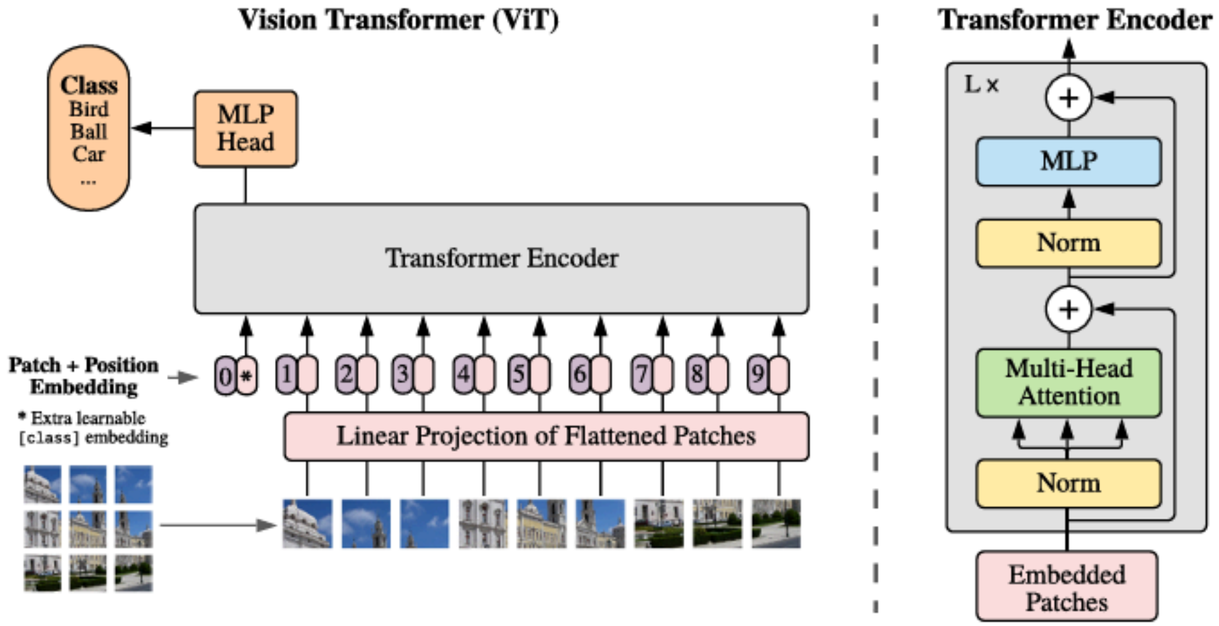
ViT

Use a **Pre-Trained Transformer Based Model** to help improve performance



Published as a conference paper at ICLR 2021

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

Alexey Dosovitskiy*,†, Lucas Beyer*, Alexander Kolesnikov*, Dirk Weissenborn*, Xiaohua Zhai*, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby*,†

*equal technical contribution, †equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

ABSTRACT

While the Transformer architecture has become the de-facto standard for natural language processing tasks, its applications to computer vision remain limited. In vision, attention is either applied in conjunction with convolutional networks, or used to replace certain components of convolutional networks while keeping their overall structure in place. We show that this reliance on CNNs is not necessary and a pure transformer applied directly to sequences of image patches can perform very well on image classification tasks. When pre-trained on large amounts of data and transferred to multiple mid-sized or small image recognition benchmarks (ImageNet, CIFAR-100, VTAB, etc.), Vision Transformer (ViT) attains excellent results compared to state-of-the-art convolutional networks while requiring substantially fewer computational resources to train.[1]

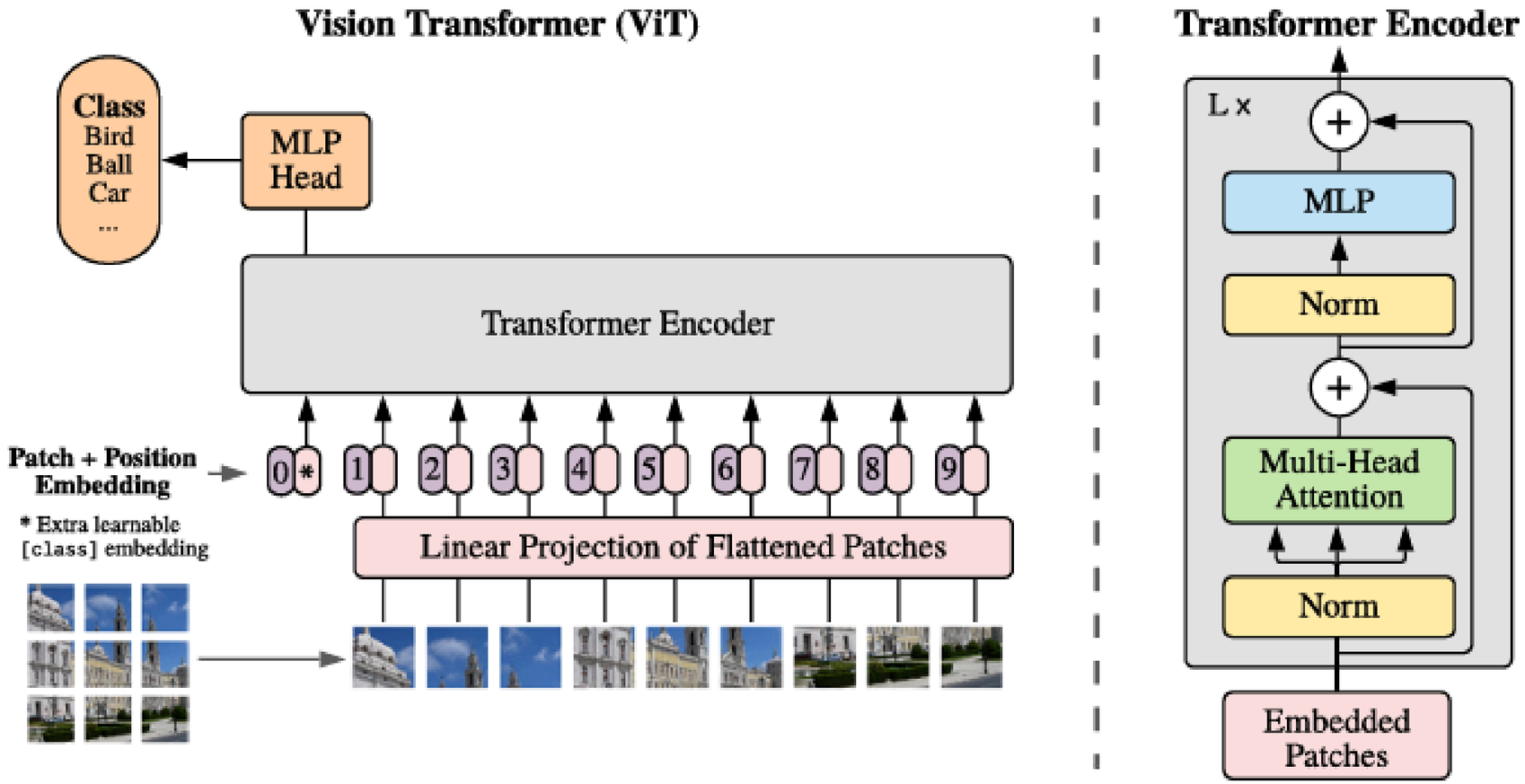
1 INTRODUCTION

Self-attention-based architectures, in particular Transformers (Vaswani et al., 2017), have become the model of choice in natural language processing (NLP). The dominant approach is to pre-train on a large text corpus and then fine-tune on a smaller task-specific dataset (Devlin et al., 2019). Thanks

What is ViT?

Inspired by Transformers, we **split an image into patches** and **provide the sequence of linear embeddings of these patches** as an **input to a Transformer**.

Image patches are treated the same way as **tokens** (words) in an NLP application



1

Input Image

224

224

$$N = \frac{HW}{P^2}$$

14 x 14

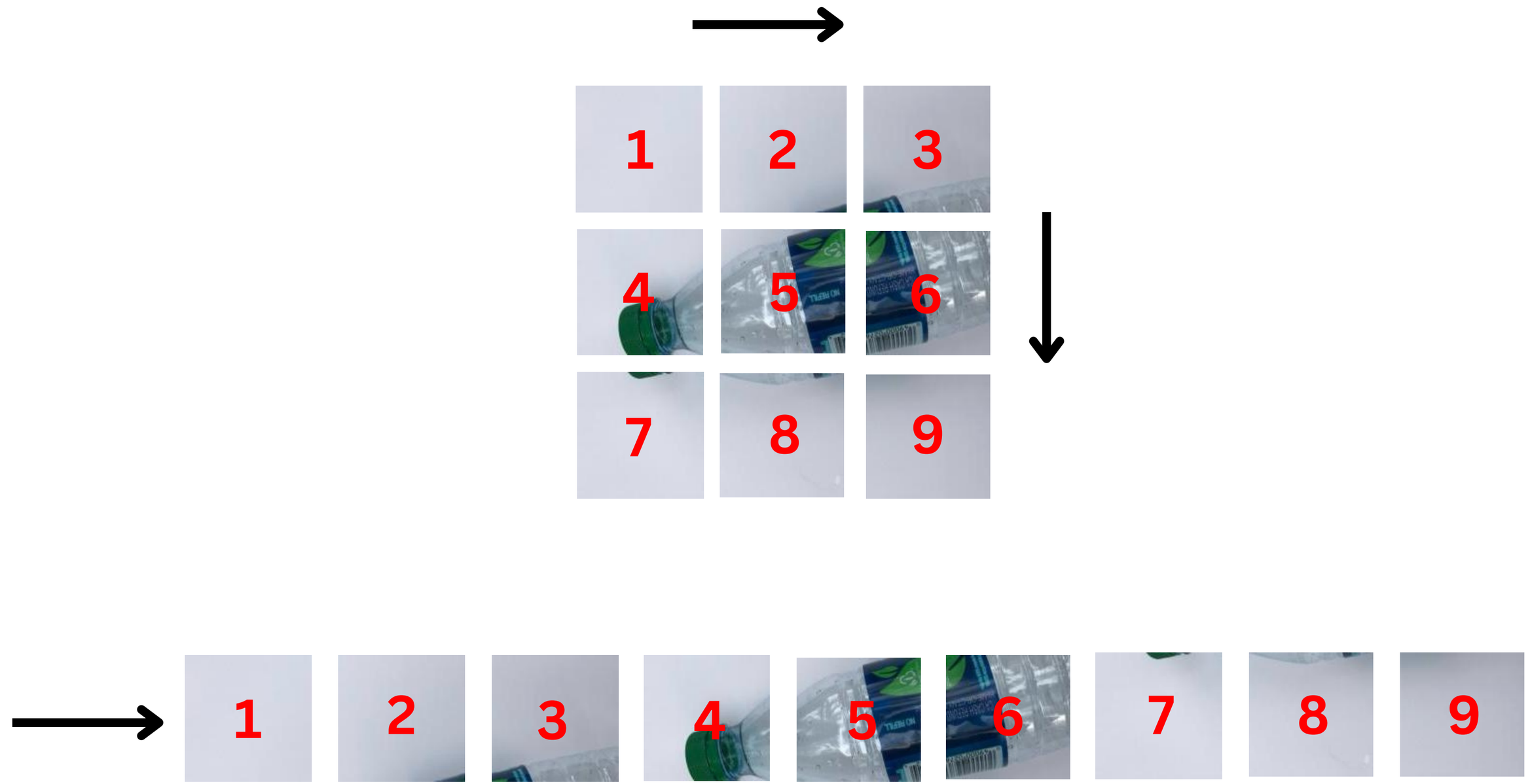
Image Patches

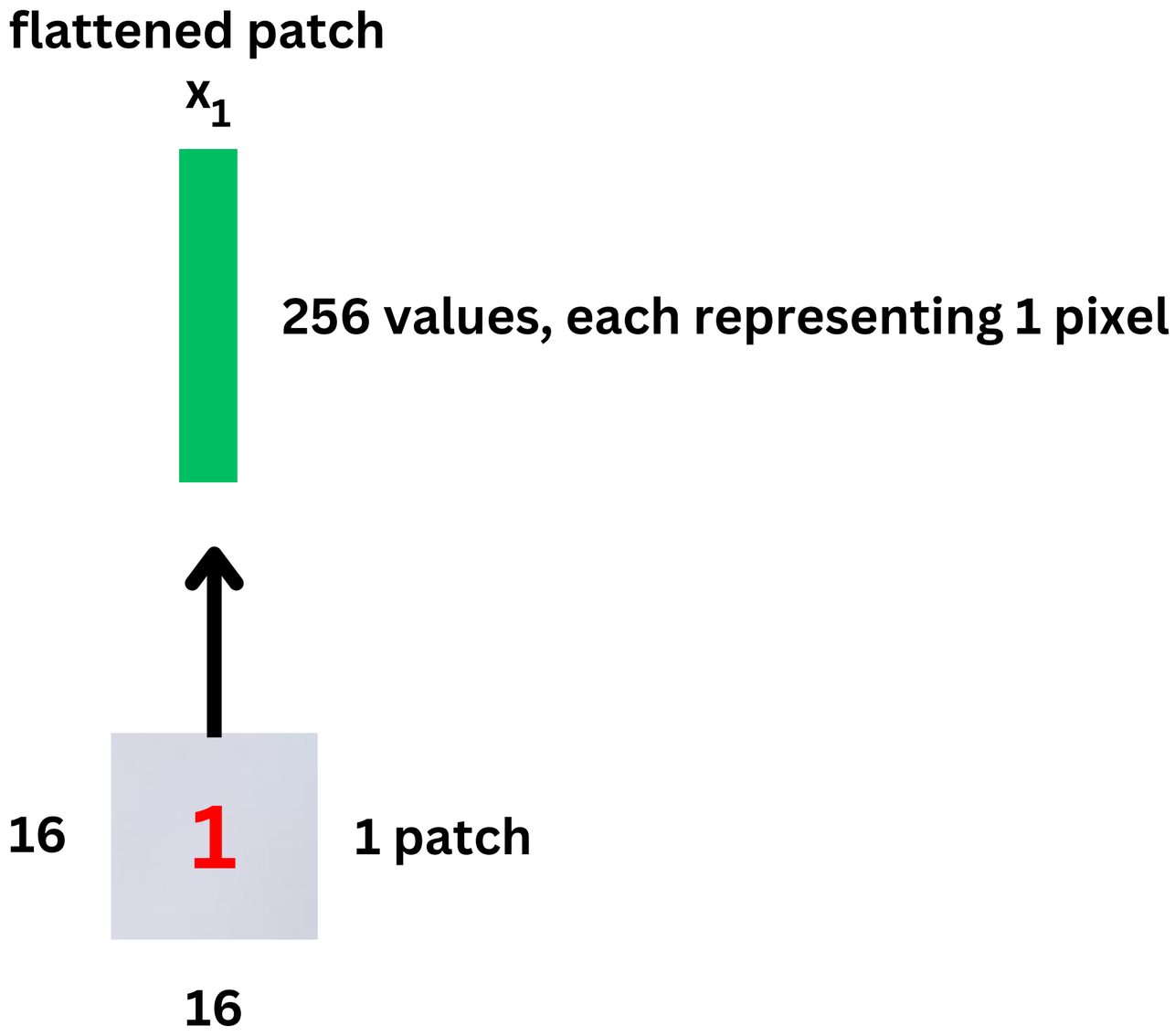
16

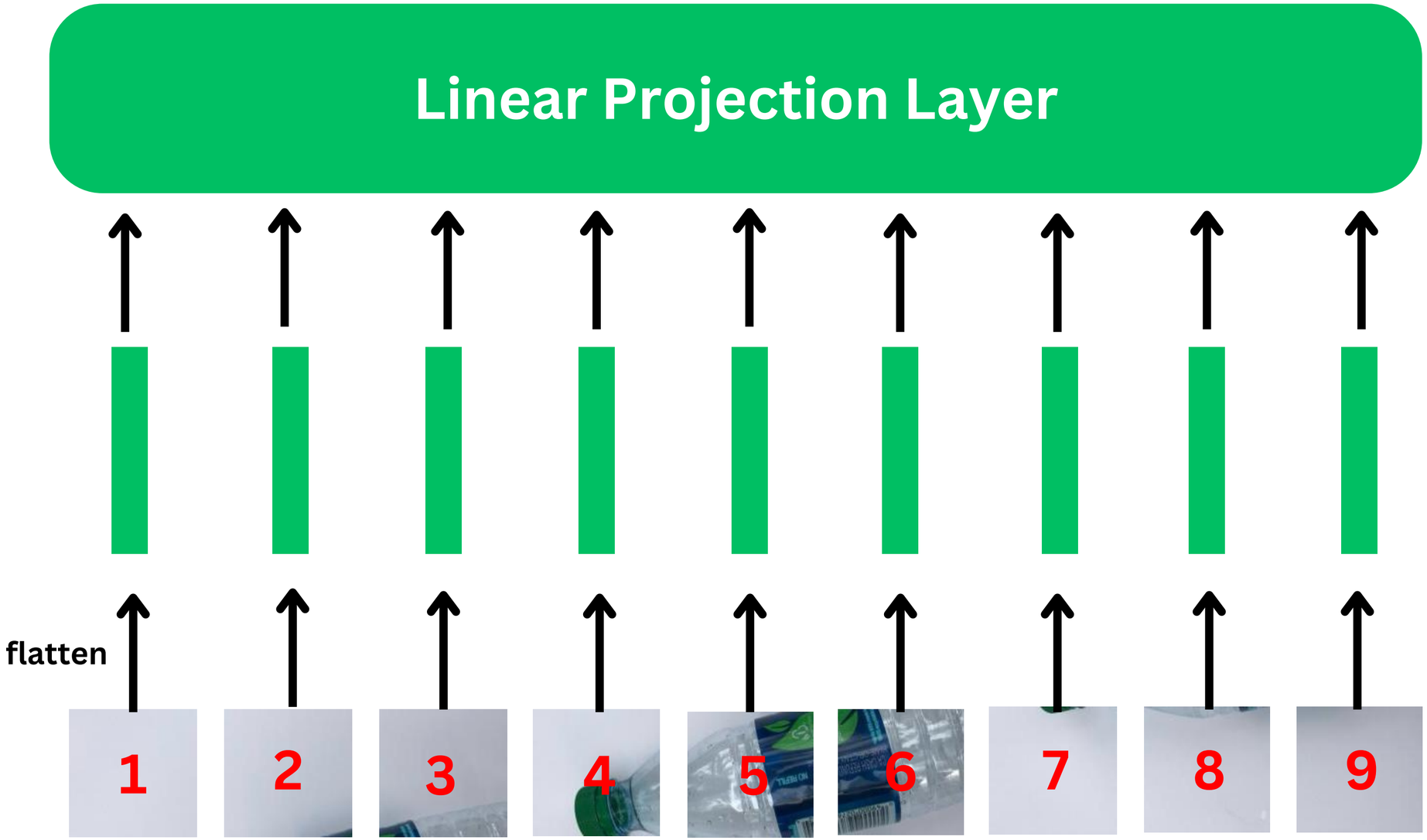
16

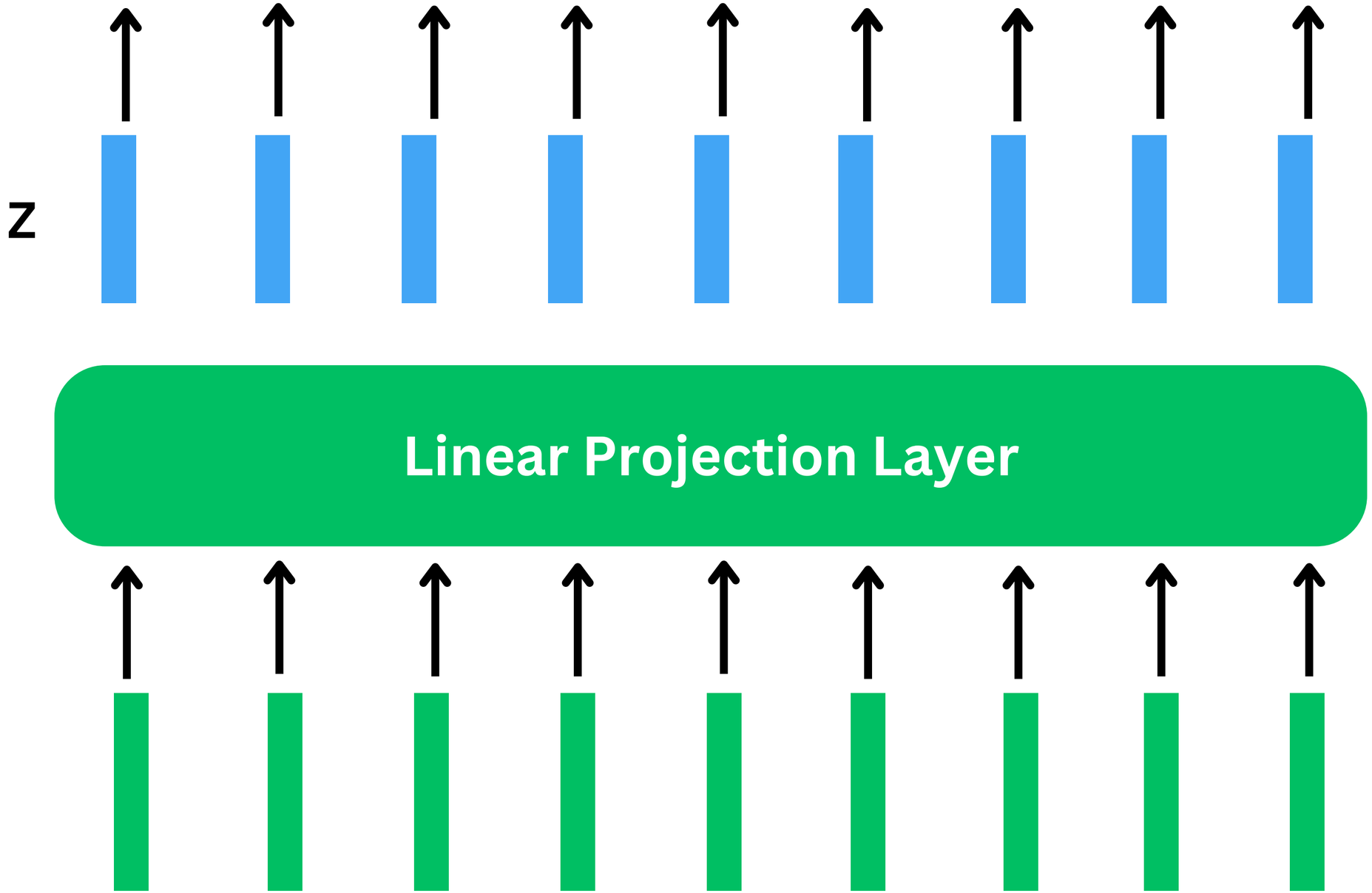
16

16 16 16

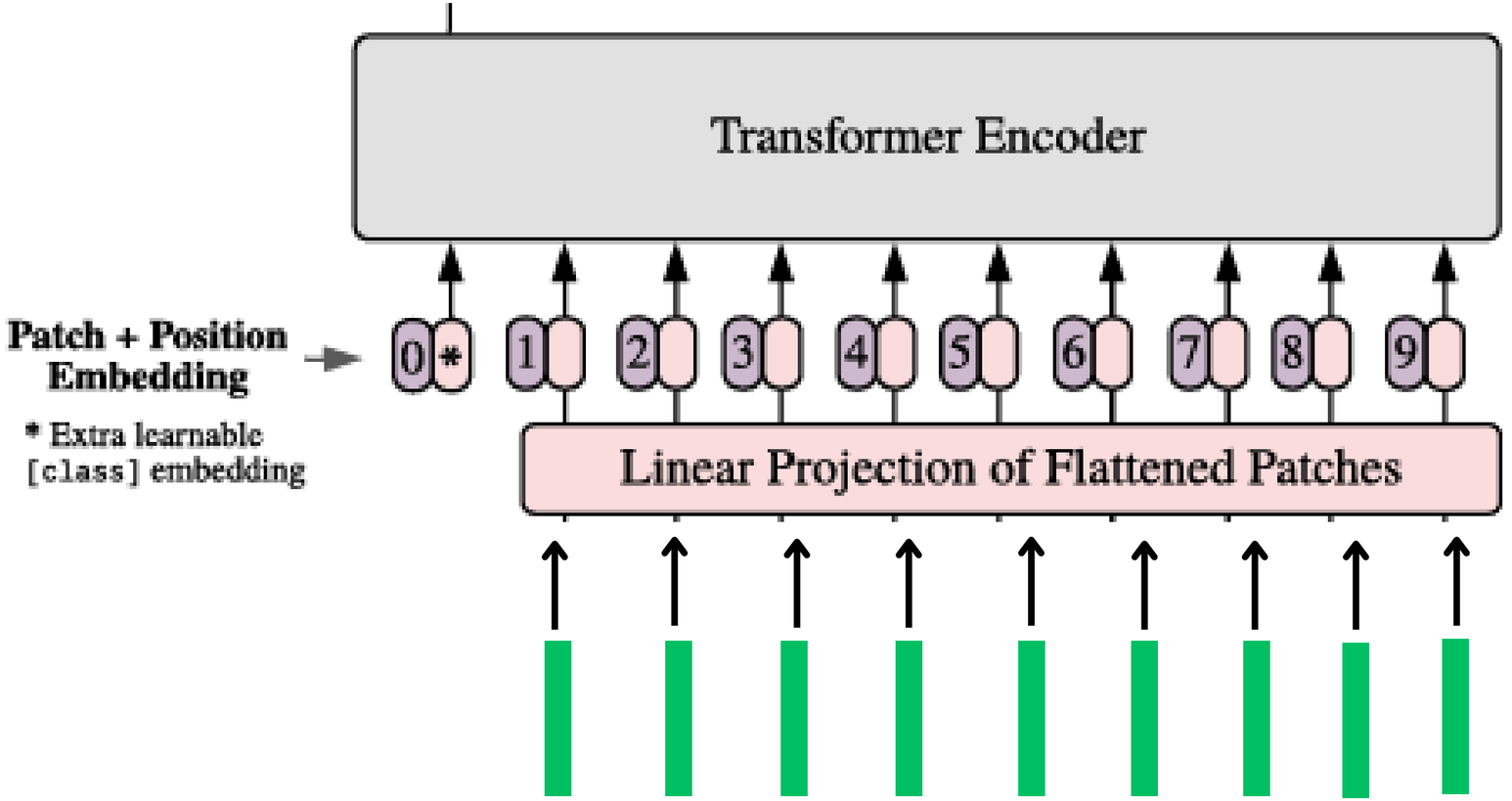

1
2
3
4
5
6
7
8
9
1
2
3
4
5
6
7
8
9

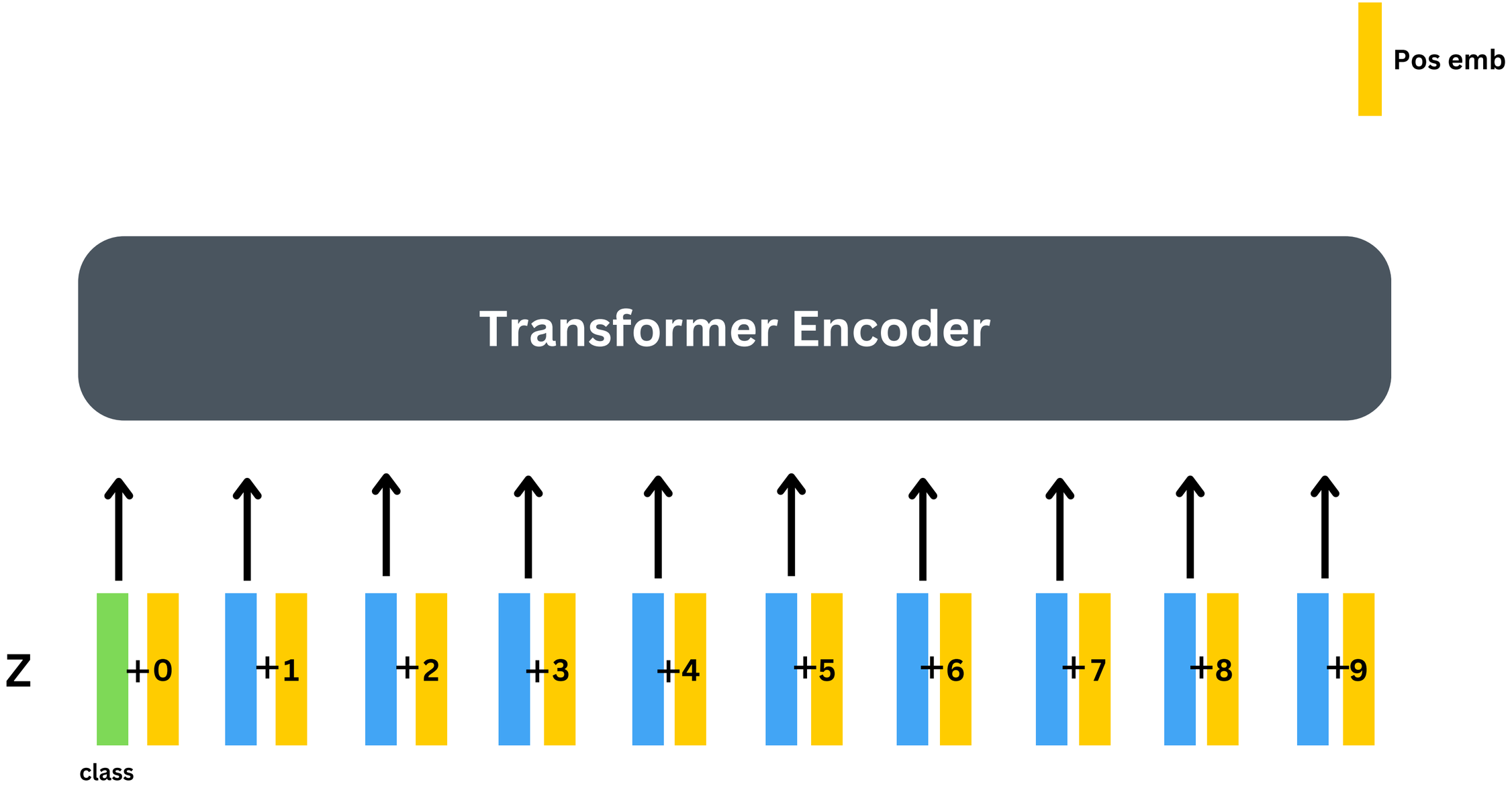

flattened patch
x_1
256 values, each representing 1 pixel
16
1
1 patch
16

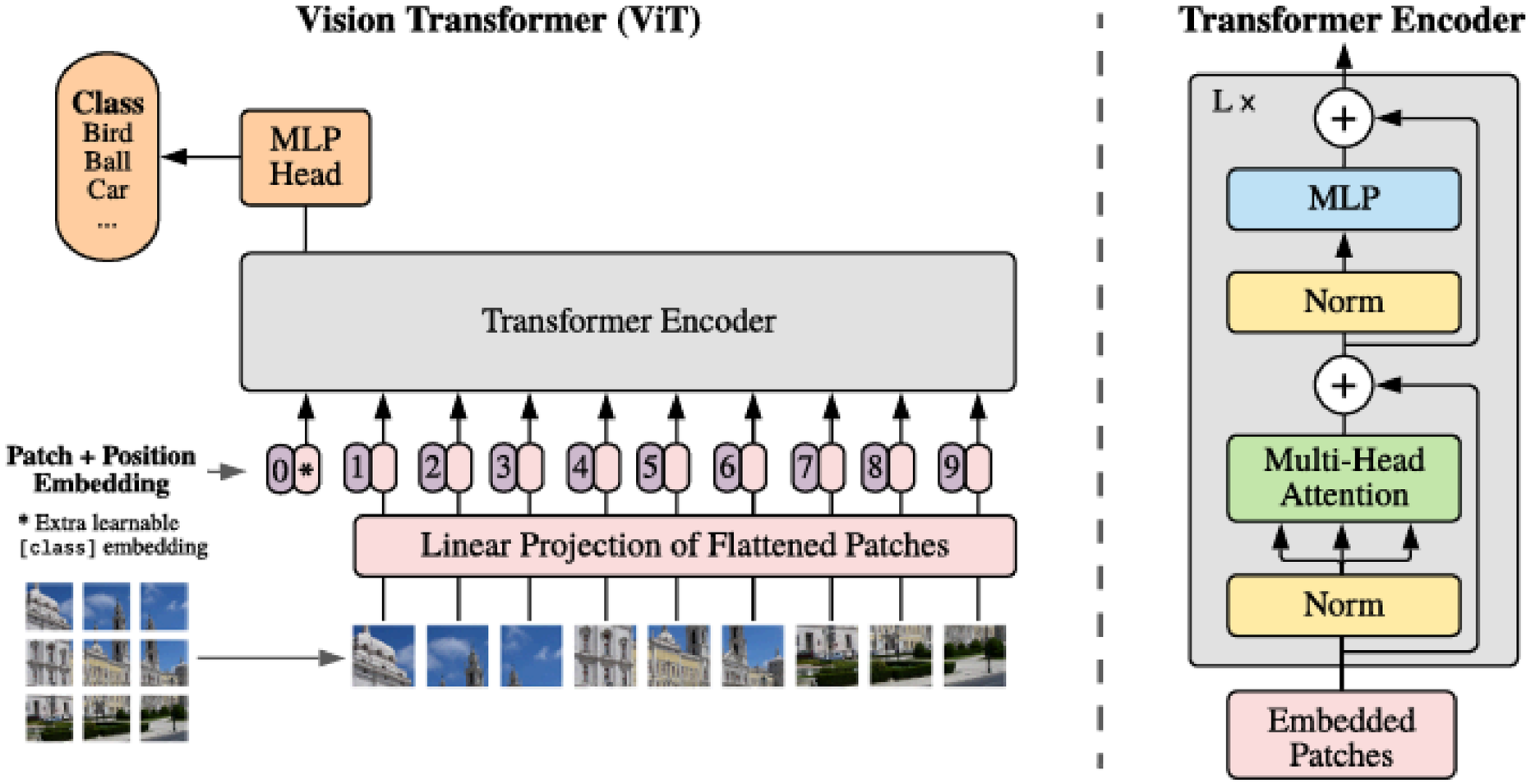

Linear Projection Layer
flatten
1
2
3
4
5
6
7
8
9


z
Linear Projection Layer

Positional Embeddings and Class Embedding



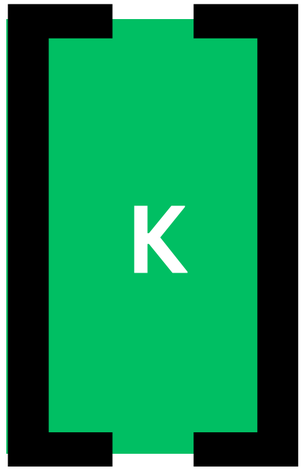

Pos emb
Transformer Encoder
Z
+0
+1
+2
+3
+4
+5
+6
+7
+8
+9
class


Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP
Head
Transformer Encoder
Patch + Position
Embedding
* Extra learnable
[class] embedding
0 * 1 2 3 4 5 6 7 8 9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head
Attention
Norm
Embedded
Patches

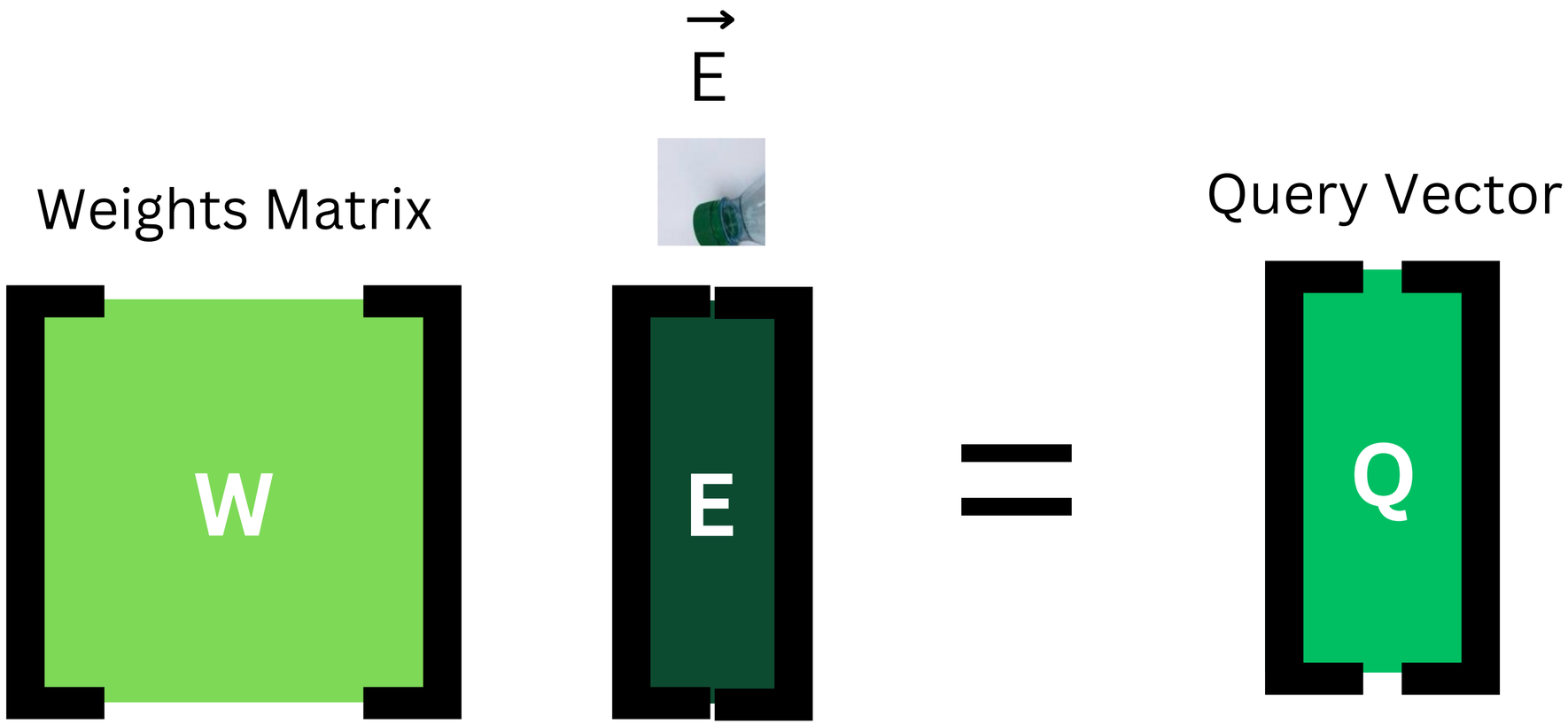
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

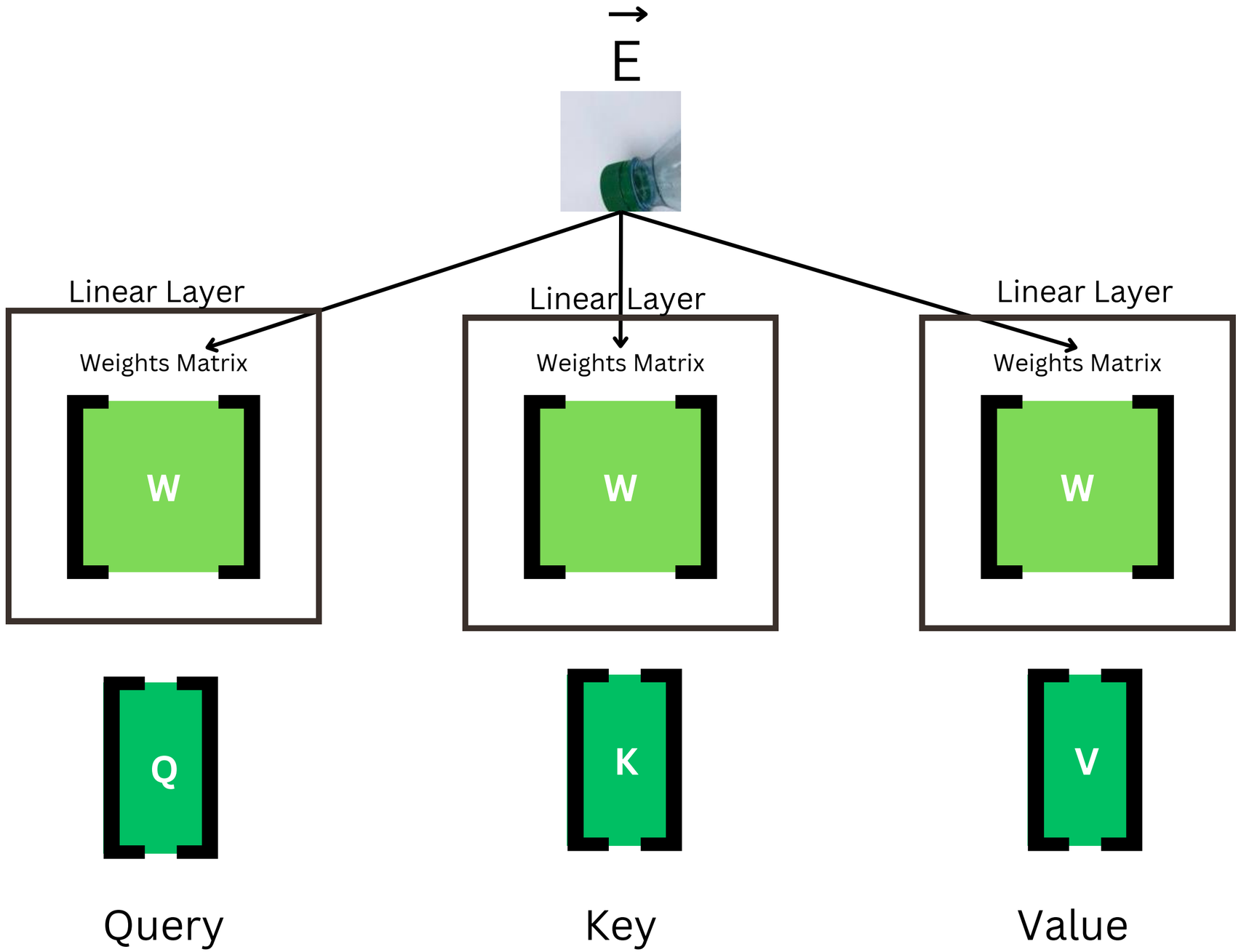
Query

Key

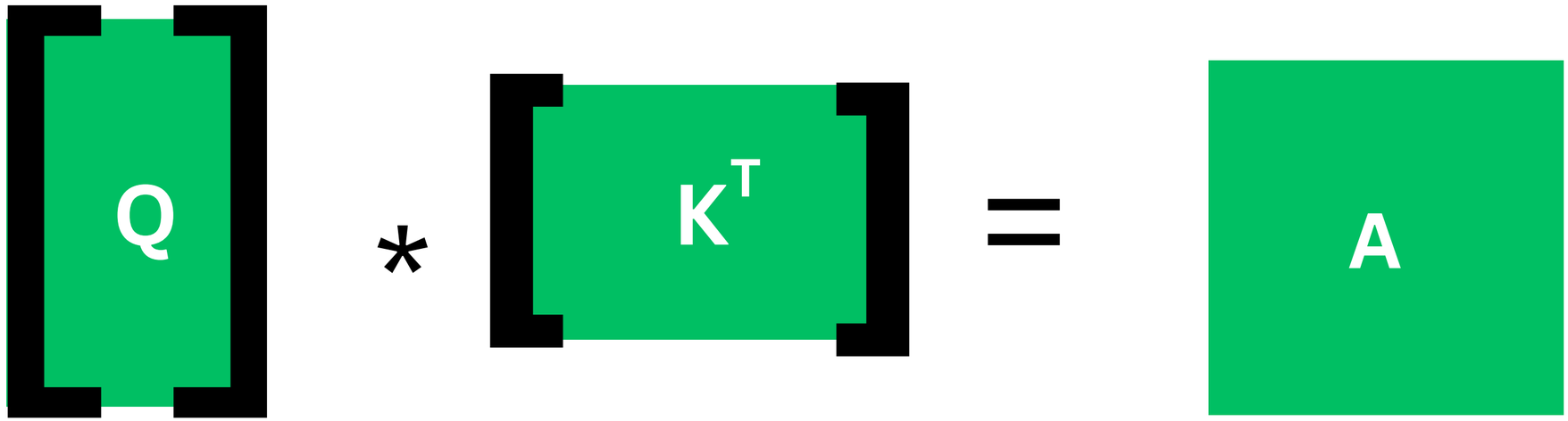


Value


Weights Matrix
W
E
E
=
Query Vector
Q


E
Linear Layer
Weights Matrix
W
Linear Layer
Weights Matrix
W
Linear Layer
Weights Matrix
W
Q
K
V
Query
Key
Value

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

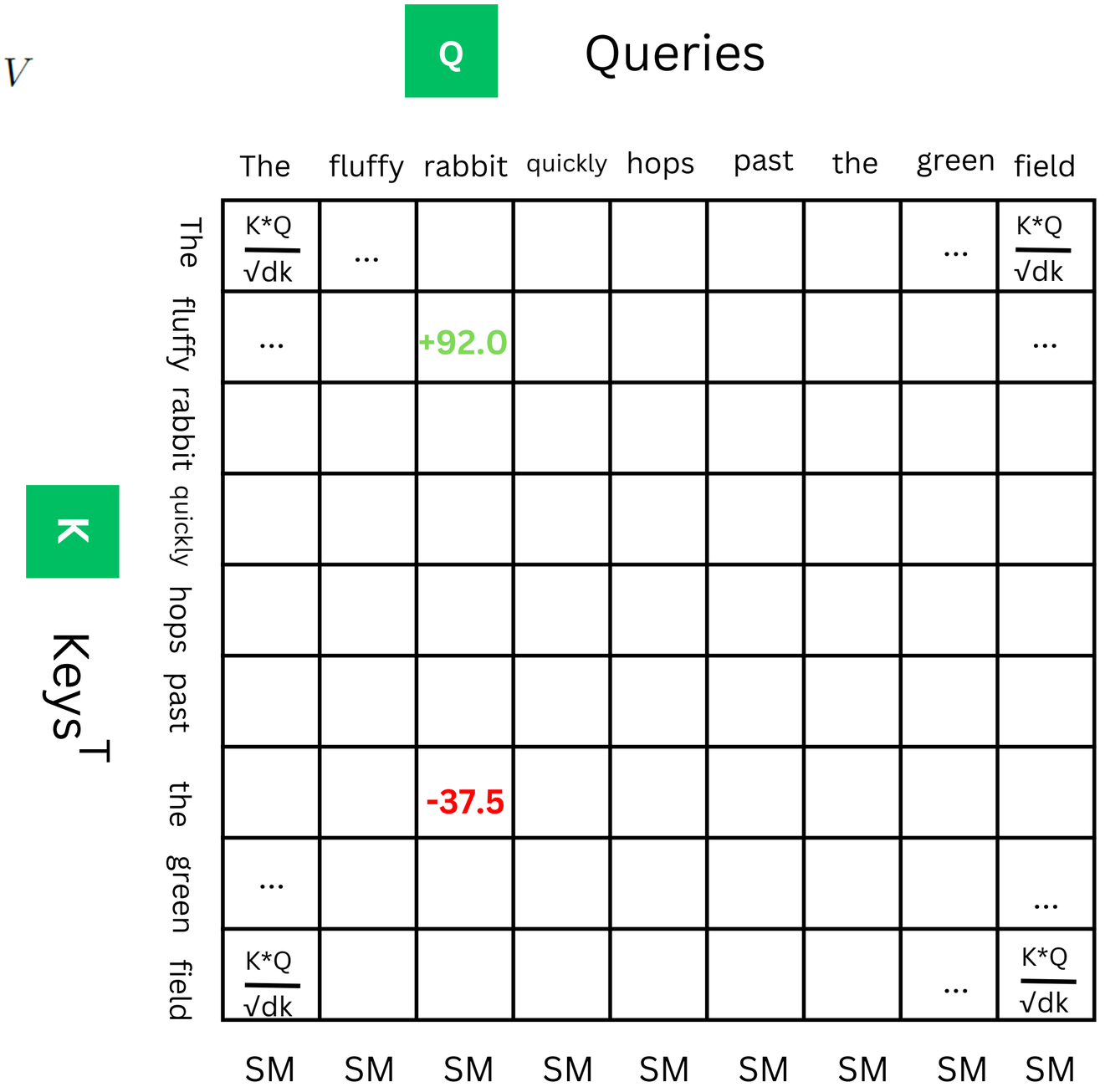


Query Key ^{T} Attn

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

Sentence:

The fluffy rabbit quickly hops past the green field



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

Image:

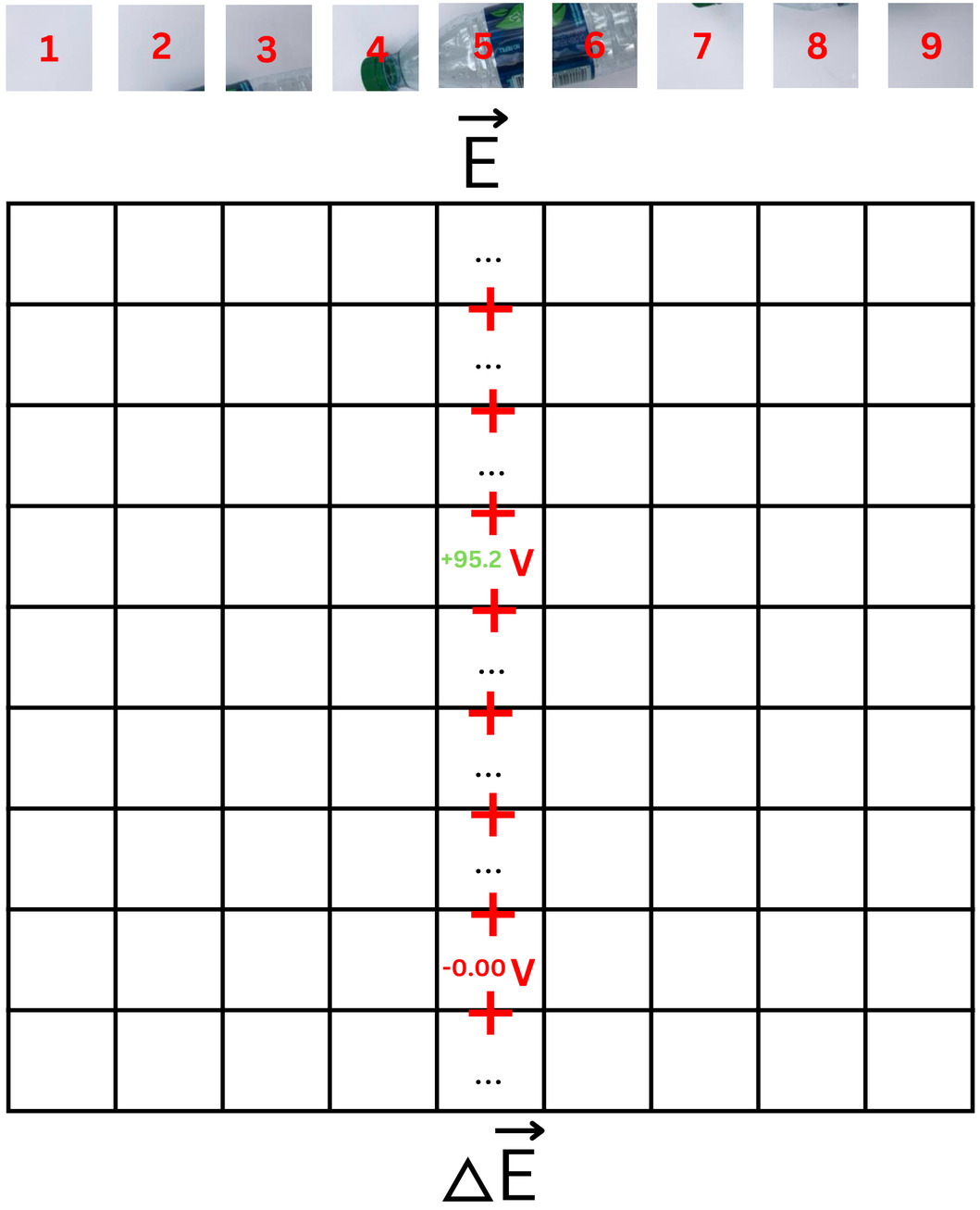
Q

Queries

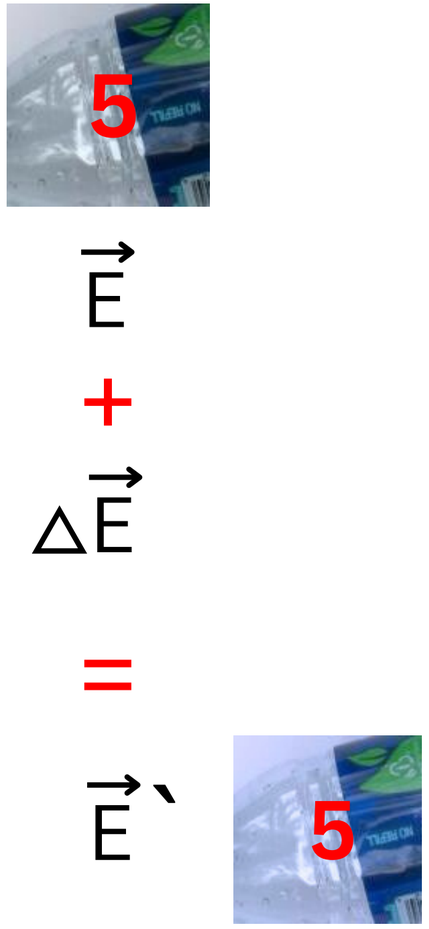
K

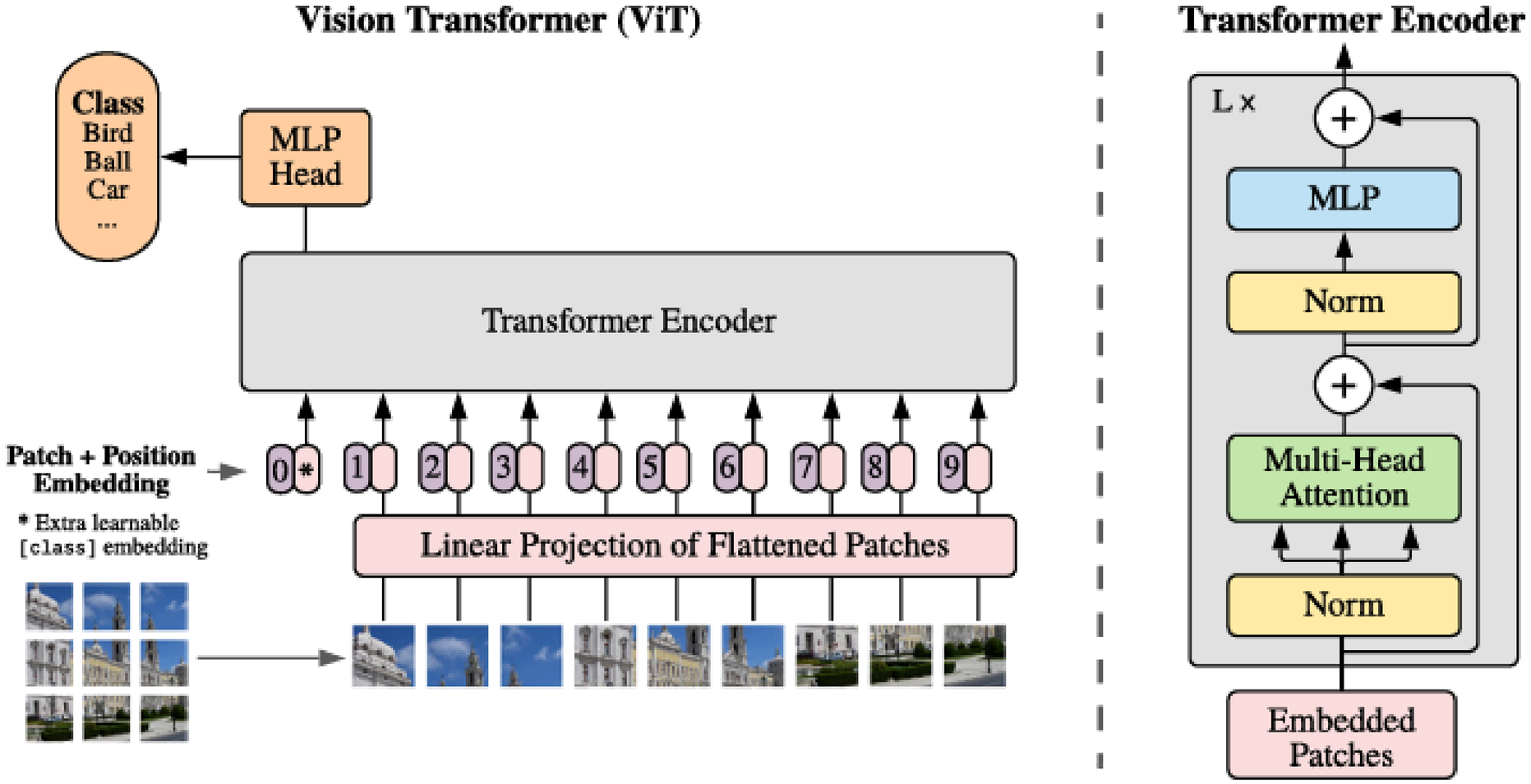
$Keys^T$

$\frac{K*Q}{\sqrt{dk}}$	...						...	$\frac{K*Q}{\sqrt{dk}}$
...								...
				+95.2				
...				-22.4				...
$\frac{K*Q}{\sqrt{dk}}$	...						...	$\frac{K*Q}{\sqrt{dk}}$
SM	SM	SM	SM	SM	SM	SM	SM	SM

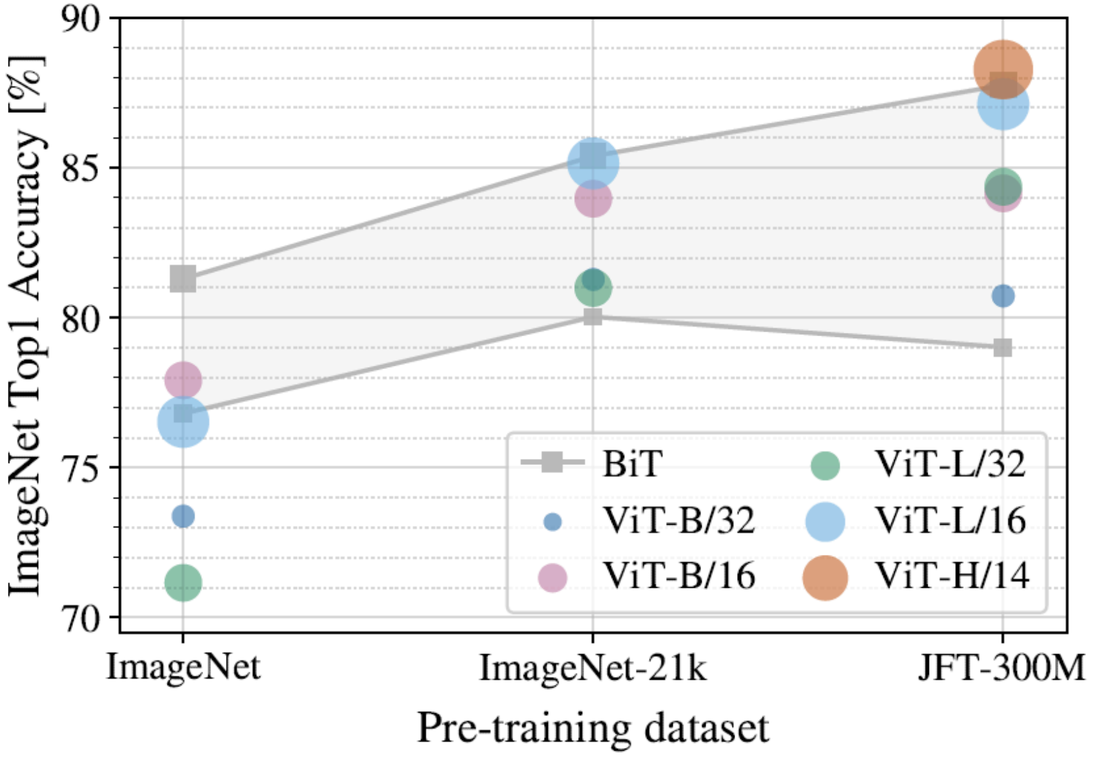


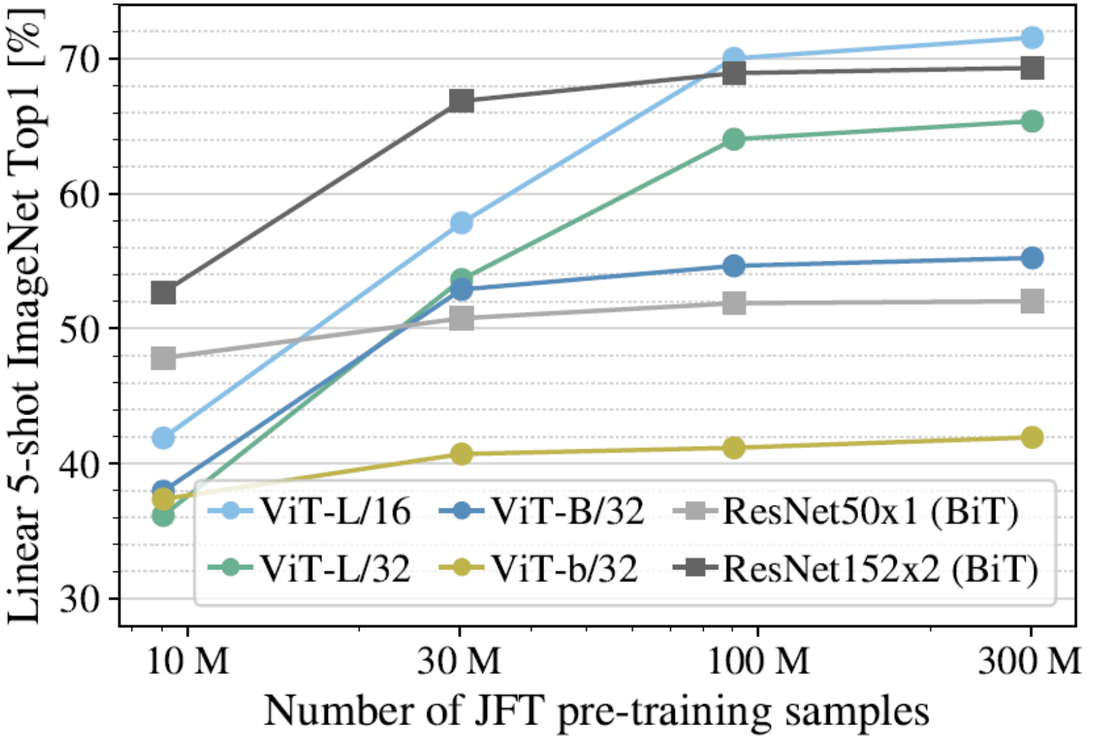
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$




Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP
Head
Transformer Encoder
Patch + Position
Embedding
* Extra learnable
[class] embedding
0 * 1 2 3 4 5 6 7 8 9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head
Attention
Norm
Embedded
Patches

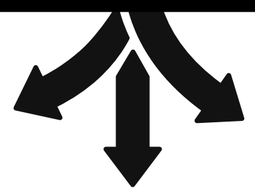
Model Performance





ViT Model Experimentation

Train, Validation & Test Split



70%
Training Set

15%
Test Set

15%
Validation Set

Test Accuracy

97%

Computational Time

~1 Hrs
T4 GPU

Fine-Tuning

- Learning Rate = 0.0002
- Batch Size = 16
- Epochs = 4

Advantages

- Ease of Parallelization
- Scalability
- Transfer Learning Capabilities