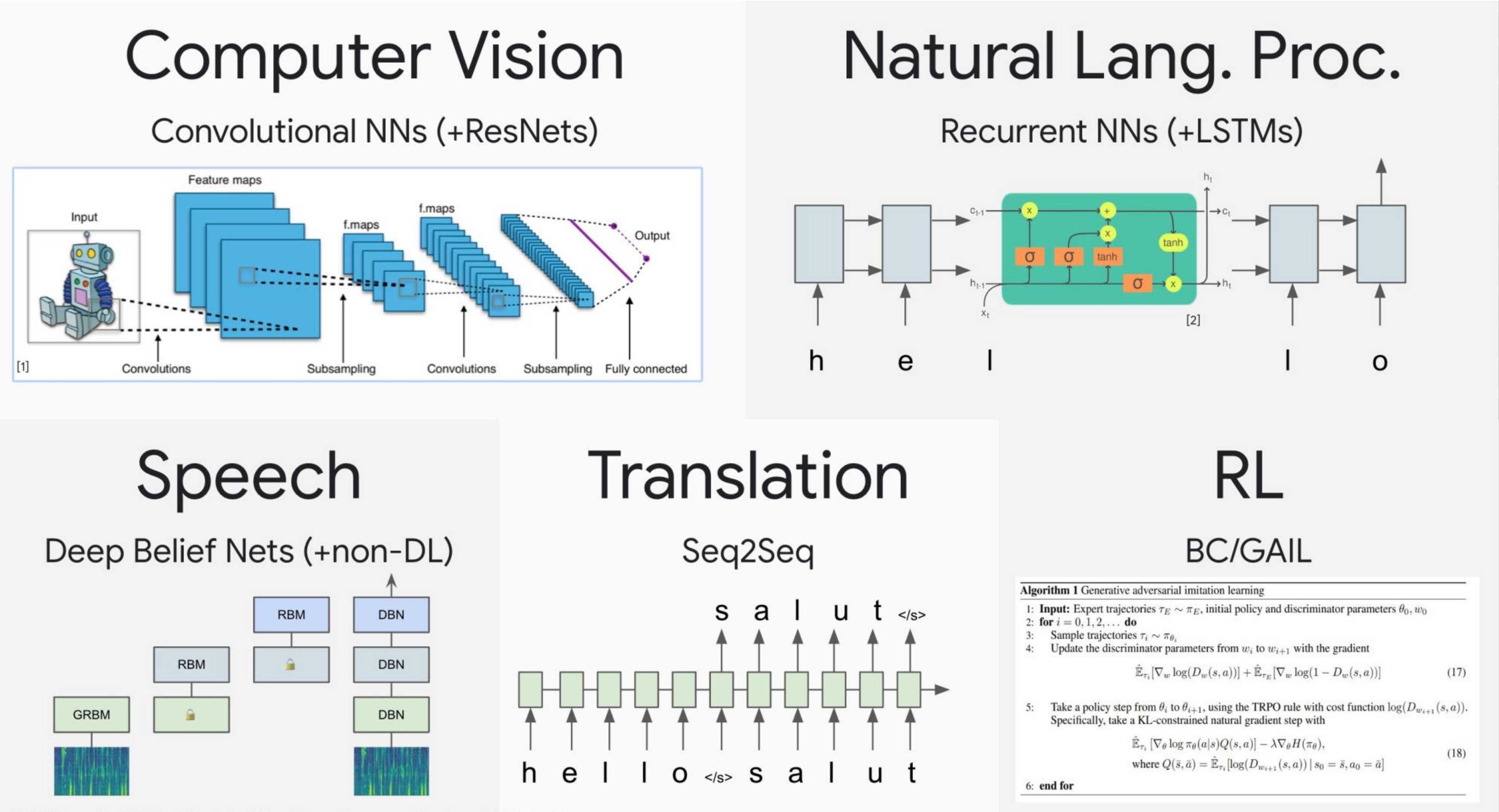
Transformers

- Asst Prof
- Dr Usman Zia
- usman.zia@sines.nust.edu.pk

Before ~2020: each task had its own NN architecture

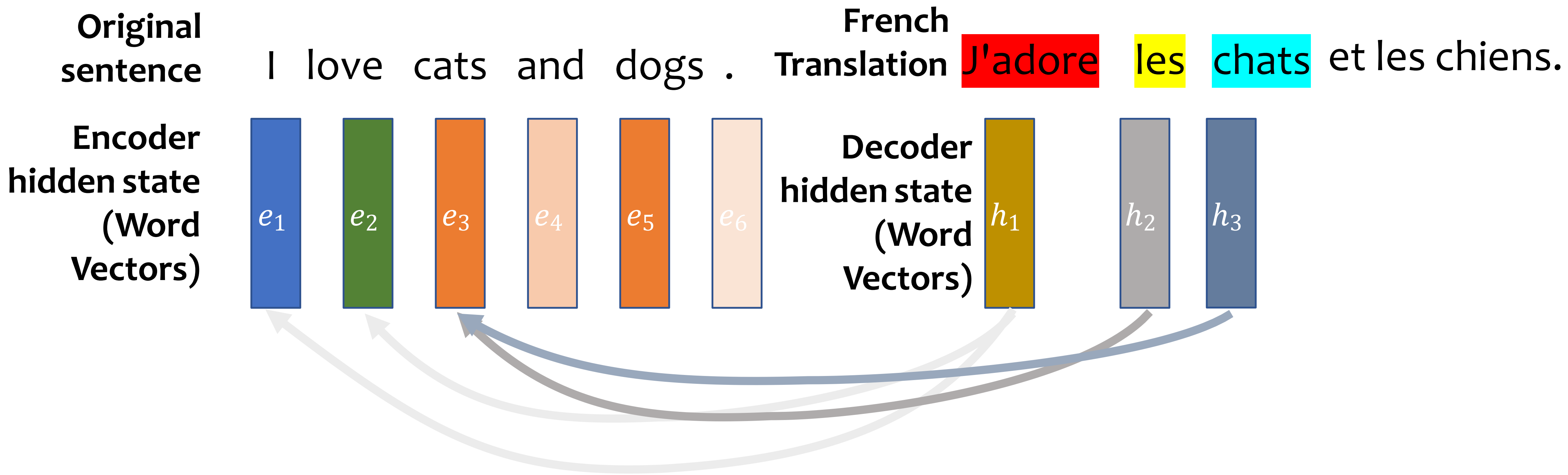


[1] CNN image CC-BY-SA by Aphex34 for Wikipedia https://commons.wikimedia.org/wiki/File:Typical_cnn.png
[2] RNN image CC-BY-SA by GChe for Wikipedia https://commons.wikimedia.org/wiki/File:The_LSTM_Cell.svg

Now: all is Transformers

Transformer cartoon (DALL-E)

Origin of Attention: Machine Translation (Seq2Seq)



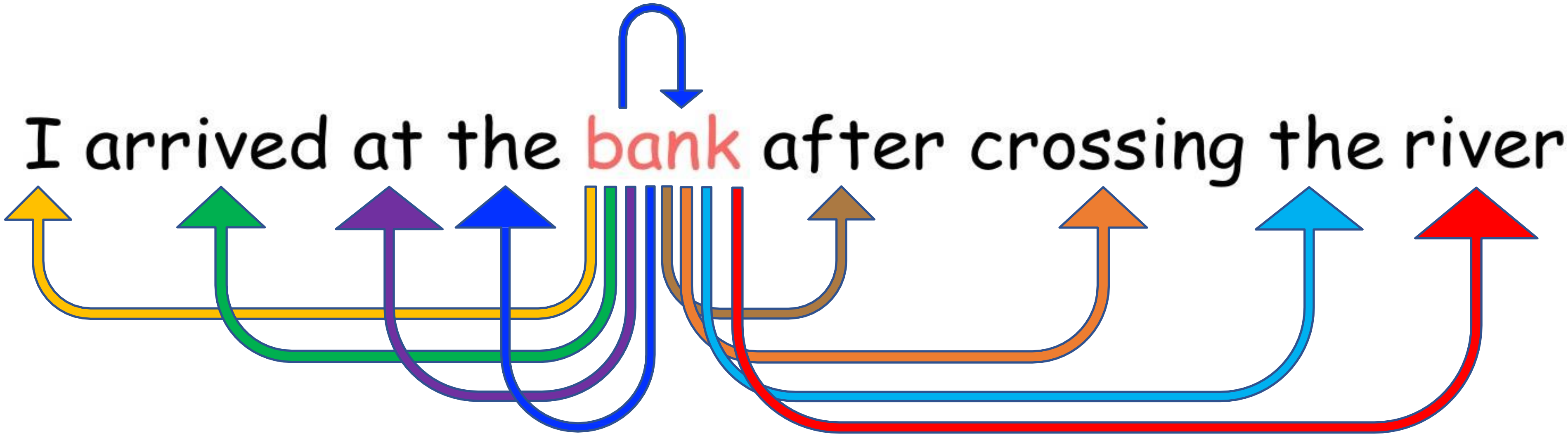
- Use **Attention** to retrieve **relevant info** from a batch of vectors.

How to retrieve relevant information?

From dictionary to feature based attention.

Transformer Key Idea: Self-Attention

New representation of each token in a sequence showing its relationship to all tokens; e.g.,



Transformer Intuition

What does bank mean in this sentence?

I arrived at the bank after crossing the …

Transformer Intuition

What does bank mean in this sentence?
- the new representation of the word disambiguates the meaning by identifying other relevant words (e.g., high attention score with “river”)

I arrived at the bank after crossing the river

vs

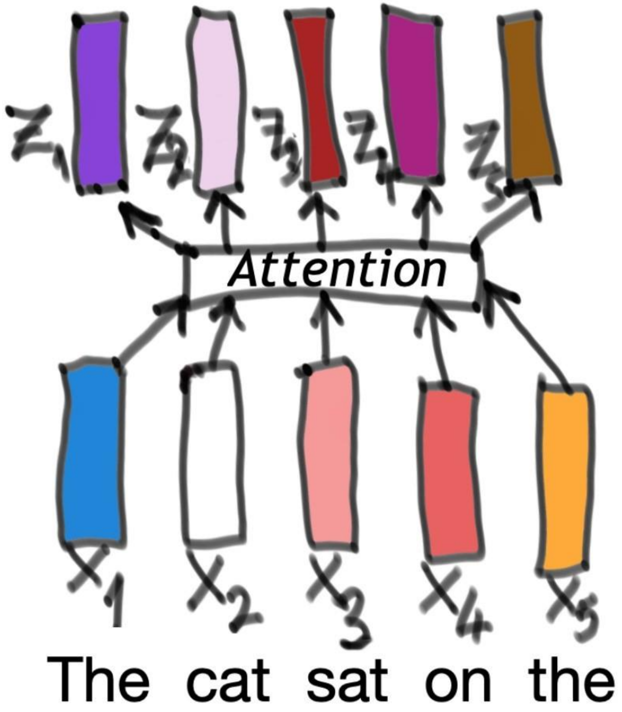
I arrived at the bank after crossing the street

Transformer: A Suggested Definition

"Any architecture designed to process a connected set of units—such as the tokens in a sequence or the pixels in an image—where the only interaction between units is through self-attention."

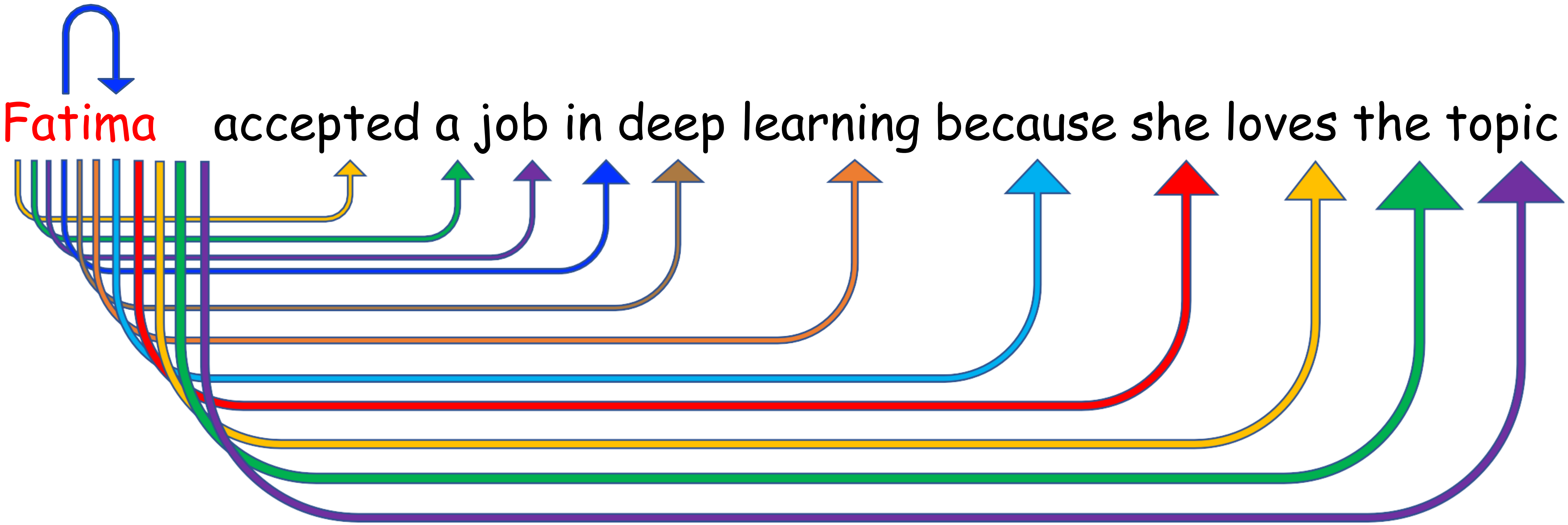
Self-Attention: Outcome

New representation of each token in a sequence showing its relationship to all tokens



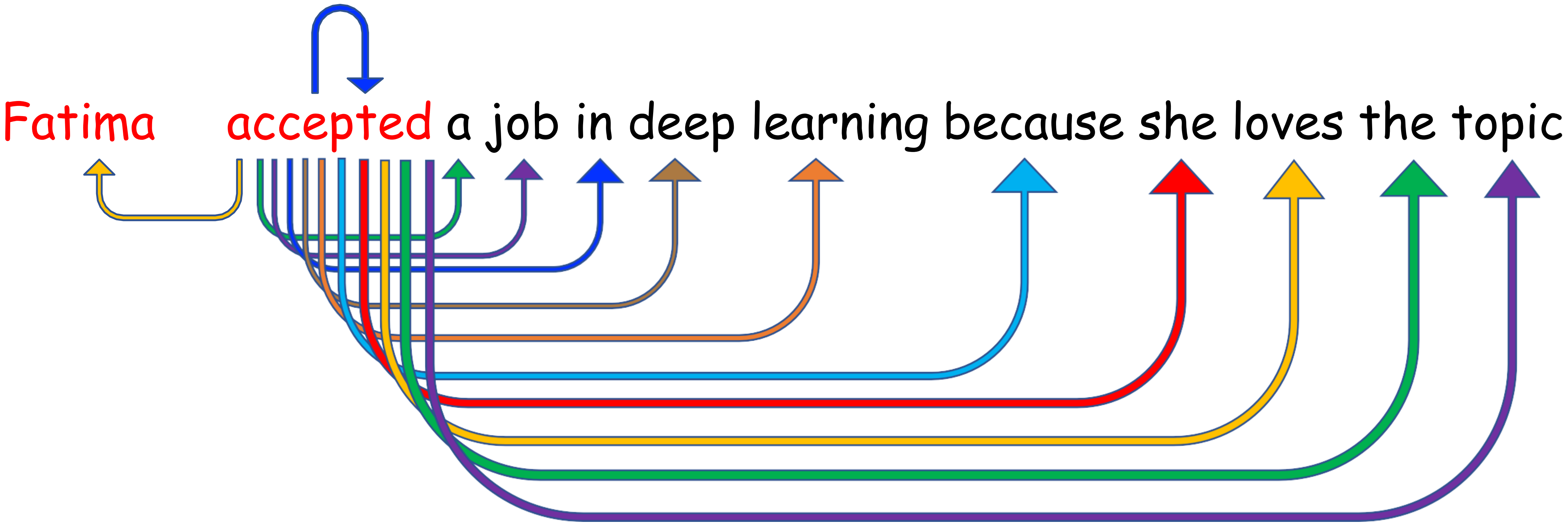
Self-Attention: Outcome

New representation of each token in a sequence showing its relationship to all tokens; e.g.,



Self-Attention: Outcome

New representation of each token in a sequence showing its relationship to all tokens; e.g.,



Self-Attention: Outcome

New representation of each token in a sequence showing its relationship to all tokens; e.g.,

Fatima accepted a job in deep learning because she loves the topic

And so on for remaining words…

Self-Attention: Disambiguates Word Meanings

New representation of each token in a sequence showing its relationship to all tokens; e.g.,

Fatima accepted a job in deep learning because she loves the topic

A better representation of "she" would encode information about "Rashonda"

Self-Attention: Disambiguates Word Meanings

New representation of each token in a sequence showing its relationship to all tokens; e.g.,

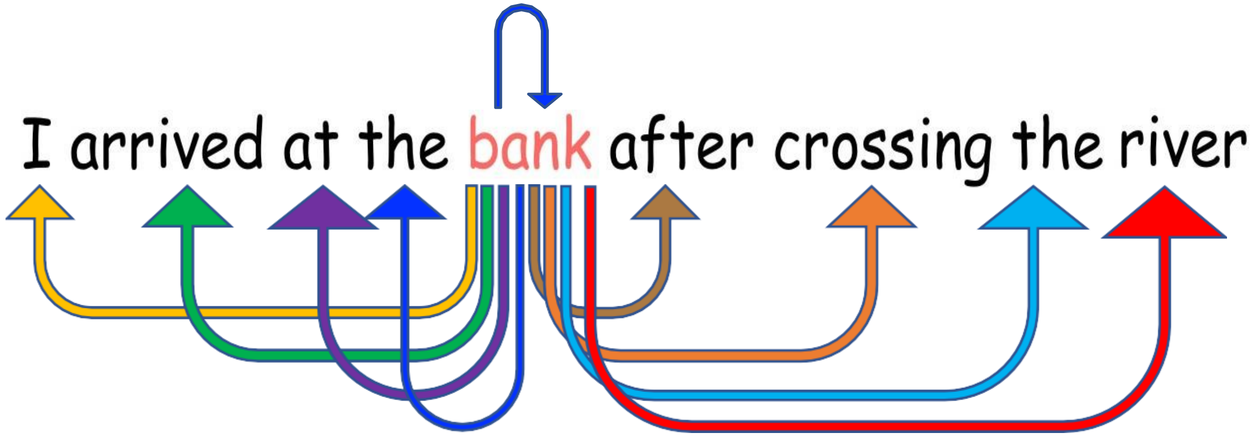
I arrived at the bank across the river

A better representation of “bank” would encode information about “river”

Self-Attention vs General Attention

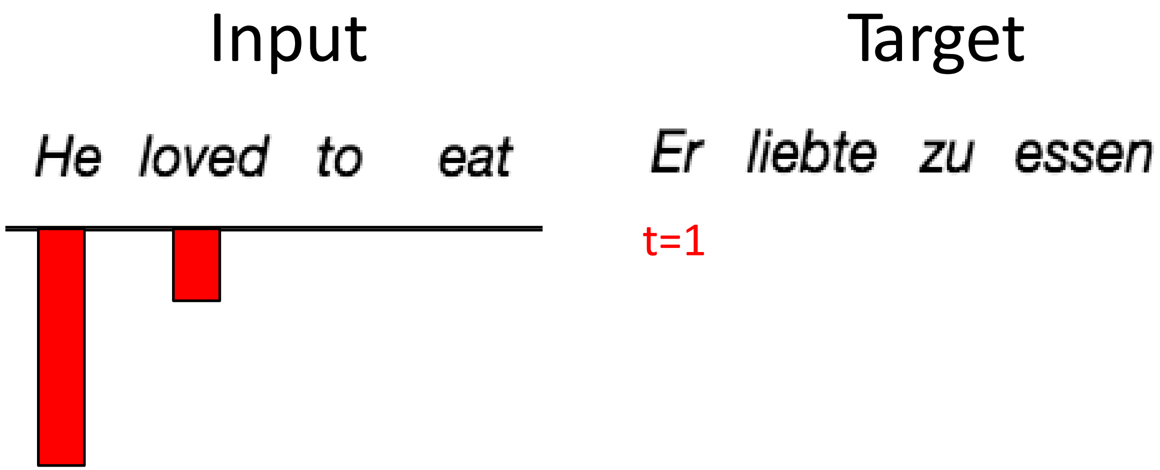
Self-attention

Relates tokens from the same source



General attention

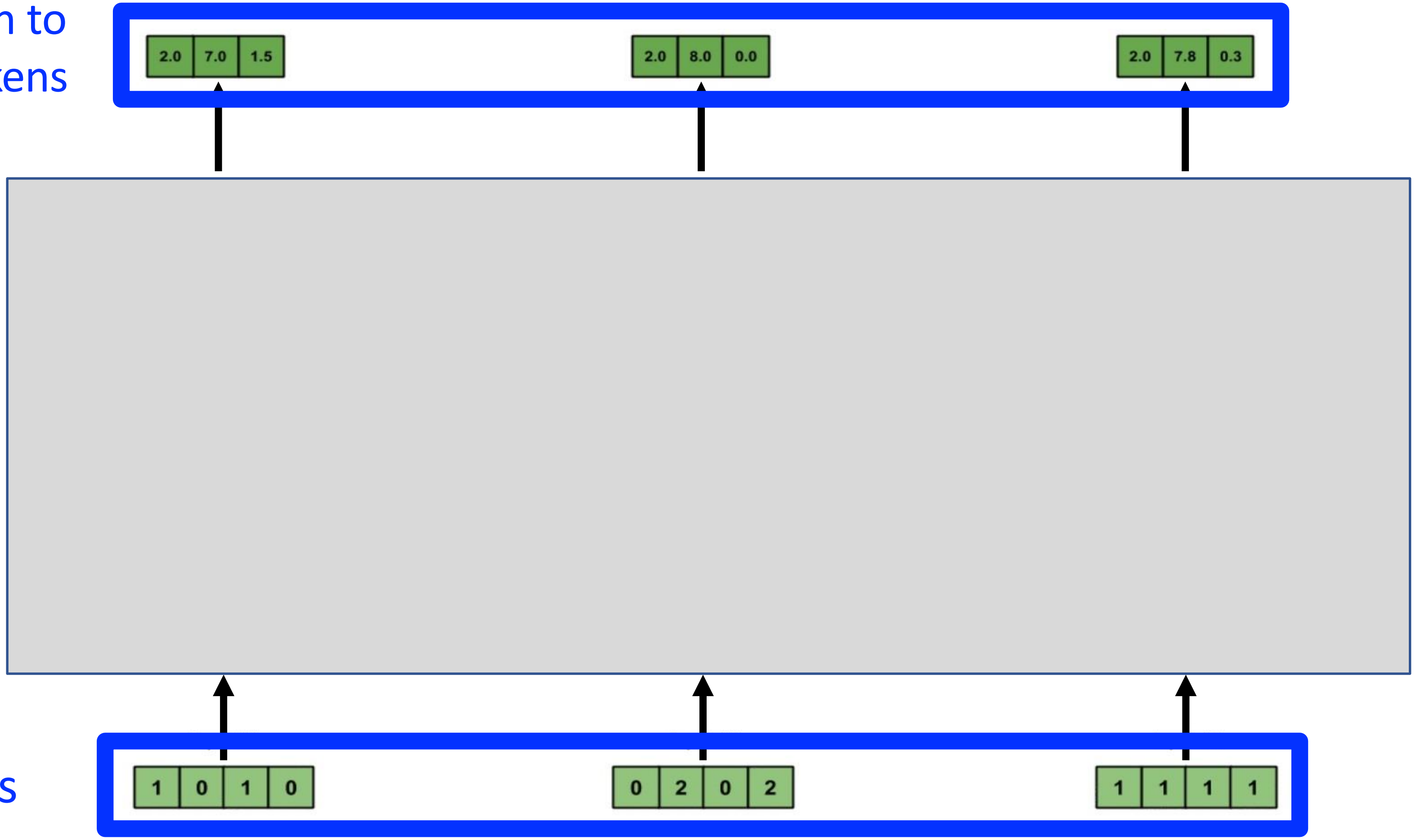
Relates tokens from different sources



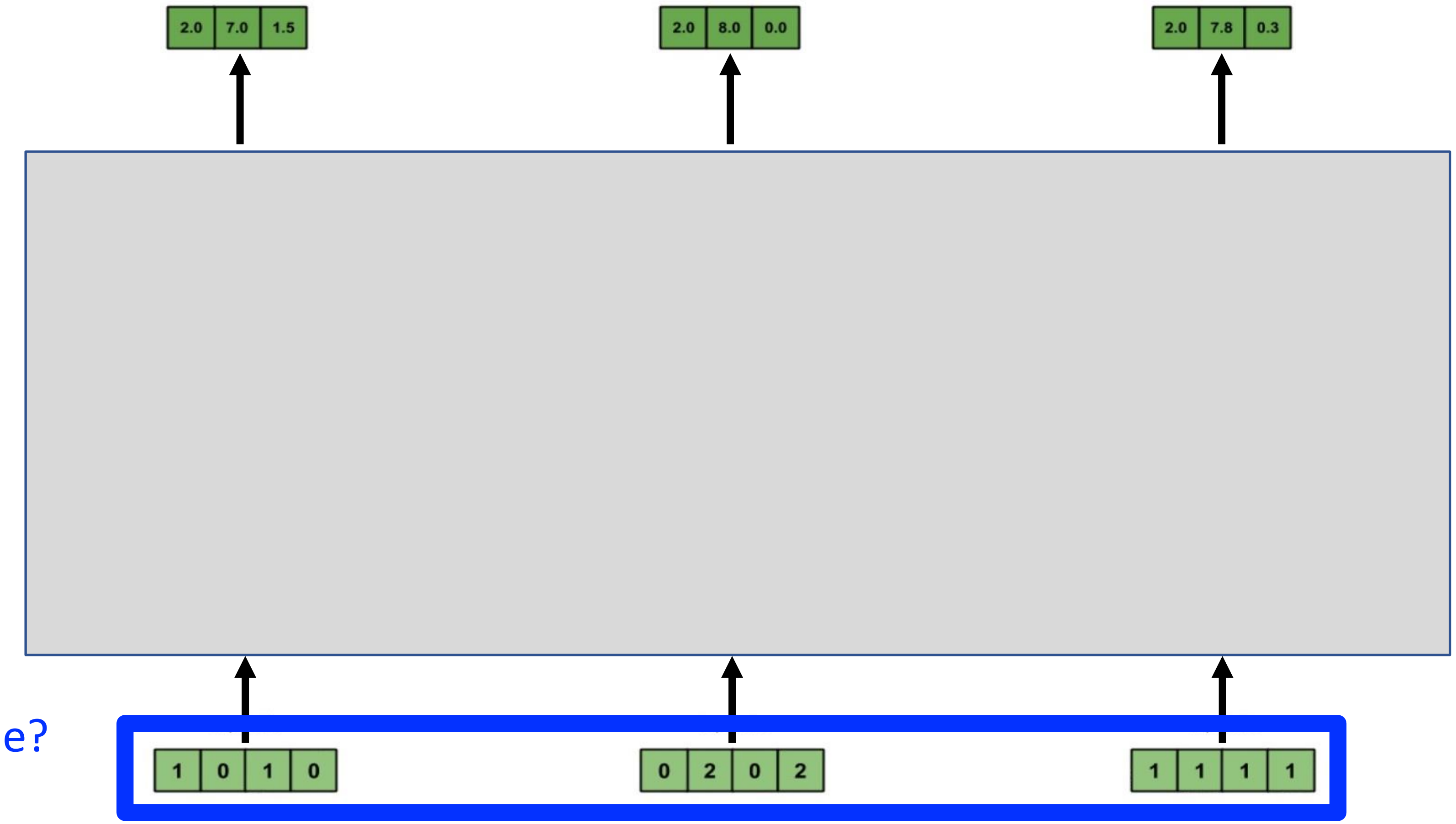
Computing Self-Attention: Example

New representation of each input token to reflect each one's relationship to all tokens

Input tokens



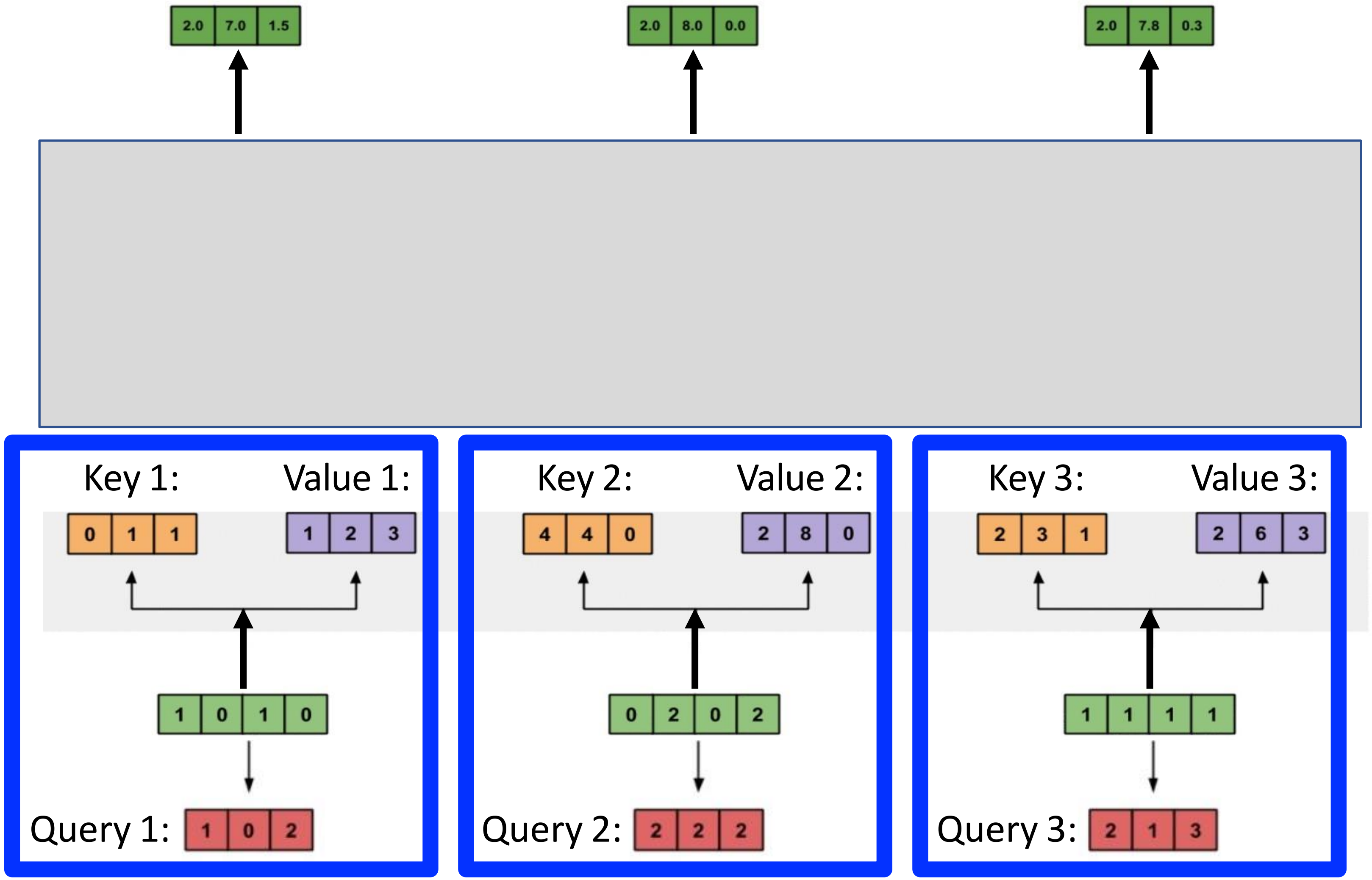
Computing Self-Attention: Example



- How many inputs are in this example?
- What is each one's dimensionality?

Computing Self-Attention: Example

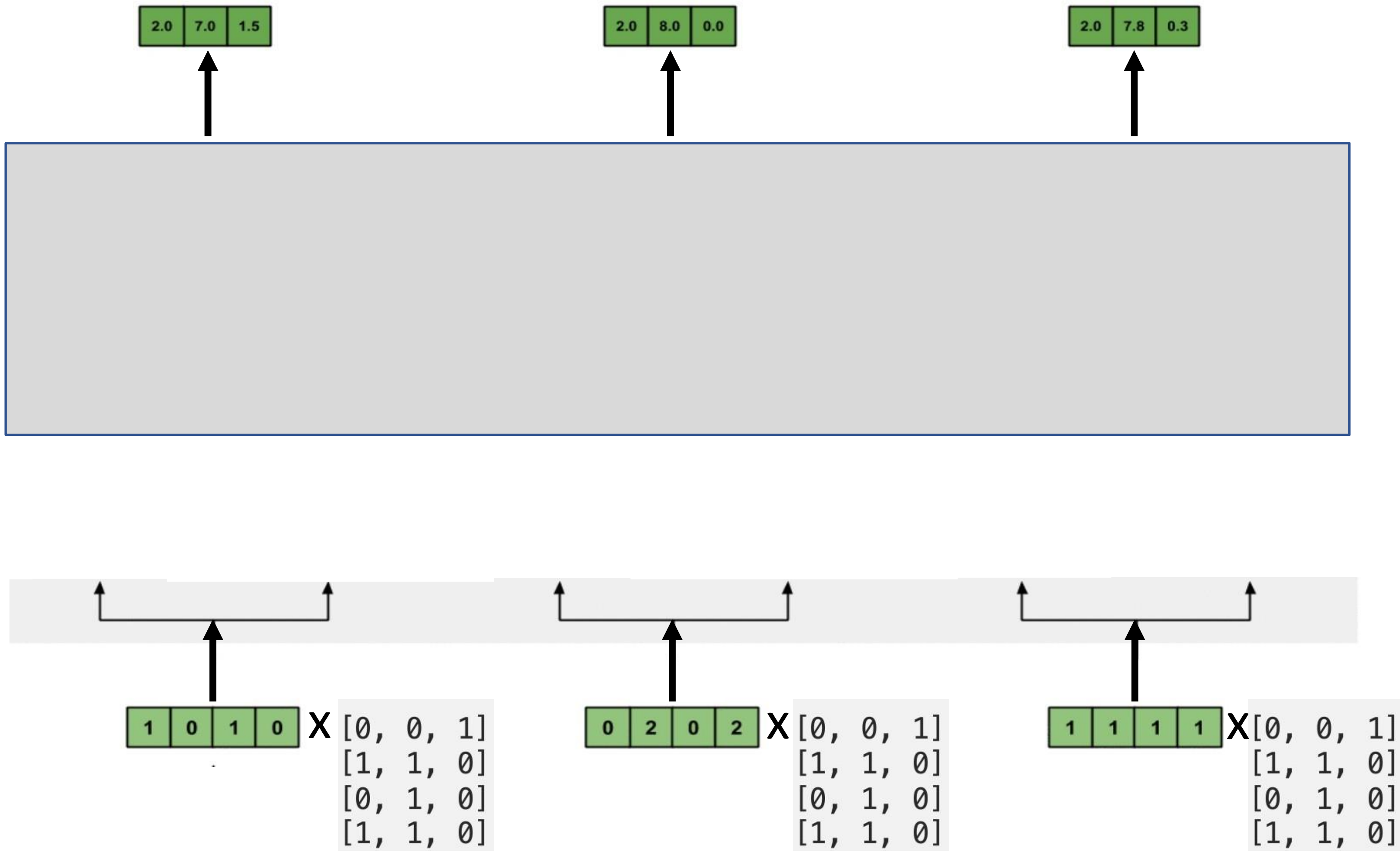
Three vectors are derived for each input by multiplying with three weight matrices (learned during training): query, key, and value



Computing Self-Attention: Example

e.g., key weights

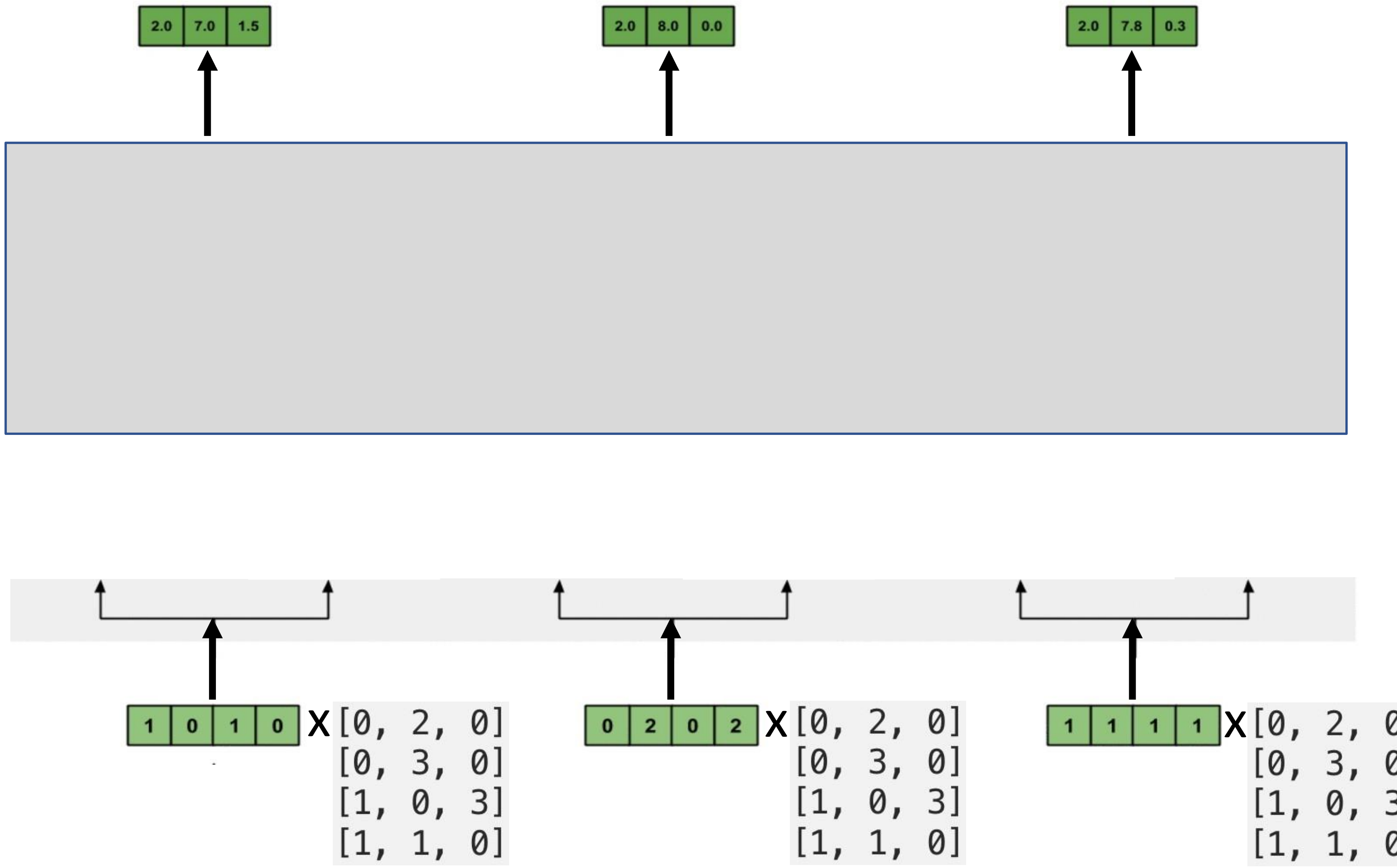
```
[0, 0, 1]
[1, 1, 0]
[0, 1, 0]
[1, 1, 0]
```



Computing Self-Attention: Example

e.g., value weights

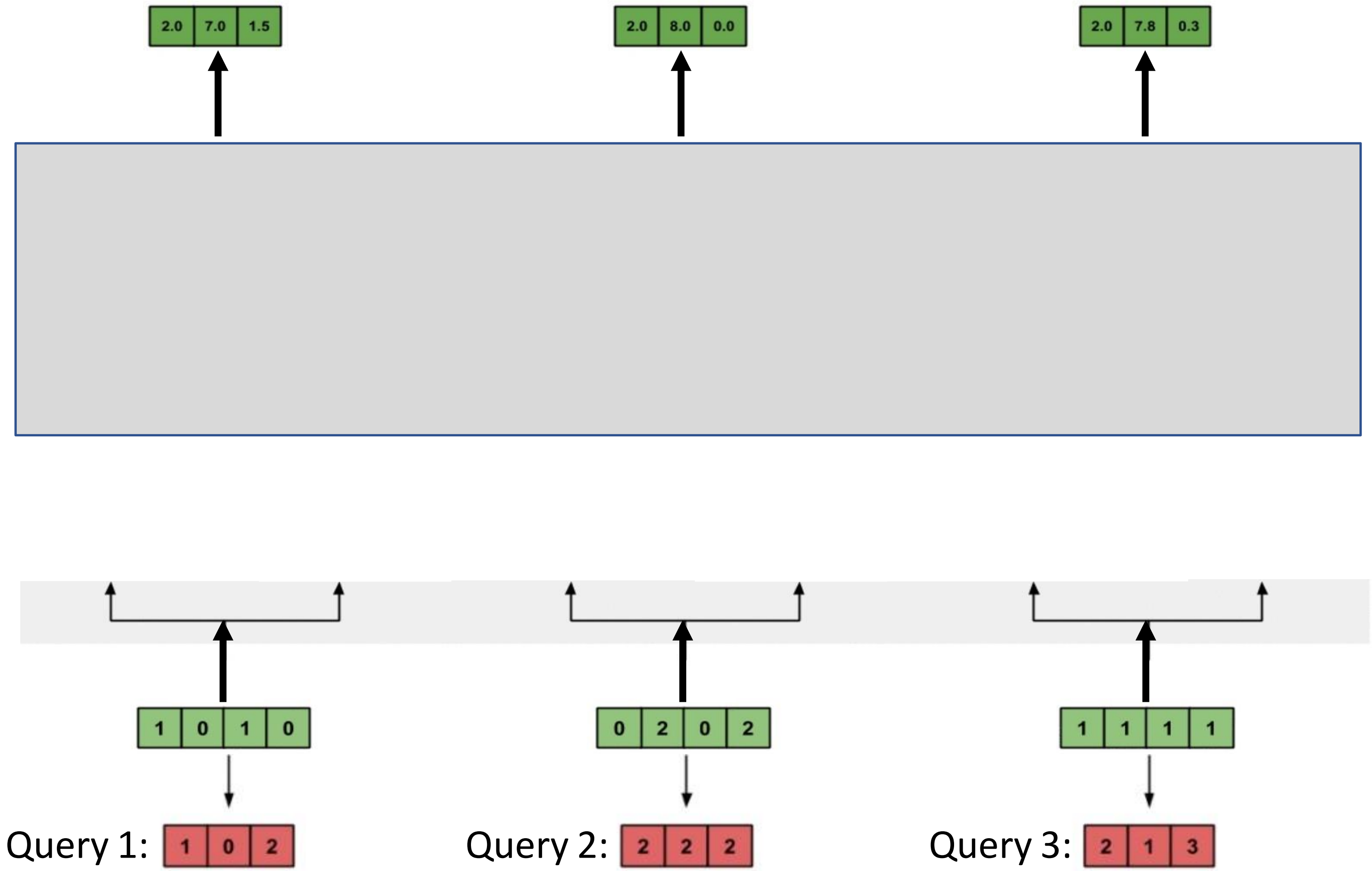
```
[0, 2, 0]
[0, 3, 0]
[1, 0, 3]
[1, 1, 0]
```



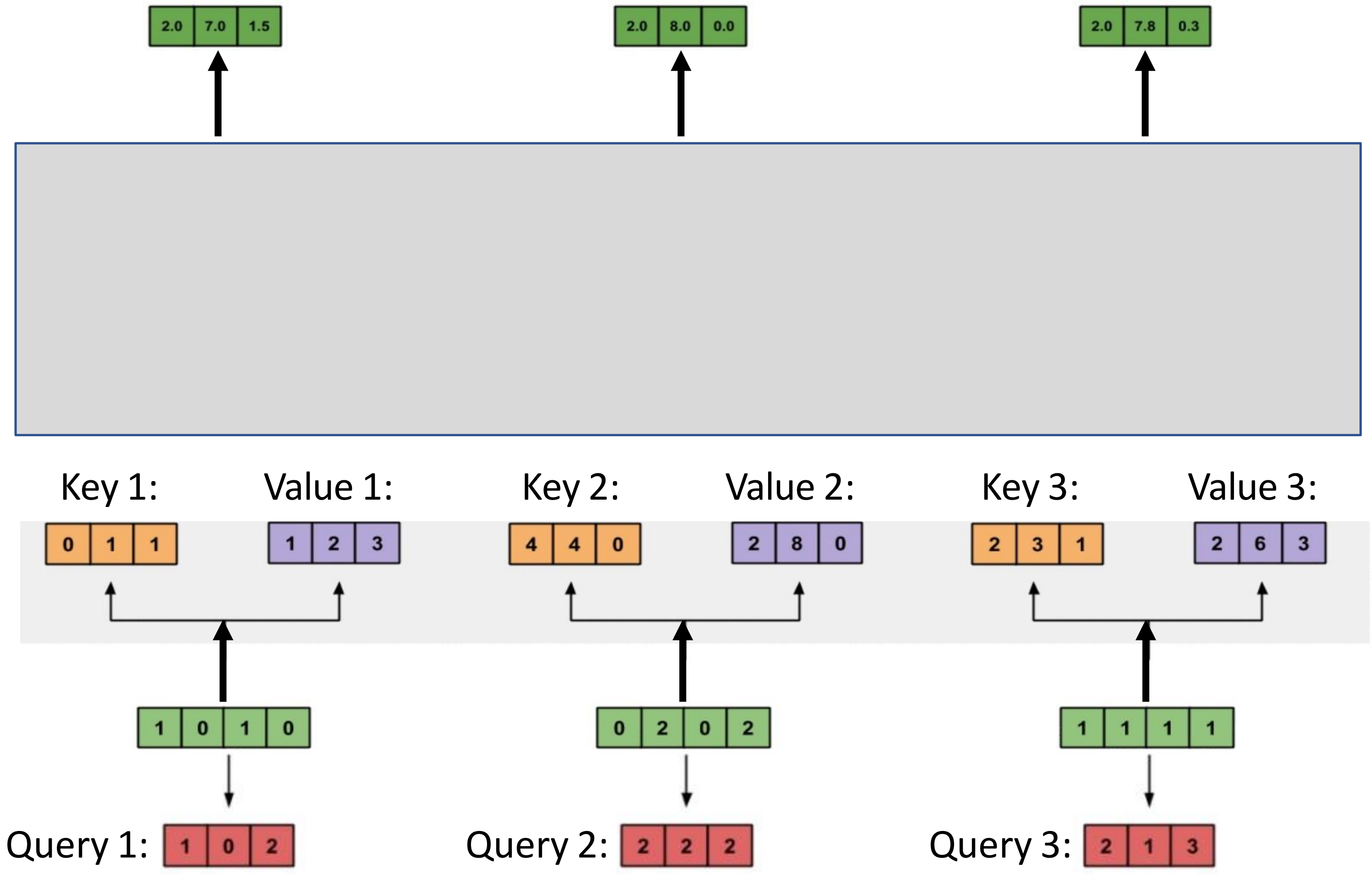
Computing Self-Attention: Example

e.g., query weights

```
[1, 0, 1]
[1, 0, 0]
[0, 0, 1]
[0, 1, 1]
```



Computing Self-Attention: Example

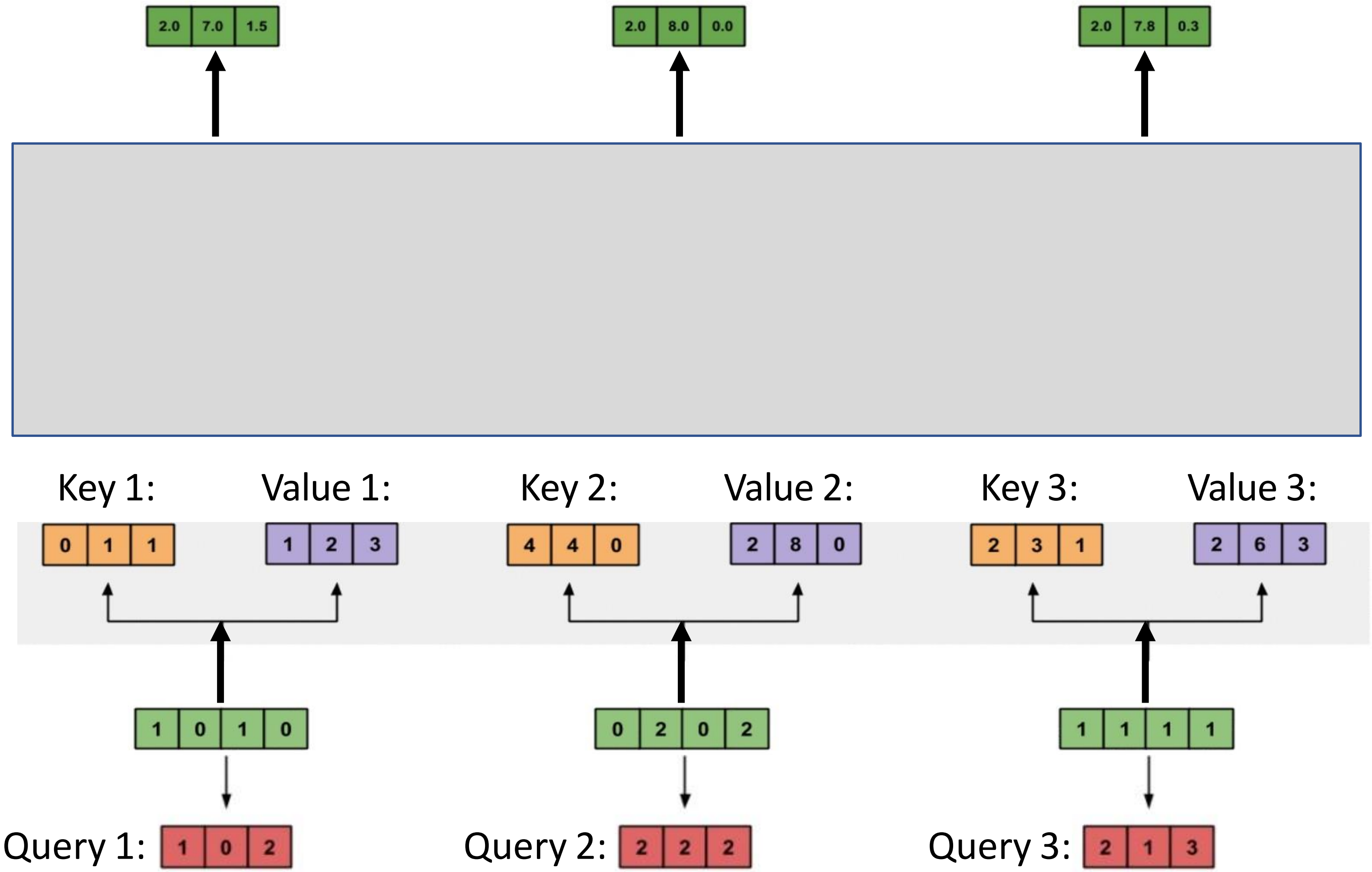


How many weight matrices are learned in this example?

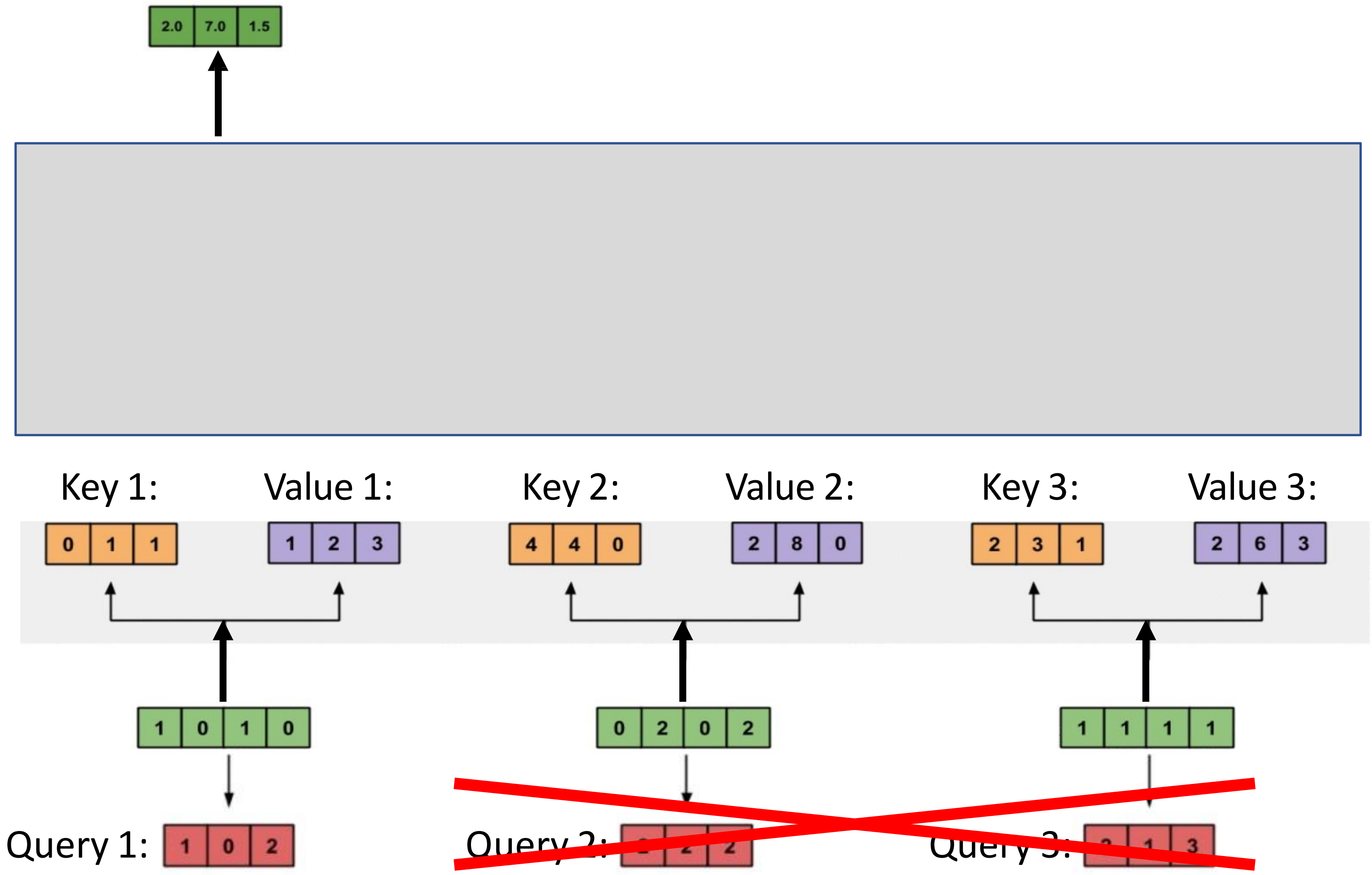
Computing Self-Attention: Example

Why do we learn the three weight matrices?

For each input, 2 of the derived vectors are used to compute **attention weights** (query and key) and the 3rd is **information** passed on for the new representation (value)



Computing Self-Attention: Example

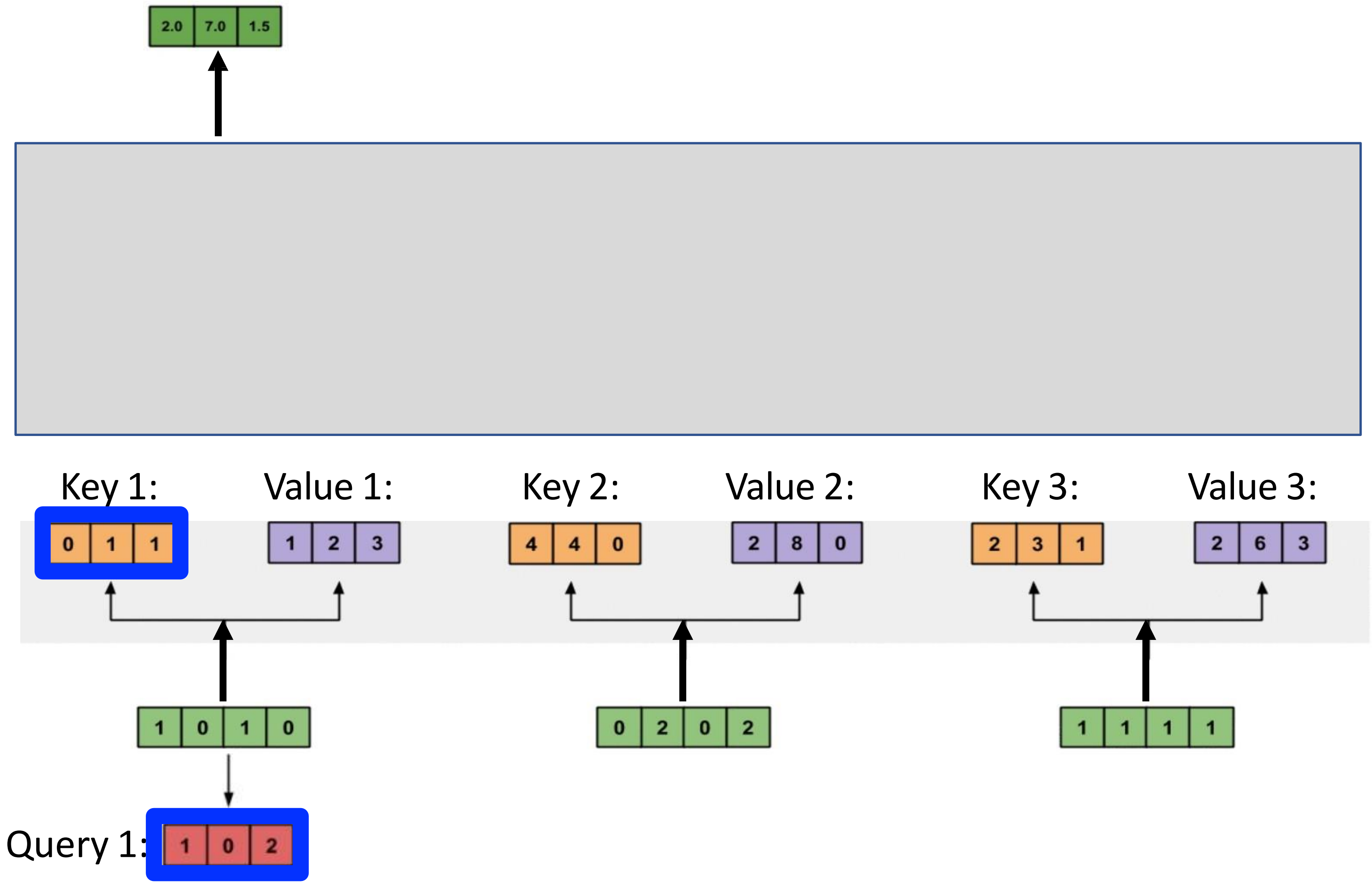


We now will examine how to find the new representation for the first input.

Computing Self-Attention: Example

Attention score: dot product of query with all keys to identify relevant tokens; e.g.,

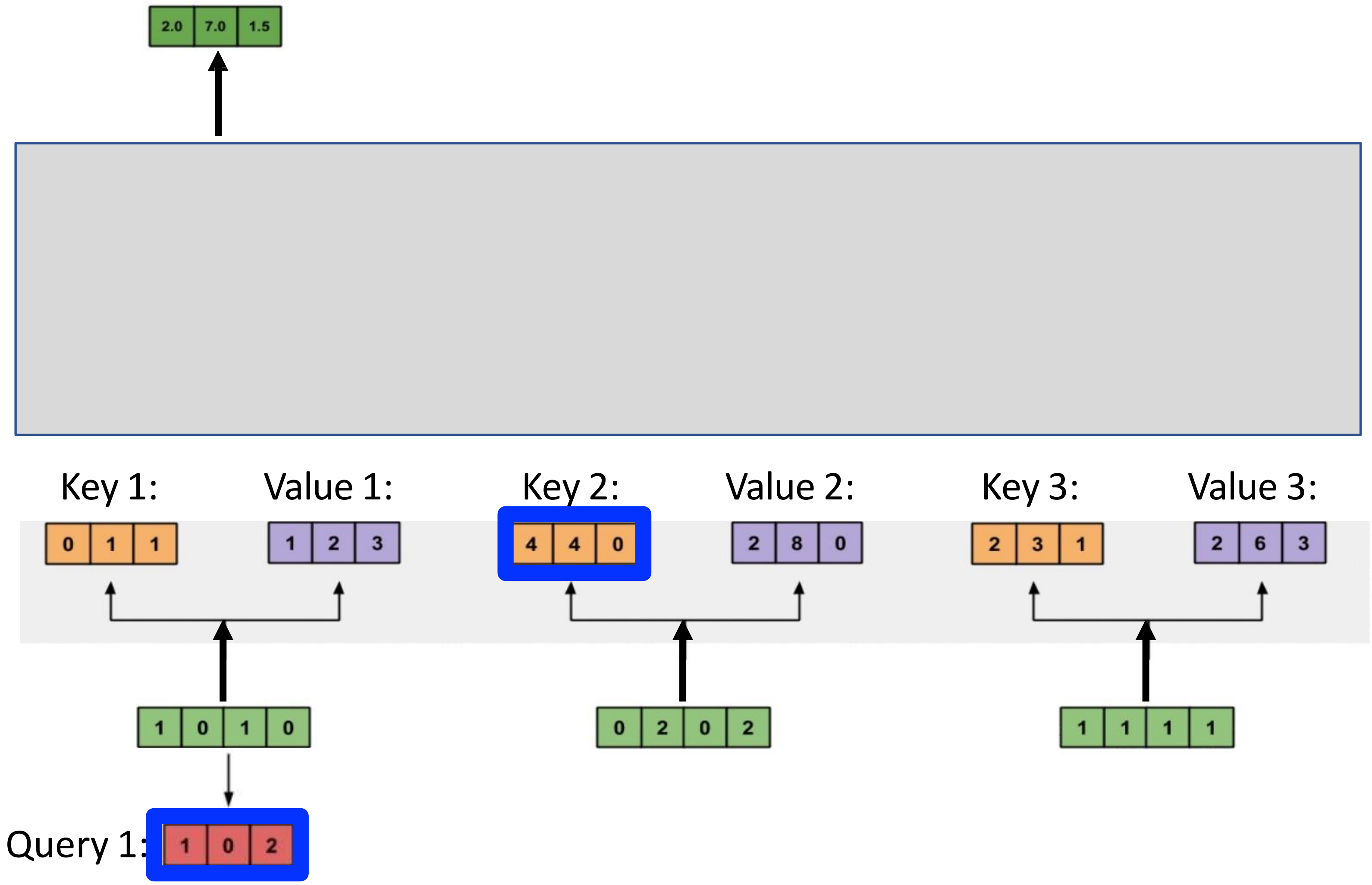
$$\begin{bmatrix} 1 & 0 & 2 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = ?$$



Computing Self-Attention: Example

Attention score: dot product of query with all keys to identify relevant tokens; e.g.,

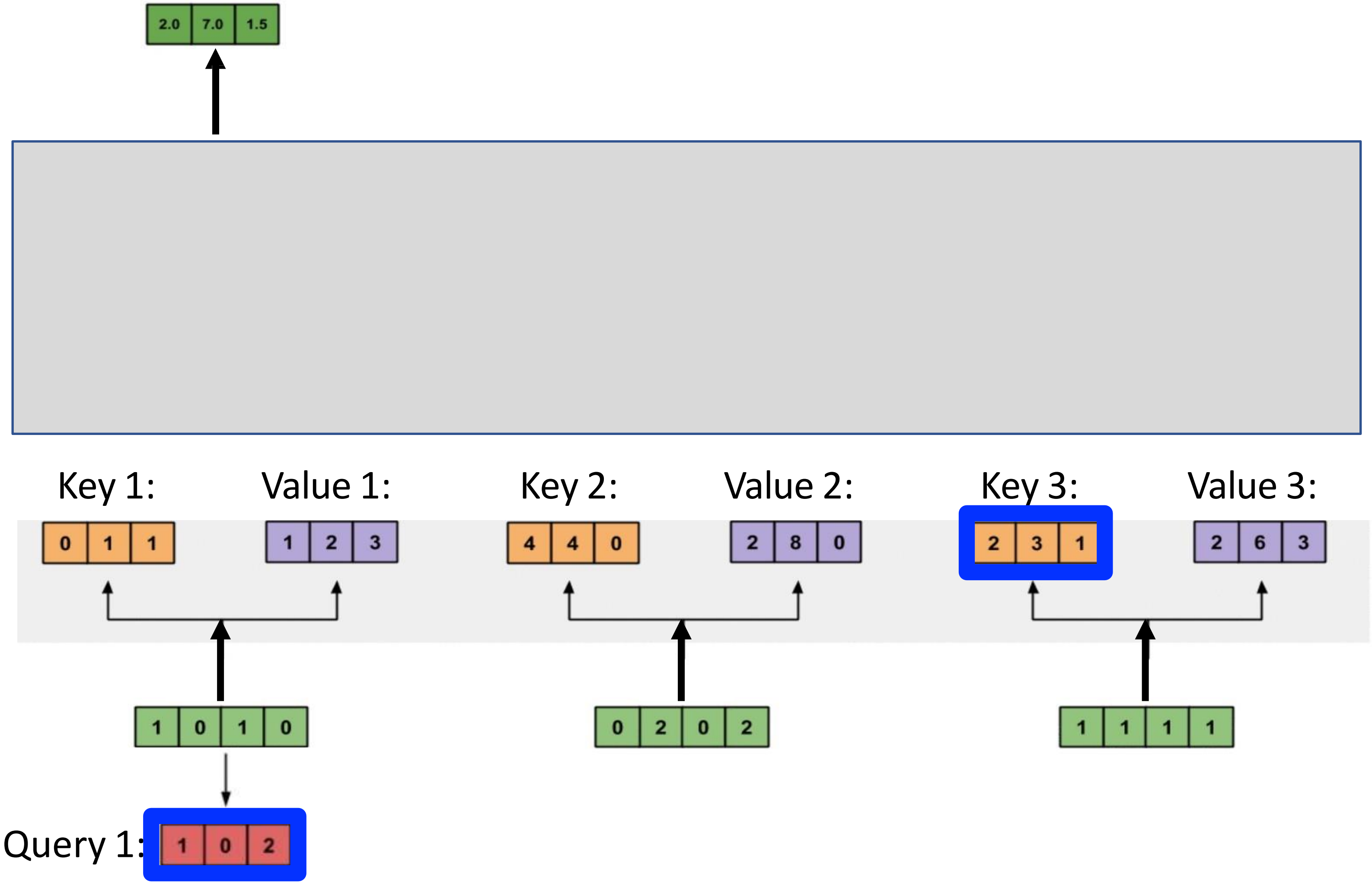
$$\begin{bmatrix} 1 & 0 & 2 \end{bmatrix} \times \begin{bmatrix} 4 \\ 4 \\ 0 \end{bmatrix} = ?$$



Computing Self-Attention: Example

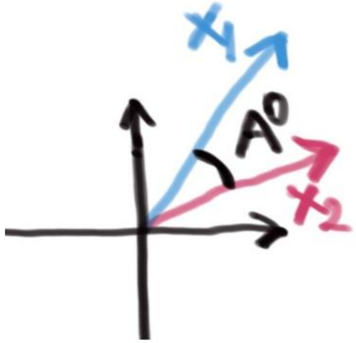
Attention score: dot product of query with all keys to identify relevant tokens; e.g.,

$$\begin{bmatrix}1 & 0 & 2\end{bmatrix} \times \begin{bmatrix}2\\3\\1\end{bmatrix} = ?$$



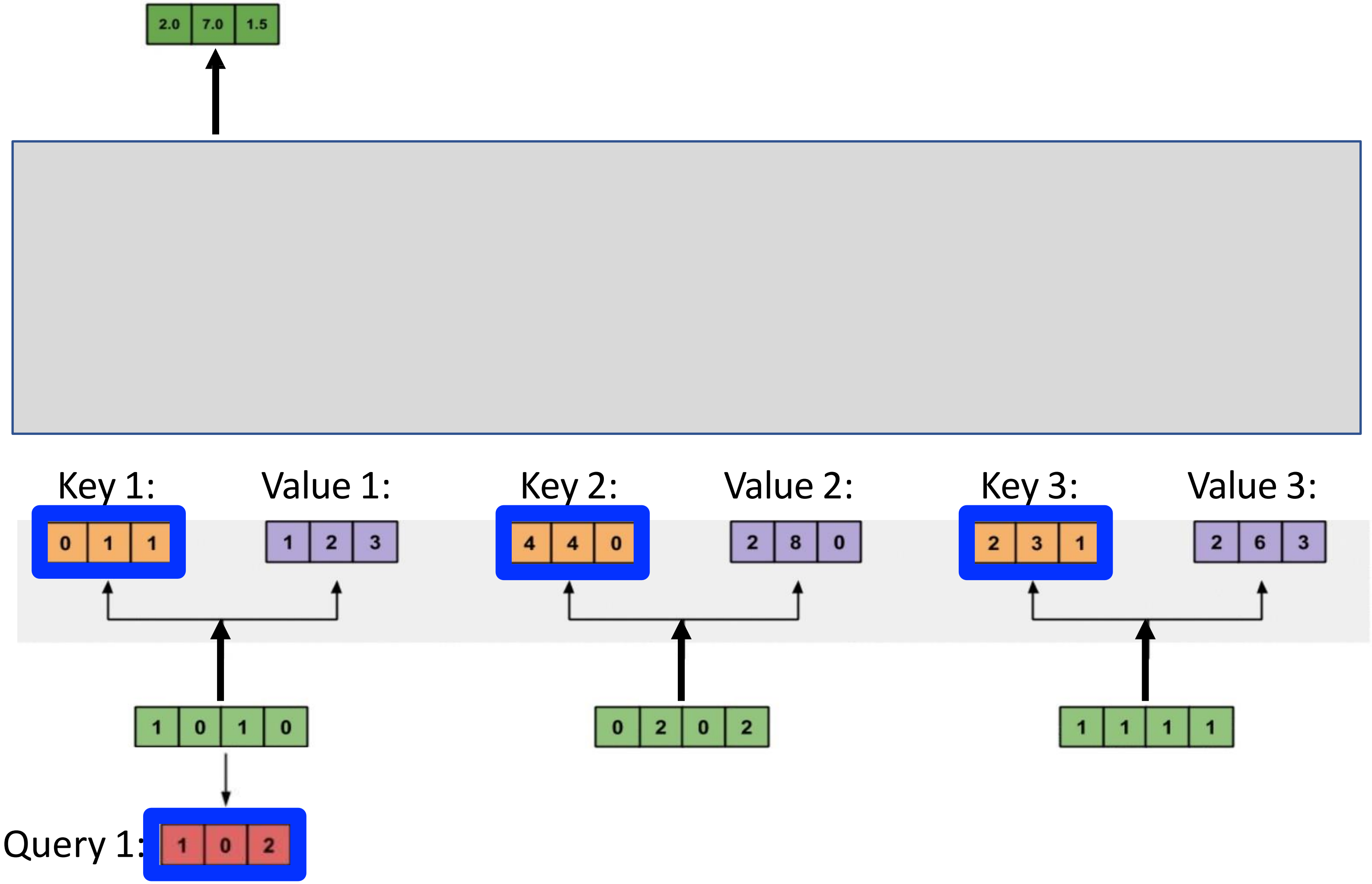
Computing Self-Attention: Example

Why dot product? Indicates similarity of two vectors
- Match = 1 (i.e., cos(0))
- Opposites = -1 (i.e., cos(180))



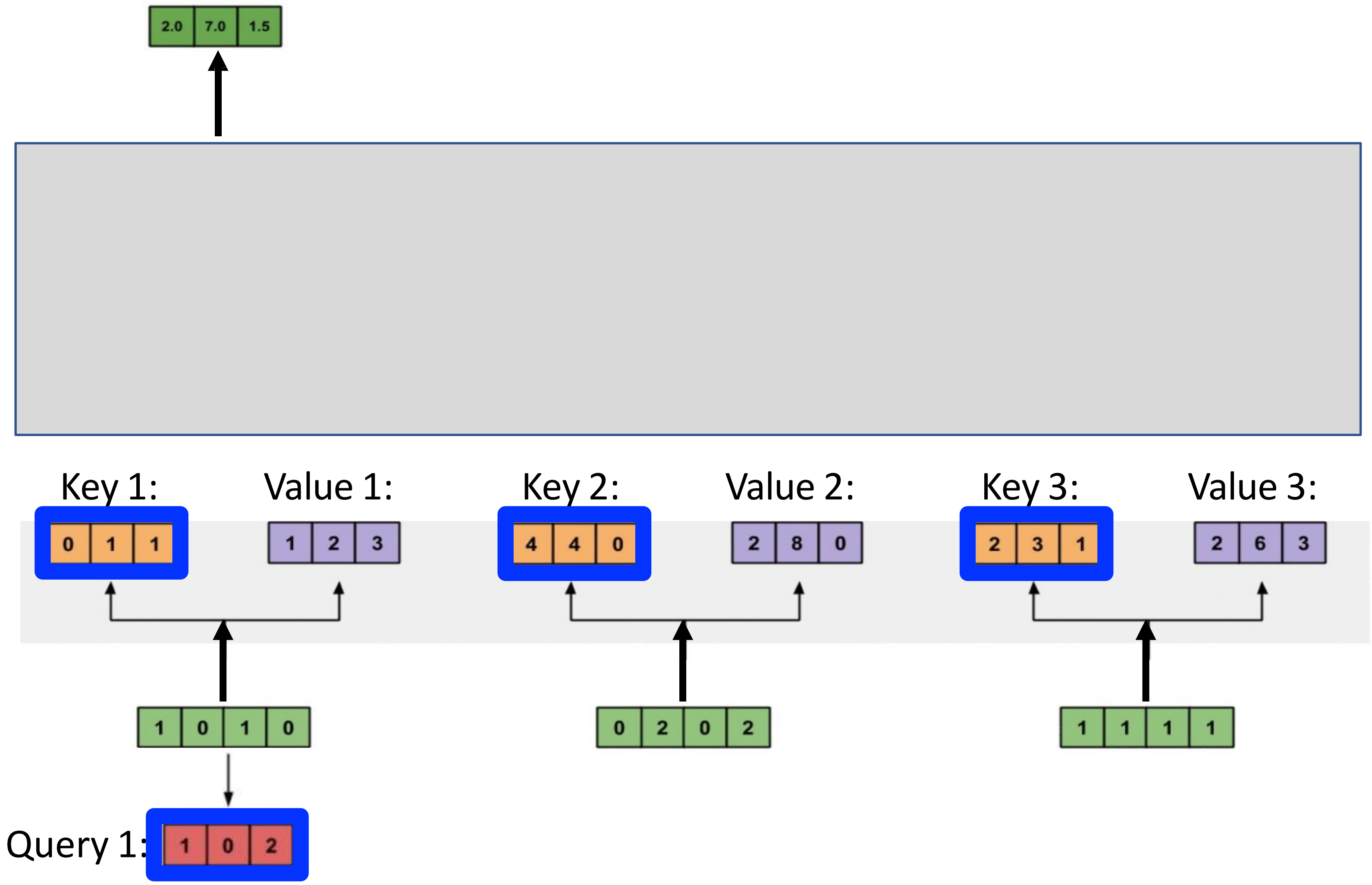
$x_1 \cdot x_2 = \cos A^{\circ} |x_1||x_2|$

$|x| =$ size / magnitude of the vector x



Computing Self-Attention: Example

Can also use similarity measures other than the dot product



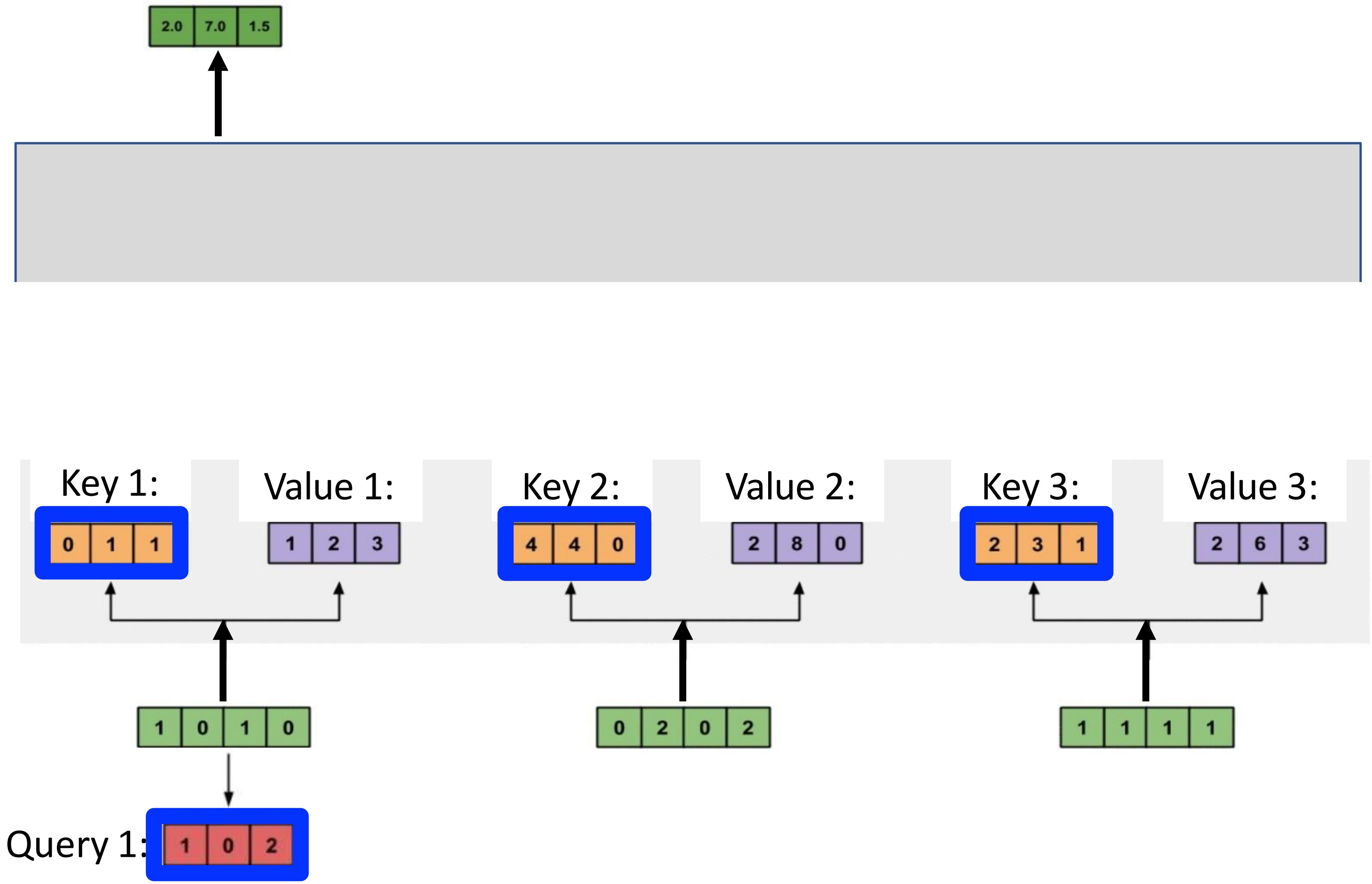
Computing Self-Attention: Example

Attention weights: softmax scores for all inputs to quantify each token's relevance; e.g.,

= softmax([2, 4, 4])

= [0.0, 0.5, 0.5])

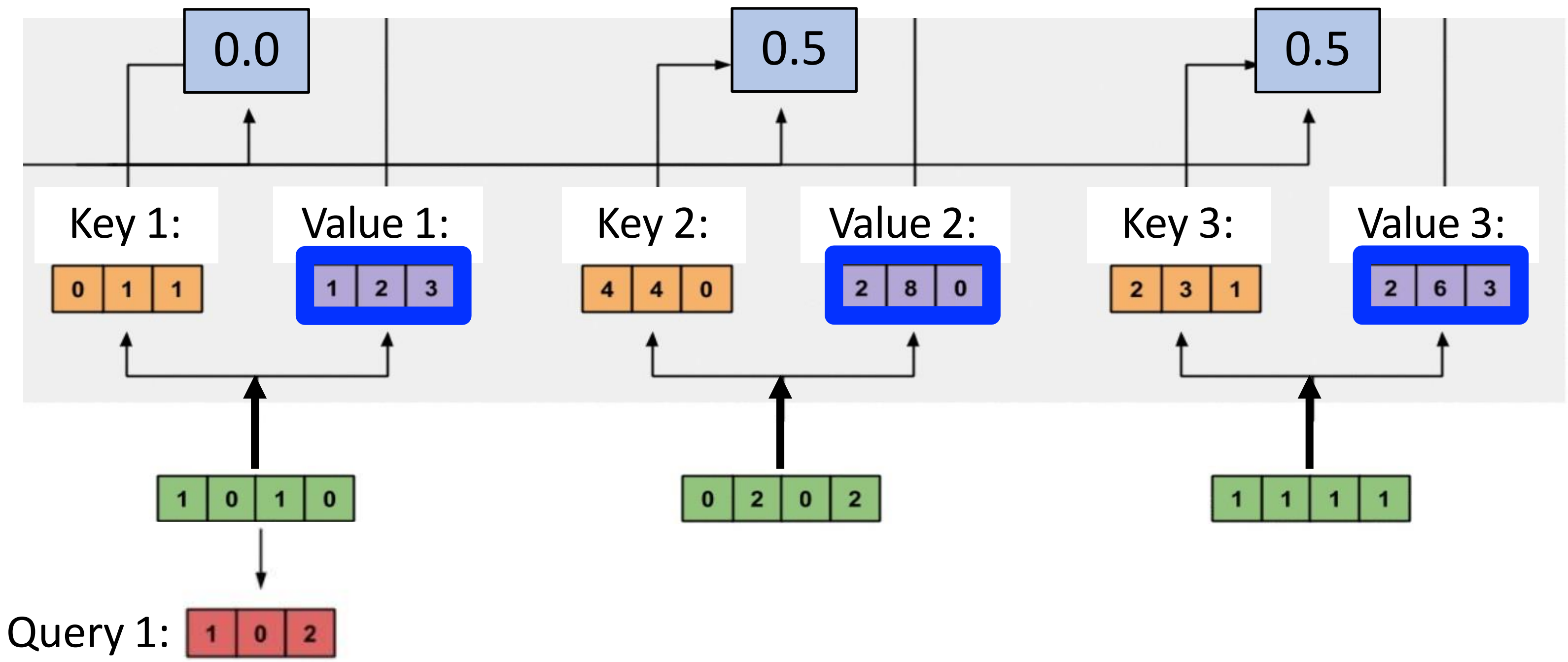
To which input(s) is input 1 most related?



Computing Self-Attention: Example

Compute new representation of input token that reflects entire input:

1. Attention weights x Values

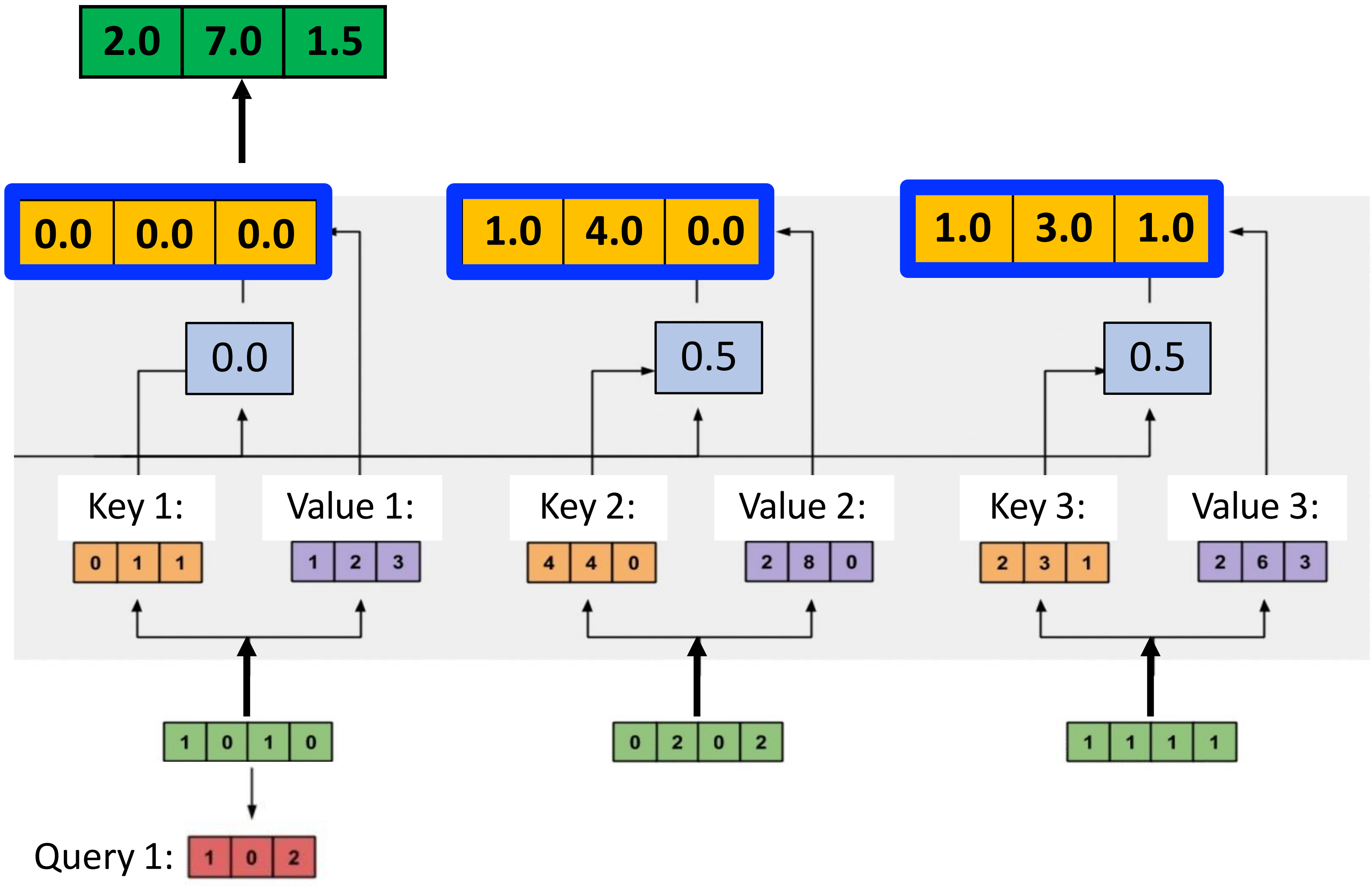


Computing Self-Attention: Example

Compute new representation of input token that reflects entire input:

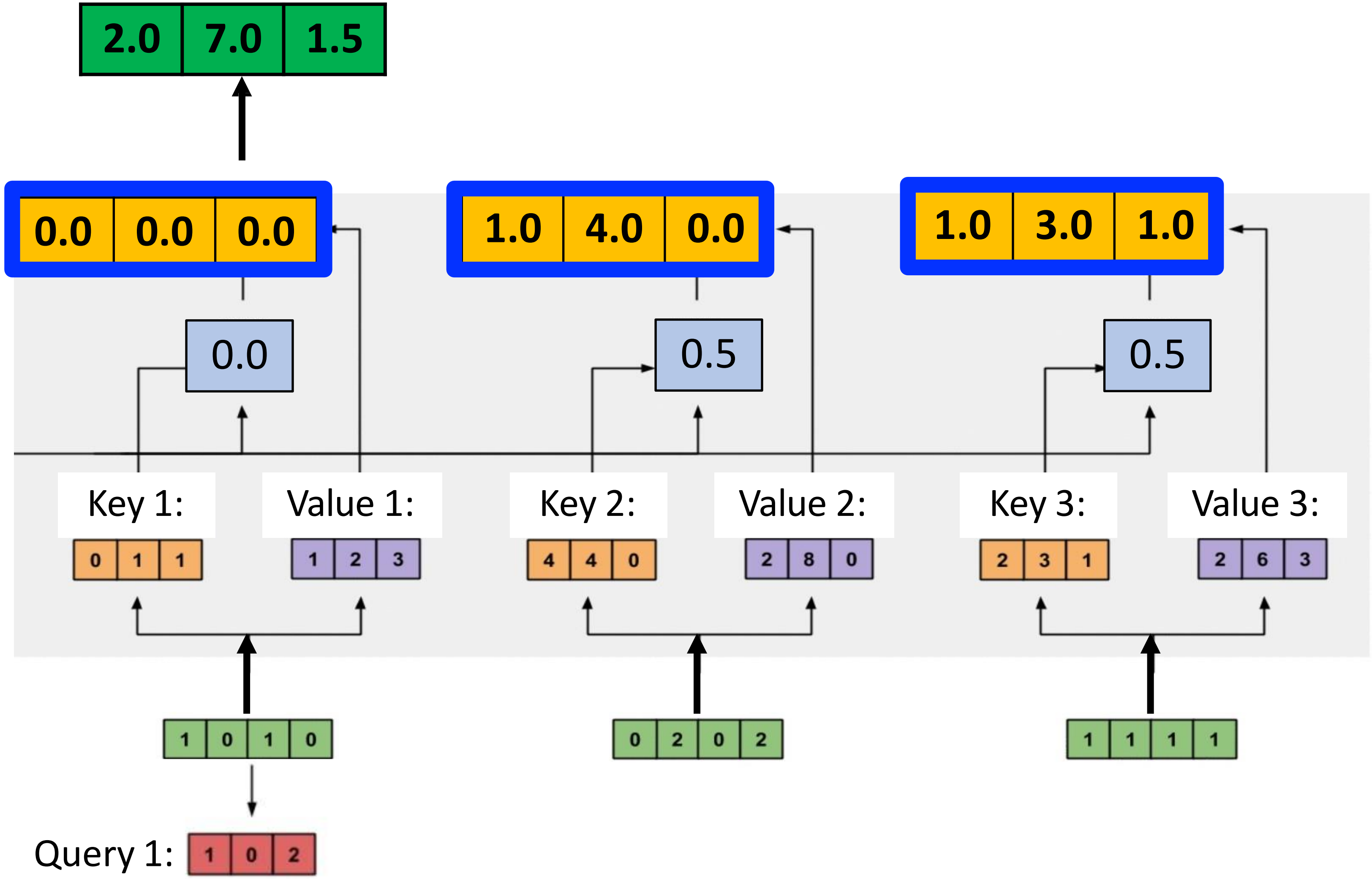
1. Attention weights x Values

2. Sum all weighted vectors



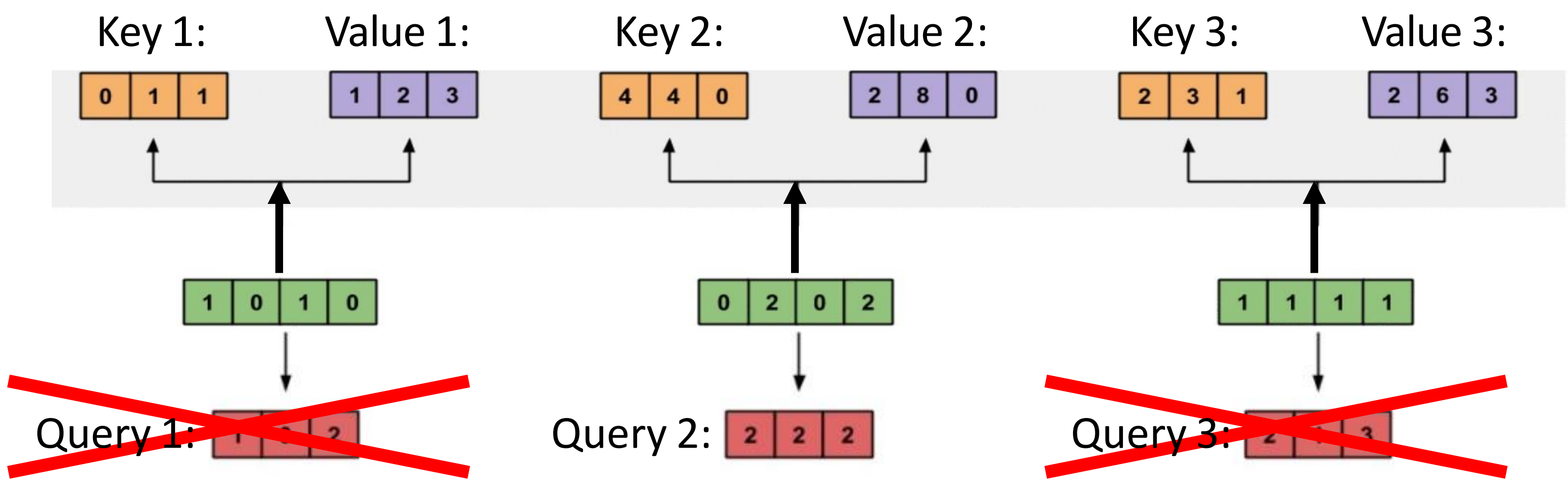
Computing Self-Attention: Example

Attention weights amplify input representations (values) that we want to pay attention to and repress the rest



Computing Self-Attention: Example

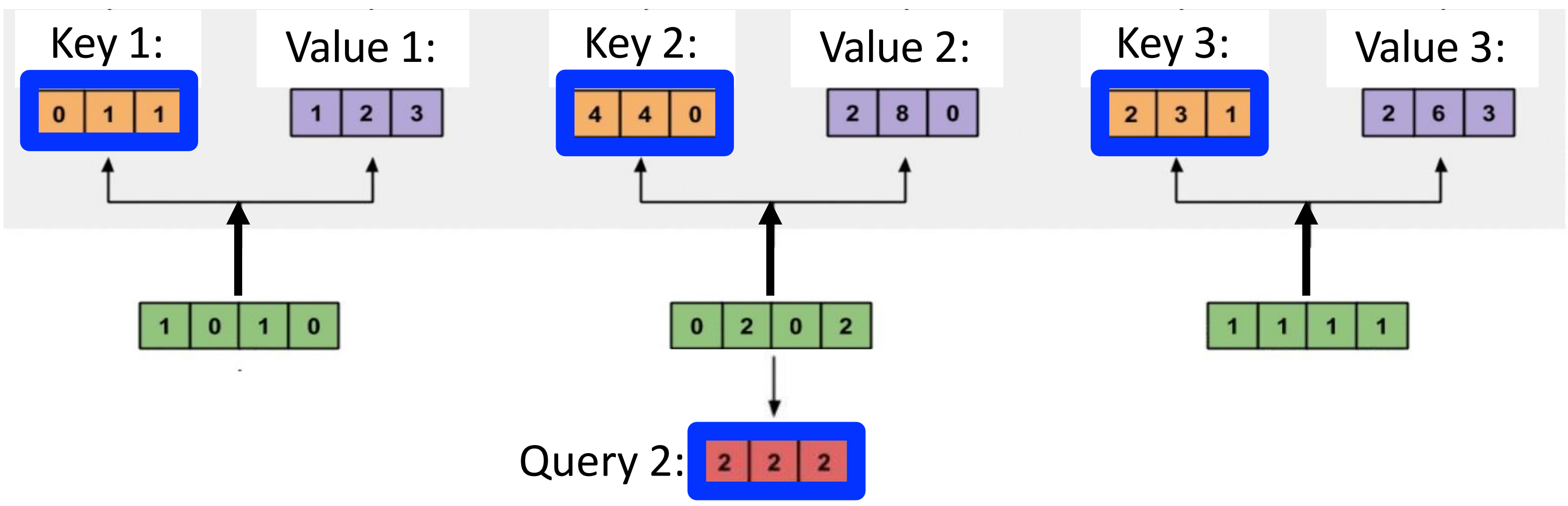
Repeat the same process for each remaining input token



Computing Self-Attention: Example

1. Compute attention weights
- Softmax resulting 3 scores from query x keys

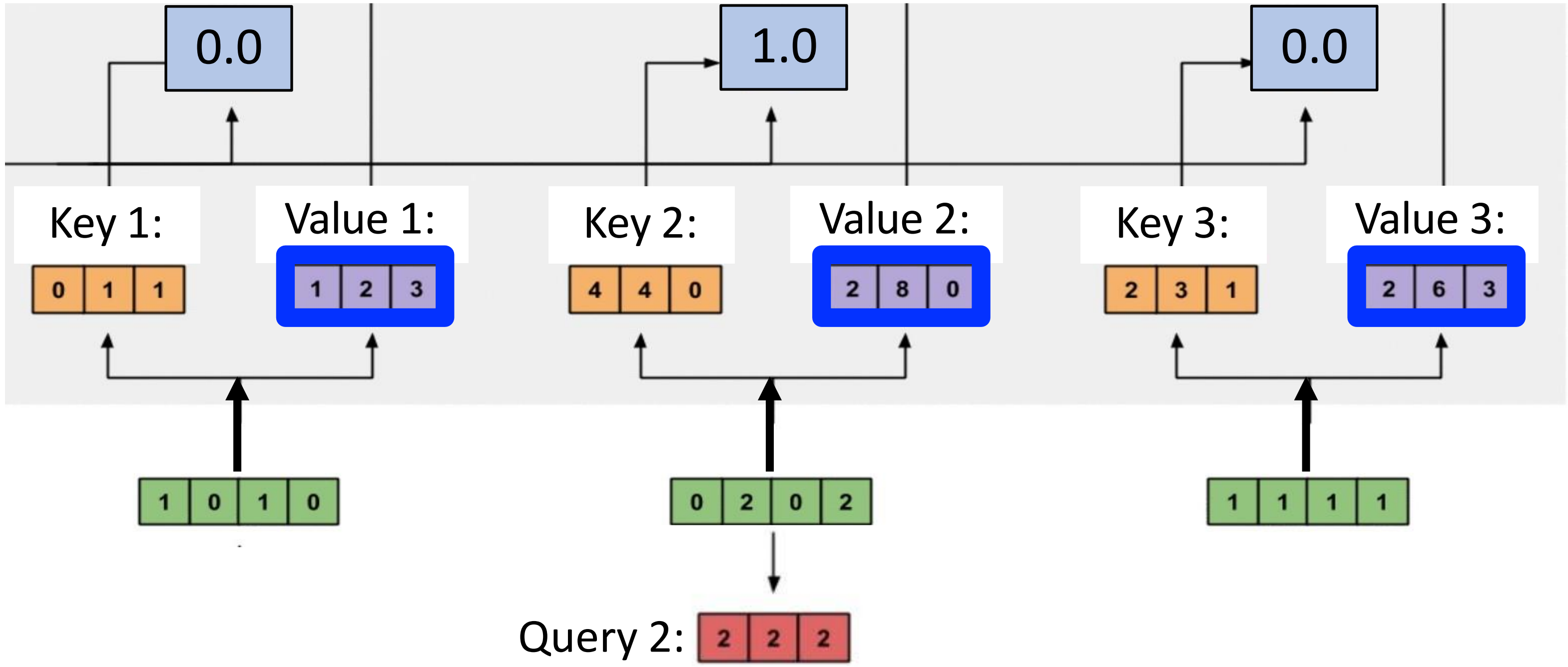
To which input(s) is input 2 most related?`



Computing Self-Attention: Example

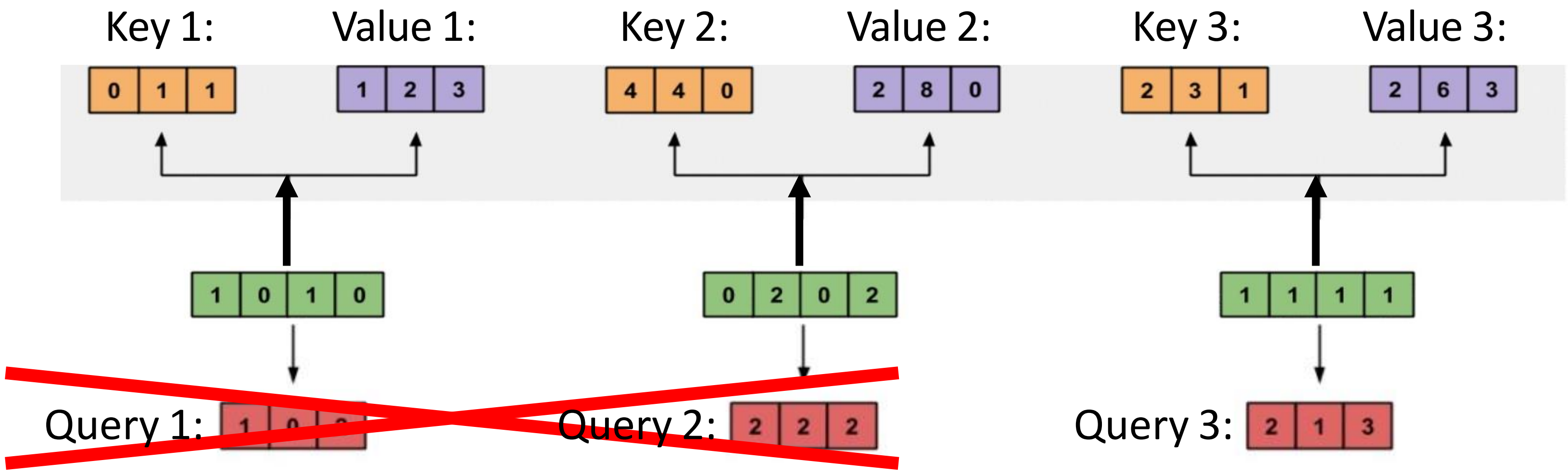
1. Compute attention weights
- Softmax resulting 3 scores from query x keys

2. Compute weighted sum of values using attention scores



Computing Self-Attention: Example

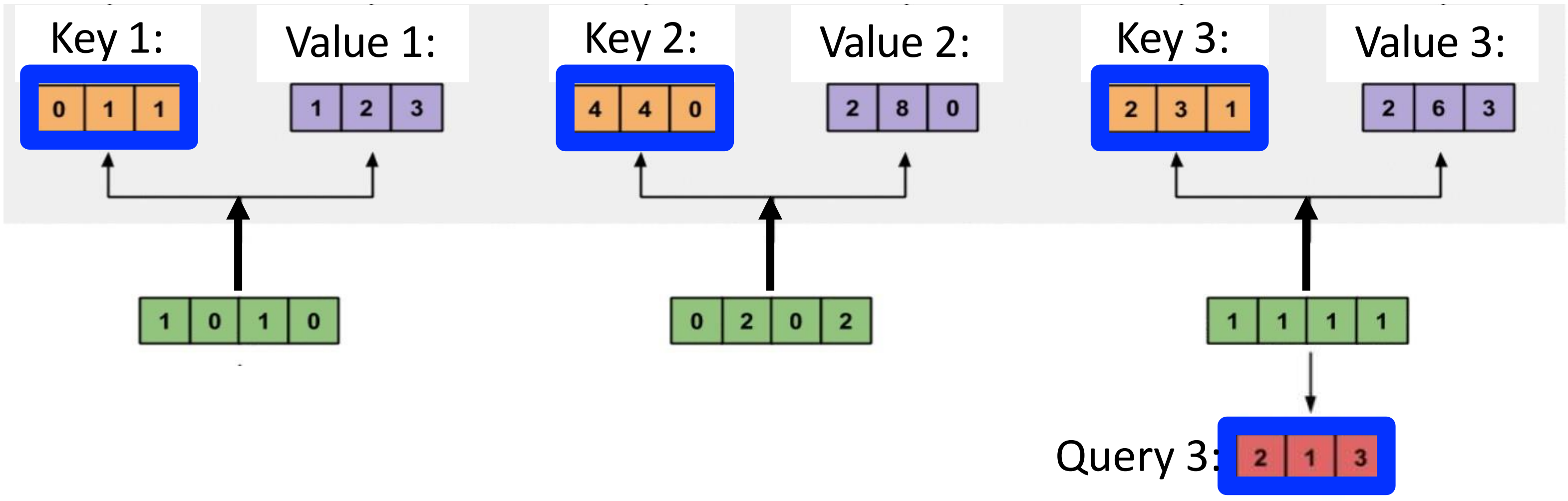
Repeat the same process for each remaining input token



Computing Self-Attention: Example

1. Compute attention weights
- Softmax resulting 3 scores from query x keys

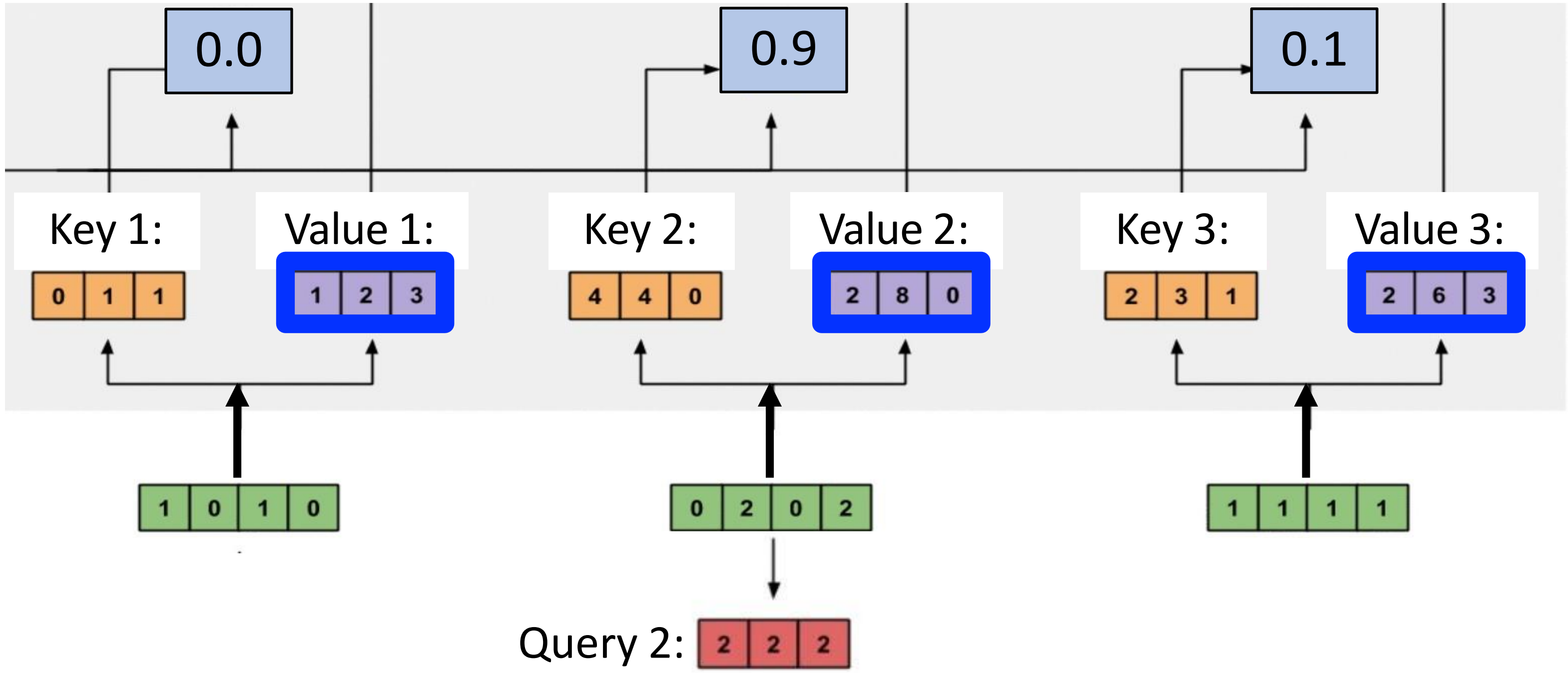
To which input(s) is input 3 most related?



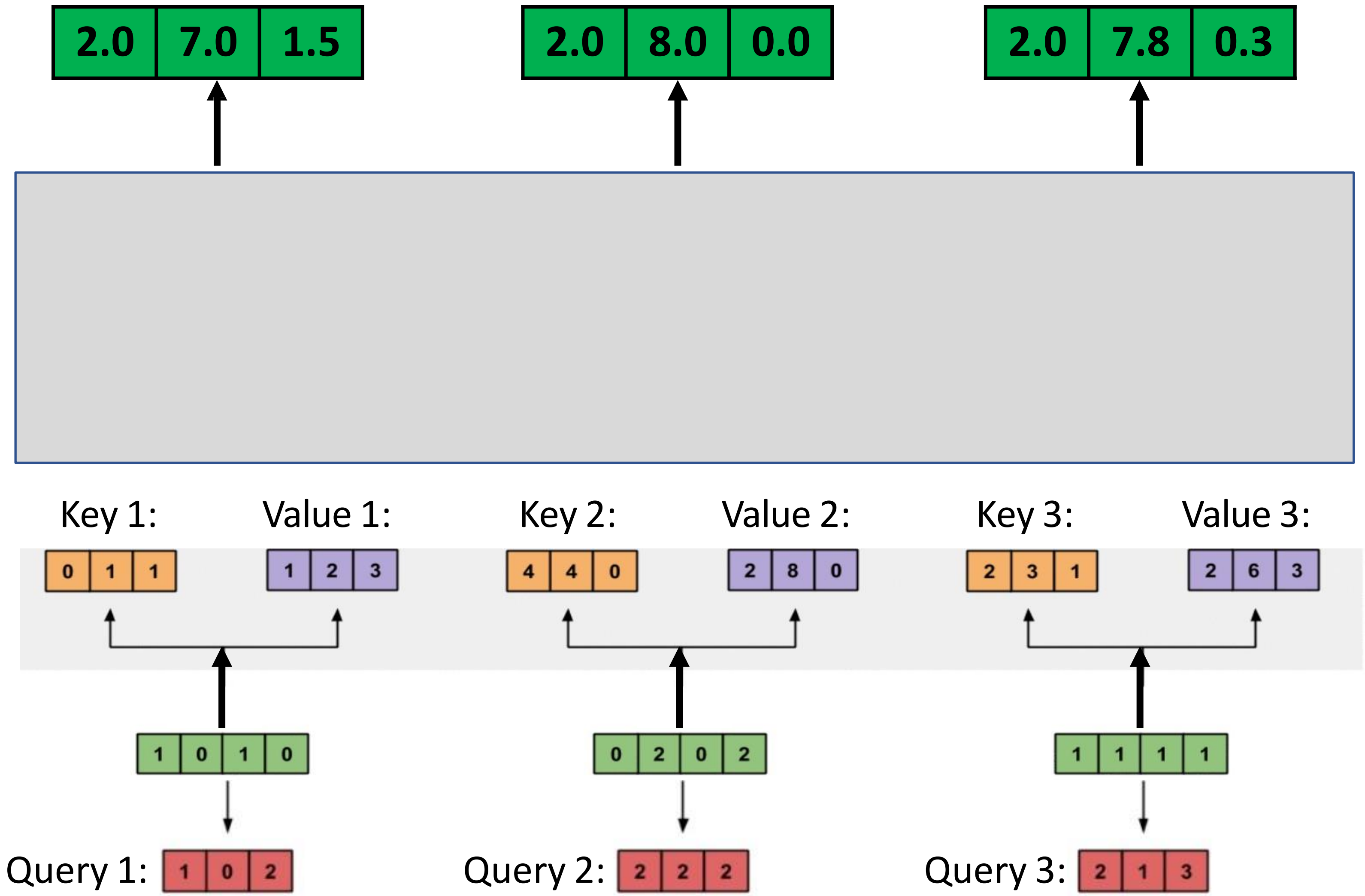
Computing Self-Attention: Example

1. Compute attention weights
- Softmax resulting 3 scores from query x keys

2. Compute weighted sum of values using attention scores

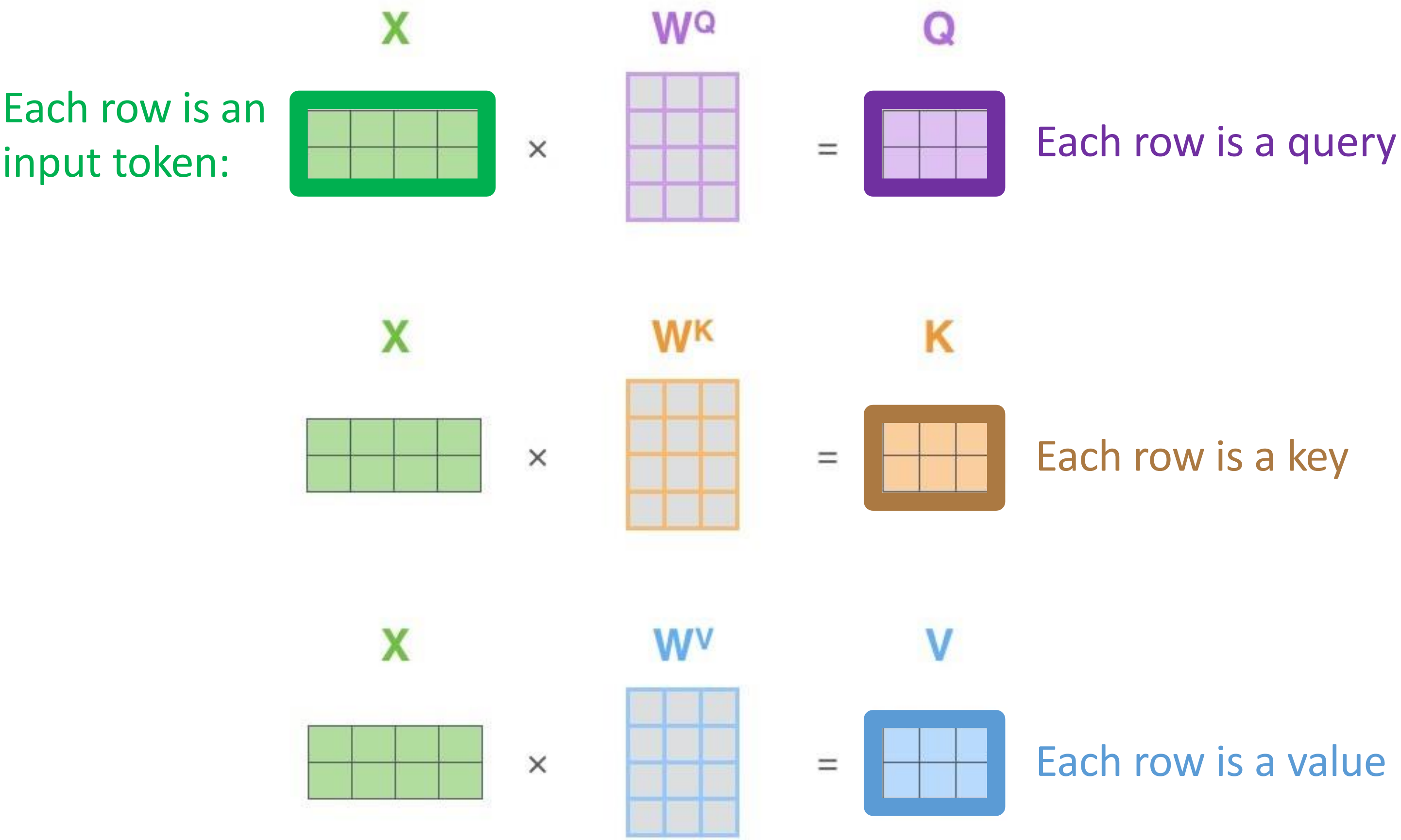


Computing Self-Attention: Example



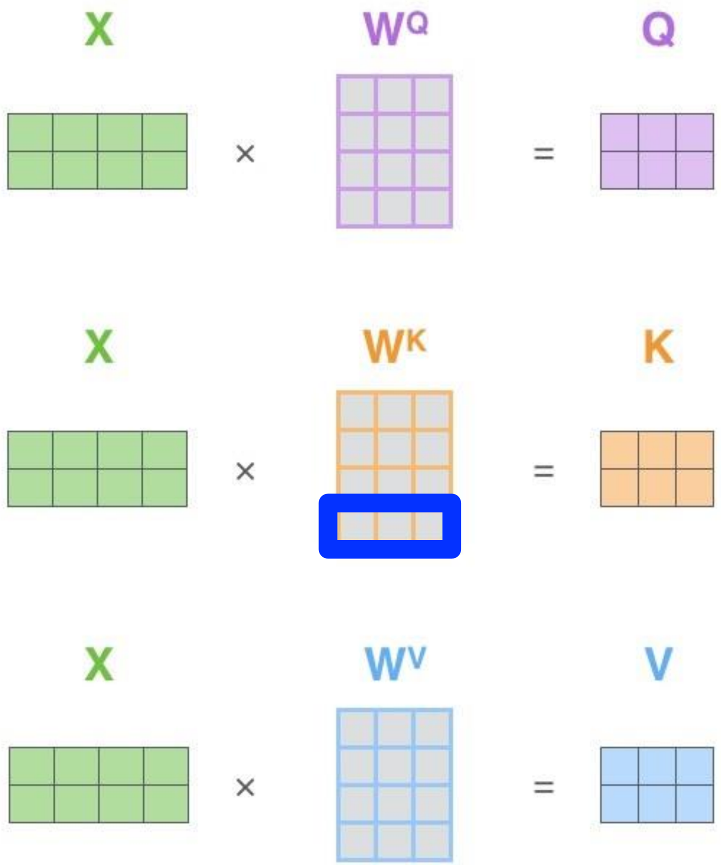
Efficient Computation for Self-Attention

Step 1

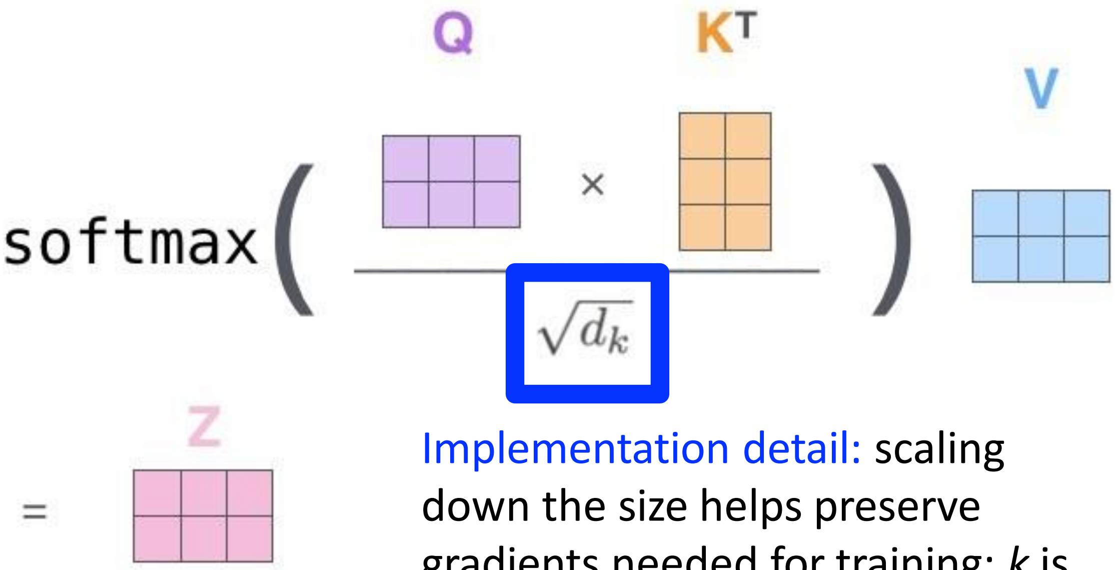


Efficient Computation for Self-Attention

Step 1

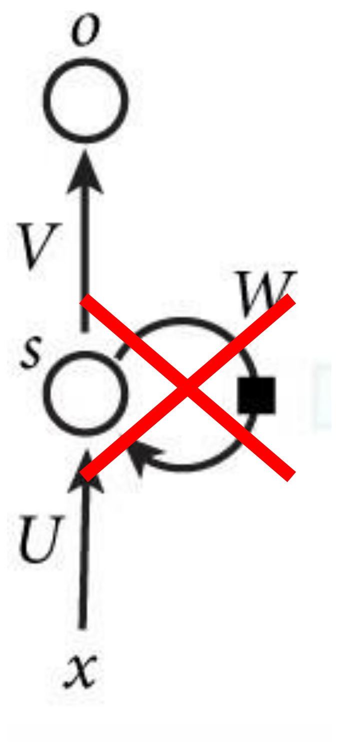


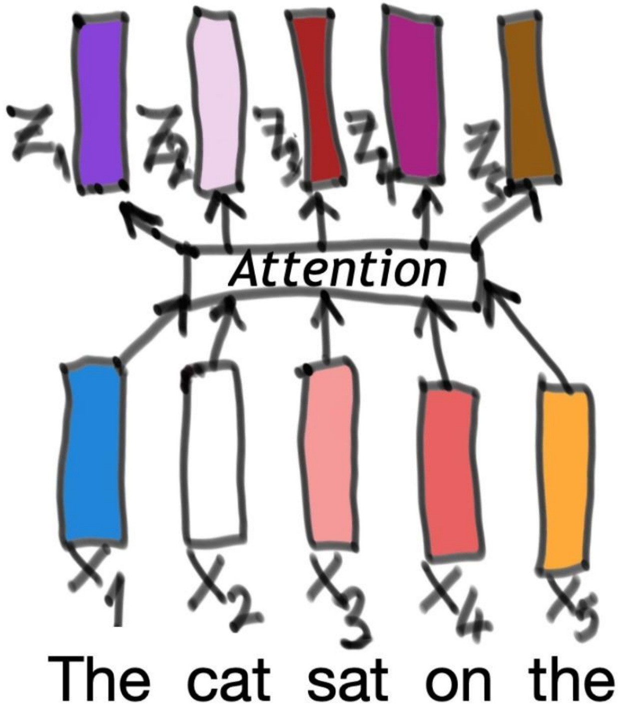
Step 2



Implementation detail: scaling down the size helps preserve gradients needed for training; *k* is dimensionality of the key vector

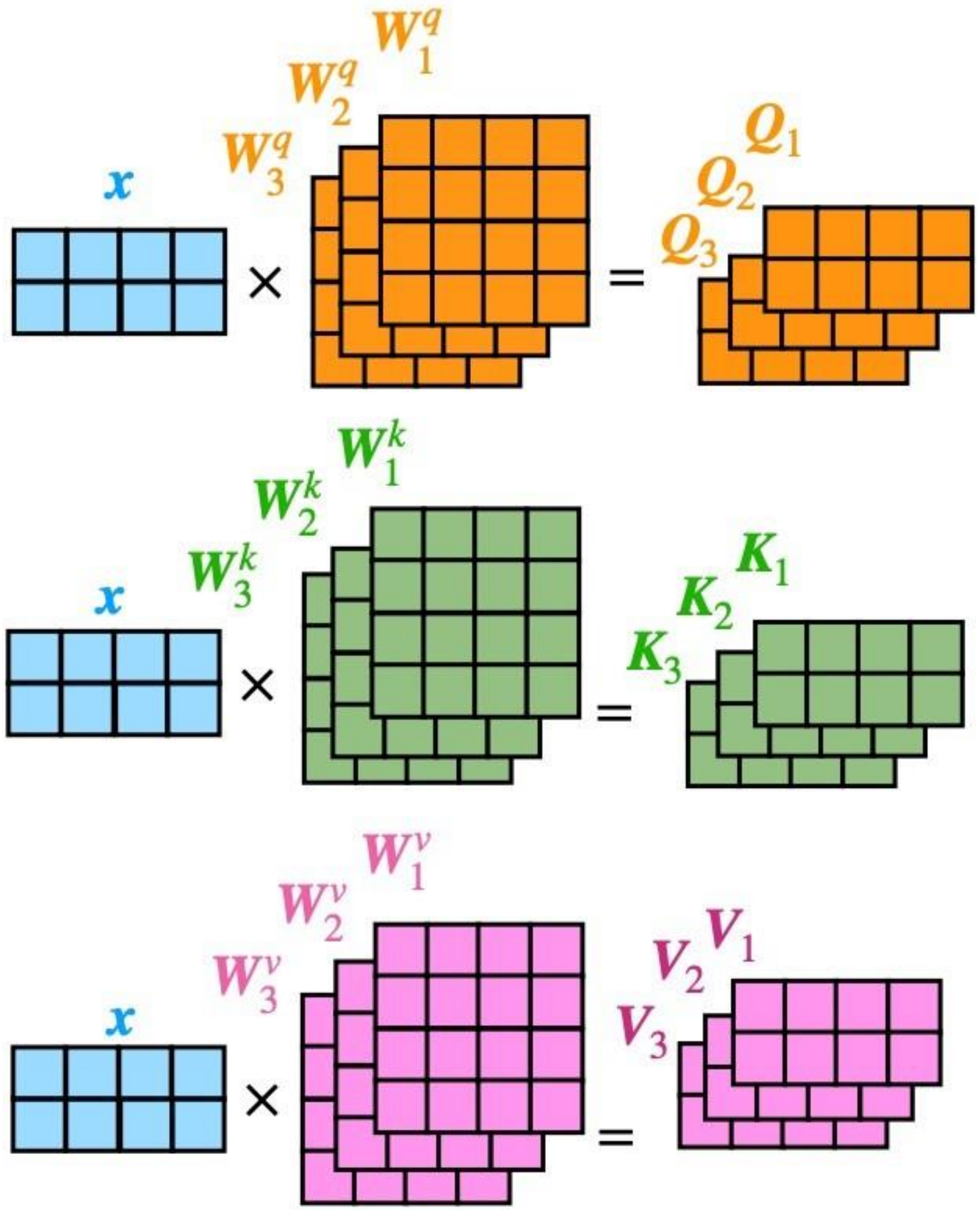
Self-Attention vs RNN: Propagates Information About Other Inputs Without Recurrent Units





Multi-head Attention

- **Goal**: enable each token to relate to other tokens in multiple ways
- **Key idea**: multiple self-attention mechanisms, each with their own key, value and query matrices

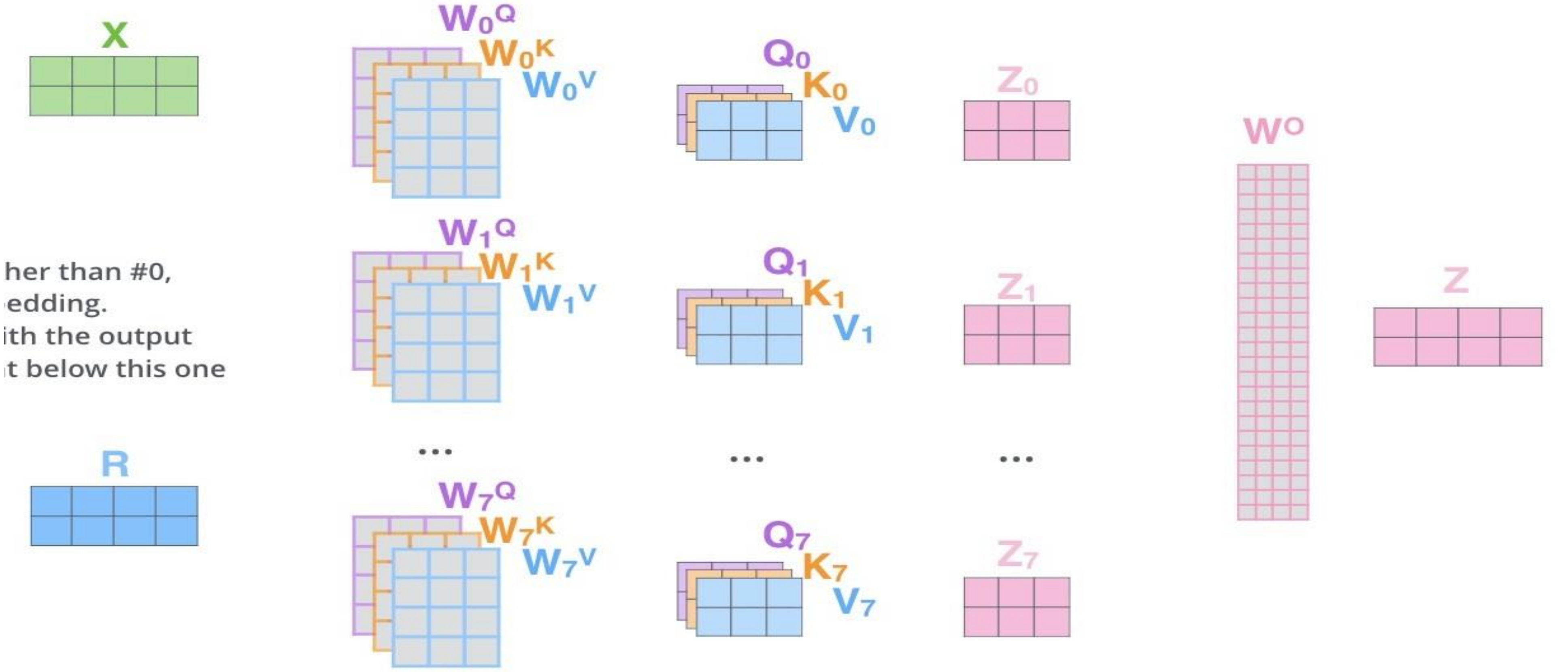


Multi-head Attention

1) Create query, key, and value vectors for all attentions heads

2) Compute new input representations

3) Condense all representations into a single representation by concatenating **z**-s and multiplying by a weight matrix



Trained Multi-head Attention Examples

Figure shows two columns of attention weights for the first two attention heads

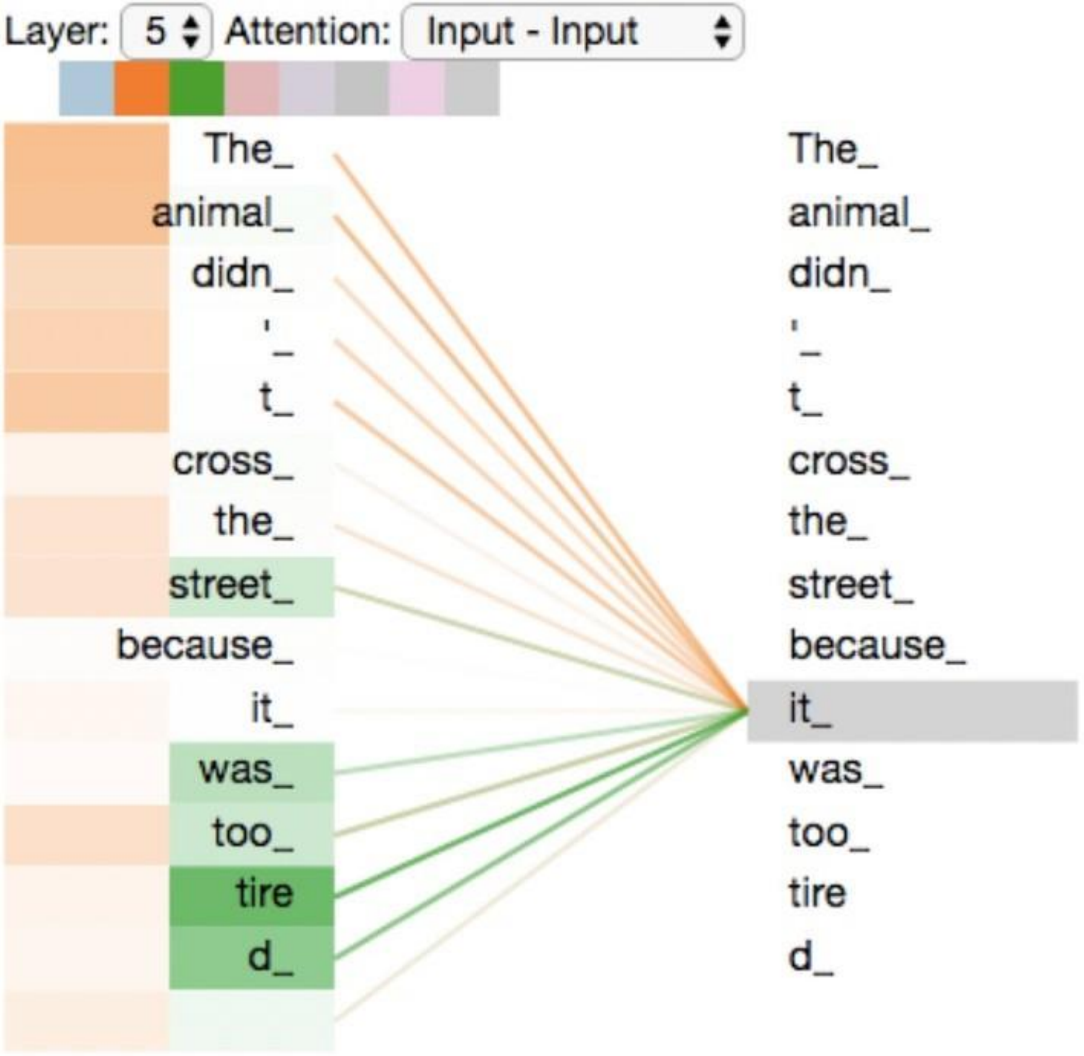
- Darker values signify larger attention scores

What does "it" focus on most in the first attention head?

- The animal (e.g., represents what is "it")

What does "it" focus on most in the second attention head?

- tired (e.g., represents how "it" feels)

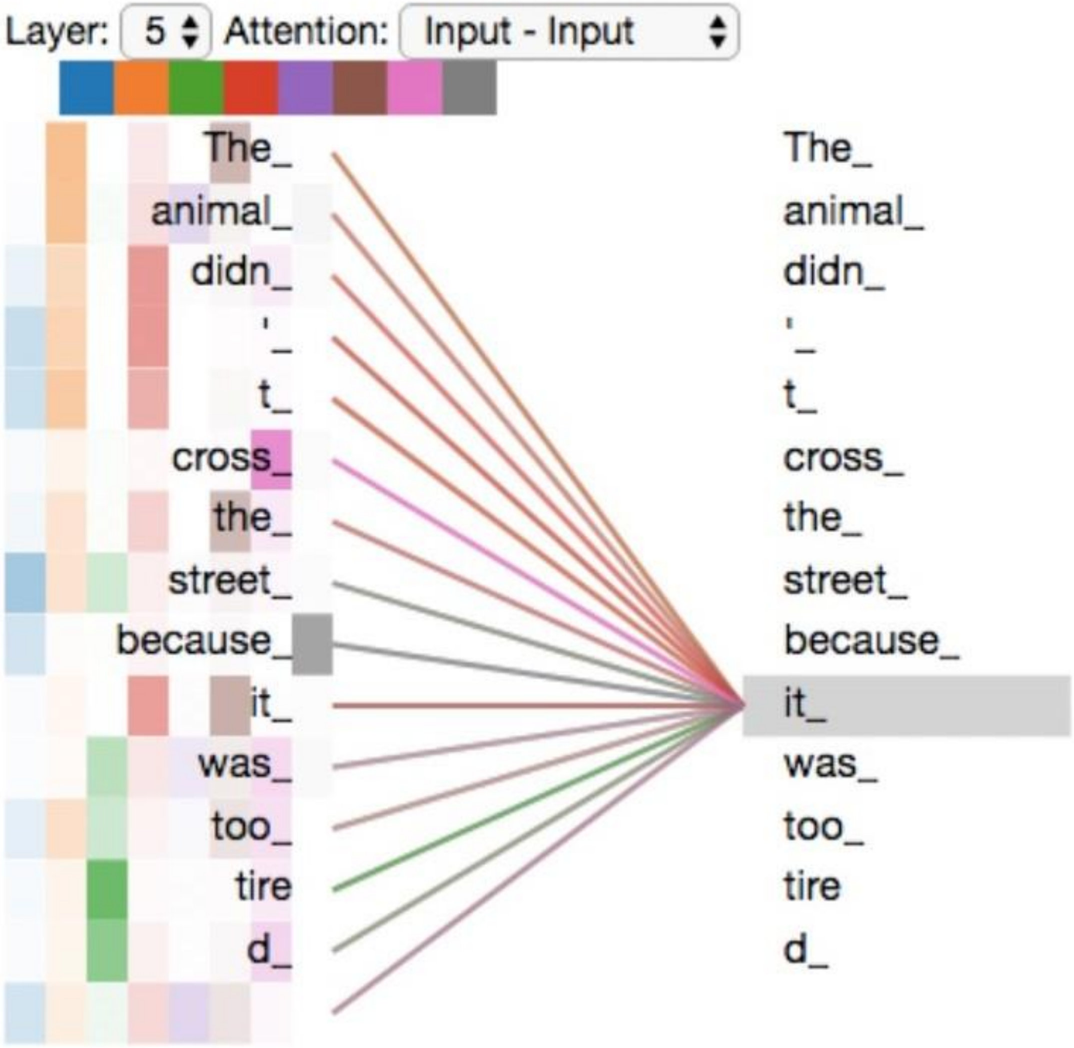


Trained Multi-head Attention Examples

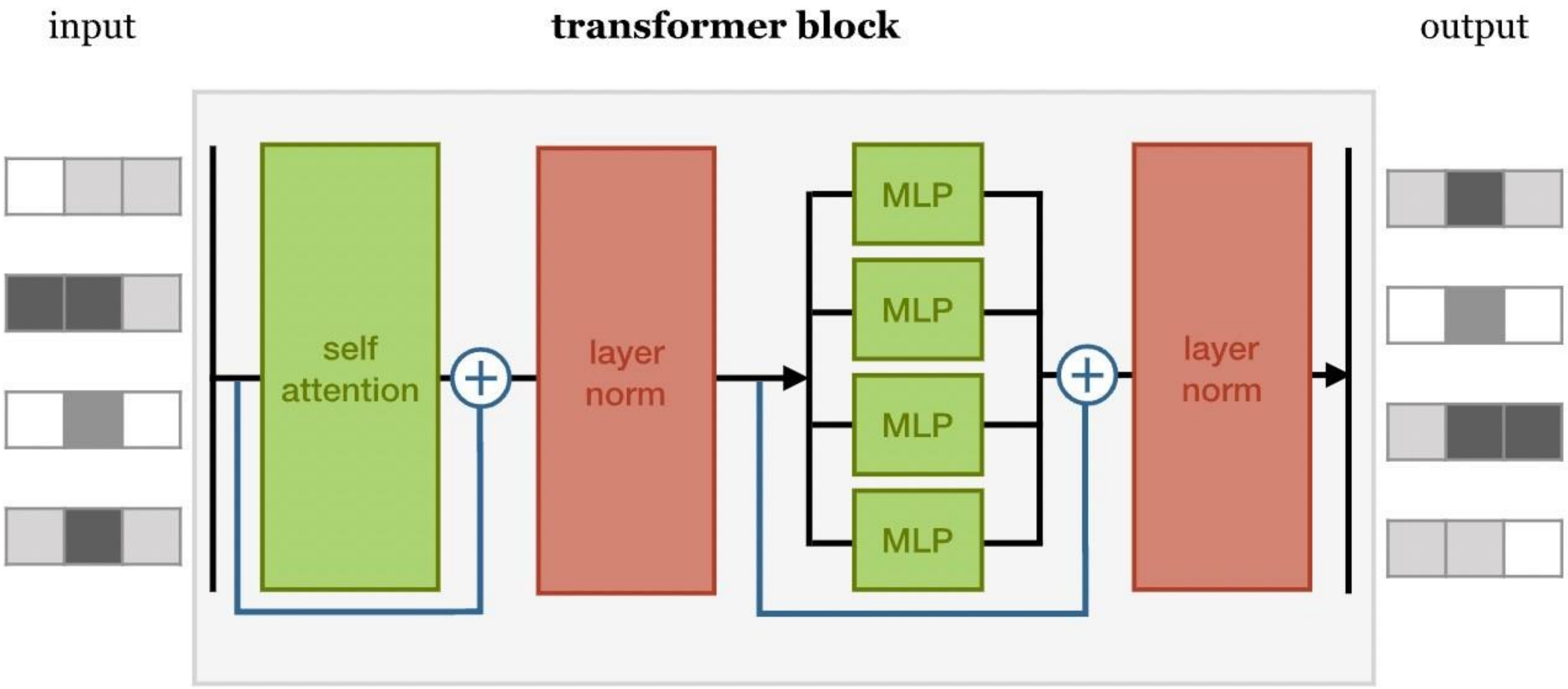
Figure shows five columns of attention weights for five attention heads

- Darker values signify larger attention scores

Attention weights may be hard to interpret

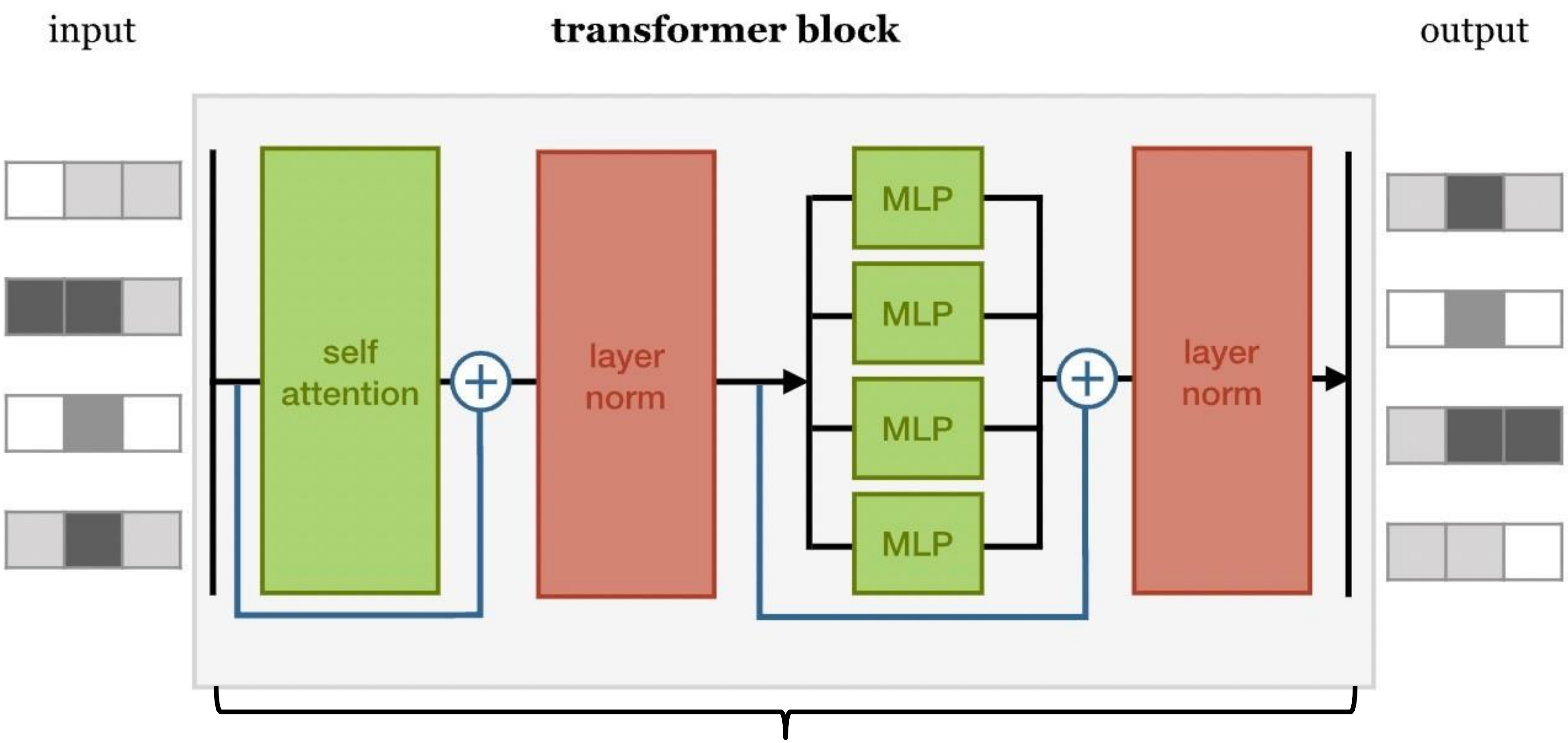


Typical Transformer Block



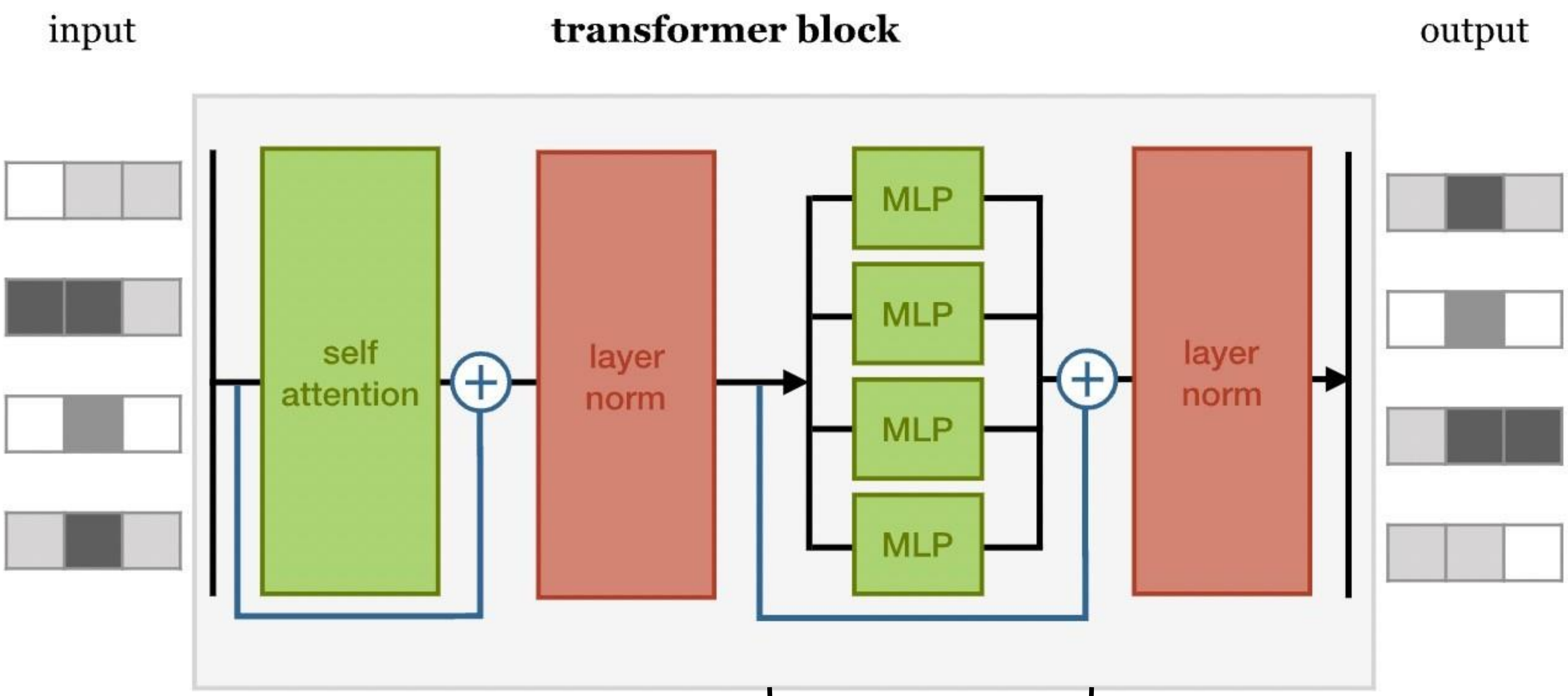
Architectures often chain together multiple transformer blocks, like that shown here

Typical Transformer Block



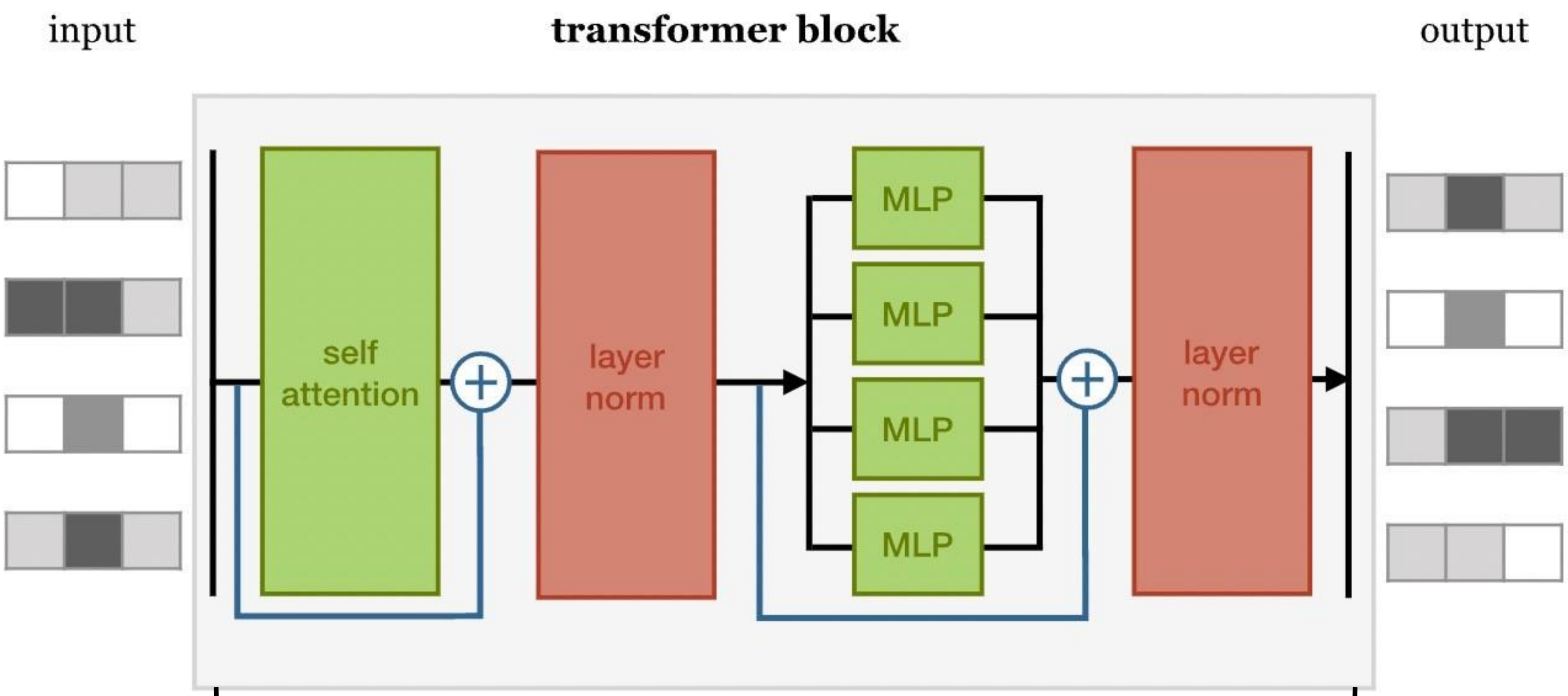
Layer normalization and residual connections improve training (i.e., faster and better results)

Typical Transformer Block



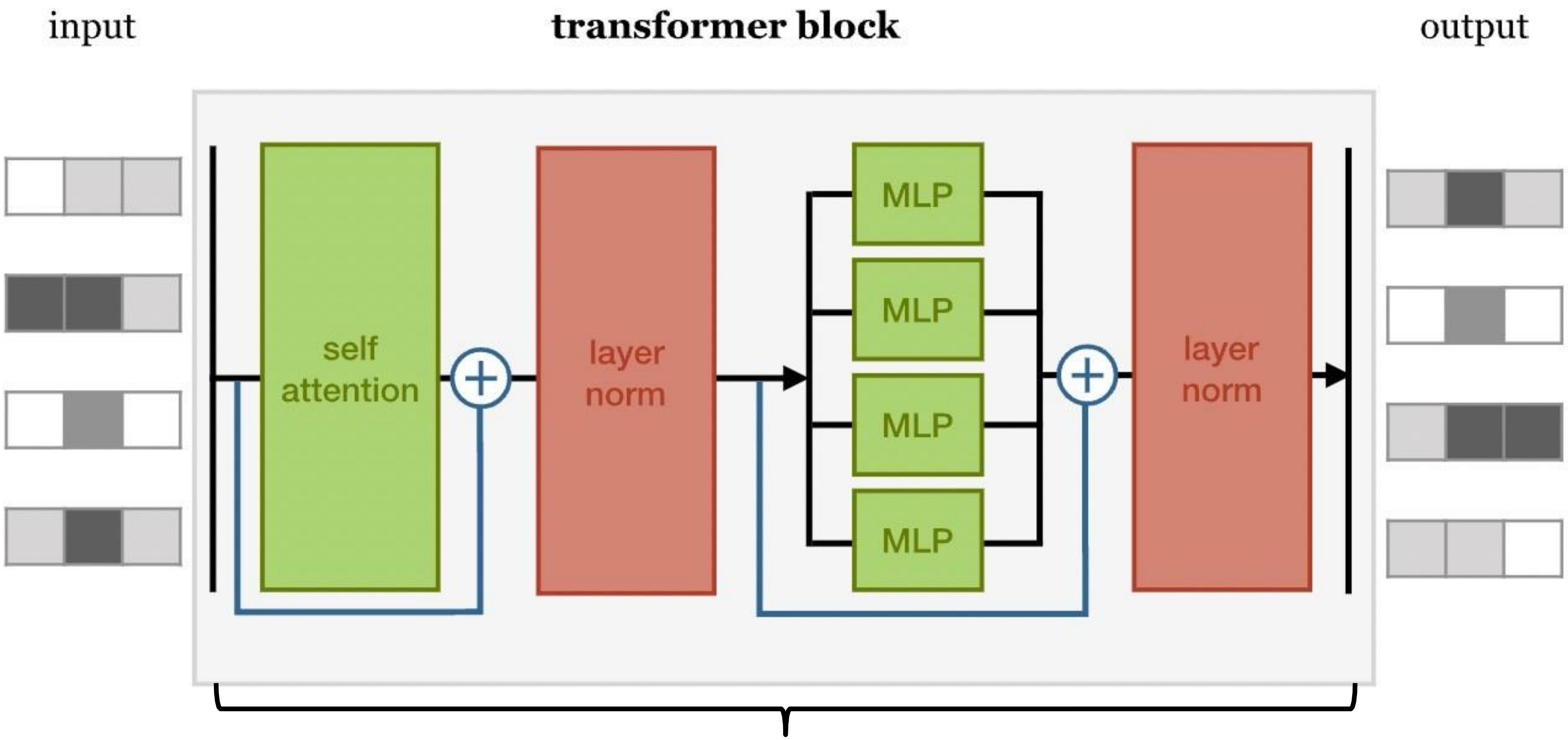
Feedforward layer per input

Typical Transformer Block



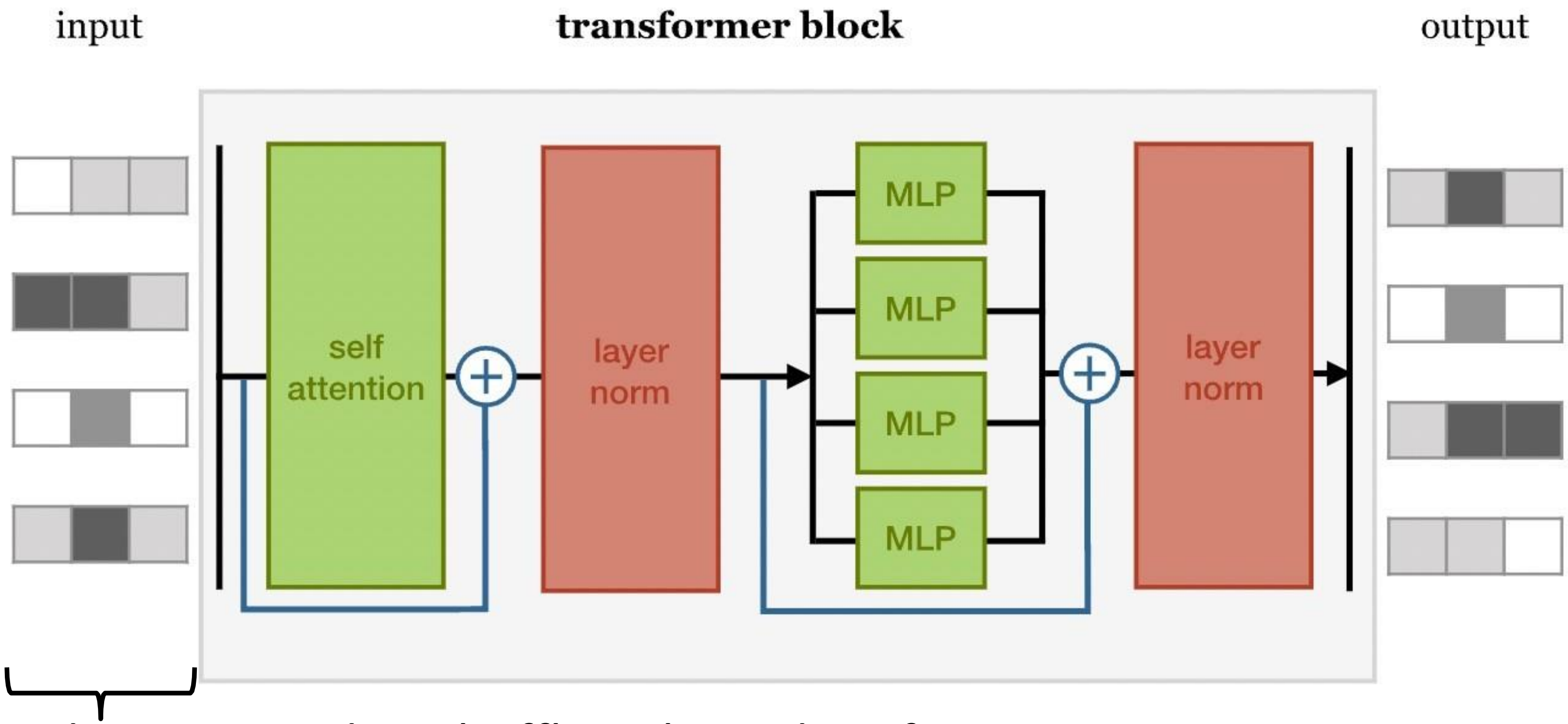
Where are non-linearities introduced in this block?

Typical Transformer Block



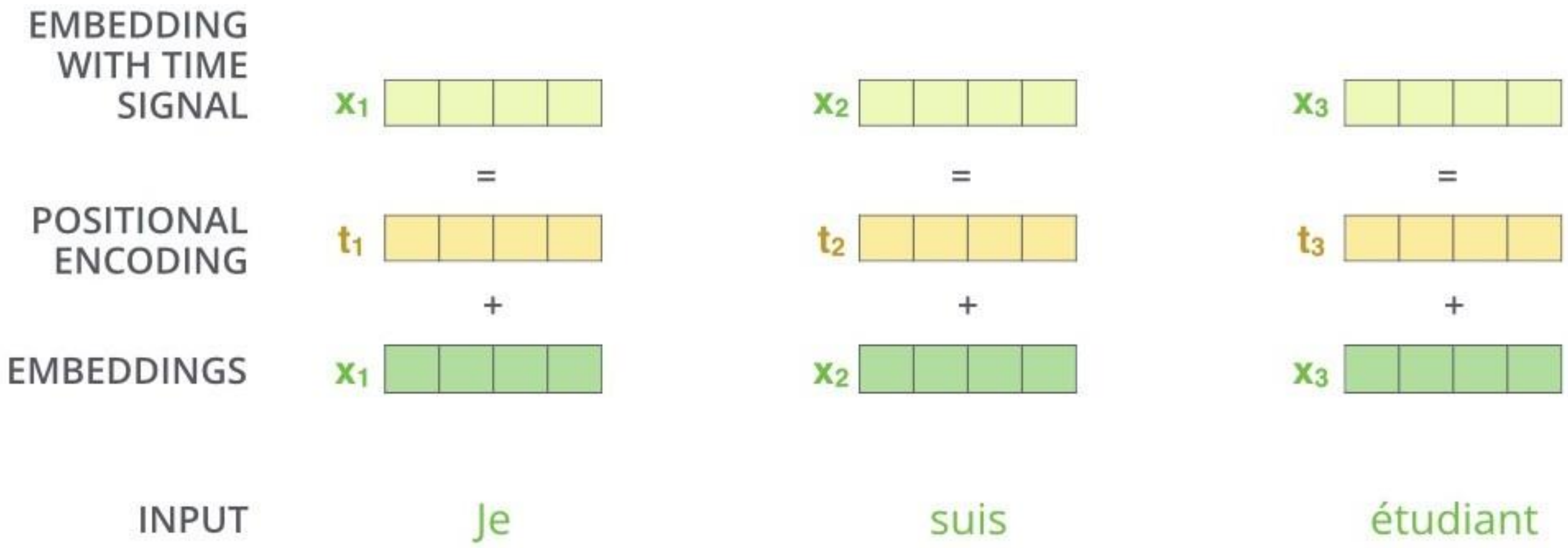
Non-linearities introduced in the softmax of self-attention, activation functions in MLP, and layer norms

Challenge: Transformers Lack Sensitivity to the Order of the Input Tokens



Input observed as a *set* and so shuffling the order of input tokens results in the same outputs except in the same shuffled order (i.e. self-attention is *permutation equivariant*)

Solution: Add Position as Input to Transformer



- Options:
  - **Position embeddings**: created by training with sequences of every length during training
  - **Position encodings**: a function mapping positions to vectors that the network learns to interpret (enables generalization to lengths not observed during training)