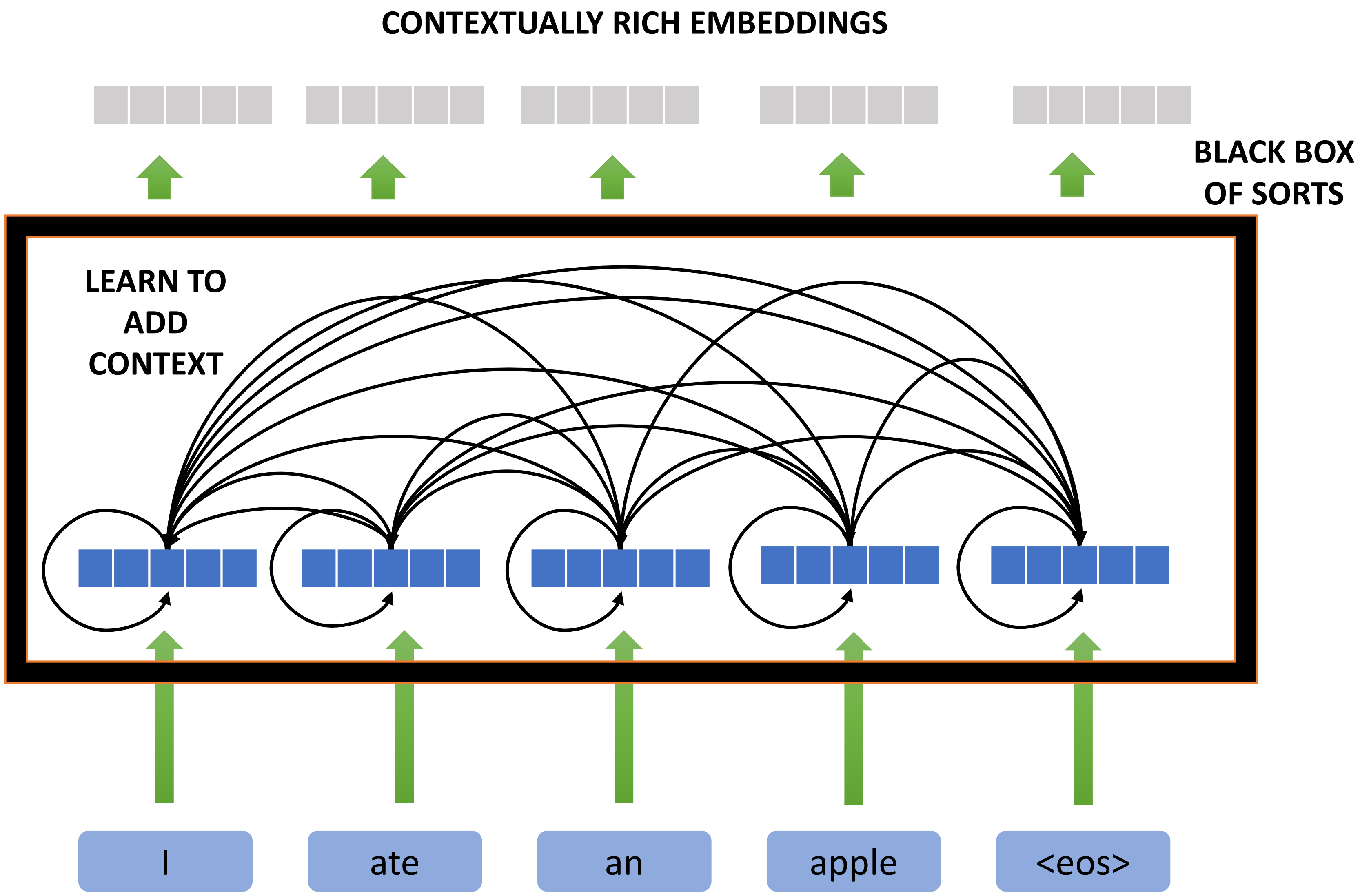
Encoder

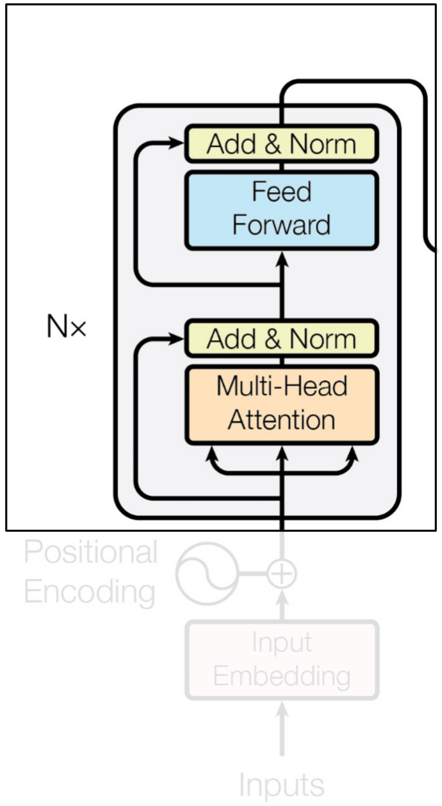
BLACK BOX
OF SORTS





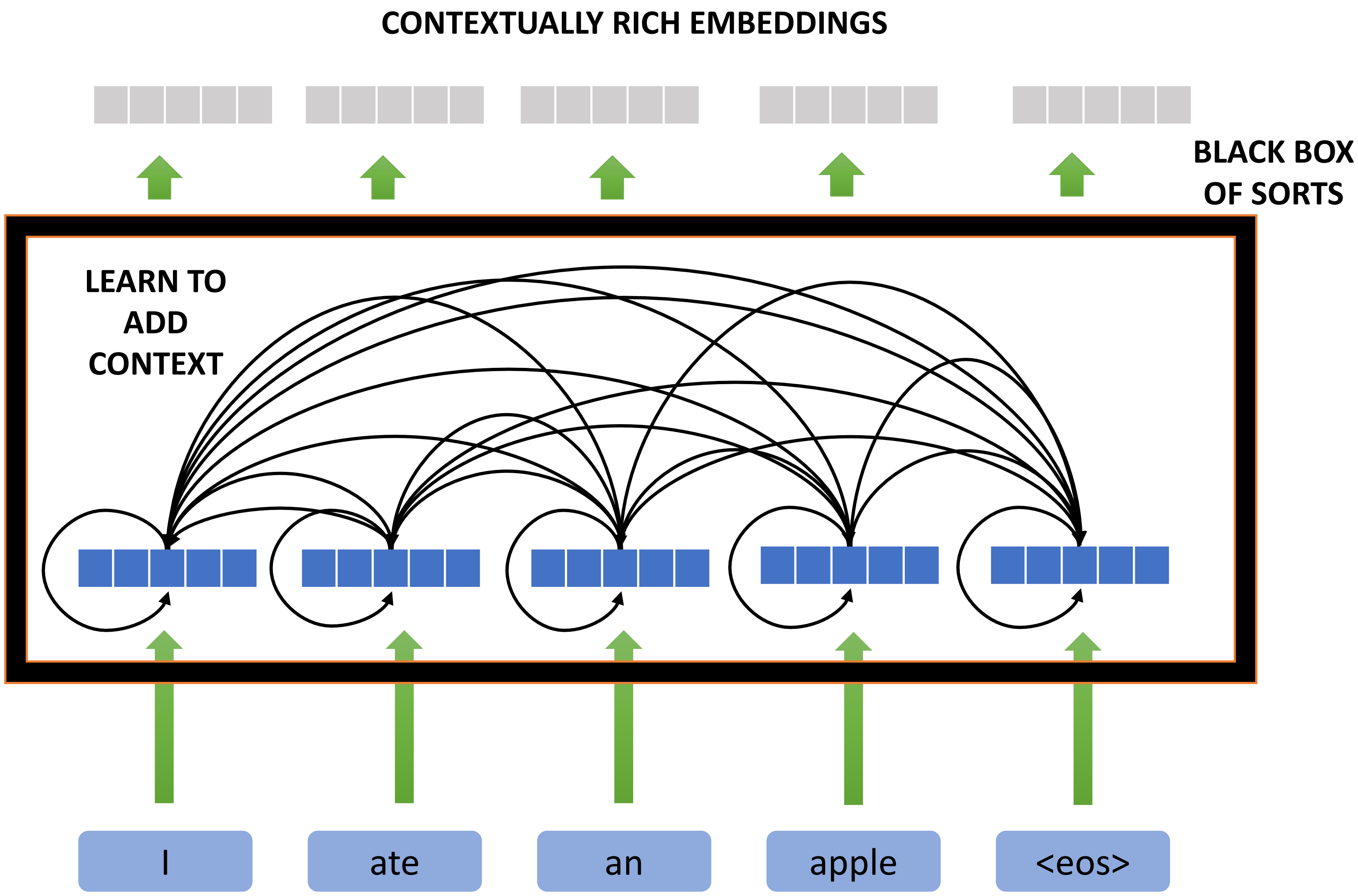
Encoder

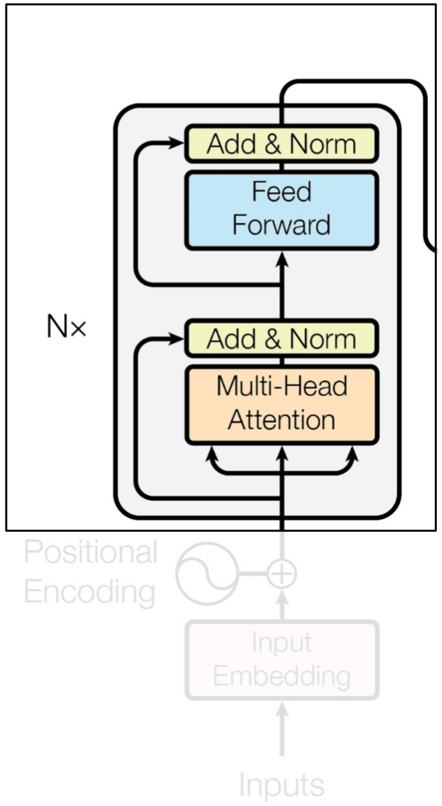

CONTEXTUALLY RICH EMBEDDINGS
BLACK BOX
OF SORTS
LEARN TO
ADD
CONTEXT
I
ate
an
apple
<eos>


Add & Norm
Feed
Forward
Nx
Add & Norm
Multi-Head
Attention
Positional
Encoding
Input
Embedding
Inputs

Encoder

$\alpha_{[i\,j]}$?

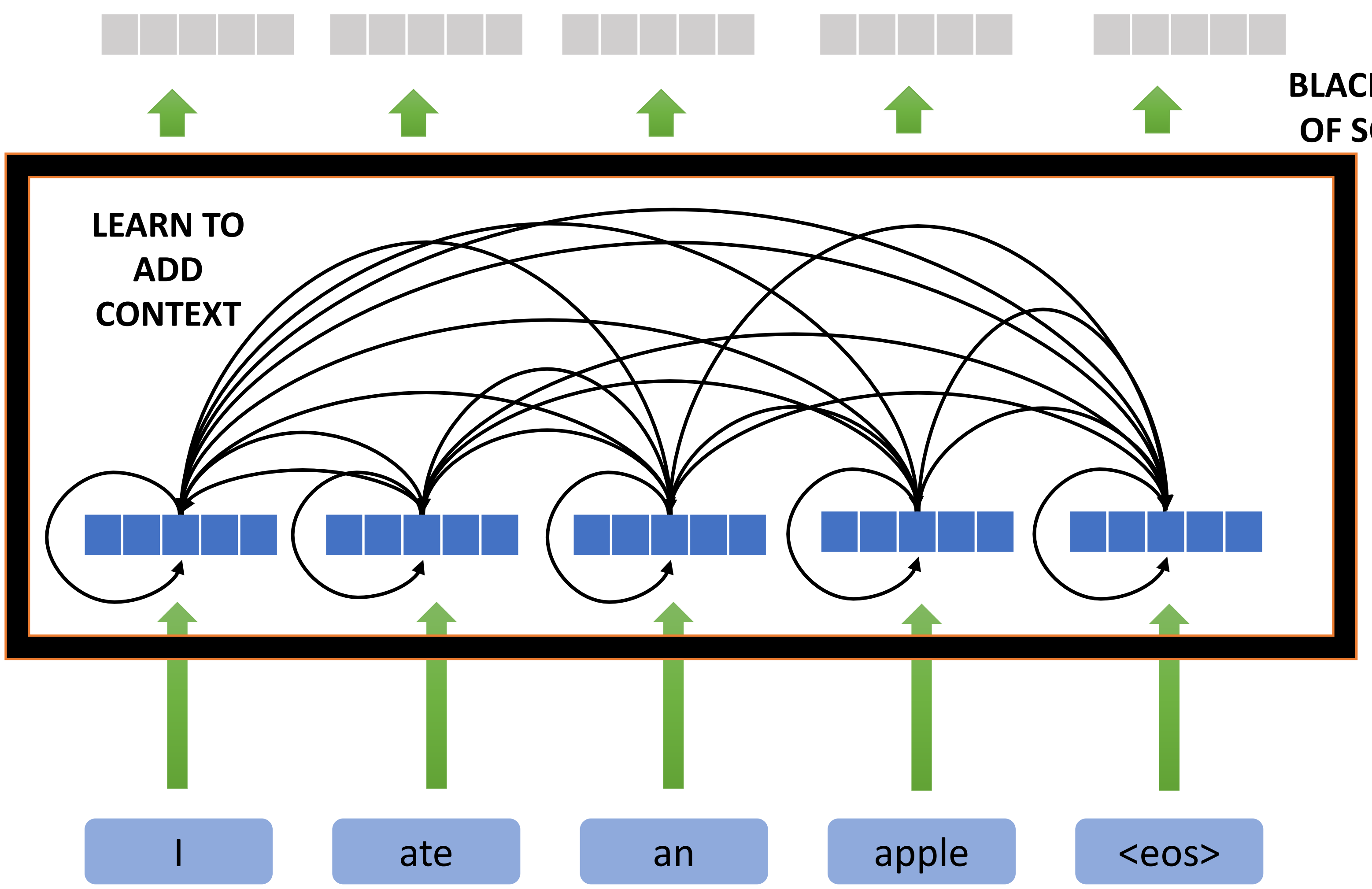




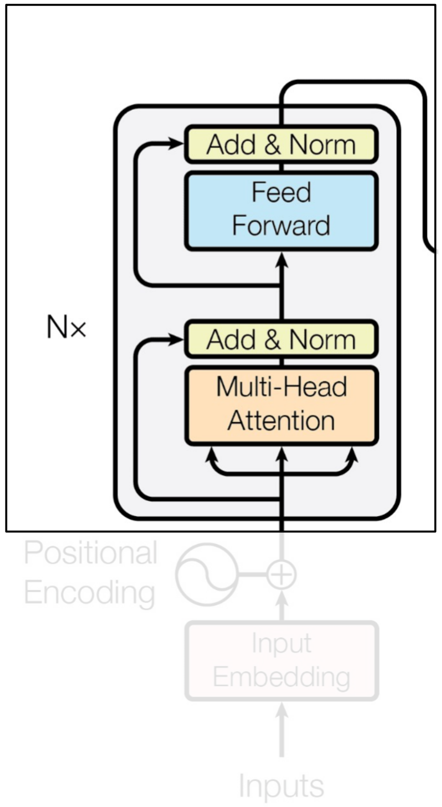
Encoder

$\alpha_{[i\ j]}$? Σ Π ?

CONTEXTUALLY RICH EMBEDDINGS



BLACK BOX
OF SORTS

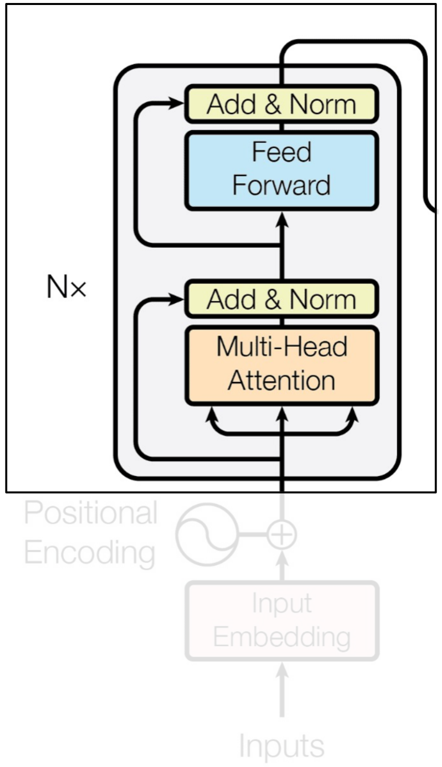


Attention

$\alpha_{[\,i\,j\,]}$?

From lecture 18:

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



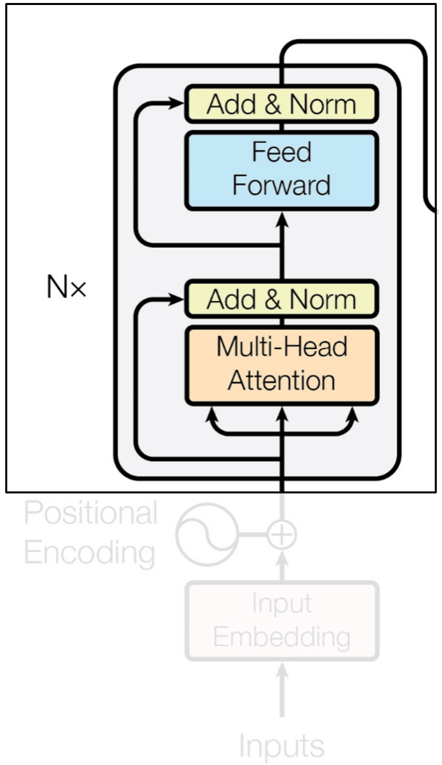
Attention

$\alpha_{[\,i\,j\,]}$?

From lecture 18:

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

- Query
- Key
- Value



Query, Key & Value

Database

{Key, Value store}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

{Query: "Order details of order_104"}

OR

{Query: "Order details of order_106"}

Database

{Key, Value store}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

{Query: “Order details of order_104”}

OR

{Query: “Order details of order_106”}

{Key, Value store}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

{Key, Value store}

{Query: “Order details of order_104”}

OR

{Query: “Order details of order_106”}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

{Key, Value store}

{Query: “Order details of order_104”}

OR

{Query: “Order details of order_106”}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

Done at the same time !!

{Query: “Order details of order_104”}

OR

{Query: “Order details of order_106”}

{Key, Value store}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query, Key & Value

{Query: "Order details of order_104"}

OR

{Query: "Order details of order_106"}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Query

1. Search for info

Key

1. Interacts directly with Queries
2. Distinguishes one object from another
3. Identify which object is the most relevant and by how much

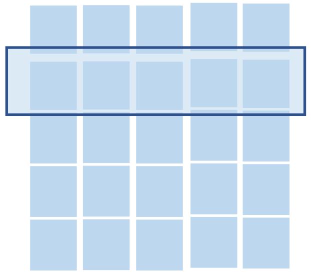
Value

1. Actual details of the object
2. More fine grained

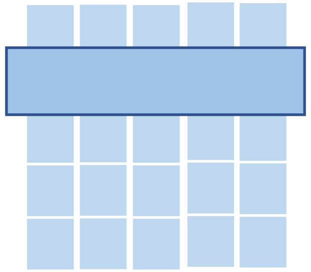
Attention

Query

Key Value Store

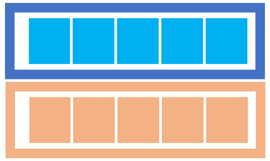


Key



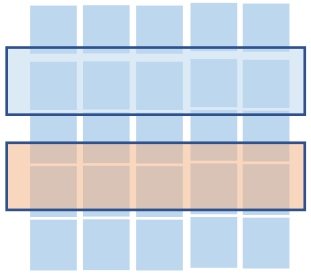
Value

Attention

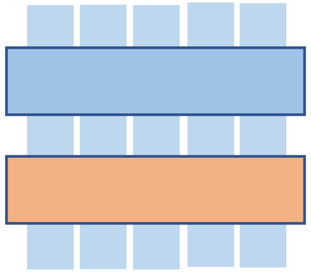


Query

Key Value Store



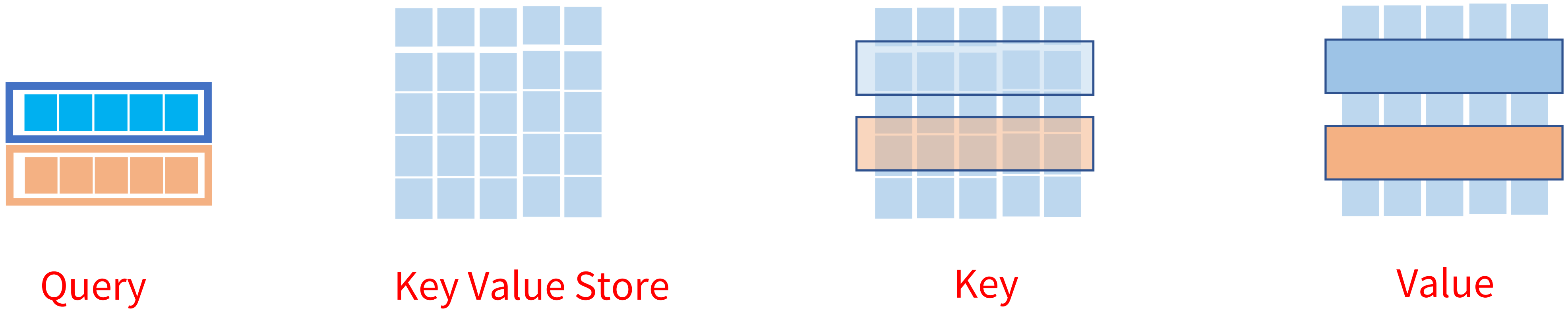
Key



Value

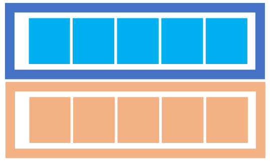
Attention

Done at the same time !!



Attention

Parallelizable !!!

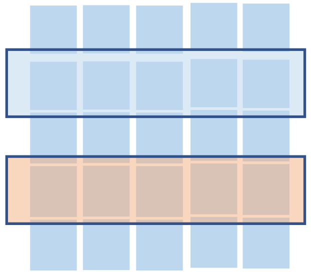


Query

Q

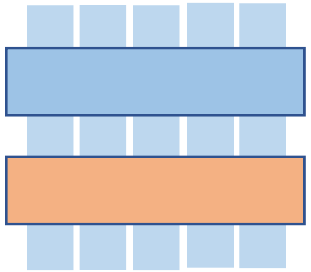
Key Value Store

QK^T



Key

$softmax(\frac{QK^T}{\sqrt{d}})$



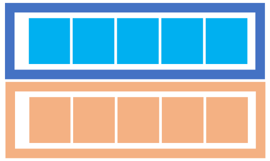
Value

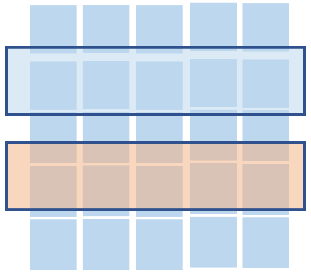
$softmax(\frac{QK^T}{\sqrt{d}})V$

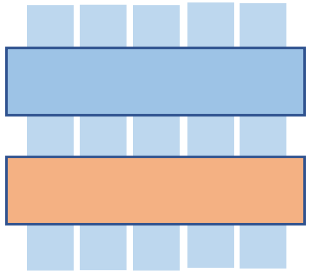
Attention

Parallelizable !!!

Attention Filter



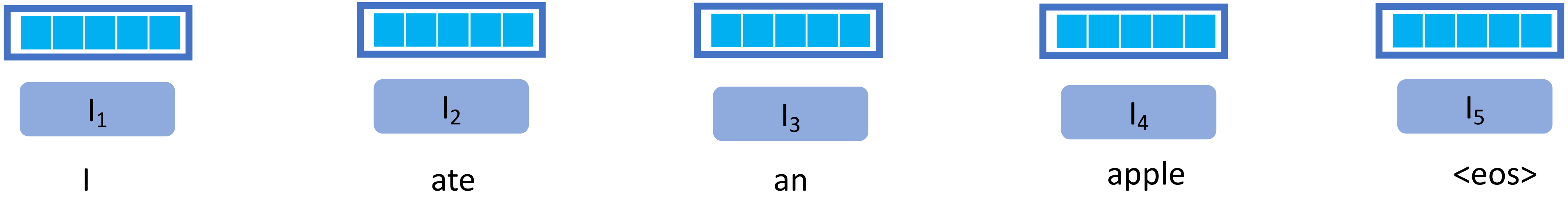




Query | Key Value Store | Key | Value

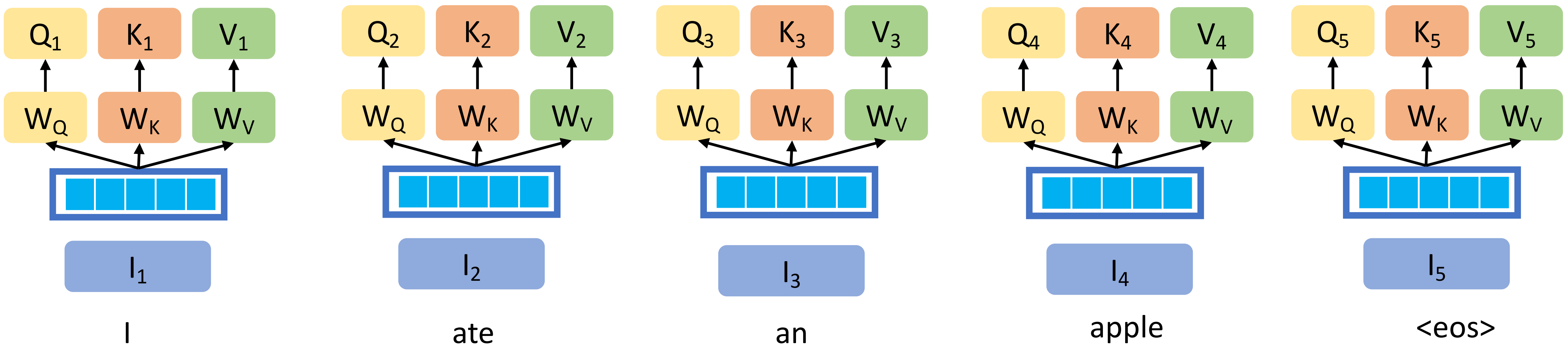
Q | QK^T | $softmax(\frac{QK^T}{\sqrt{d}})$ | $softmax(\frac{QK^T}{\sqrt{d}})V$

Attention



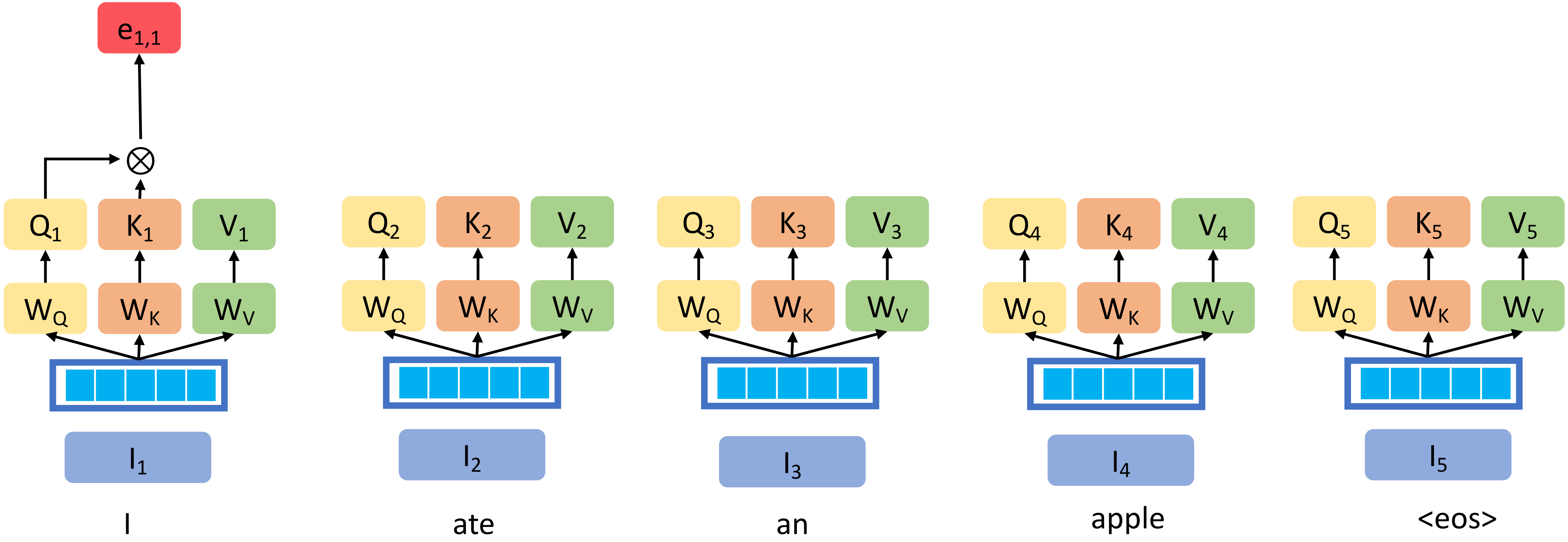
Dimensions across QKV have been dropped for brevity

Attention



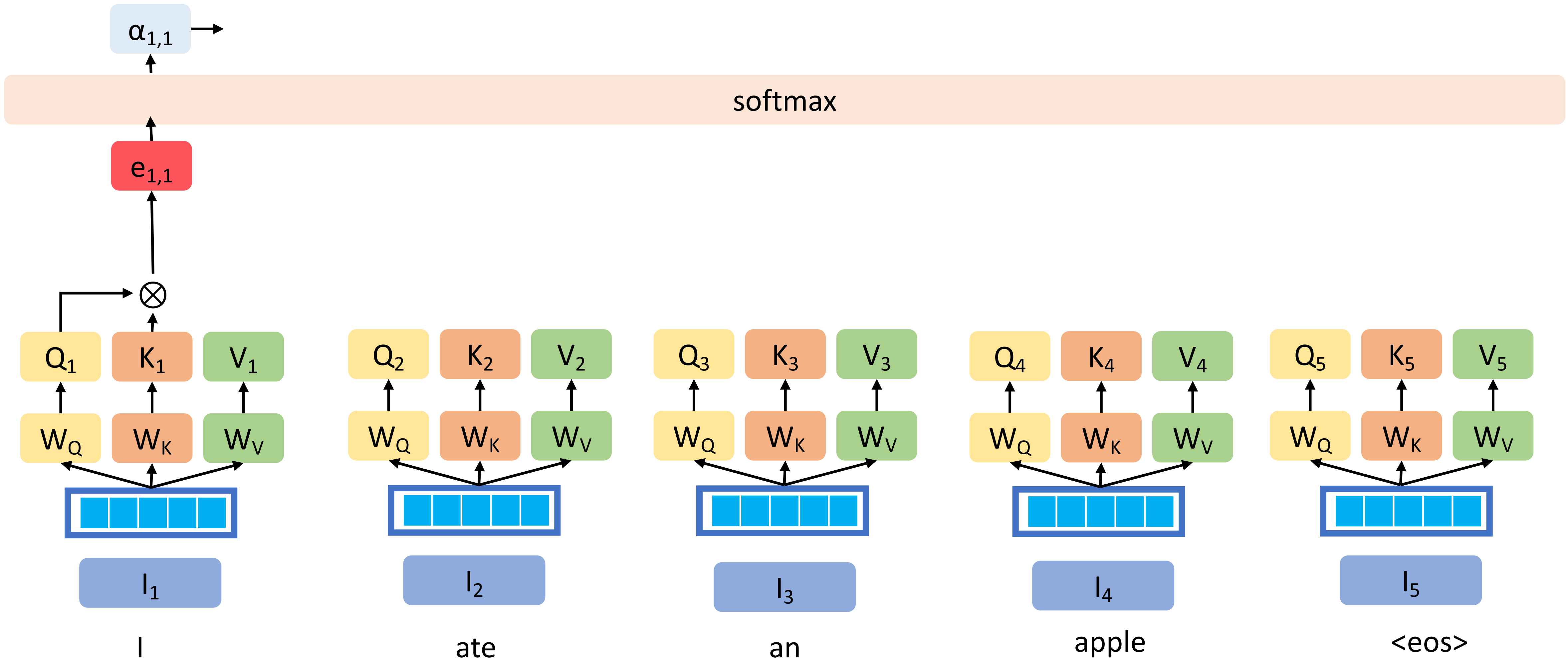
Dimensions across QKV have been dropped for brevity

Attention

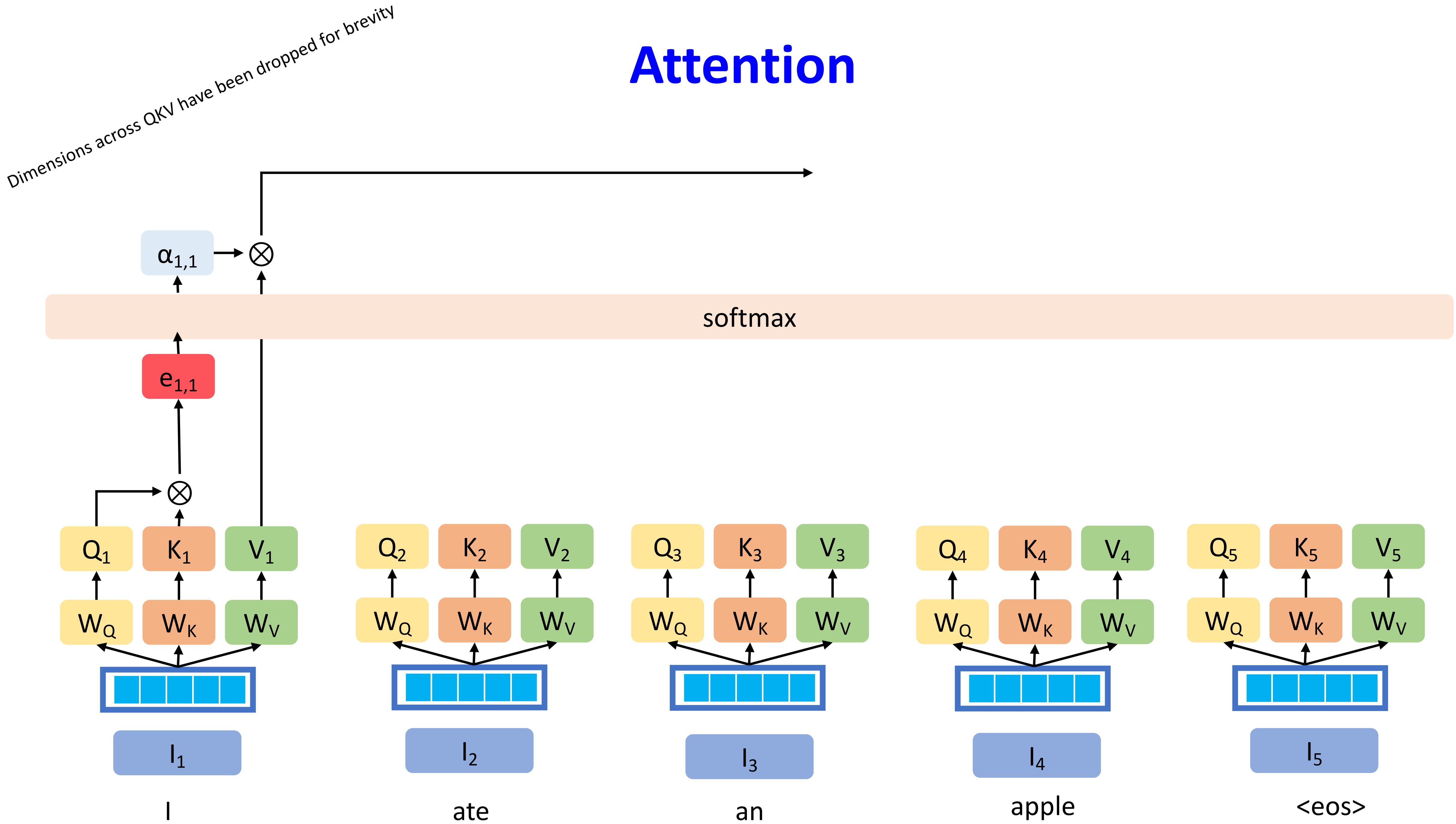


Attention

Dimensions across QKV have been dropped for brevity

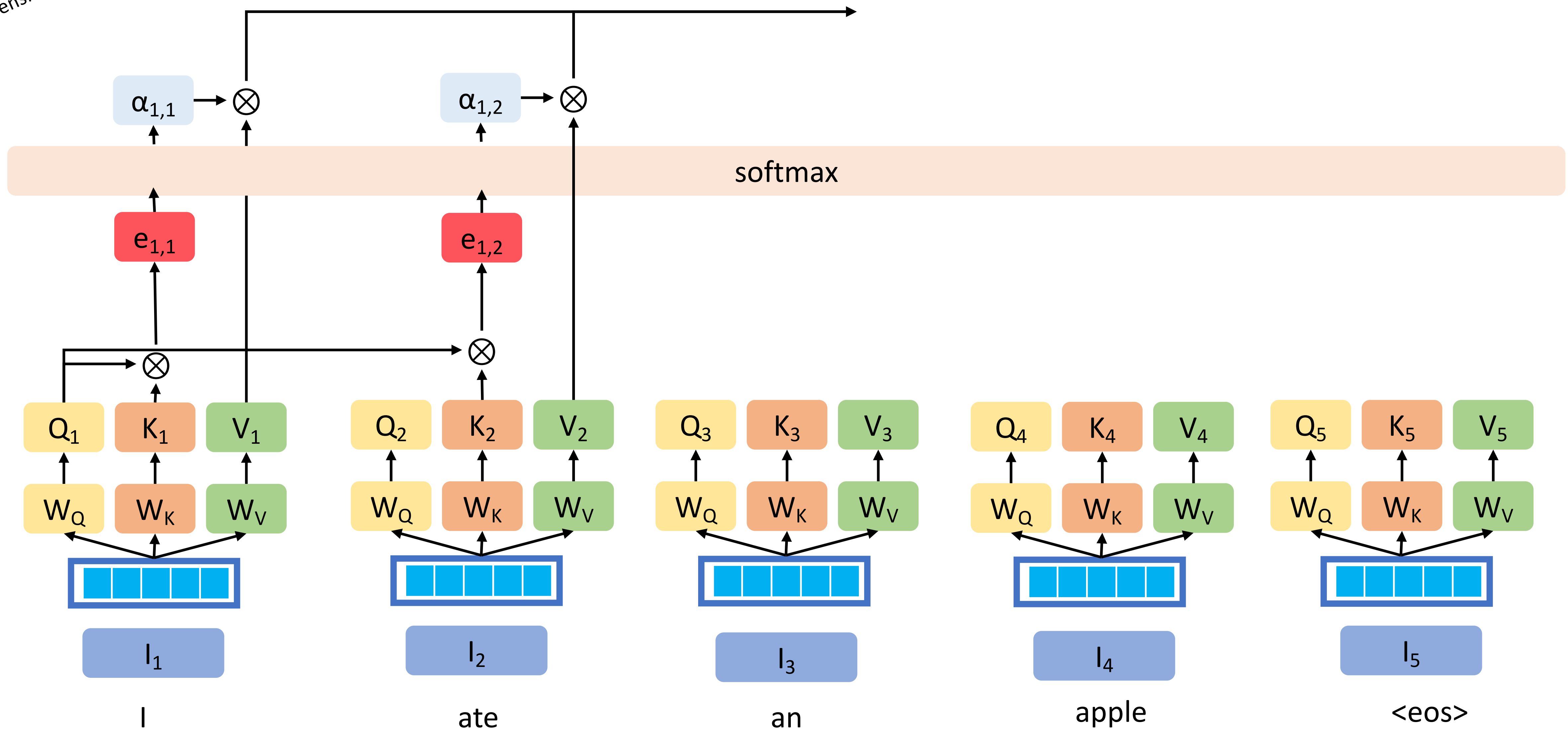


Attention


Dimensions across QKV have been dropped for brevity
$\alpha_{1,1}$
softmax
$e_{1,1}$
Q_1 K_1 V_1
Q_2 K_2 V_2
Q_3 K_3 V_3
Q_4 K_4 V_4
Q_5 K_5 V_5
W_Q W_K W_V
I_1 I_2 I_3 I_4 I_5
I
ate
an
apple
<eos>

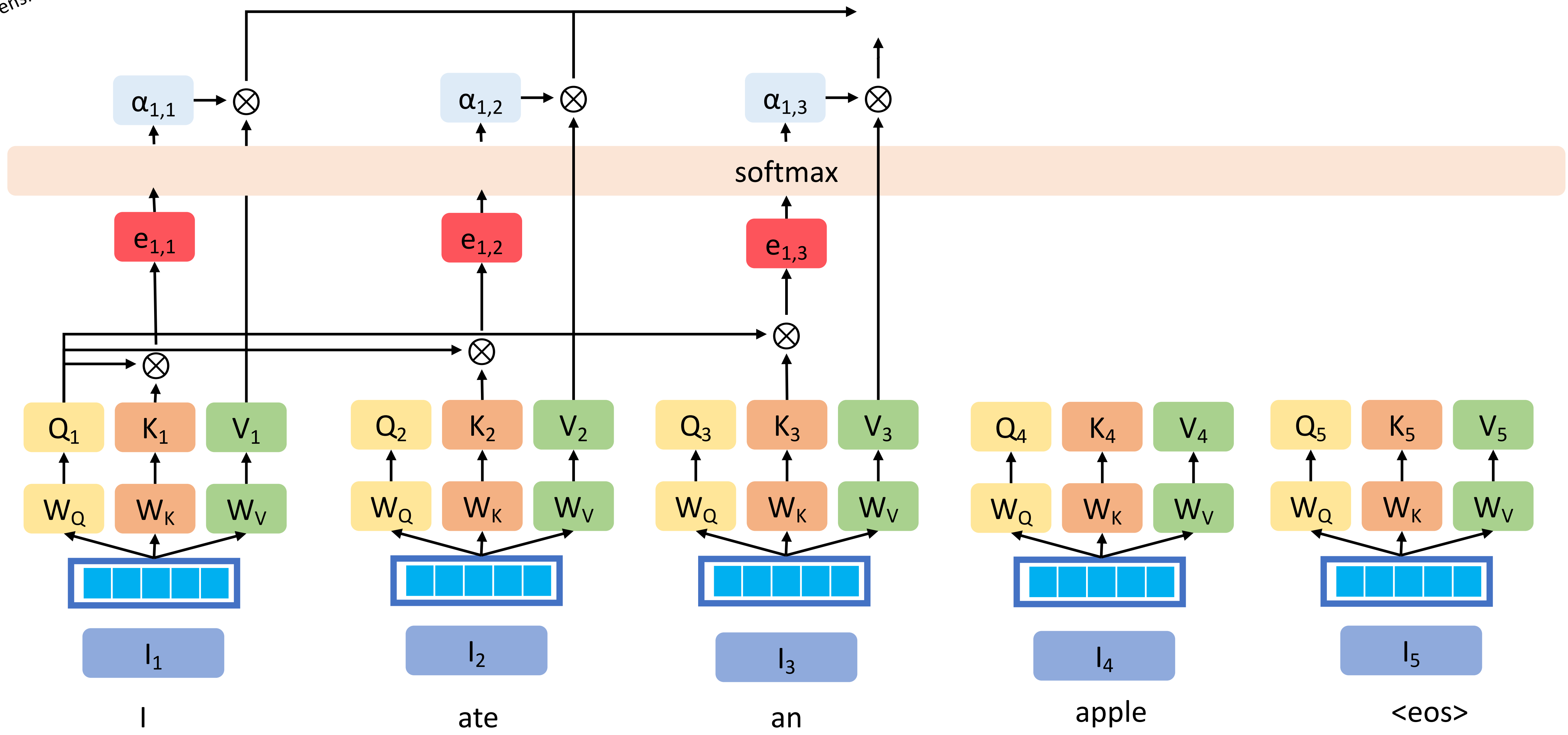
Attention

Dimensions across QKV have been dropped for brevity

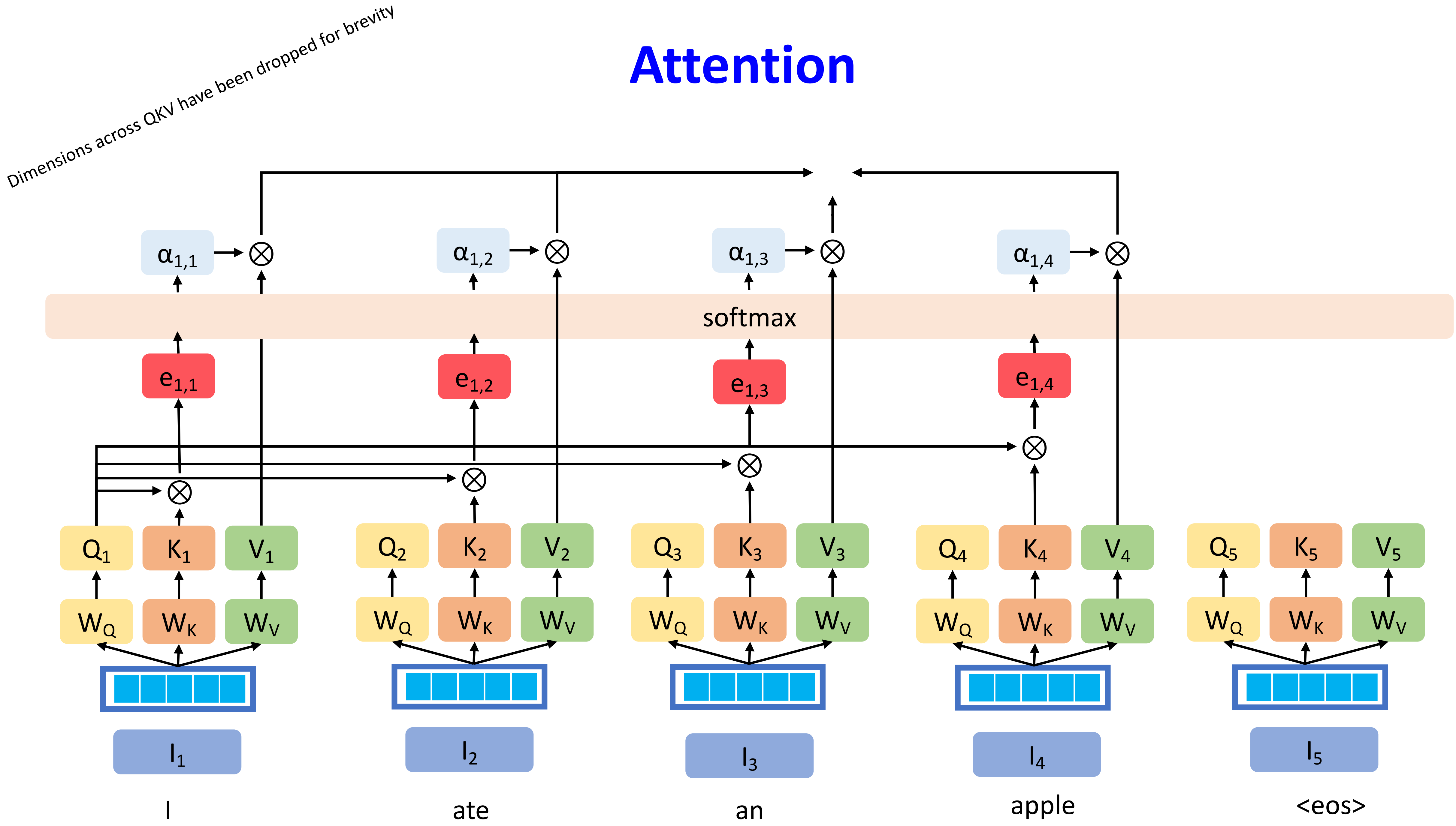


Attention

Dimensions across QKV have been dropped for brevity

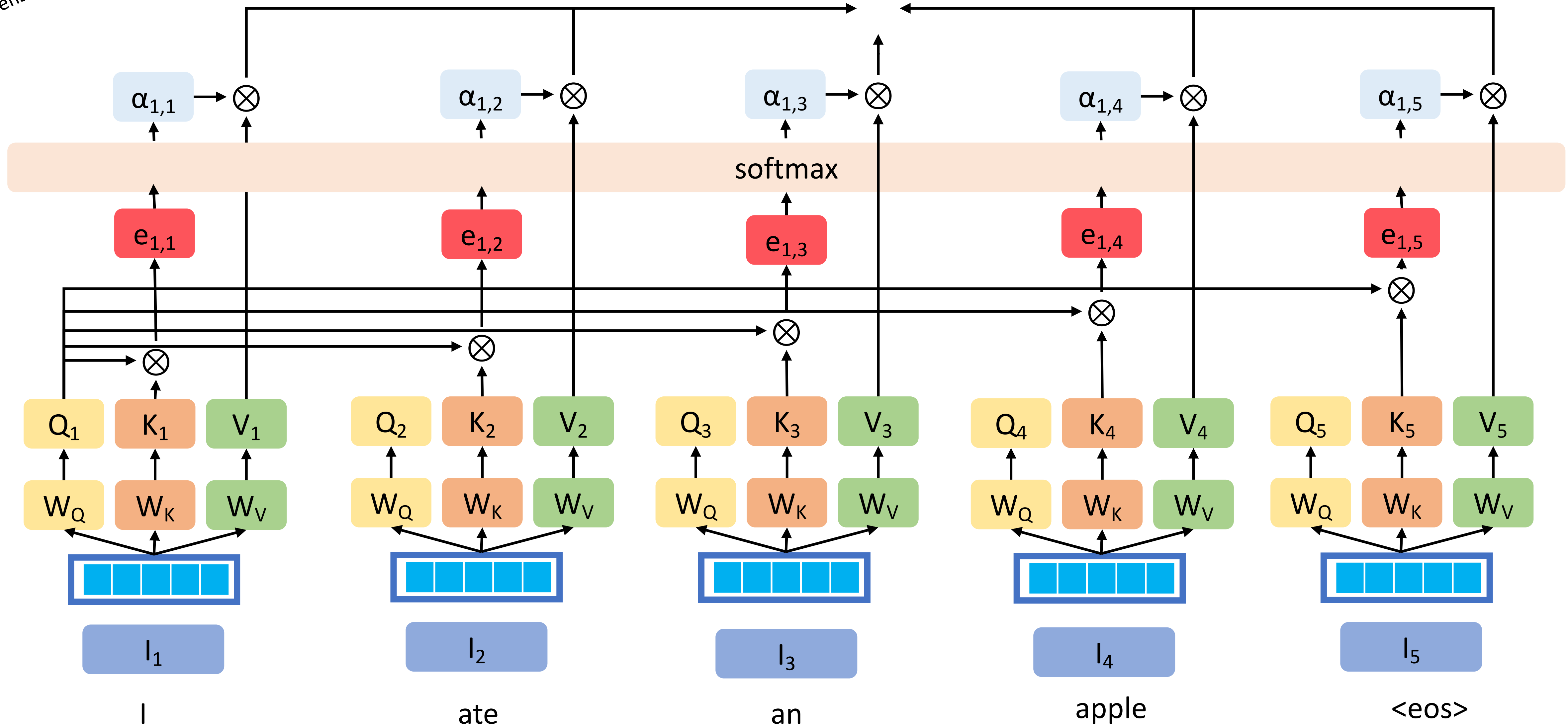


Attention


Dimensions across QKV have been dropped for brevity
softmax
α1,1
α1,2
α1,3
α1,4
e1,1
e1,2
e1,3
e1,4
Q1 K1 V1
Q2 K2 V2
Q3 K3 V3
Q4 K4 V4
Q5 K5 V5
WQ WK WV
I1
I2
I3
I4
I5
I
ate
an
apple
<eos>

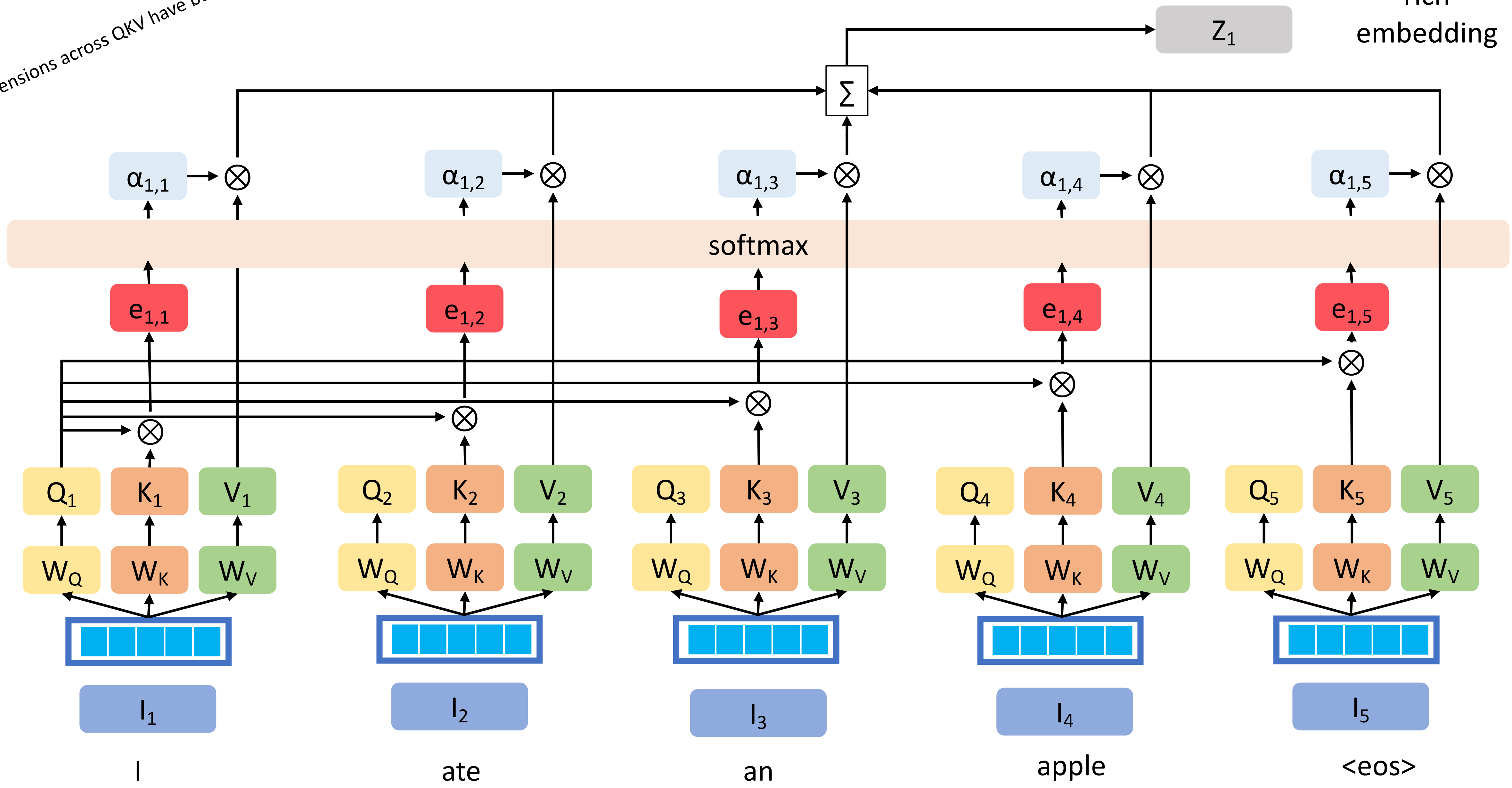
Attention

Dimensions across QKV have been dropped for brevity



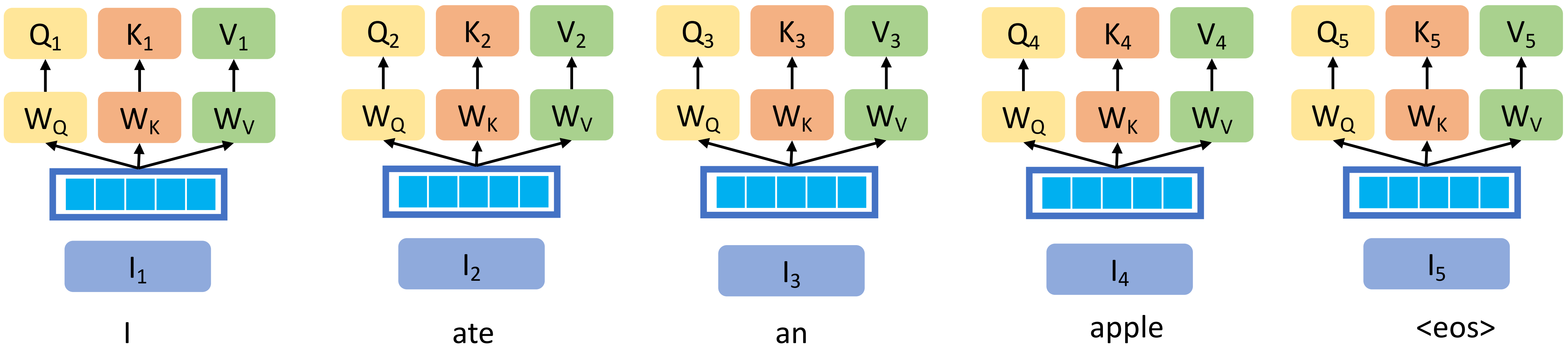
Attention

Dimensions across QKV have been dropped for brevity

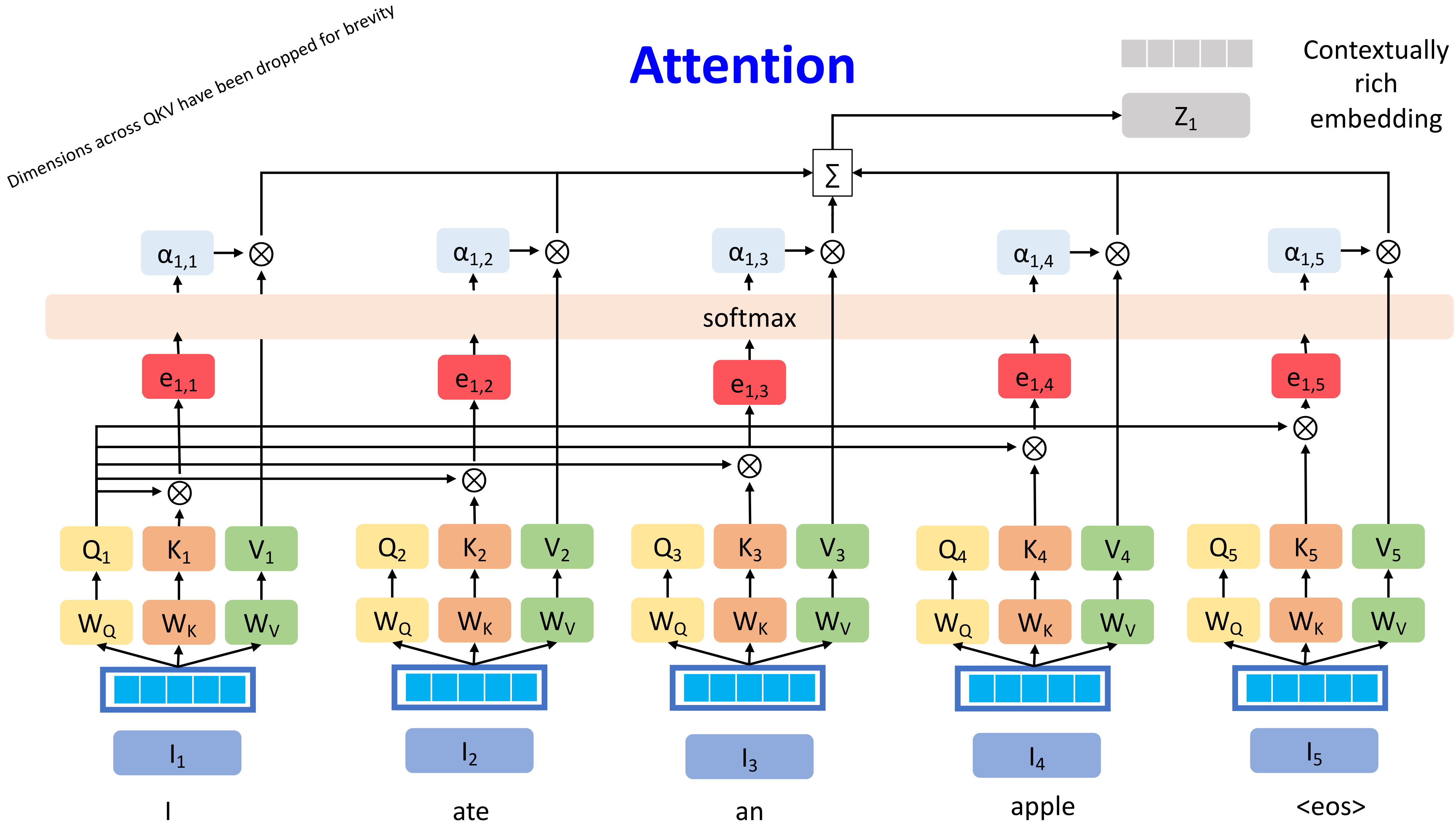


Dimensions across QKV have been dropped for brevity

Attention



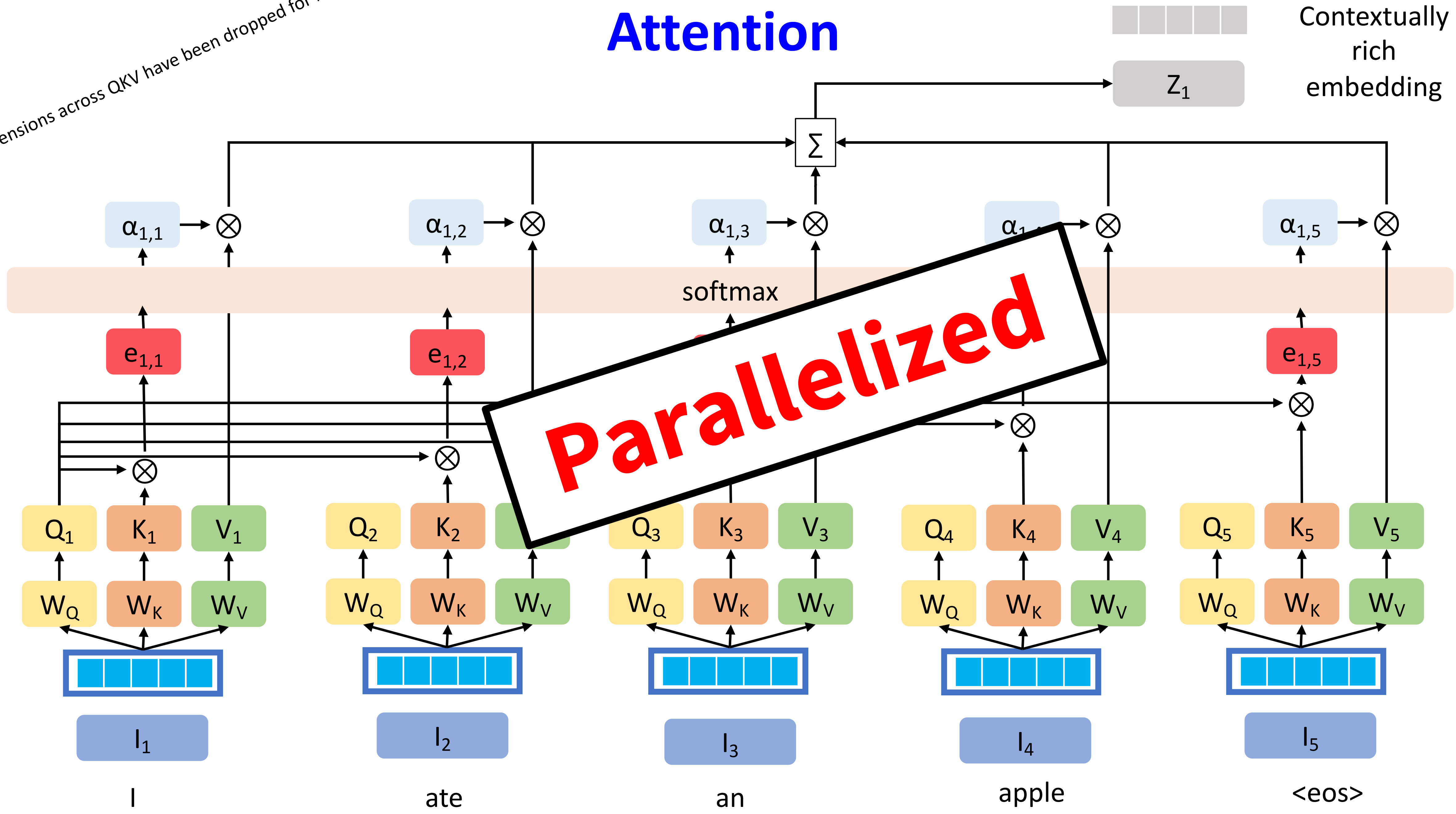
Attention



Dimensions across QKV have been dropped for brevity
Contextually rich embedding
Z_1
Σ
$\alpha_{1,1}$ $\alpha_{1,2}$ $\alpha_{1,3}$ $\alpha_{1,4}$ $\alpha_{1,5}$
softmax
$e_{1,1}$ $e_{1,2}$ $e_{1,3}$ $e_{1,4}$ $e_{1,5}$
Q_1 K_1 V_1 Q_2 K_2 V_2 Q_3 K_3 V_3 Q_4 K_4 V_4 Q_5 K_5 V_5
W_Q W_K W_V
I_1 I_2 I_3 I_4 I_5
I
ate
an
apple
<eos>

Attention

Dimensions across QKV have been dropped for brevity



Poll 1 @1296

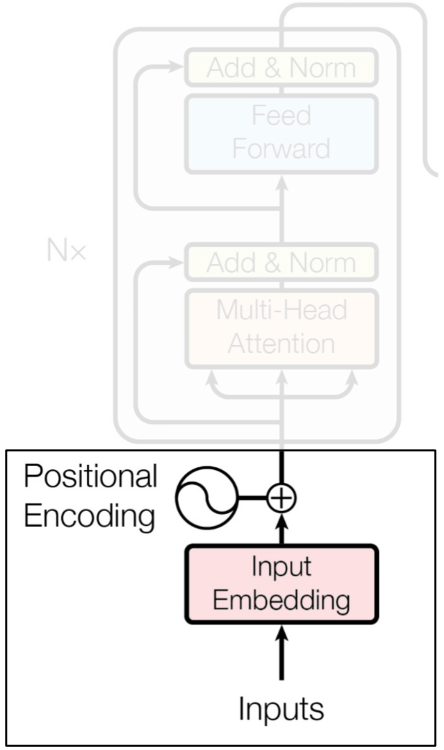
Which of the following are true about attention? (Select all that apply)

a. To calculate attention weights for input $\mathbf{I_2}$, you would use key $\mathbf{k_2}$, and all queries

b. To calculate attention weights for input $\mathbf{I_2}$, you would use query $\mathbf{q_2}$, and all keys

c. We scale the QK^T product to bring attention weights in the range of [0,1]

d. We scale the QK^T product to allow for numerical stability



Poll 1 @1296

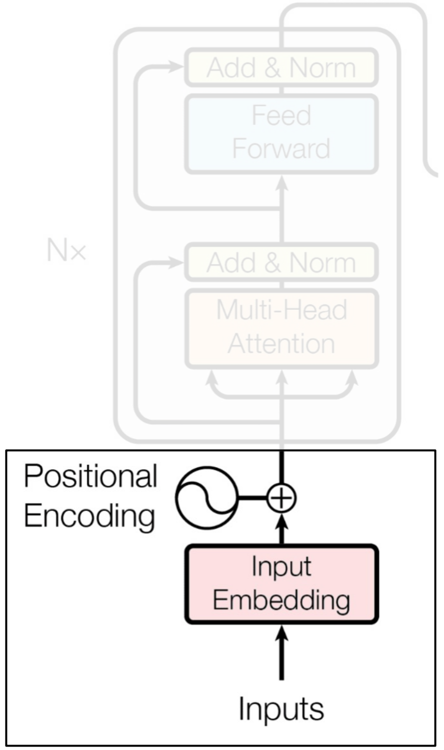
Which of the following are true about attention? (Select all that apply)

a. To calculate attention weights for input **I_2**, you would use key **k_2**, and all queries

b. To calculate attention weights for input I_2, you would use query q_2, and all keys

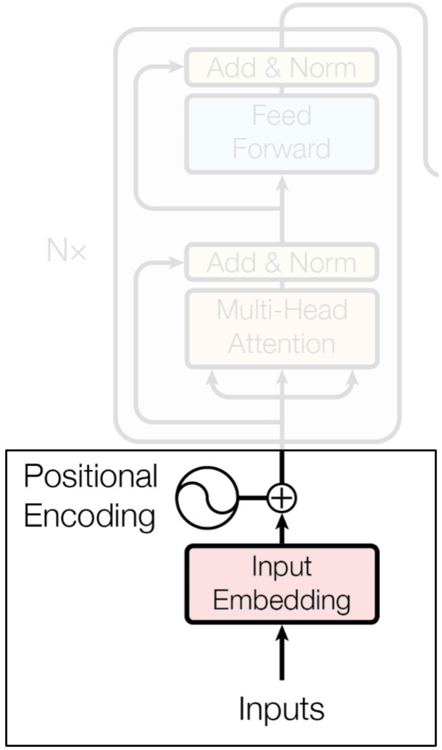
c. We scale the QK^T product to bring attention weights in the range of [0,1]

d. We scale the QK^T product to allow for numerical stability

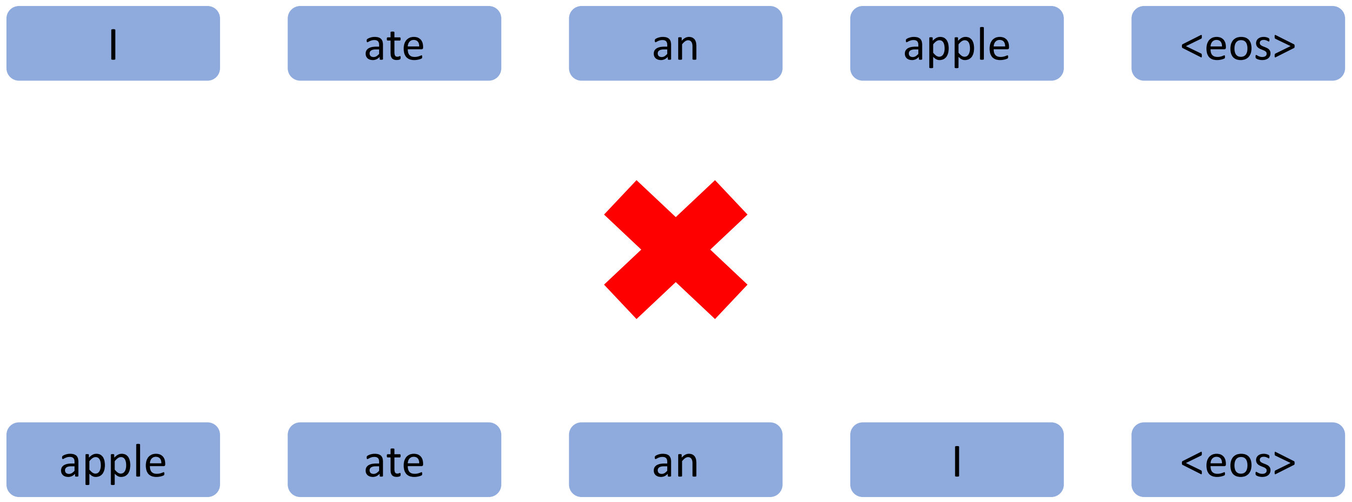


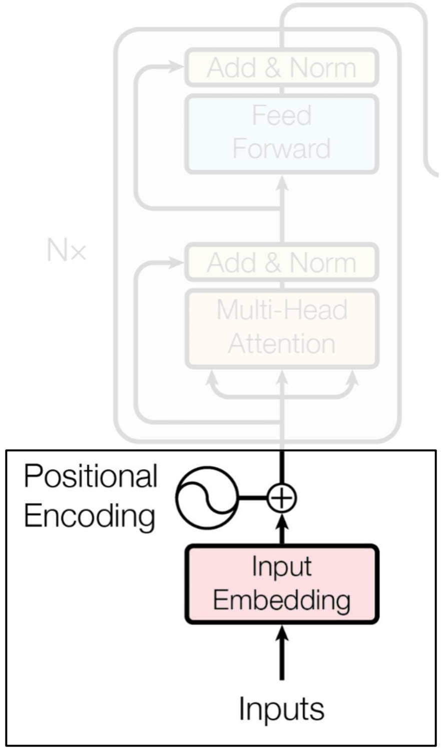
Positional Encoding

I	ate	an	apple	<eos>



Positional Encoding



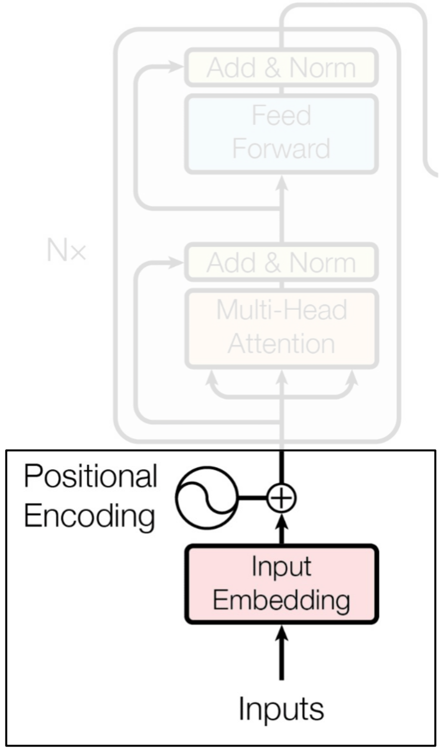


Positional Encoding

Positional Encoding

Requirements for Positional Encodings

- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic



Positional Encoding

Positional Encoding

Requirements for Positional Encodings

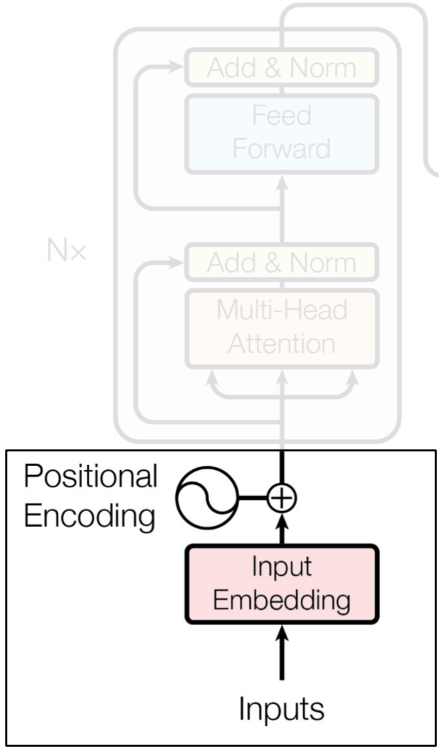
- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic

Possible Candidates :

$$P_{t+1} = P_t + \Delta c$$

$$P_{t+1} = e^{P_t \Delta c}$$

$$P_{t+1} = P_t^{\,t\Delta c}$$



Positional Encoding

Positional Encoding

Requirements for Positional Encodings

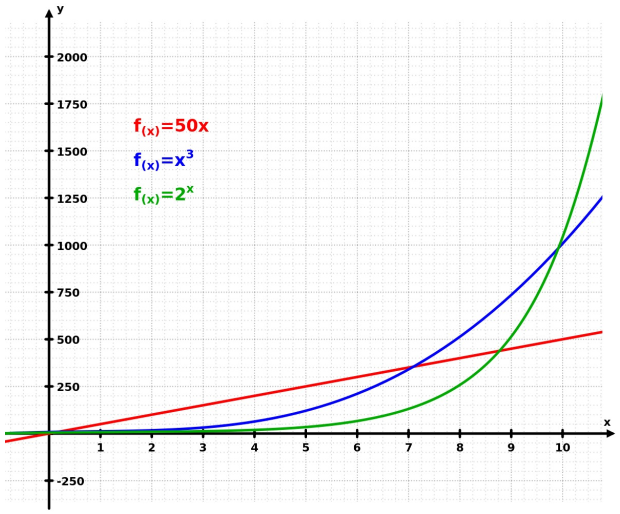
- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic

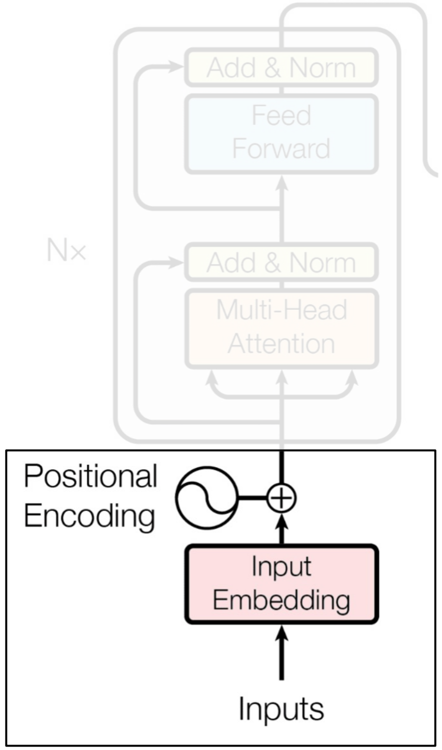
Possible Candidates :

$$P_{t+1} = P_t + \Delta c$$

$$P_{t+1} = e^{P_{t_\Delta} c}$$

$$P_{t+1} = P_t^{\,t\Delta c}$$





Positional Encoding

Positional Encoding

Requirements for Positional Encodings

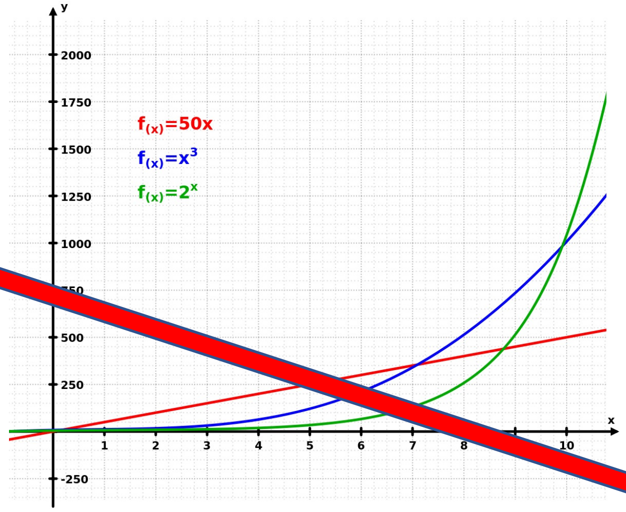
- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic
- **Bounded**

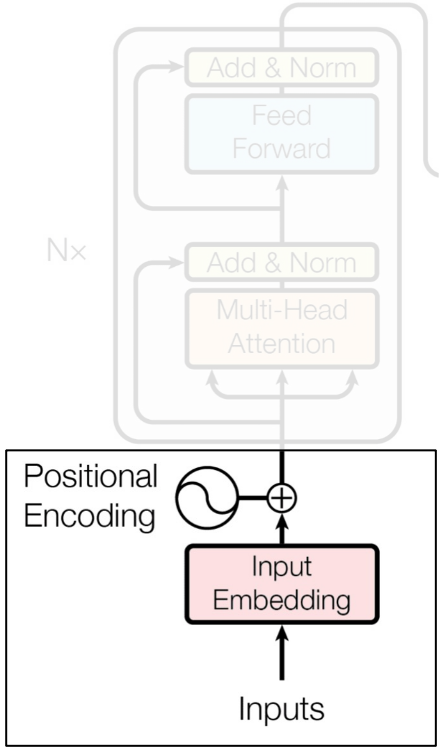
Possible Candidates :

$$P_{t+1} = P_t + \Delta c$$

$$P_{t+1} = e^{P_t \Delta c}$$

$$P_{t+1} = P_t^{\,t\Delta c}$$





Positional Encoding

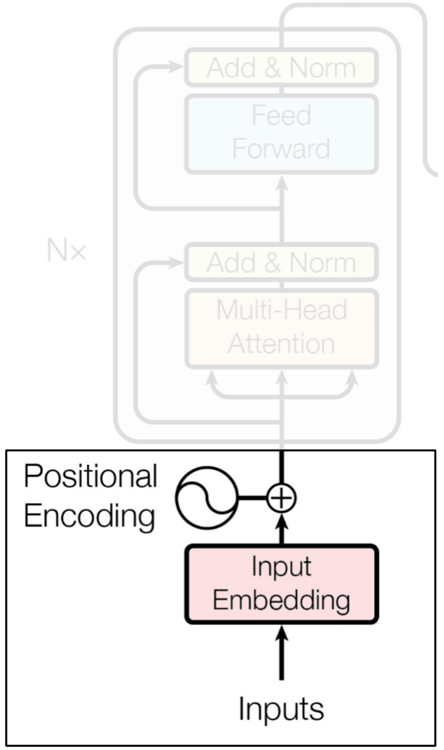
Positional Encoding

Requirements for Positional Encodings

- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic
- **Bounded**

Possible Candidates :

$$\mathbf{P(t + t') = M^{t'} \times P(t)}$$



Positional Encoding

Positional Encoding

Requirements for Positional Encodings

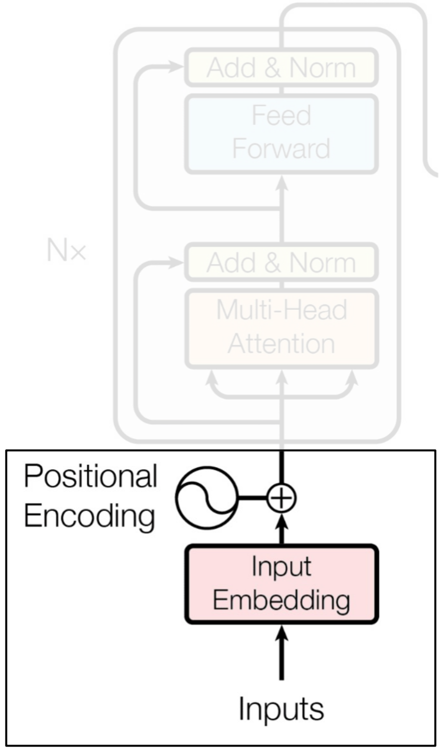
- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic
- **Bounded**

Possible Candidates :

$$\mathbf{P(t + t') = M^{t'} \times P(t)}$$

M ?

1. **Should be a unitary matrix**
2. **Magnitudes of eigen value should be 1 -> norm preserving**



Positional Encoding

Positional Encoding

Requirements for Positional Encodings

- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – not cyclic
- **Bounded**

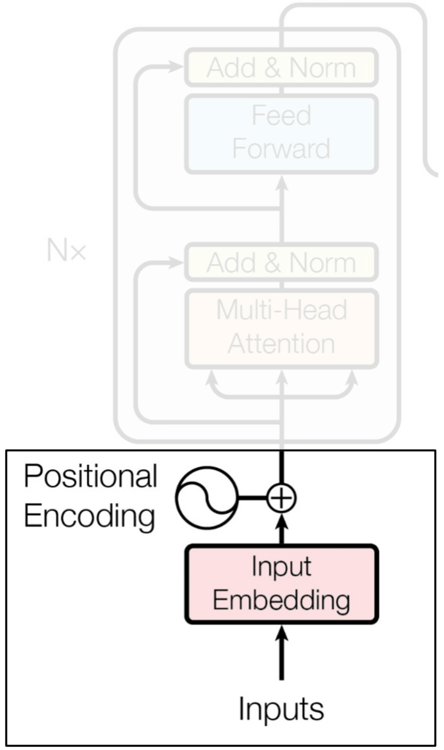
Possible Candidates :

$$P(t + t') = M^{t'} \times P(t)$$

M

1. **The matrix can be learnt**
2. **Produces unique rotated embeddings each time**

Positional Encoding



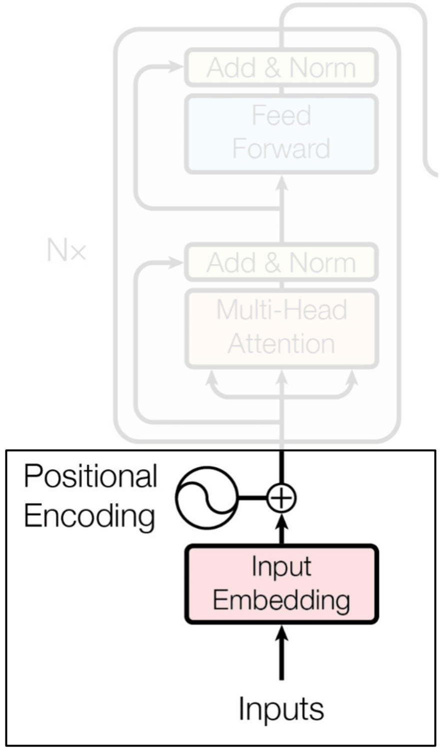
Rotary Positional Embedding

ROFORMER: ENHANCED TRANSFORMER WITH ROTARY POSITION EMBEDDING

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \begin{pmatrix} \cos m\theta & -\sin m\theta \\ \sin m\theta & \cos m\theta \end{pmatrix} \begin{pmatrix} W^{(11)}_{\{q,k\}} & W^{(12)}_{\{q,k\}} \\ W^{(21)}_{\{q,k\}} & W^{(22)}_{\{q,k\}} \end{pmatrix} \begin{pmatrix} x^{(1)}_m \\ x^{(2)}_m \end{pmatrix}$$

Table 2: Comparing RoFormer and BERT by fine tuning on downstream GLEU tasks.

Model	MRPC	SST-2	QNLI	STS-B	QQP	MNLI(m/mm)
BERTDevlin et al. [2019]	88.9	93.5	90.5	85.8	71.2	84.6/83.4
RoFormer	**89.5**	90.7	88.0	**87.0**	**86.4**	80.2/79.8



REF: Rotary Positional Embeddings

Positional Encoding

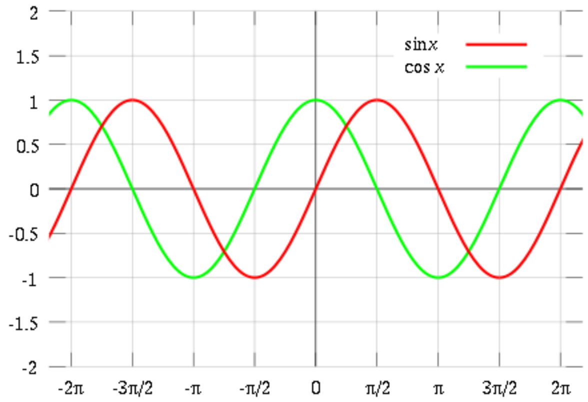
Requirements for Positional Encodings

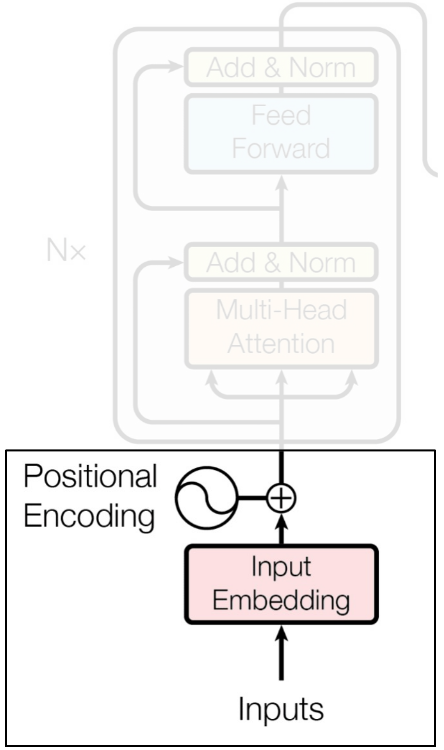
- Some representation of time ? (like **seq2seq** ?)
- Should be unique for each position – **not cyclic**
- Bounded

Actual Candidates :

*sine(**g(t)**)*

*cosine(**g(t)**)*





Positional Encoding

Positional Encoding

Requirements for g(t)

- *Must have same dimensions as input embeddings*
- *Must produce overall unique encodings*

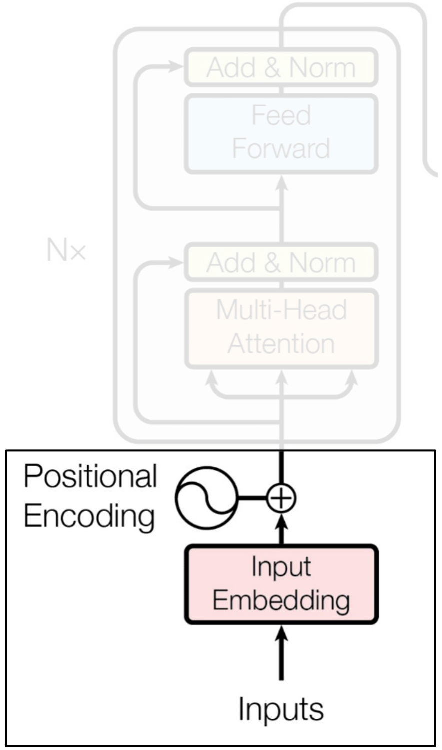
pos -> idx of the token in input sentence

i -> i^{th} dimension out of d

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

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

Positional Encoding



Positional Encoding

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

Requirements for g(t)

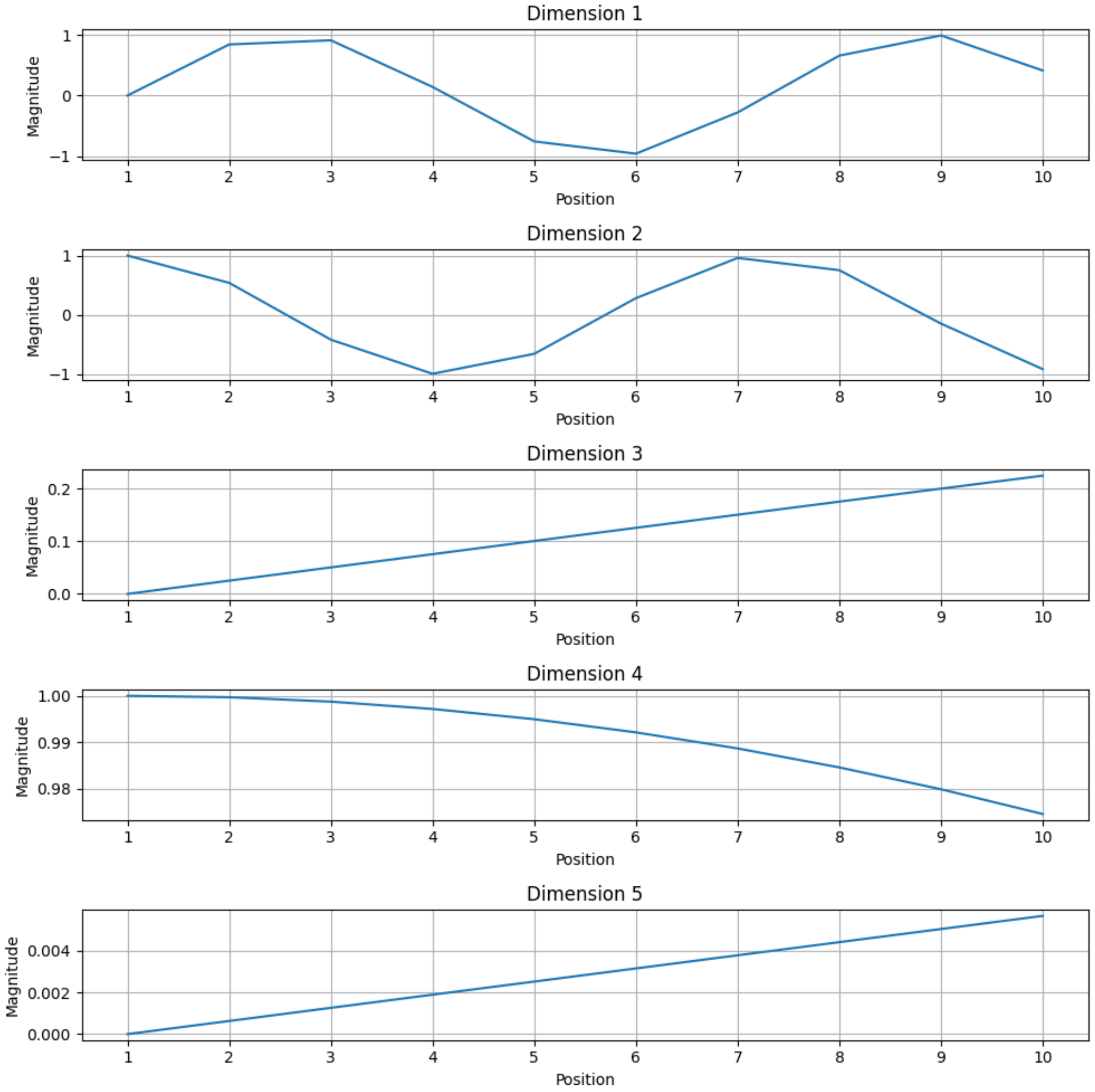
- *Must have same dimensions as input embeddings*
- *Must produce overall unique encodings*

pos -> idx of the token in input sentence

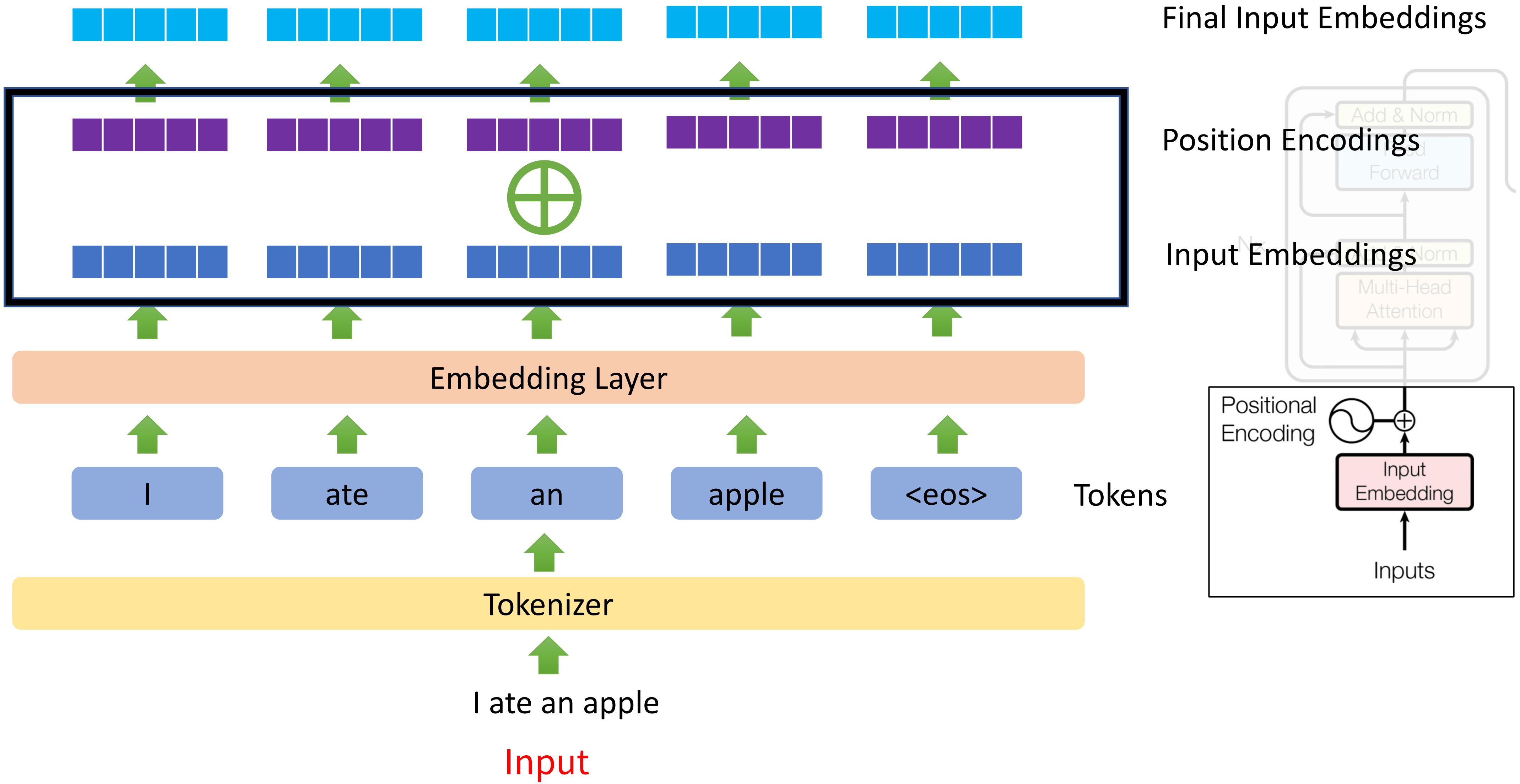
i -> i^{th} dimension out of d

Positional Encoding:

	0	1	2	3	4
Dim 1	0.000	0.841	0.909	0.141	-0.757
Dim 2	1.000	0.540	-0.416	-0.990	-0.654
Dim 3	0.000	0.025	0.050	0.075	0.100
Dim 4	1.000	1.000	0.999	0.997	0.995
Dim 5	0.000	0.001	0.001	0.002	0.003

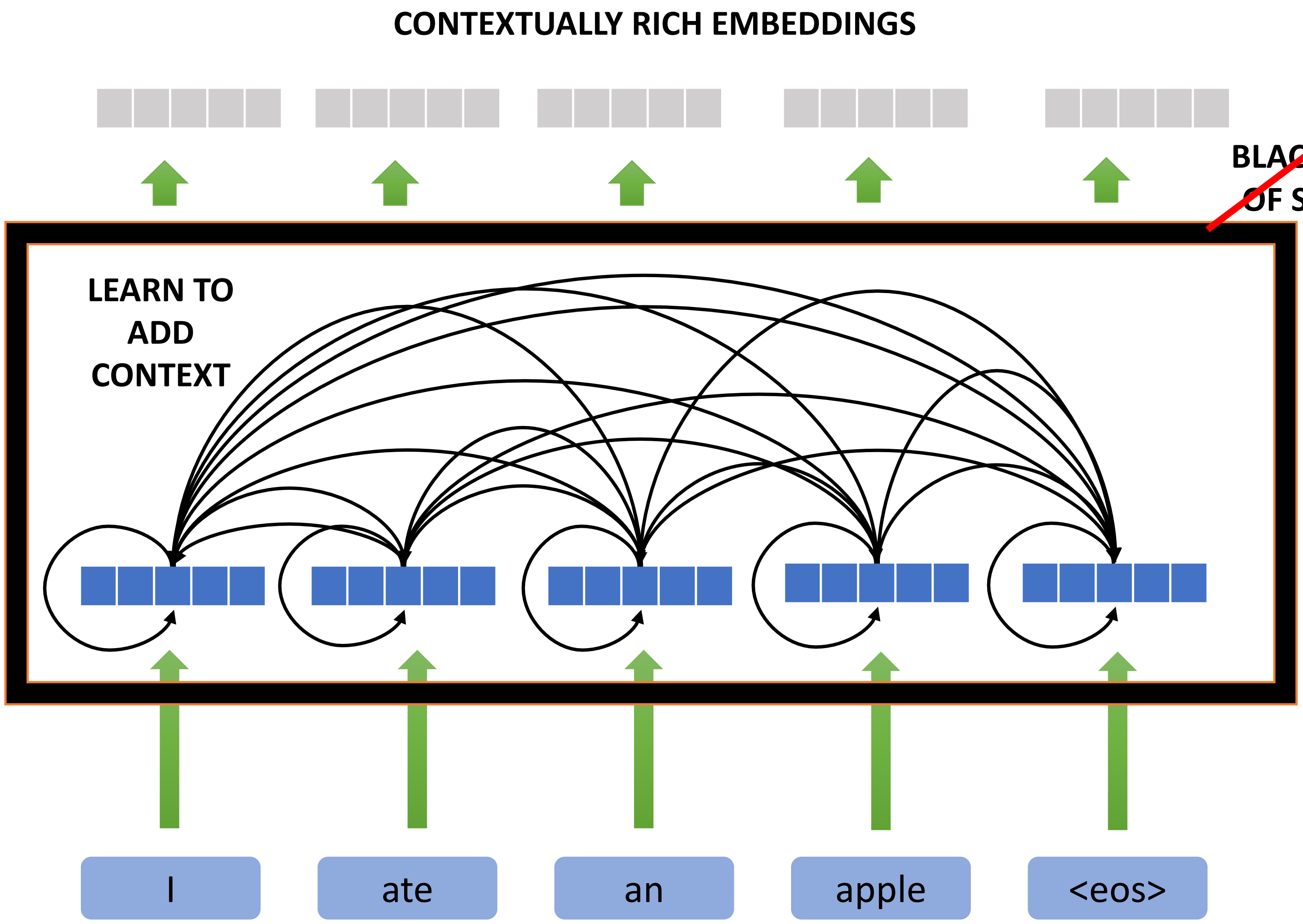


Positional Encoding

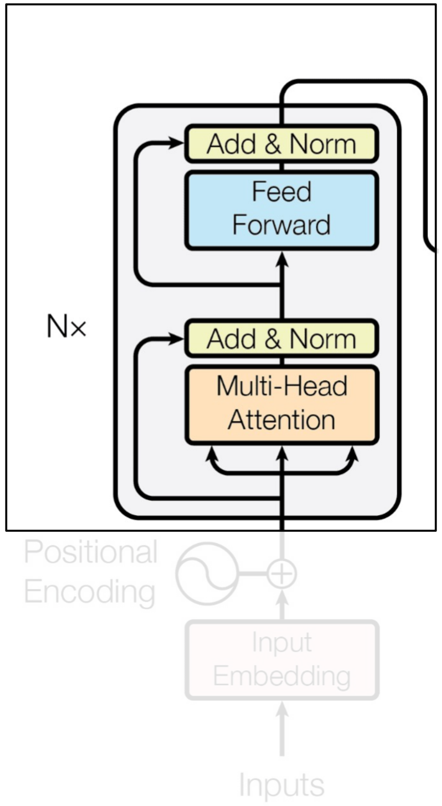


Encoder

$\alpha_{[\, i\ j\,]}$ Σ



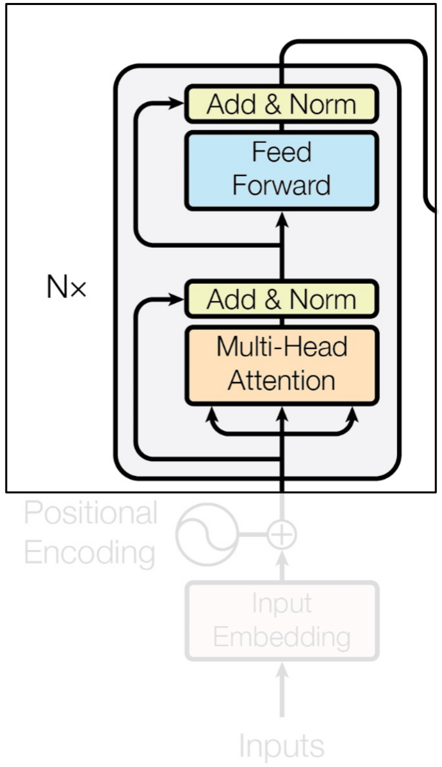
BLACK BOX
OF SORTS



Self Attention

From lecture 18:

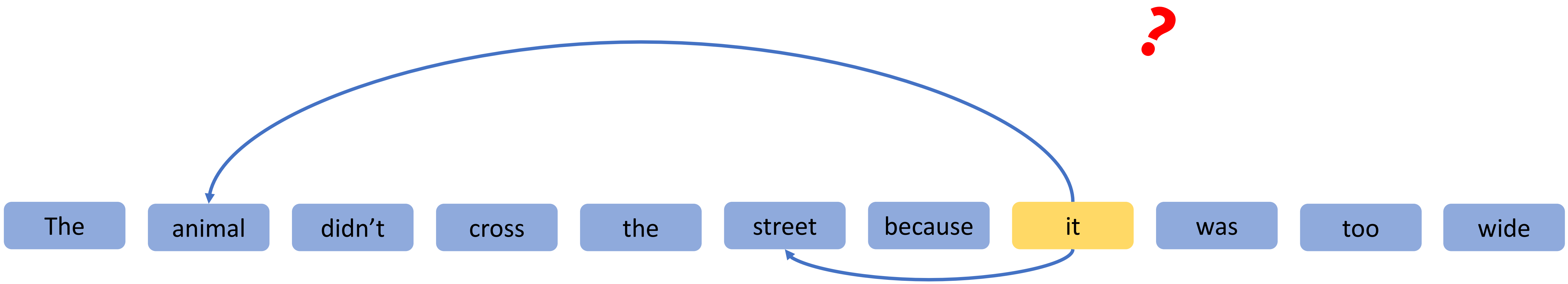
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



Self Attention

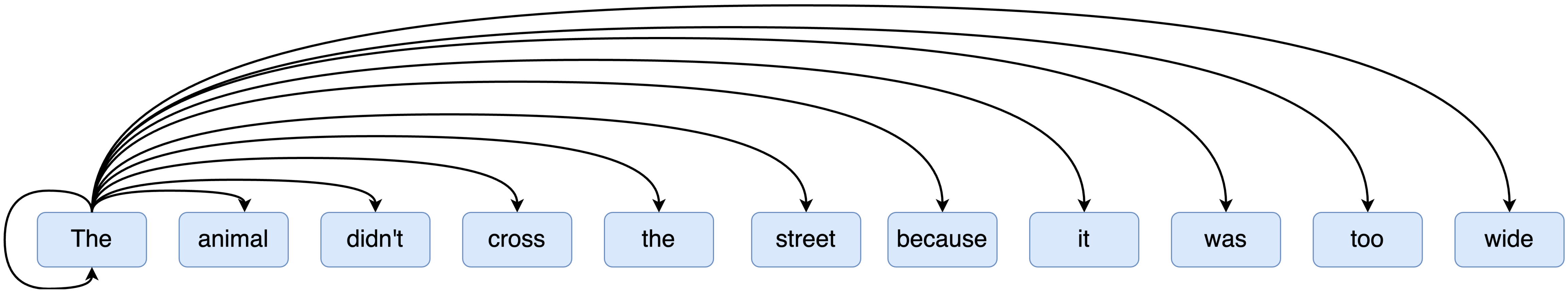
The animal didn’t cross the street because it was too wide

Self Attention

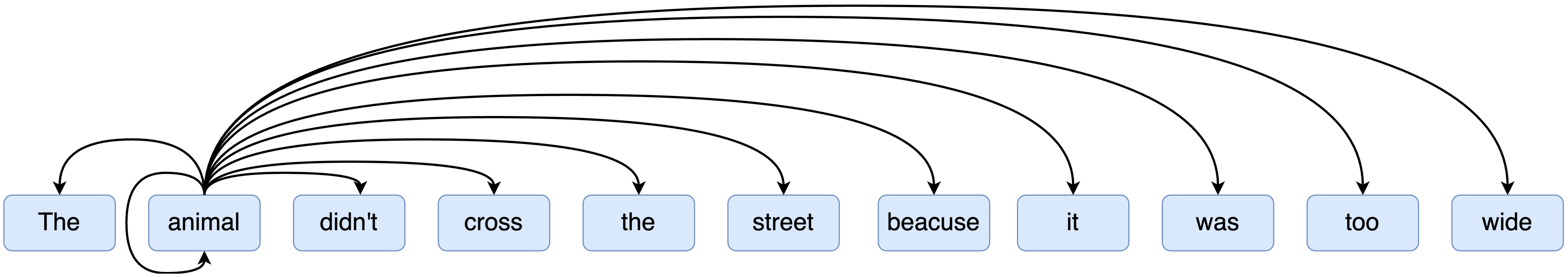


coreference resolution ?

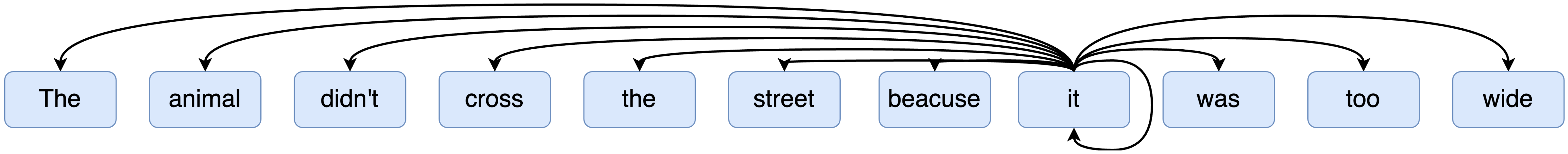
Self Attention



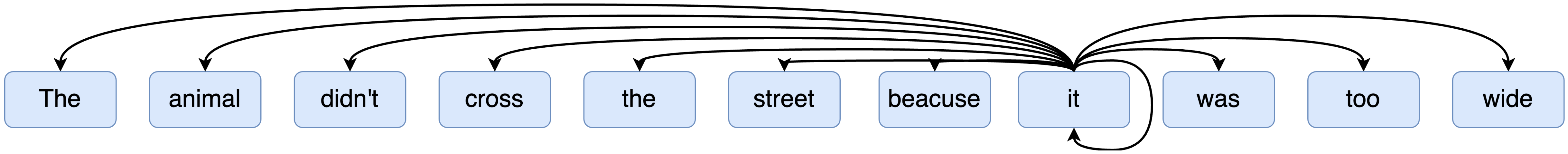
Self Attention



Self Attention



Self Attention

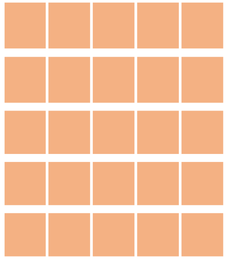


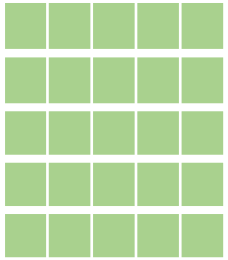
SELF

Query Inputs = Key Inputs = Value Inputs

Self Attention

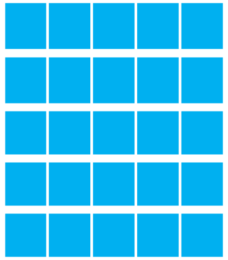
$R^{d_{model} \times d_{model}}$



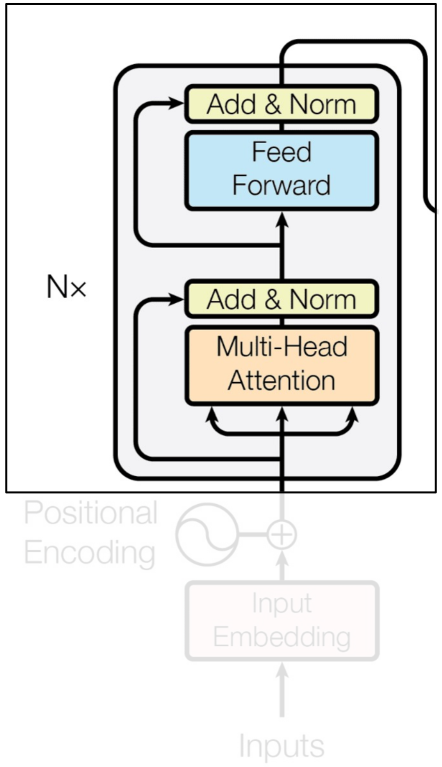


W_Q W_K W_V

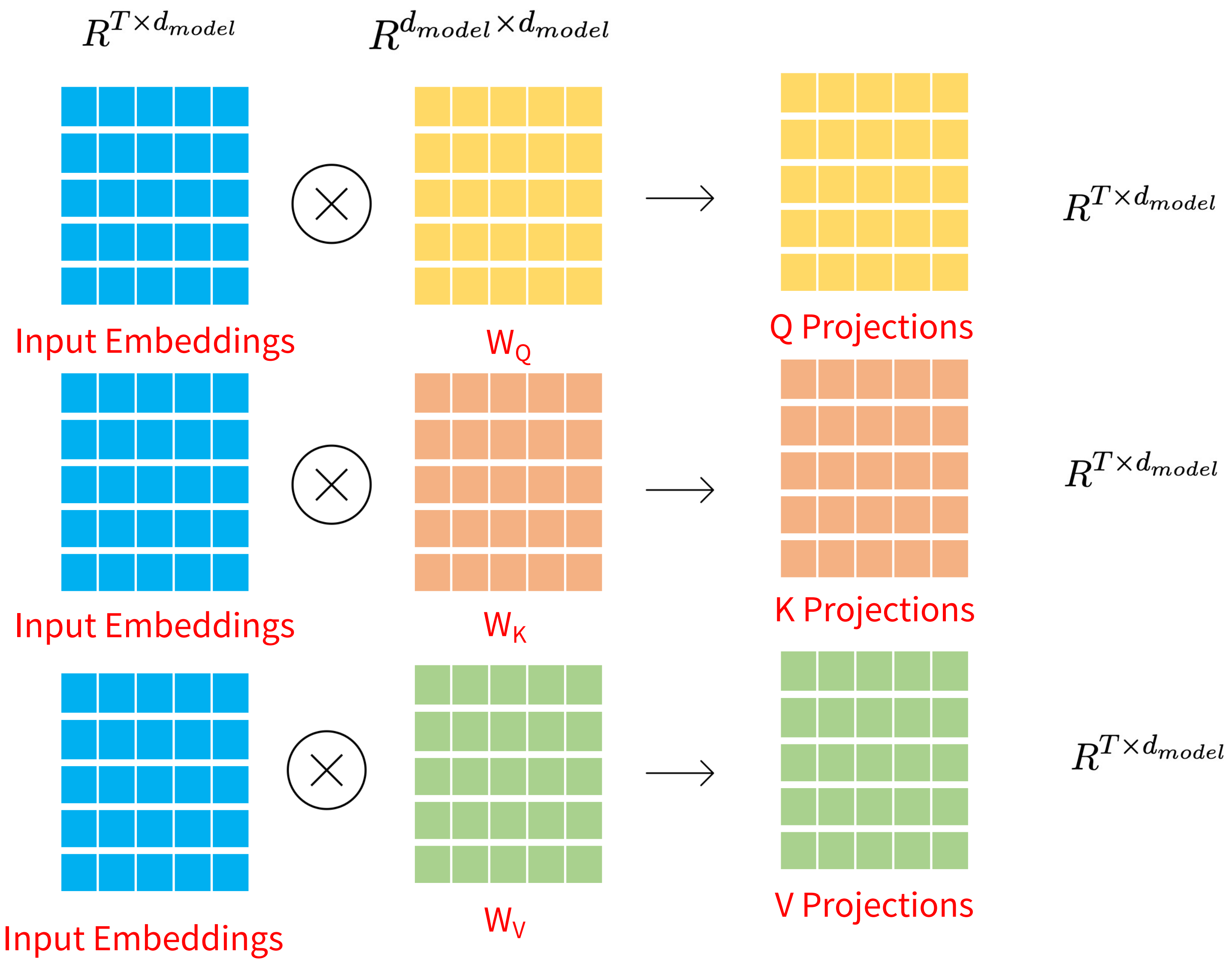
$R^{T \times d_{model}}$

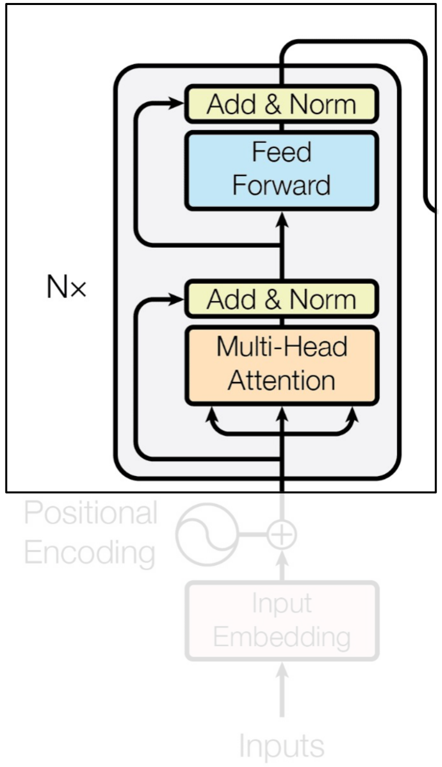


Input Embeddings



Self Attention



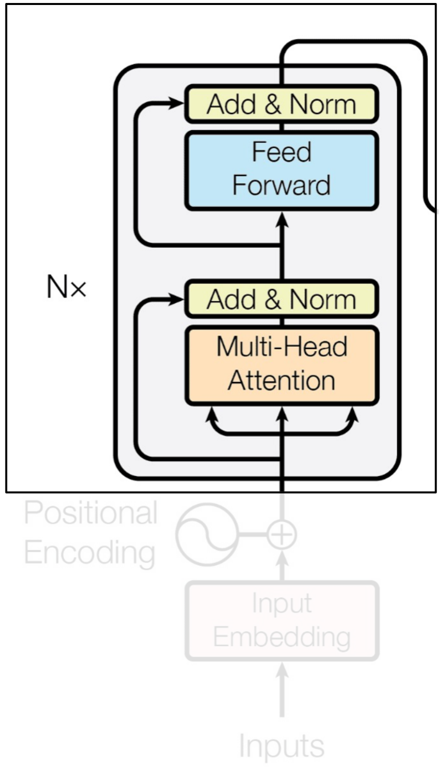


Self Attention

$R^{T \times d_{model}}$ $R^{d_{model} \times T}$

$$\text{softmax}\left(\frac{Q_{\text{Projection}} \otimes \left(K_{\text{Projection}} \right)^{T}}{\sqrt{d_{model}}} \right)$$

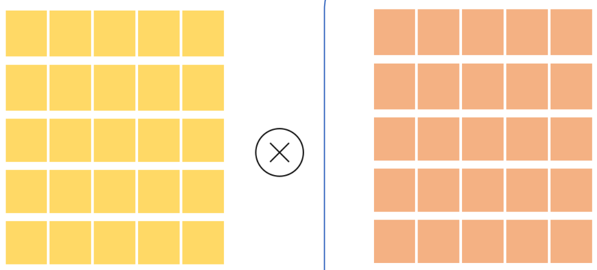
$R^{T \times T}$



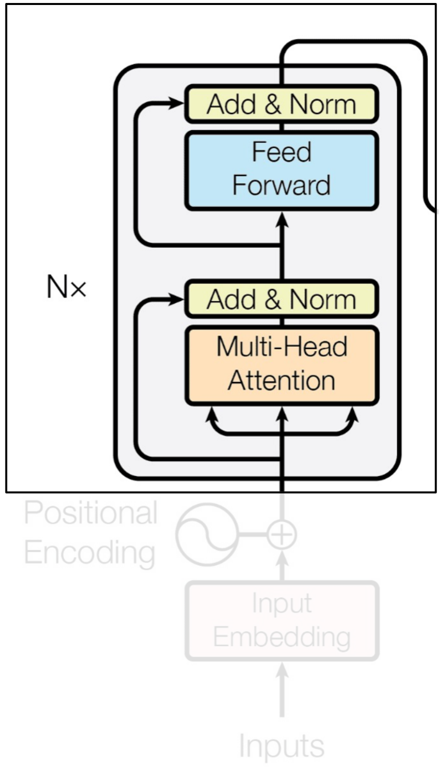
Self Attention

$\Theta(T^2 \times d_{model})$

$$\text{softmax}\left(\frac{\underset{Q_{Projection}}{R^{T\times d_{model}}} \otimes \left(\underset{K_{Projection}}{R^{d_{model}\times T}}\right)^{T}}{\sqrt{d_{model}}} \right)$$



$R^{T\times T}$



Self Attention

$\Theta(T^2 \times d_{model})$

softmax

$R^{T \times d_{model}}$ $R^{d_{model} \times T}$

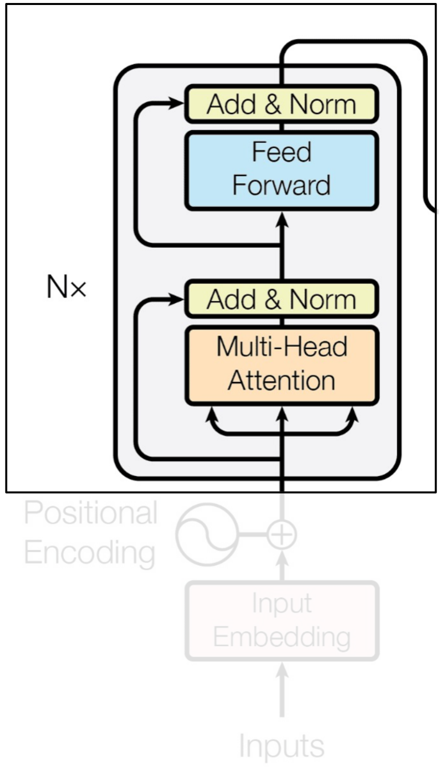
T

$R^{T \times d_{model}}$

$Q_{Projection}$ $\otimes$ $K_{Projection}$ $\otimes$ $V_{Projection}$

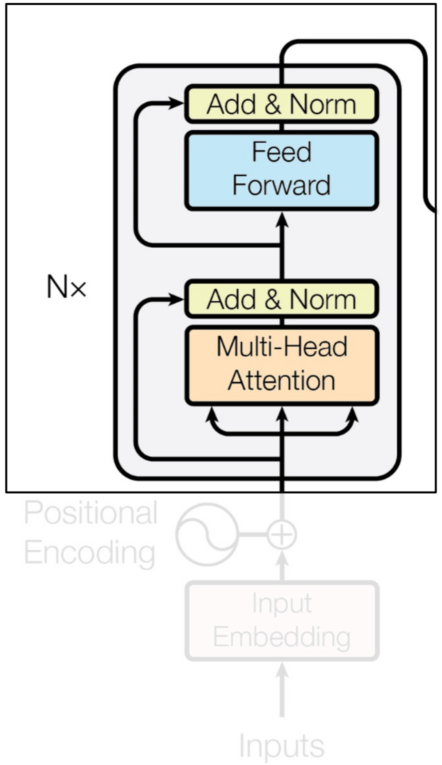
$\sqrt{d_{model}}$

$R^{T \times T}$



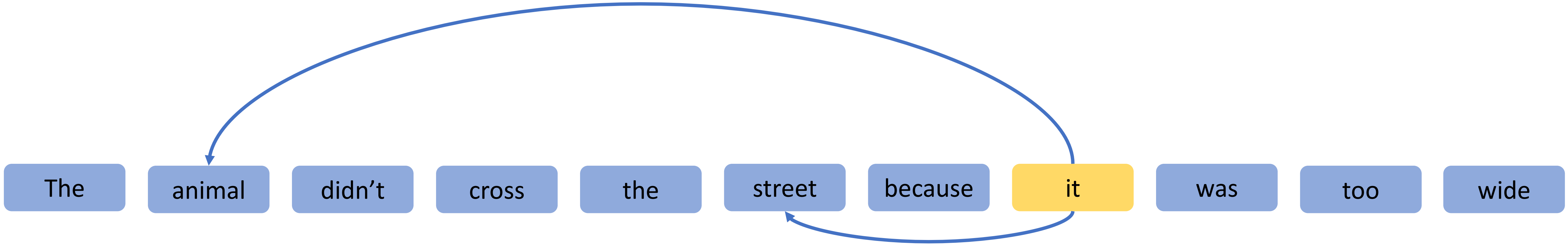
Self Attention

$R^{T \times d_{model}}$



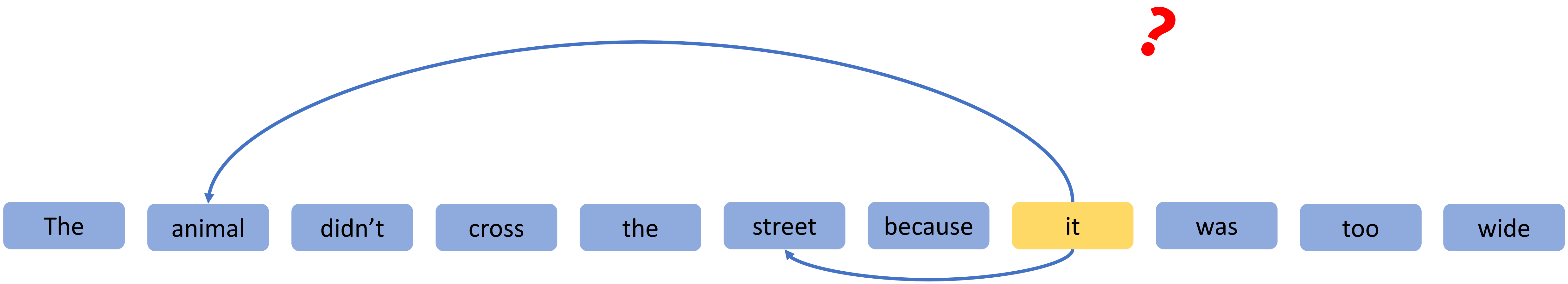
Attention: Z

Self Attention



coreference resolution

Self Attention



Sentence boundaries ?

coreference resolution

Context ?

Semantic relationships ?

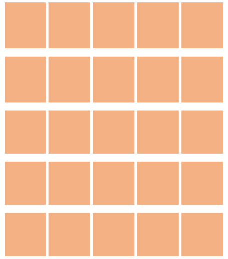
Part of Speech ?

Comparisons ?

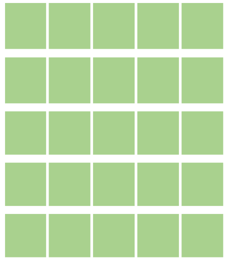
Self Attention

$R^{d_{model} \times d_{model}}$

W_Q

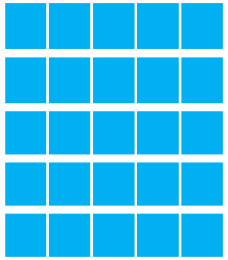


W_K

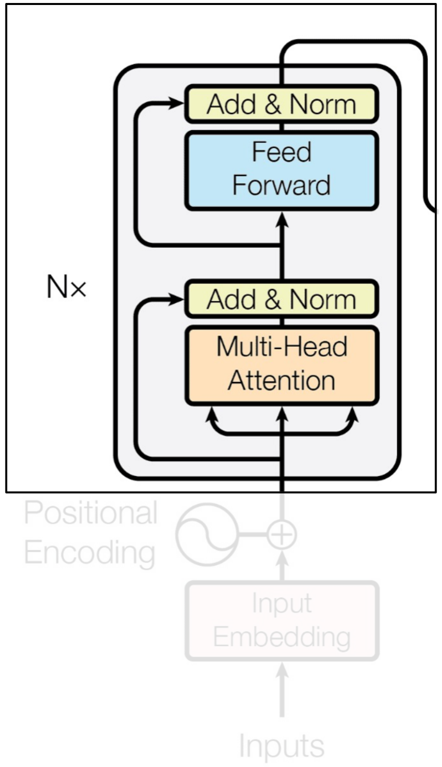


W_V

$R^{T \times d_{model}}$

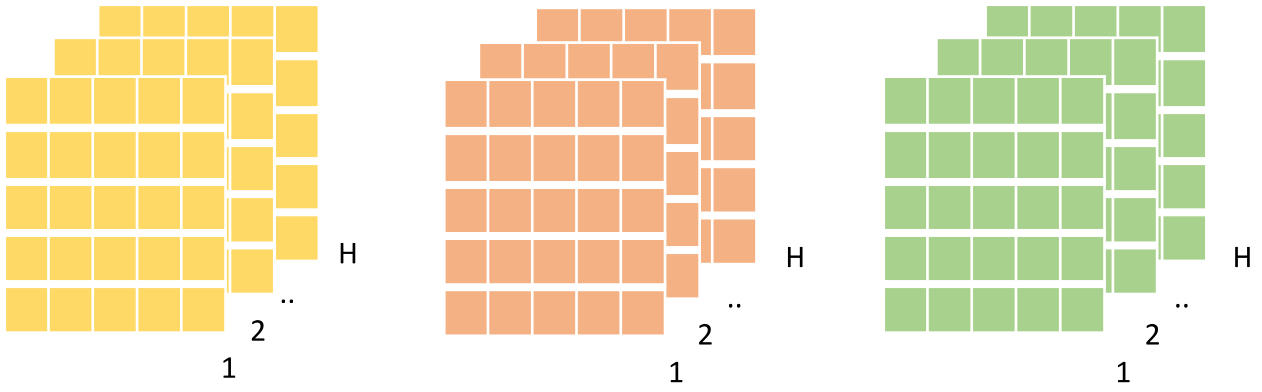


Input Embeddings

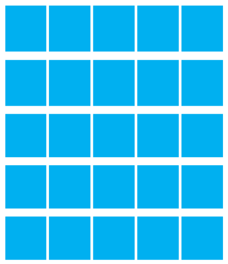


Multi-Head Attention

$R^{d_{model} \times d_h}$



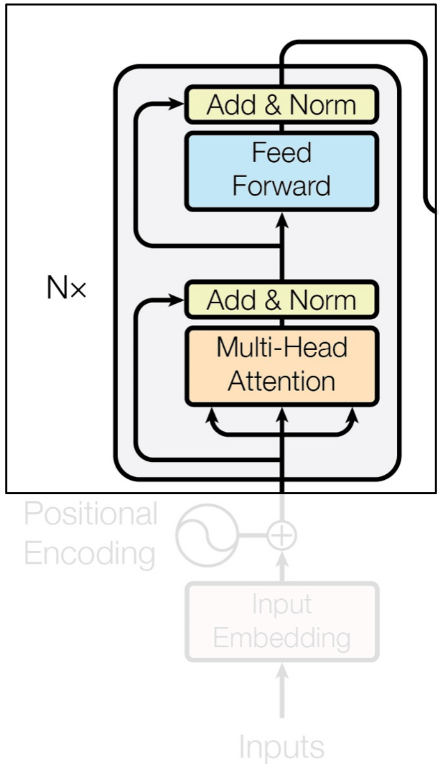
$W_{Q1}, W_{Q2}, \ldots W_{QH},$ $W_{K1}, W_{K2}, \ldots W_{KH},$ $W_{V1}, W_{V2}, \ldots W_{VH},$



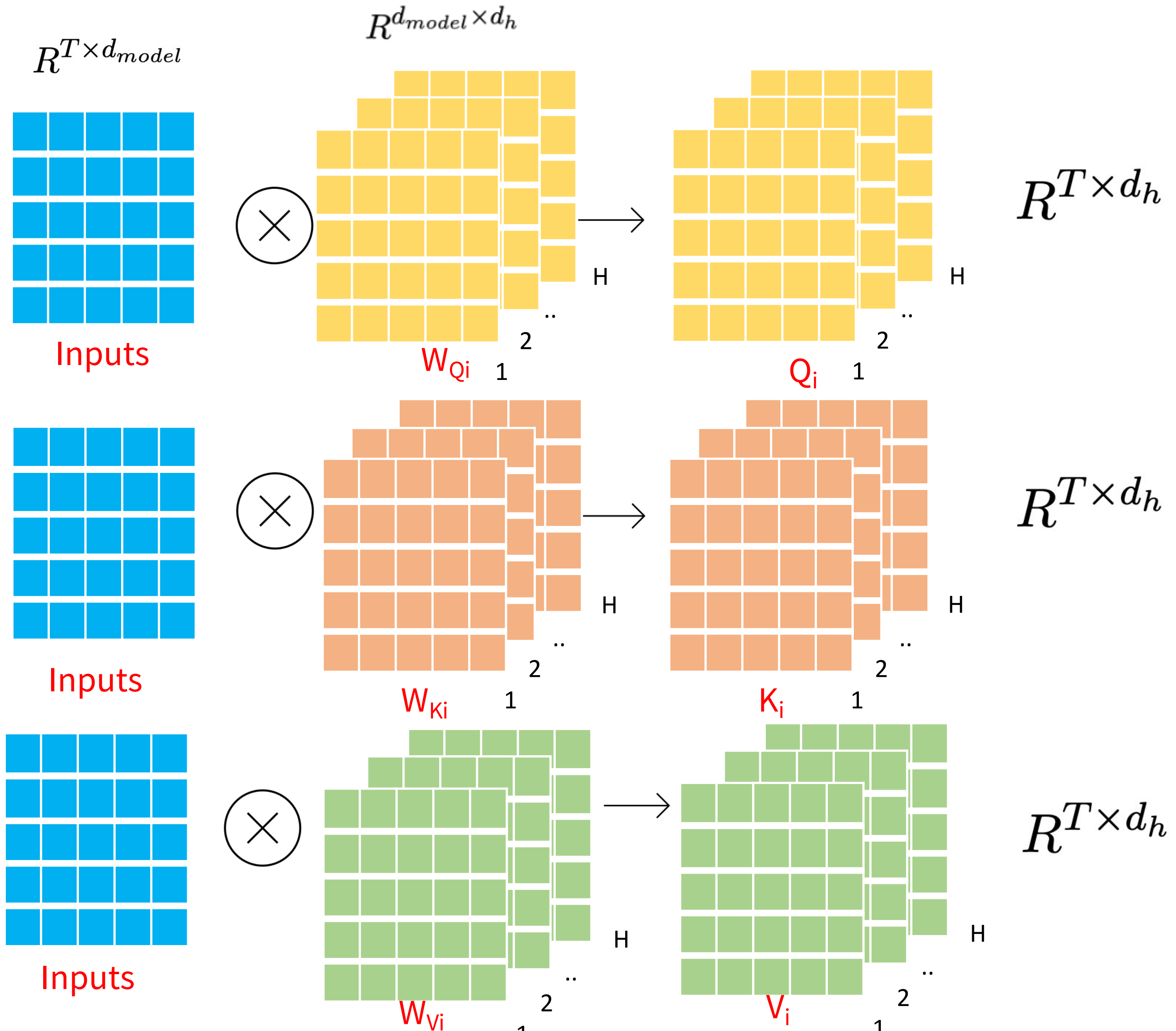
Input Embeddings

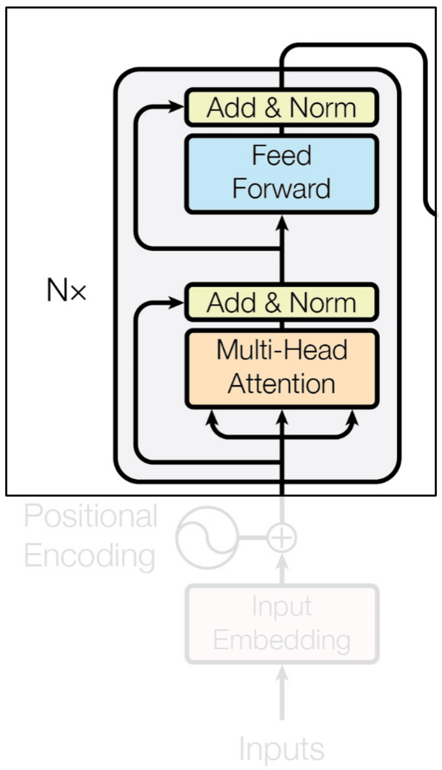
$R^{T \times d_{model}}$

$$d_h = \frac{d_{model}}{h}$$



Multi-Head Attention



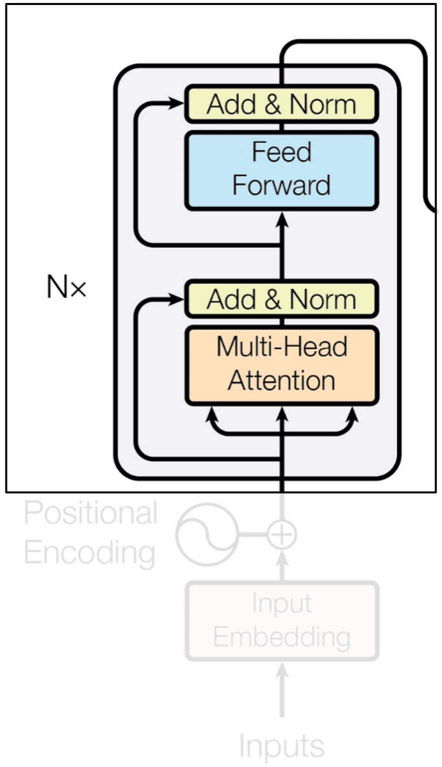


Multi-Head Attention

$R^{T \times d_h}$ $R^{d_h \times T}$ $R^{T \times d_h}$

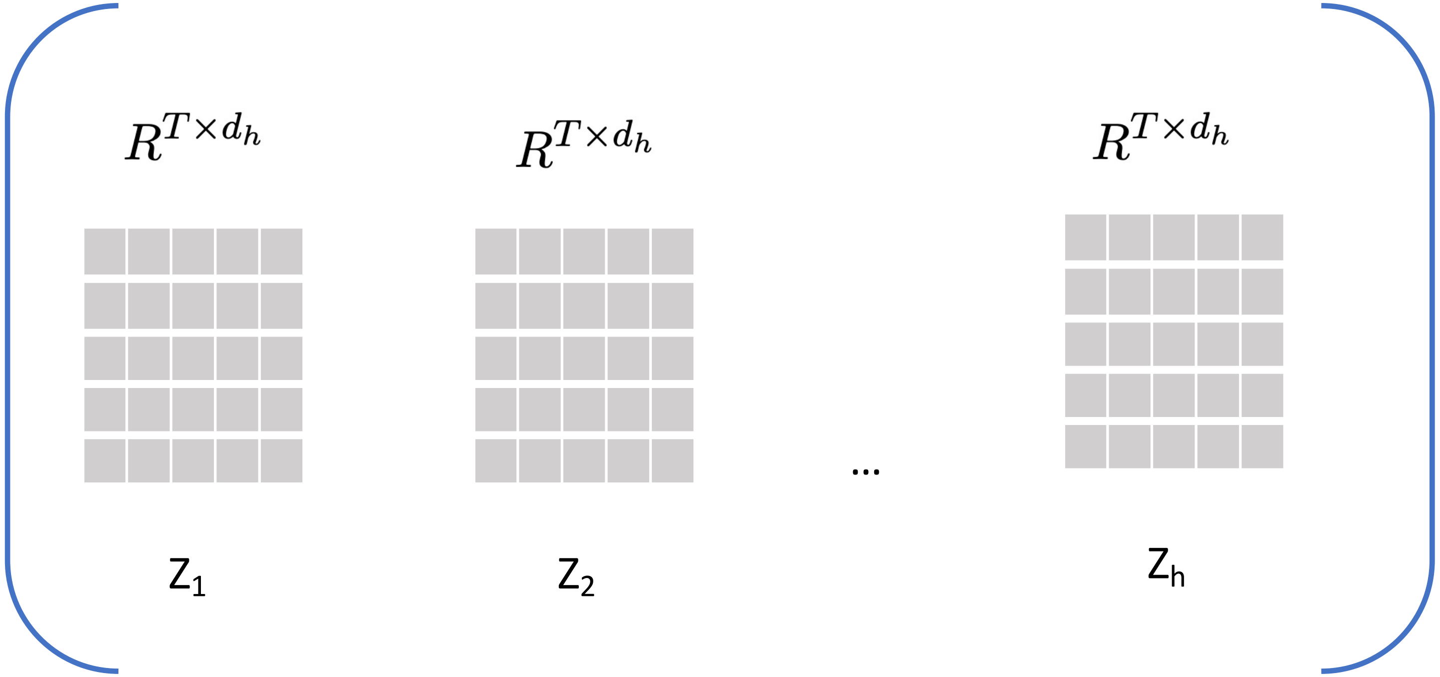
$$\text{softmax}\left(\frac{Q_i \, (K_i)^{T}}{\sqrt{d_{model}}}\right) \otimes V_i$$

$R^{T \times T}$

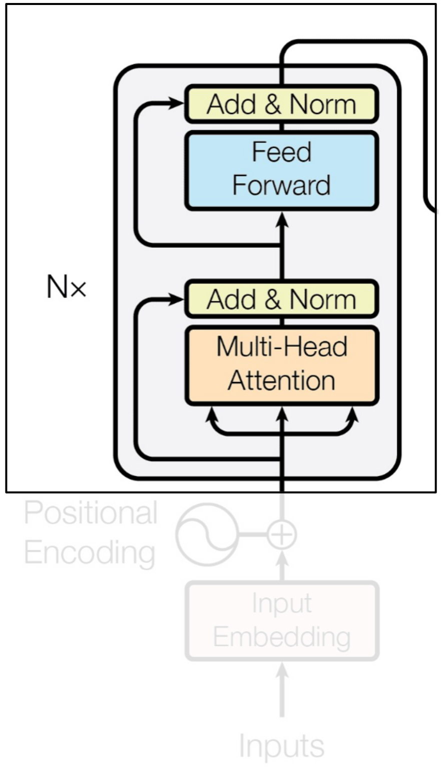


for all i ∈ [1, h]

Multi-Head Attention



CONCAT

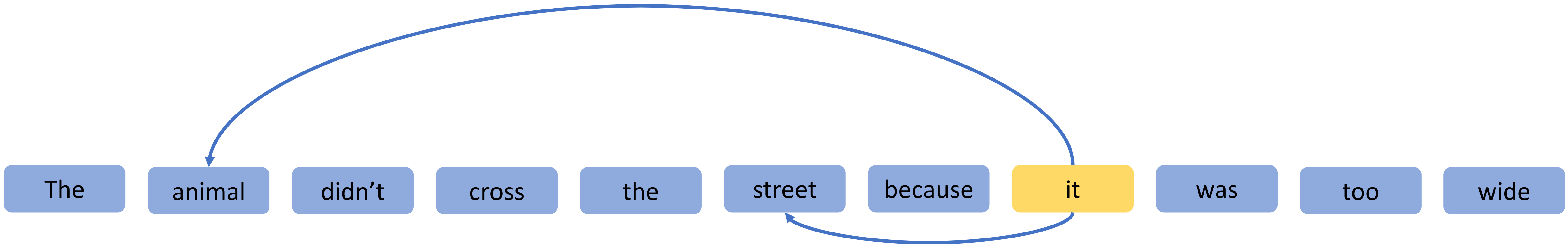


Multi Head Attention : Z

$$d_h = \frac{d_{model}}{h}$$

$R^{T\times d_{model}}$

Multi-Head Attention



Sentence boundaries ?

coreference resolution

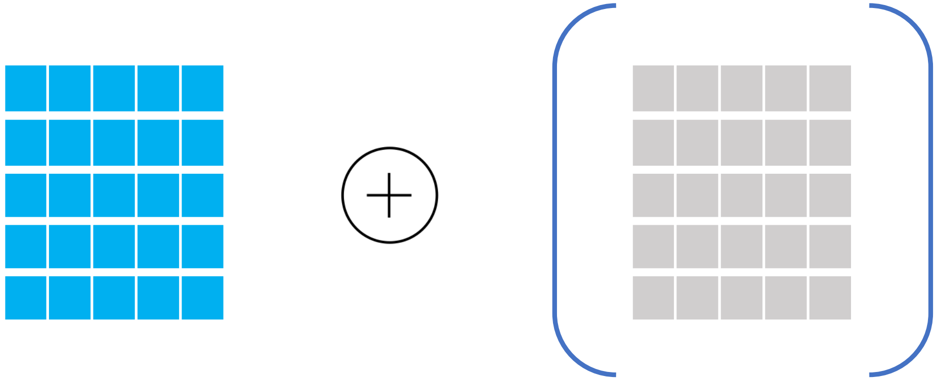
Context ?

Semantic relationships ?

Part of Speech ?

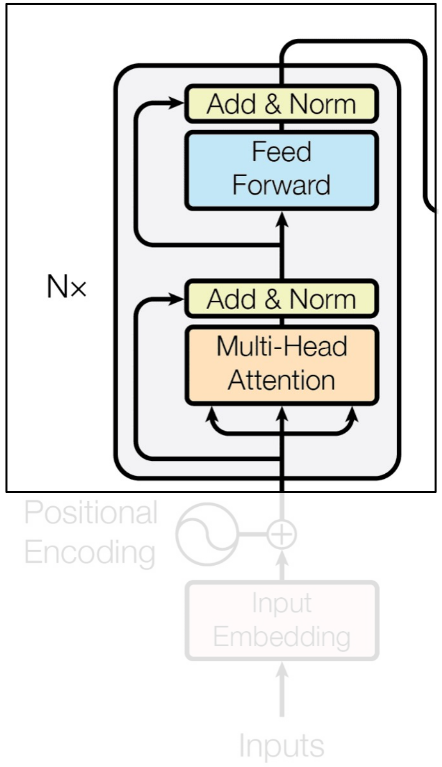
Comparisons ?

Add & Norm



Input

Norm(Z)



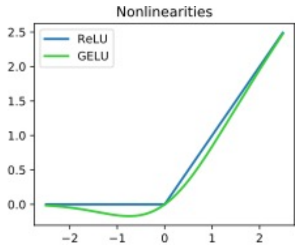
Normalization(Z)

- Mean 0, Std dev 1
- Stabilizes training
- Regularization effect

Add -> Residuals

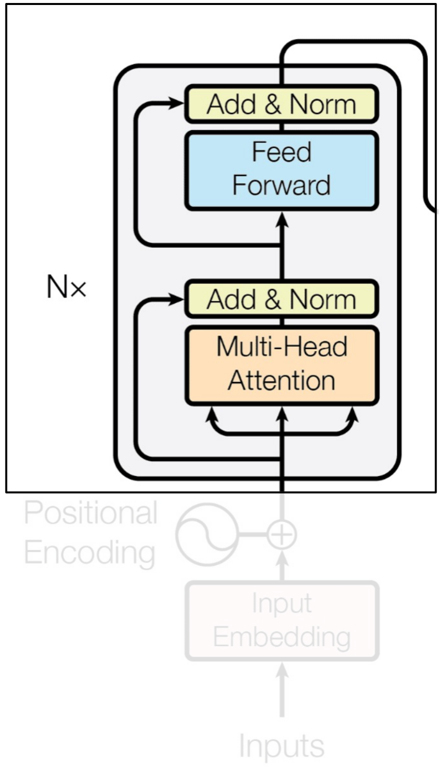
- Avoid vanishing gradients
- Train deeper networks

Feed Forward

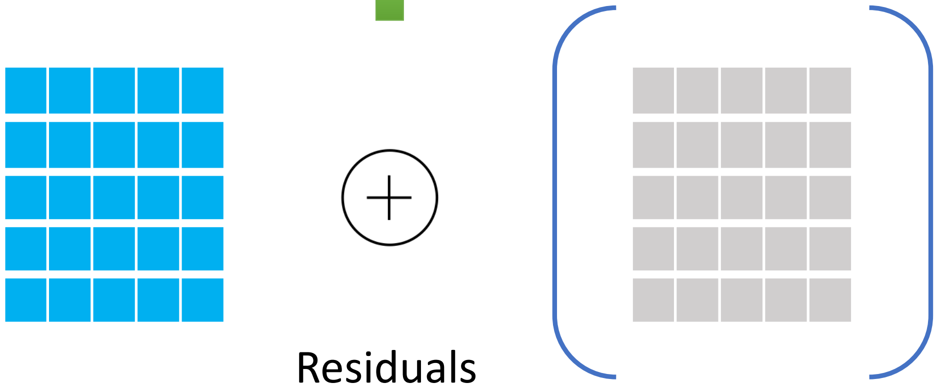


Feed Forward

- Non Linearity
- Complex Relationships
- Learn from each other



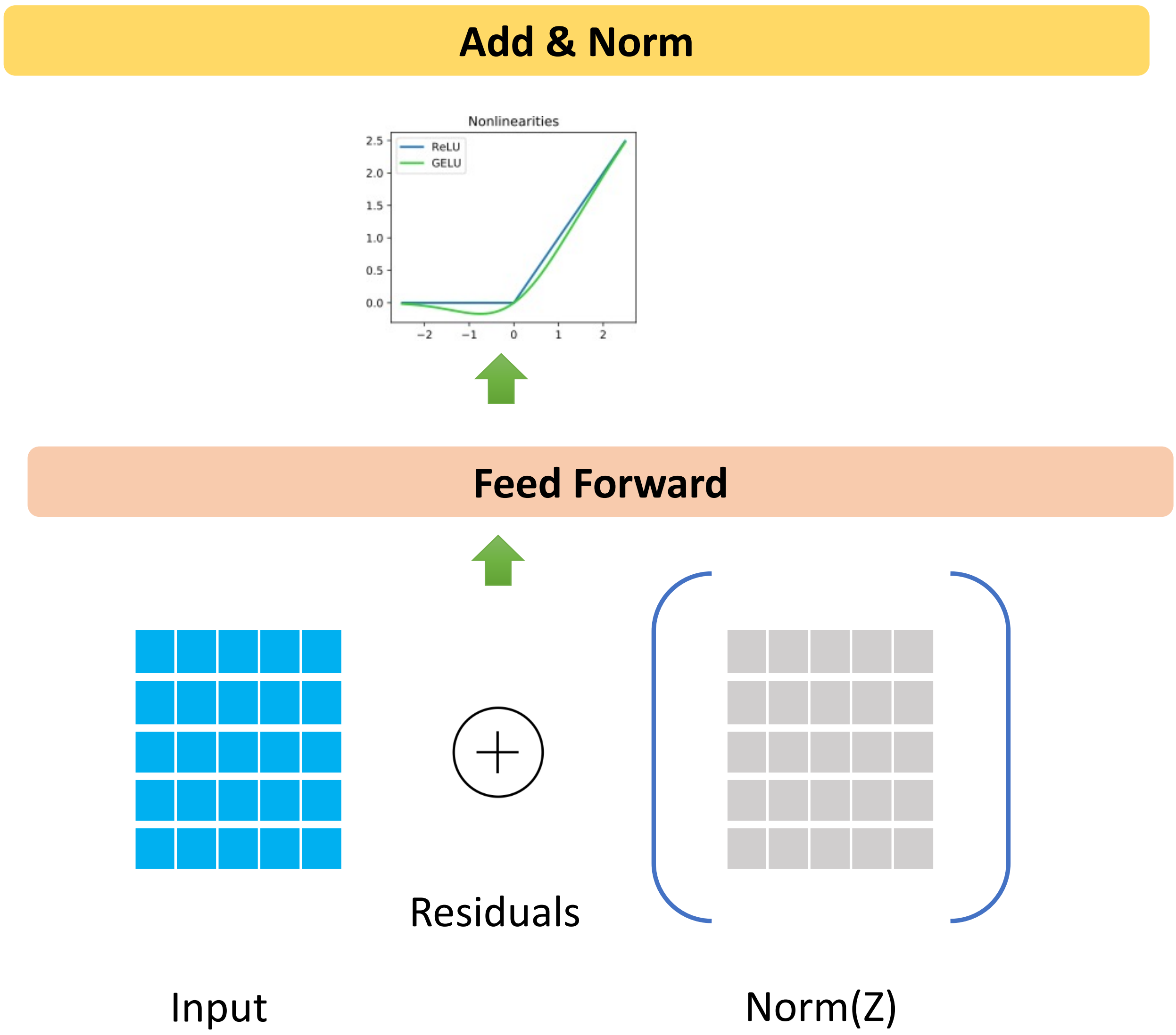
Feed Forward

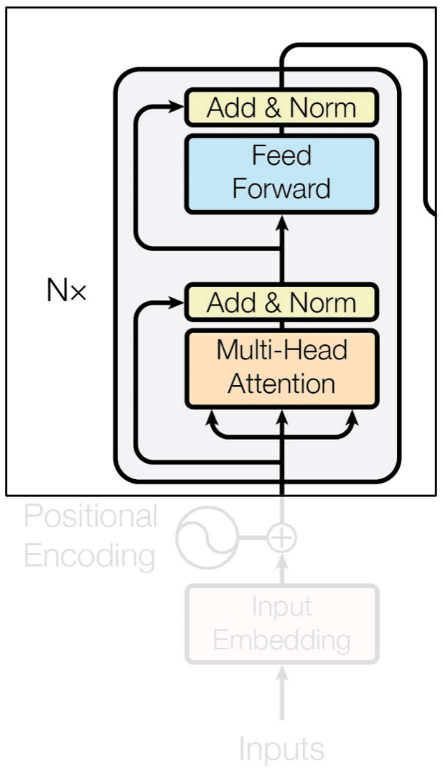


Input

Norm(Z)

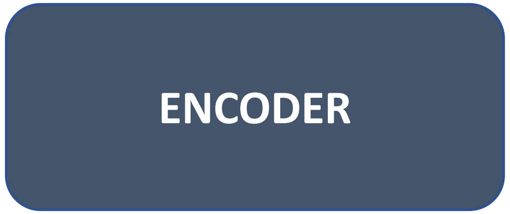
Add & Norm

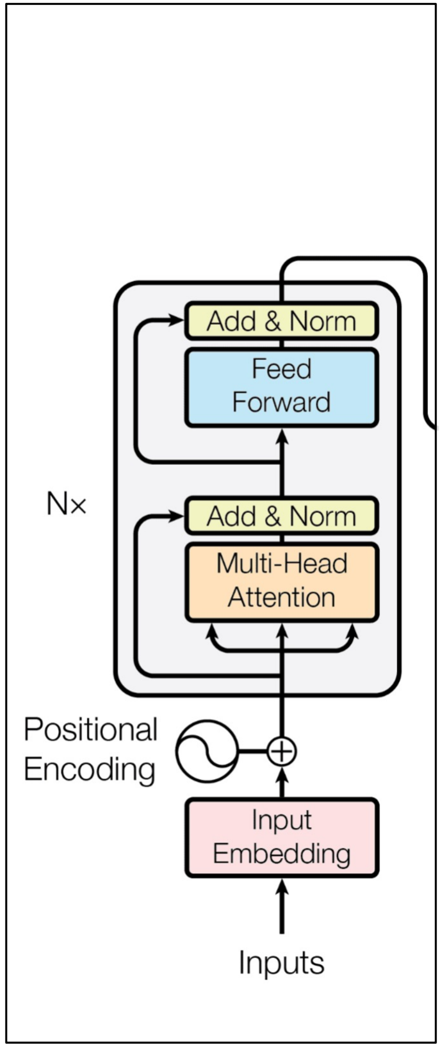

Add & Norm
Nonlinearities
ReLU
GELU
2.5
2.0
1.5
1.0
0.5
0.0
−2
−1
0
1
2
Feed Forward
Residuals
Input
Norm(Z)


Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head Attention
Positional Encoding
Input Embedding
Inputs

Encoders

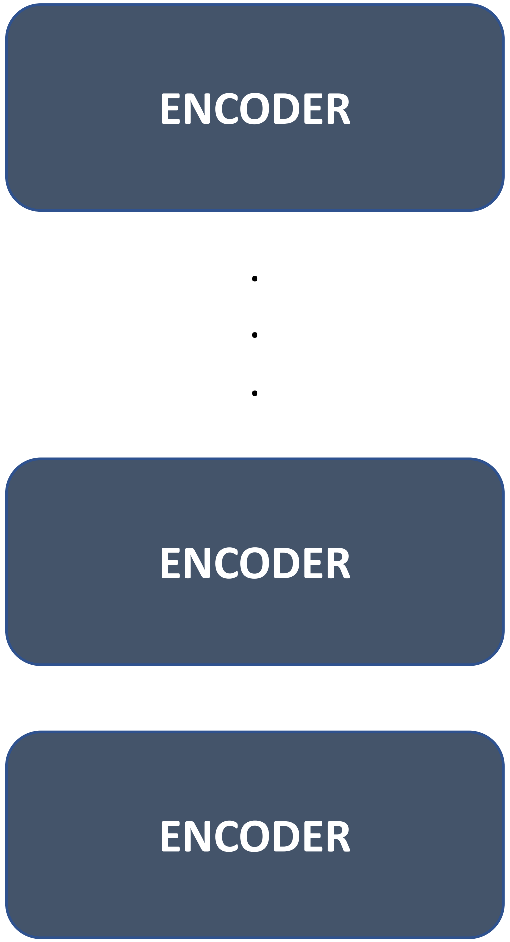
Encoder





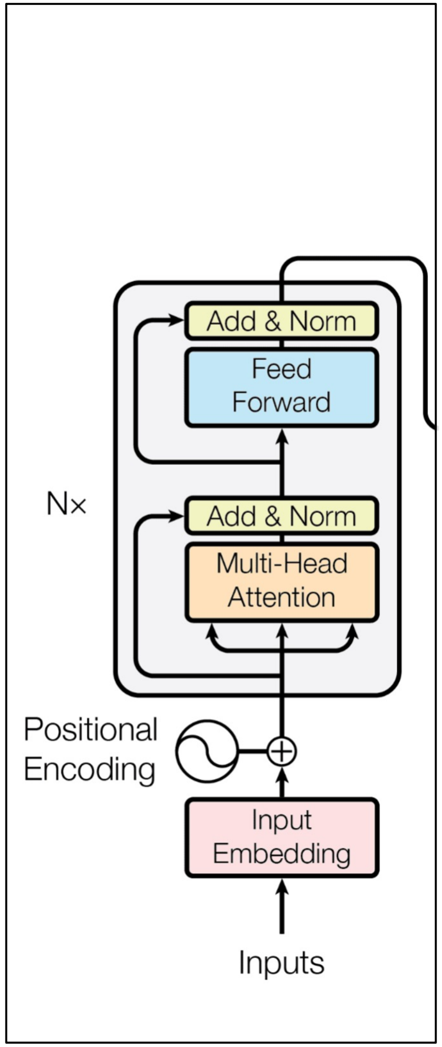
Encoders

Encoder



Input to Encoder_{i+1}

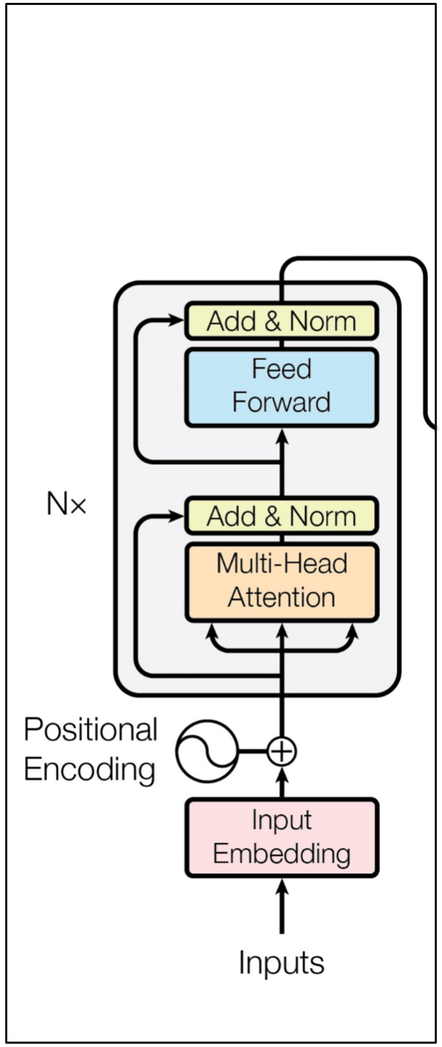
Output from Encoder_i



Transformers

- ✓ Tokenizaton
- ✓ Input Embeddings
- ✓ Position Encodings
- ✓ Residuals
- ✓ Query
- ✓ Key
- ✓ Value
- ✓ Add & Norm
- ✓ Encoder
- Decoder

- ✓ Attention
- ✓ Self Attention
- ✓ Multi Head Attention
- Masked Attention
- Encoder Decoder Attention
- Output Probabilities / Logits
- Softmax
- Encoder-Decoder models
- Decoder only models



Machine Translation

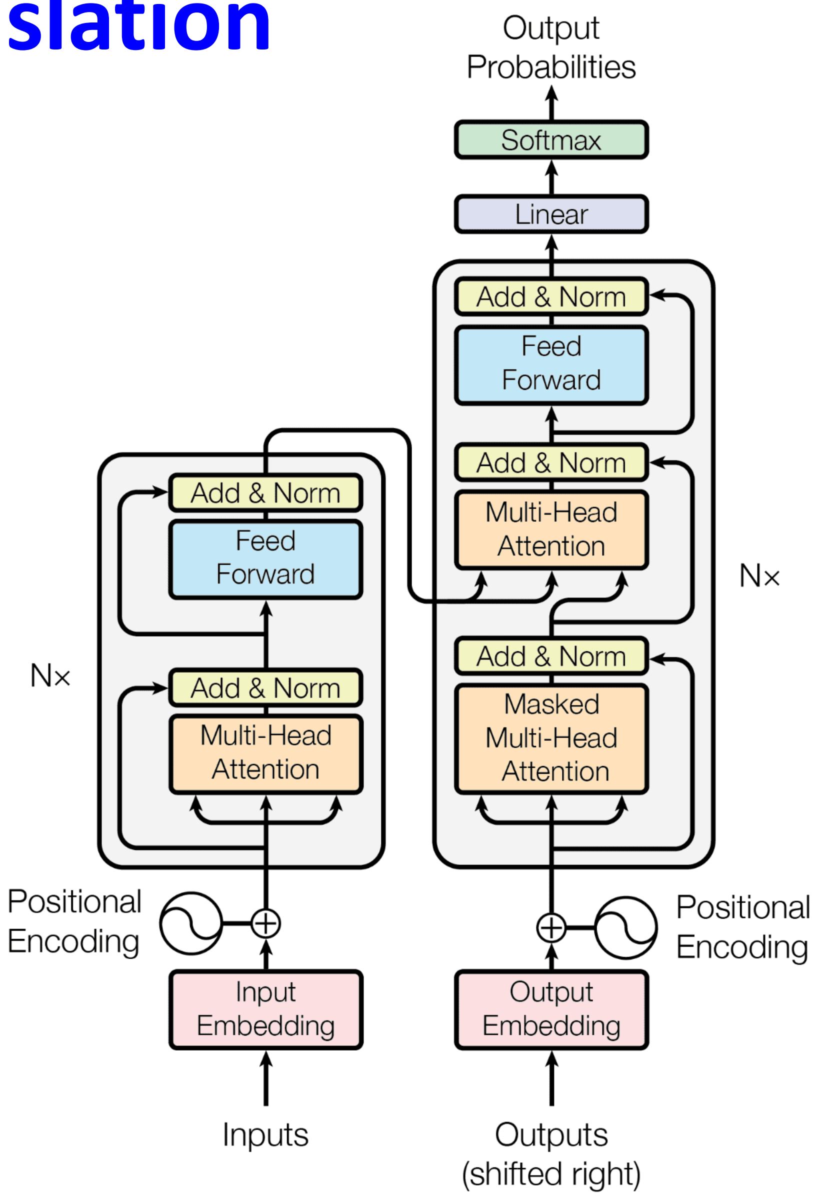
Targets

Ich have einen apfel gegessen

⬆

Inputs

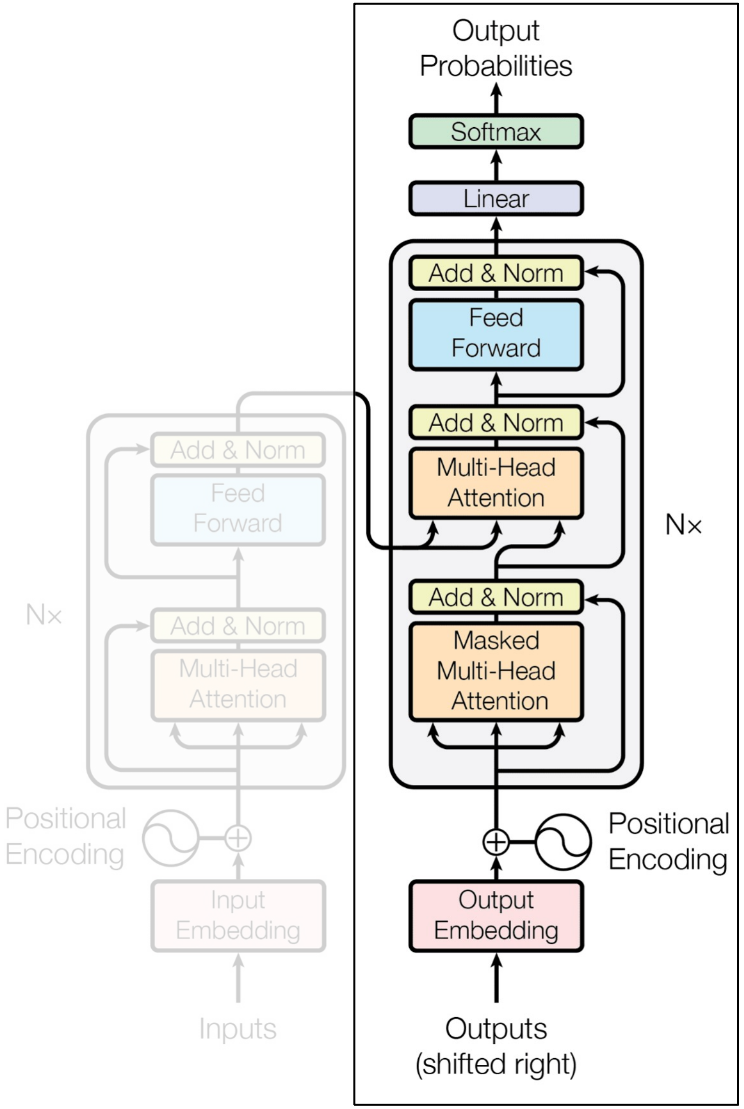
I ate an apple



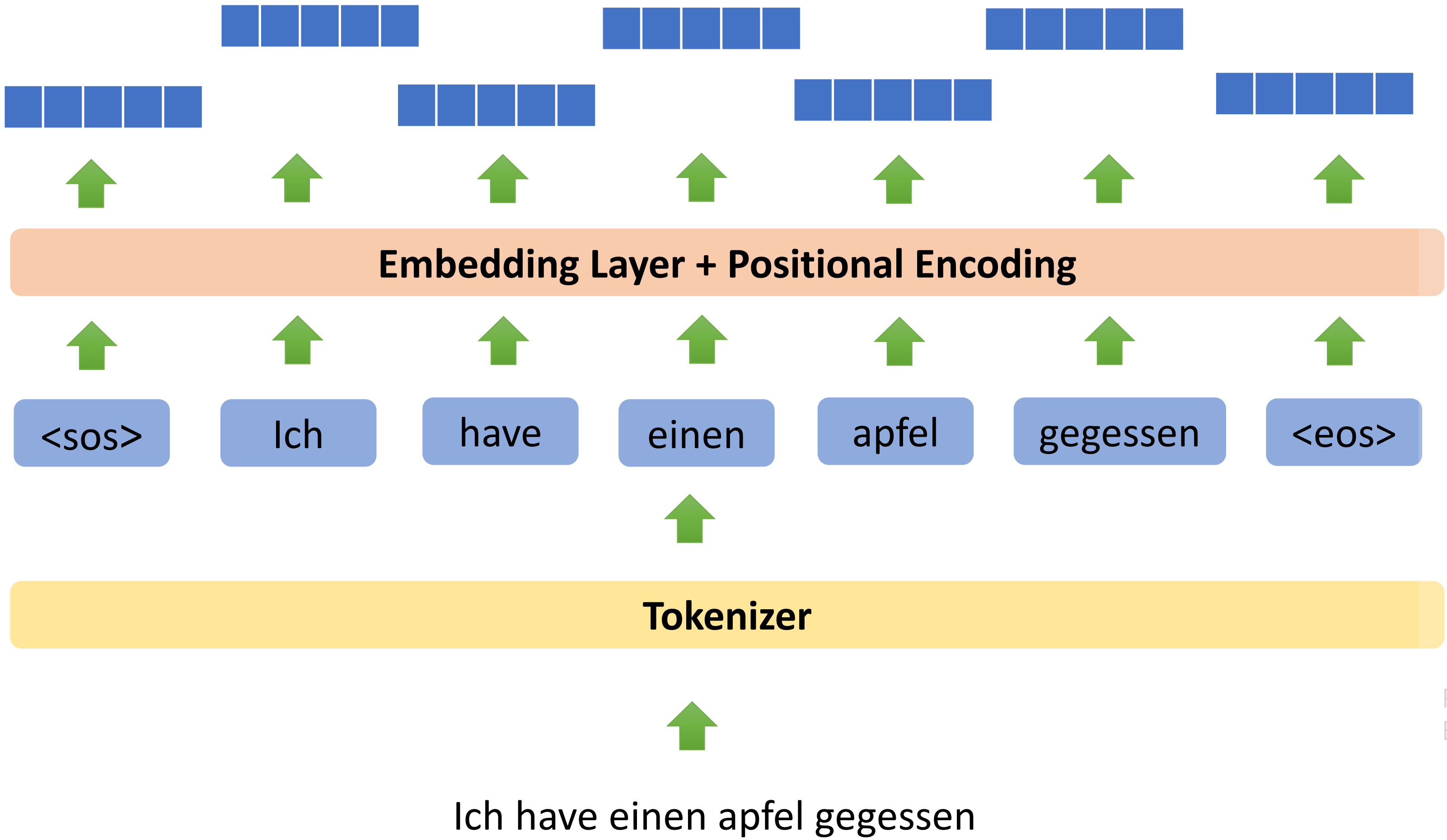
Targets

Targets

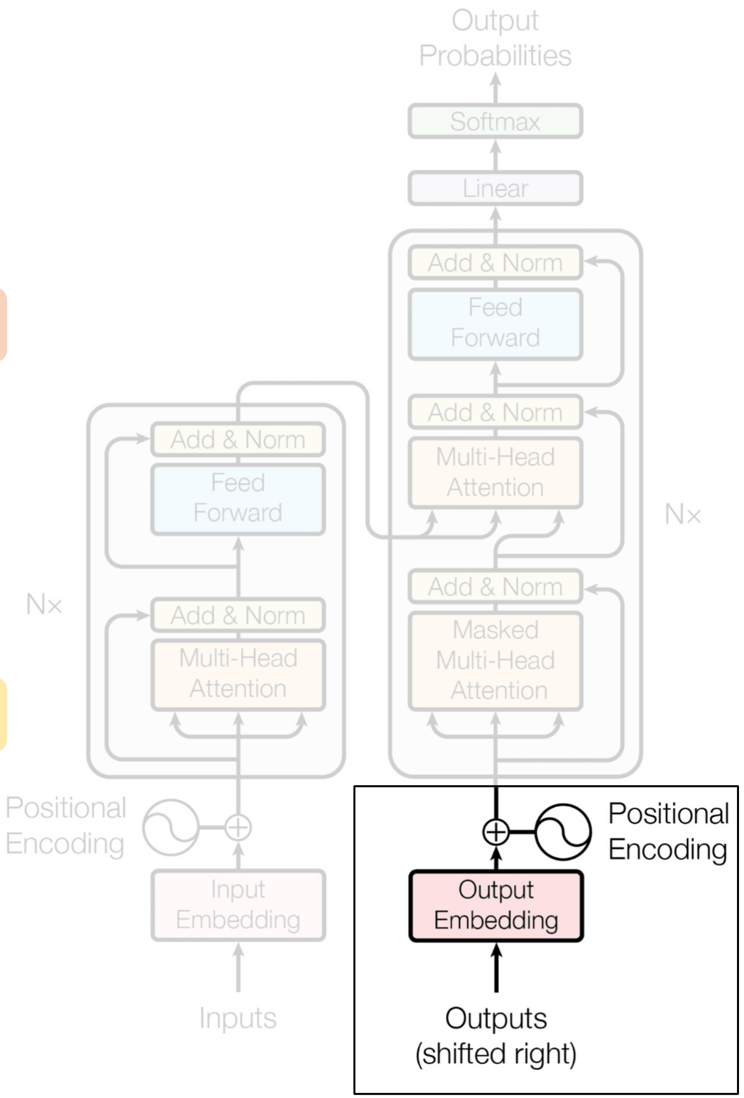
Ich have einen apfel gegessen



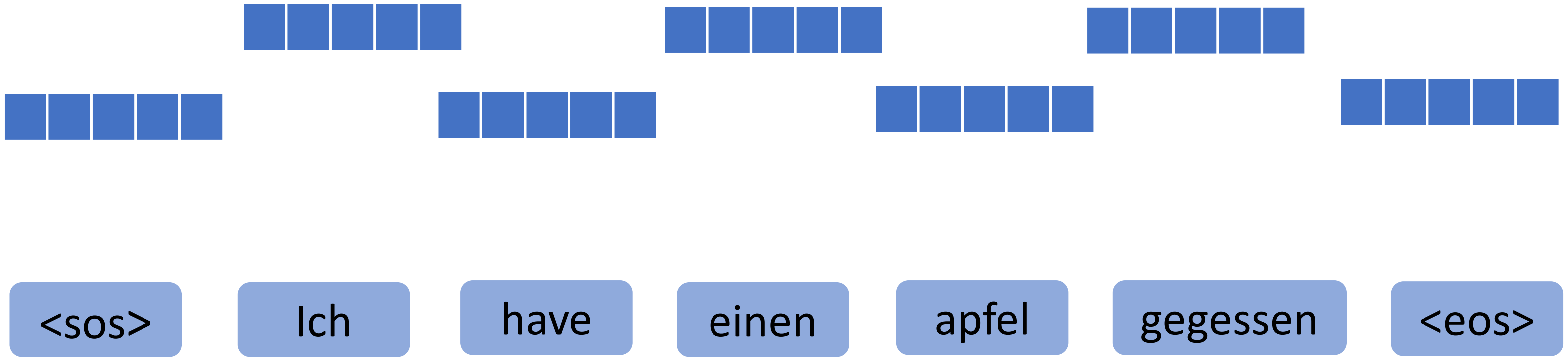
Targets

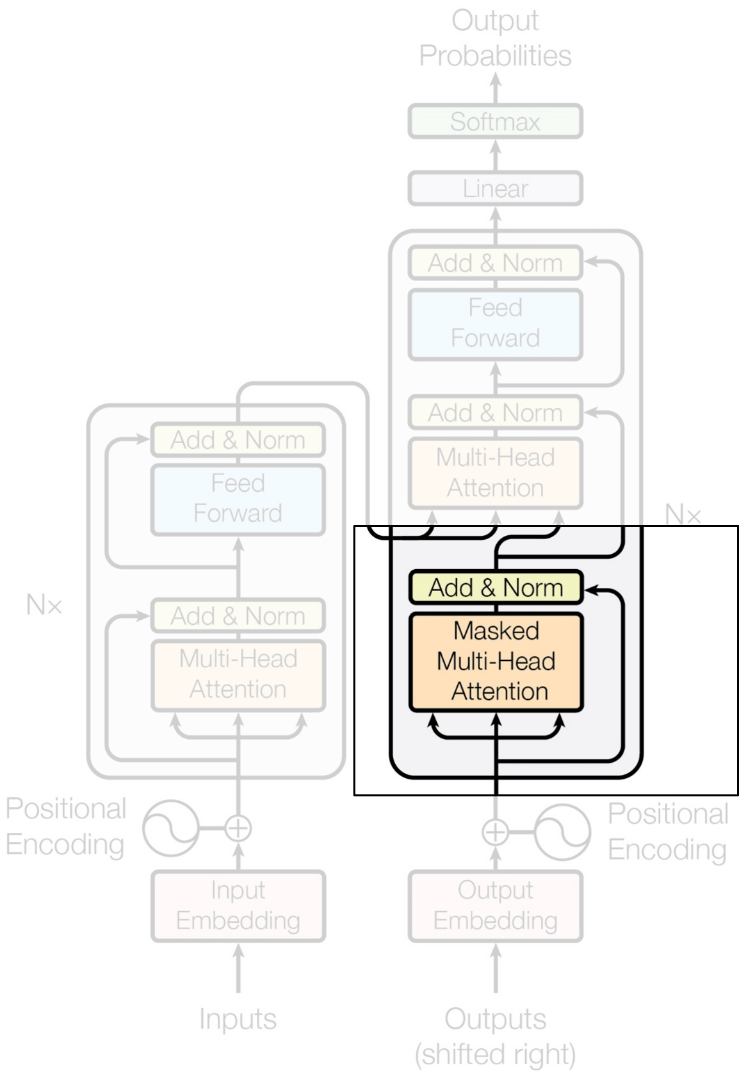


Generate Target Emebeddings



Masked Multi Head Attention

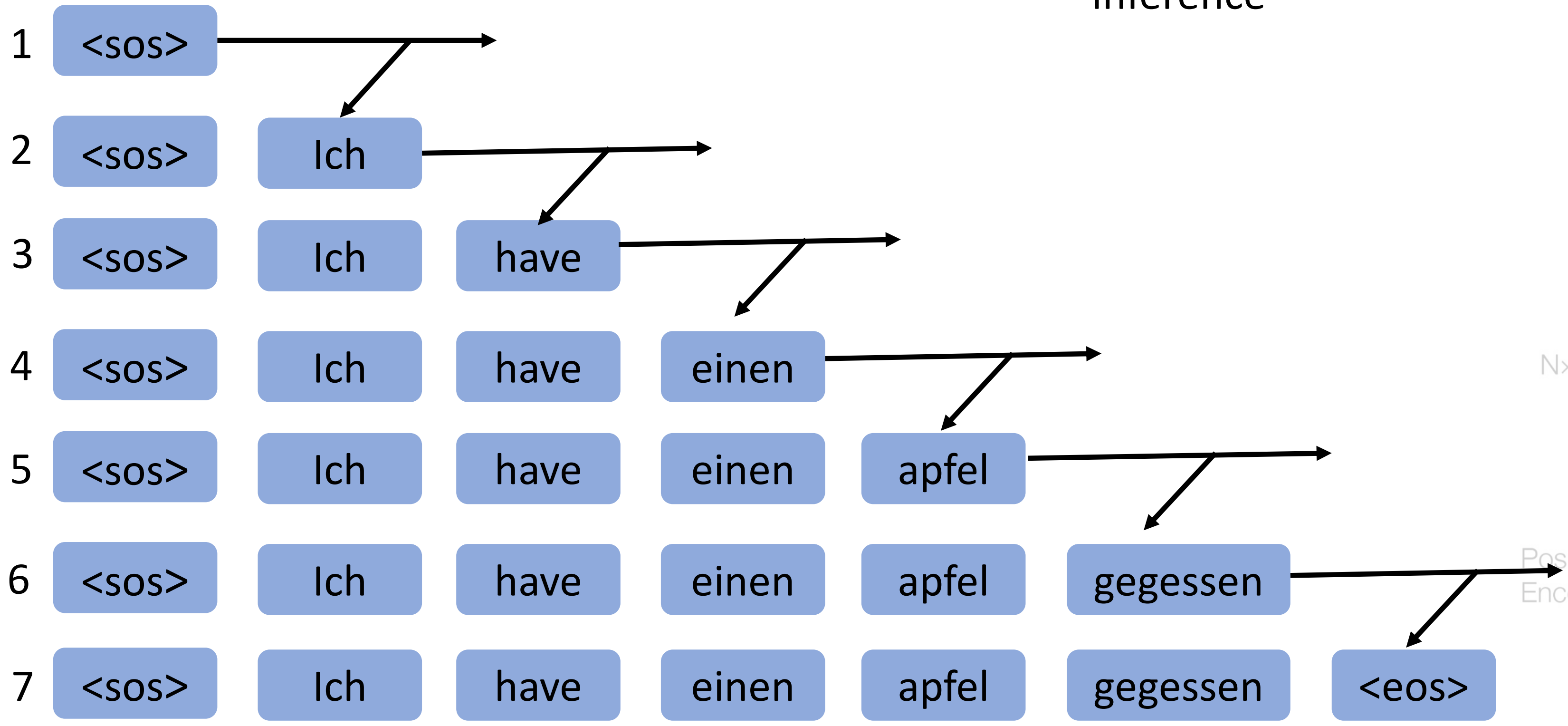


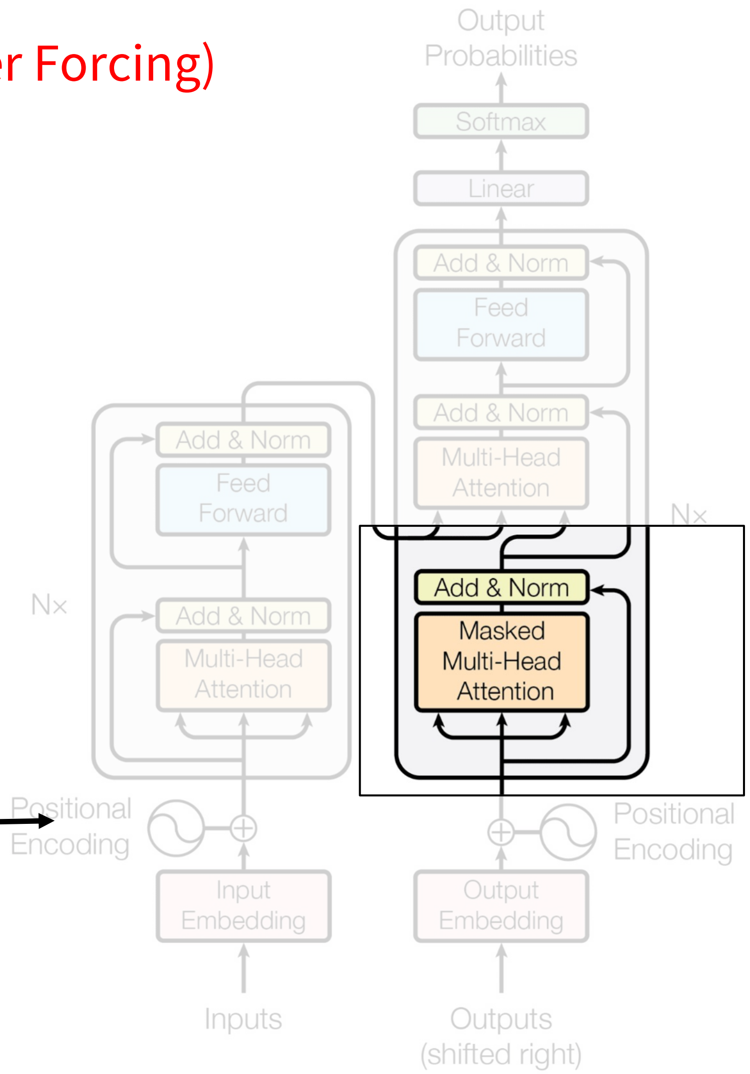


Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

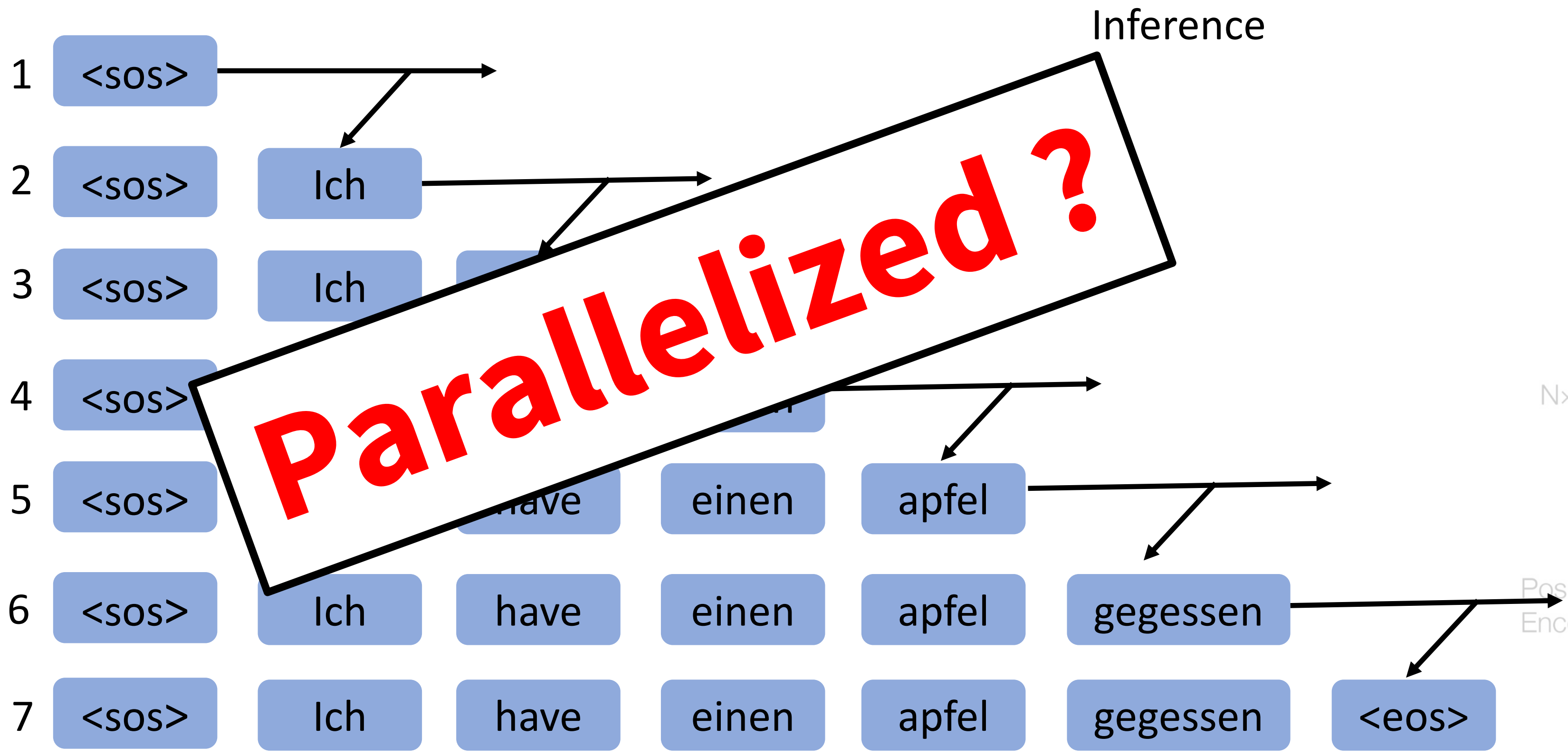
Inference

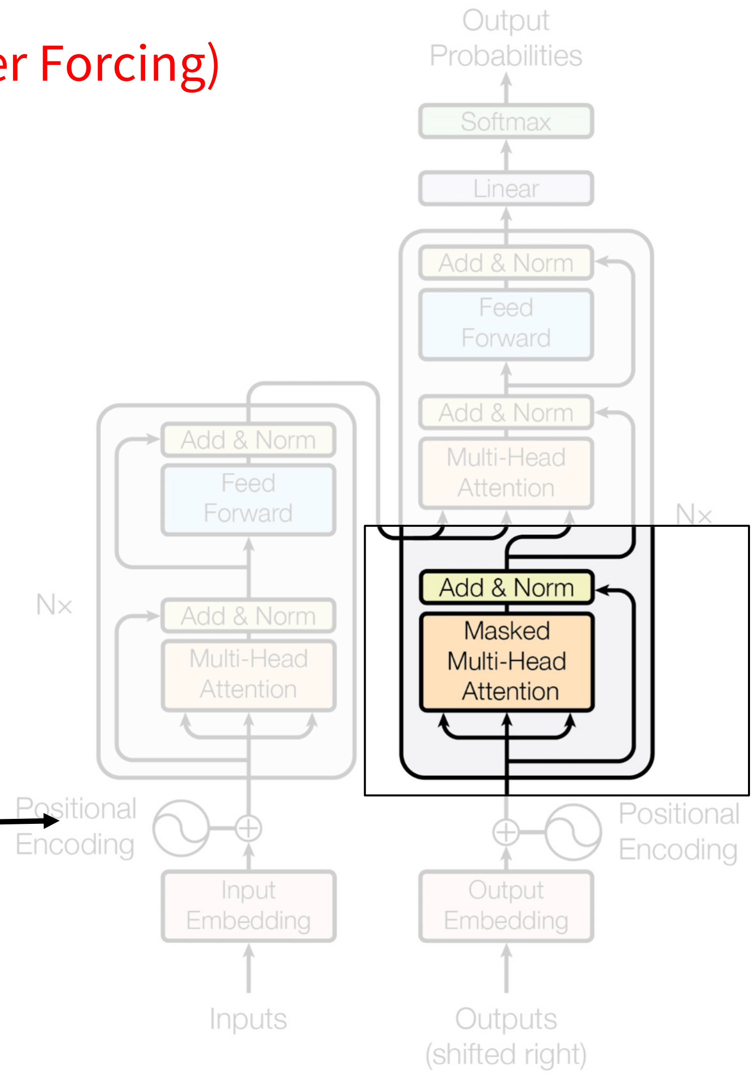




Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)



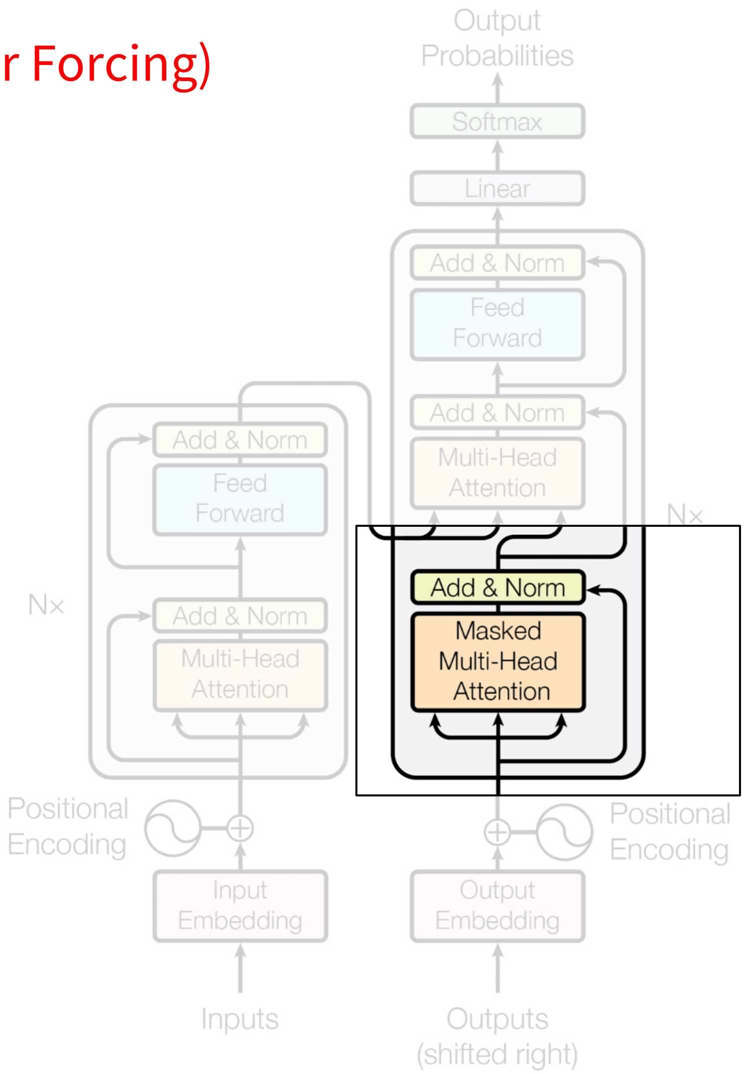


Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

Training

<sos> Ich have einen apfel gegessen <eos>



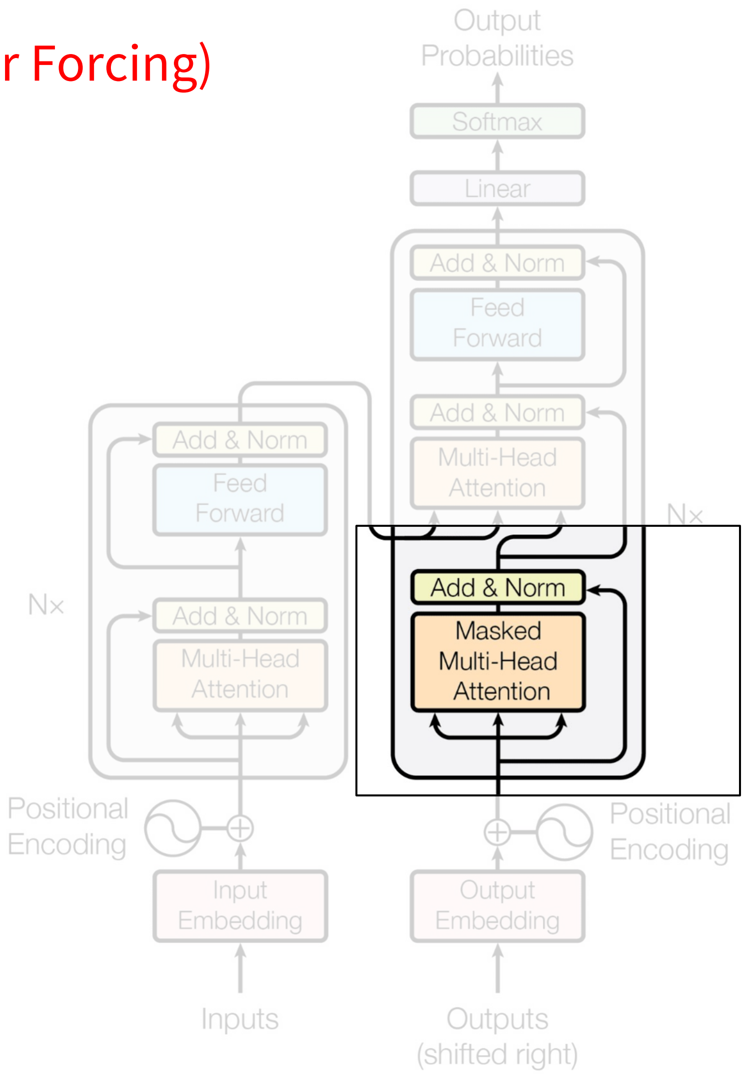
Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

Training

<sos> Ich have einen apfel gegessen <eos>

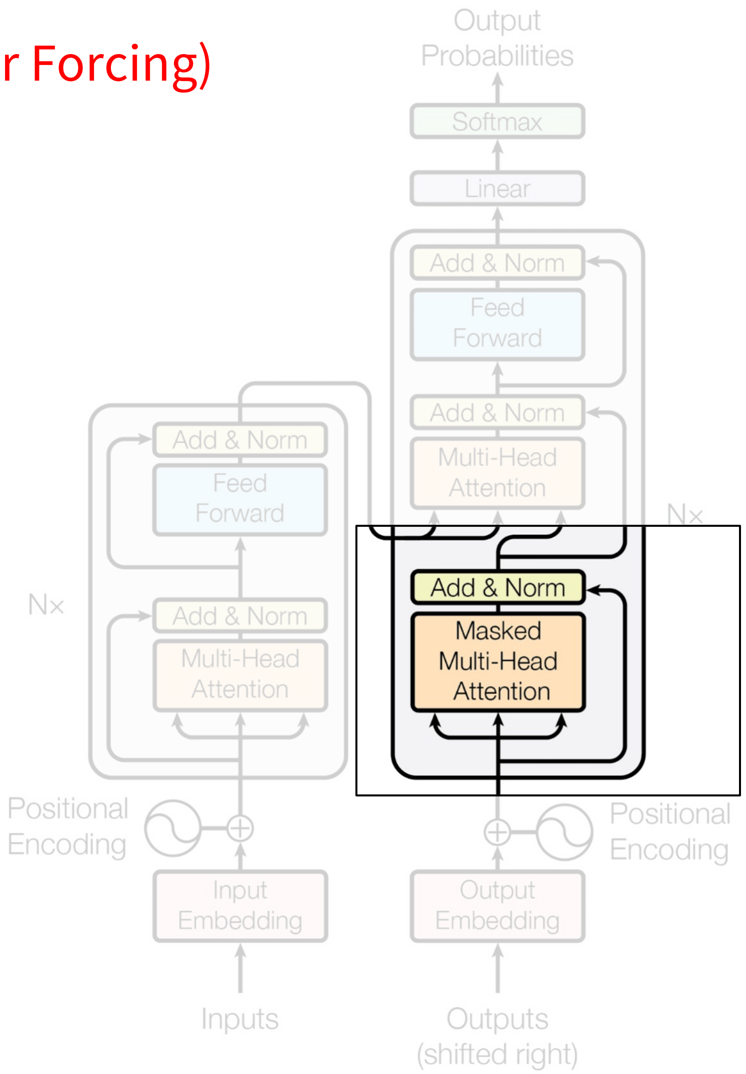
Outputs at time T should only pay attention to outputs until time T-1



Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

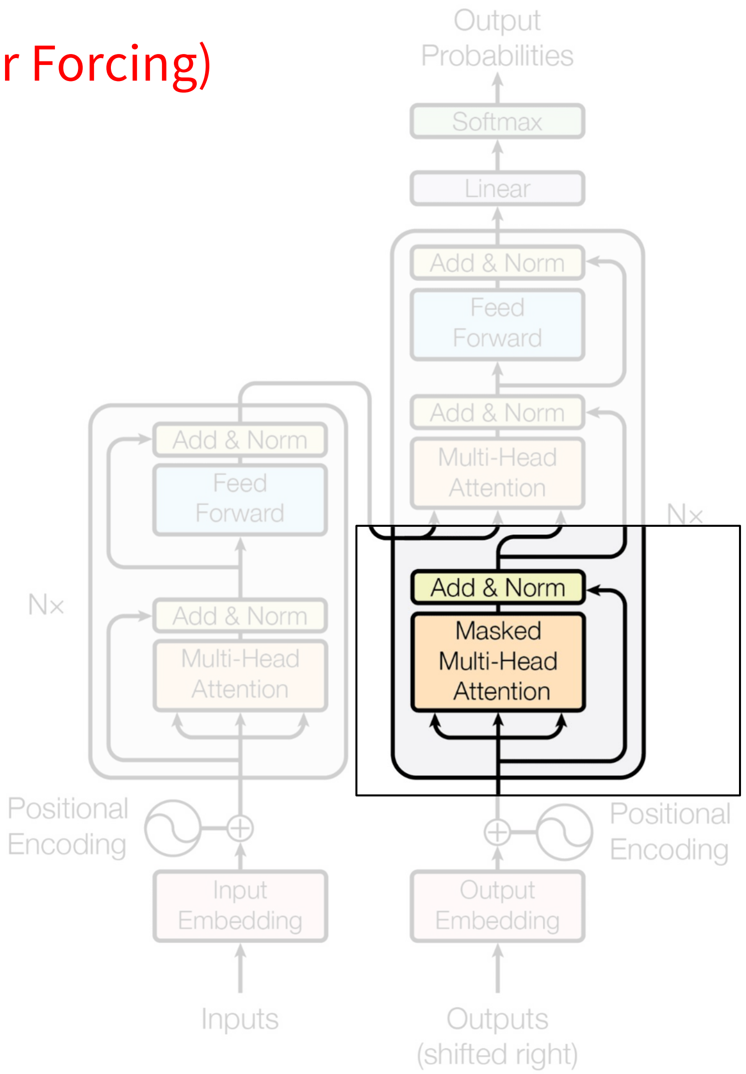
1	<sos>	Ich	have	einen	apfel	gegessen	<eos>
2	<sos>	Ich	have	einen	apfel	gegessen	<eos>
3	<sos>	Ich	have	einen	apfel	gegessen	<eos>
4	<sos>	Ich	have	einen	apfel	gegessen	<eos>
5	<sos>	Ich	have	einen	apfel	gegessen	<eos>
6	<sos>	Ich	have	einen	apfel	gegessen	<eos>
7	<sos>	Ich	have	einen	apfel	gegessen	<eos>



Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

1	<sos>	Ich	have	einen	apfel	gegessen	<eos>
2	<sos>	Ich	have	einen	apfel	gegessen	<eos>
3	<sos>	Ich	have	einen	apfel	gegessen	<eos>
4	<sos>	Ich	have	einen	apfel	gegessen	<eos>
5	<sos>	Ich	have	einen	apfel	gegessen	<eos>
6	<sos>	Ich	have	einen	apfel	gegessen	<eos>
7	<sos>	Ich	have	einen	apfel	gegessen	<eos>

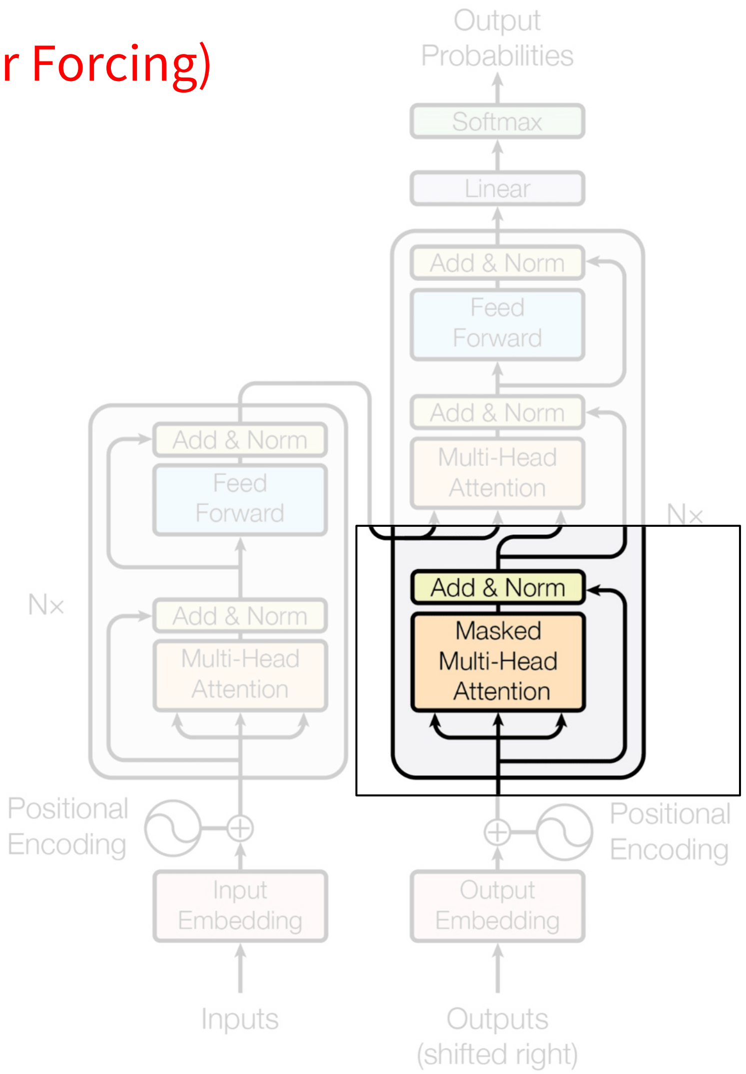


Mask the available attention values ?

Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

1	<sos>	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
2	<sos>	Ich	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
3	<sos>	Ich	have	$-\infty$	$-\infty$	$-\infty$	$-\infty$
4	<sos>	Ich	have	einen	$-\infty$	$-\infty$	$-\infty$
5	<sos>	Ich	have	einen	apfel	$-\infty$	$-\infty$
6	<sos>	Ich	have	einen	apfel	gegessen	$-\infty$
7	<sos>	Ich	have	einen	apfel	gegessen	<eos>

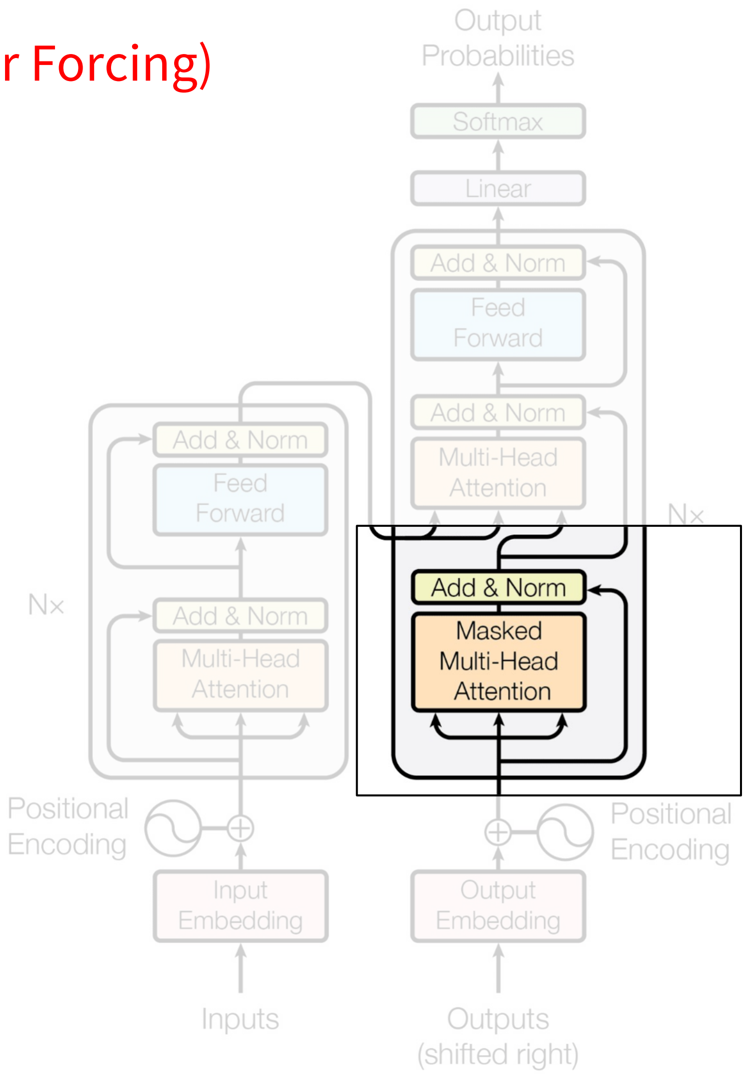


Masked Multi Head Attention

Decoding step by step (using Teacher Forcing)

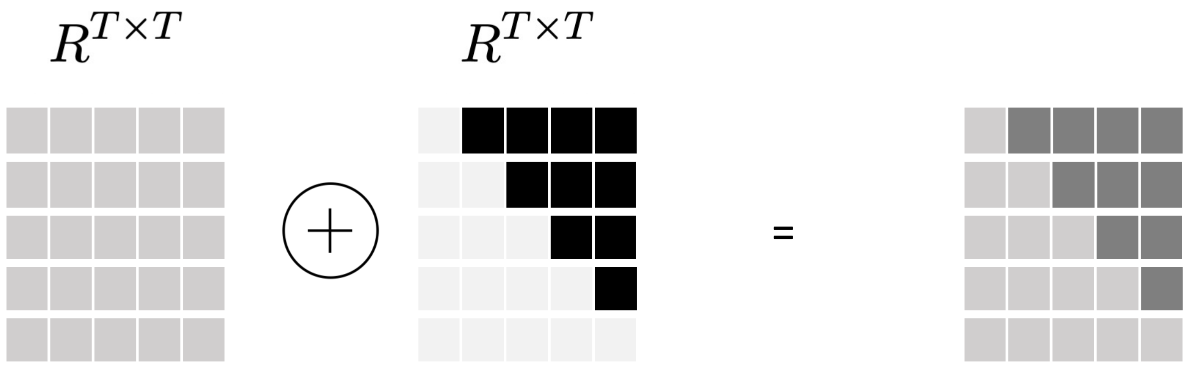
1	<sos>	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
2	<sos>	Ich	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
3	<sos>	Ich	have	$-\infty$	$-\infty$	$-\infty$	$-\infty$
4	<sos>	Ich	have	einen	$-\infty$	$-\infty$	$-\infty$
5	<sos>	Ich	have	einen	apfel	$-\infty$	$-\infty$
6	<sos>	Ich	have	einen	apfel	gegessen	$-\infty$
7	<sos>	Ich	have	einen	apfel	gegessen	<eos>

Softmax -> $-\infty$ -> 0

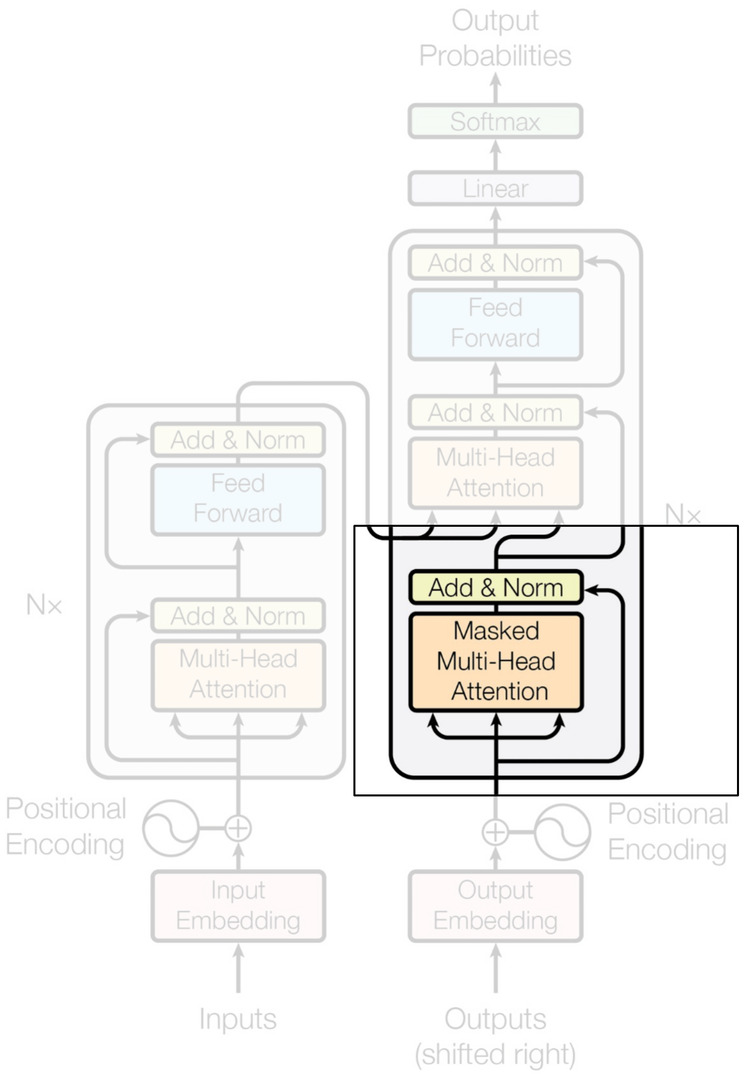


Masked Multi Head Attention

Masked Multi Head Attention



QK^T Attention Mask: M Masked Attention

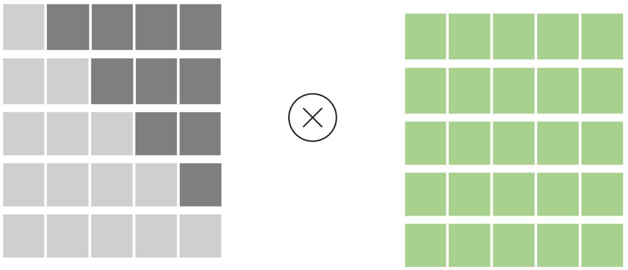


Masked Multi Head Attention : Z’

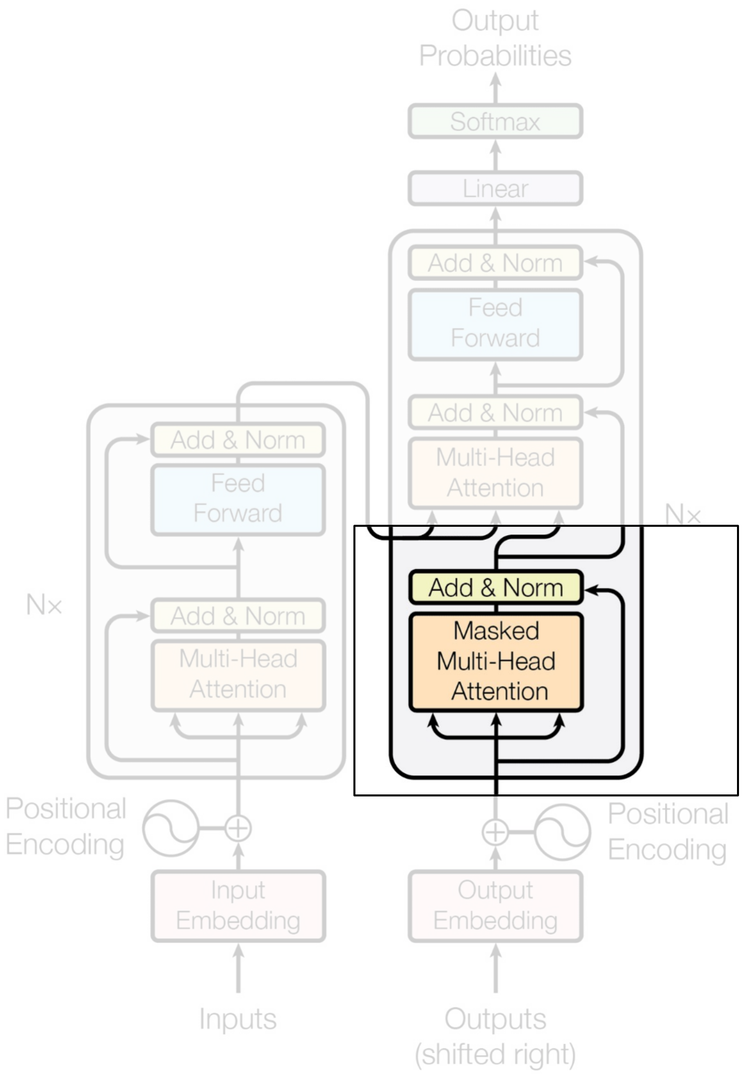
Masked Multi Head Attention

Masked Multi Head Attention

$R^{T \times T}$ $R^{T \times d_h}$



Masked Attention Values

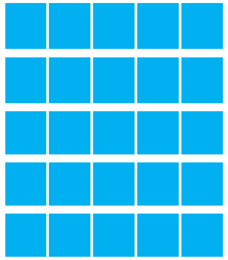


Masked Multi Head Attention : Z’

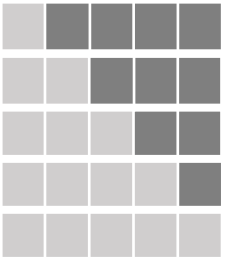
Encoder Decoder Attention

Encoder Decoder Attention ?

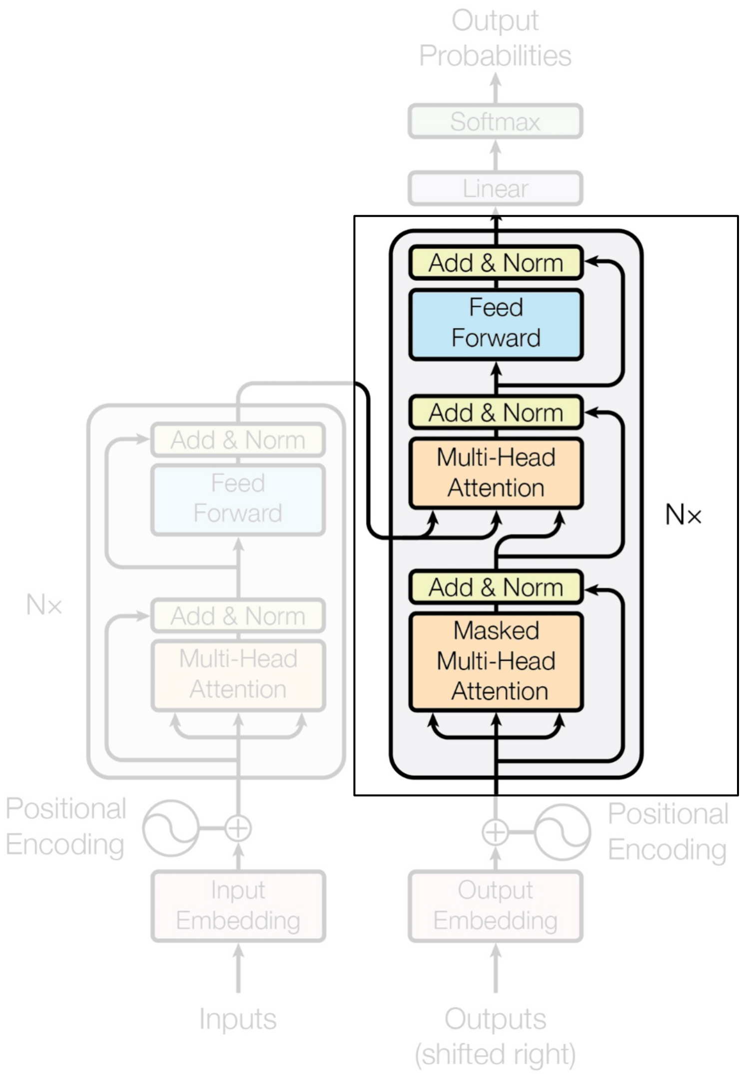
Add & Norm



Input

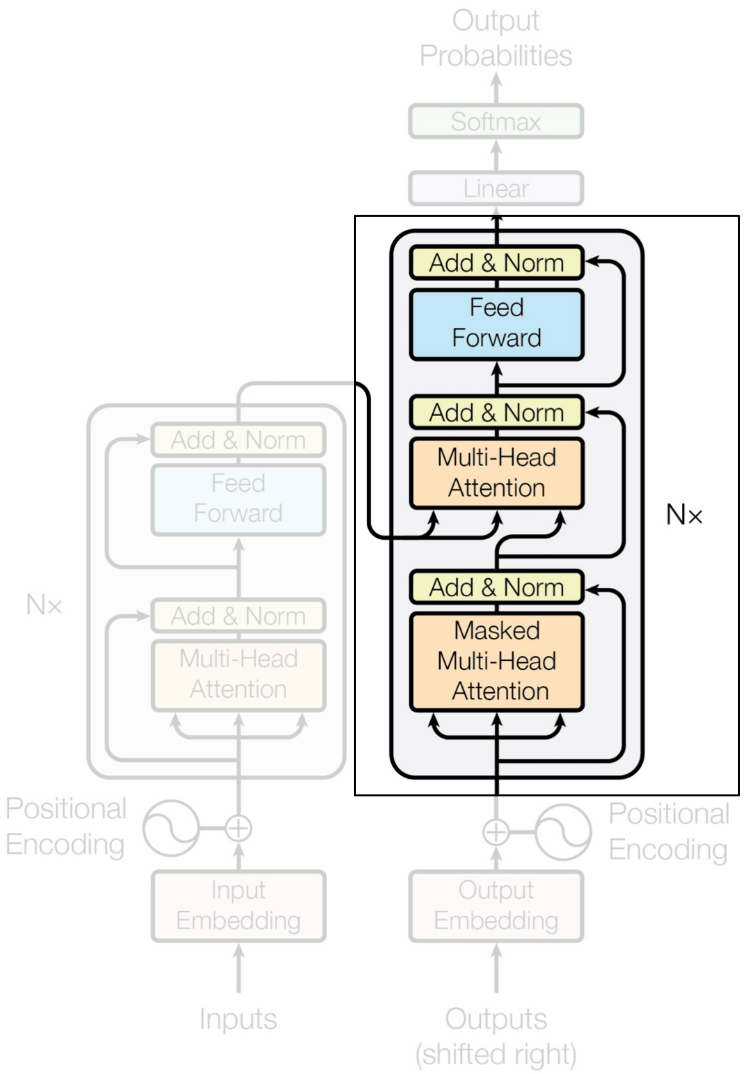


Norm(Z')



Encoder Decoder Attention

Encoder Decoder Attention ?



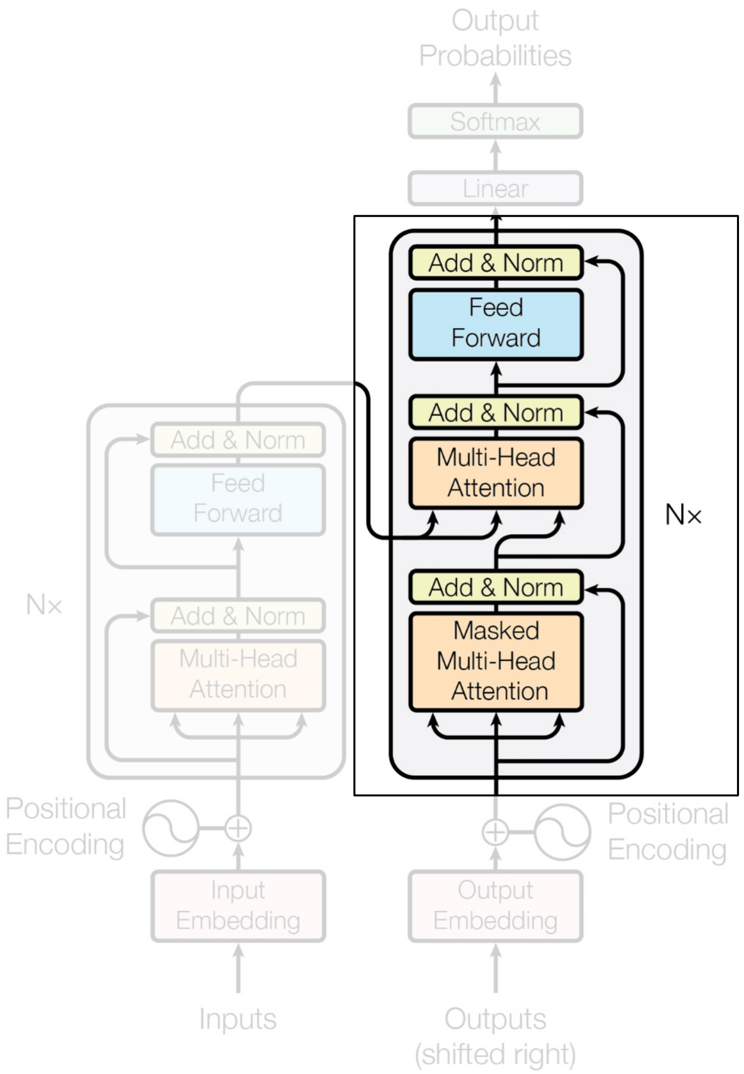
Encoder Decoder Attention

Encoder **Self** Attention

1. Queries from Encoder Inputs
2. Keys from Encoder Inputs
3. Values from Encoder Inputs

Decoder **Masked Self** Attention

1. Queries from Decoder Inputs
2. Keys from Decoder Inputs
3. Values from Decoder Inputs



Attention

{Key, Value store}

{Query: “Order details of order_104”}

{Query: “Order details of order_106”}

```
{"order_100": {"items":"a1", "delivery_date":"a2", ...}},
{"order_101": {"items":"b1", "delivery_date":"b2", ...}},
{"order_102": {"items":"c1", "delivery_date":"c2", ...}},
{"order_103": {"items":"d1", "delivery_date":"d2", ...}},
{"order_104": {"items":"e1", "delivery_date":"e2", ...}},
{"order_105": {"items":"f1", "delivery_date":"f2", ...}},
{"order_106": {"items":"g1", "delivery_date":"g2", ...}},
{"order_107": {"items":"h1", "delivery_date":"h2", ...}},
{"order_108": {"items":"i1", "delivery_date":"i2", ...}},
{"order_109": {"items":"j1", "delivery_date":"j2", ...}},
{"order_110": {"items":"k1", "delivery_date":"k2", ...}}
```

Encoder Decoder Attention

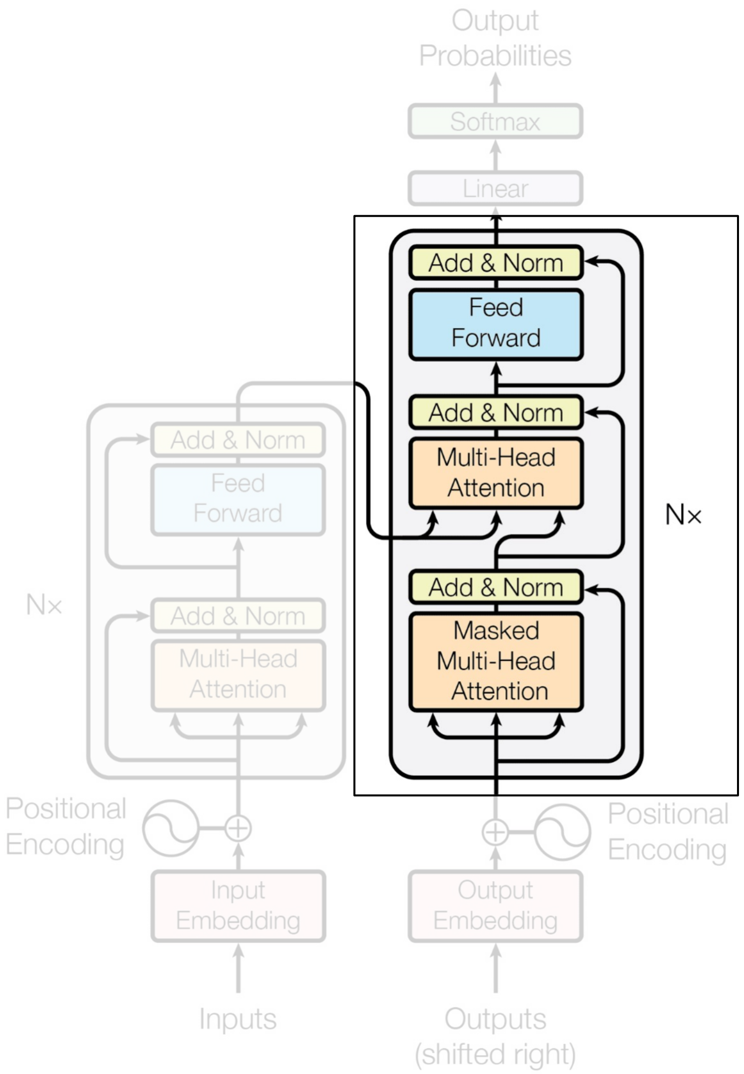
Encoder

Keys from **Encoder Outputs**
Values from **Encoder Outputs**

Decoder

Queries from **Decoder Inputs**

NOTE: Every decoder block receives the same FINAL encoder output



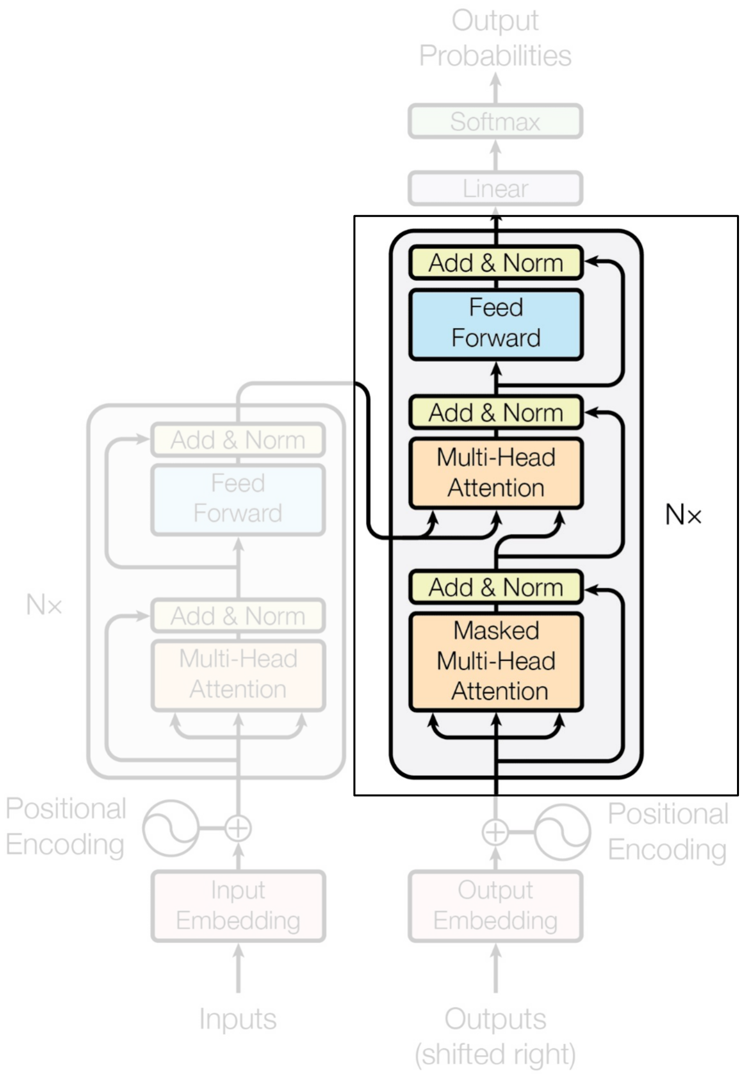
Encoder Decoder Attention

$R^{T_d \times d_{model}}$ Z''

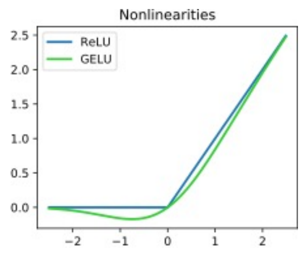
$R^{T_d \times T_e}$ $\text{softmax}\left(\frac{Q_d {K_e}^T}{\sqrt{d_{model}}}\right).$ V_e $R^{T_e \times d_{model}}$

$R^{T_d \times T_e}$ $\text{softmax}\left(\frac{Q_d {K_e}^T}{\sqrt{d_{model}}}\right)$

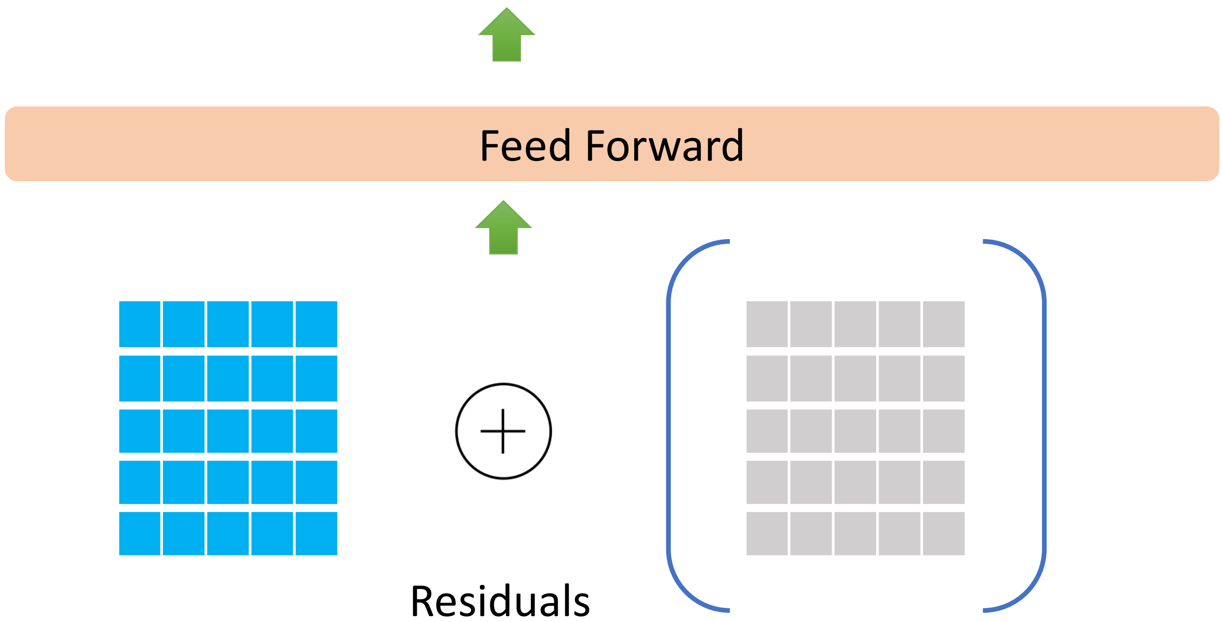
$R^{T_d \times d_{model}}$ Q_d K_e $R^{T_e \times d_{model}}$



Encoder Decoder Attention

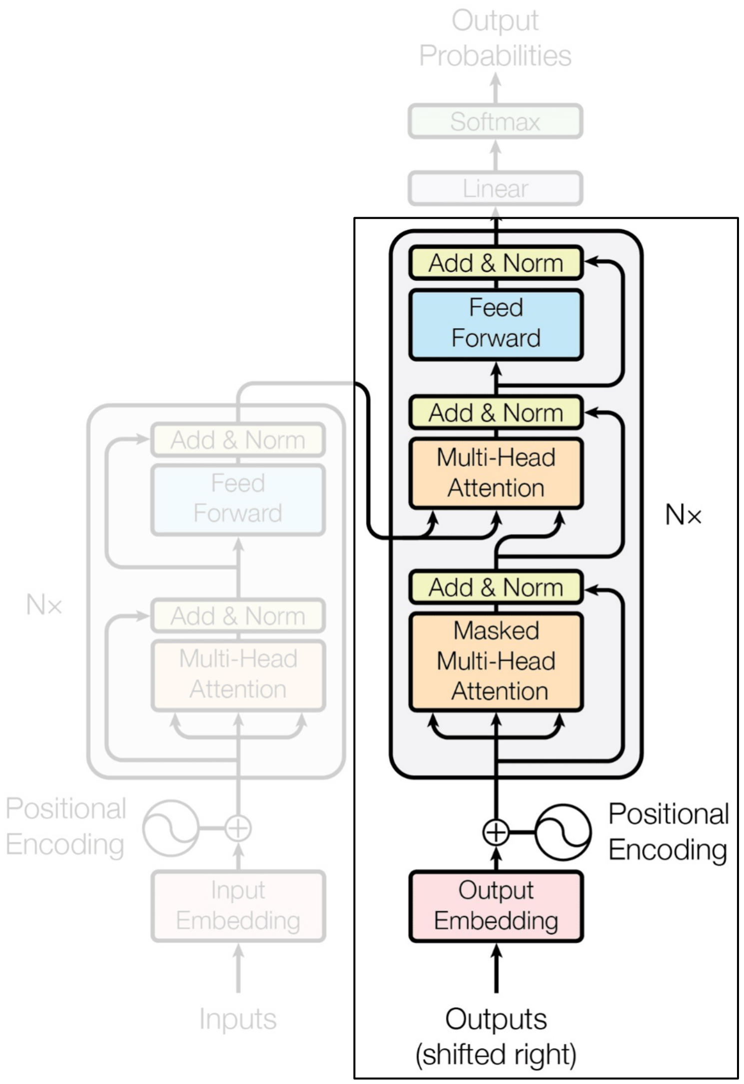


- Non Linearity
- Complex Relationships
- Learn from each other



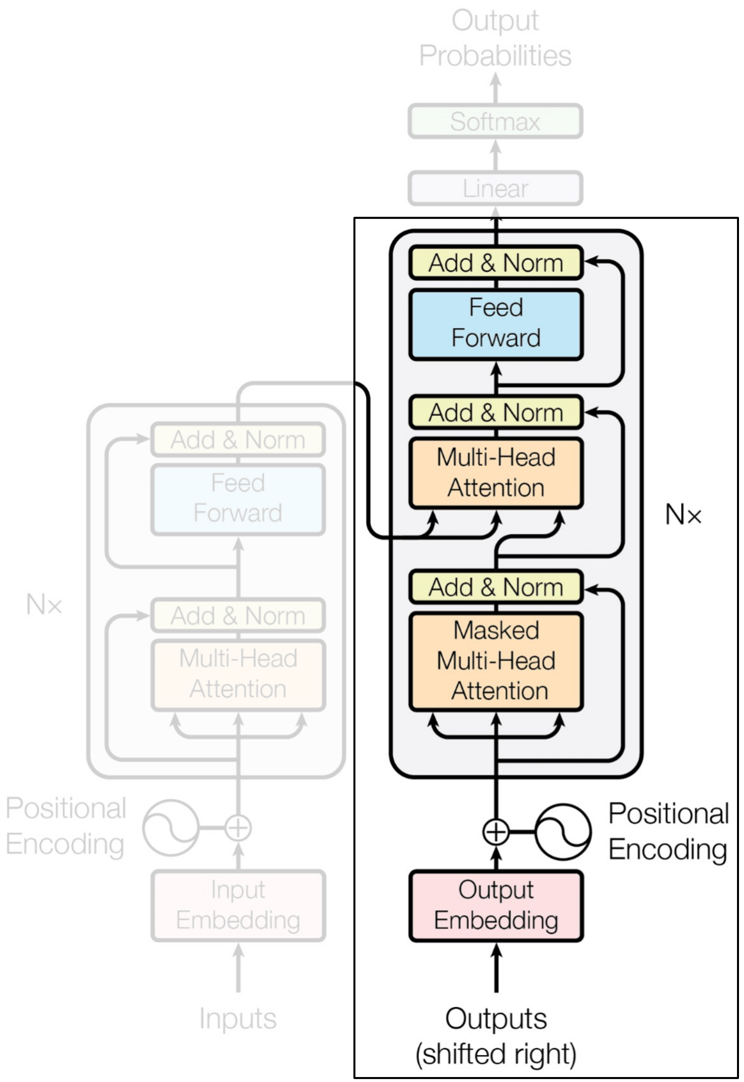
Add n Norm Decoder Self Attn

Norm(Z'')

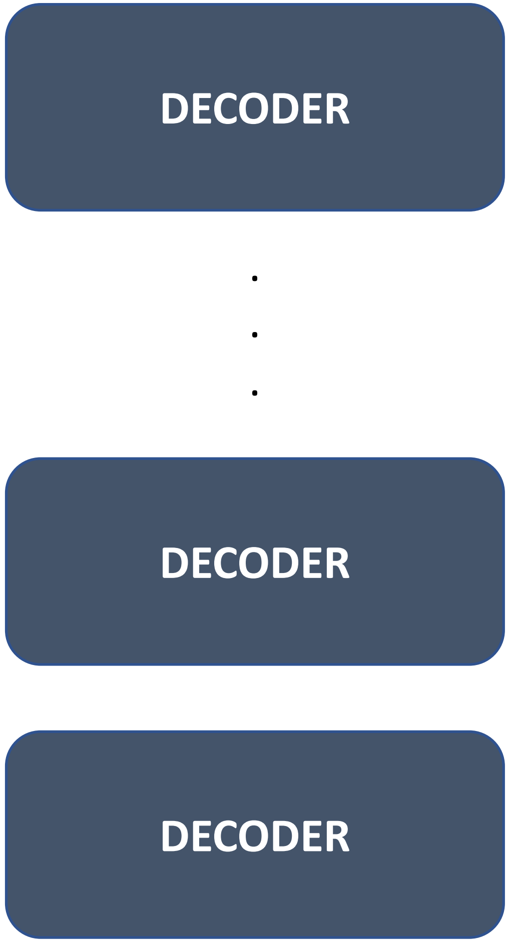


Decoder

DECODER

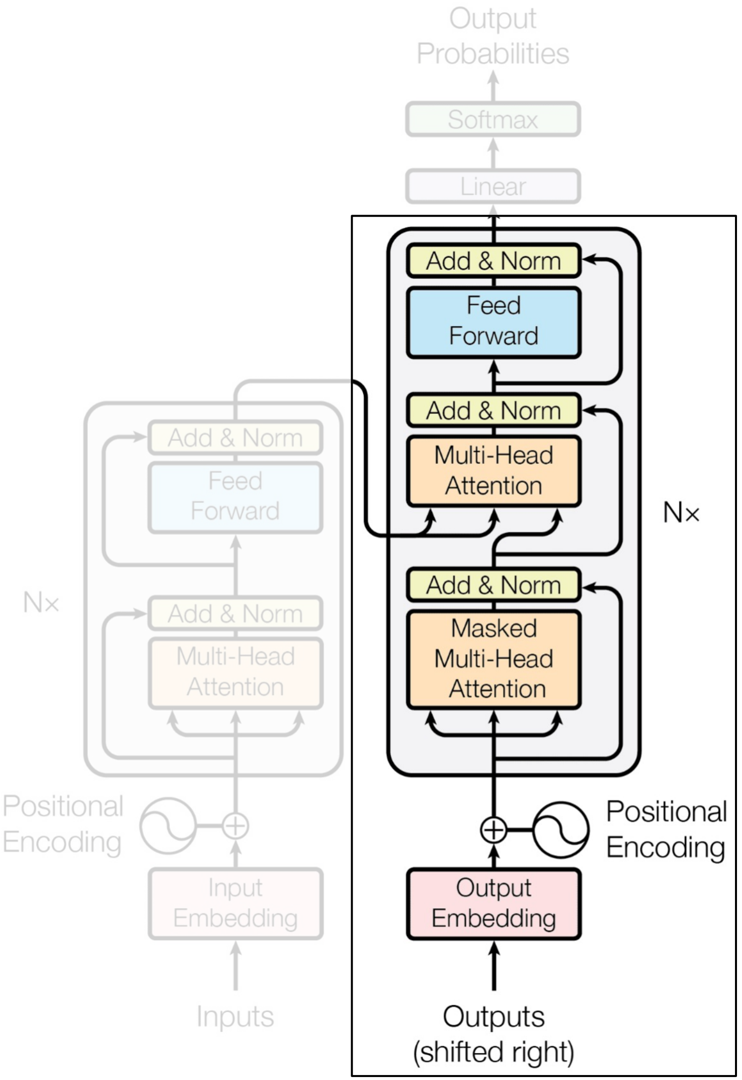


Decoder

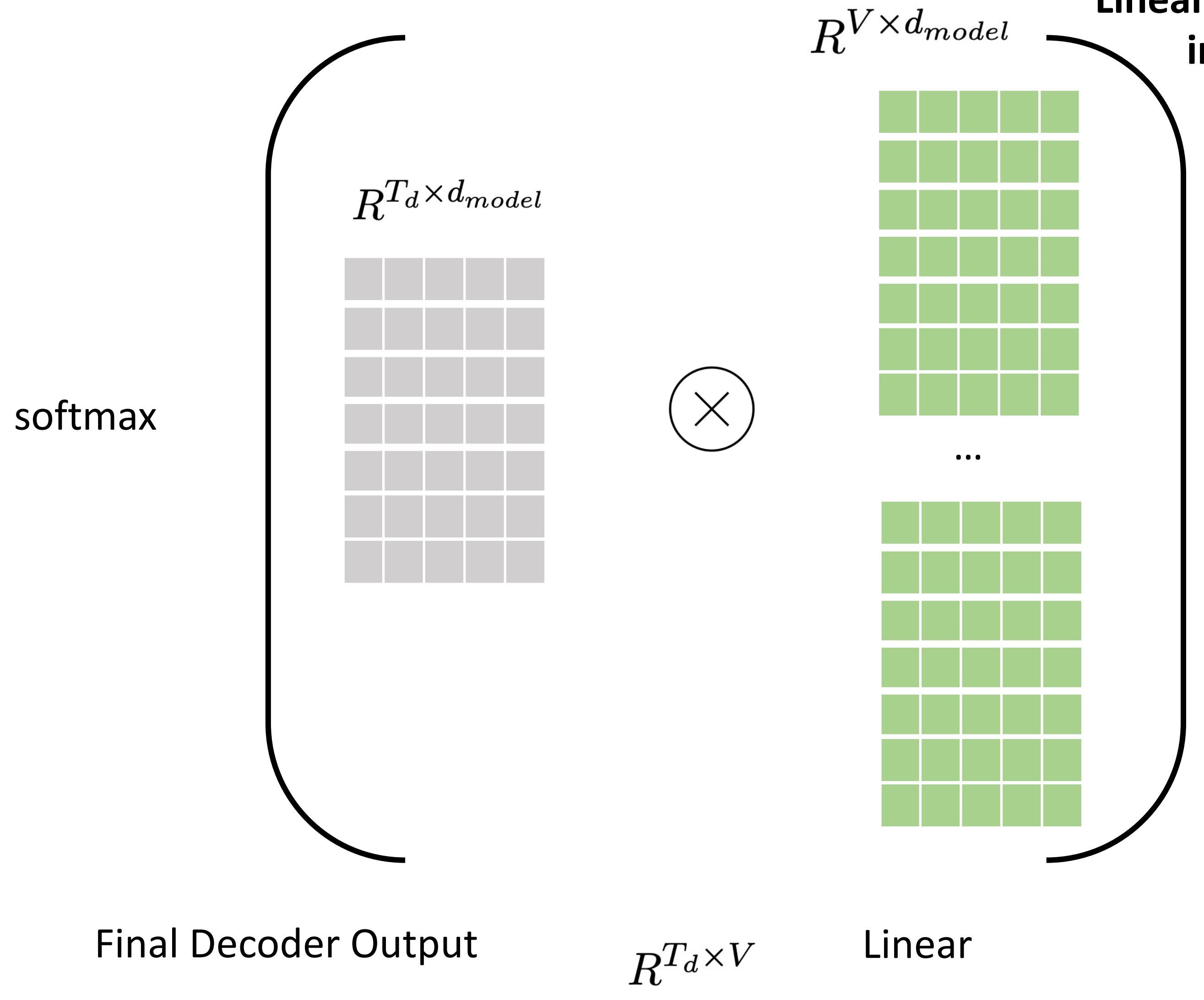


$R^{T_d \times d_{model}}$

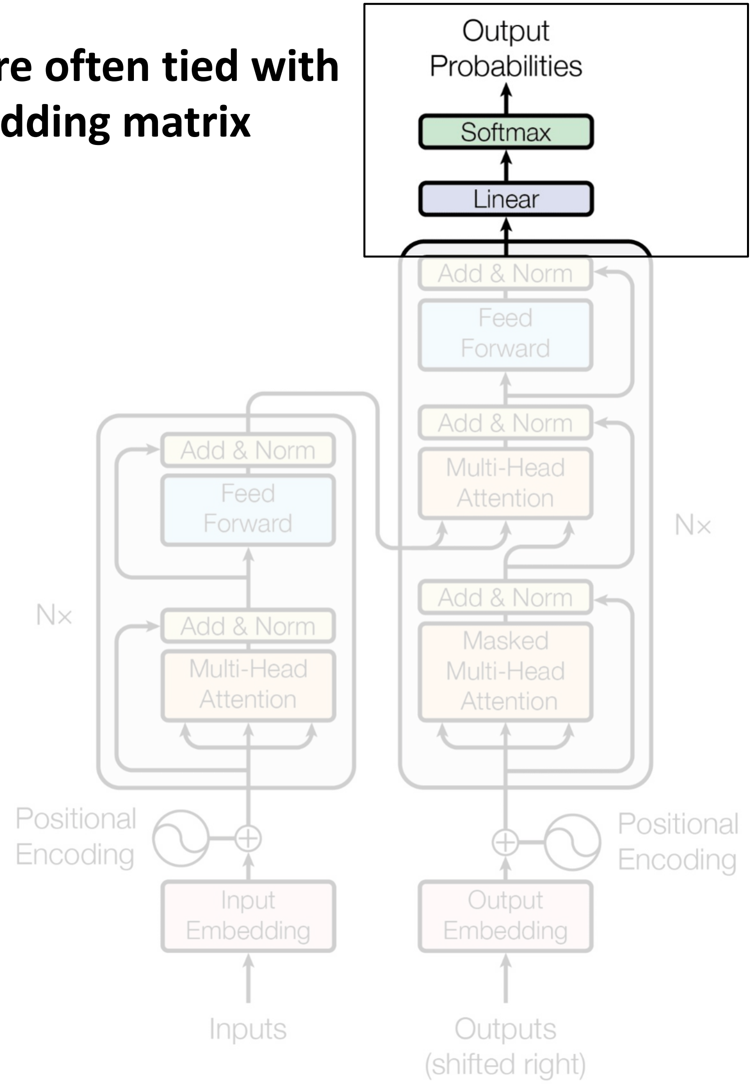
Decoder output



Linear

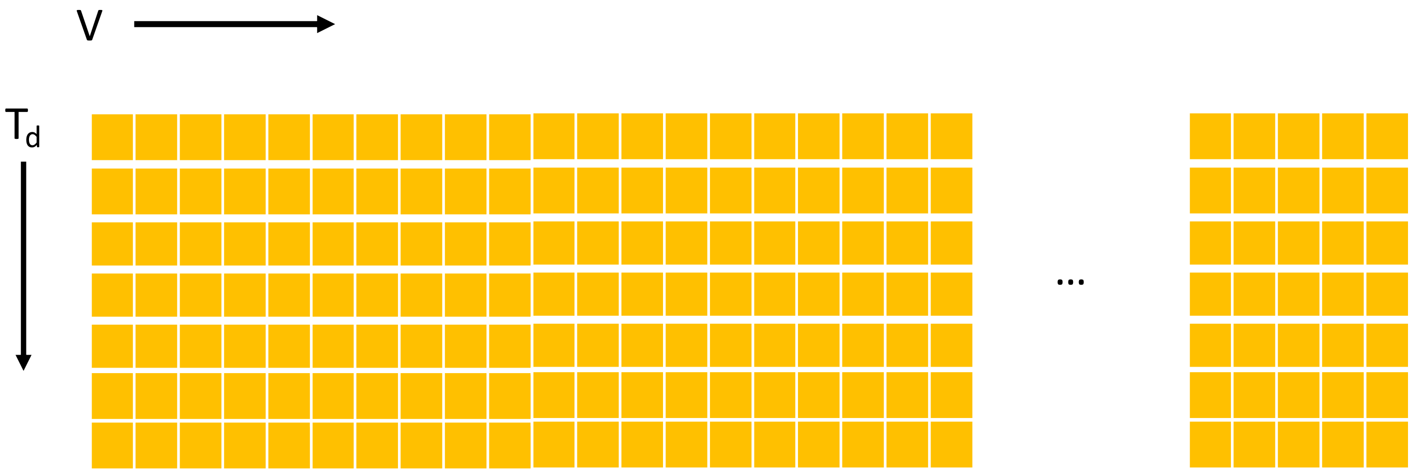


Linear weights are often tied with input embedding matrix

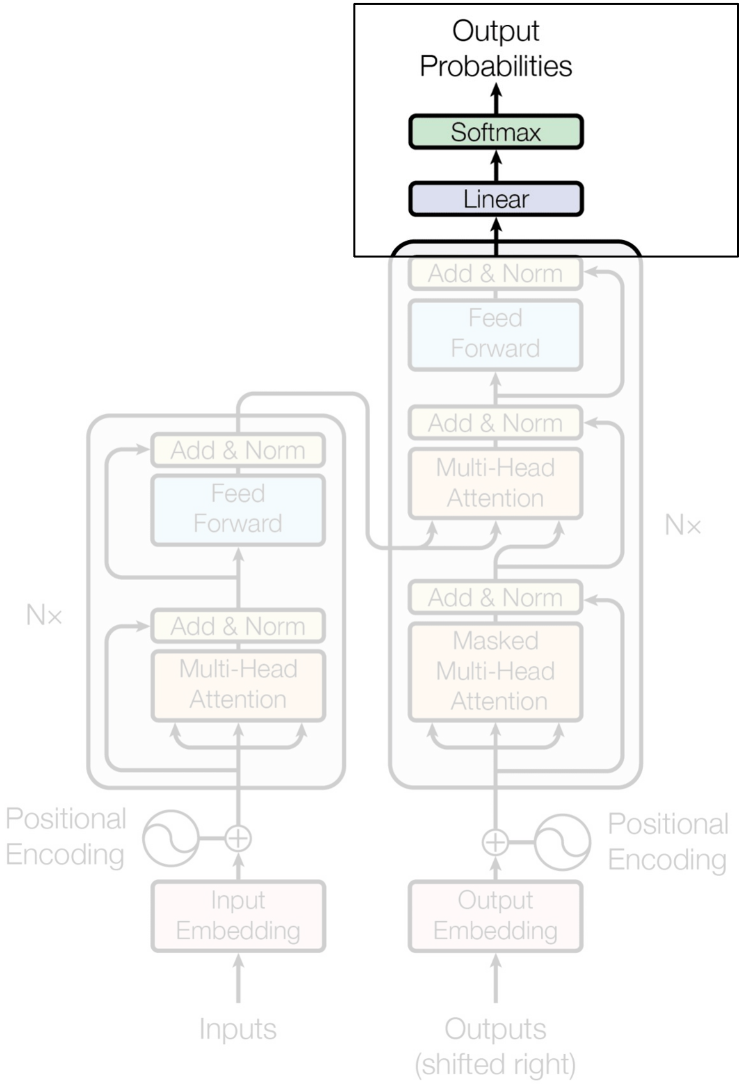


Softmax

Output Probabilities



$$R^{T_d \times V}$$



Poll 2 (@1297)

Which of the following are true about transformers?

a. Transformers can always be run in parallel
b. Transformer decoders can only be parallelized during training
c. Positional encodings help parallelize the transformer encoder
d. Queries, keys, and values are obtained by splitting the input into 3 equal segments
e. Multiheaded attention helps transformers find different kinds of relations between the tokens
f. During decoding, decoder outputs function as queries and keys while the values come from the encoder

Poll 2 (@1126)

Which of the following are true about transformers?

a. Transformers can always be run in parallel

b. **Transformer decoders can only be parallelized during training**

c. Positional encodings help parallelize the transformer encoder

d. Queries, keys, and values are obtained by splitting the input into 3 equal segments

e. **Multiheaded attention helps transformers find different kinds of relations between the tokens**

f. During decoding, decoder outputs function as queries and keys while the values come from the encoder

Transformers

Targets

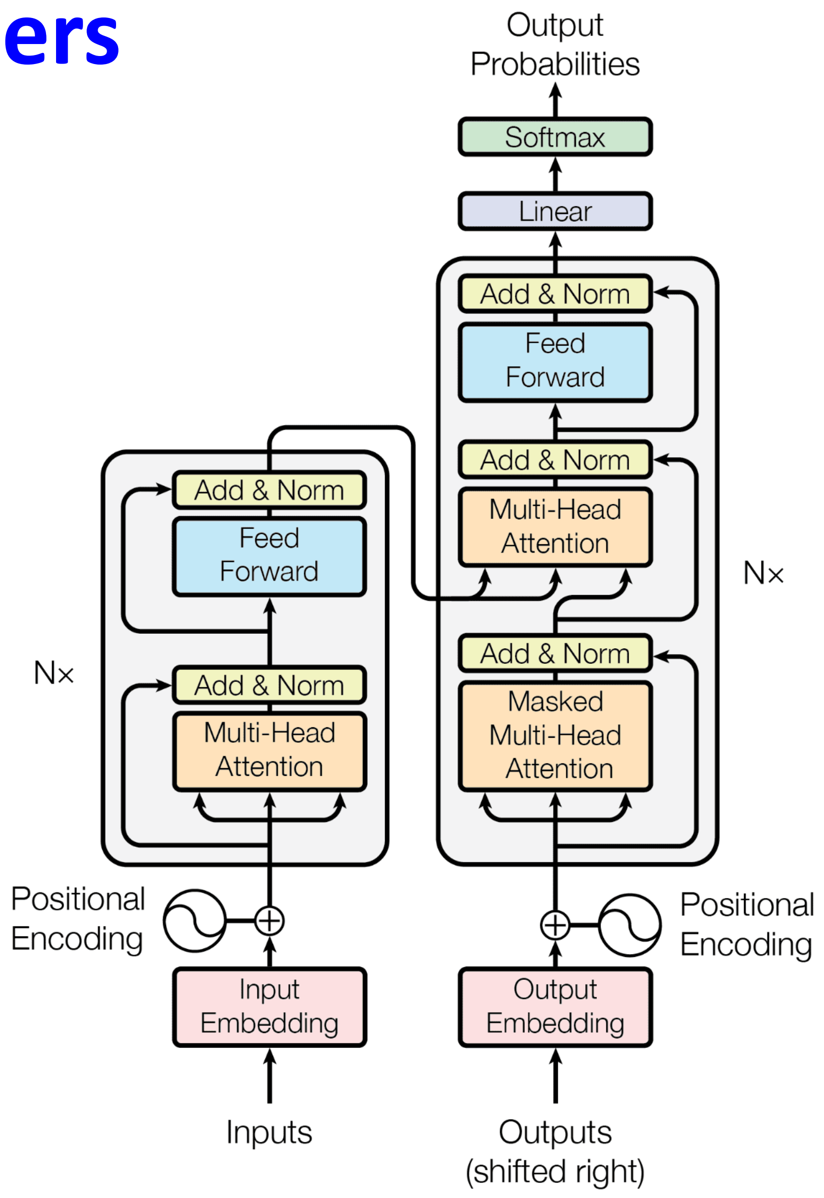
Ich have einen apfel gegessen

⬆

Inputs

I ate an apple

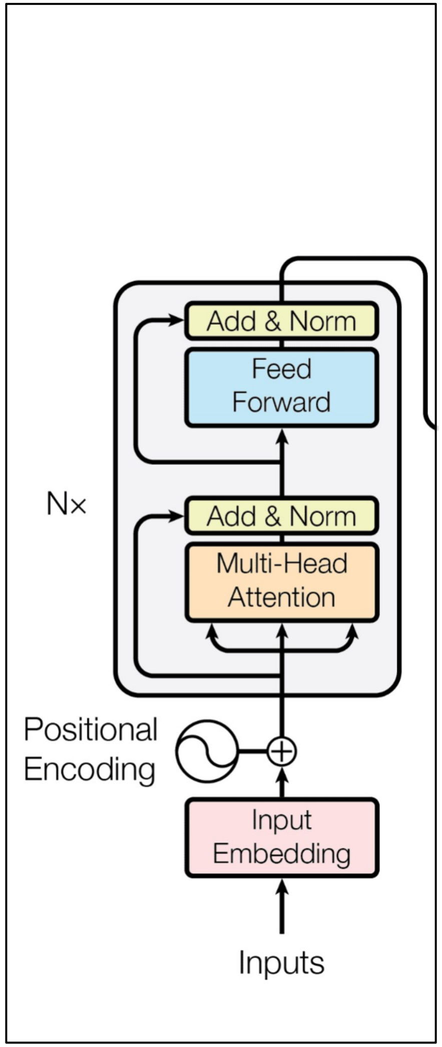
Machine Translation



Transformers

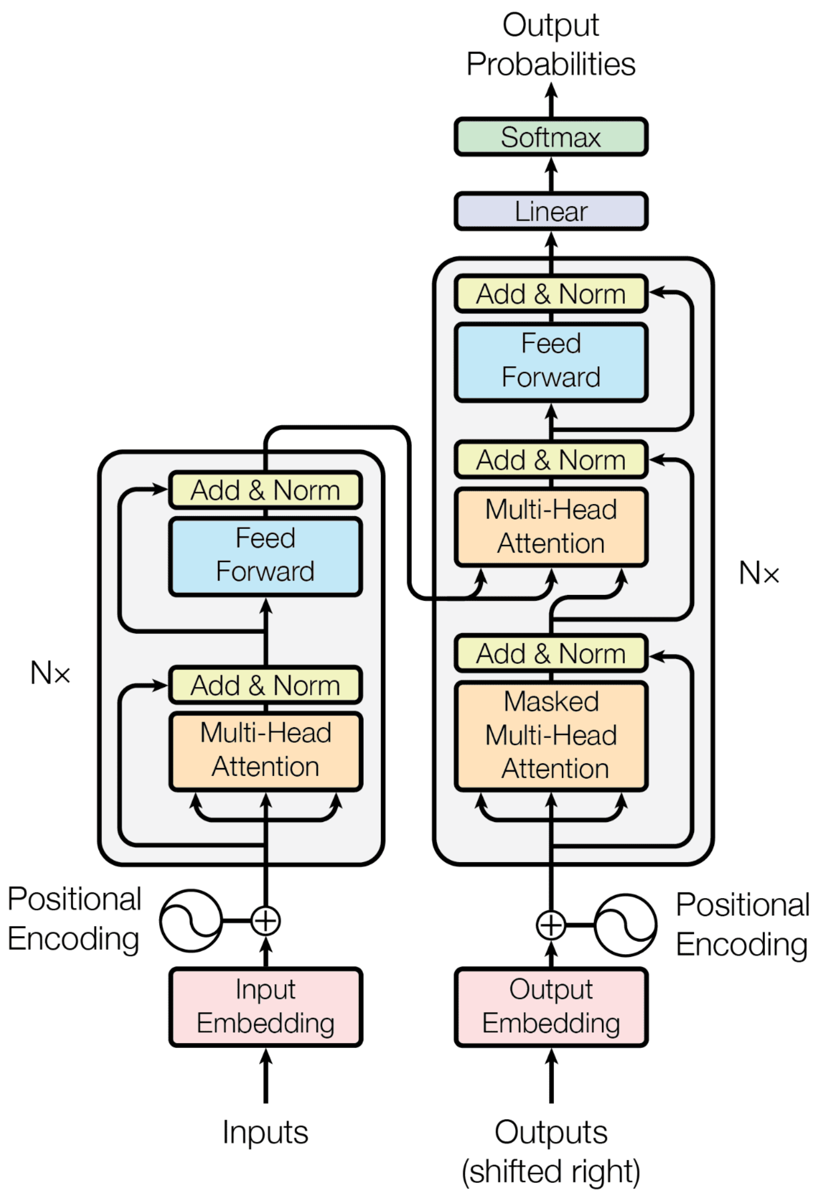
- ✓ Tokenizaton
- ✓ Input Embeddings
- ✓ Position Encodings
- ✓ Residuals
- ✓ Query
- ✓ Key
- ✓ Value
- ✓ Add & Norm
- ✓ Encoder
- ✓ Decoder

- ✓ Attention
- ✓ Self Attention
- ✓ Multi Head Attention
- ✓ Masked Attention
- ✓ Encoder Decoder Attention
- ✓ Output Probabilities / Logits
- ✓ Softmax
- **Encoder-Decoder models**
- **Decoder only models**



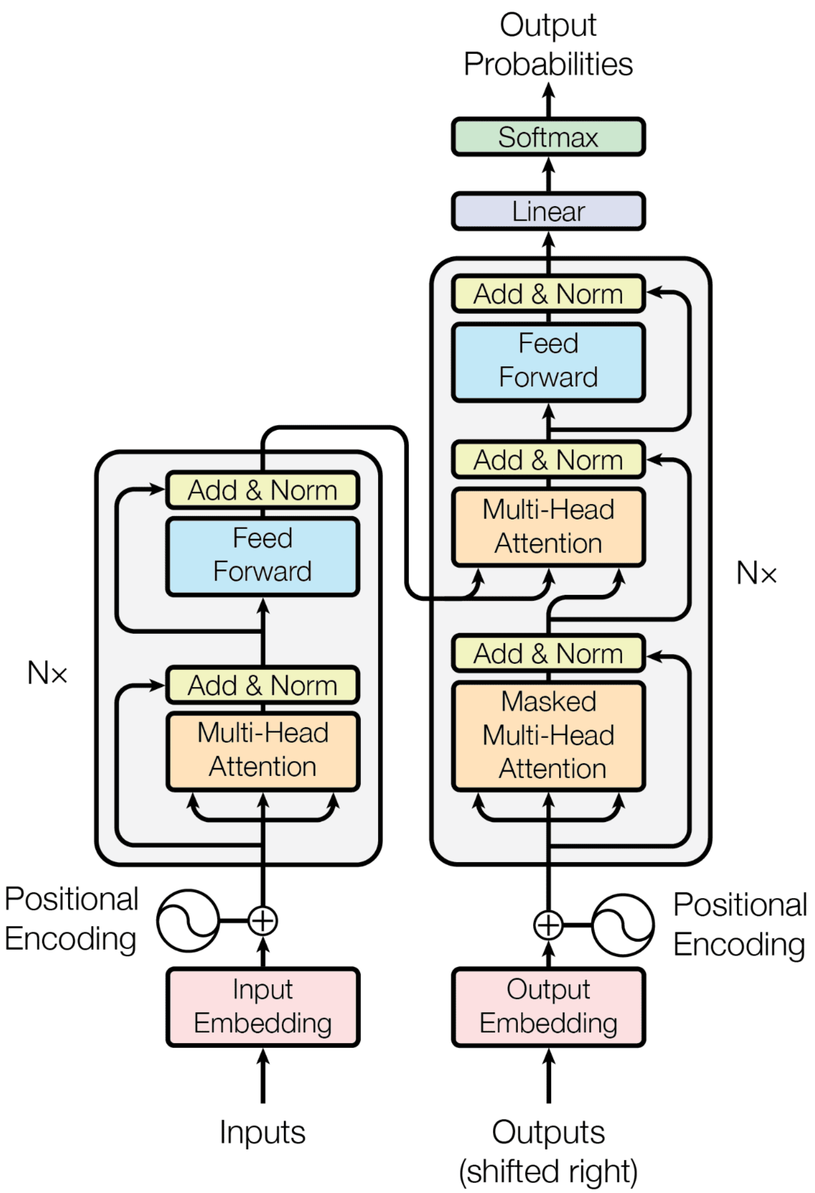
Part 2

LLM



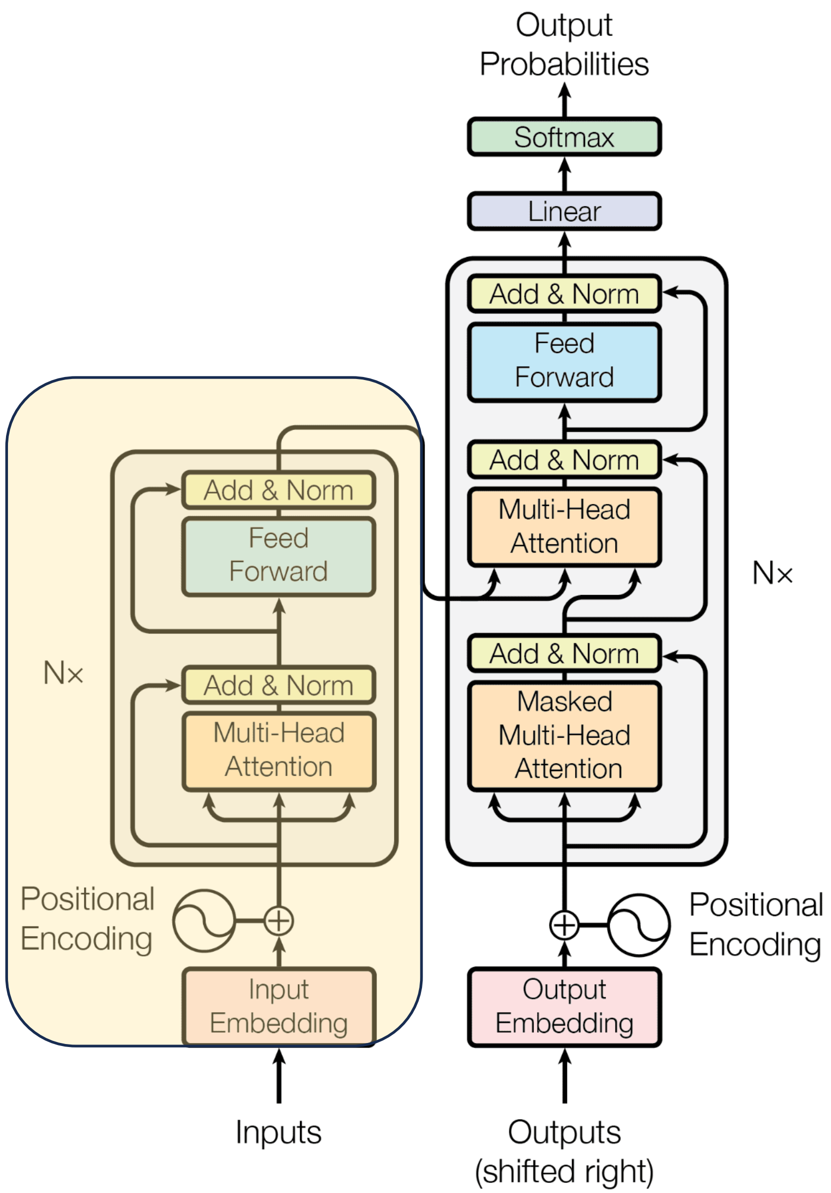
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Transformers, mid-2017



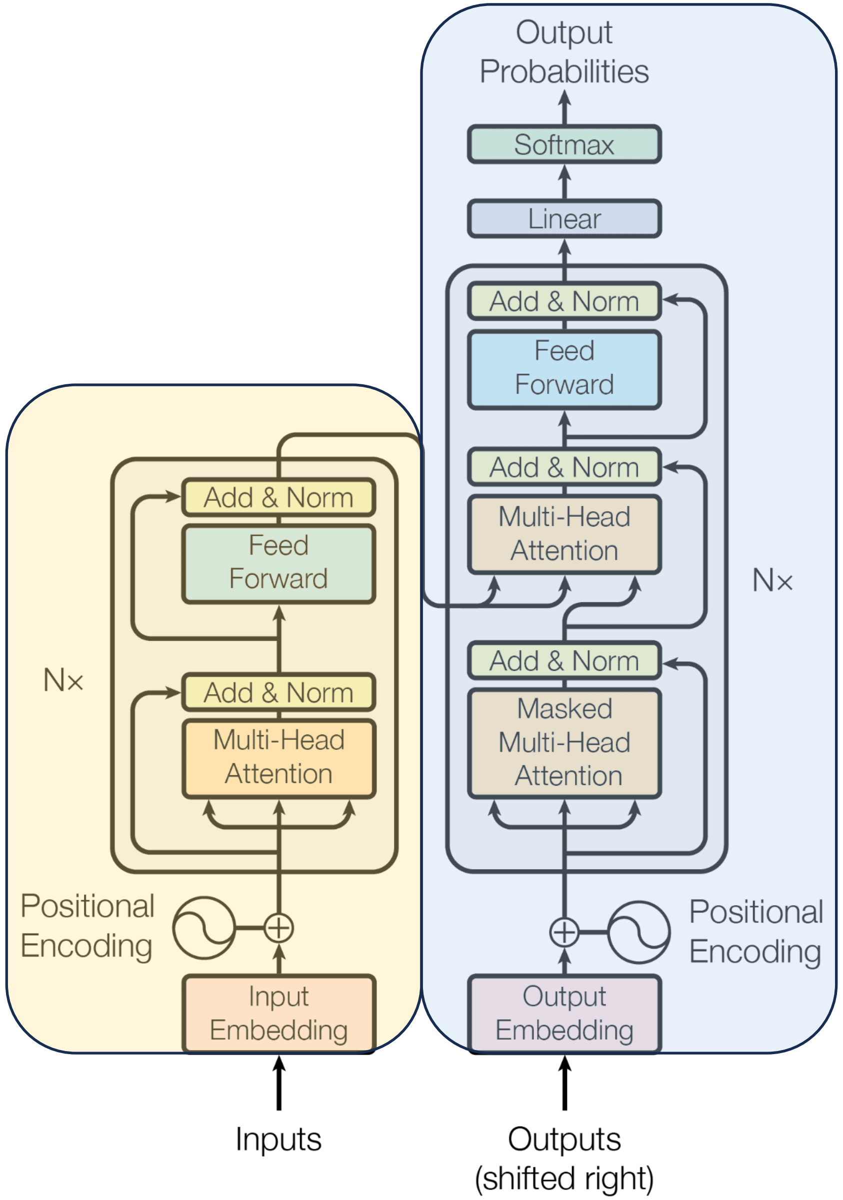
Transformers, mid-2017

Representation



Transformers, mid-2017

Representation

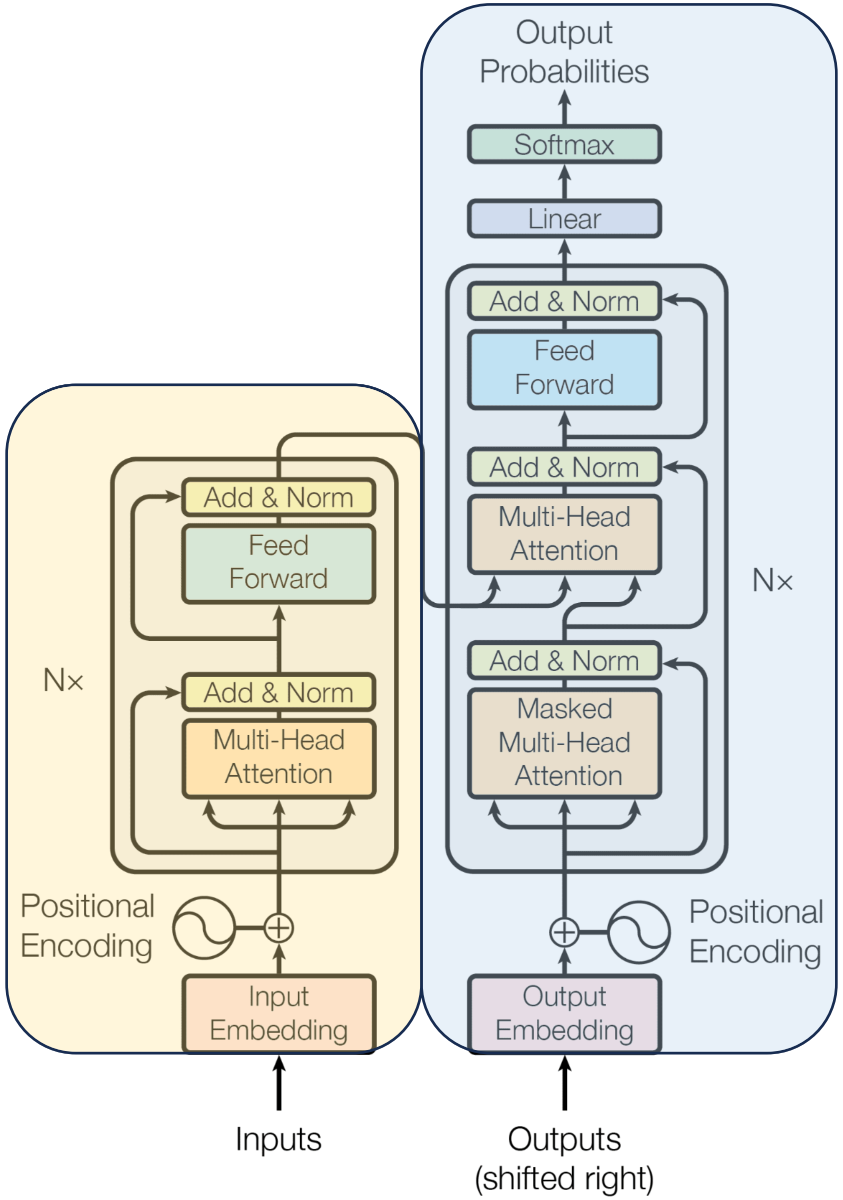


Generation

Transformers, mid-2017

Input – input tokens
Output – hidden states

Representation



Input – output tokens and hidden states*
Output – output tokens

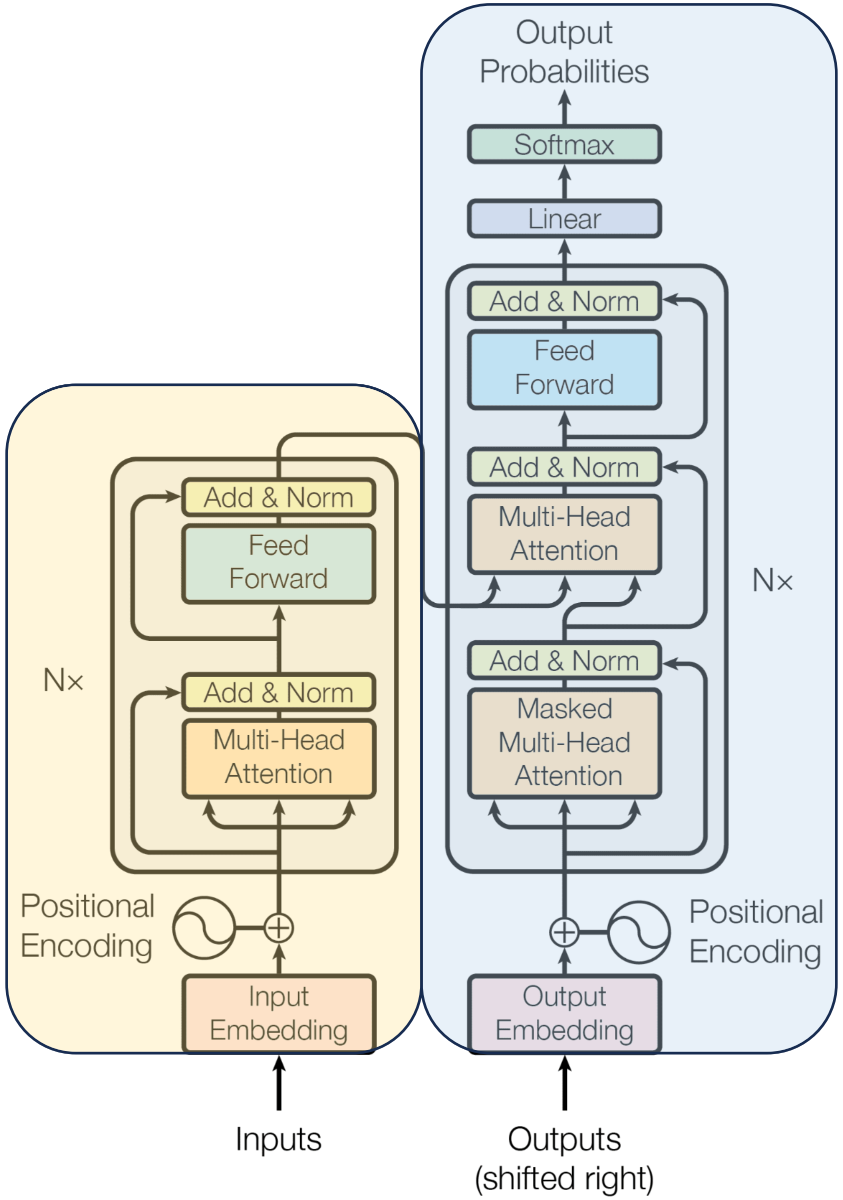
Generation

Transformers, mid-2017

Input – input tokens
Output – hidden states

Model can see all timesteps

Representation



Input – output tokens and hidden states*
Output – output tokens

Model can only see previous timesteps

Generation

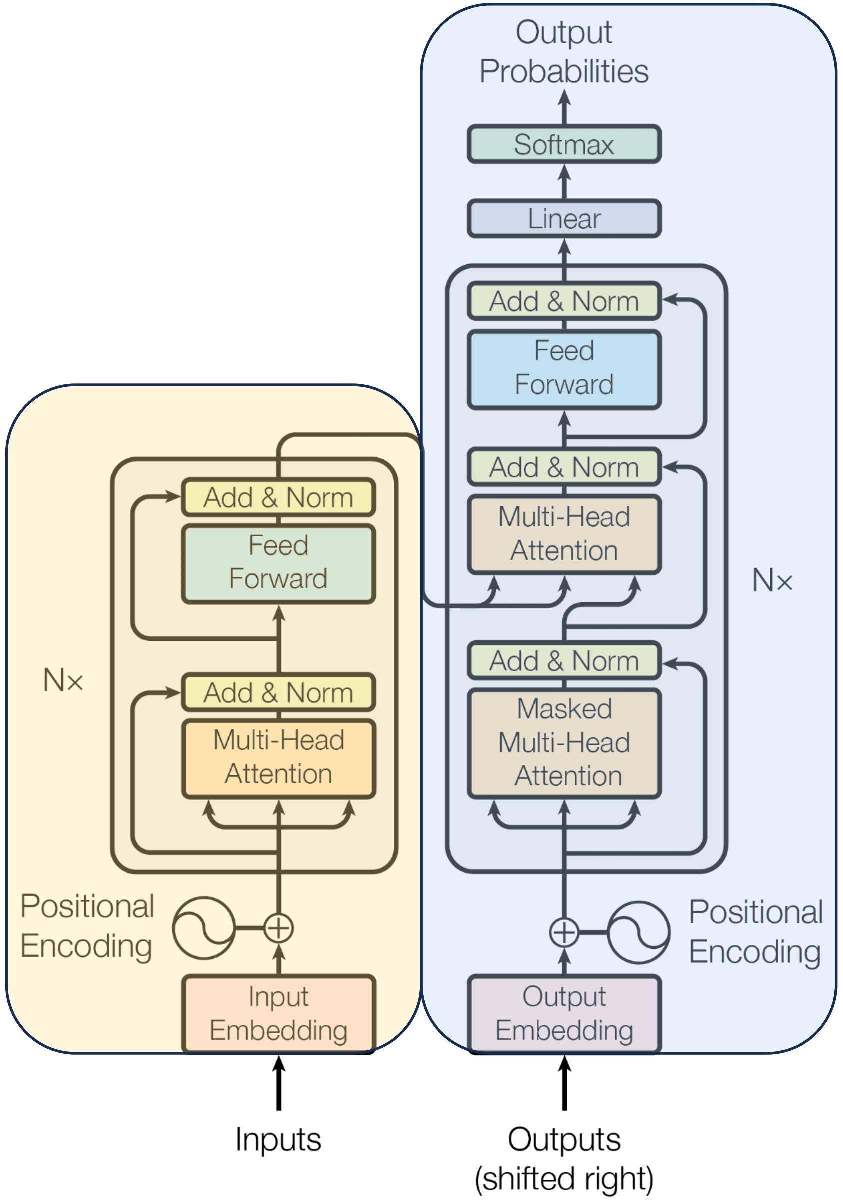
Transformers, mid-2017

Input – input tokens
Output – hidden states

Model can see all timesteps

Does not usually output tokens, so no inherent auto-regressivity

Representation



Input – output tokens and hidden states*
Output – output tokens

Model can only see previous timesteps

Model is auto-regressive with previous timesteps' outputs

Generation

Transformers, mid-2017

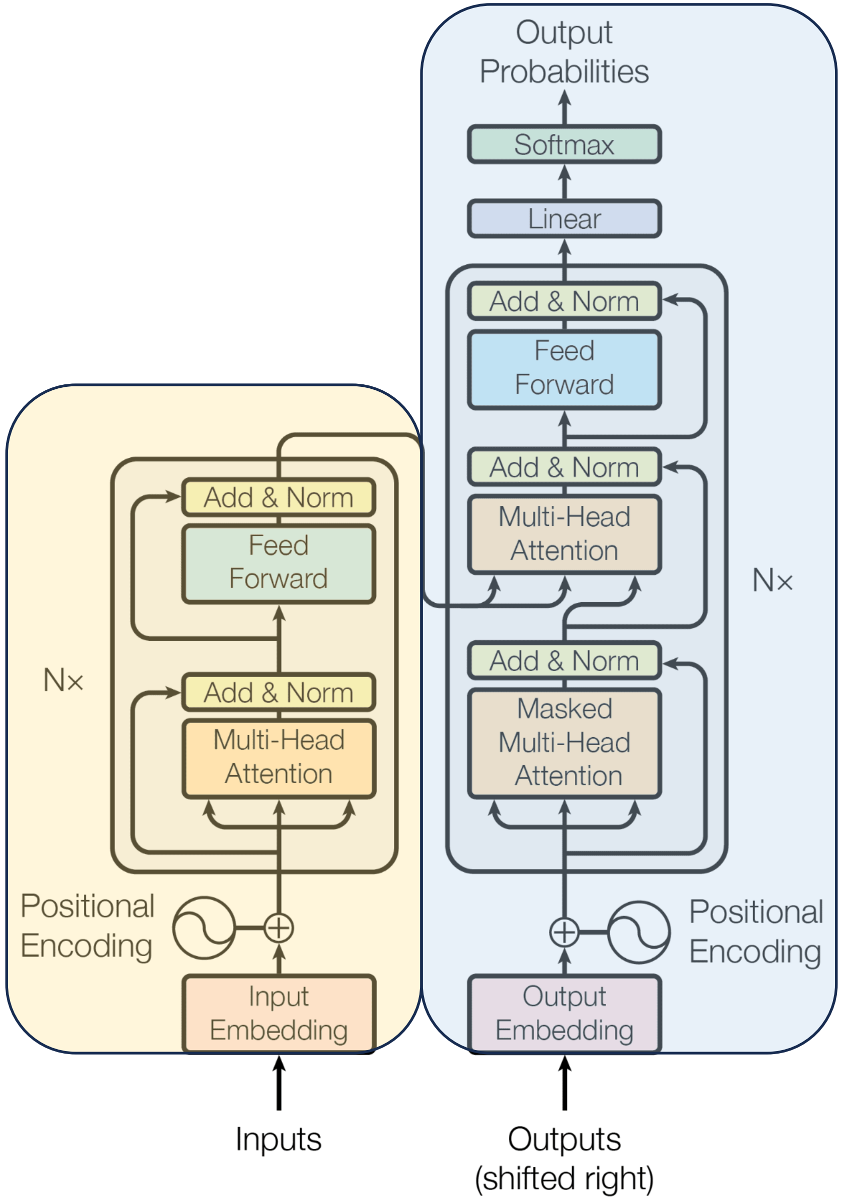
Input – input tokens
Output – hidden states

Model can see all timesteps

Does not usually output tokens, so no inherent auto-regressivity

Can also be adapted to generate tokens *by appending a module that maps hidden state dimensionality to vocab size*

Representation



Input – output tokens and hidden states*
Output – output tokens

Model can only see previous timesteps

Model is auto-regressive with previous timesteps' outputs

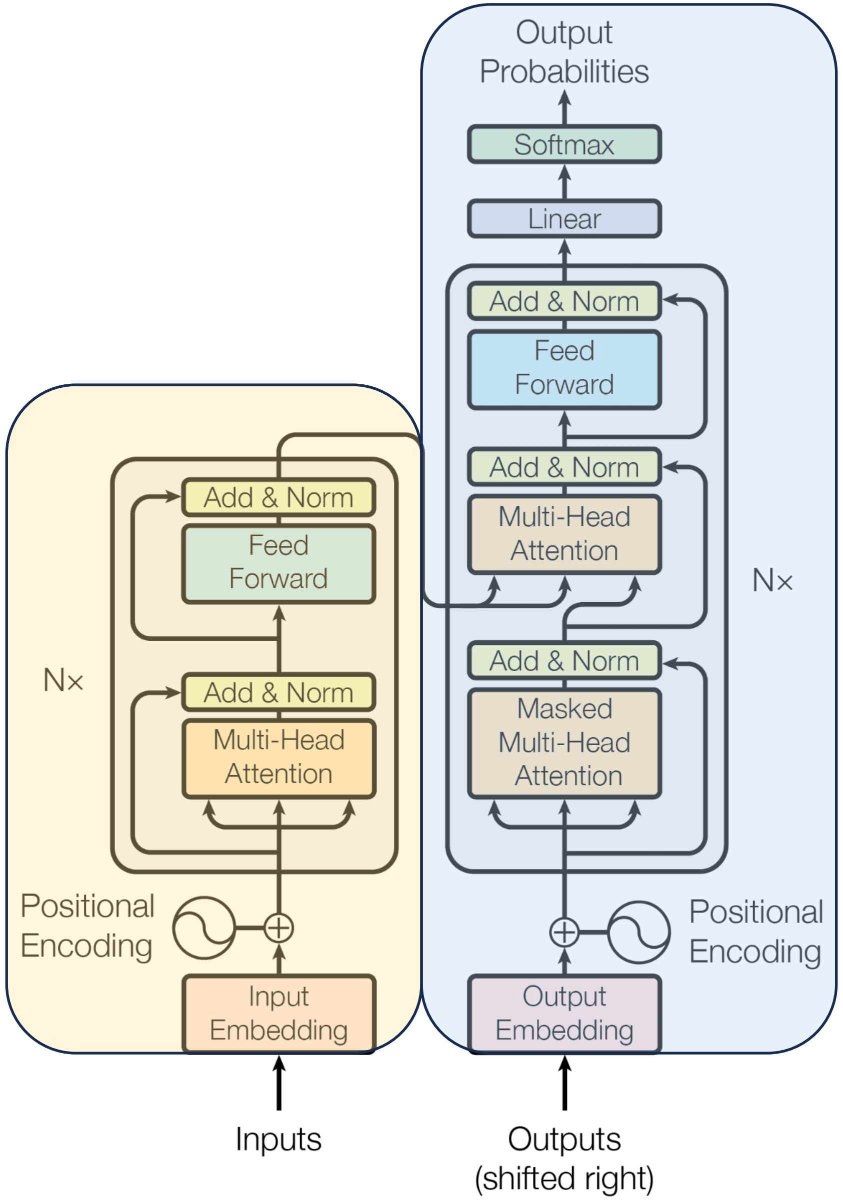
Can also be adapted to generate hidden states *by looking before token outputs*

Generation

2018 – The Inception of the LLM Era

BERT
Oct 2018

Representation

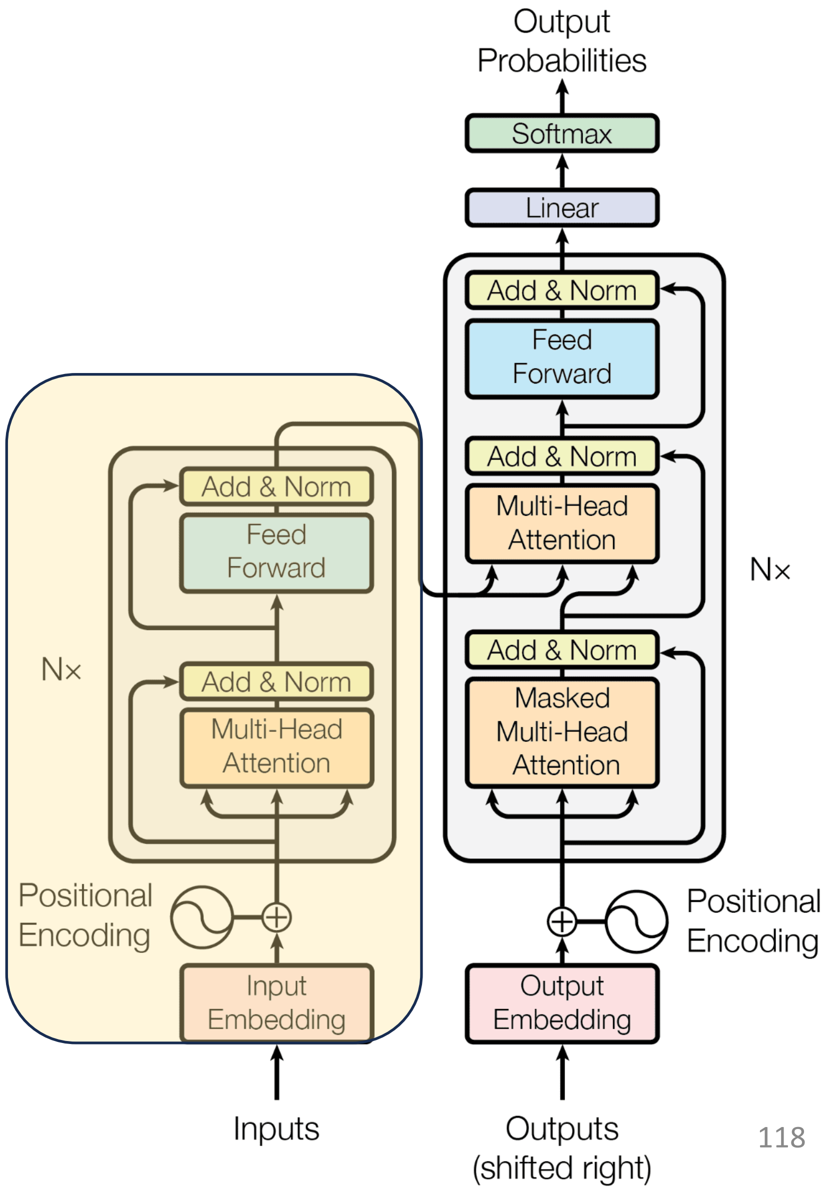


GPT
Jun 2018

Generation

BERT - Bidirectional Encoder Representations

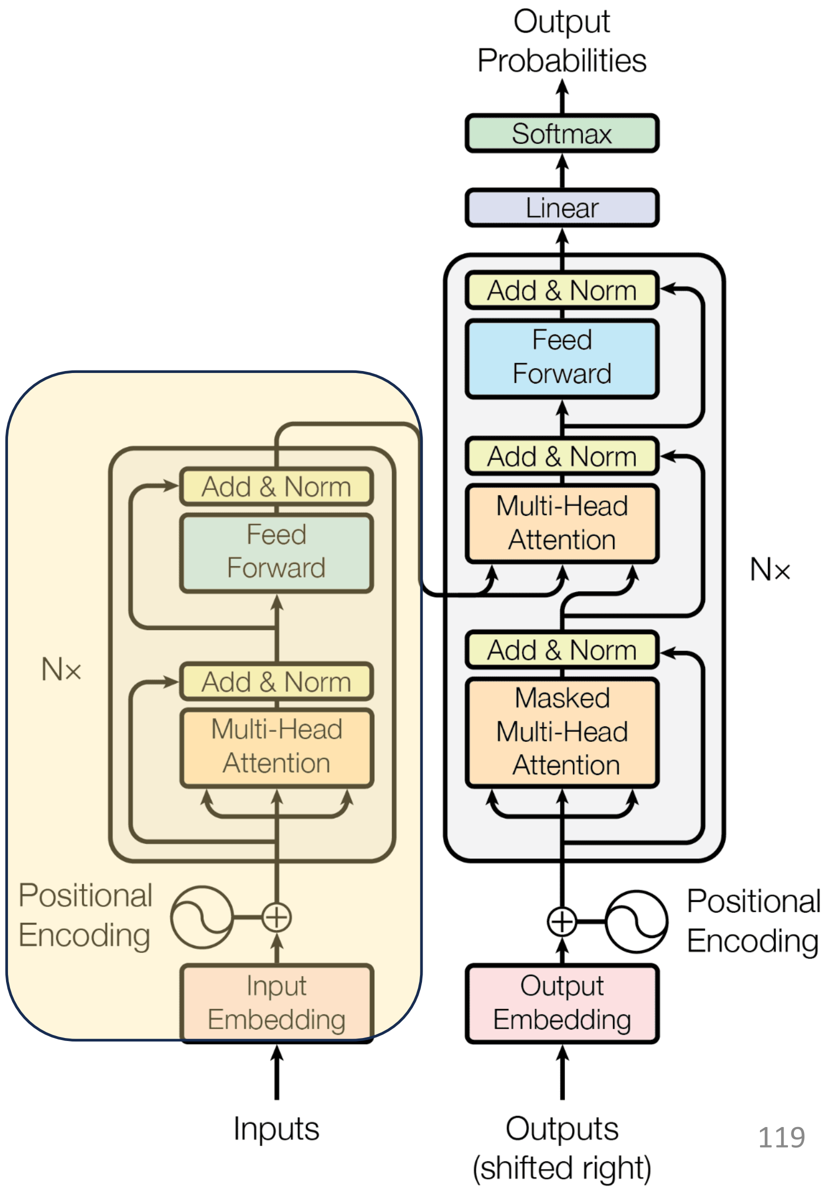
- One of the biggest challenges in LM-building used to be the lack of task-specific training data.
- What if we learn an effective representation that can be applied to a variety of downstream tasks?
  - Word2vec (2013)
  - GloVe (2014)



BERT - Bidirectional Encoder Representations

BERT Pre-Training Corpus:

- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words



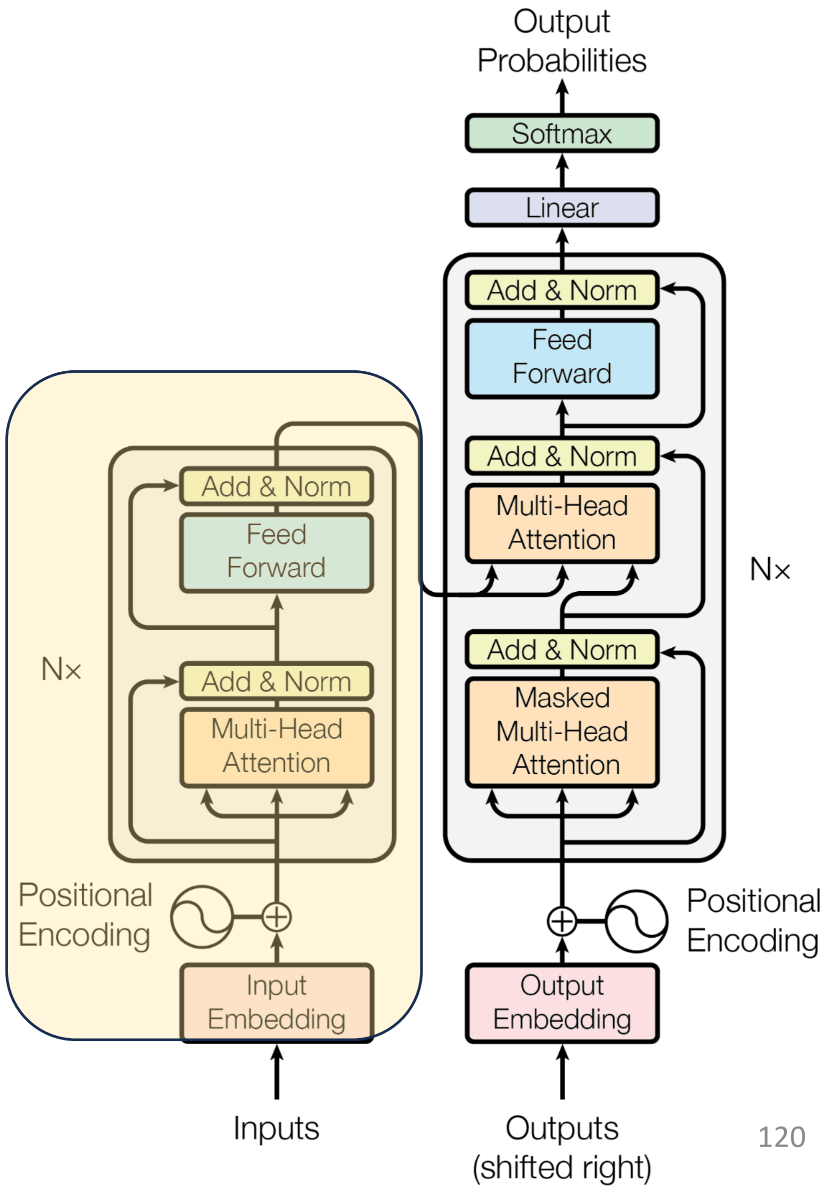
BERT - Bidirectional Encoder Representations

BERT Pre-Training Corpus:

- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words

BERT Pre-Training Tasks:

- MLM (Masked Language Modeling)
- NSP (Next Sentence Prediction)

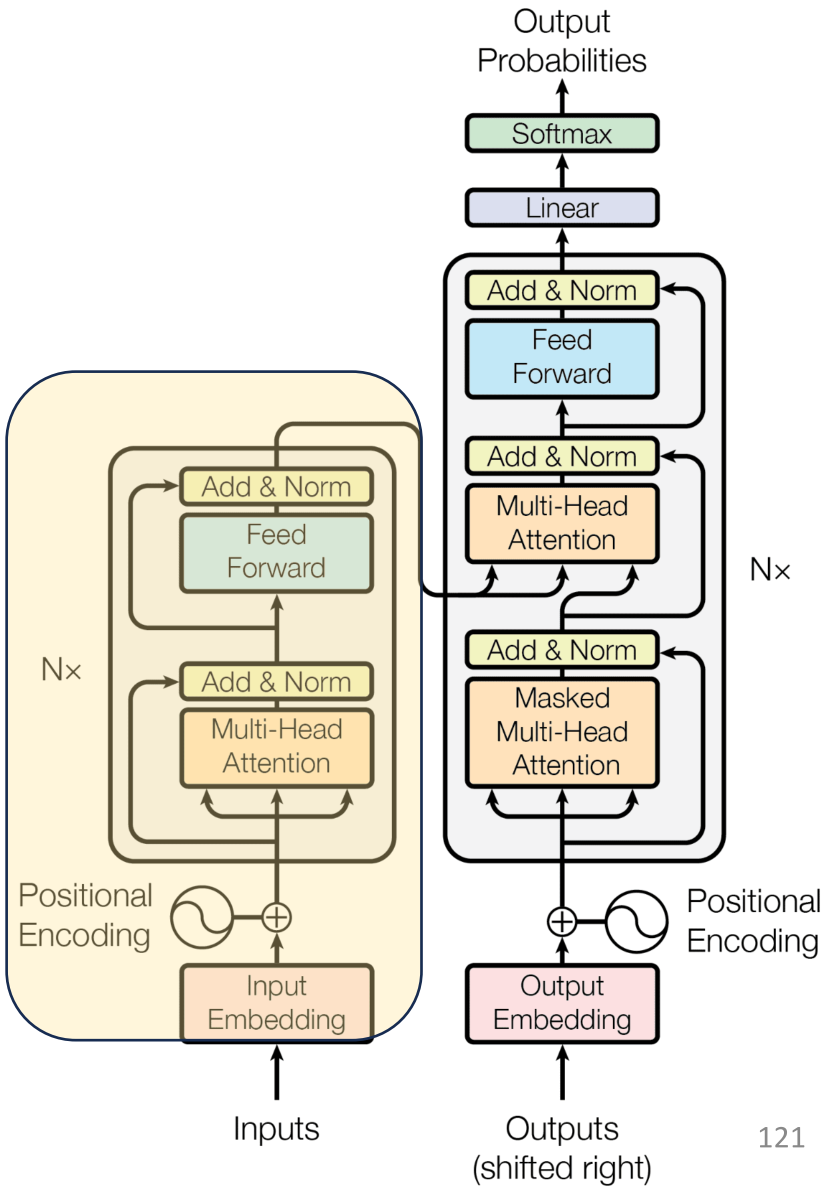


BERT - Bidirectional Encoder Representations

BERT Pre-Training Corpus:
- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words

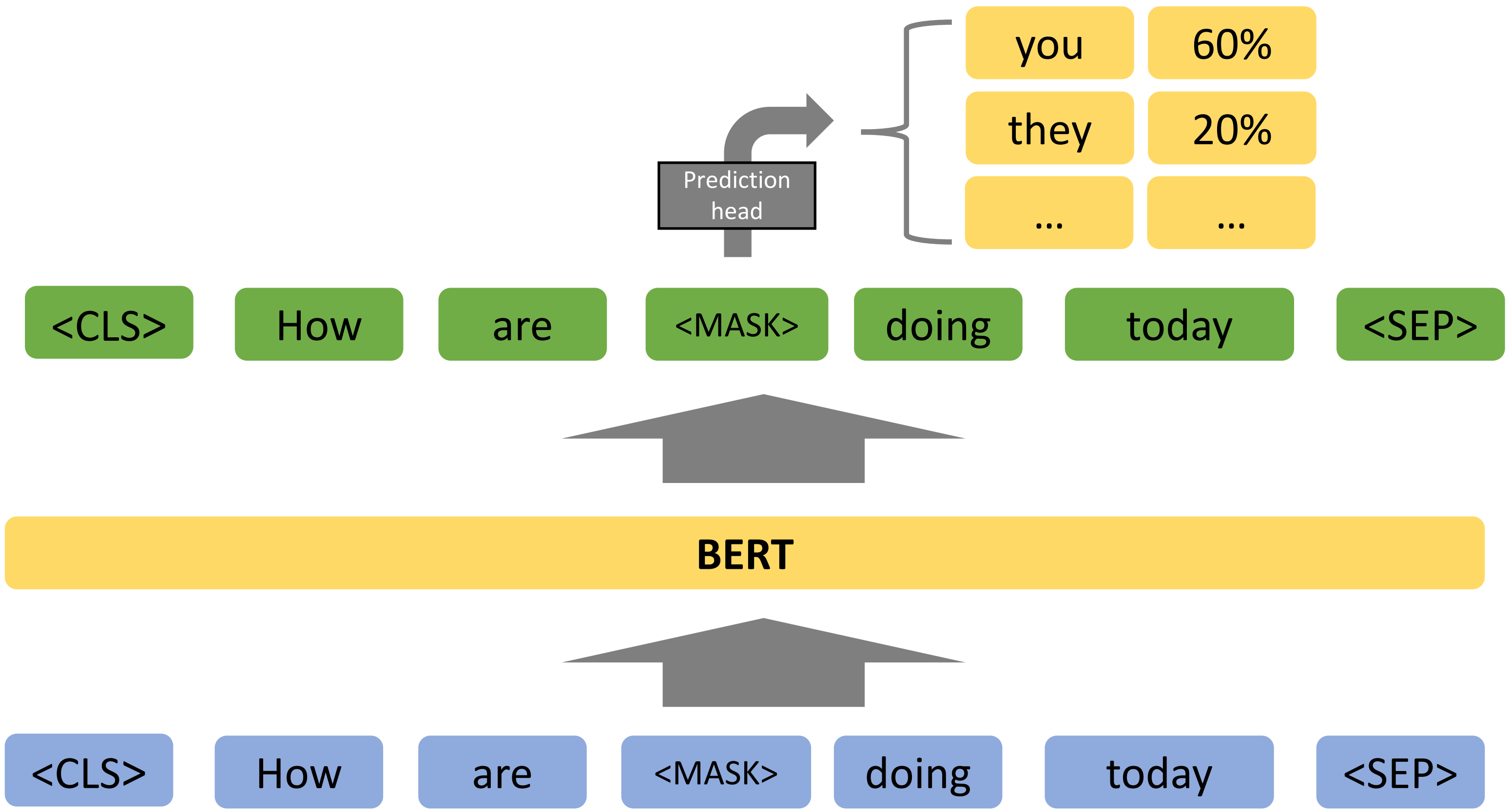
BERT Pre-Training Tasks:
- MLM (Masked Language Modeling)
- NSP (Next Sentence Prediction)

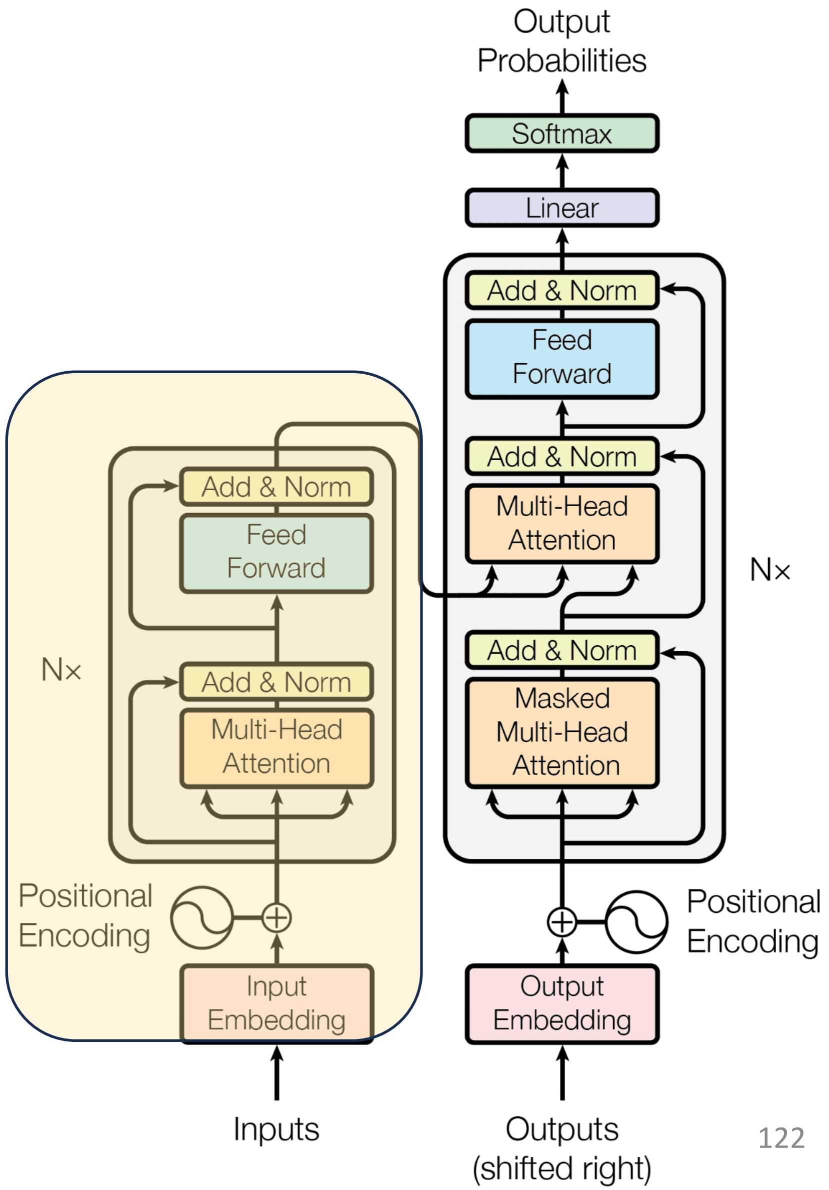
BERT Pre-Training Results:
- BERT-Base – 110M Params
- BERT-Large – 340M Params



BERT - Bidirectional Encoder Representations

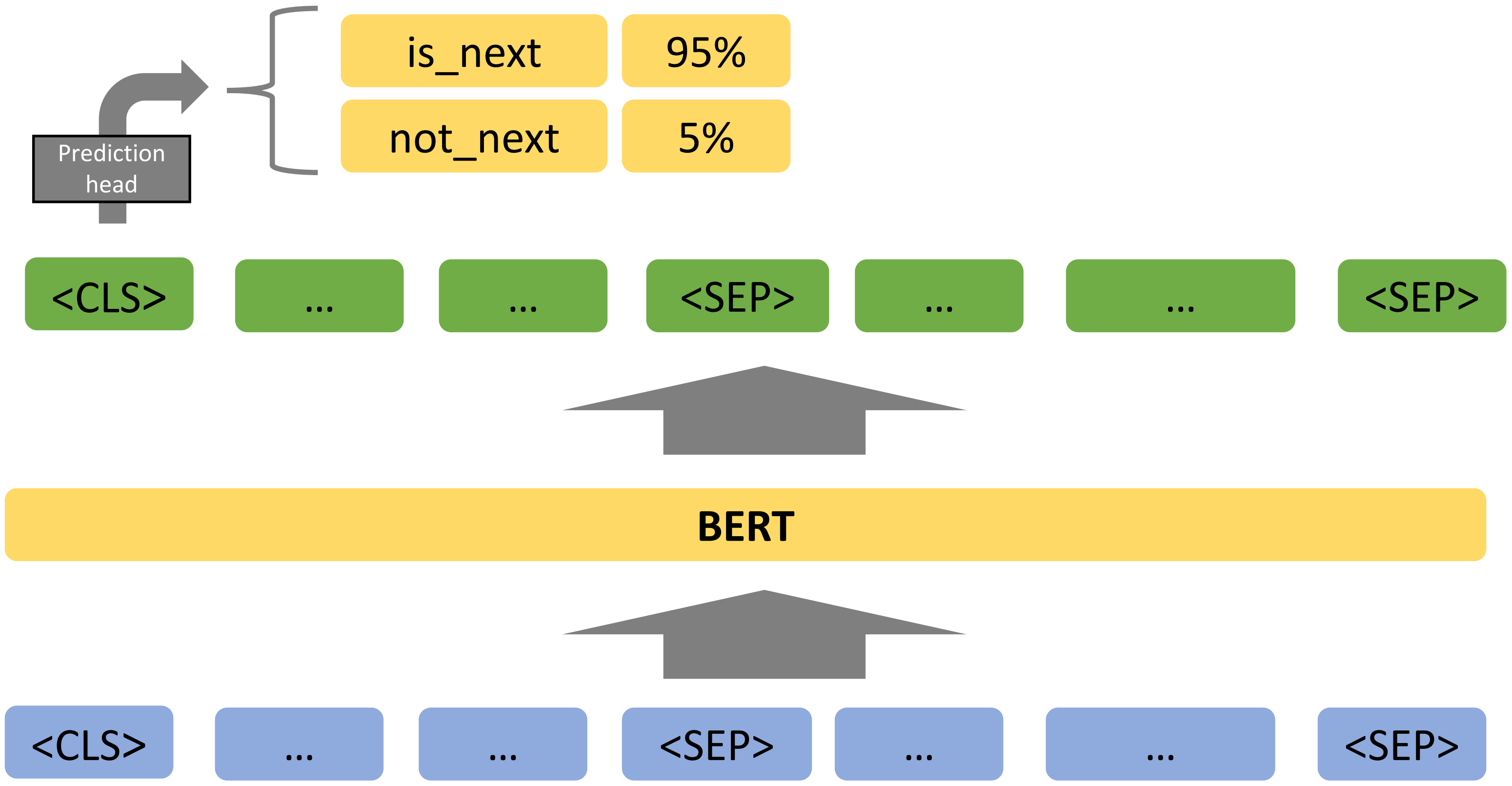
MLM (Masked Language Modeling)

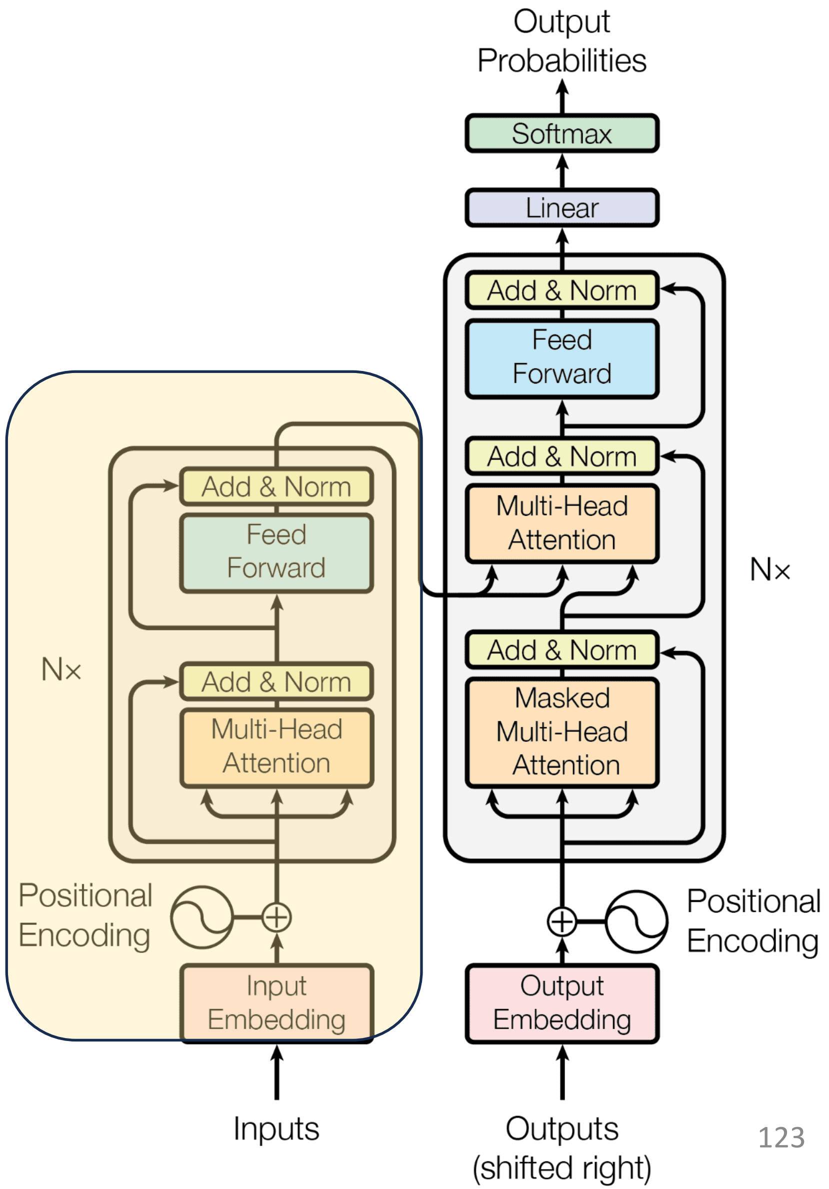




BERT - Bidirectional Encoder Representations

MLM (Masked Language Modeling)

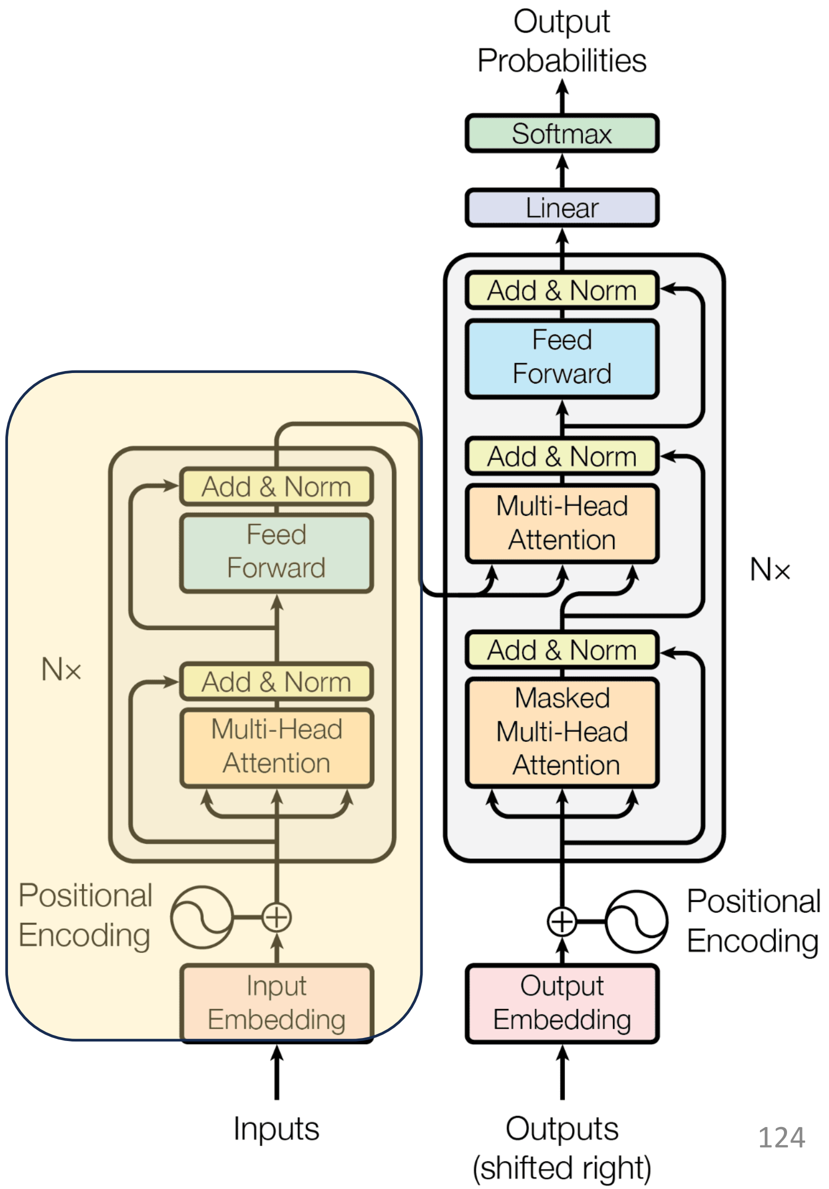




BERT - Bidirectional Encoder Representations

BERT Fine-Tuning:

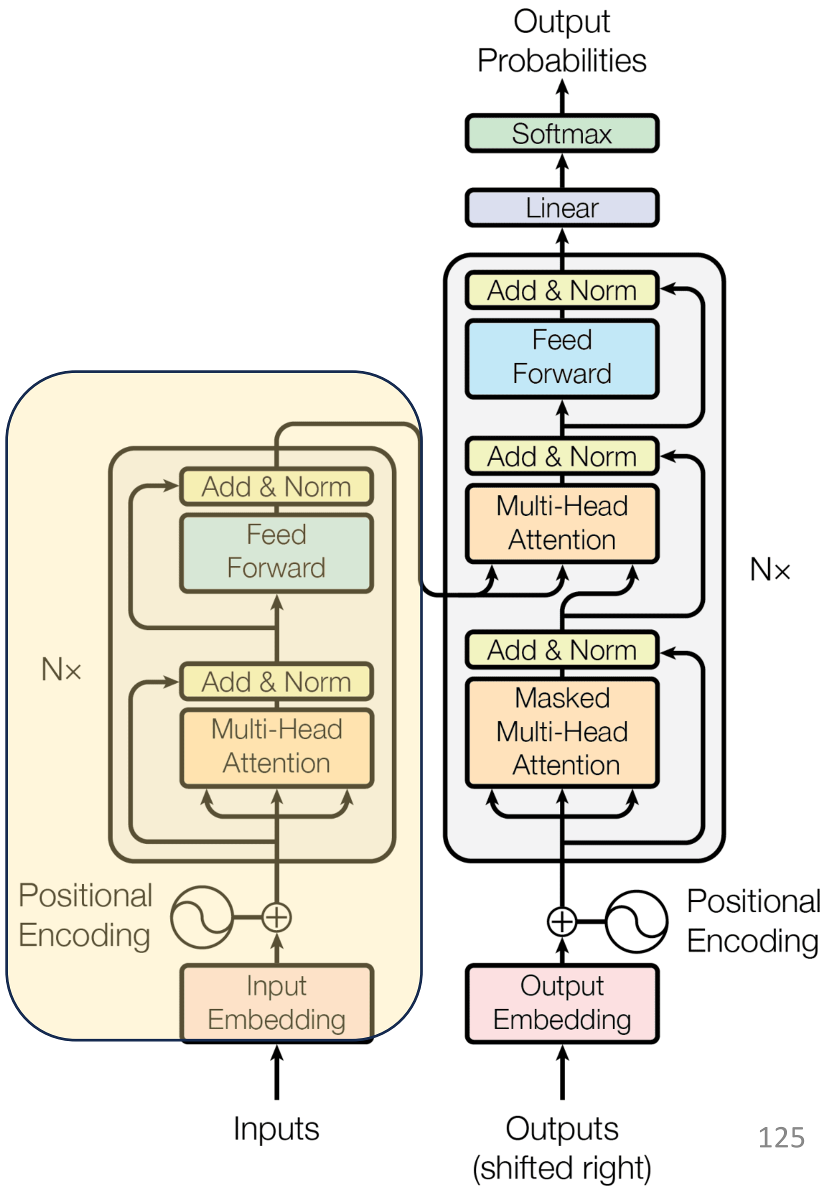
- Simply add a task-specific module after the last encoder layer to map it to the desired dimension.
  - *Classification Tasks:*
    - *Add a feed-forward layer on top of the encoder output for the [CLS] token*
  - *Question Answering Tasks:*
    - *Train two extra vectors to mark the beginning and end of answer from paragraph*
  - …



BERT - Bidirectional Encoder Representations

BERT Evaluation:

- General Language Understanding Evaluation (GLUE)
  - Sentence pair tasks
  - Single sentence classification
- Standford Question Answering Dataset (SQuAD)



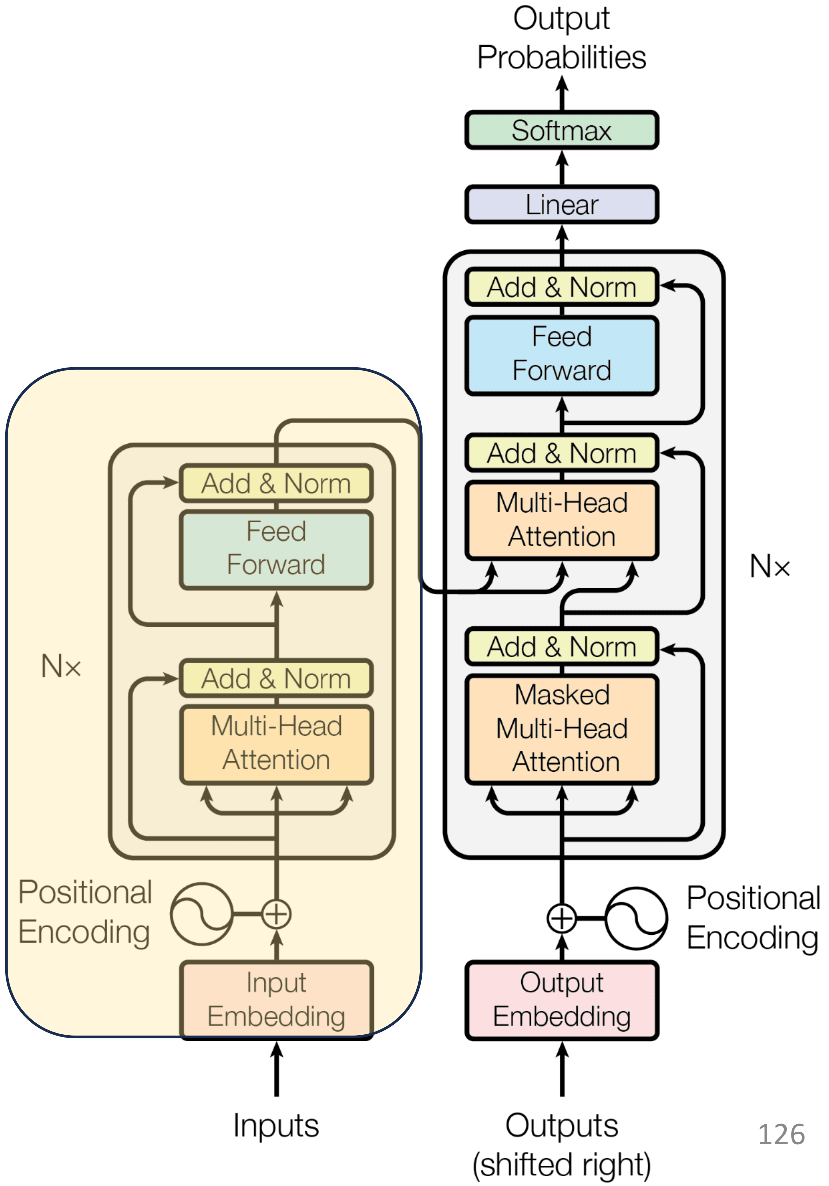
BERT - Bidirectional Encoder Representations

BERT Evaluation:

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Average
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

System	Dev EM	Dev F1	Test EM	Test F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

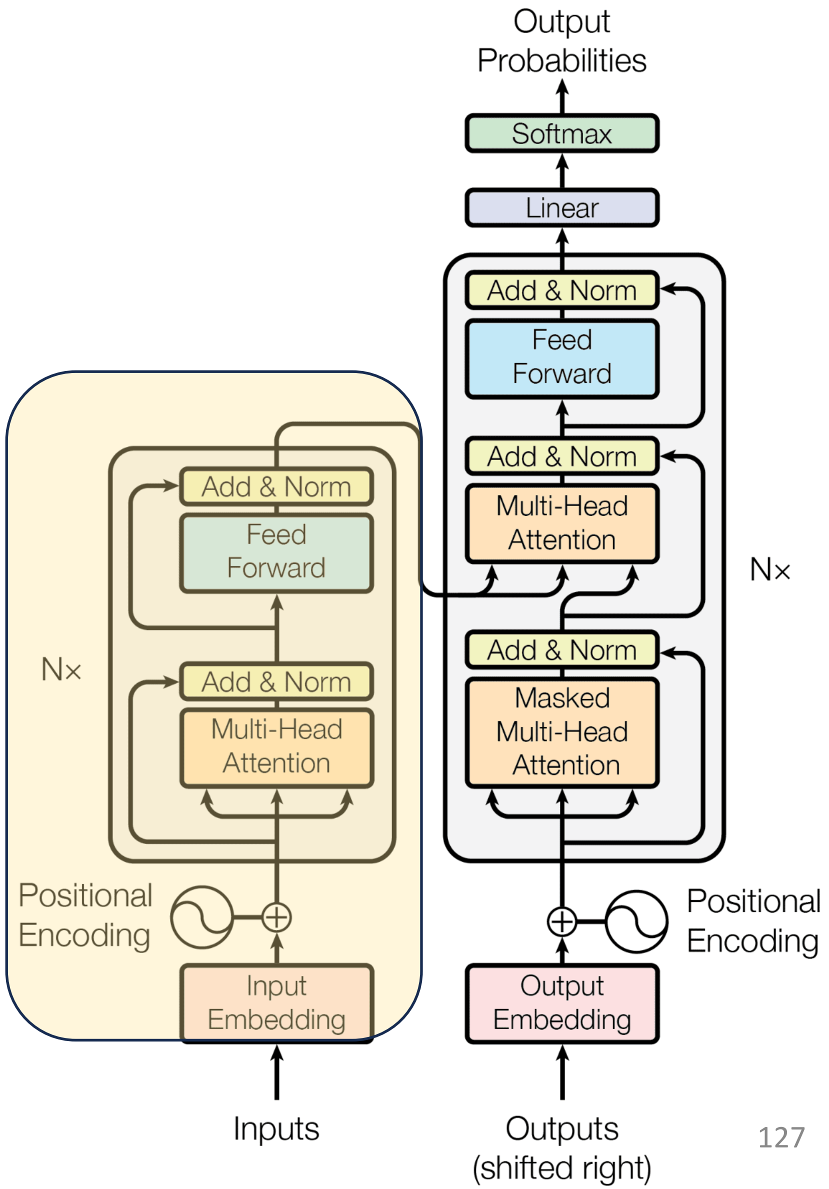
Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.



BERT - Bidirectional Encoder Representations

What is our takeaway from BERT?

- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.

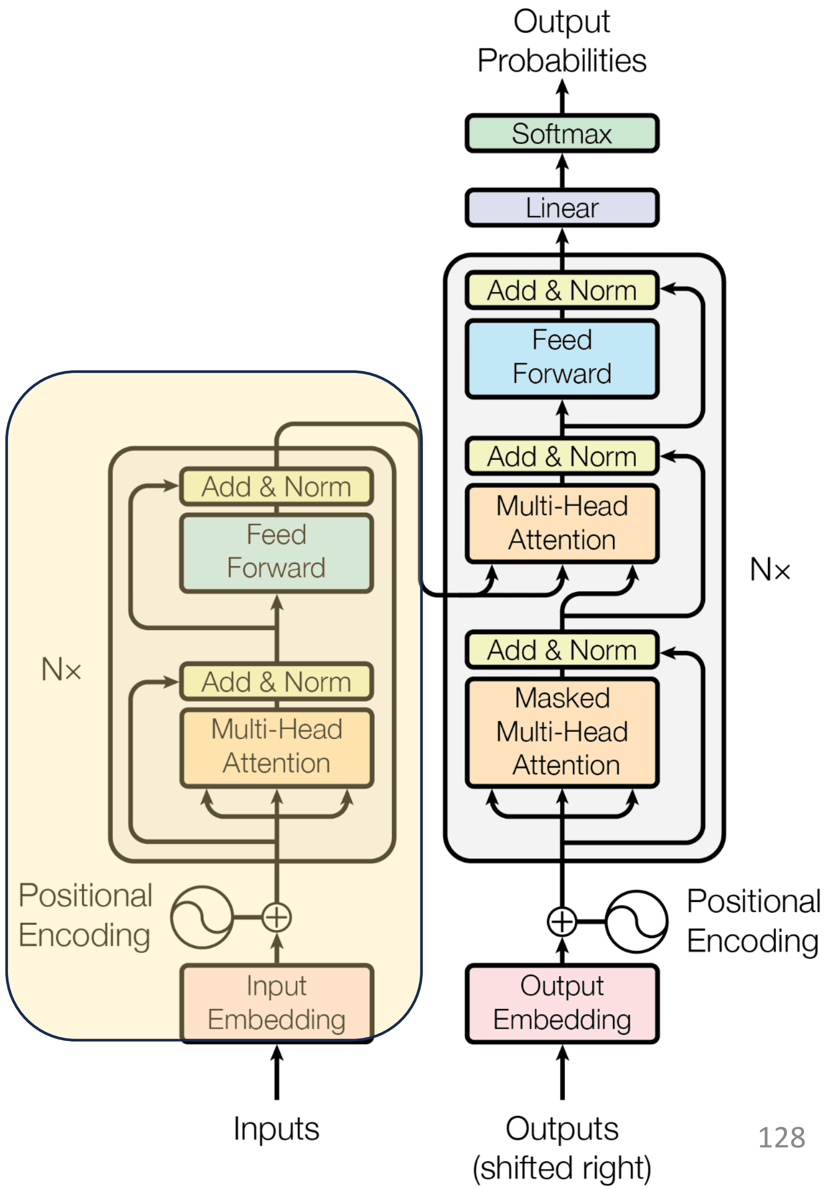


BERT - Bidirectional Encoder Representations

What is our takeaway from BERT?

- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.

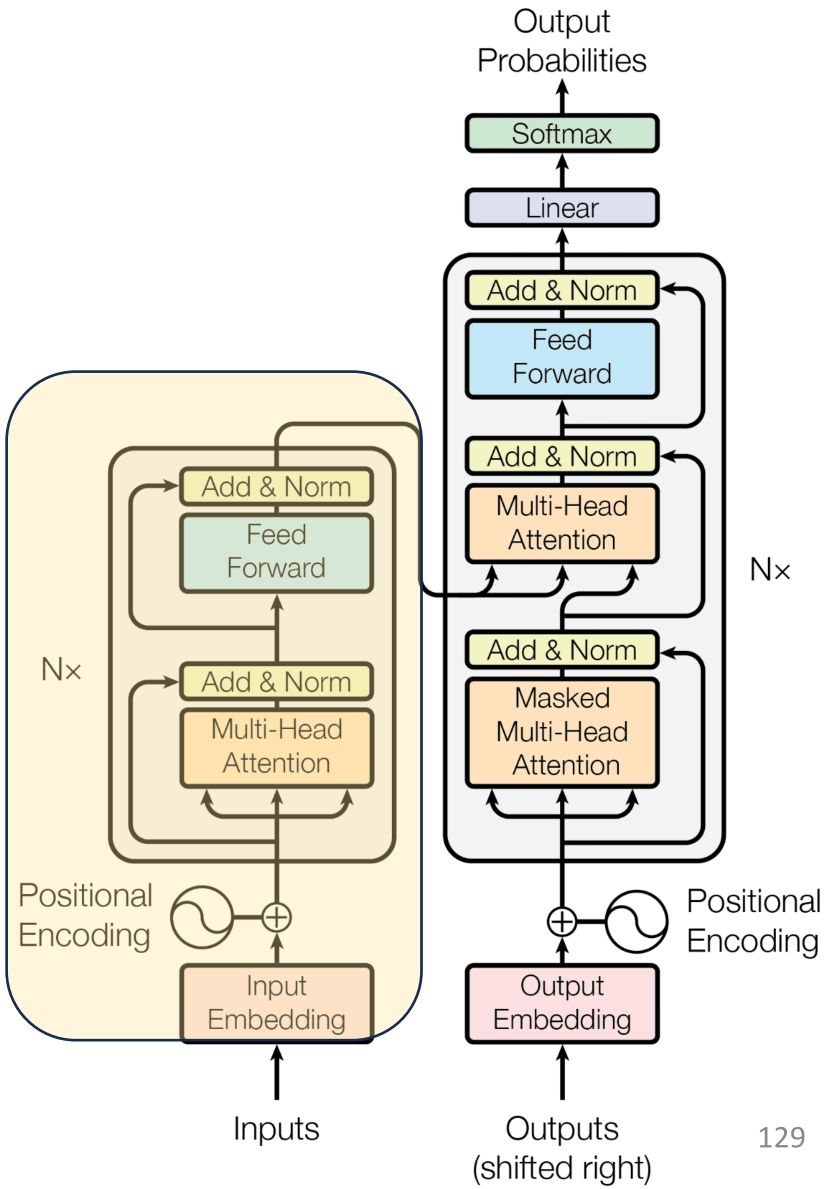
- **Different NLP tasks seem to be highly transferable with each other…**
  - As long as we have effective representations, that seems to form a general model which can serve as the backbone for many specialized models.



BERT - Bidirectional Encoder Representations

What is our takeaway from BERT?

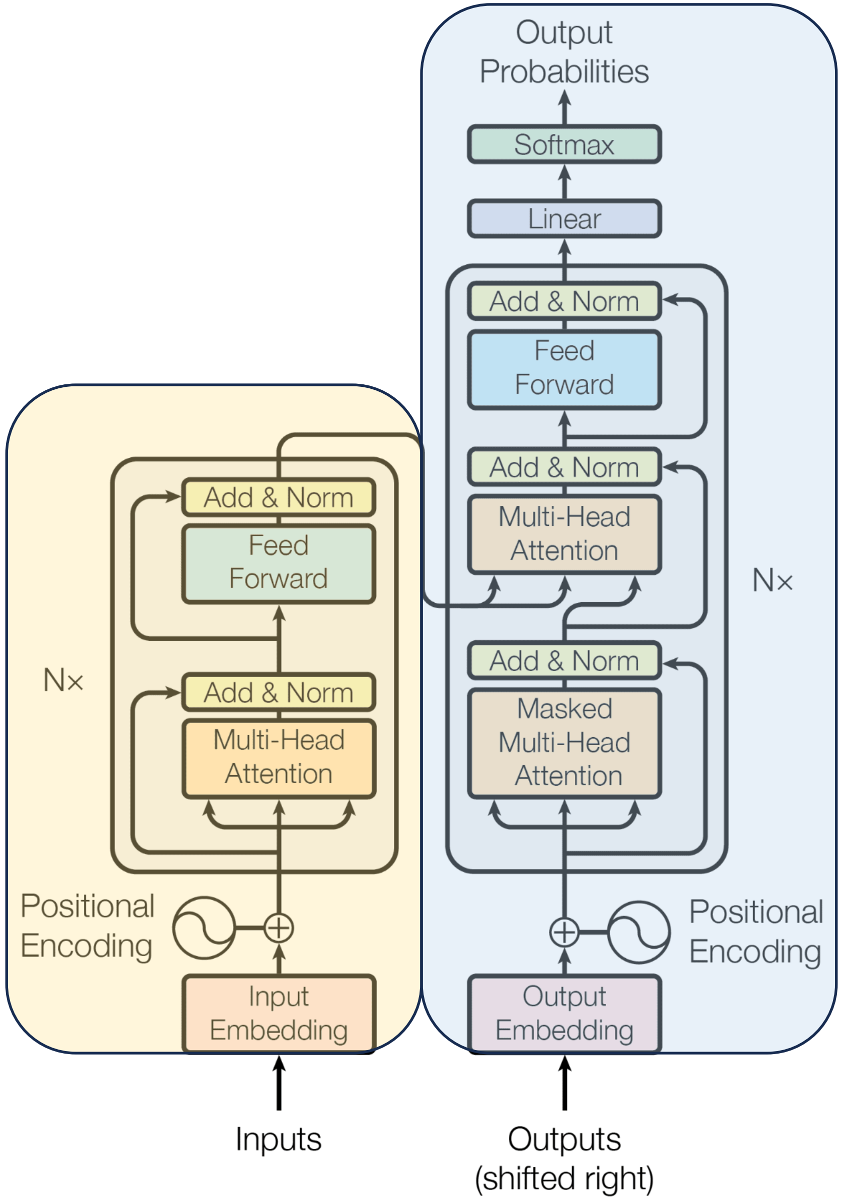
- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.
- **Different NLP tasks seem to be highly transferable with each other…**
  - As long as we have effective representations, that seems to form a general model which can serve as the backbone for many specialized models.
- **And scaling works!!!**
  - 340M is considered large in 2018



2018 – The Inception of the LLM Era

BERT
Oct 2018

Representation

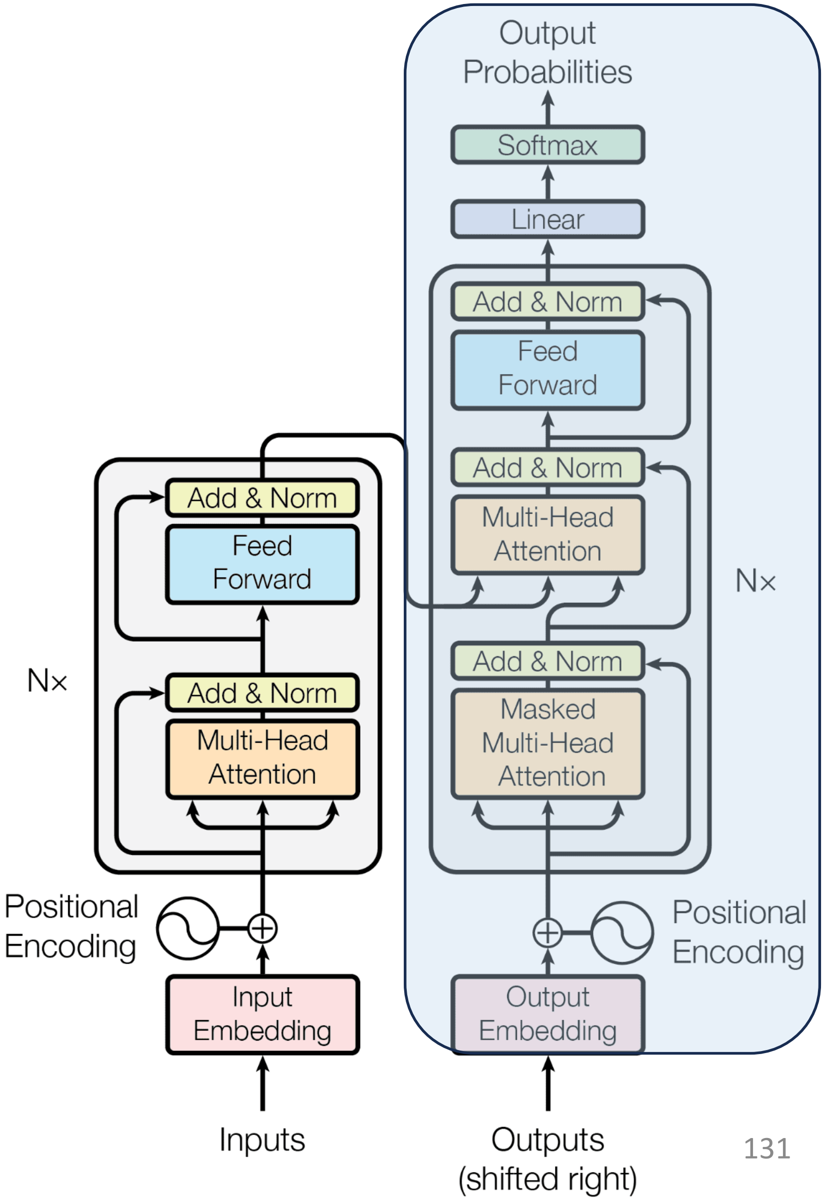


GPT
Jun 2018

Generation

GPT – Generative Pretrained Transformer

- Similarly motivated as BERT, though differently designed
    - Can we leverage large amounts of unlabeled data to pretrain an LM that understands general patterns?



GPT – Generative Pretrained Transformer

GPT Pre-Training Corpus:

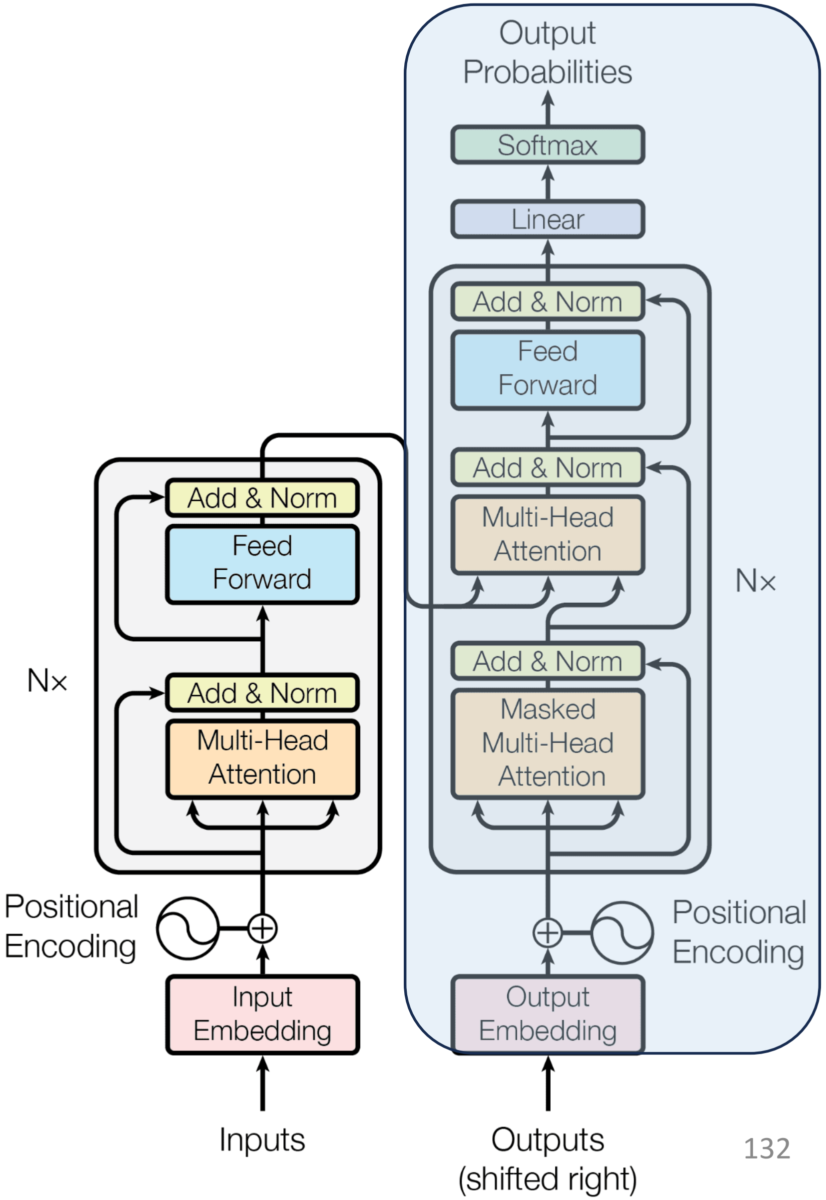
- Similarly, BooksCorpus and English Wikipedia

GPT Pre-Training Tasks:

- Predict the next token, given the previous tokens
    - More learning signals than MLM

GPT Pre-Training Results:

- GPT – 117M Params
    - Similarly competitive on GLUE and SQuAD



GPT – Generative Pretrained Transformer

GPT Fine-Tuning:

- Prompt-format task-specific text as a continuous stream for the model to fit

Summarization

Summarize this article:

The summary is:

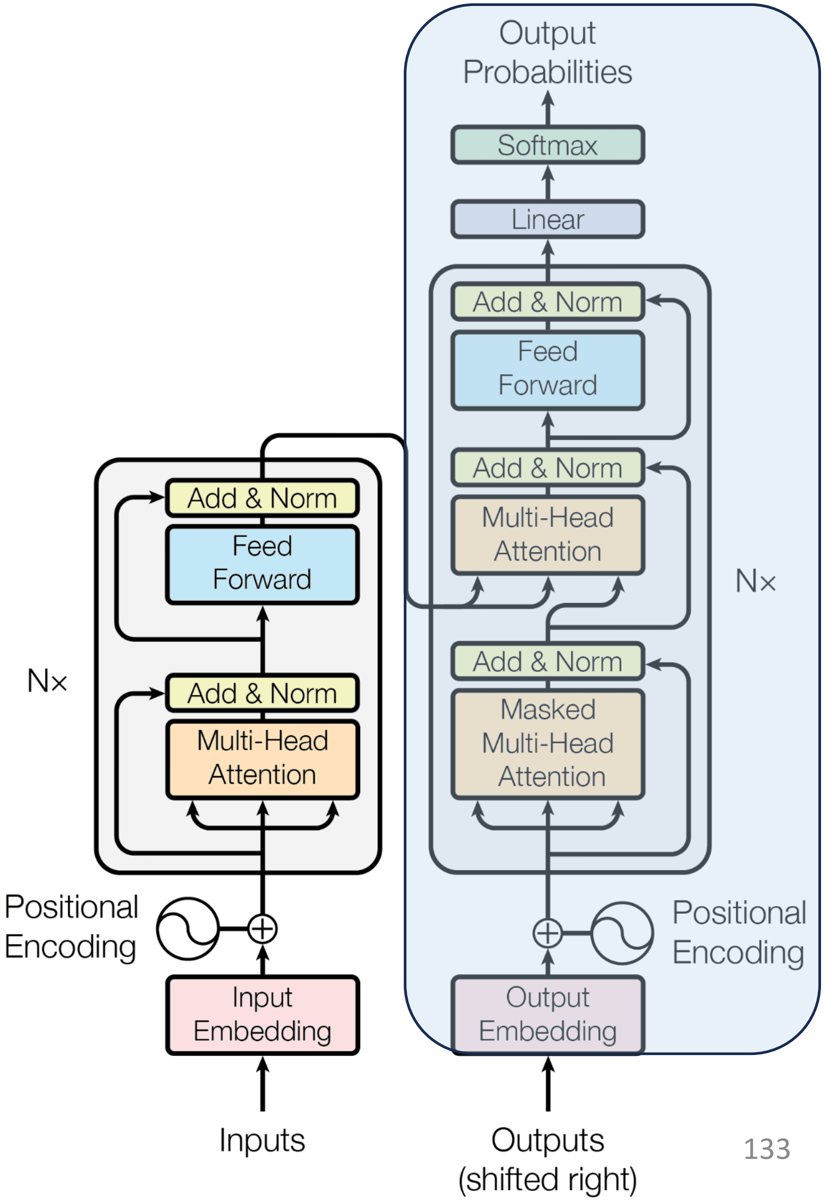
QA

Answer the question based on the context.

Context:

Question:

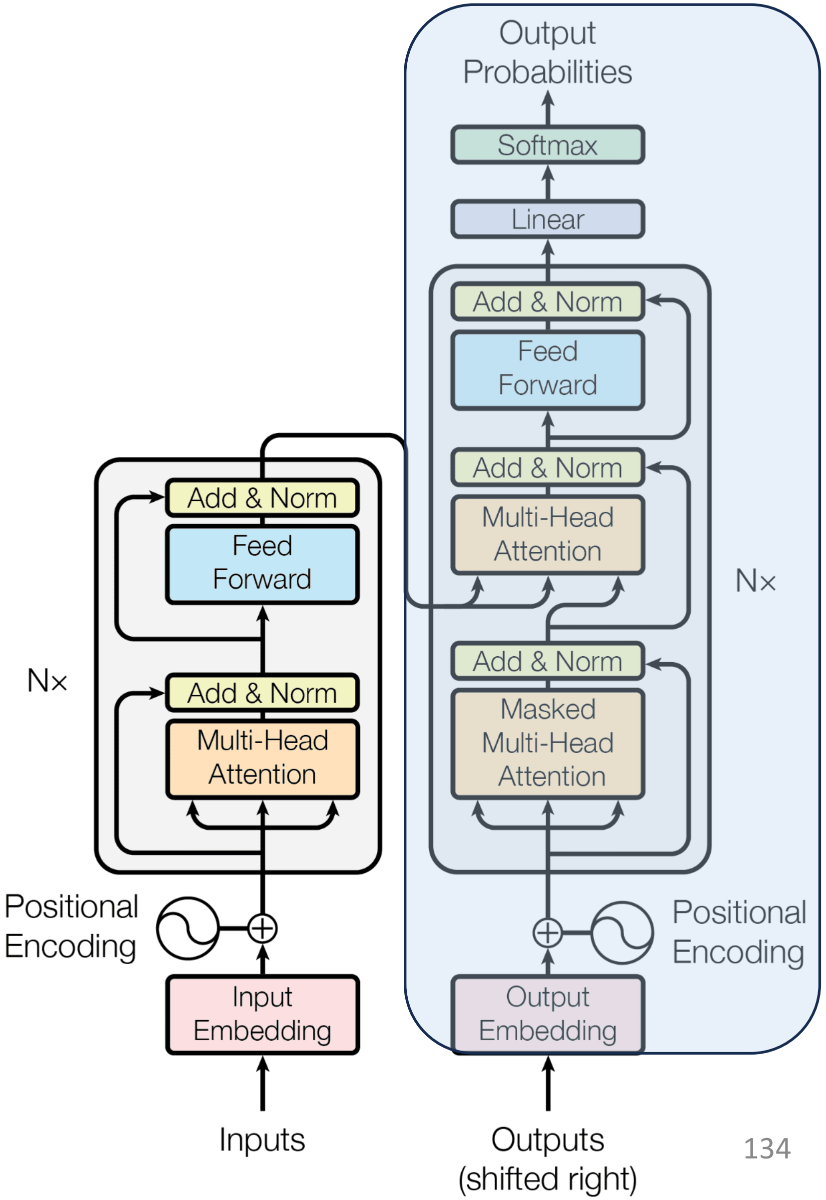
Answer:



GPT – Generative Pretrained Transformer

What is our takeaway from GPT?

- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.

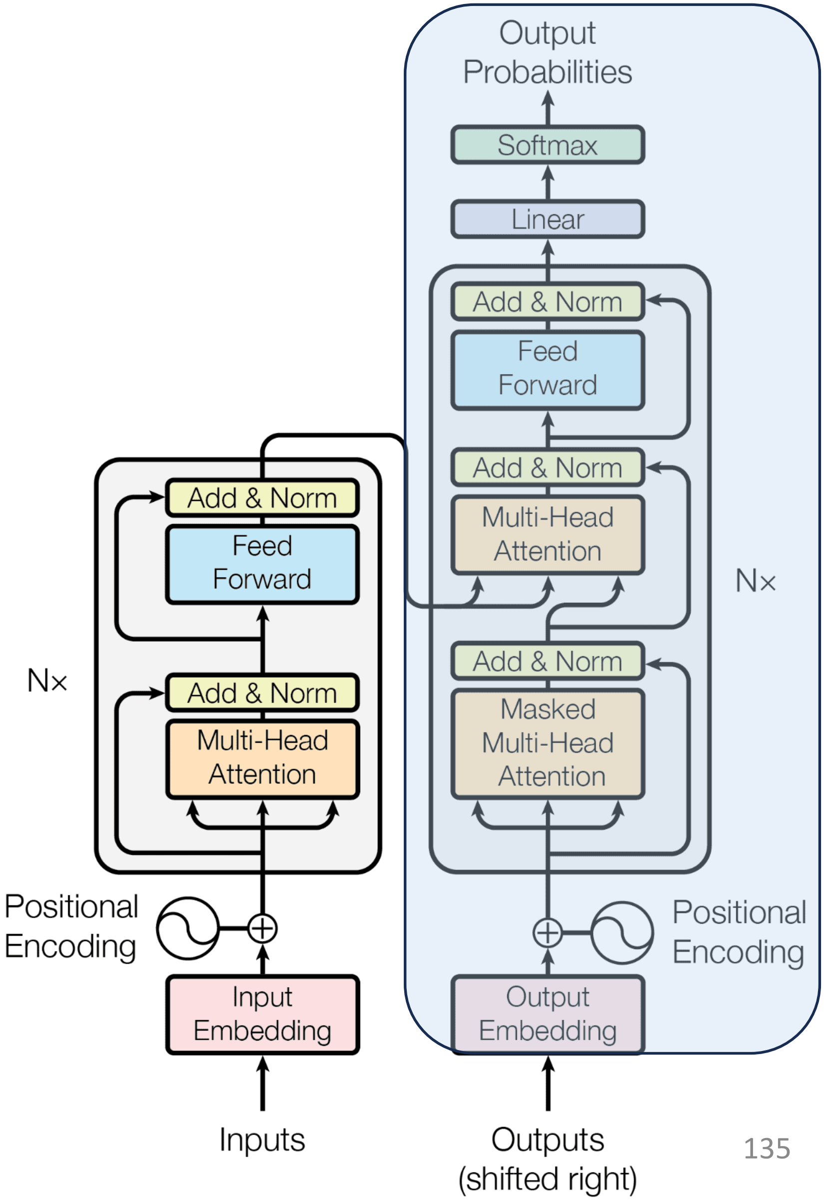


GPT – Generative Pretrained Transformer

What is our takeaway from GPT?

- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.

- **Language Model as a Knowledge Base**
  - Specifically, a generatively pretrained model seems to have a decent zero-shot performance on a range of NLP tasks.



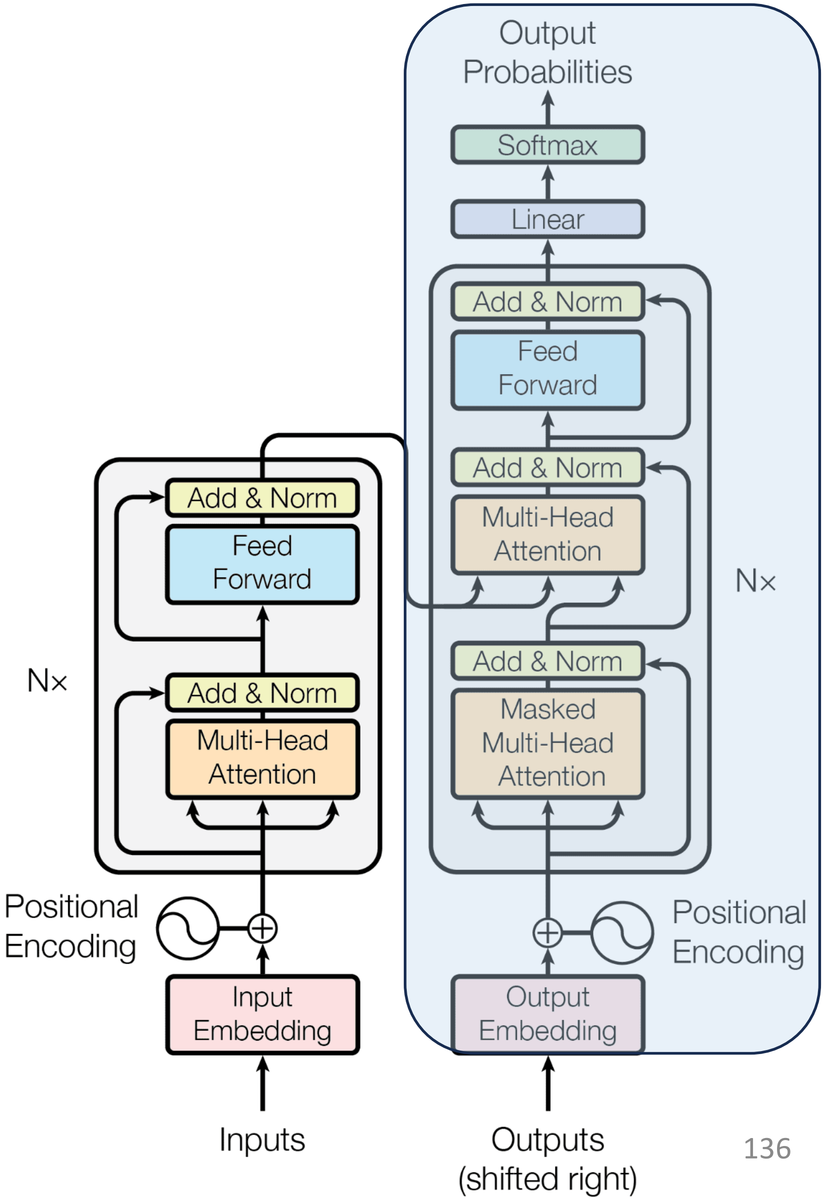
GPT – Generative Pretrained Transformer

What is our takeaway from GPT?

- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.

- **Language Model as a Knowledge Base**
  - Specifically, a generatively pretrained model seems to have a decent zero-shot performance on a range of NLP tasks.

- **And scaling works!!!**



Poll
Piazza @1291

The original GPT's parameter count is closest to…

A. 117
B. 117K
C. 117M
D. 117B

Poll
Piazza @1291

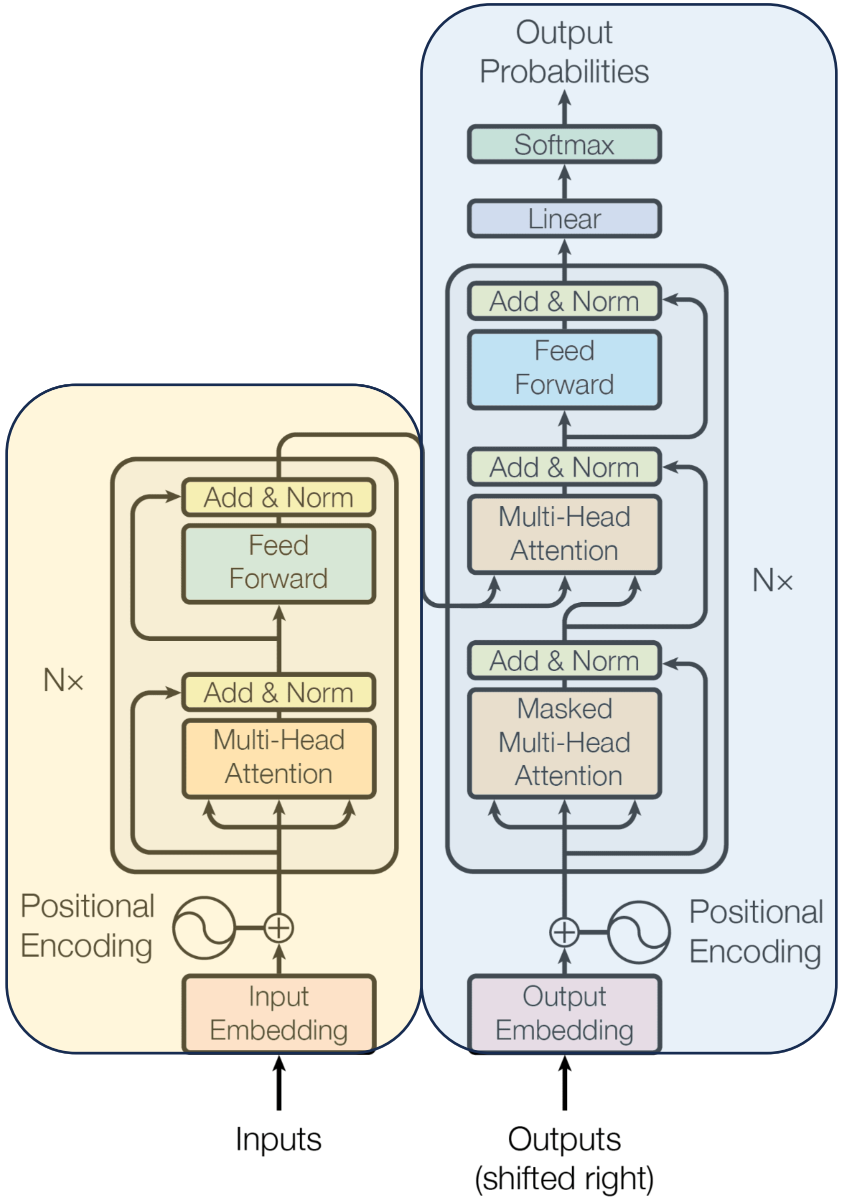
The original GPT's parameter count is closest to…

A. 117
B. 117K
C. 117M
D. 117B

The LLM Era – Paradigm Shift in Machine Learning

BERT
Oct 2018

Representation



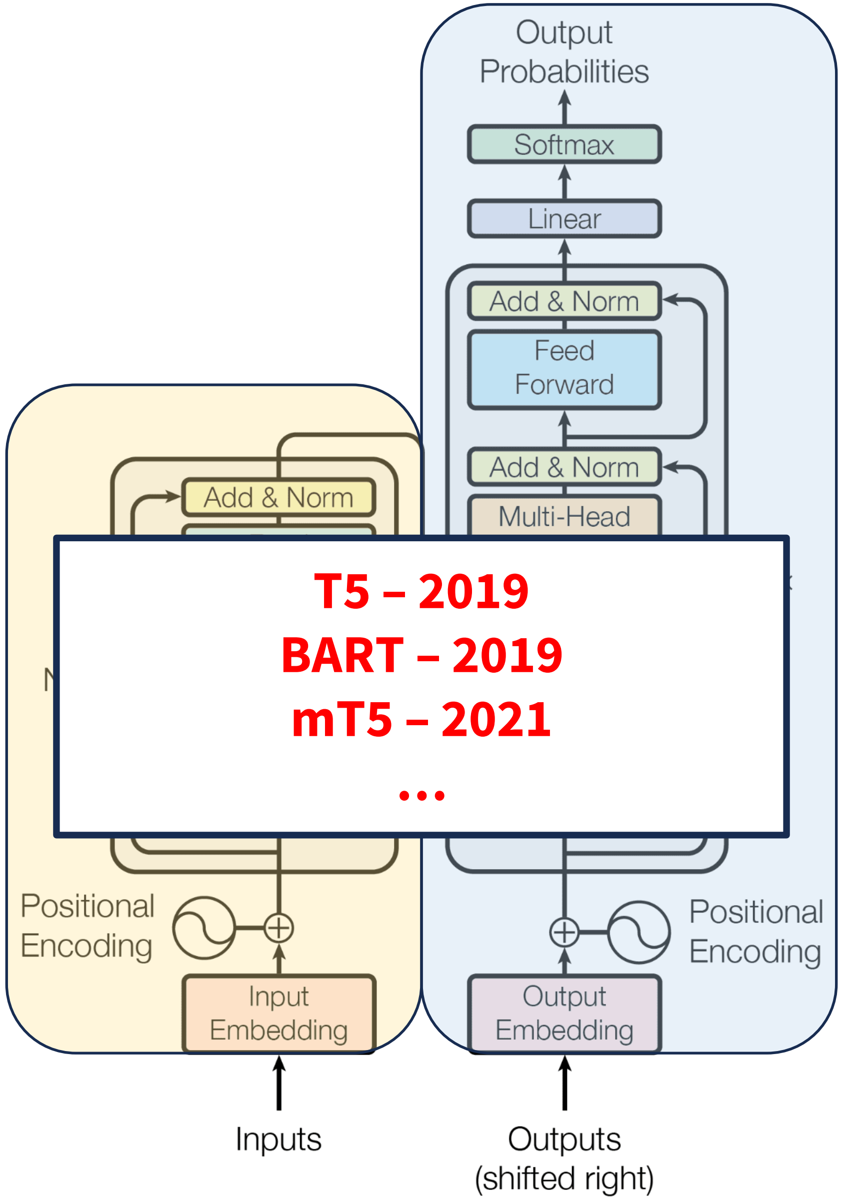
GPT
Jun 2018

Generation

The LLM Era – Paradigm Shift in Machine Learning

BERT – 2018
DistilBERT – 2019
RoBERTa – 2019
ALBERT – 2019
ELECTRA – 2020
DeBERTa – 2020
…

Representation



T5 – 2019
BART – 2019
mT5 – 2021
…

GPT – 2018
GPT-2 – 2019
GPT-3 – 2020
GPT-Neo – 2021
GPT-3.5 (ChatGPT) – 2022
LLaMA – 2023
GPT-4 – 2023
…

Generation

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?

Since LLMs

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?

Since LLMs

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs	**Since LLMs**

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs | **Since LLMs**

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?
- **Overfitting vs Generalization**
  - How do we balance complexity and capacity to prevent overfitting while maintaining good performance?

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?
- **Overfitting vs Generalization**
  - How do we balance complexity and capacity to prevent overfitting while maintaining good performance?

Since LLMs

- **Pre-training and Fine-tuning**
  - How do we leverage large scales of unlabeled data out there previously under-leveraged?

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?
- **Overfitting vs Generalization**
  - How do we balance complexity and capacity to prevent overfitting while maintaining good performance?

Since LLMs

- **Pre-training and Fine-tuning**
  - How do we leverage large scales of unlabeled data out there previously under-leveraged?
- **Zero-shot and Few-shot learning**
  - How can we make models perform on tasks they are *not* trained on?

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?
- **Overfitting vs Generalization**
  - How do we balance complexity and capacity to prevent overfitting while maintaining good performance?

Since LLMs

- **Pre-training and Fine-tuning**
  - How do we leverage large scales of unlabeled data out there previously under-leveraged?
- **Zero-shot and Few-shot learning**
  - How can we make models perform on tasks they are *not* trained on?
- **Prompting**
  - How do we make models understand their task simply by describing it in natural language?

The LLM Era – Paradigm Shift in Machine Learning

From both BERT and GPT, we learn that…

- Transformers seem to provide a new class of generalist models that are capable of capturing knowledge which is more fundamental than task-specific abilities.

Before LLMs

- **Feature Engineering**
  - How do we design or select the best features for a task?
- **Model Selection**
  - Which model is best for which type of task?
- **Transfer Learning**
  - Given scarce labeled data, how do we transfer knowledge from other domains?
- **Overfitting vs Generalization**
  - How do we balance complexity and capacity to prevent overfitting while maintaining good performance?

Since LLMs

- **Pre-training and Fine-tuning**
  - How do we leverage large scales of unlabeled data out there previously under-leveraged?
- **Zero-shot and Few-shot learning**
  - How can we make models perform on tasks they are *not* trained on?
- **Prompting**
  - How do we make models understand their task simply by describing it in natural language?
- **Interpretability and Explainability**
  - How can we *understand* the inner workings of our own models?

The LLM Era – Paradigm Shift in Machine Learning

- **What has caused this paradigm shift?**

The LLM Era – Paradigm Shift in Machine Learning

- **What has caused this paradigm shift?**

  - **Problem in recurrent networks**
    - Information is effectively lost during encoding of long sequences
    - Sequential nature disables parallel training and favors late timestep inputs

The LLM Era – Paradigm Shift in Machine Learning

- **What has caused this paradigm shift?**

  - **Problem in recurrent networks**
    - Information is effectively lost during encoding of long sequences
    - Sequential nature disables parallel training and favors late timestep inputs

  - **Solution: Attention mechanism**
    - Handling long-range dependencies
    - Parallel training
    - Dynamic attention weights based on inputs

The LLM Era – Paradigm Shift in Machine Learning

- **Attention and Transformer – is this the end?**

The LLM Era – Paradigm Shift in Machine Learning

- **Attention and Transformer – is this the end?**
  - **Problem in current Transformer-based LLMs??**

Poll
Piazza @1292

What might be a flaw of our current Transformer-based LLMs?

Freeform response

The LLM Era – Paradigm Shift in Machine Learning

- **Attention and Transformer – is this the end?**

  - **Problem in current Transformer-based LLMs??**
    - True understanding the material vs. memorization and pattern-matching
    - Cannot reliably follow rules – factual hallucination e.g. inability in arithmetic

The LLM Era – Paradigm Shift in Machine Learning

- **Attention and Transformer – is this the end?**
  - **Problem in current Transformer-based LLMs??**
    - True understanding the material vs. memorization and pattern-matching
    - Cannot reliably follow rules – factual hallucination e.g. inability in arithmetic
  - **Solution: ???**

Looking Back

It is true that language models are just programmed to predict the next token. But that is not as simple as you might think.

In fact, all animals, including us, are just programmed to survive and reproduce, and yet amazingly complex and beautiful stuff comes from it.

- Sam Altman*

***Paraphrased by IDL TAs**