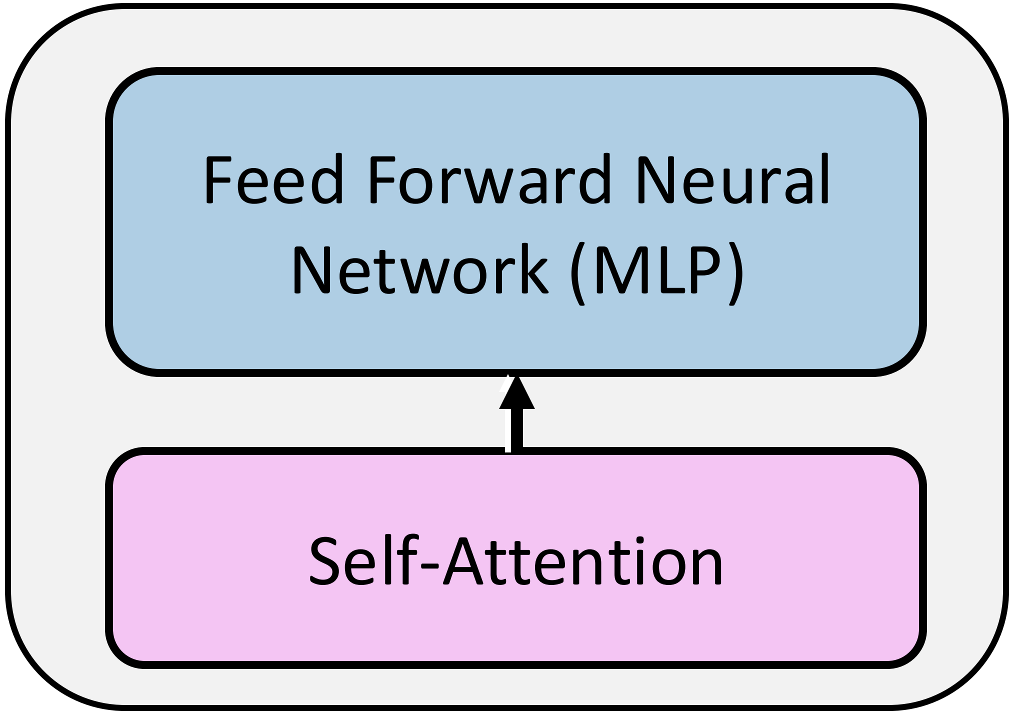
Transformers

Rowel Atienza, Ph.D.
University of the Philippines
github.com/roatienza
2023

Transformer Encoder/Decoder Unit Details

Operations: Linear, Layer Norm, Activation, Tensor Multiply/Add, Softmax



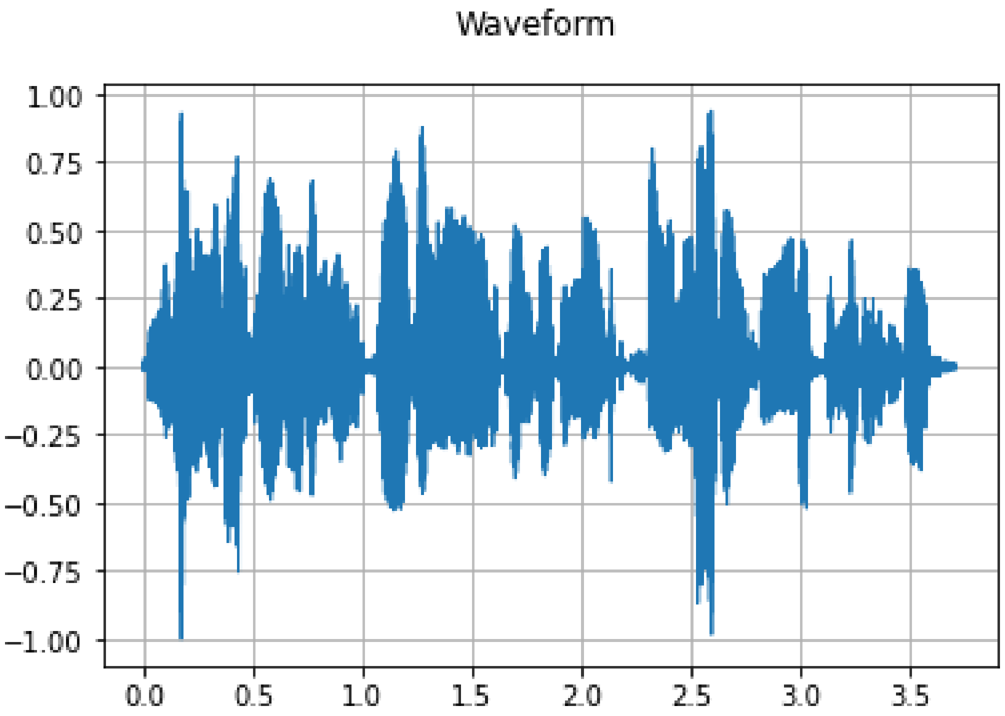
Encoder Block

Types of data that transformers can process

Any

Wikipedia
The Free Encyclopedia

COCO 2017 Keypoint Detection Task



Input Embedding is an $n - dim$ vector

$\boldsymbol{x}_1^T$

.1	-2	.4	-1

cat

$\boldsymbol{x}_2^T$

.3	1	-1	2

opened

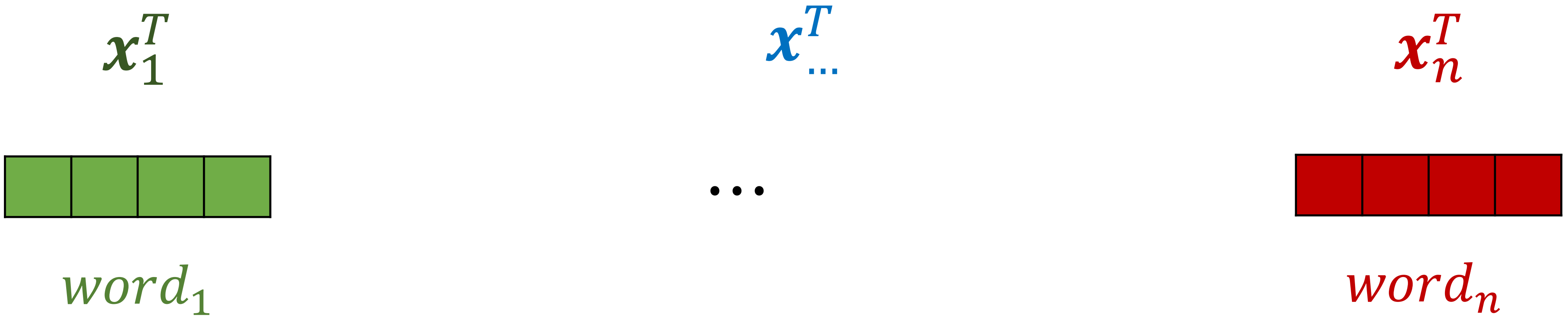
$\boldsymbol{x}_3^T$

.1	.0	1	-1

its

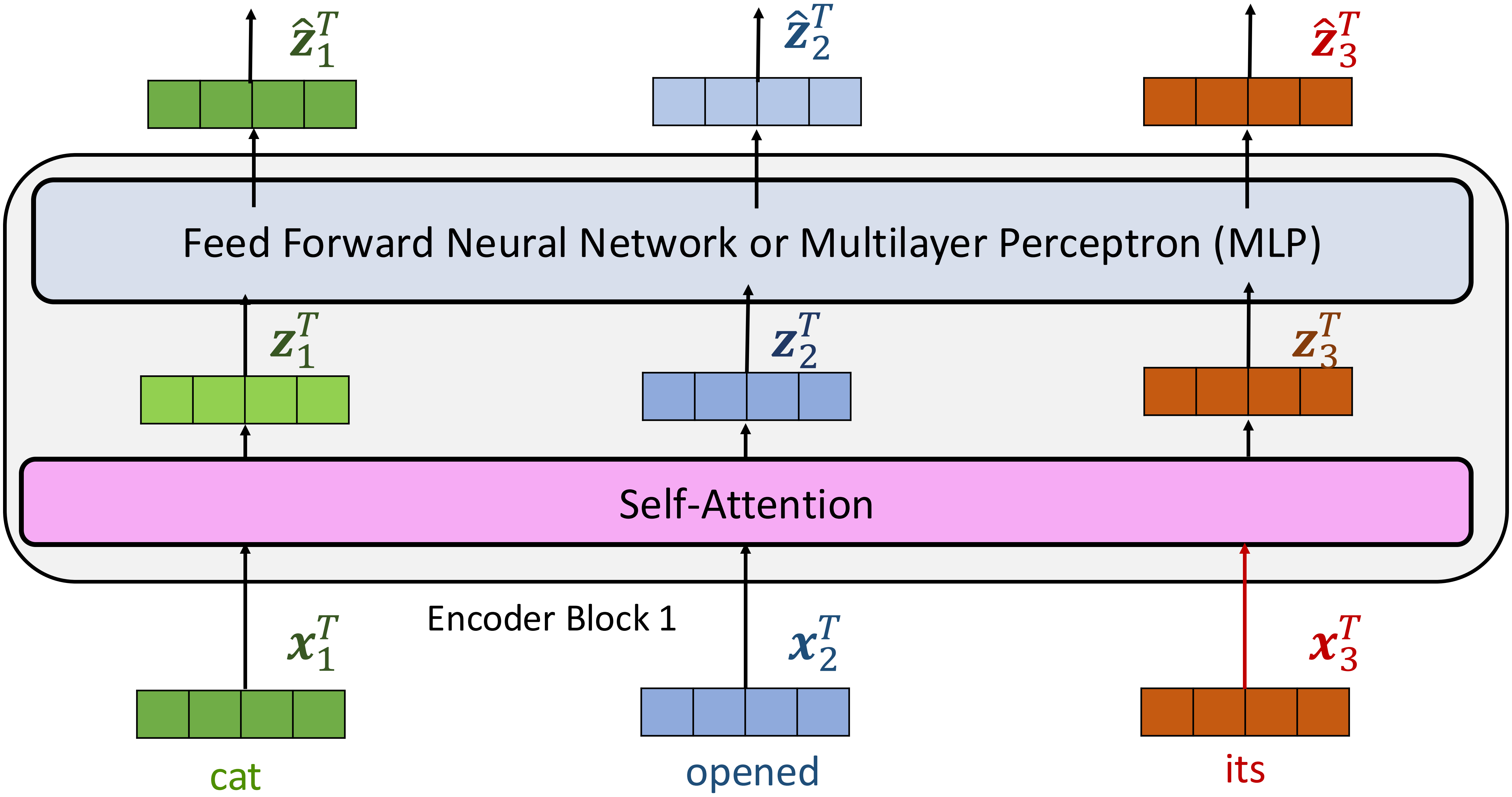
Example: Each word is converted into a 512-dim embedding vector.
In the simple example above, it is 4-dim.

The Length of the Input is $\boldsymbol{n}$

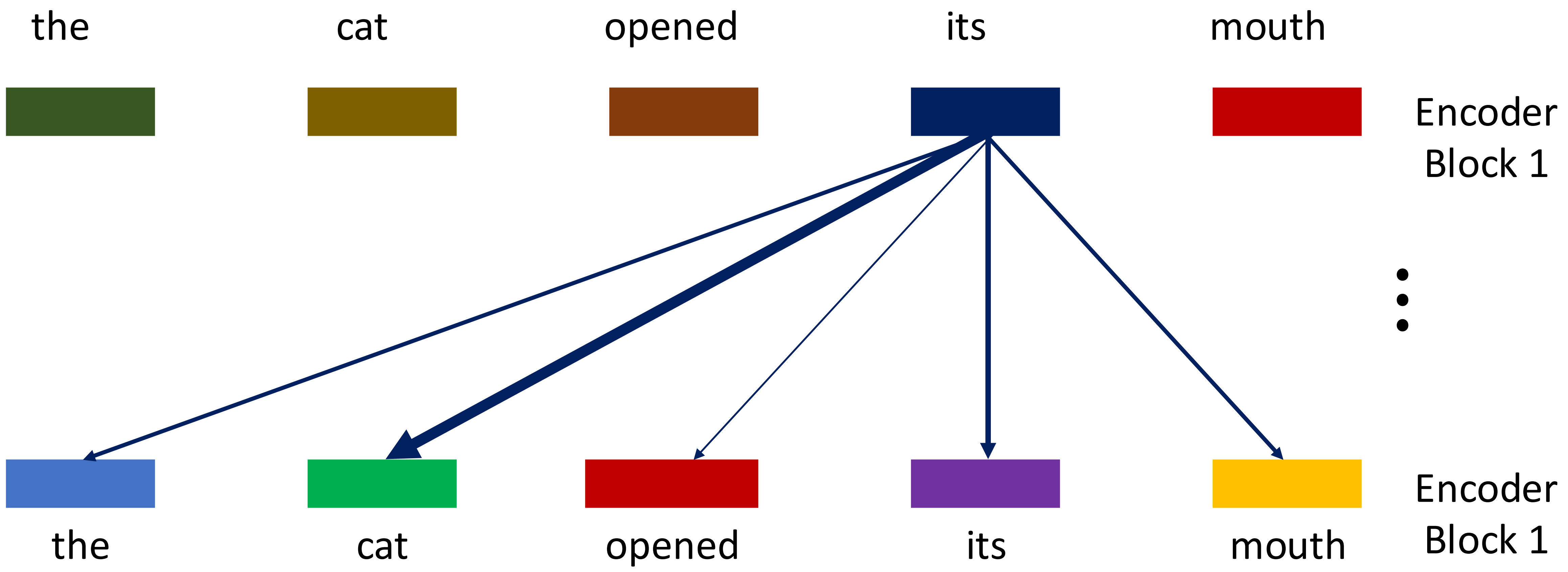


Example: n could be the maximum possible length of a sentence.

Encoder with Latent Variables $\boldsymbol{z}_i$



Attention between 2 words

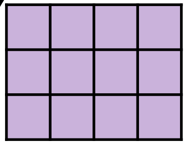
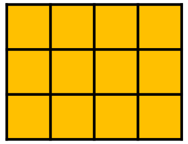
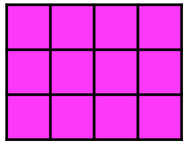


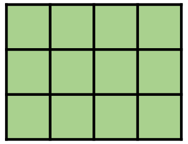
Attention as measured by the width of the arrow

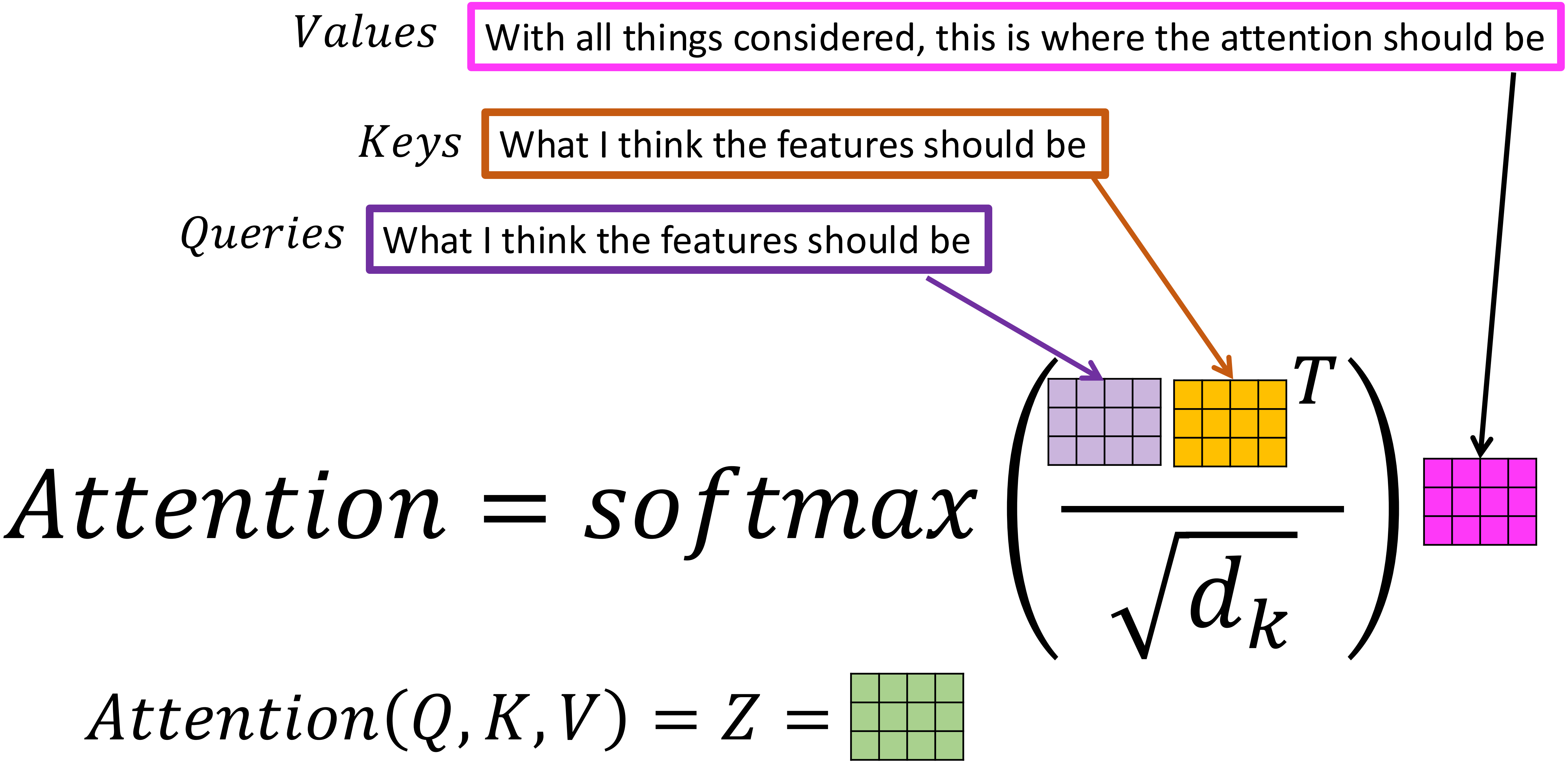
$Query \quad Key \quad Value$

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

d_k is keys/queries dim (e.g. 4)

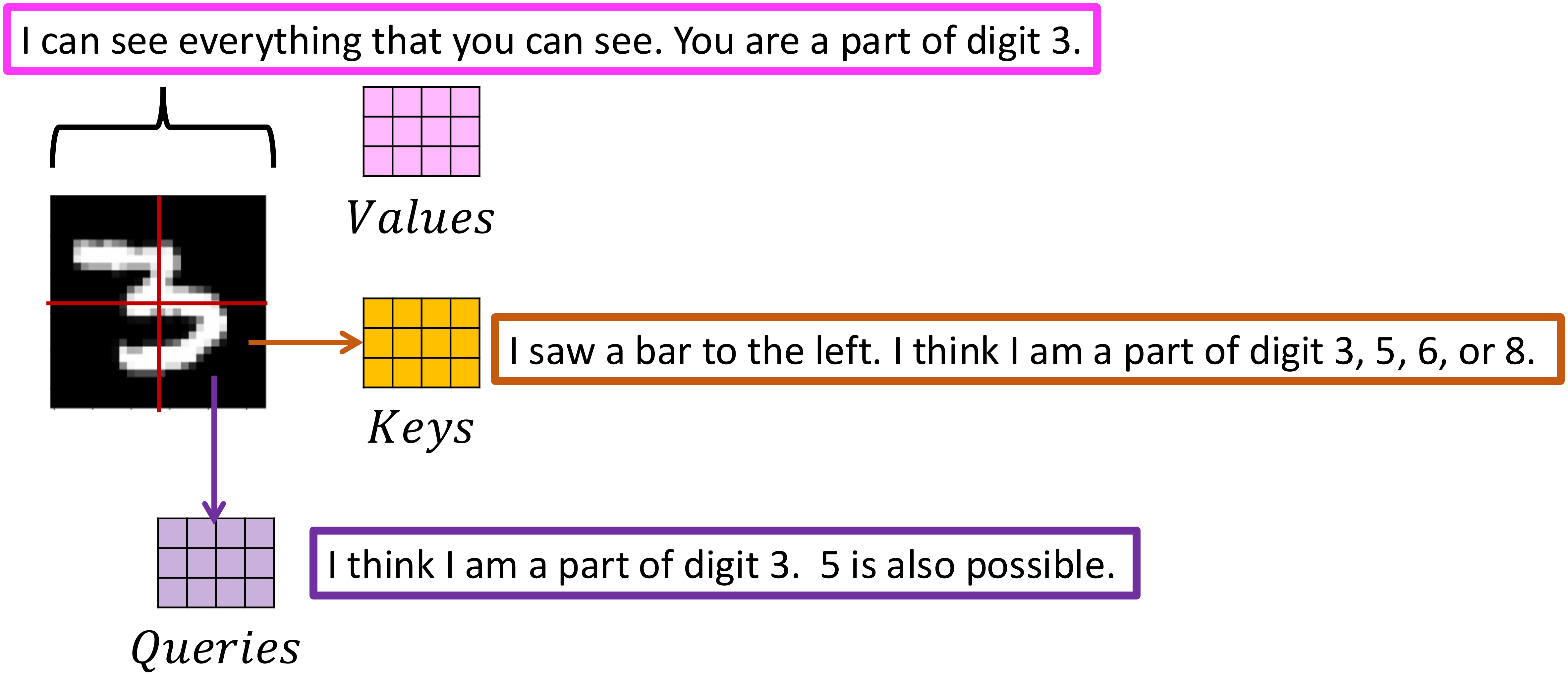
$Attention = softmax\Big($   ^{T} $/ \sqrt{d_k}\Big)$ 

$Attention(Q, K, V) = Z =$ 



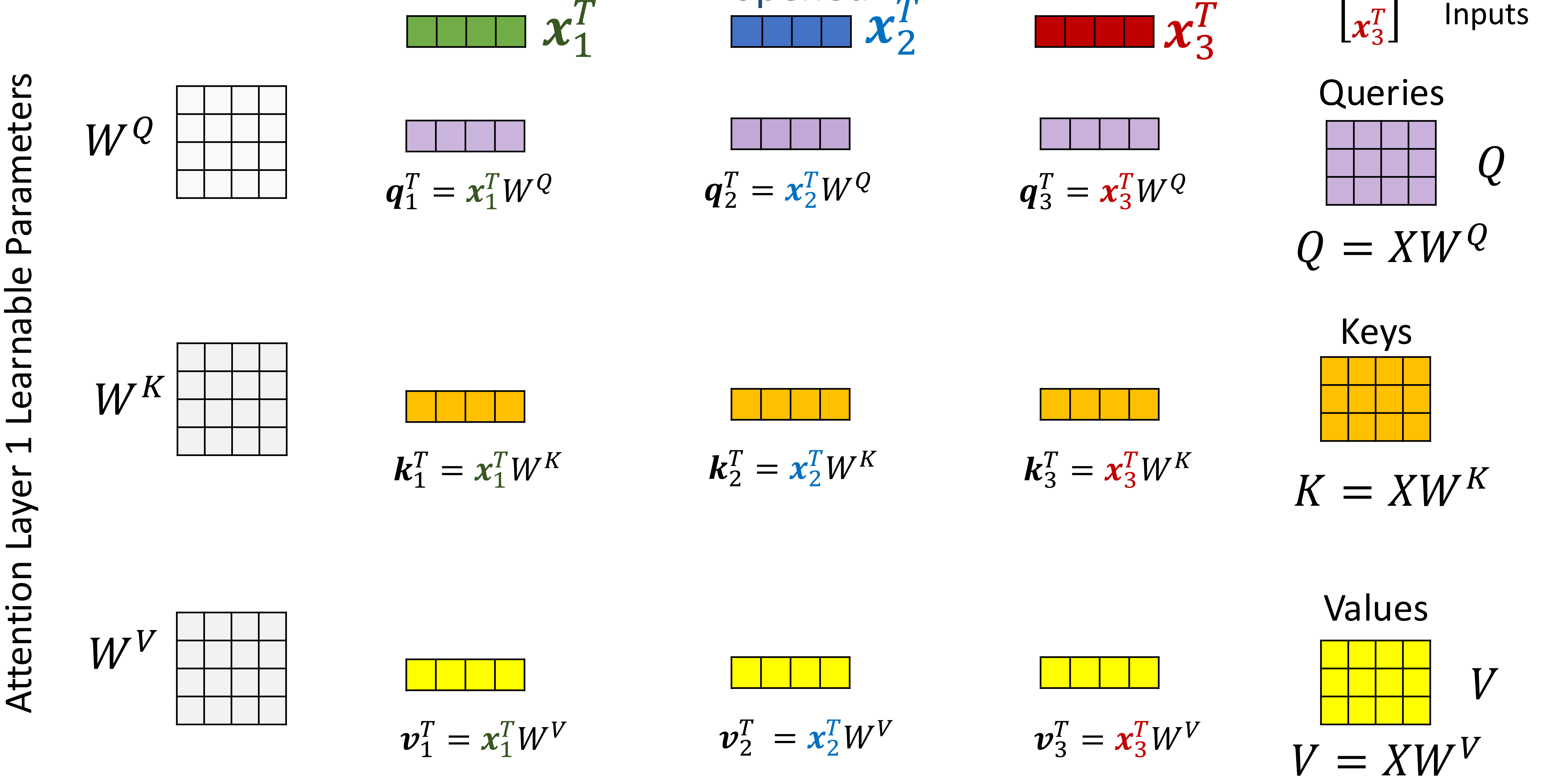
$Values$
With all things considered, this is where the attention should be
$Keys$
What I think the features should be
$Queries$
What I think the features should be
$Attention = softmax\left(\frac{^{T}}{\sqrt{d_k}}\right)$
$Attention(Q, K, V) = Z =$

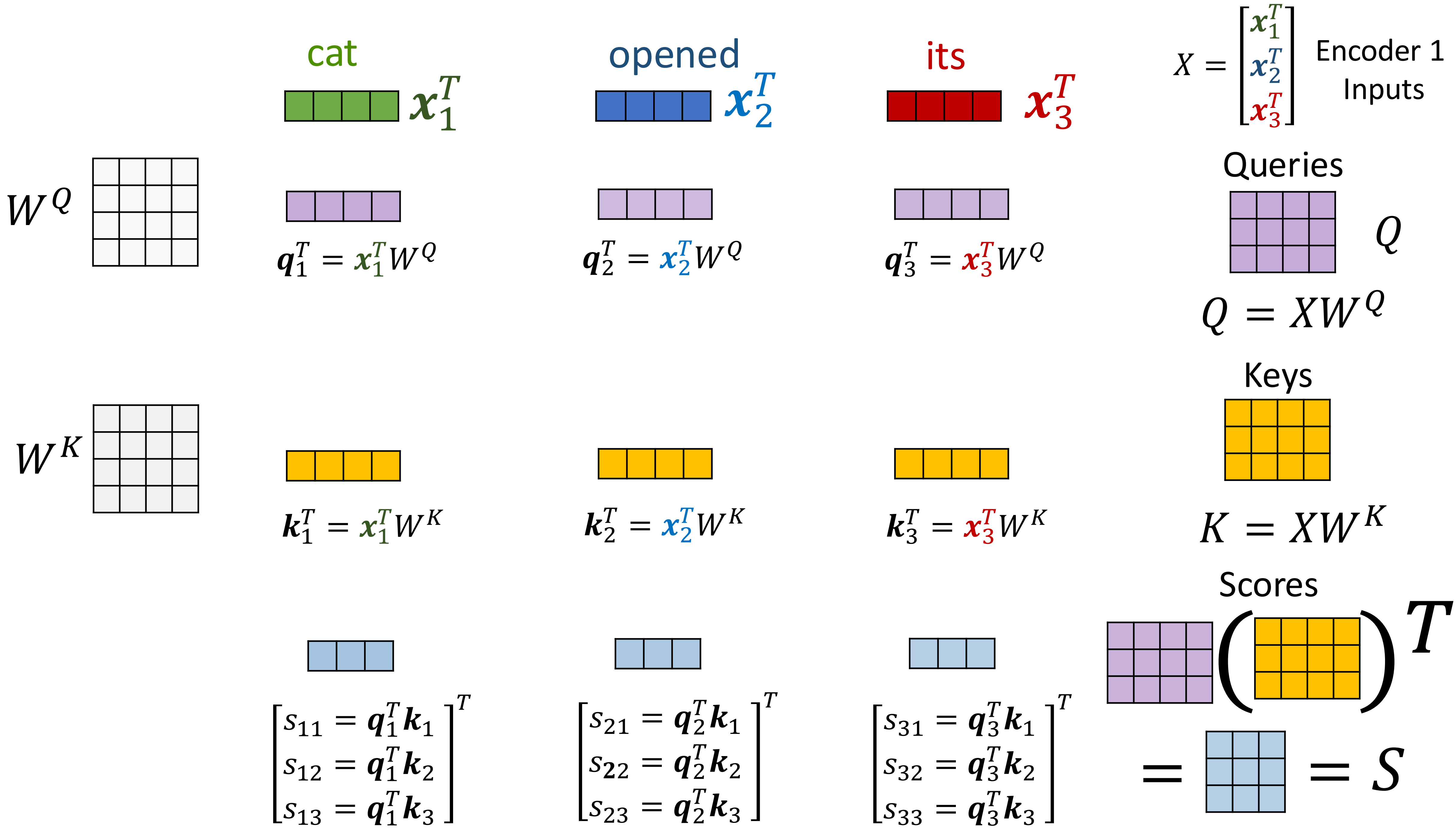
Consider an Attention Layer Examining a Digit



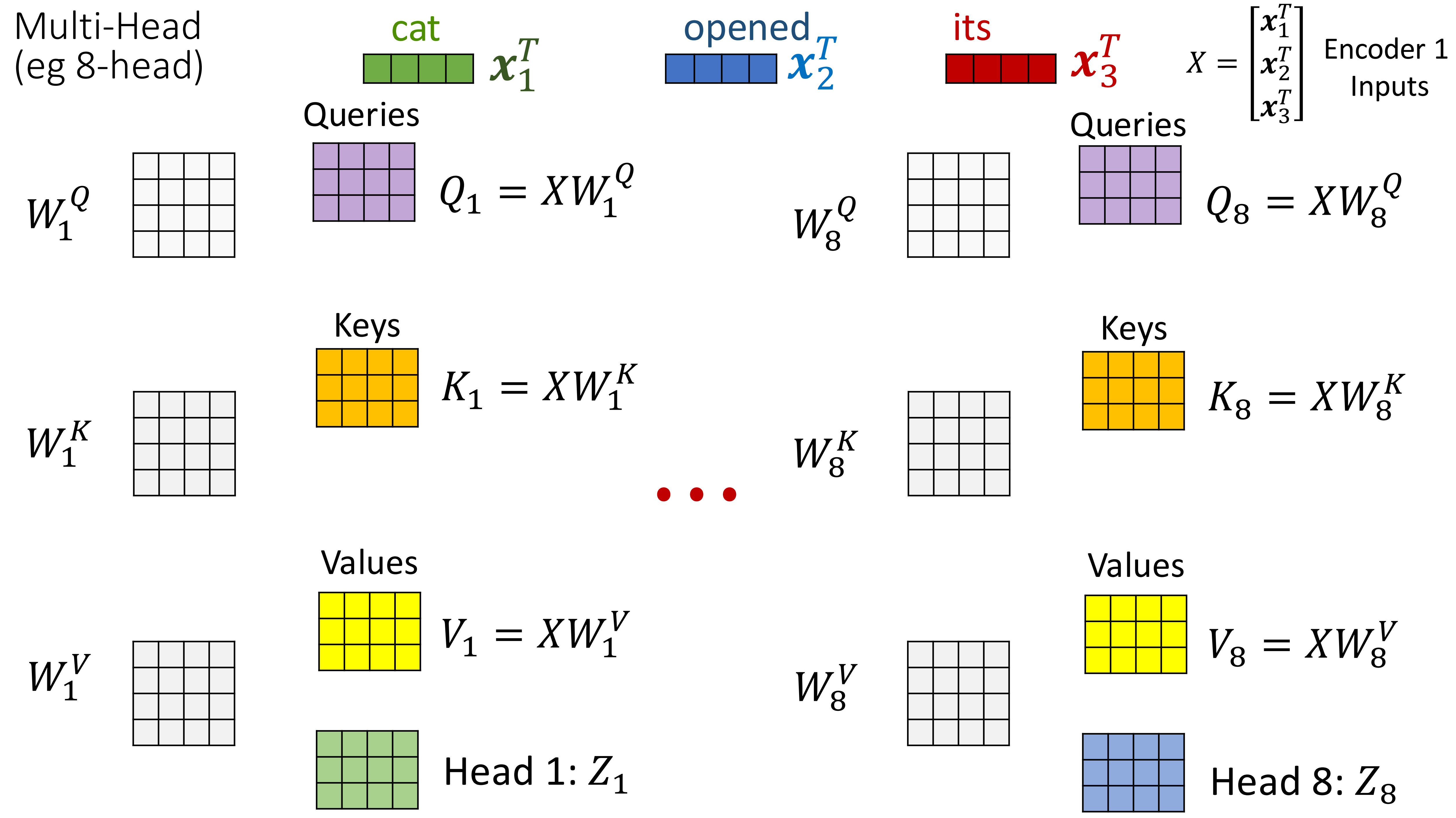
Example: Let us focus on the lower-right patch only

Self-Attention



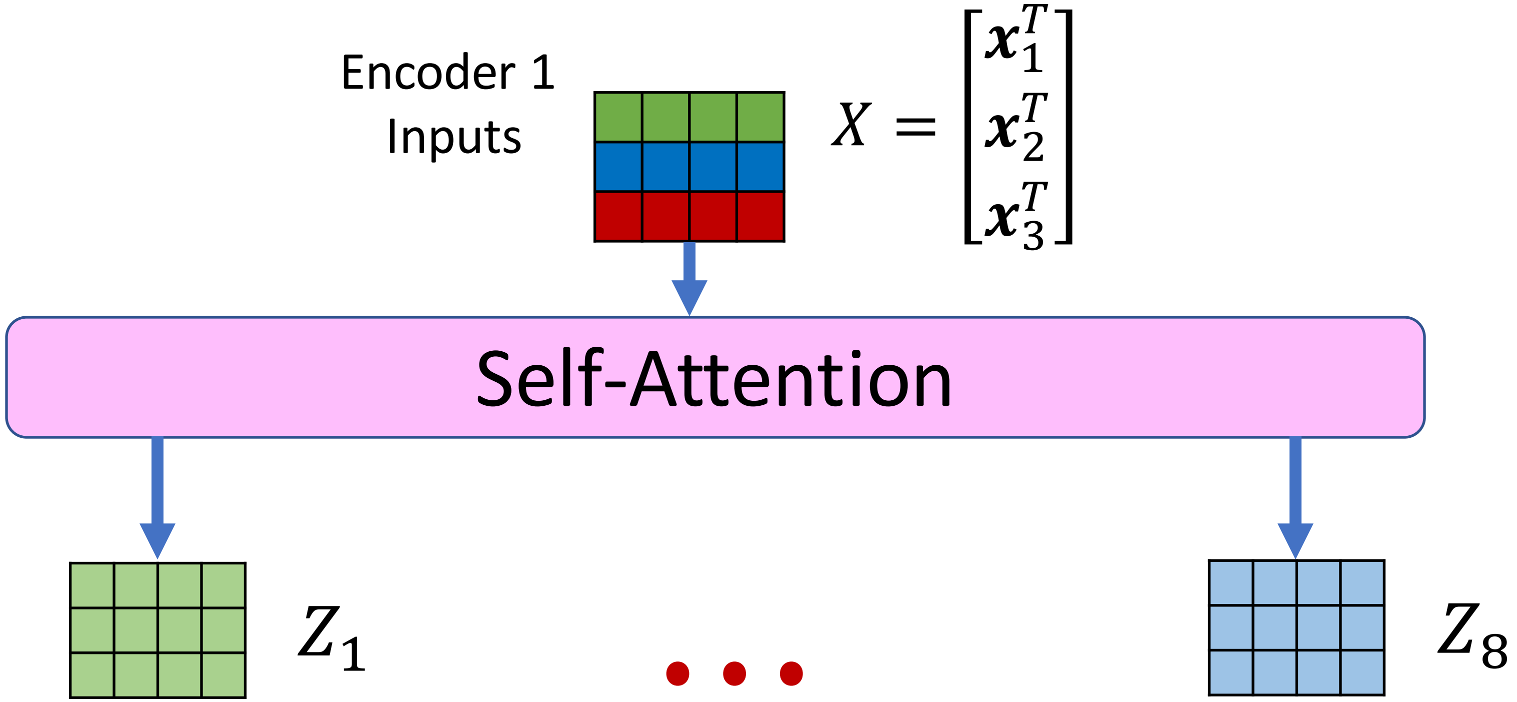


cat
x_1^T
opened
x_2^T
its
x_3^T
$X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix}$
Encoder 1 Inputs
W^Q
$q_1^T = x_1^T W^Q$
$q_2^T = x_2^T W^Q$
$q_3^T = x_3^T W^Q$
Queries
Q
$Q = XW^Q$
W^K
$k_1^T = x_1^T W^K$
$k_2^T = x_2^T W^K$
$k_3^T = x_3^T W^K$
Keys
$K = XW^K$
$\begin{bmatrix} s_{11} = q_1^T k_1 \\ s_{12} = q_1^T k_2 \\ s_{13} = q_1^T k_3 \end{bmatrix}^T$
$\begin{bmatrix} s_{21} = q_2^T k_1 \\ s_{22} = q_2^T k_2 \\ s_{23} = q_2^T k_3 \end{bmatrix}^T$
$\begin{bmatrix} s_{31} = q_3^T k_1 \\ s_{32} = q_3^T k_2 \\ s_{33} = q_3^T k_3 \end{bmatrix}^T$
Scores
$= S$

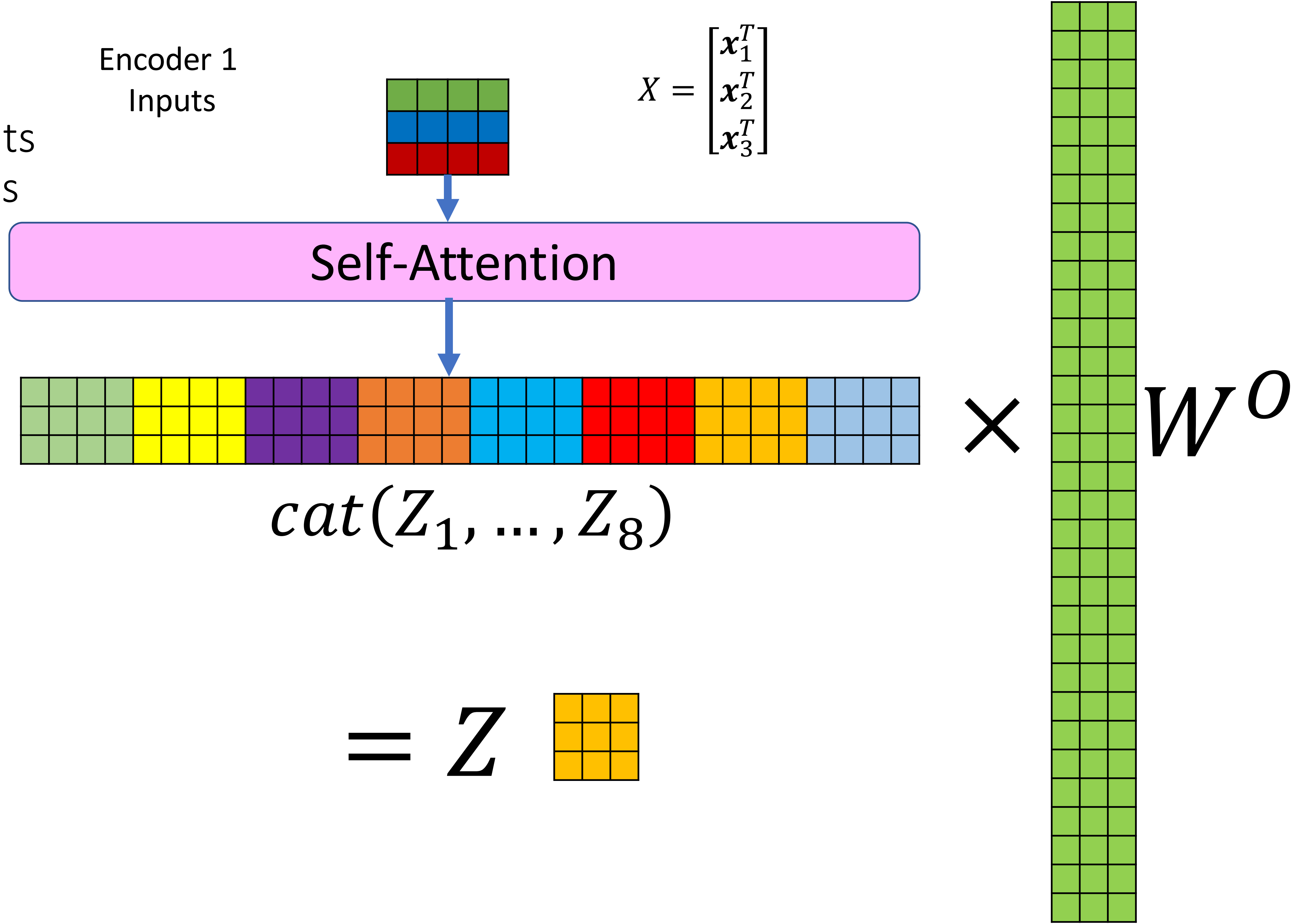


Multi-Head (eg 8-head)
cat
$\boldsymbol{x}_1^T$
opened
$\boldsymbol{x}_2^T$
its
$\boldsymbol{x}_3^T$
$X = \begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \boldsymbol{x}_3^T \end{bmatrix}$
Encoder 1 Inputs
Queries
W_1^Q
$Q_1 = XW_1^Q$
Keys
W_1^K
$K_1 = XW_1^K$
Values
W_1^V
$V_1 = XW_1^V$
Head 1: Z_1
Queries
W_8^Q
$Q_8 = XW_8^Q$
Keys
W_8^K
$K_8 = XW_8^K$
Values
W_8^V
$V_8 = XW_8^V$
Head 8: Z_8

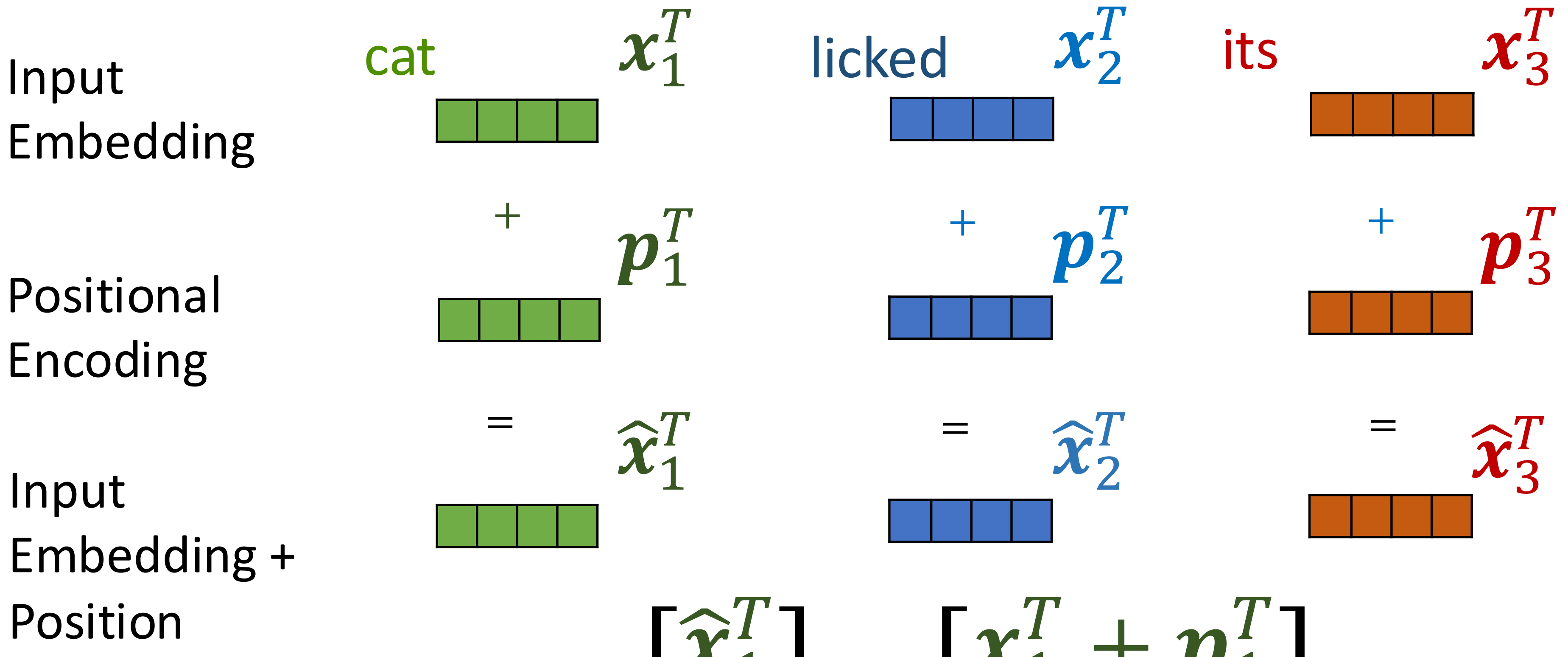
Multi-Head
(eg 8-head)



Multi-Head
(eg 8-head)
Merge Outputs
Apply Weights



Adding Position Info to Inputs



$$\hat{X} = \begin{bmatrix} \widehat{\boldsymbol{x}}_1^T \\ \widehat{\boldsymbol{x}}_2^T \\ \widehat{\boldsymbol{x}}_3^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1^T + \boldsymbol{p}_1^T \\ \boldsymbol{x}_2^T + \boldsymbol{p}_2^T \\ \boldsymbol{x}_3^T + \boldsymbol{p}_3^T \end{bmatrix}$$

Positional Encoding

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_k}}}\right) \quad dim = 2i \;\; is\ even$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_k}}}\right) \quad dim = 2i + 1 \;\; is\ odd$$

$pos = 0,1, \ldots n_{pos-1}$

$dim = 0,1, \ldots n_{dim-1}$

$n = 10000$, d_k is embedding dim

Other positional encoding methods: learnable

Positional Encoding

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_k}}}\right) \quad dim = 2i \ \ is\ even$$

$$d_k = 4$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_k}}}\right) \quad dim = 2i+1 \ \ is\ odd$$

	i / pos	0	0	1	1
cat	0	$\sin\left(\frac{0}{10000^{\frac{2(0)}{d_k}}}\right)$	$\cos\left(\frac{0}{10000^{\frac{2(0)}{d_k}}}\right)$	$\sin\left(\frac{0}{10000^{\frac{2(1)}{d_k}}}\right)$	$\cos\left(\frac{0}{10000^{\frac{2(1)}{d_k}}}\right)$
opened	1	$\sin\left(\frac{1}{10000^{\frac{2(0)}{d_k}}}\right)$	$\cos\left(\frac{1}{10000^{\frac{2(0)}{d_k}}}\right)$	$\sin\left(\frac{1}{10000^{\frac{2(1)}{d_k}}}\right)$	$\cos\left(\frac{1}{10000^{\frac{2(1)}{d_k}}}\right)$
its	2	$\sin\left(\frac{2}{10000^{\frac{2(0)}{d_k}}}\right)$	$\cos\left(\frac{2}{10000^{\frac{2(0)}{d_k}}}\right)$	$\sin\left(\frac{2}{10000^{\frac{2(1)}{d_k}}}\right)$	$\cos\left(\frac{2}{10000^{\frac{2(1)}{d_k}}}\right)$

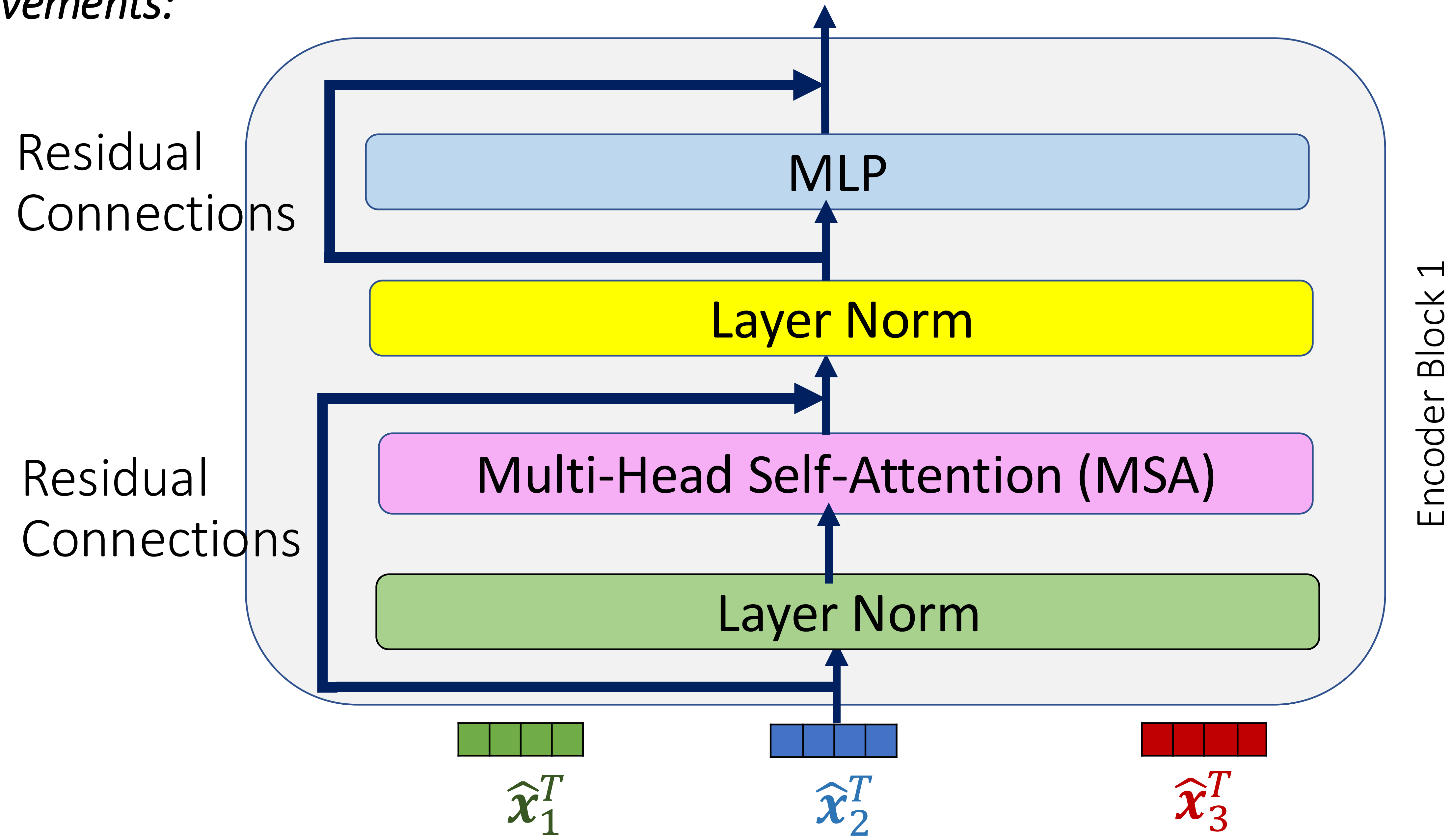
Positional Encoding Matrix with d=4, n=100

Sequence	Index of token, k	i=0	i=0	i=1	i=1
I	0	$P_{00}=\sin(0) = 0$	$P_{01}=\cos(0) = 1$	$P_{02}=\sin(0) = 0$	$P_{03}=\cos(0) = 1$
am	1	$P_{10}=\sin(1/1) = 0.84$	$P_{11}=\cos(1/1) = 0.54$	$P_{12}=\sin(1/10) = 0.10$	$P_{13}=\cos(1/10) = 1.0$
a	2	$P_{20}=\sin(2/1) = 0.91$	$P_{21}=\cos(2/1) = -0.42$	$P_{22}=\sin(2/10) = 0.20$	$P_{23}=\cos(2/10) = 0.98$
Robot	3	$P_{30}=\sin(3/1) = 0.14$	$P_{31}=\cos(3/1) = -0.99$	$P_{32}=\sin(3/10) = 0.30$	$P_{33}=\cos(3/10) = 0.96$

Positional Encoding Matrix for the sequence 'I am a robot'

https://machinelearningmastery.com/a-gentle-introduction-to-positional-encoding-in-transformer-models-part-1/

Improvements:



Layer Normalization vs Batch Normalization

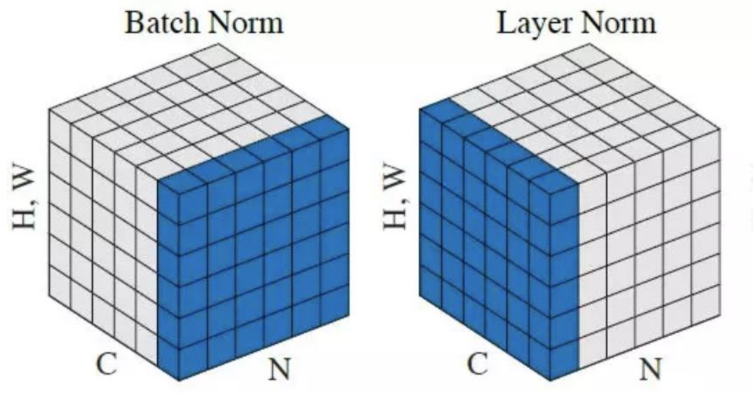
$$\mu = \frac{1}{d}\sum_{i}^{d} x_i$$

$$\sigma^2 = \frac{1}{d}\sum_{i}^{d} (x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

- Batch Normalization – d is across the entire batch
- Layer Normalization - d is across the layer

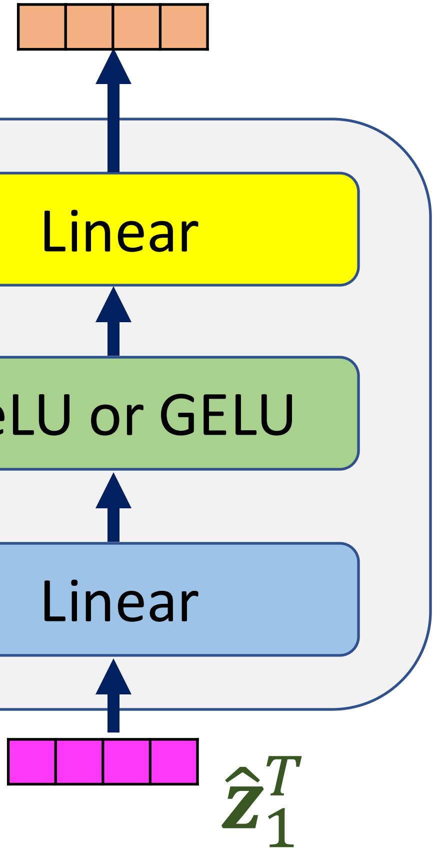
$\hat{x}_i$ is the normalized feature

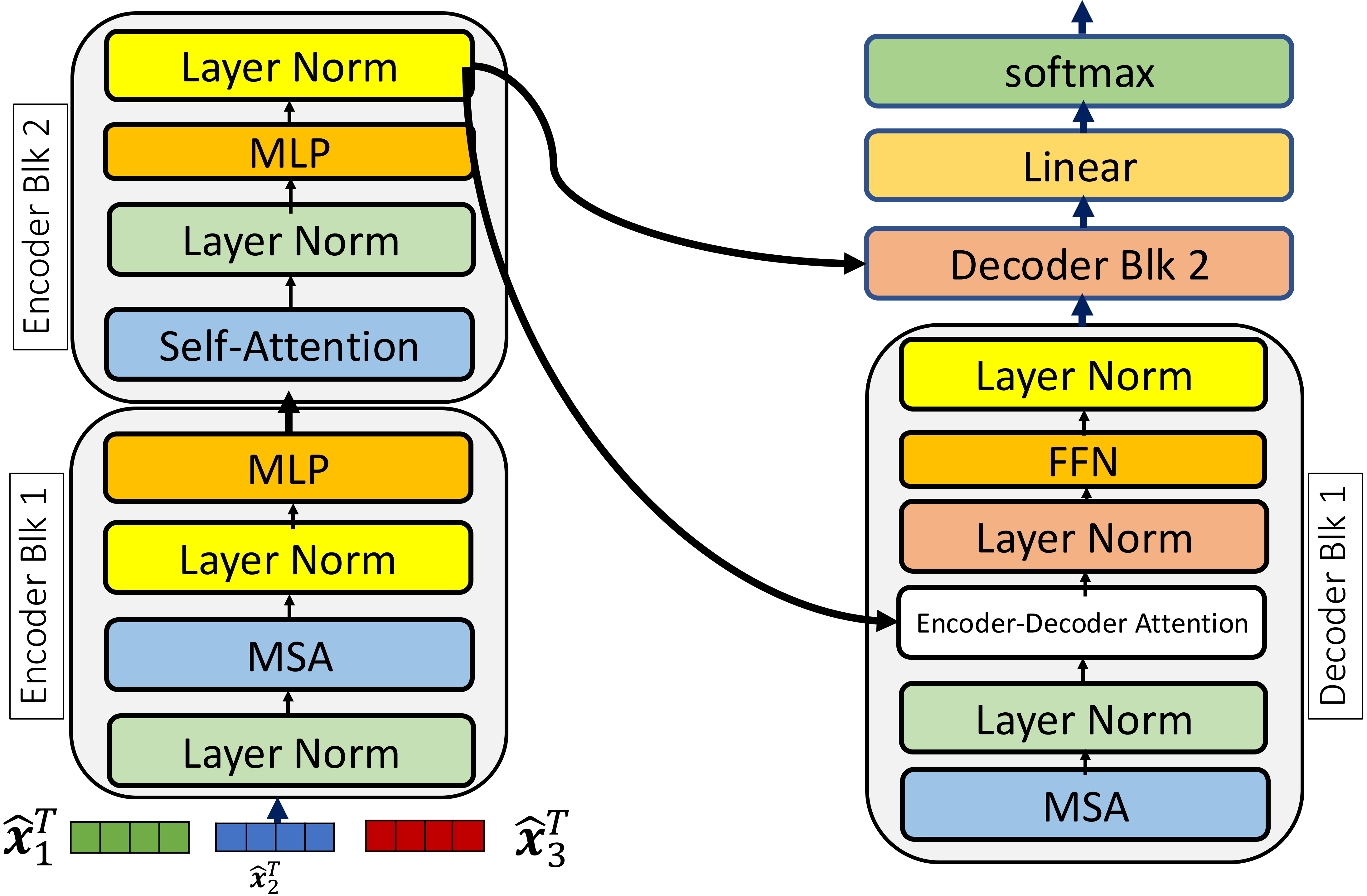


https://paperswithcode.com/method/layer-normalization

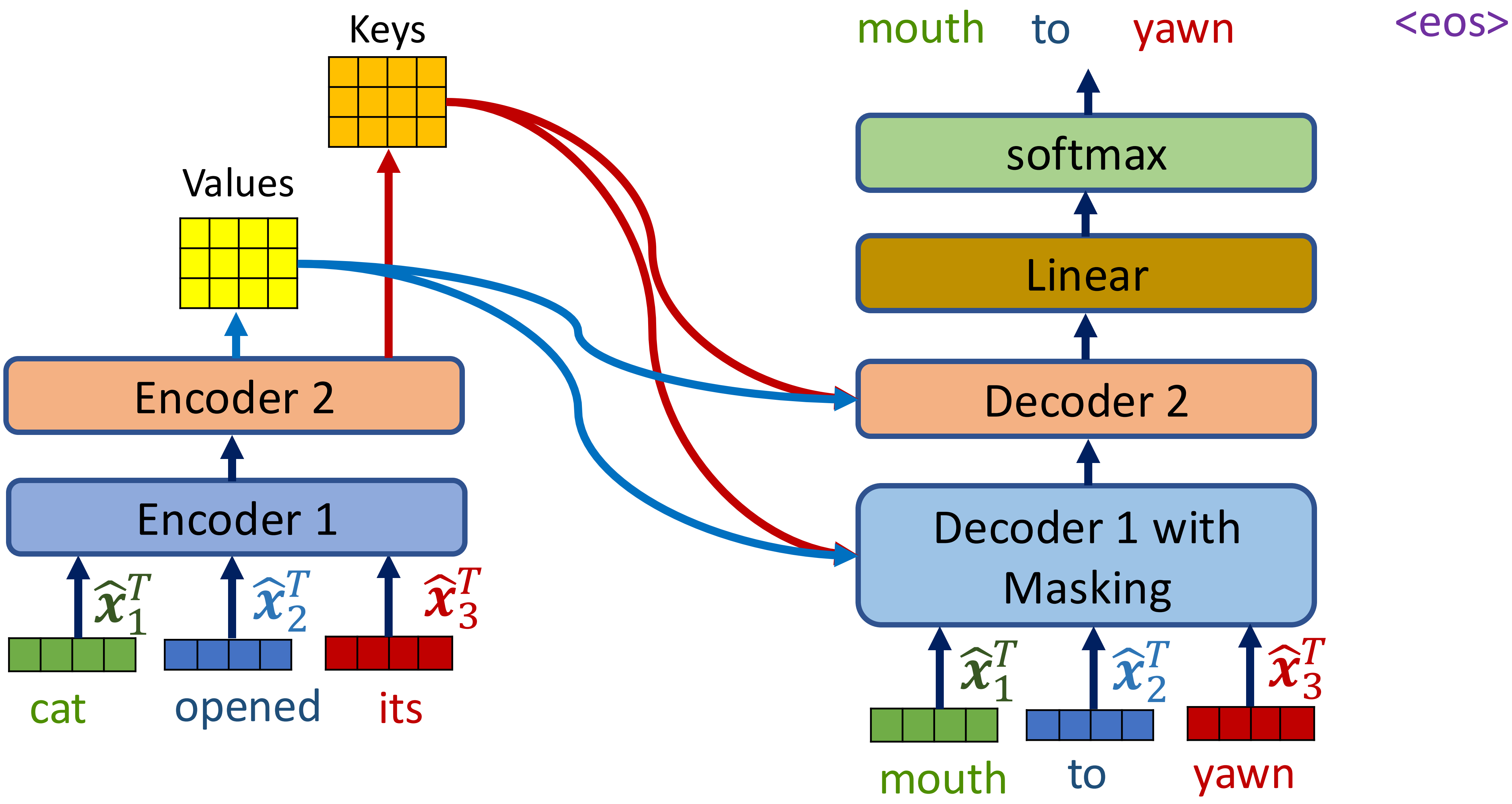
FFN: Feed Forward Neural Network (MLP)

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





Layer Norm
MLP
Layer Norm
Self-Attention
Encoder Blk 2
MLP
Layer Norm
MSA
Layer Norm
Encoder Blk 1
$\widehat{\boldsymbol{x}}_1^T$
$\widehat{\boldsymbol{x}}_2^T$
$\widehat{\boldsymbol{x}}_3^T$
softmax
Linear
Decoder Blk 2
Layer Norm
FFN
Layer Norm
Encoder-Decoder Attention
Layer Norm
MSA
Decoder Blk 1



Masking prevents Decoder 1 from seeing the future.
Decoder 1 relies only on the previous outputs.

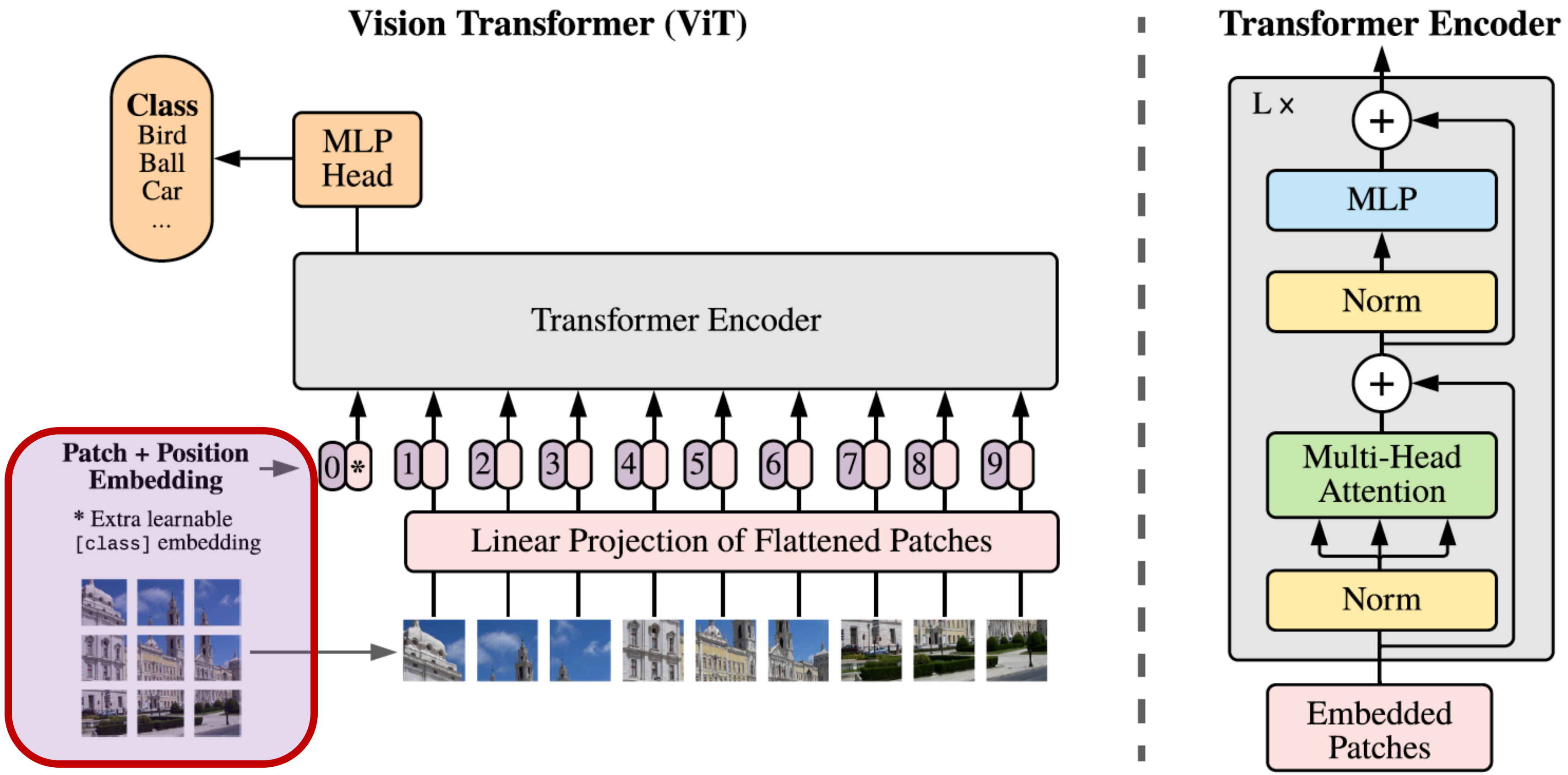


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings to the resulting sequence of vectors, and feed the patches to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

AN IMAGE IS WORTH 16X16 WORDS:
TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE, ICLR 2021

Inductive Bias

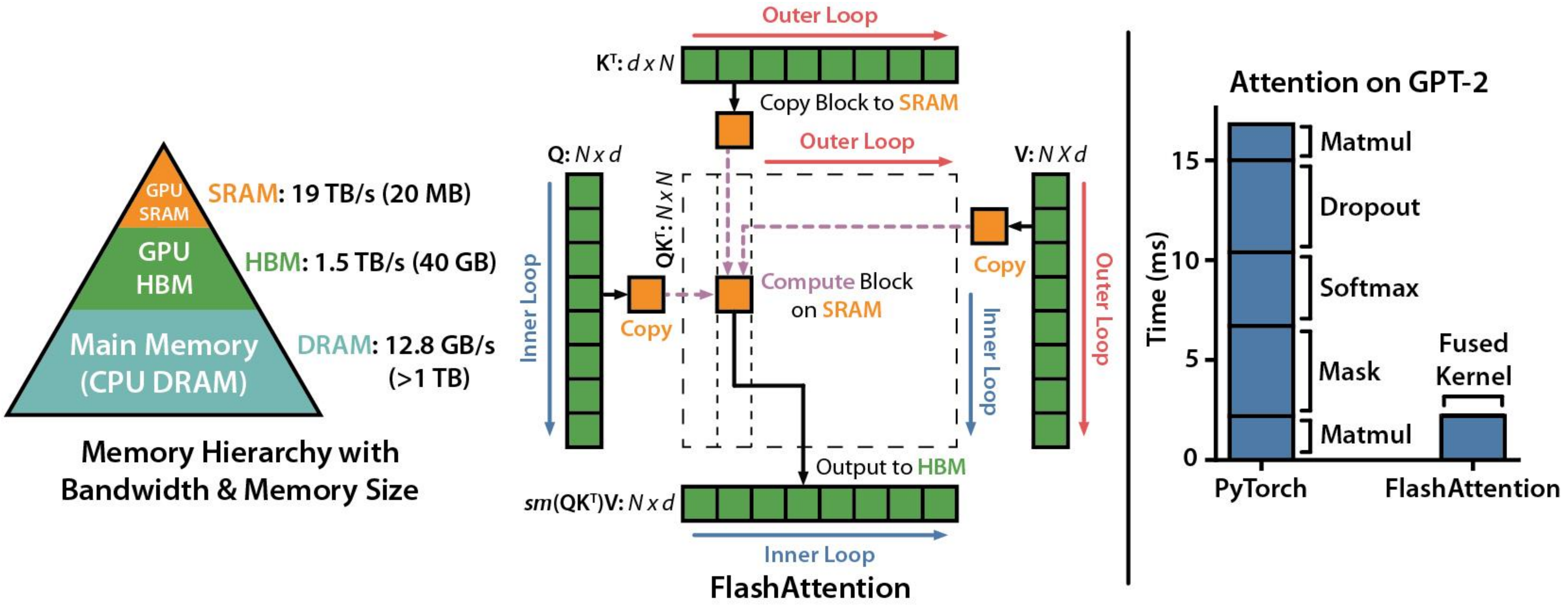
Transformers lack some inductive biases inherent to CNNs, such as translation equivariance and scale invariance (w/ maxpool), and therefore do not generalize well when trained on insufficient amounts of data.

However, the picture changes if we train the models on large datasets (14M-300M images). We find that large scale training trumps inductive bias.

Improvements on Transformers

- Flash Attention
- Rotary Positional Embedding

Flash Attention



Dao, Tri, et al. "Flashattention: Fast and memory-efficient exact attention with io-awareness NeuRIPS 2022

Dao, Tri. "Flashattention-2: Faster attention with better parallelism and work partitioning ICLR 2023

Rotary Position Encoding

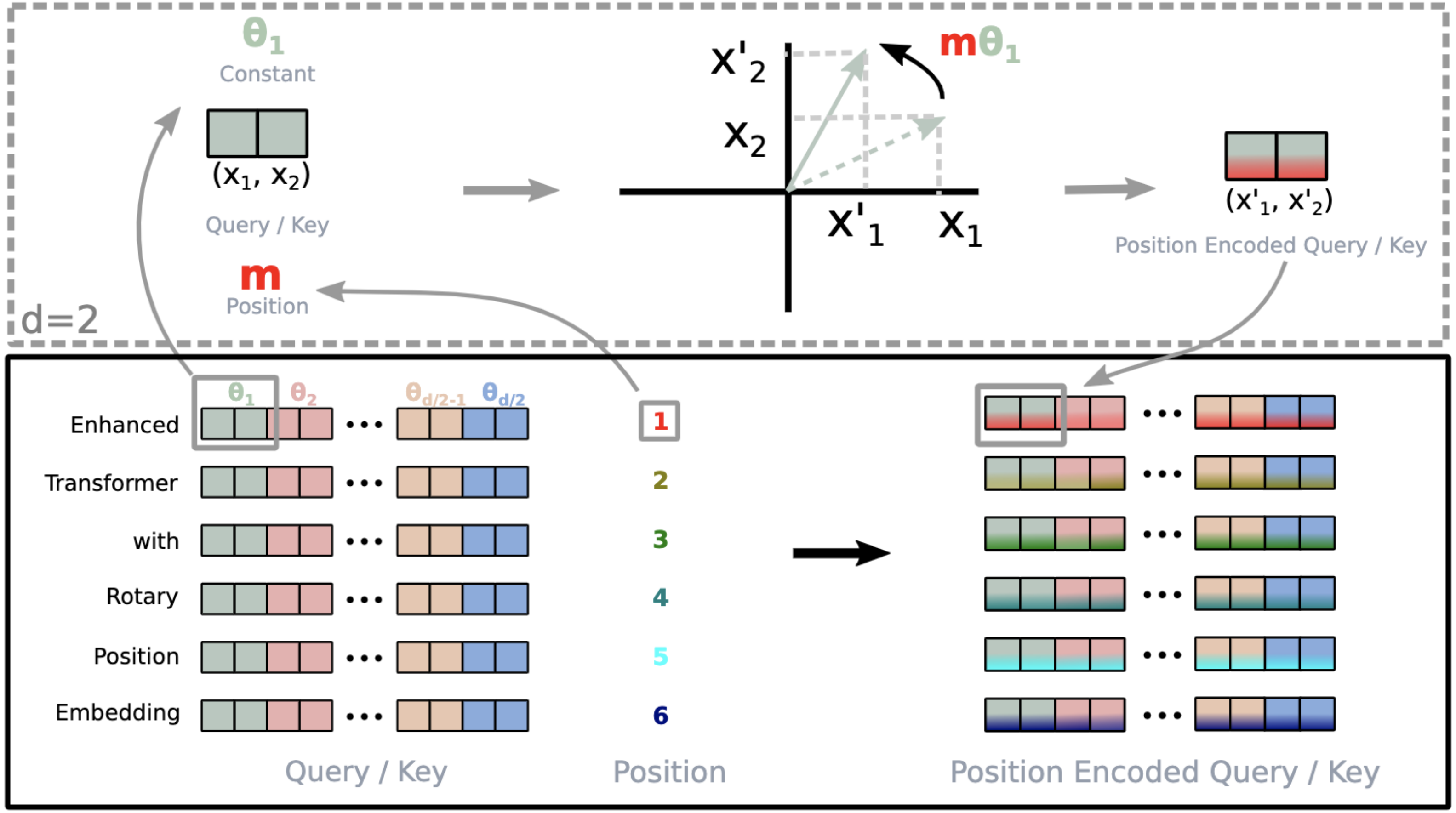


Figure 1: Implementation of Rotary Position Embedding(RoPE).

Su, Jianlin, et al. "Roformer: Enhanced transformer with rotary position embedding." Neurocomputing 568 (2024): 127063.

References

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.

Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020). ICLR 2021.

Illustrated Transformer, http://jalammar.github.io/illustrated-transformer/

Transformers from Scratch, http://peterbloem.nl/blog/transformers

Transformer Family, https://lilianweng.github.io/lil-log/2020/04/07/the-transformer-family.html

In Summary

Transformers could be the most important breakthrough in the recent history of deep learning

Transformers have been used to produce state-of-the-art performances in language, vision, audio, and multi-modal domains

Expect more development in this field in the near future

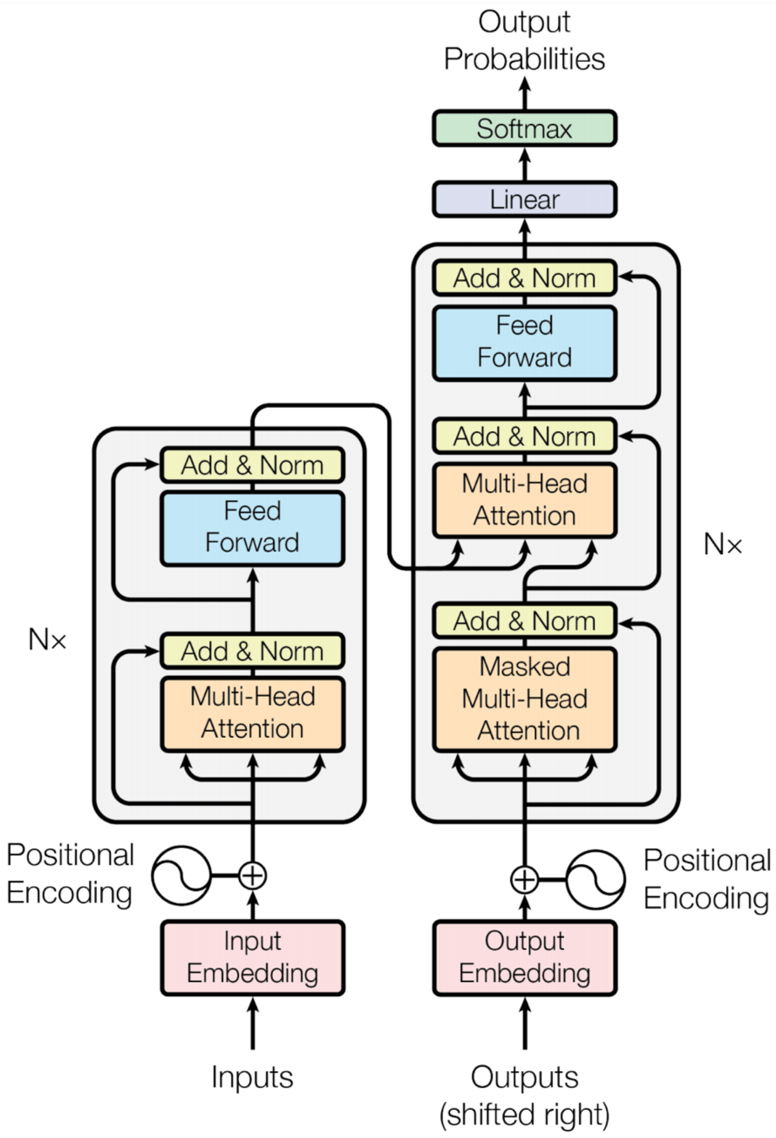


Figure 1: The Transformer - model architecture.

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.

Code demo is next