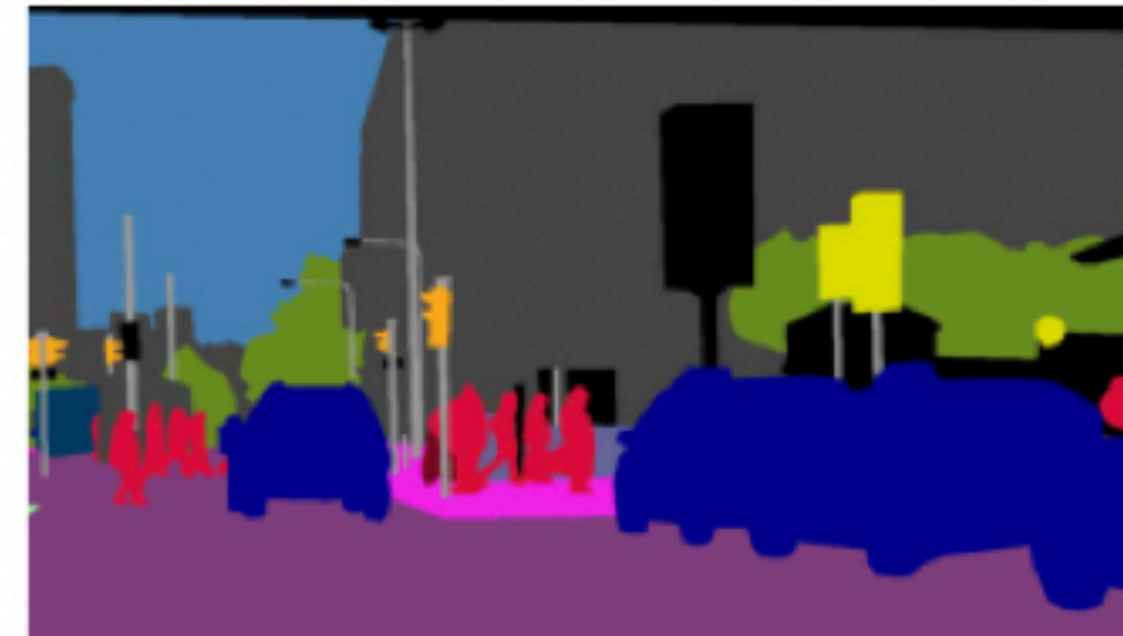


Panoptic Segmentation with Transformers Tutorial

Mennatullah Siam

Semantic/Instance/Panoptic Segmentation

semantic



Semantic/Instance/Panoptic Segmentation

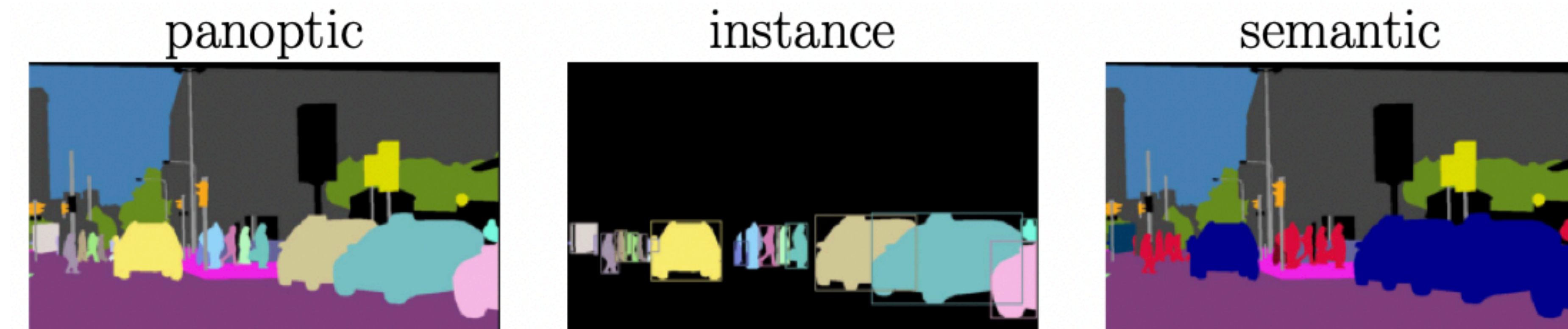
instance



semantic



Semantic/Instance/Panoptic Segmentation



Holistic Scene Understanding

Transformers

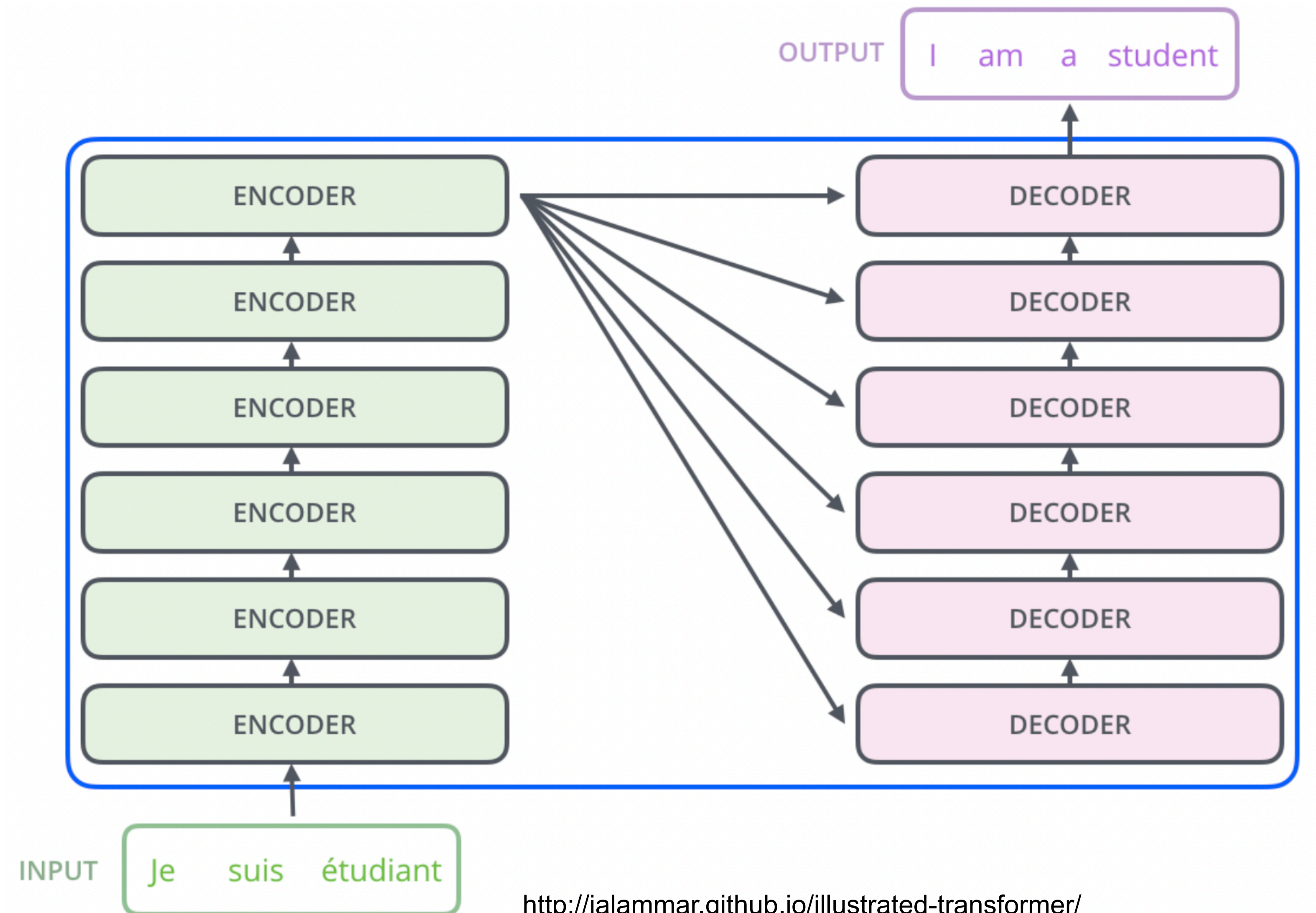
Example



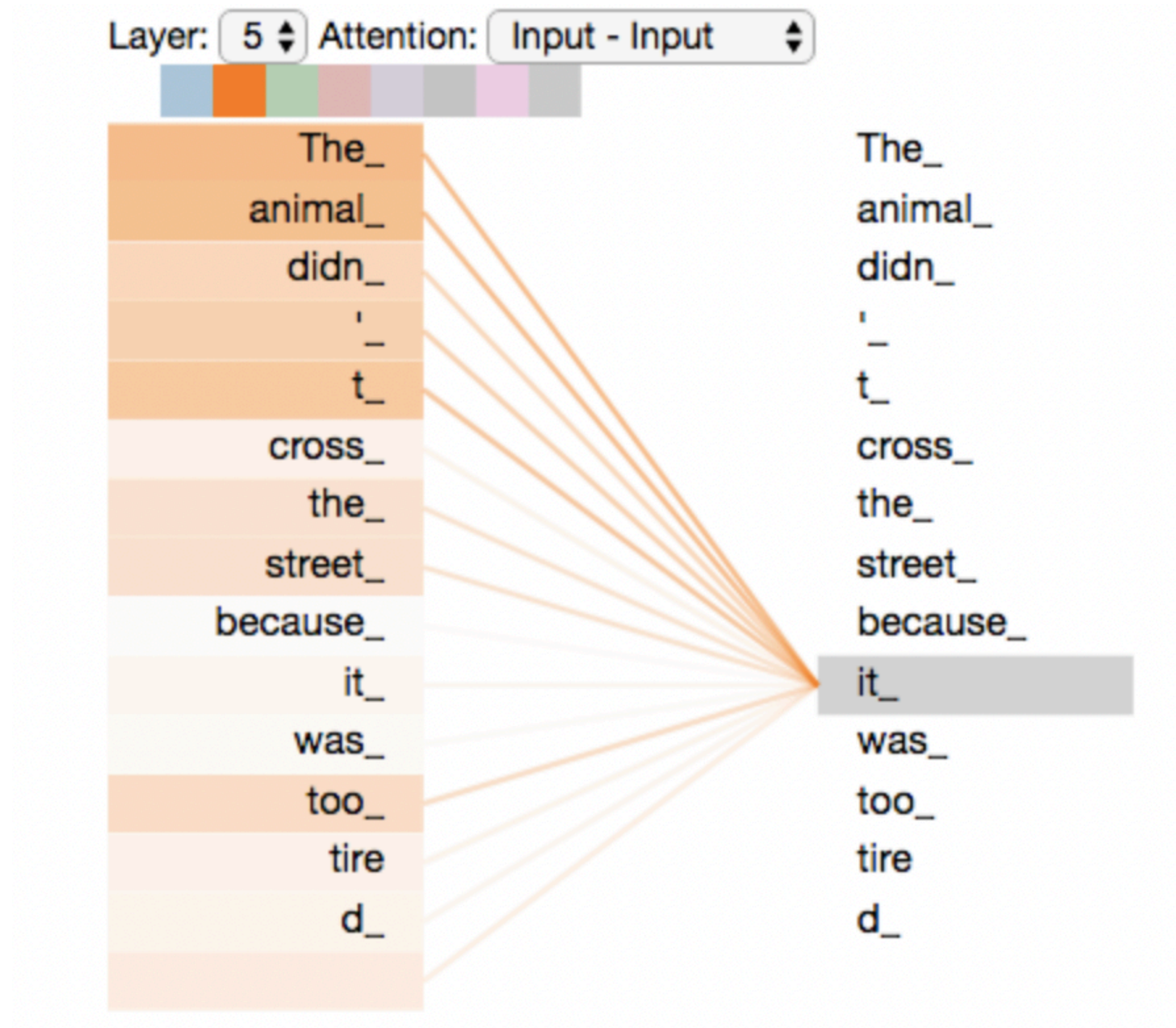
<http://jalammar.github.io/illustrated-transformer/>

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Transformers



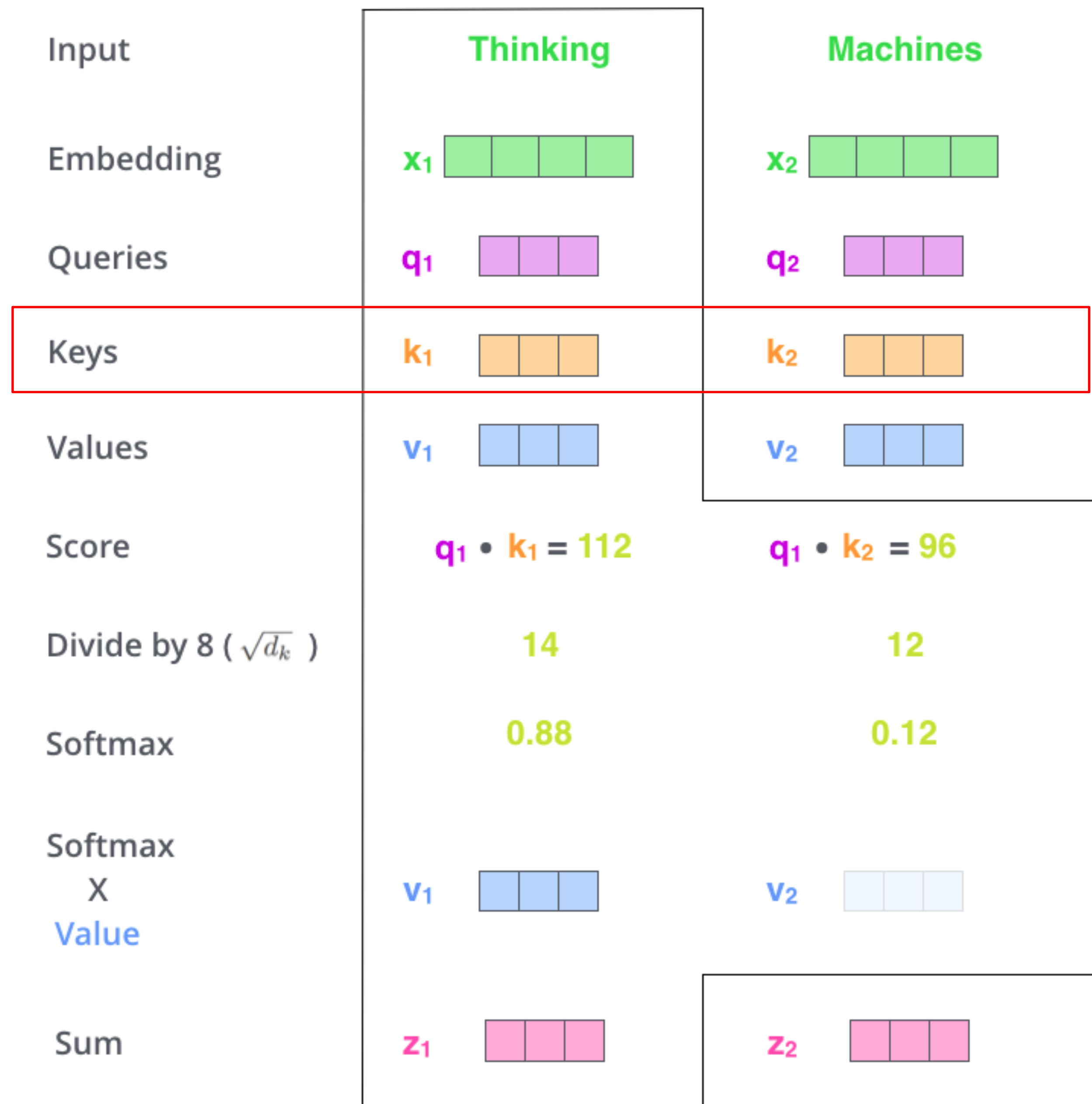
Self Attention



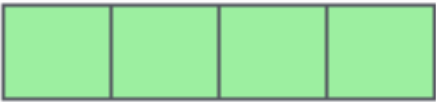
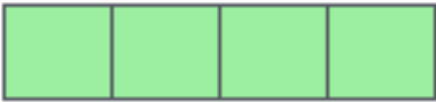


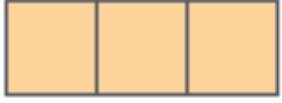
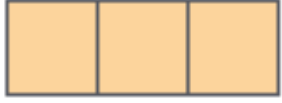

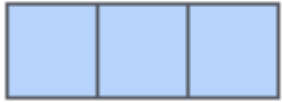
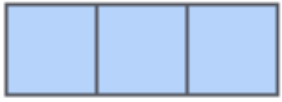
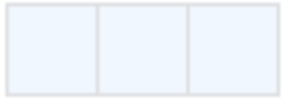
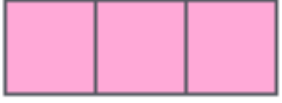
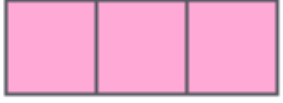
<http://jalammar.github.io/illustrated-transformer/>

Single Head Attention

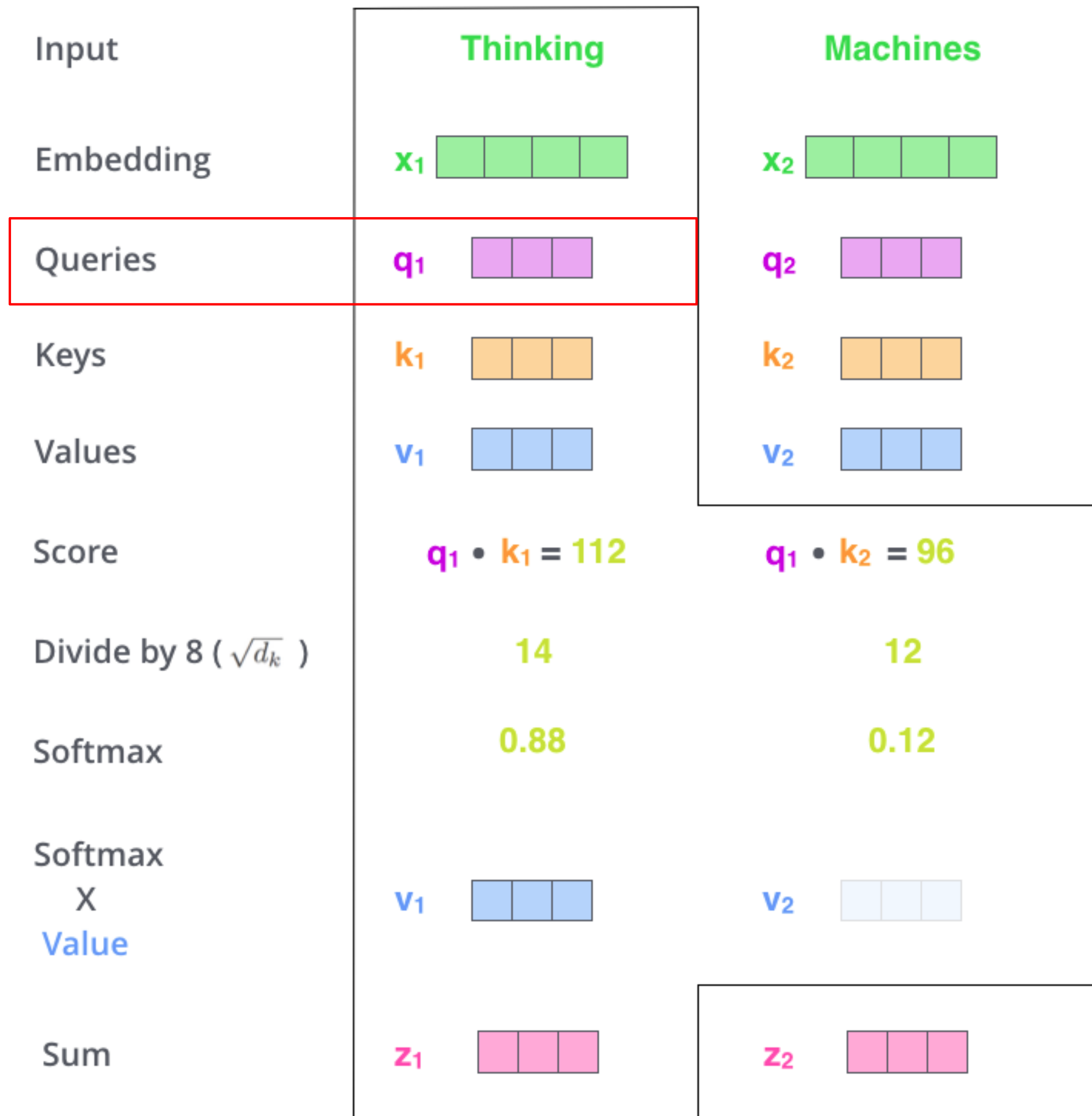
Think of this as your
Dictionary keys used for
addressing



Detailed information - what I want to read out from memory

Input	Thinking	Machines
Embedding	x_1 	x_2 
Queries	q_1 	q_2 
Keys	k_1 	k_2 
Values	v_1 	v_2 
Score	$q_1 \cdot k_1 = 112$	$q_1 \cdot k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12
Softmax X Value	v_1 	v_2 
Sum	z_1 	z_2 

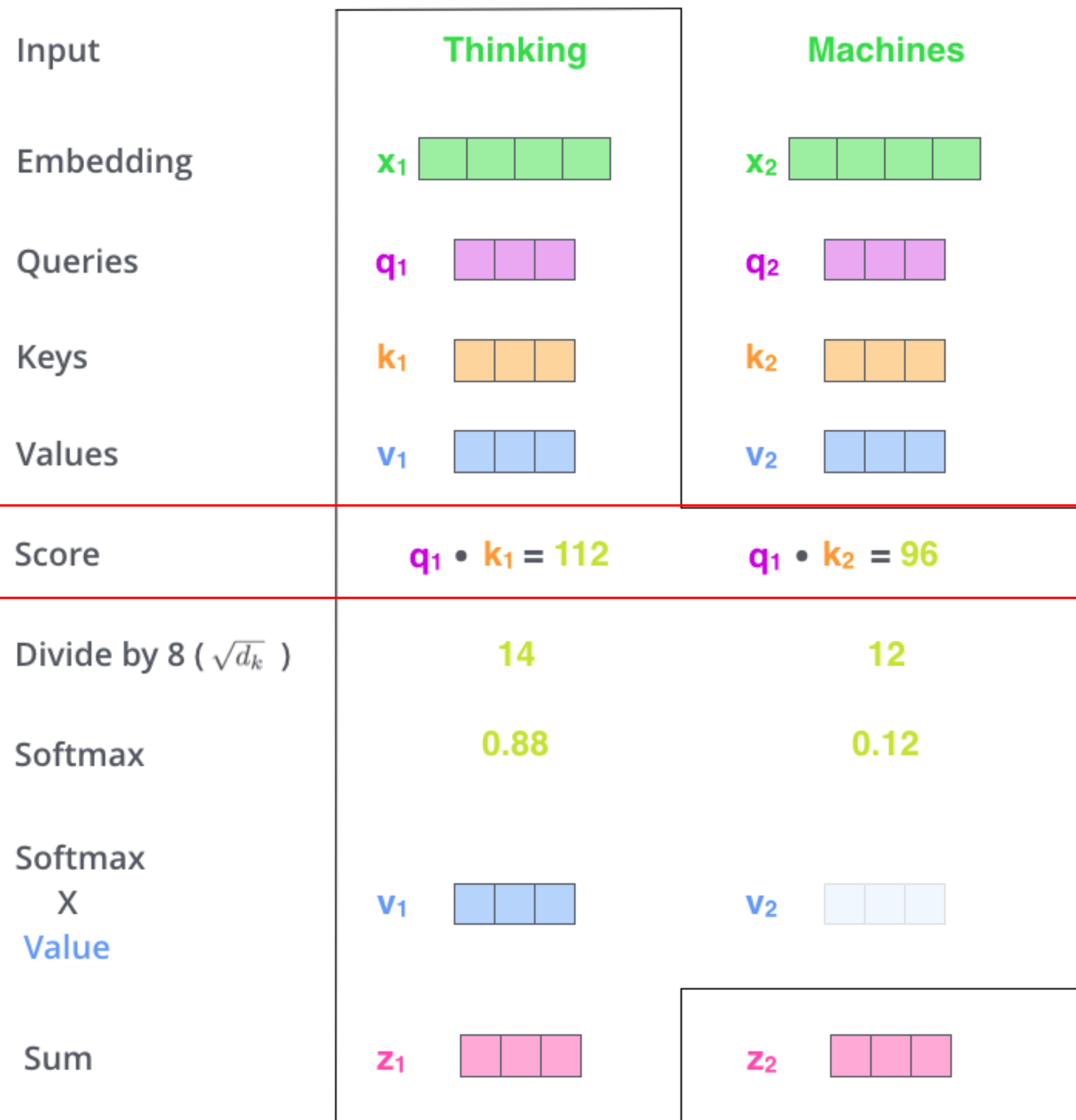
My current word embedding



1] Relate

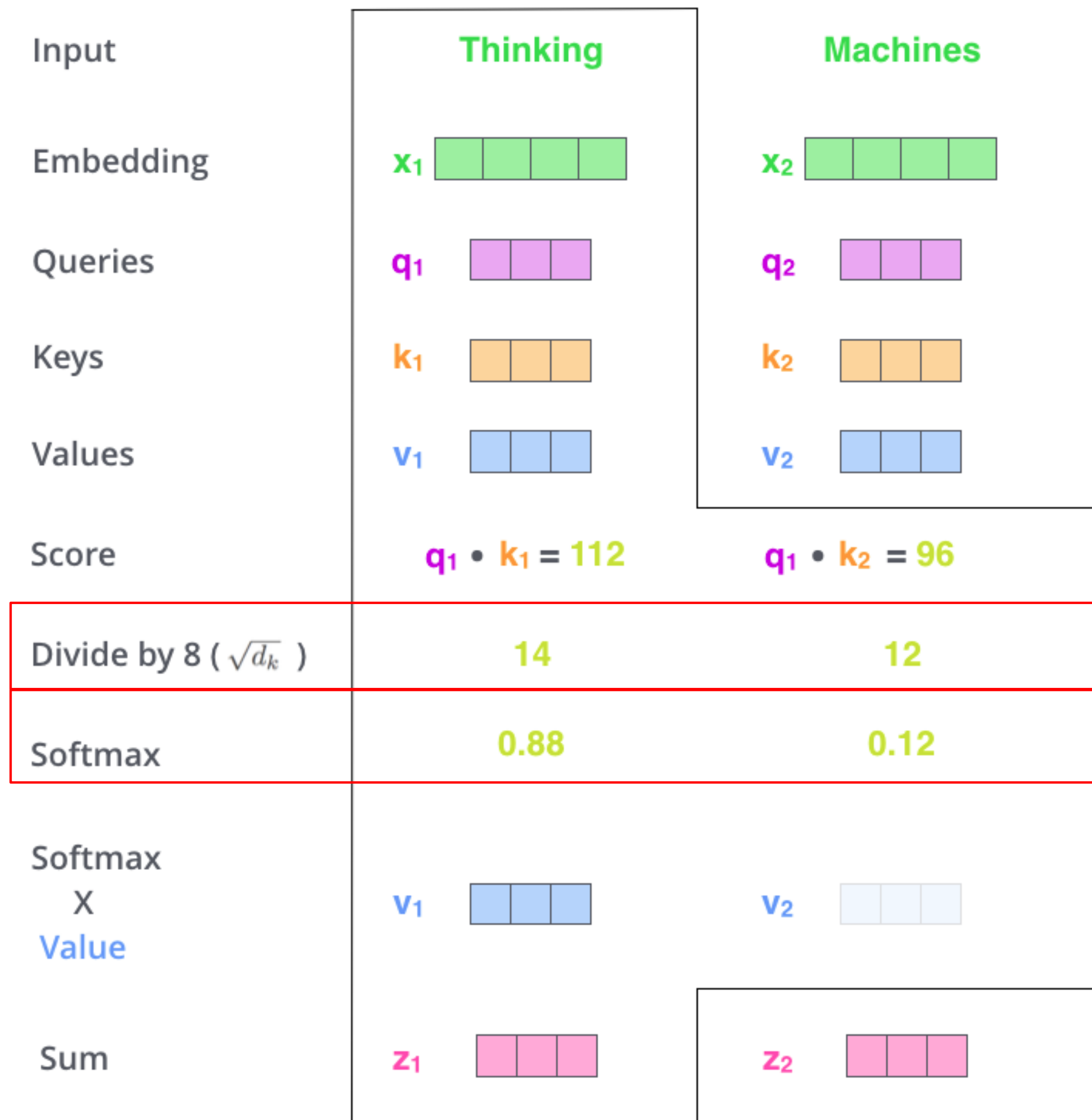
Compatibility bet. each
embedding in the
dictionary to myself

Scalar, Pairwise



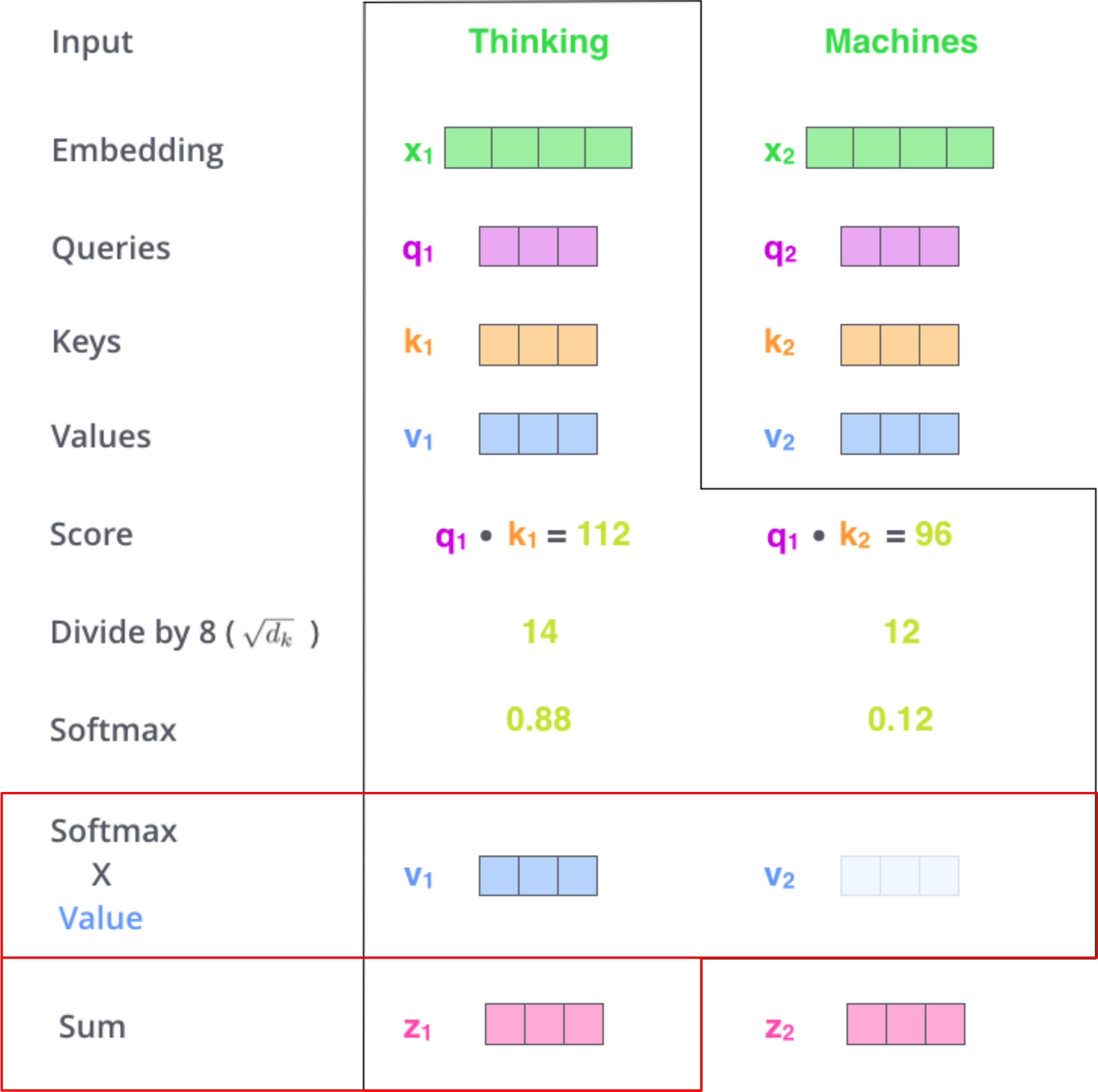
1] Relate

Scaled: because for large values of d_k
 → large values of dot product
 → pushes the softmax to have small gradients.



2] Aggregate

Aggregate information from all tokens



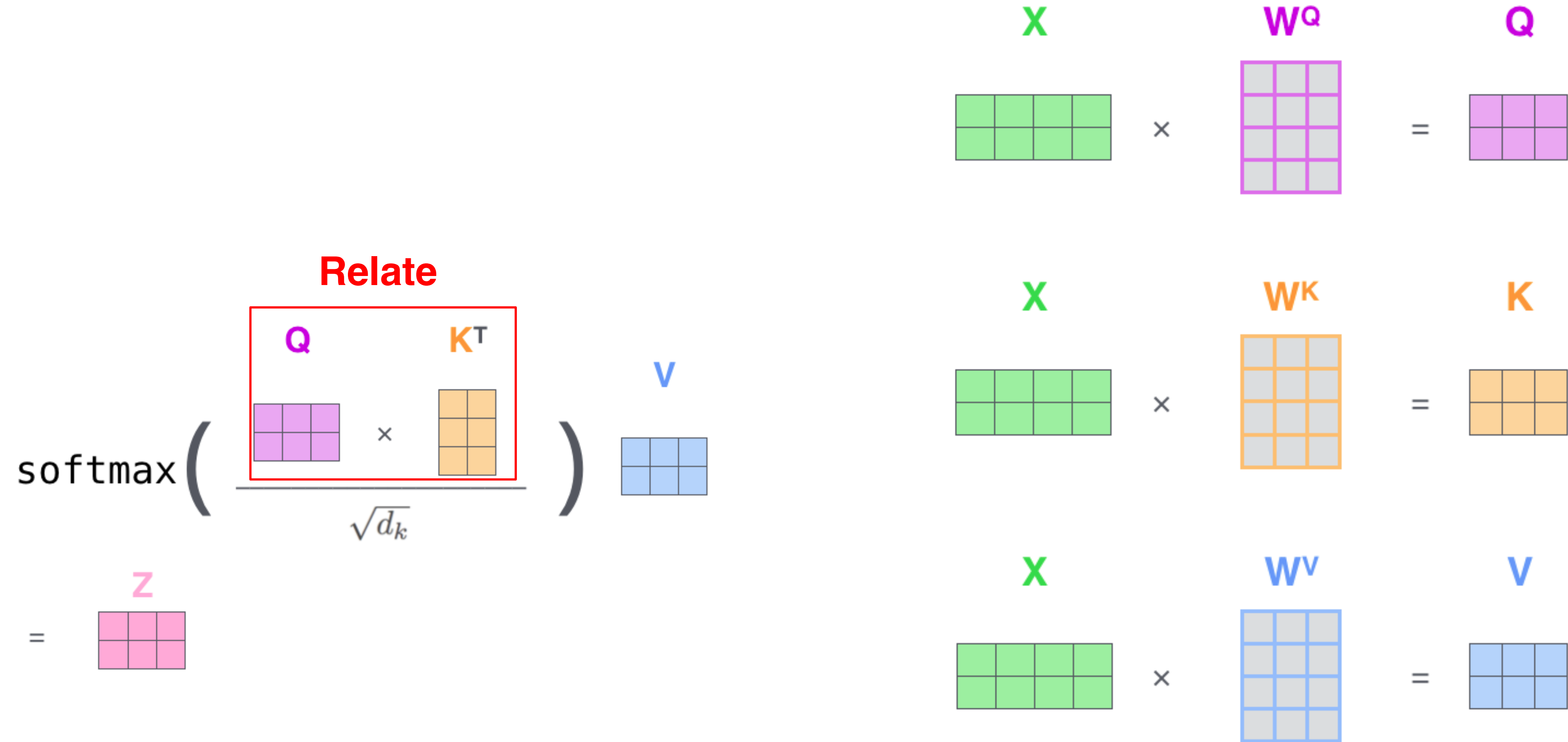
Single-Headed Attention

$$\begin{matrix} \textcolor{green}{X} & & \textcolor{violet}{W^Q} & & \textcolor{violet}{Q} \\ \begin{array}{|c|c|c|c|} \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \end{array} & \times & \begin{array}{|c|c|c|} \hline \textcolor{violet}{\square} & \textcolor{violet}{\square} & \textcolor{violet}{\square} \\ \hline \textcolor{violet}{\square} & \textcolor{violet}{\square} & \textcolor{violet}{\square} \\ \hline \textcolor{violet}{\square} & \textcolor{violet}{\square} & \textcolor{violet}{\square} \\ \hline \end{array} & = & \begin{array}{|c|c|c|} \hline \textcolor{violet}{\square} & \textcolor{violet}{\square} & \textcolor{violet}{\square} \\ \hline \textcolor{violet}{\square} & \textcolor{violet}{\square} & \textcolor{violet}{\square} \\ \hline \end{array} \end{matrix}$$

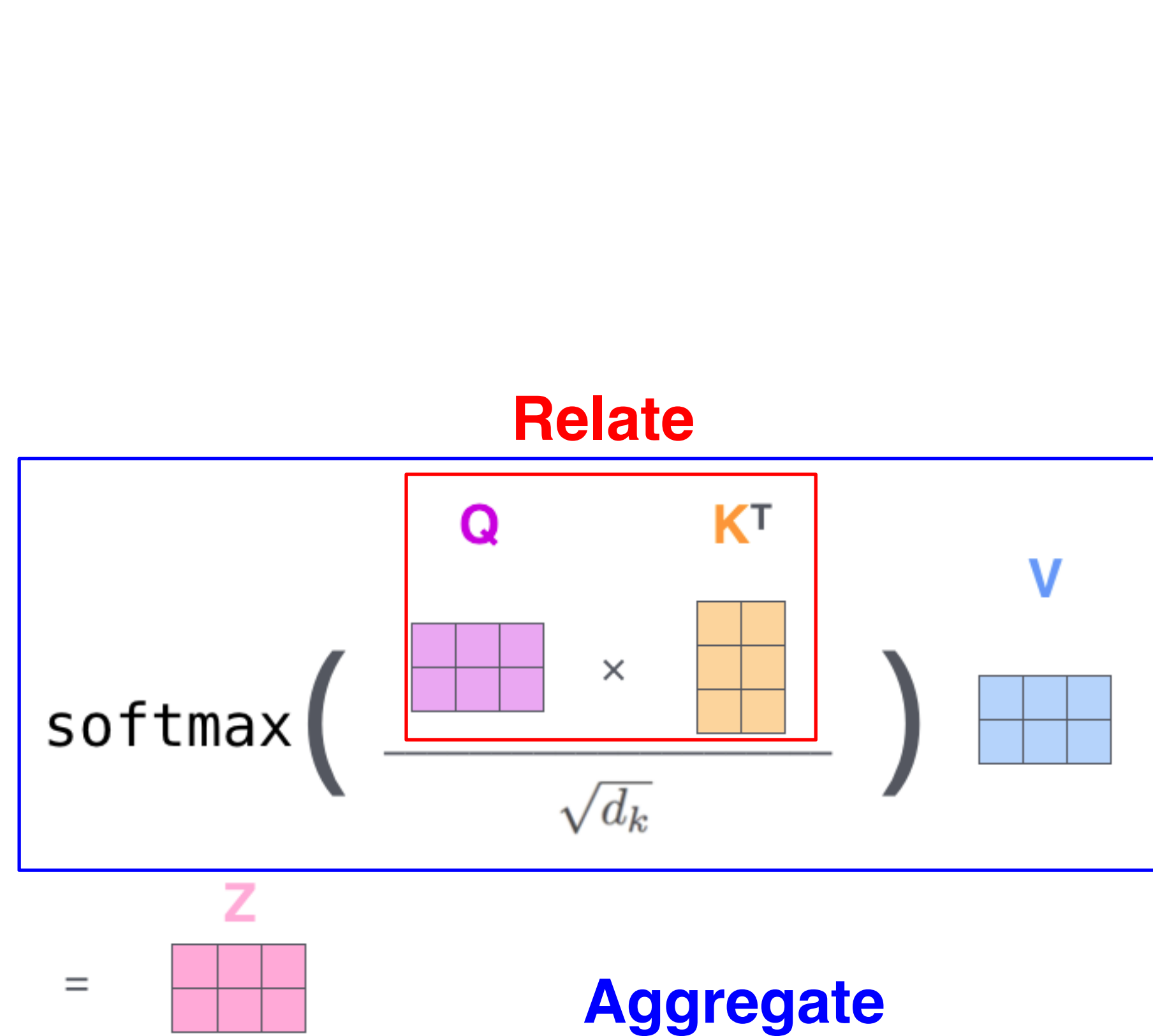
$$\begin{matrix} \textcolor{green}{X} & & \textcolor{brown}{W^K} & & \textcolor{brown}{K} \\ \begin{array}{|c|c|c|c|} \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \end{array} & \times & \begin{array}{|c|c|c|} \hline \textcolor{brown}{\square} & \textcolor{brown}{\square} & \textcolor{brown}{\square} \\ \hline \textcolor{brown}{\square} & \textcolor{brown}{\square} & \textcolor{brown}{\square} \\ \hline \textcolor{brown}{\square} & \textcolor{brown}{\square} & \textcolor{brown}{\square} \\ \hline \end{array} & = & \begin{array}{|c|c|c|} \hline \textcolor{brown}{\square} & \textcolor{brown}{\square} & \textcolor{brown}{\square} \\ \hline \textcolor{brown}{\square} & \textcolor{brown}{\square} & \textcolor{brown}{\square} \\ \hline \end{array} \end{matrix}$$

$$\begin{matrix} \textcolor{green}{X} & & \textcolor{blue}{W^V} & & \textcolor{blue}{V} \\ \begin{array}{|c|c|c|c|} \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} & \textcolor{green}{\square} \\ \hline \end{array} & \times & \begin{array}{|c|c|c|} \hline \textcolor{blue}{\square} & \textcolor{blue}{\square} & \textcolor{blue}{\square} \\ \hline \textcolor{blue}{\square} & \textcolor{blue}{\square} & \textcolor{blue}{\square} \\ \hline \textcolor{blue}{\square} & \textcolor{blue}{\square} & \textcolor{blue}{\square} \\ \hline \end{array} & = & \begin{array}{|c|c|c|} \hline \textcolor{blue}{\square} & \textcolor{blue}{\square} & \textcolor{blue}{\square} \\ \hline \textcolor{blue}{\square} & \textcolor{blue}{\square} & \textcolor{blue}{\square} \\ \hline \end{array} \end{matrix}$$

Single-Headed Attention



Single-Headed Attention



Multihead Attention

1) This is our
input sentence*

2) We embed
each word*

Thinking
Machines



* In all encoders other than #0,
we don't need embedding.
We start directly with the output
of the encoder right below this one



Multihead Attention

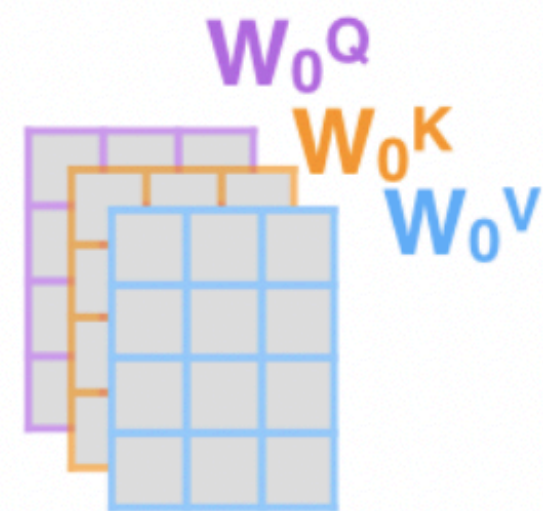
1) This is our
input sentence*

Thinking
Machines

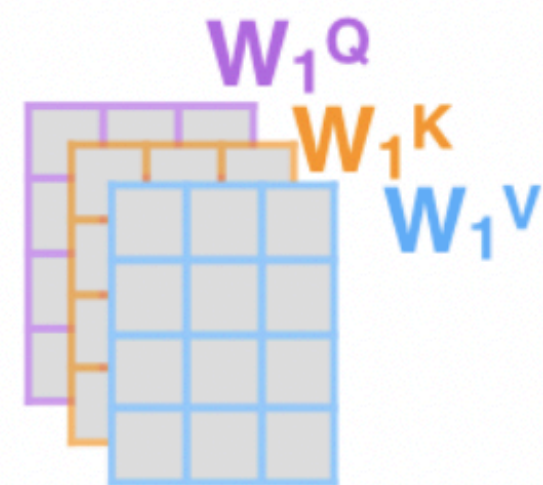
2) We embed
each word*



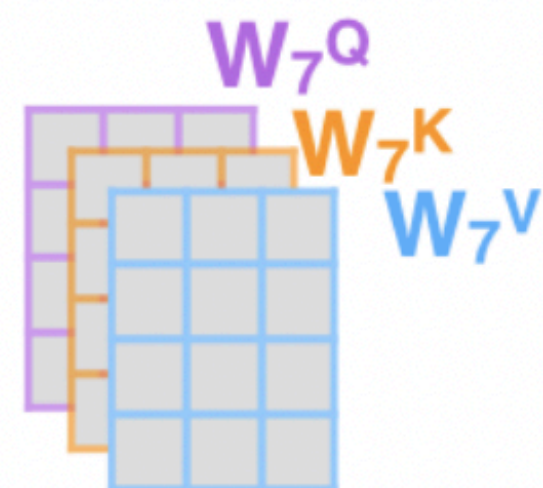
3) Split into 8 heads.
We multiply X or
 R with weight matrices



* In all encoders other than #0,
we don't need embedding.
We start directly with the output
of the encoder right below this one



...



Multihead Attention

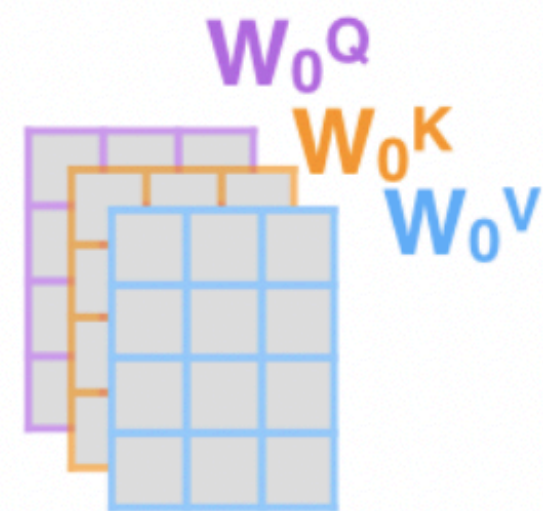
1) This is our input sentence*

Thinking
Machines

2) We embed each word*



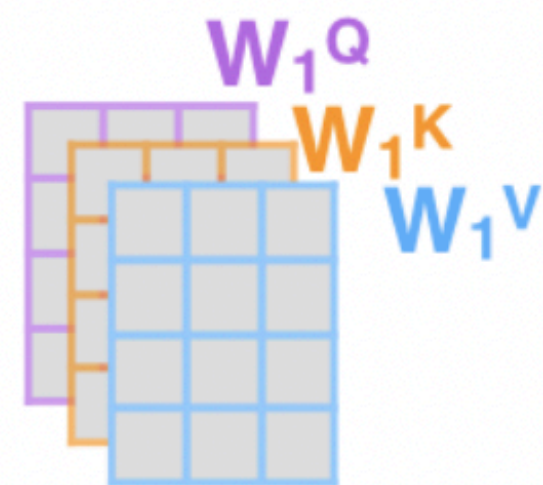
3) Split into 8 heads.
We multiply X or R with weight matrices



4) Calculate attention using the resulting $Q/K/V$ matrices

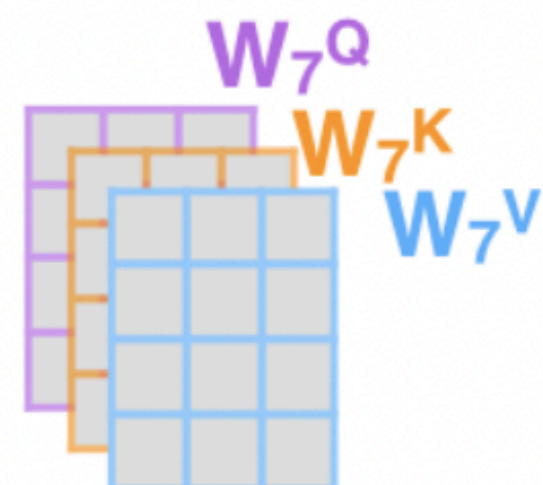


* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one



...

...



Multihead Attention

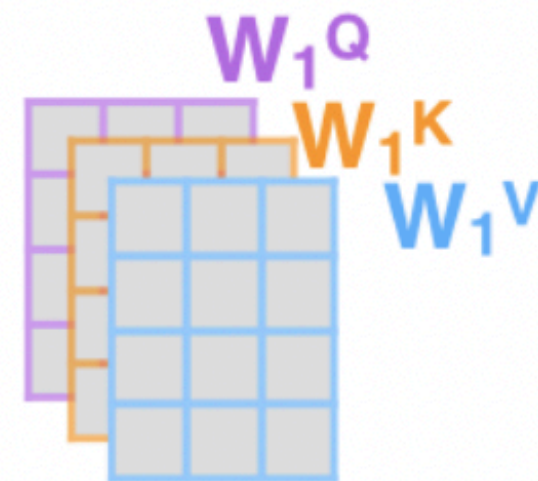
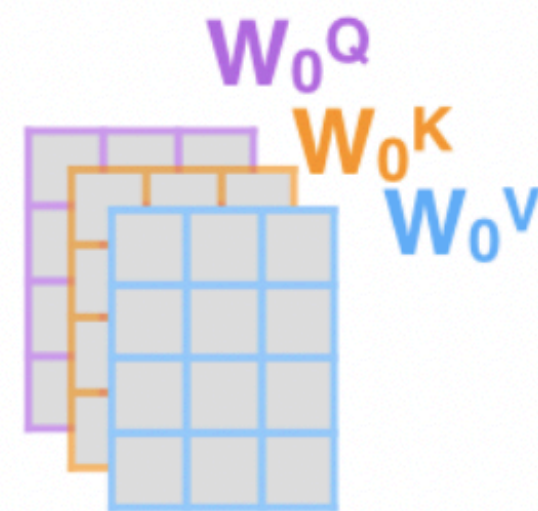
1) This is our input sentence*

Thinking
Machines

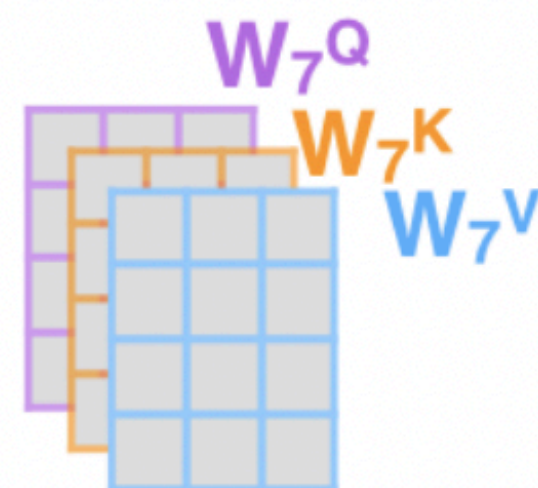
2) We embed each word*



3) Split into 8 heads.
We multiply X or R with weight matrices



...



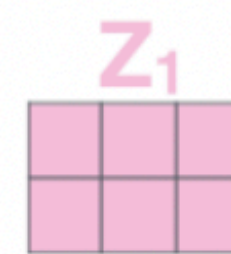
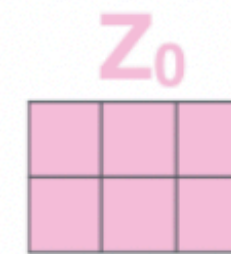
4) Calculate attention using the resulting $Q/K/V$ matrices



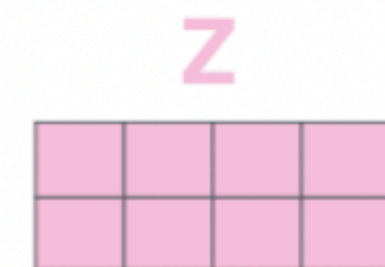
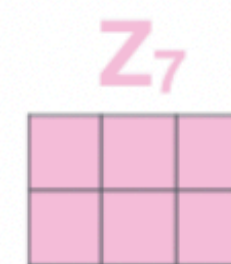
...



5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer



...



* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one



Transformers in Computer Vision

April 2020

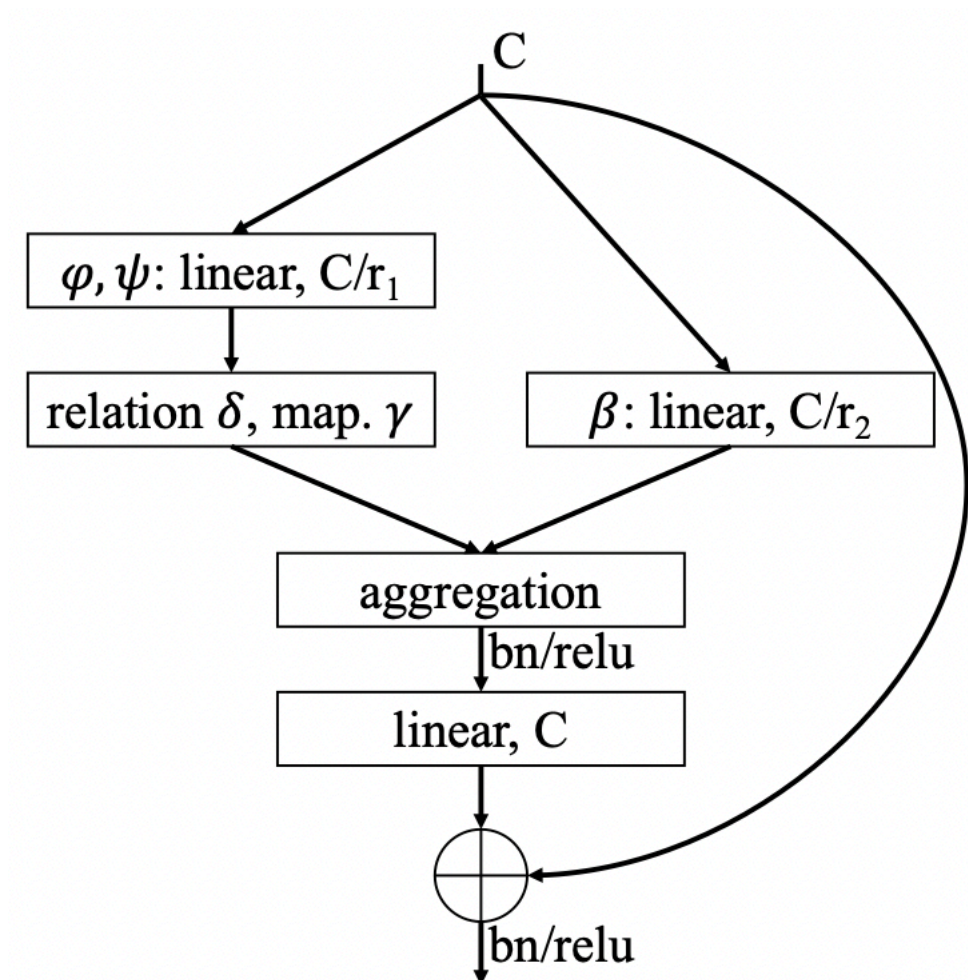
May 2020

October 2020

Exploring Self Attention for
Image Recognition

DetR

Vision Transformers



Transformers in Computer Vision

April 2020

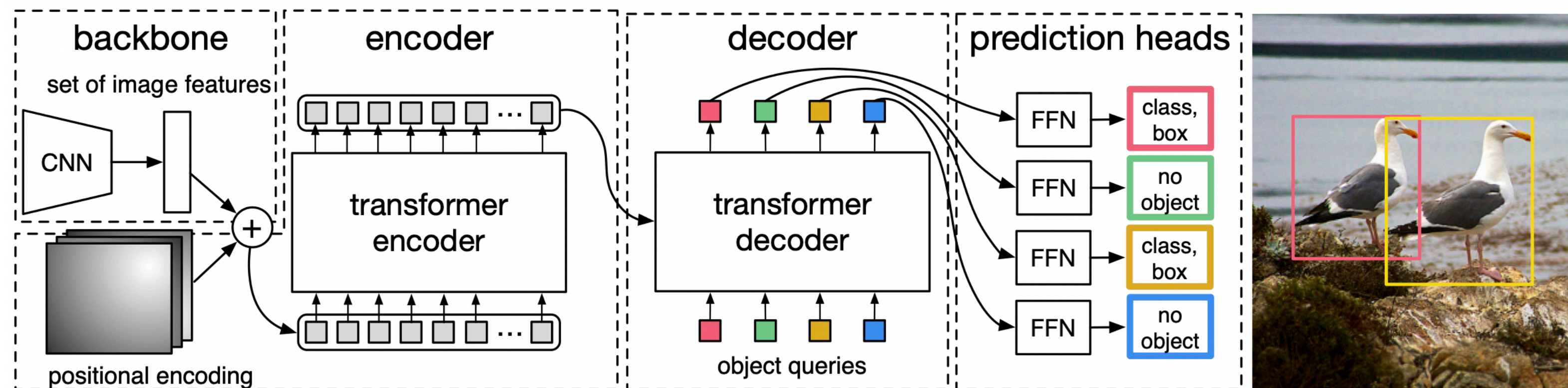
May 2020

June 2021

Exploring Self Attention for
Image Recognition

DetR

Vision Transformers



Transformers in Computer Vision

April 2020

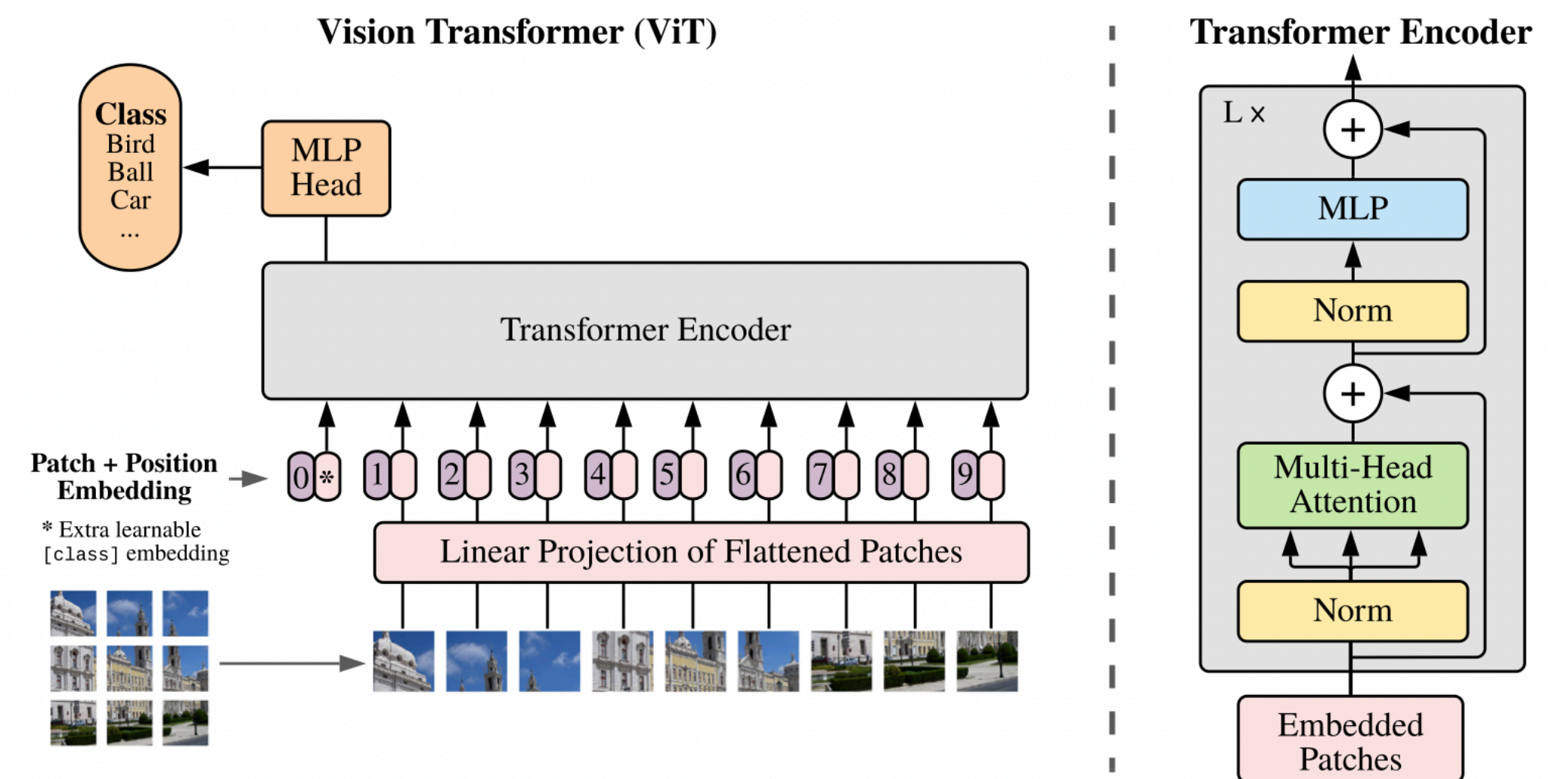
May 2020

June 2021

Exploring Self Attention for
Image Recognition

DetR

Vision Transformers

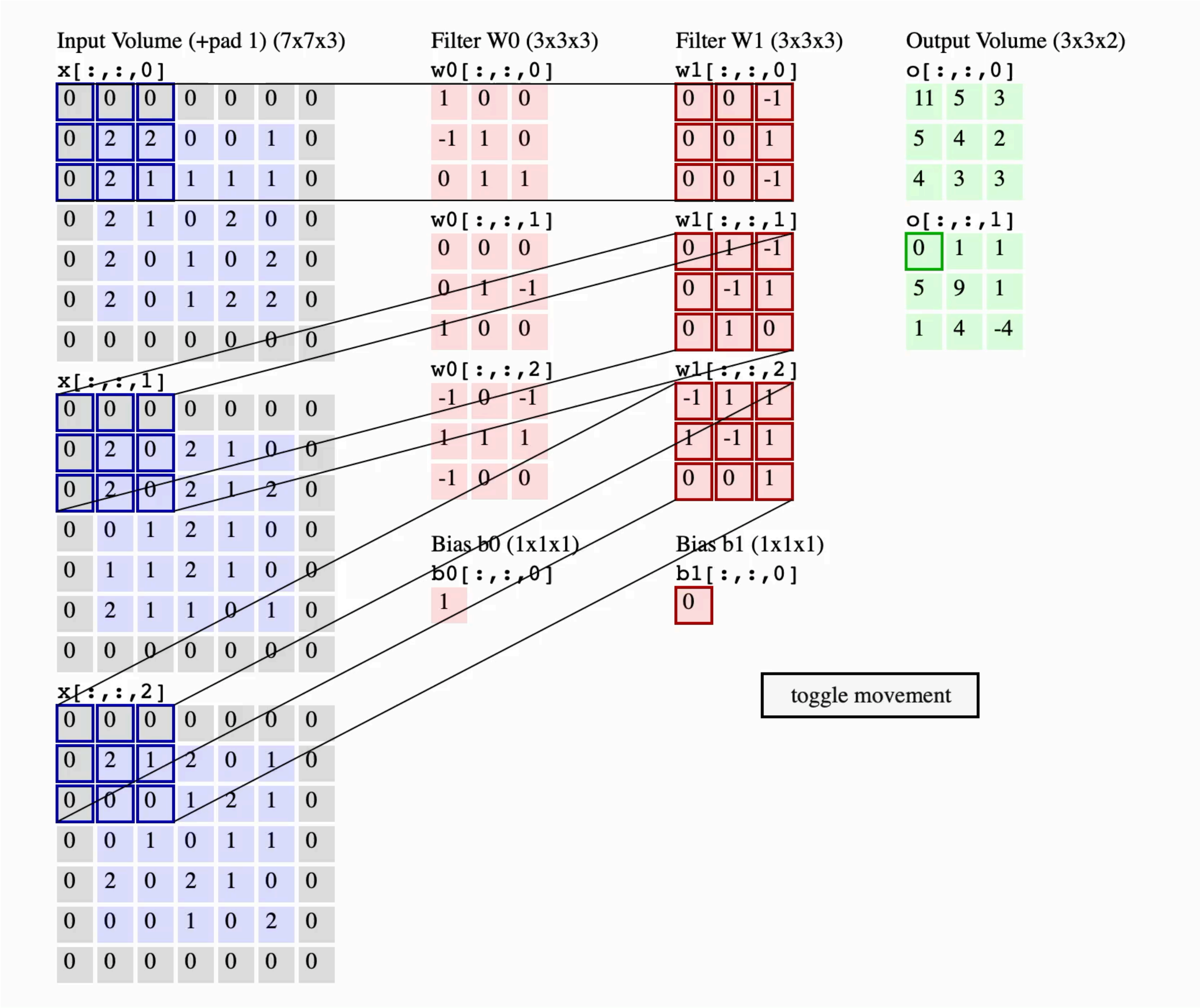


Self Attention: A New Perspective

Convolutional Networks

Aggregate Function

Fixed Weights



Self Attention: A New Perspective

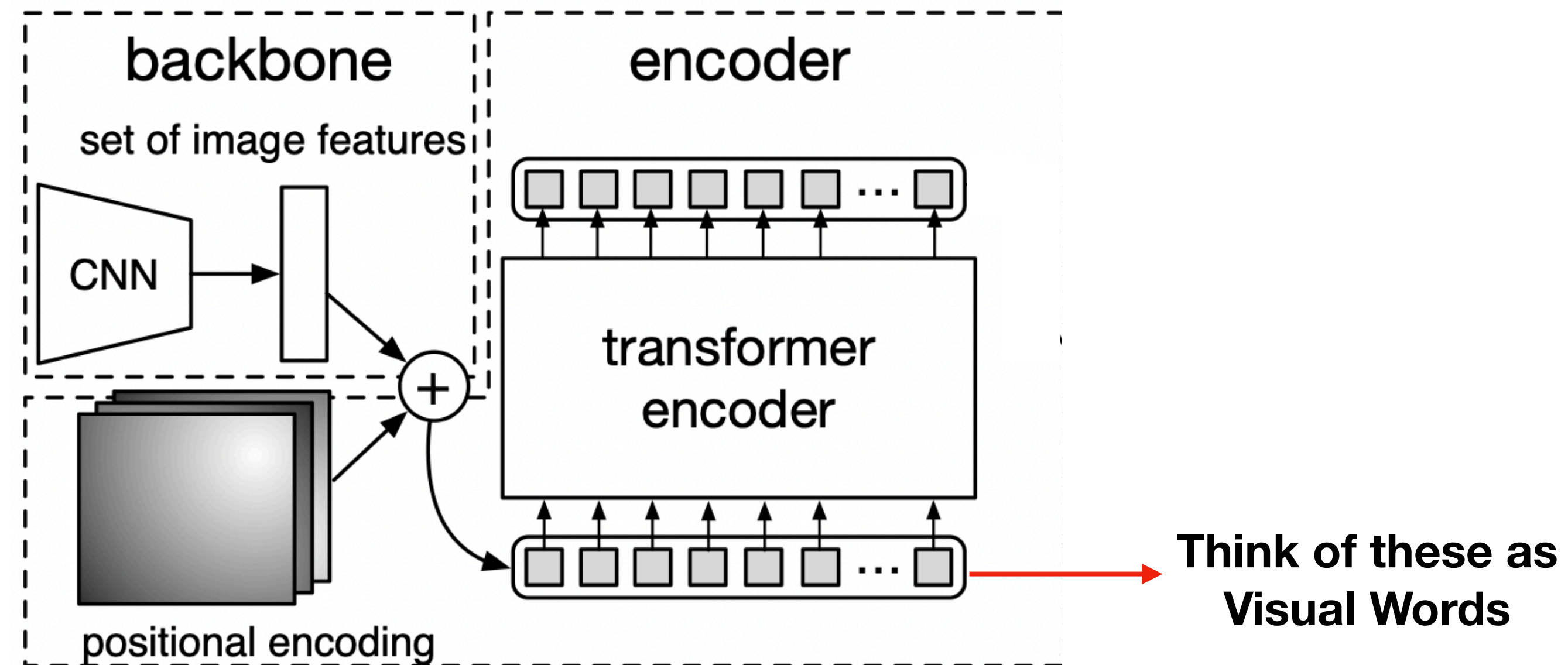
Transformers

$$\text{softmax}\left(\frac{\mathbf{Q} \times \mathbf{K}^T}{\sqrt{d_k}}\right) \mathbf{V}$$

= \mathbf{Z}

Aggregate Function

Content Adaptive Weights



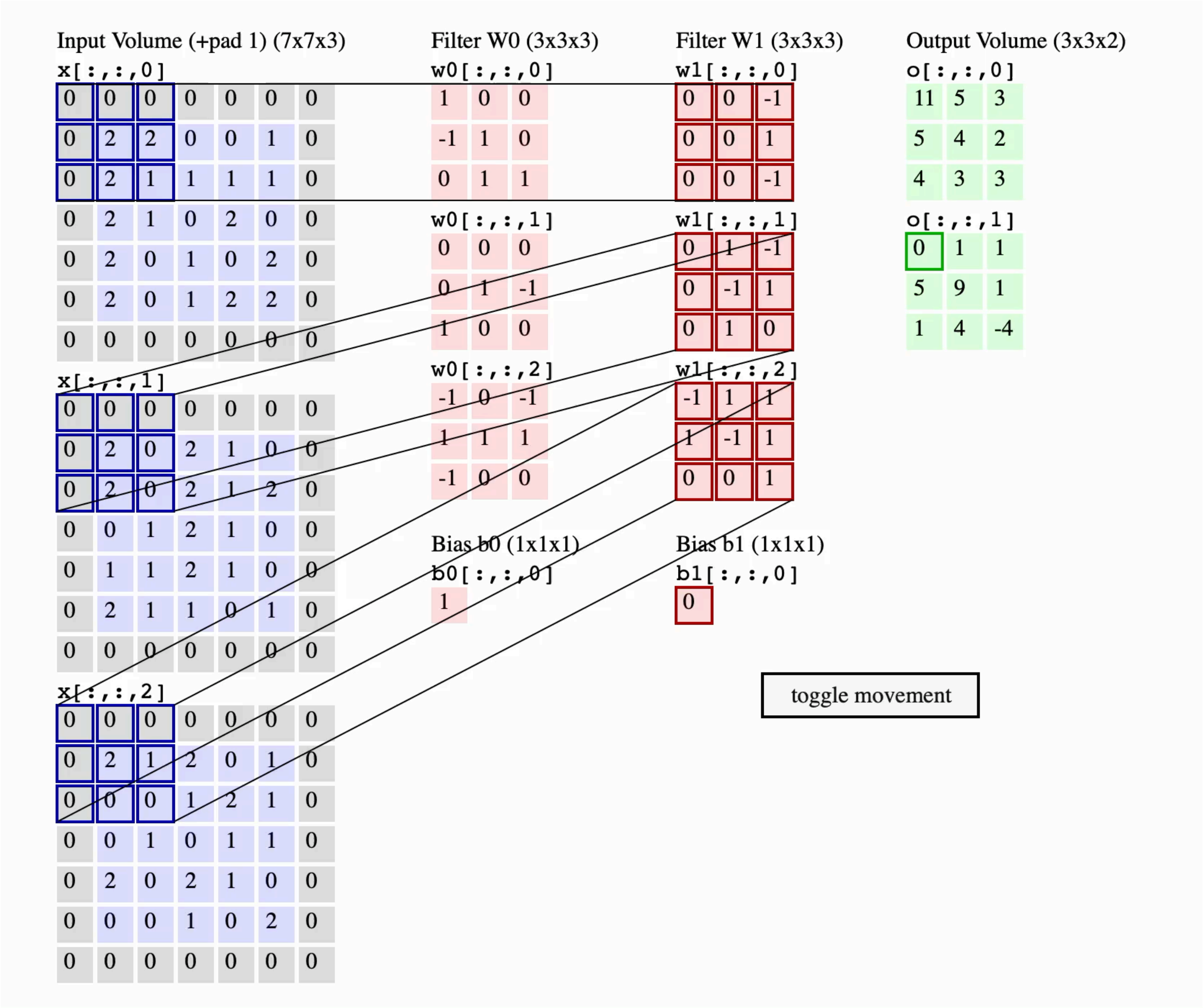
Tokens: Flattened CNN features

Self Attention: A New Perspective

Convolutional Networks

Aggregate Function

Local

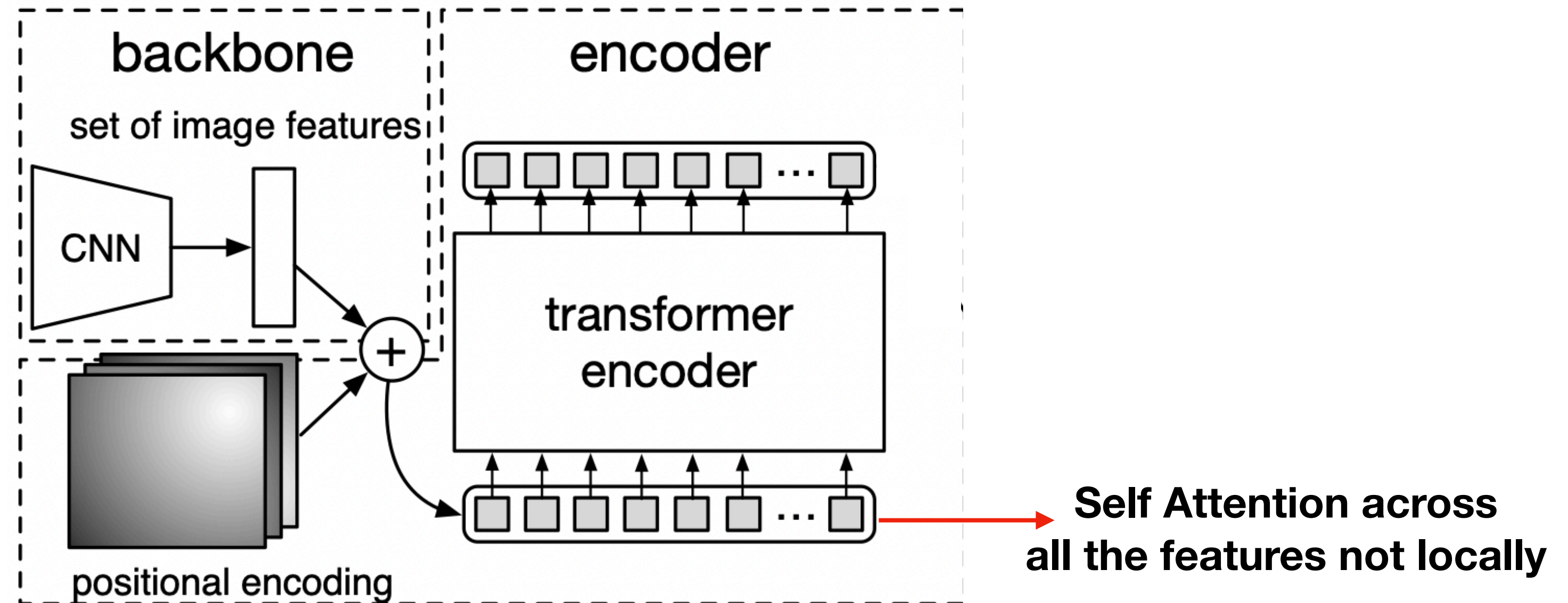


Self Attention: A New Perspective

Transformers

Aggregate Function

Global Context



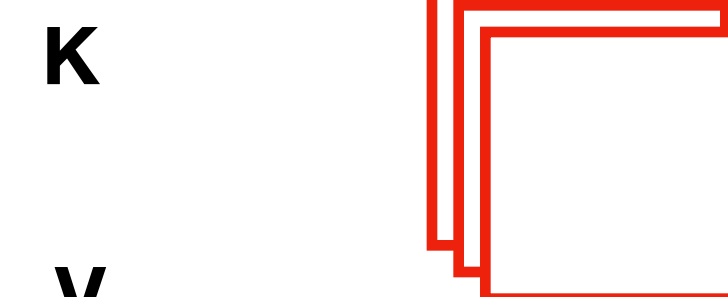
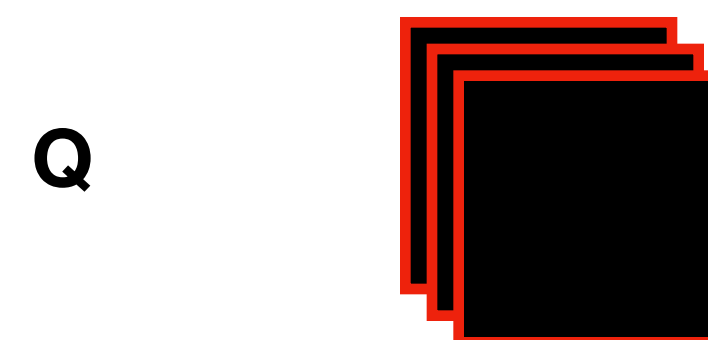
Tokens: Flattened CNN features

Self Attention vs Cross Attention

$$\text{softmax}\left(\frac{\overset{\text{Q}}{\begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \\ \hline \end{array}} \times \overset{\text{K}^T}{\begin{array}{|c|c|} \hline \square & \square \\ \hline \square & \square \\ \hline \square & \square \\ \hline \end{array}}}{\sqrt{d_k}}\right) \overset{\text{V}}{\begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \\ \hline \end{array}}$$

= $\overset{\text{Z}}{\begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \\ \hline \end{array}}$

MultiHeaded Attention in both

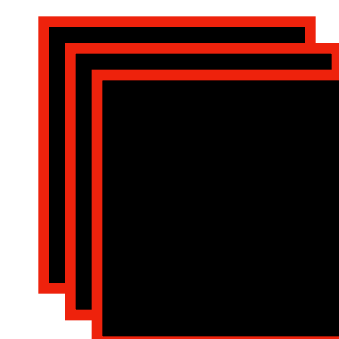


Cross
Attention

Q

K

v



Self
Attention

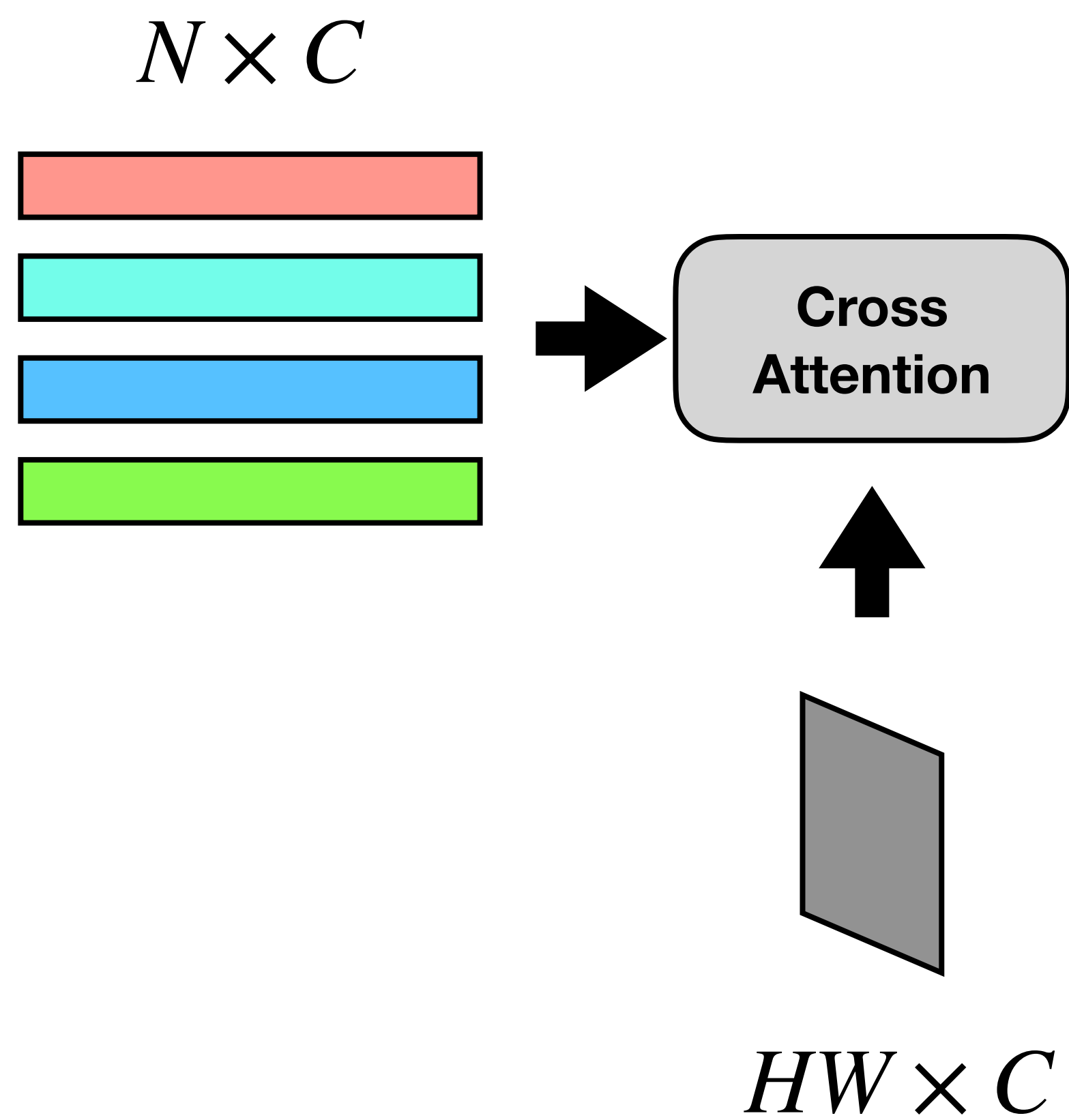
Cross Attention

$N \times C$

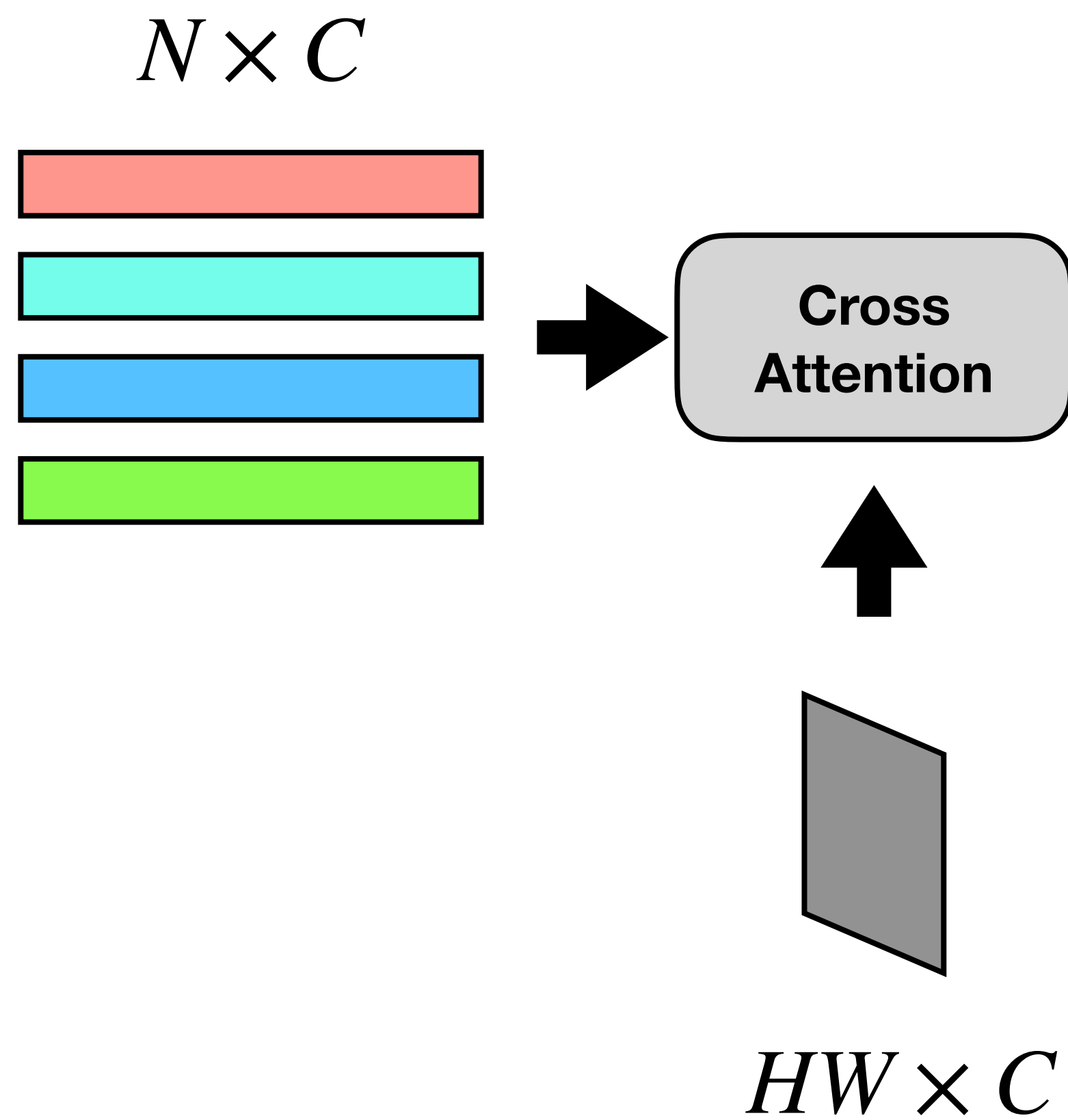


Learnable Queries

Cross Attention

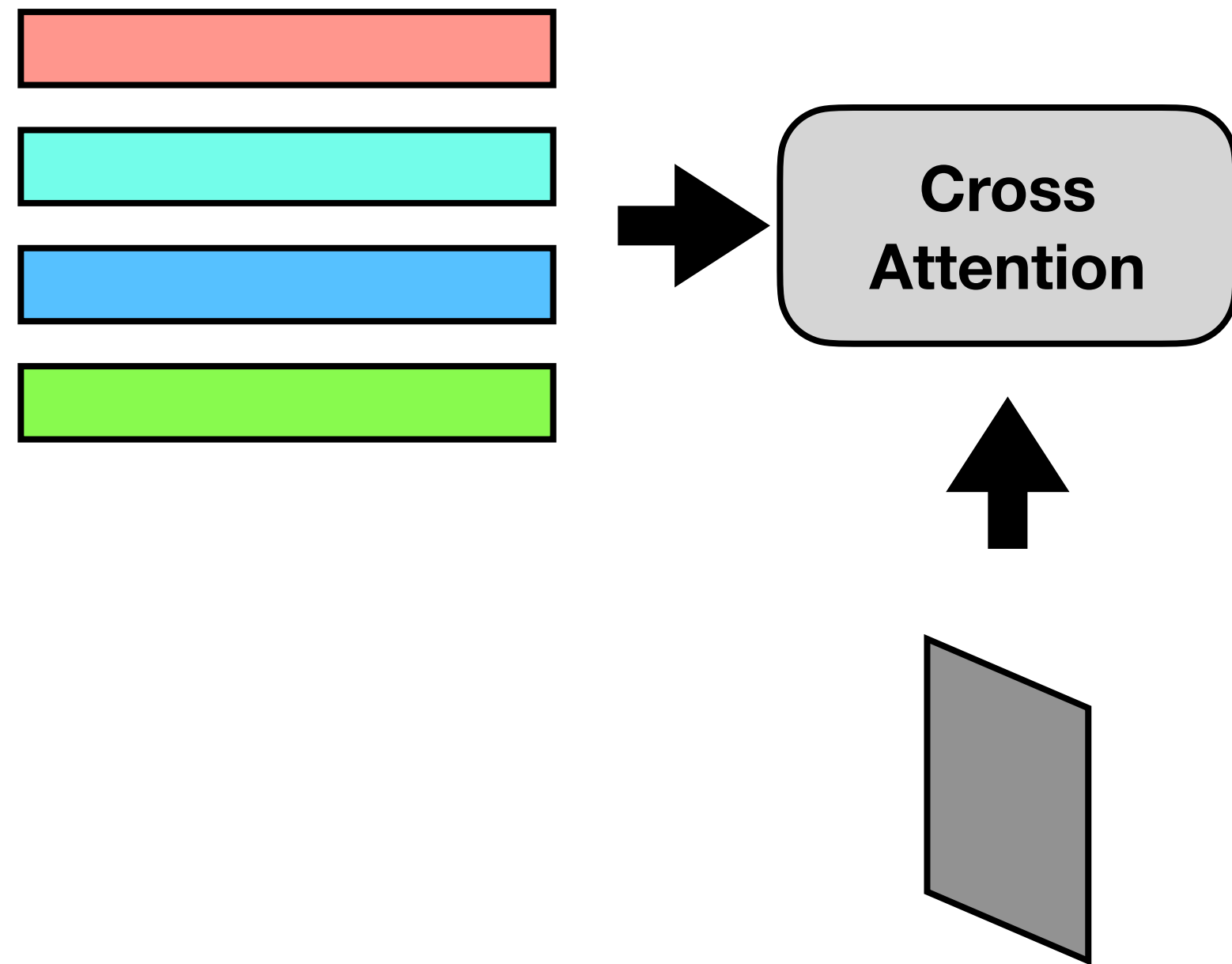


Cross Attention



$$\mathbf{X}_l = \text{softmax}(\mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

Cross Attention



$$\mathbf{X}_l = \text{softmax}(\mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

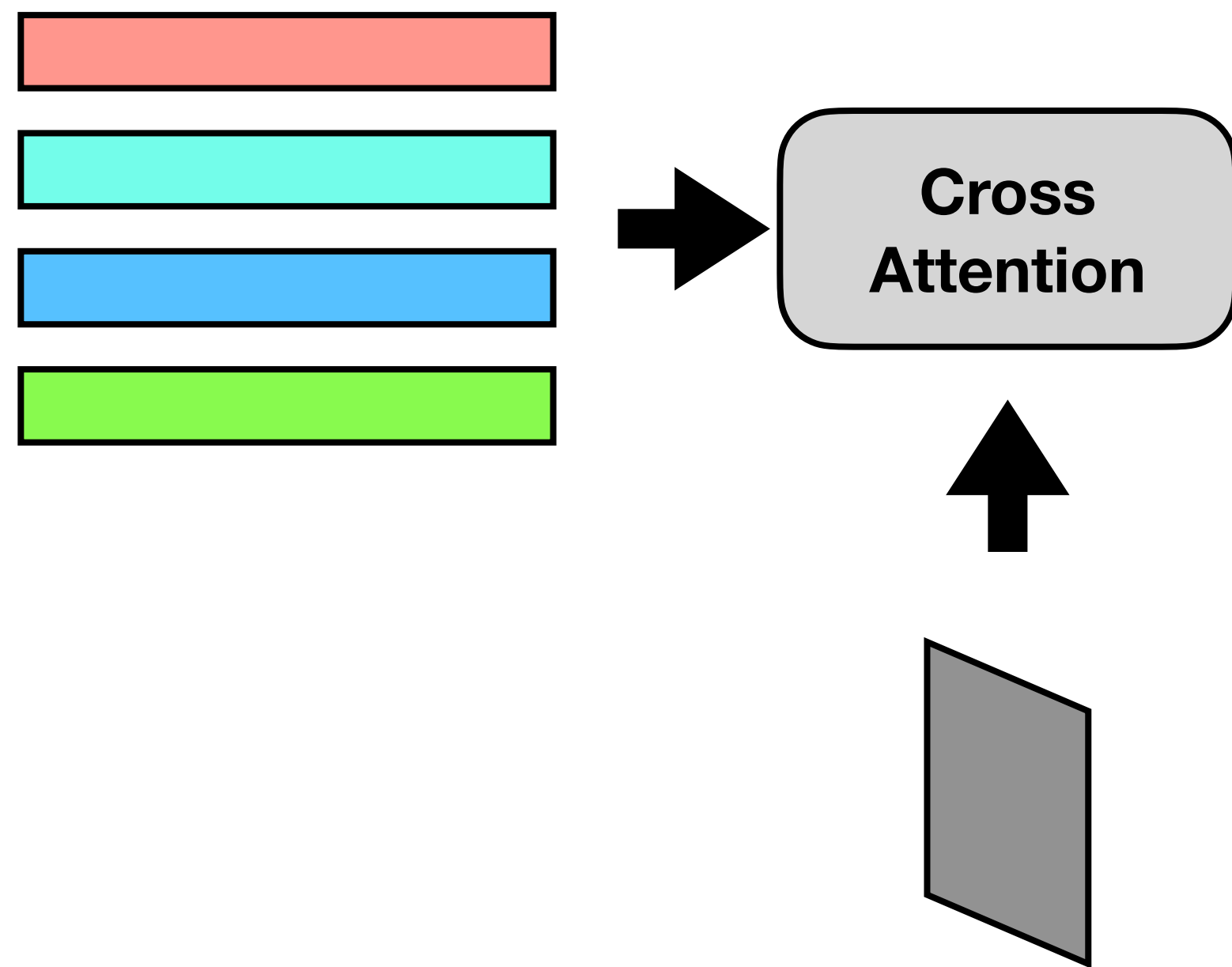
$$N \times C \quad C \times HW$$

$$N \times HW$$

Attention Maps

Relate

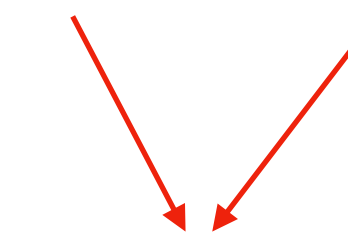
Cross Attention



$$\mathbf{X}_l = \text{softmax}(\mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

$$N \times HW \quad HW \times C$$

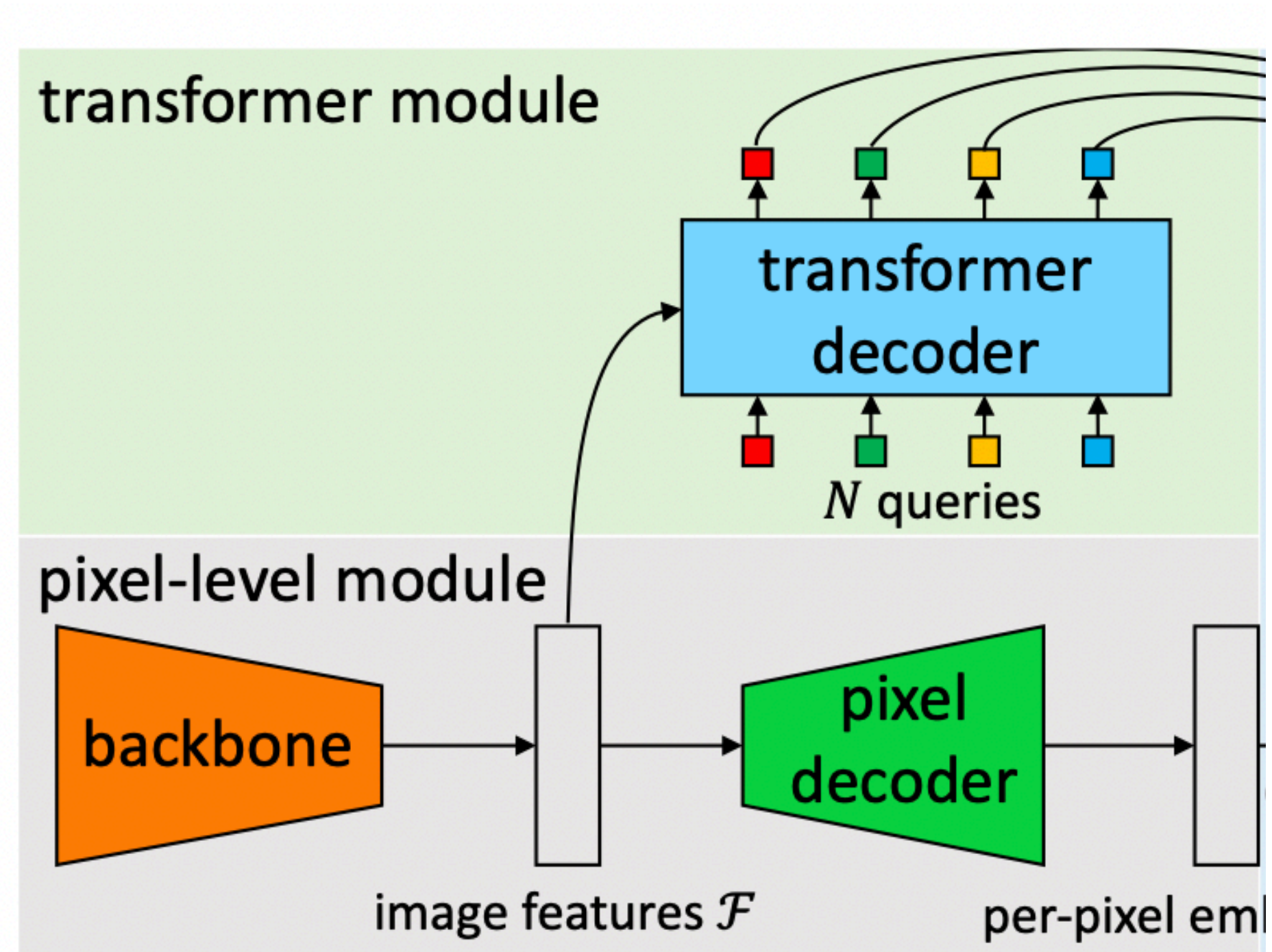
Attention Maps



$$N \times C$$

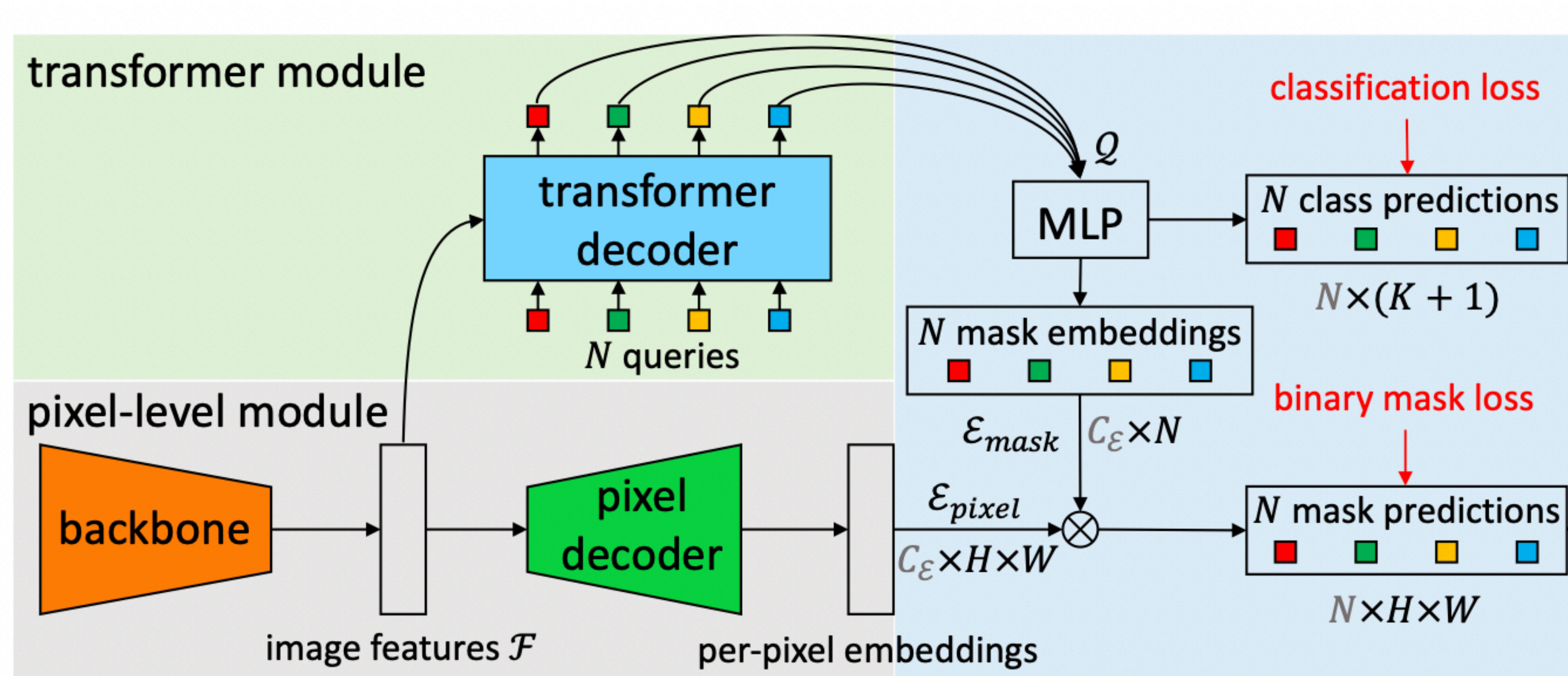
Aggregate

MaskFormer



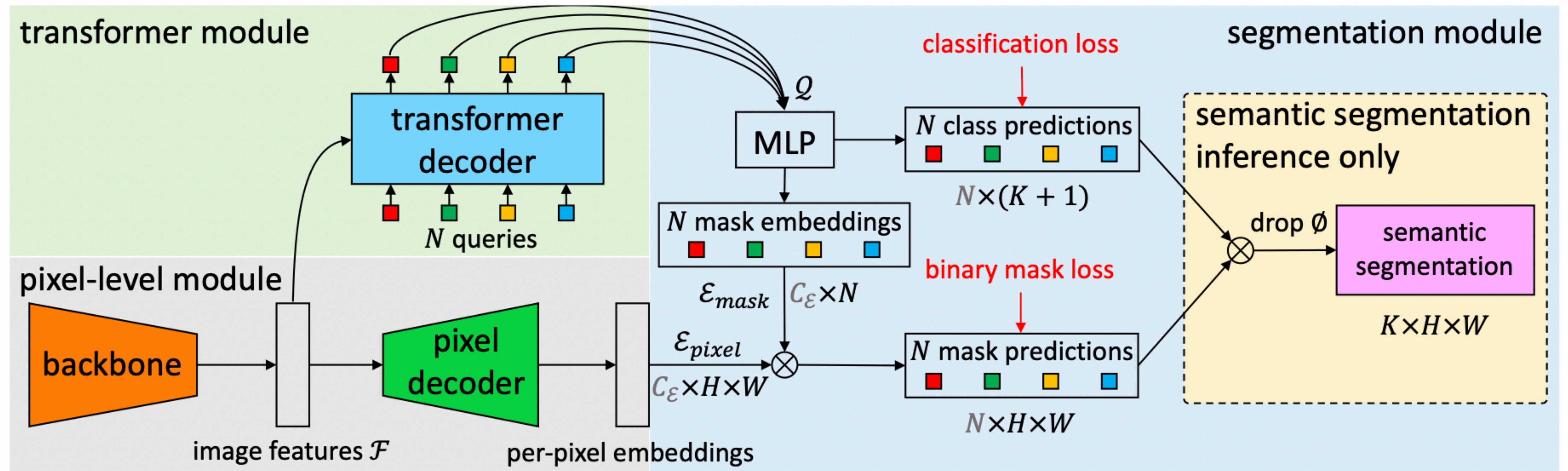
Cheng, Bowen, Alex Schwing, and Alexander Kirillov. "Per-pixel classification is not all you need for semantic segmentation." *Advances in Neural Information Processing Systems* 34 (2021): 17864-17875.

MaskFormer



Cheng, Bowen, Alex Schwing, and Alexander Kirillov. "Per-pixel classification is not all you need for semantic segmentation." *Advances in Neural Information Processing Systems* 34 (2021): 17864-17875.

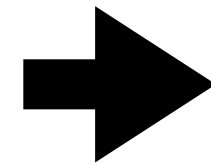
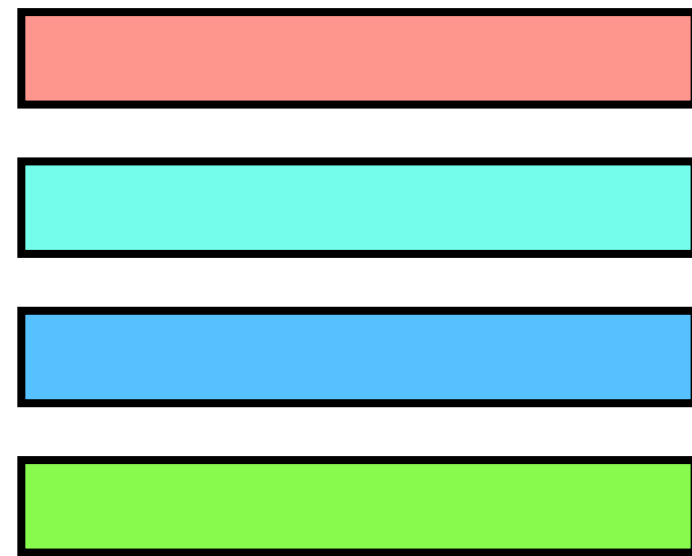
MaskFormer



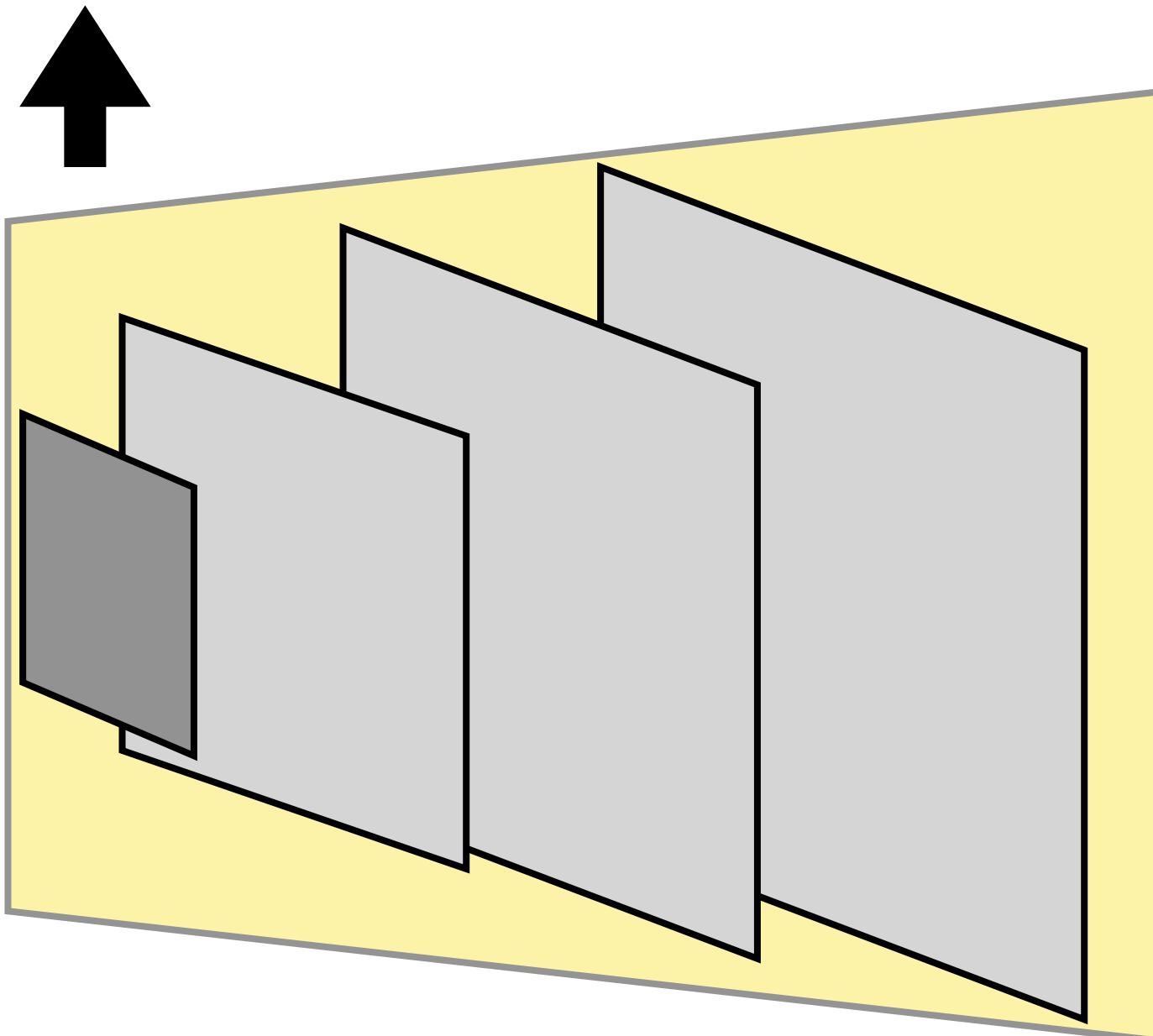
Mask2Former

Multiscale Transformer Decoder

Q



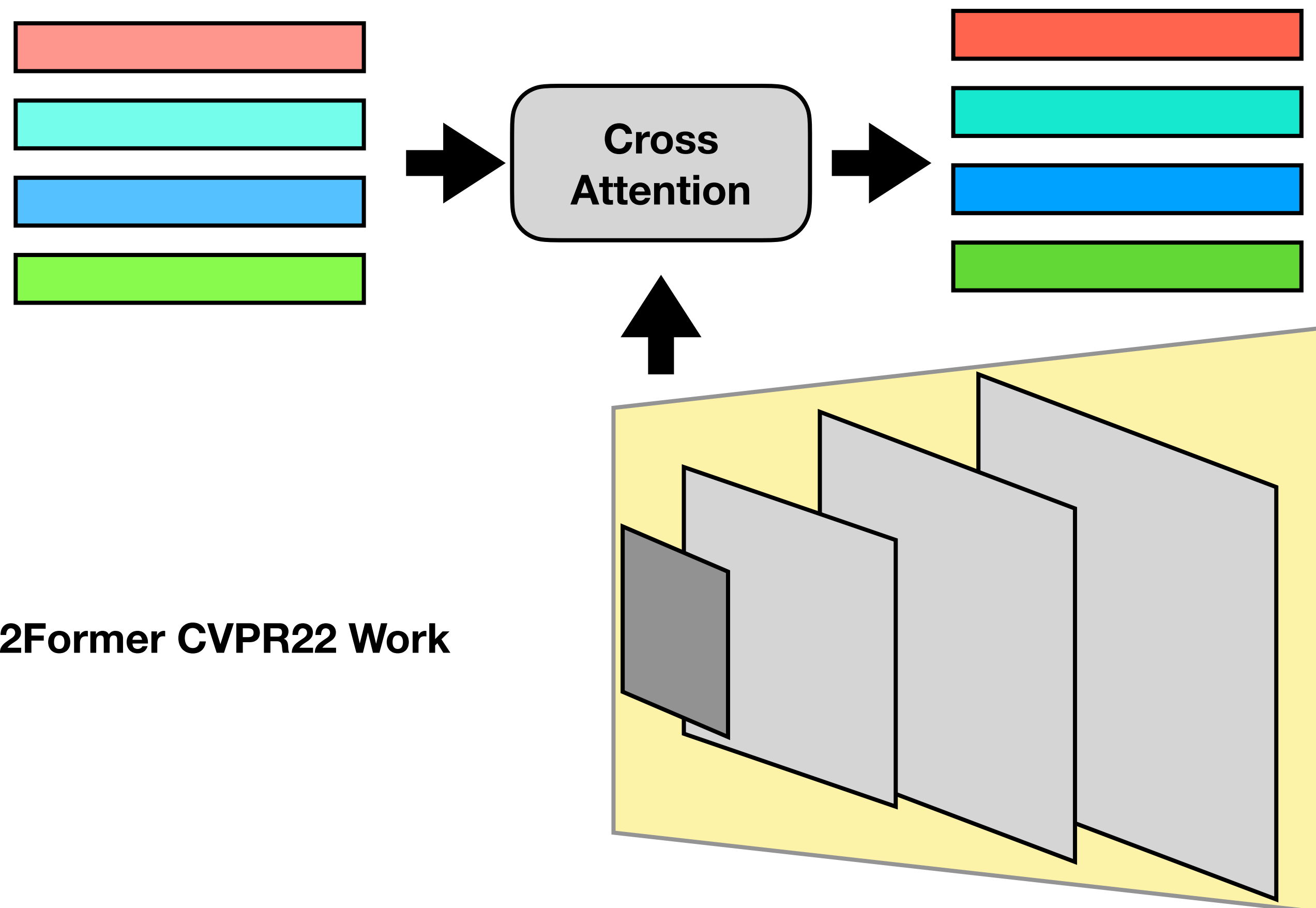
K, V



Mask2Former CVPR22 Work

Mask2Former

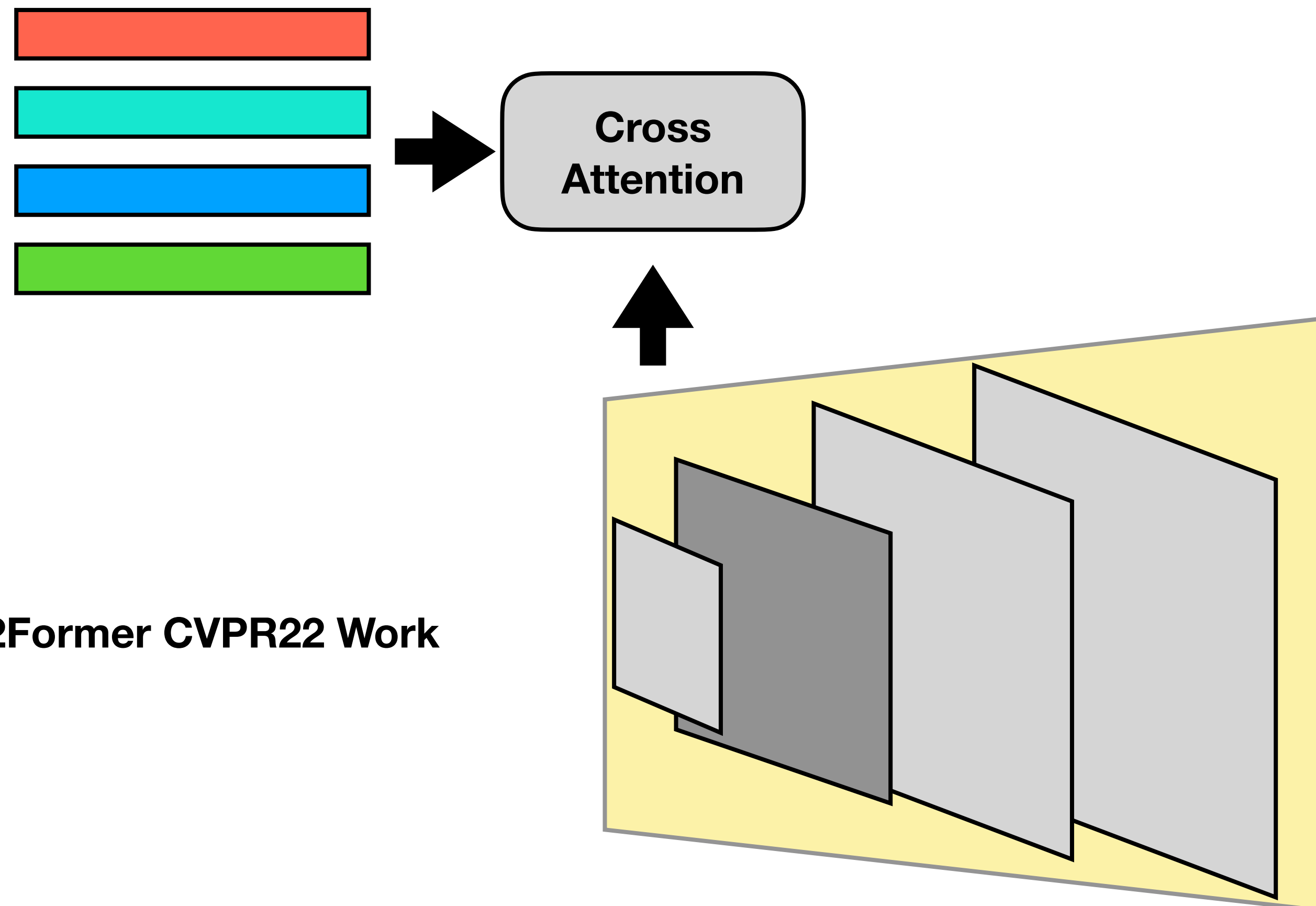
Multiscale Transformer Decoder



Mask2Former CVPR22 Work

Mask2Former

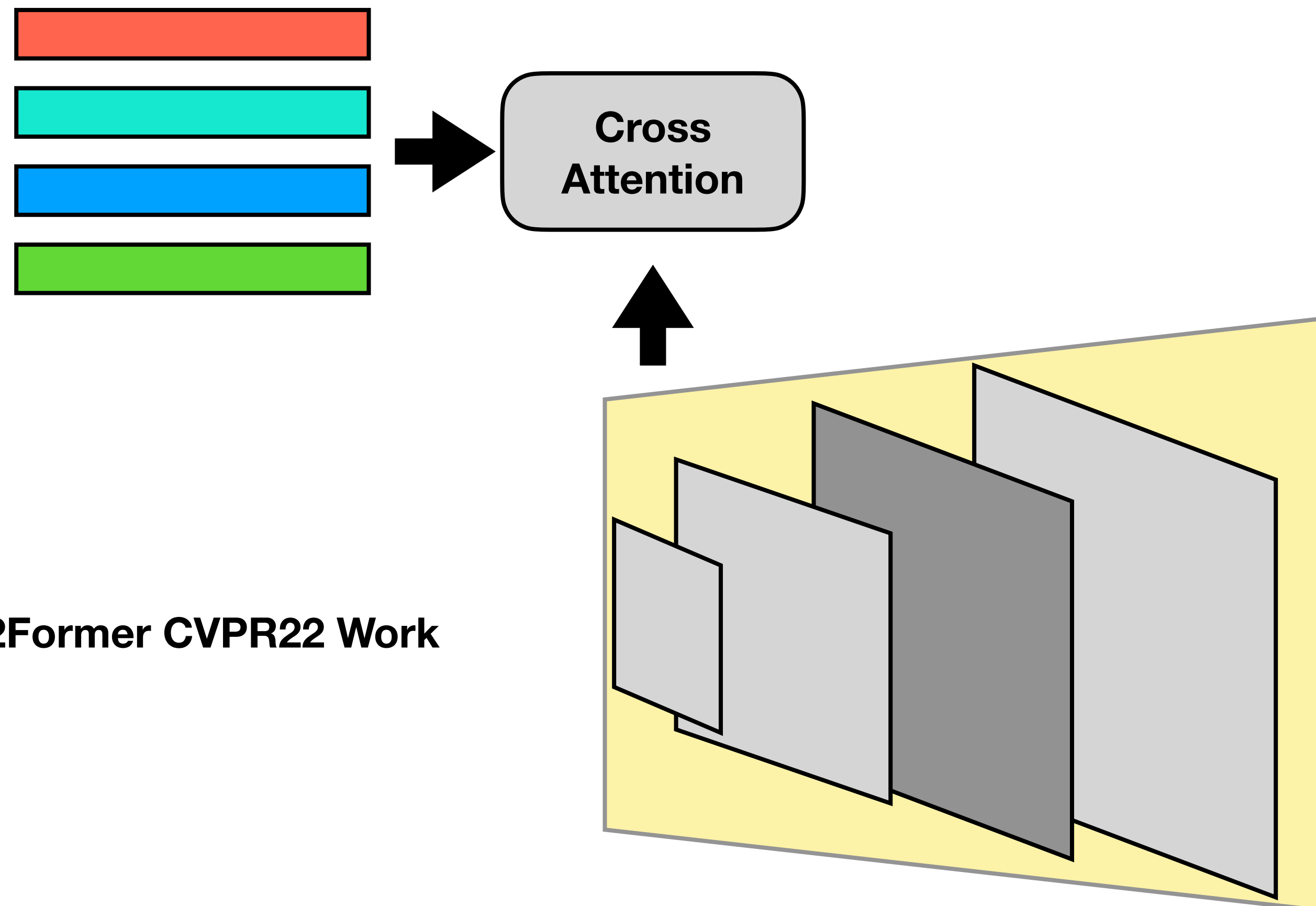
Multiscale Transformer Decoder



Mask2Former CVPR22 Work

Mask2Former

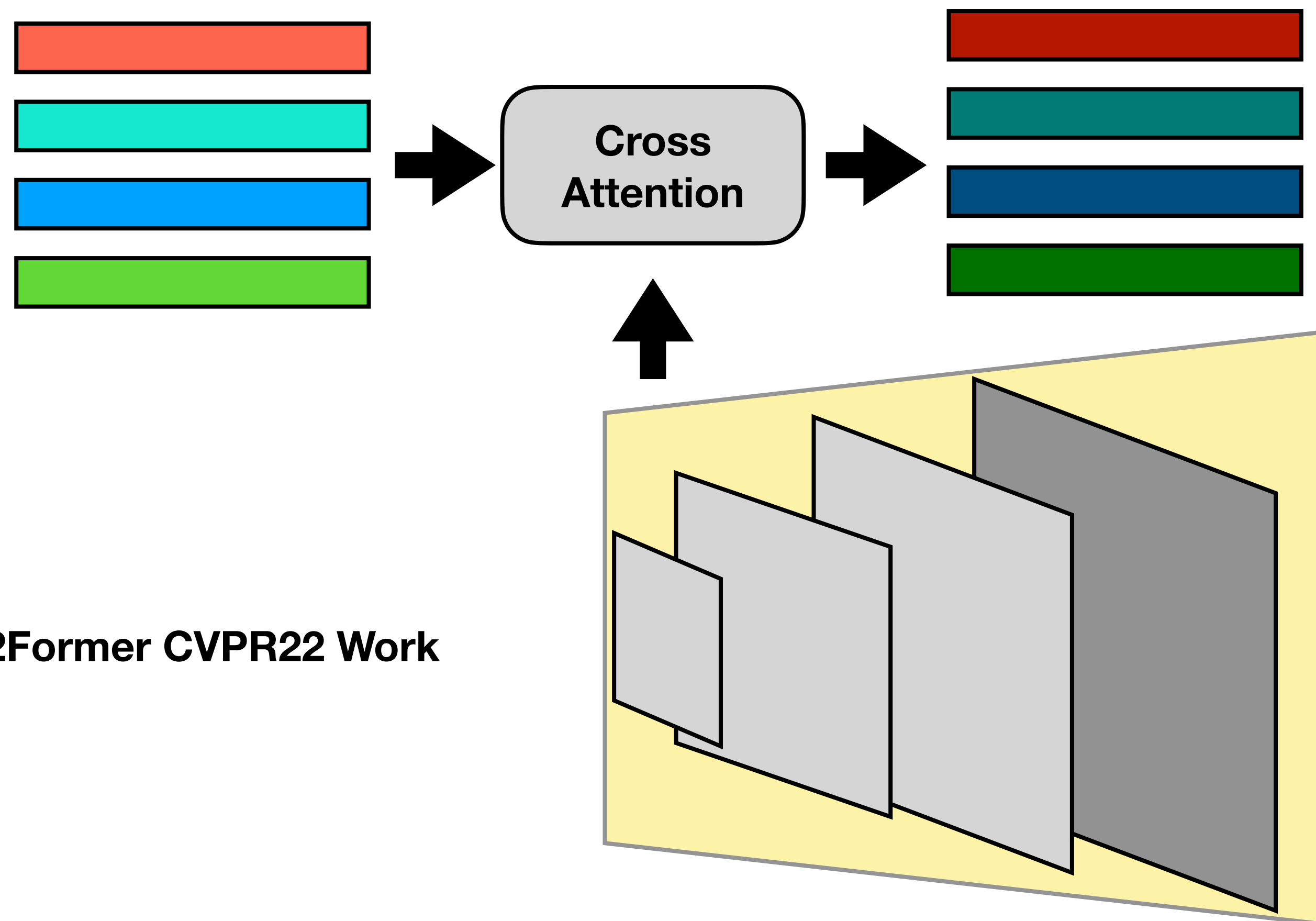
Multiscale Transformer Decoder



Mask2Former CVPR22 Work

Mask2Former

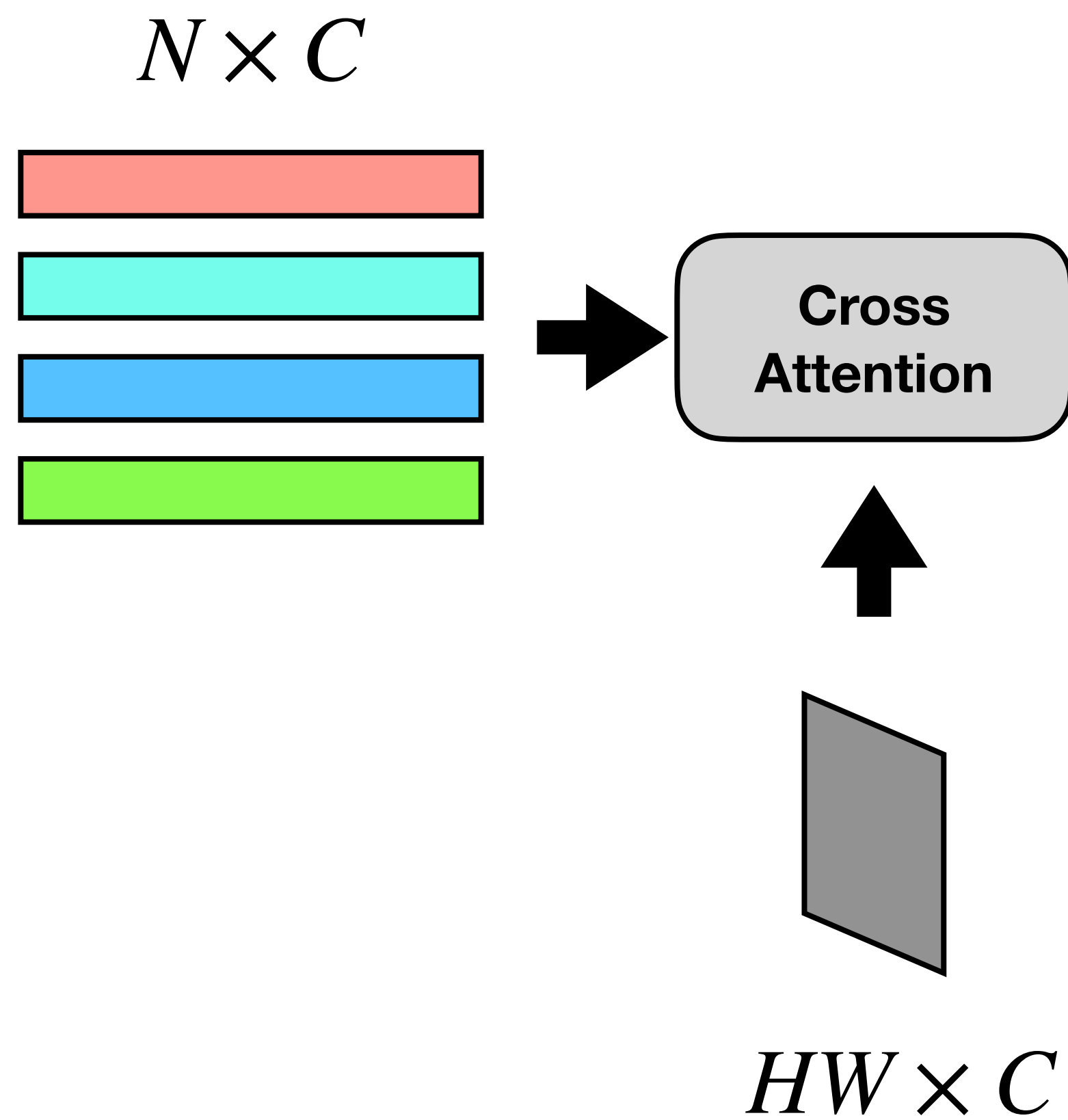
Multiscale Transformer Decoder



Mask2Former CVPR22 Work

Mask2Former

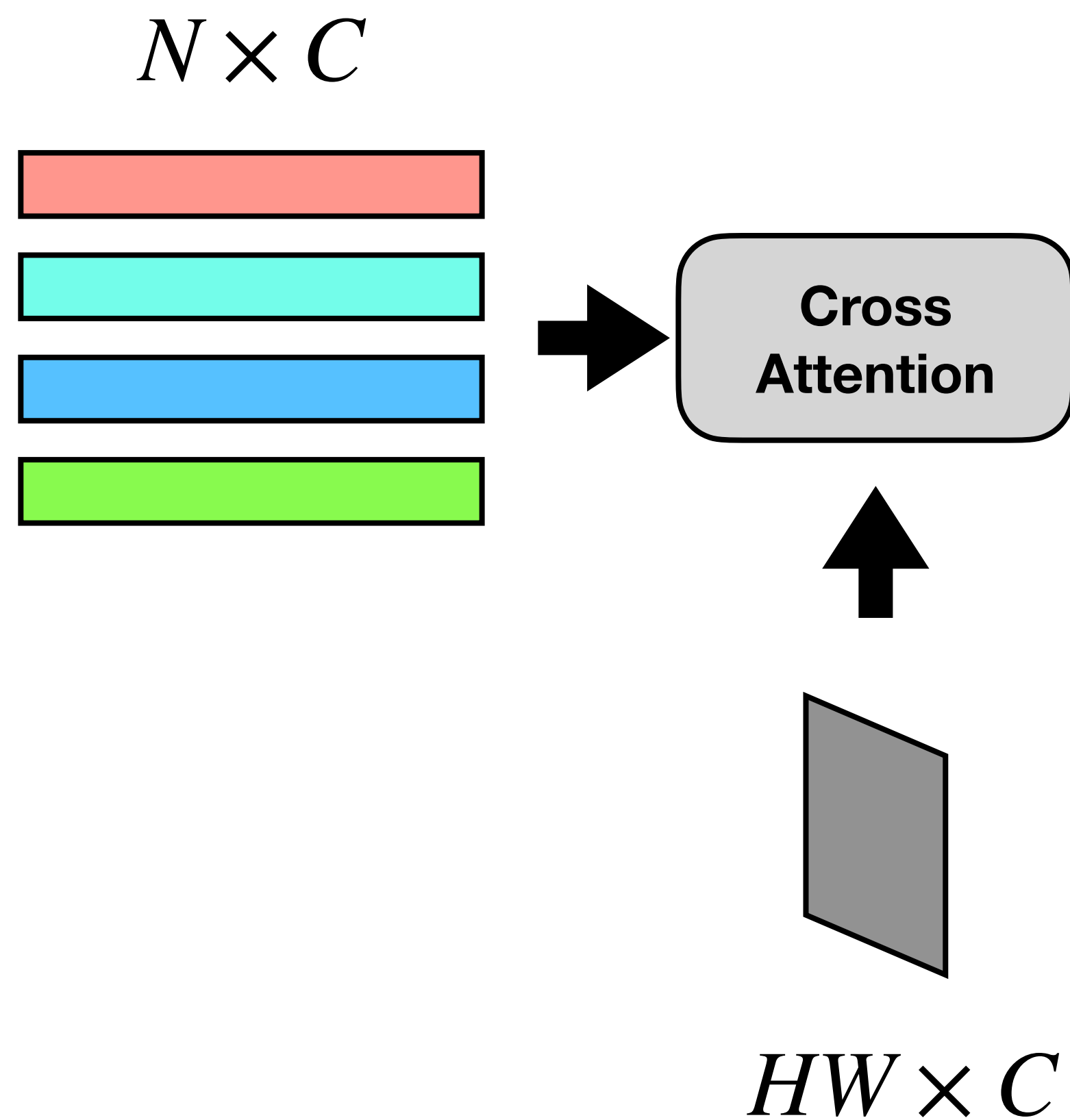
Masked Attention



$$\mathbf{X}_l = \text{softmax}(\mathcal{M}_{l-1} + \mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

Mask2Former

Masked Attention

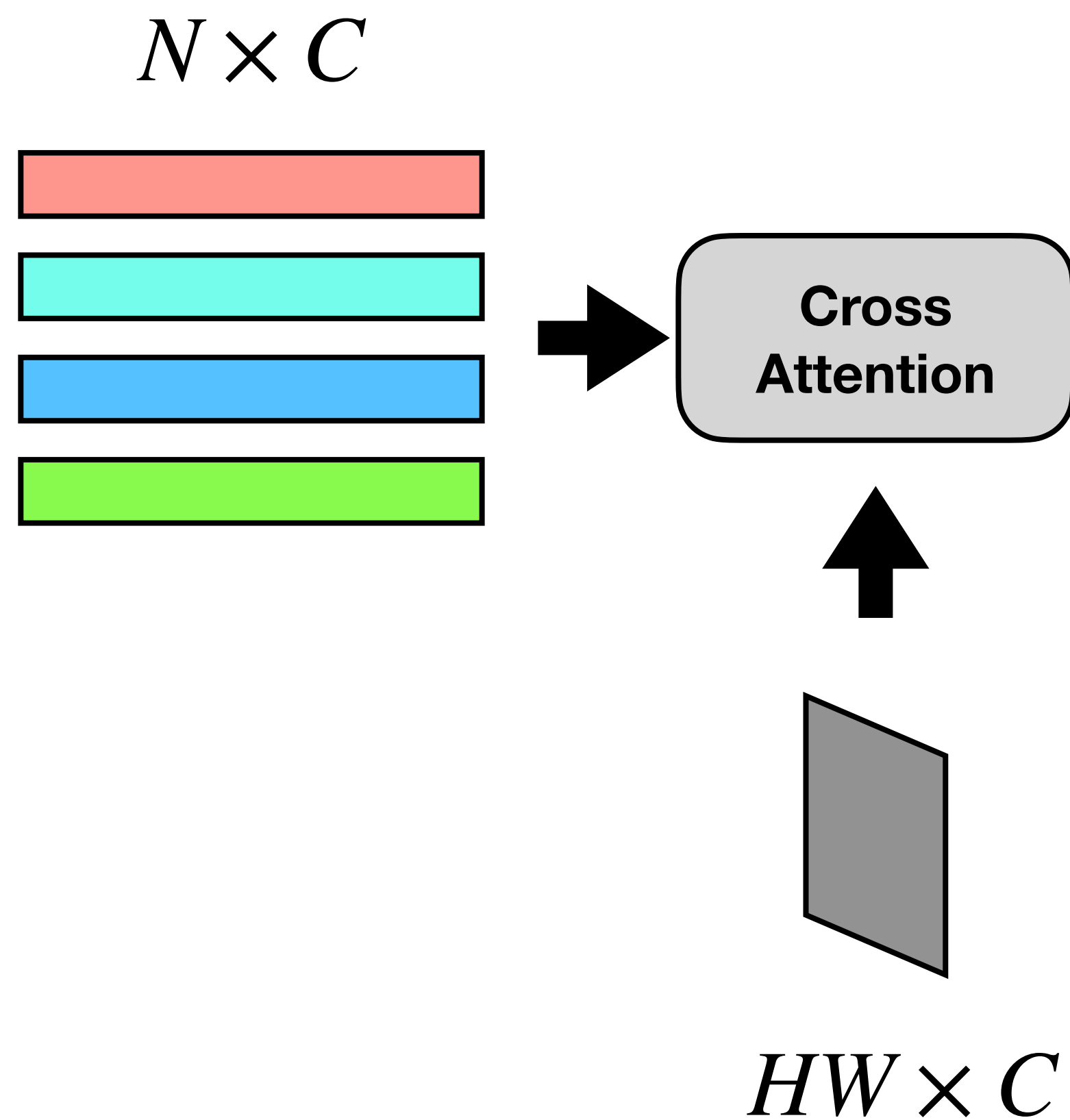


$$\mathbf{X}_l = \text{softmax}(\mathcal{M}_{l-1} + \mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

$$\mathcal{M}_{l-1}(x, y) = \begin{cases} 0 & \text{if } \mathbf{M}_{l-1}(x, y) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

Mask2Former

Masked Attention



$$\mathbf{M}_{l-1} \quad N \times HW$$

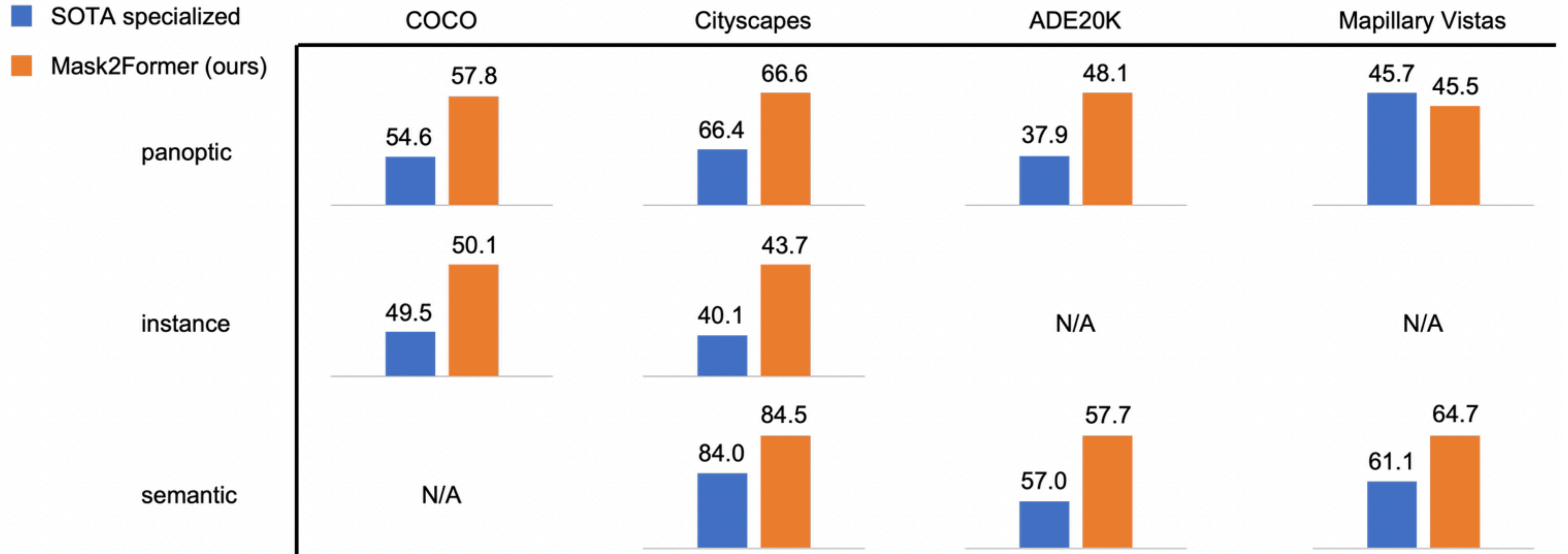
Binary Mask Predictions $\{0, 1\}$

$$\mathbf{X}_l = \text{softmax}(\mathcal{M}_{l-1} + \mathbf{Q}_l \mathbf{K}_l^T) \mathbf{V}_l + \mathbf{X}_{l-1}.$$

$$\mathcal{M}_{l-1}(x, y) = \begin{cases} 0 & \text{if } \mathbf{M}_{l-1}(x, y) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

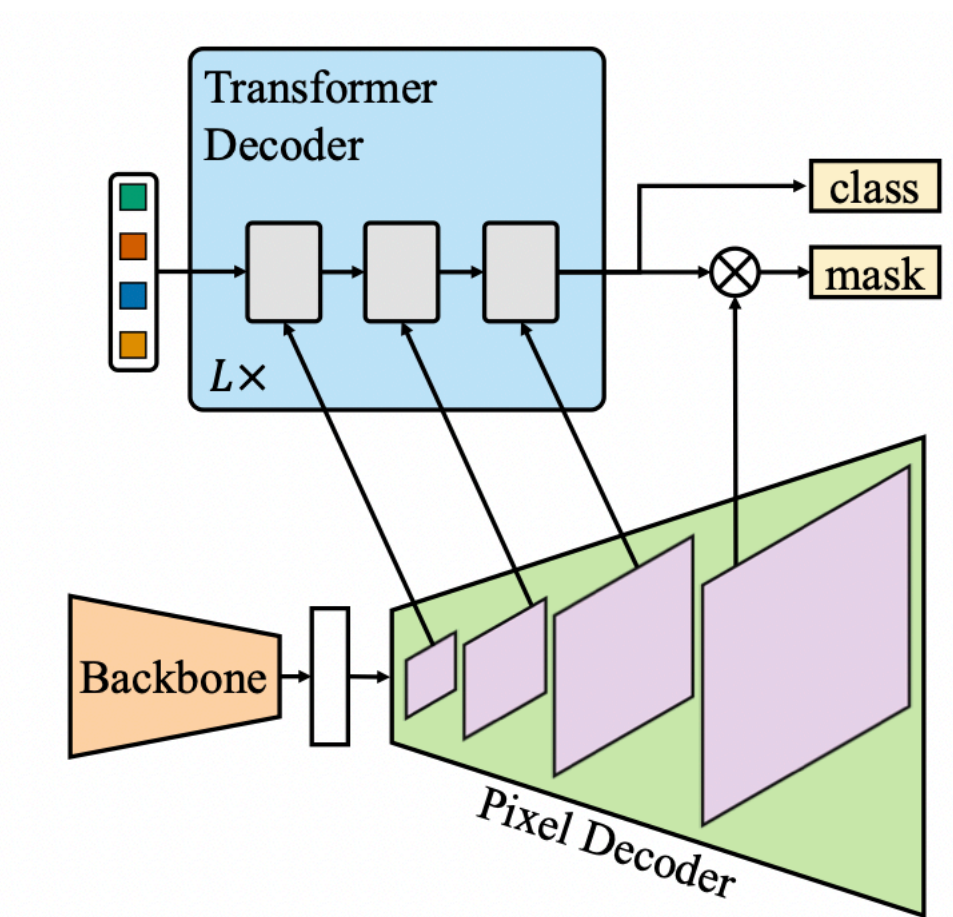
Mask2Former

Masked Cross Attention

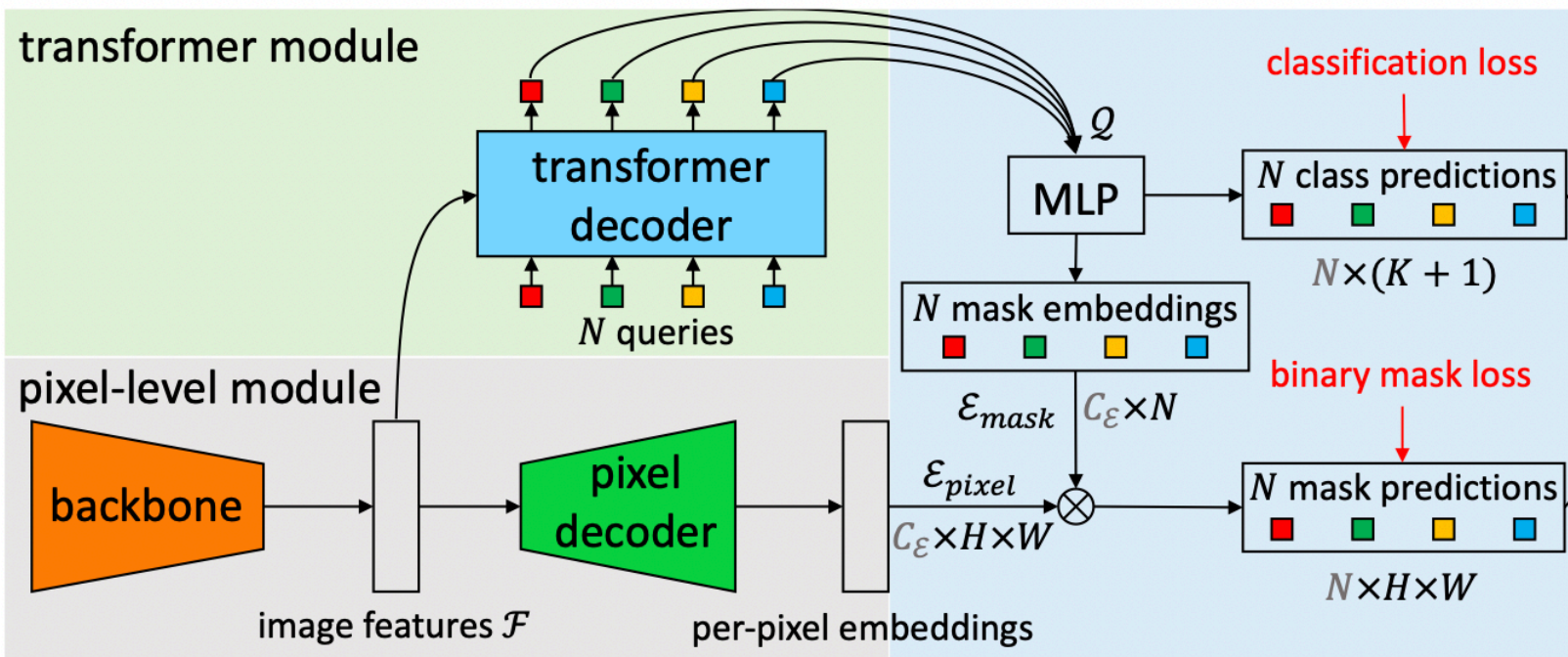


Questions

Mask2Former, CVPR'22



MaskFormer, NeurIPS'21



MED-VT, CVPR'23

