

**COPYRIGHT WARNING:** Copyright in these original lectures is owned by Monash University. You may transcribe, take notes, download or stream lectures for the purpose of your research and study only. If used for any other purpose, (excluding exceptions in the Copyright Act 1969 (Cth)) the University may take legal action for infringement of copyright.

Do not share, redistribute, or upload the lecture to a third party without a written permission!

# FIT3181/5215 Deep Learning

Week 10: Vision Transformer and Model Fine-Tuning Techniques

**Lecturer: Trung Le**

Email: [trunglm@monash.edu](mailto:trunglm@monash.edu)

# Outline

- Revision of Transformers
- Vision Transformer
- Swin Transformer
- Model Fine-Tuning for Transformer
  - Prompt-Tuning
  - LoRA
  - Adapter
- Acknowledgment:
  - [https://web.eecs.umich.edu/~justincj/slides/eecs498/WI2022/598\\_WI2022\\_lecture18.pdf](https://web.eecs.umich.edu/~justincj/slides/eecs498/WI2022/598_WI2022_lecture18.pdf)
  - <https://www.slideshare.net/slideshow/transfomers-in-vision-from-zero-to-hero-dlippptx/253343336>

# What do you think when you hear the word «Transformer»?

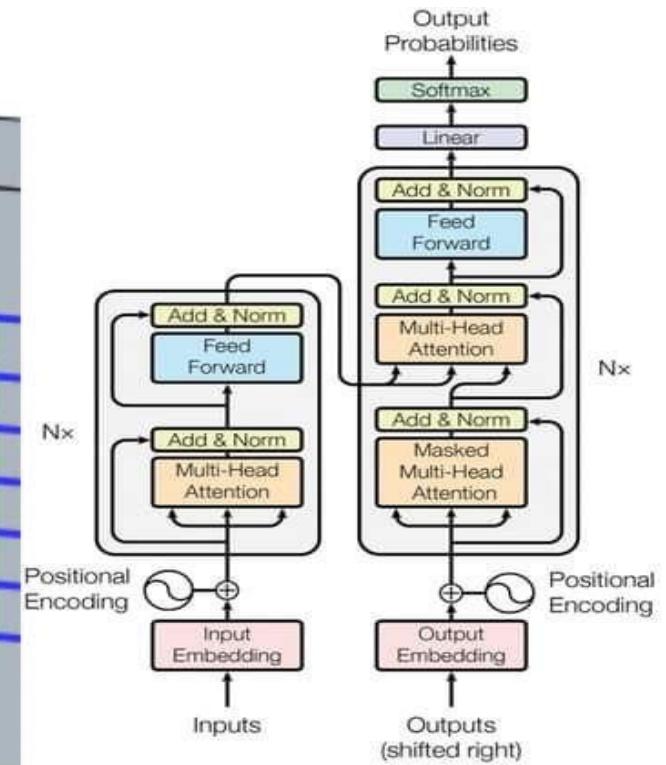
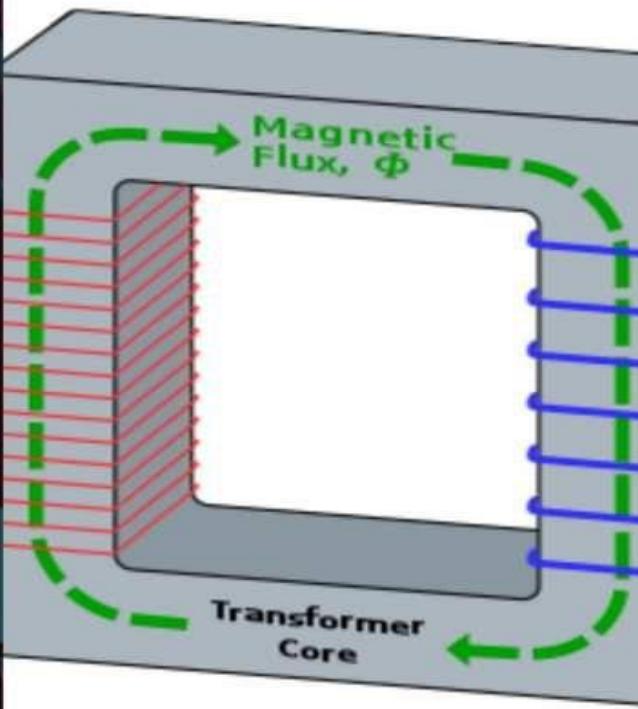


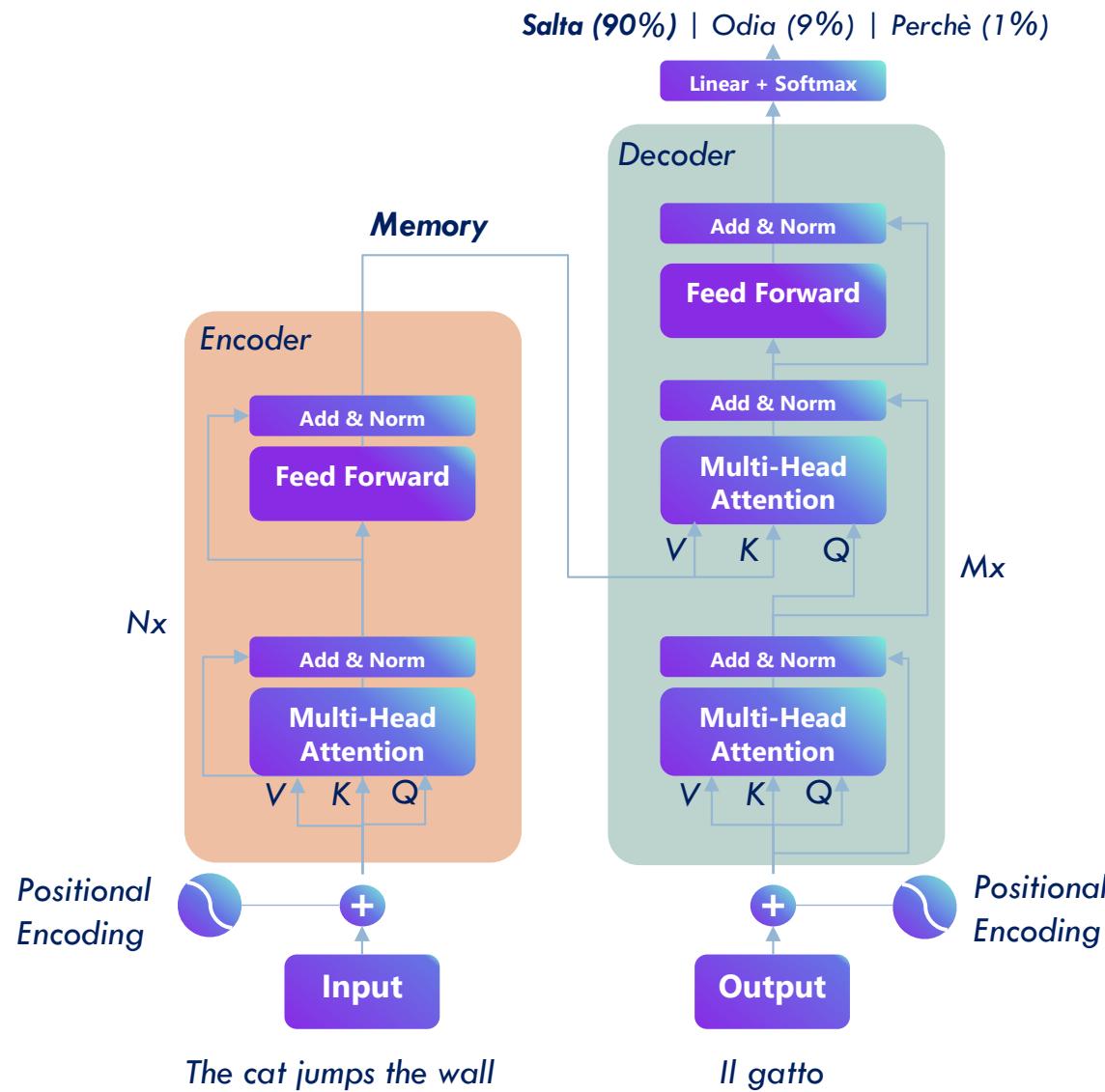
Figure 1: The Transformer - model architecture.

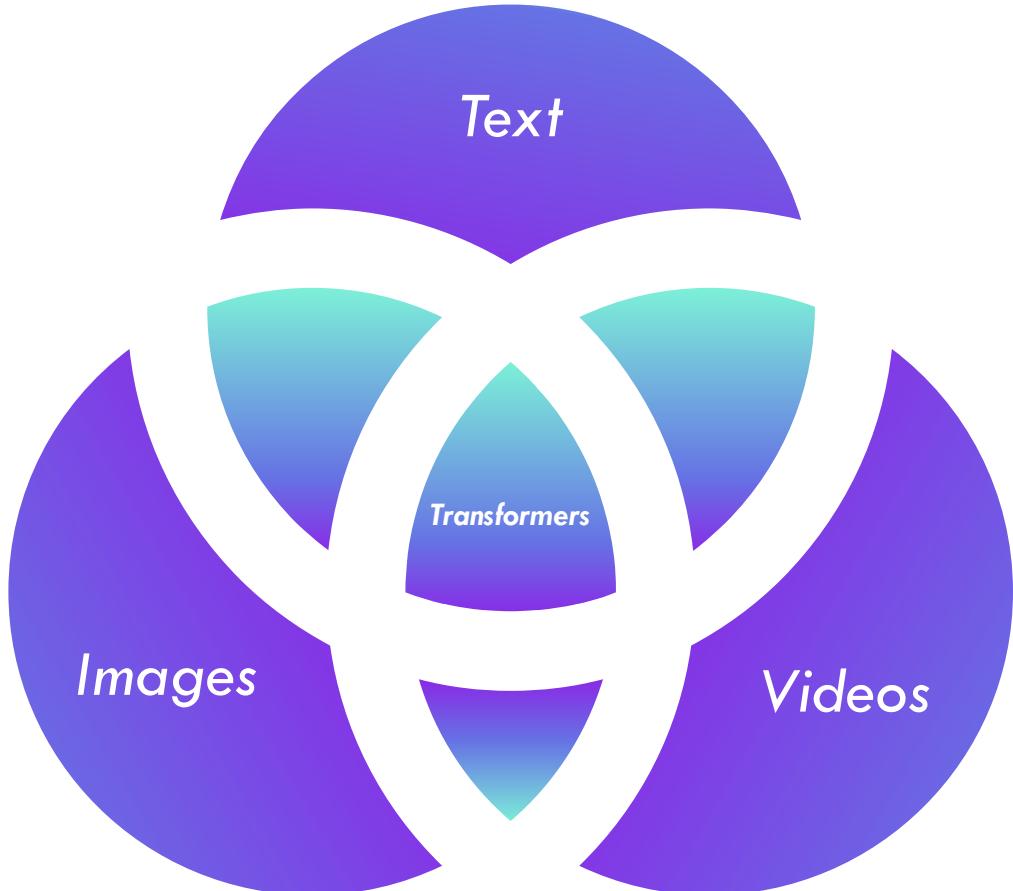
Transformers  
at school

Transformers  
at college

Transformers  
today

# The Transformer «today»





# History

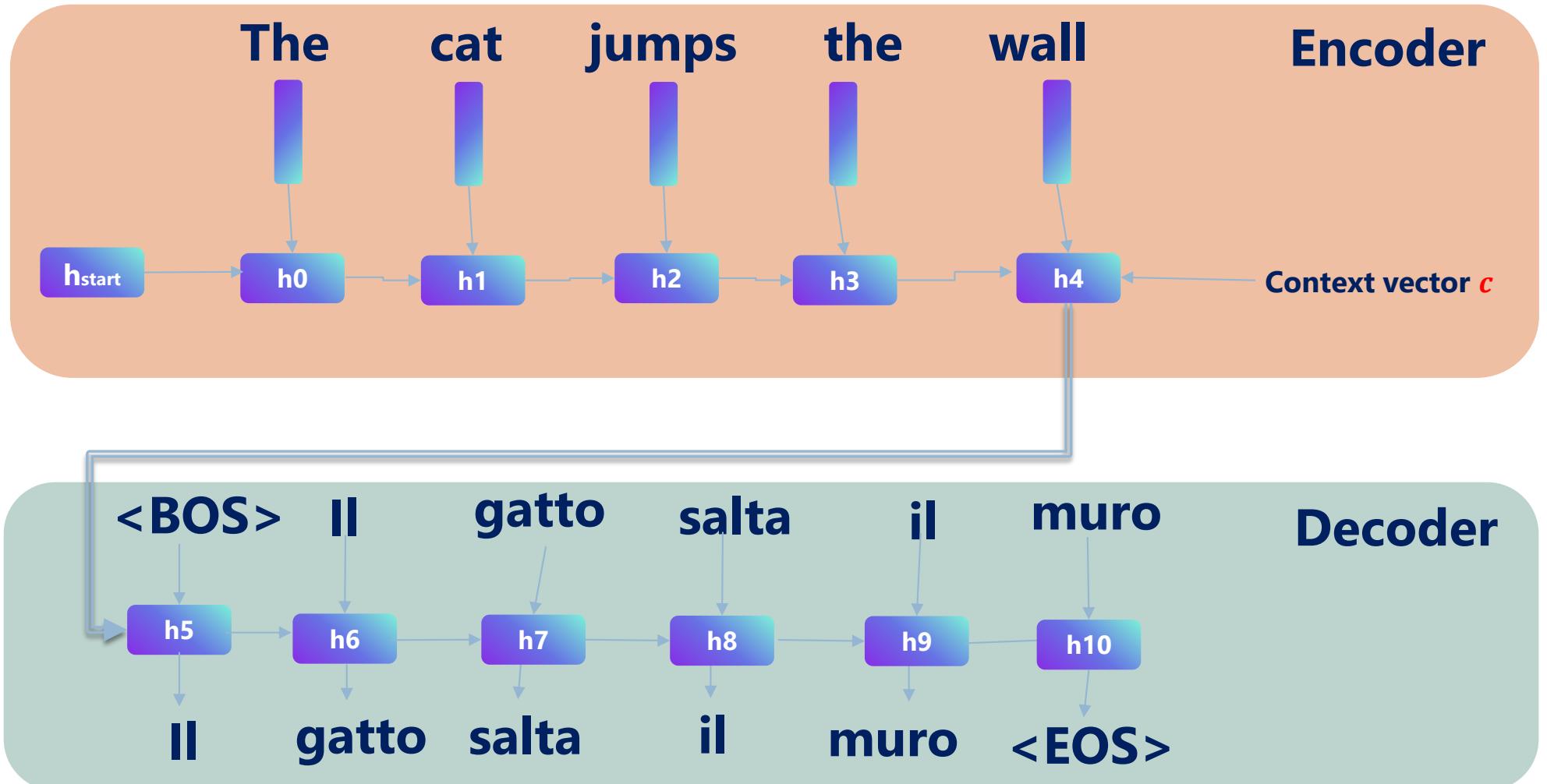
**2017** *Introduced transformers in NLP*

**2020** *Vision Transformers*

**2021** *Transformers for video understanding*

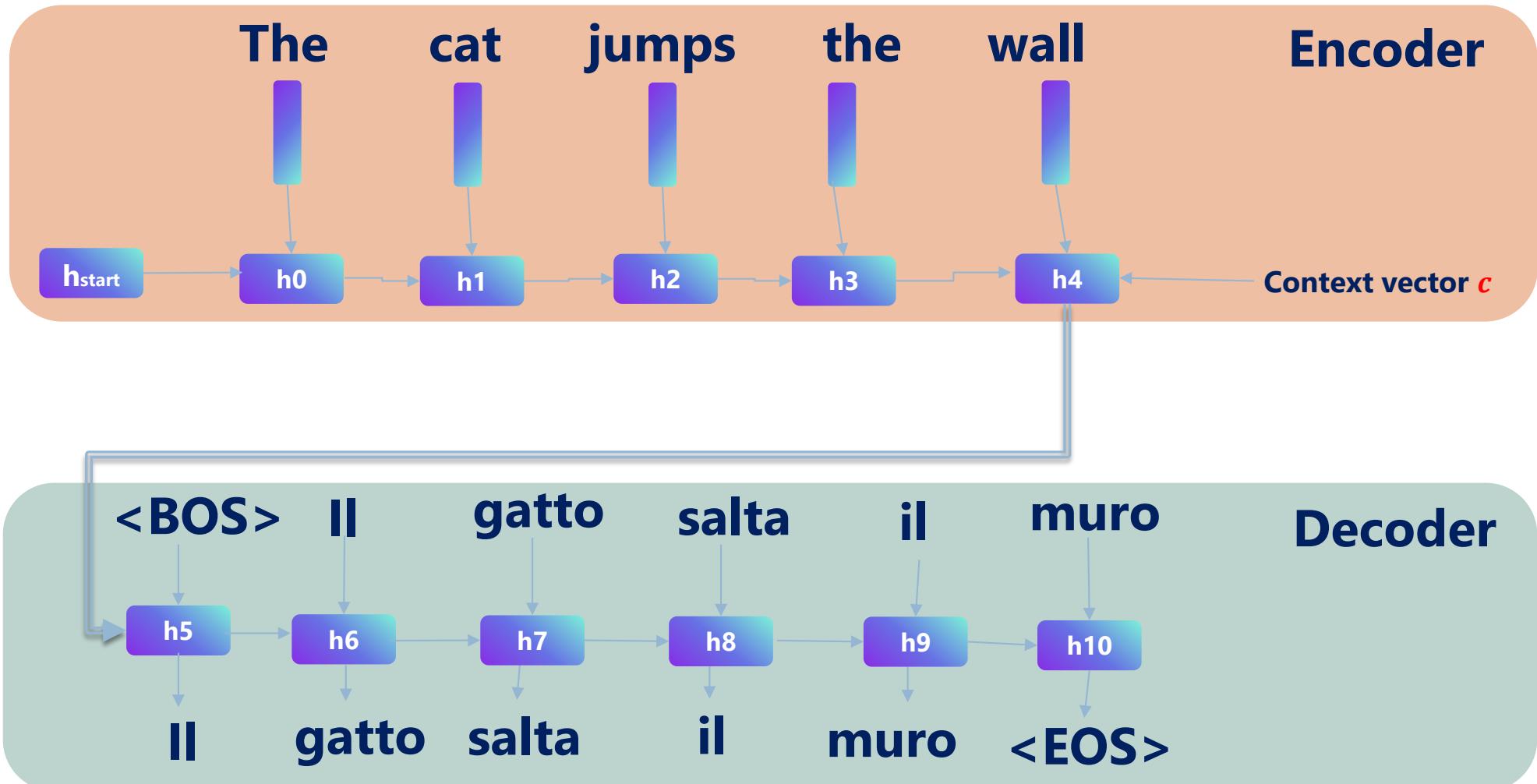
**Now** *Computer Vision Revolution!*

# A step back: Recurrent Networks (RNNs)



# Problems

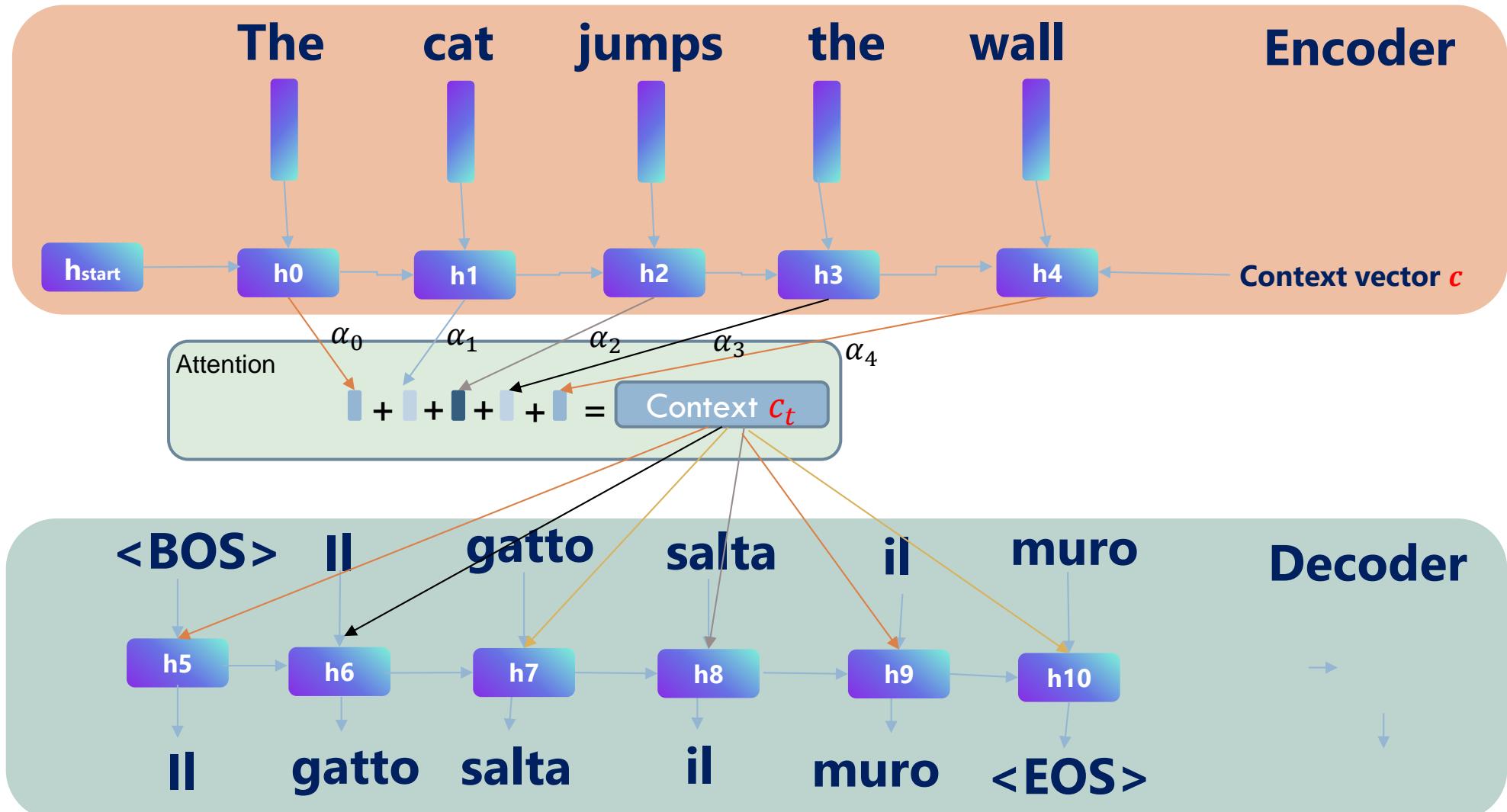
1. We **forget tokens too far** in the past in the context vector **c**
2. We need to wait the previous token to compute the next hidden-state



# Solving problem 1

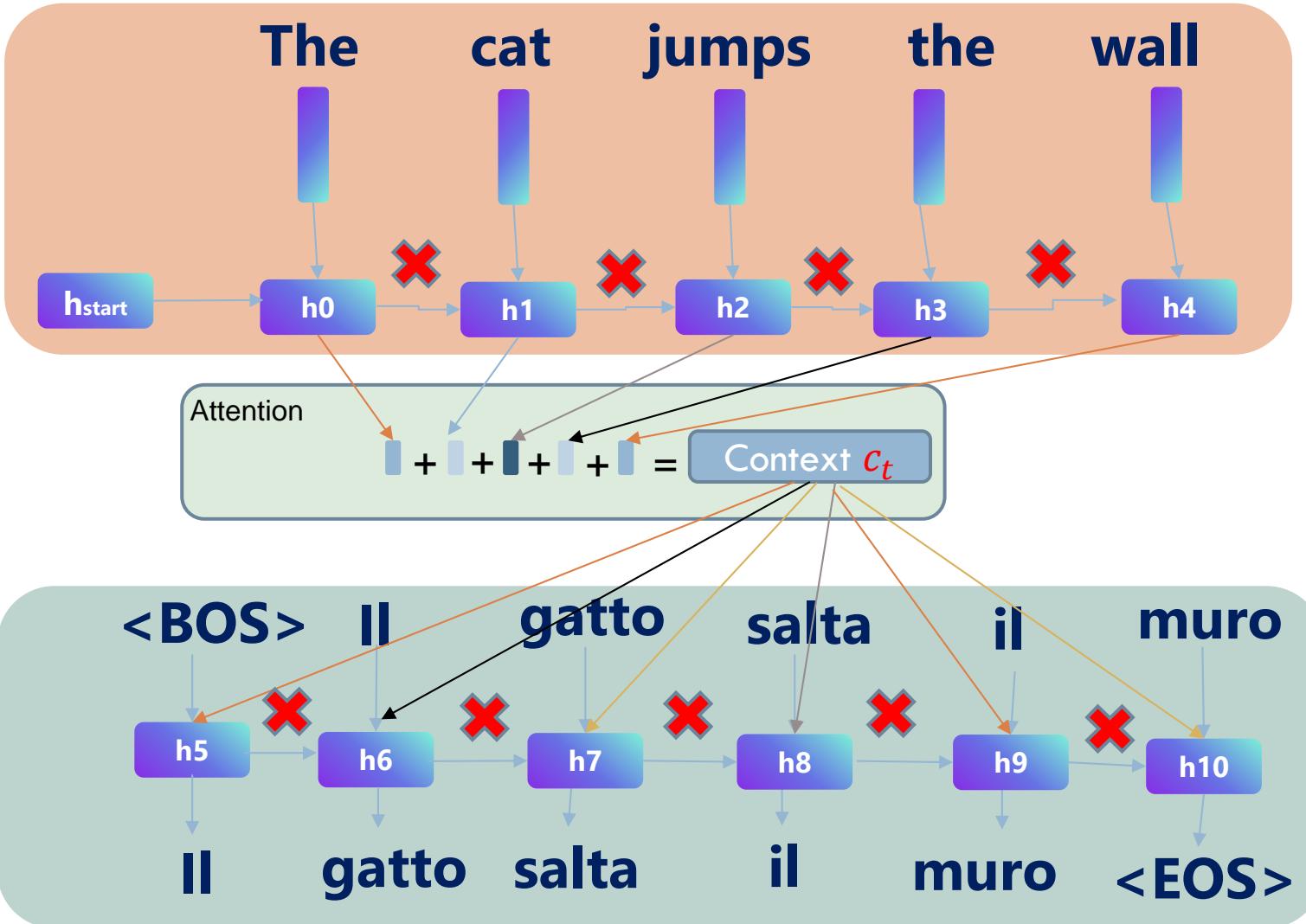
"We forget tokens too far in the past"

Solution: Add an **attention** mechanism



# Solving problem 2

"We need to wait the previous token to compute the next hidden-state"



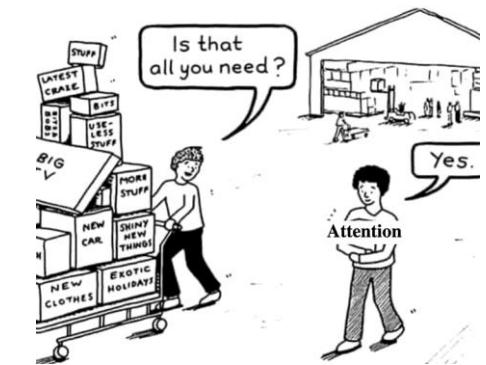
Solution

Throw away recurrent connections

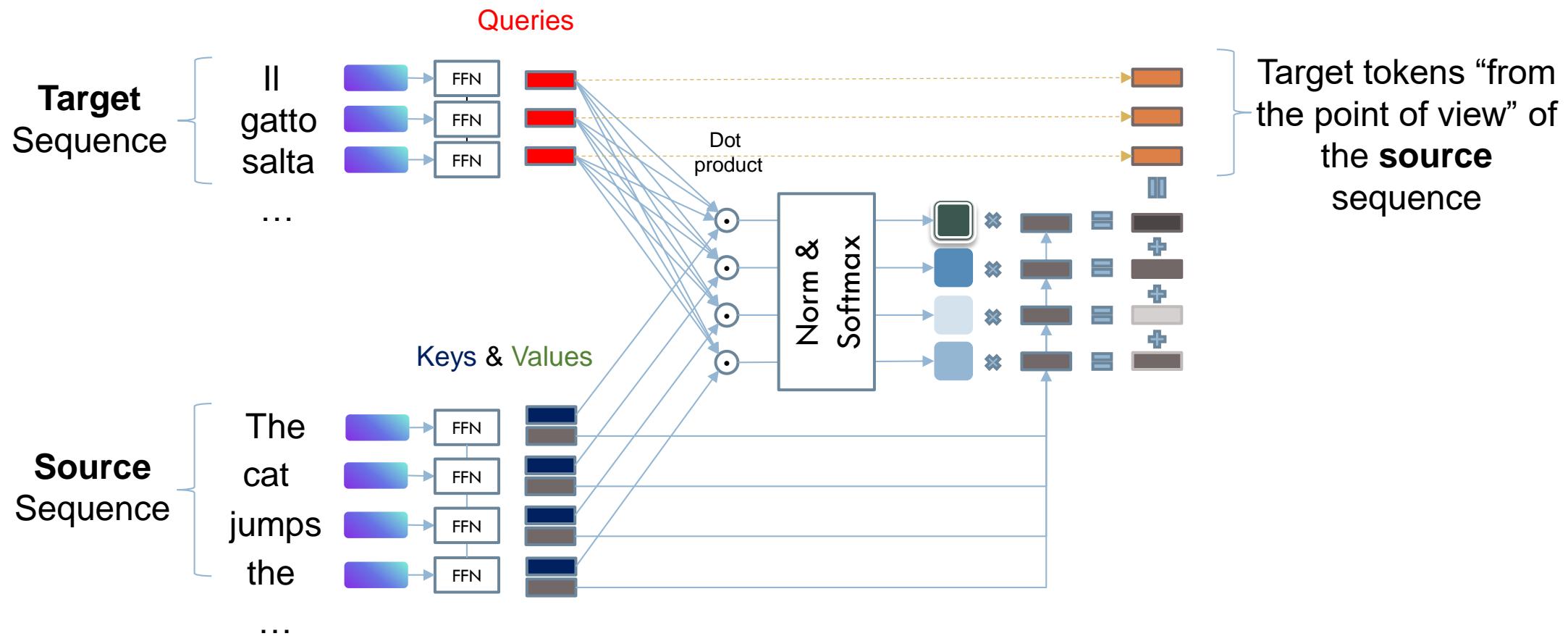


2017 paper

"Attention Is All You Need"

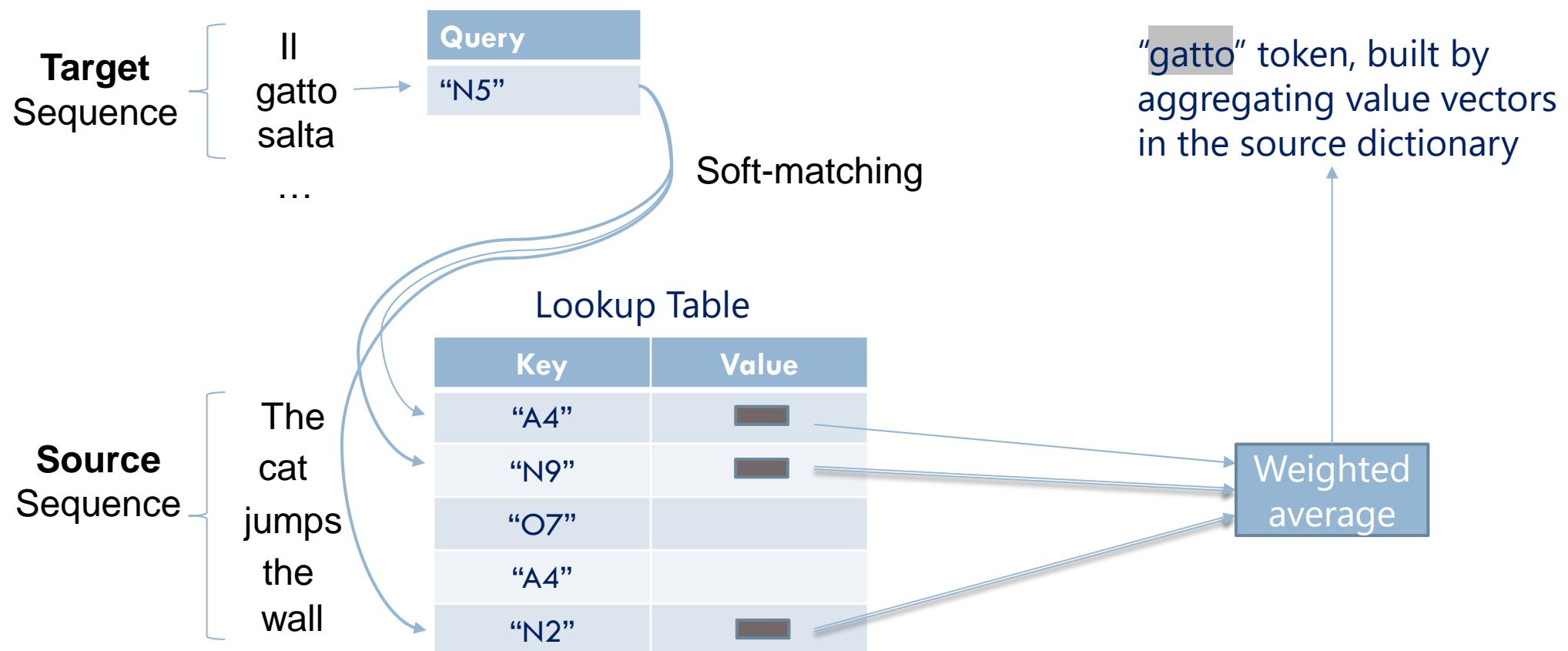


# Transformer's Attention Mechanism



# Transformer's Attention Mechanism

From a different perspective

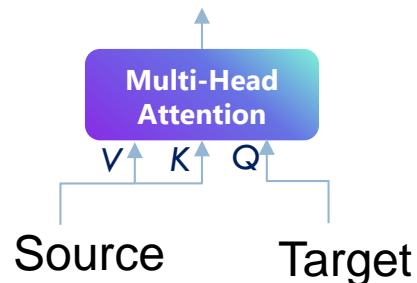


# Cross-Attention and Self-Attention

## Cross-Attention



- Source  $\neq$  Target
  - Queries from Target
  - Key, Values from Source

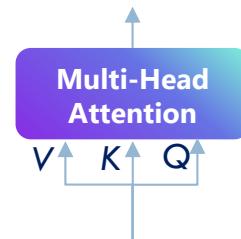


- Captures inter-sequence dependencies

## Self-Attention



- Source = Target
  - Key, Queries, Values obtained from the same sentence
- Captures intra-sequence dependencies

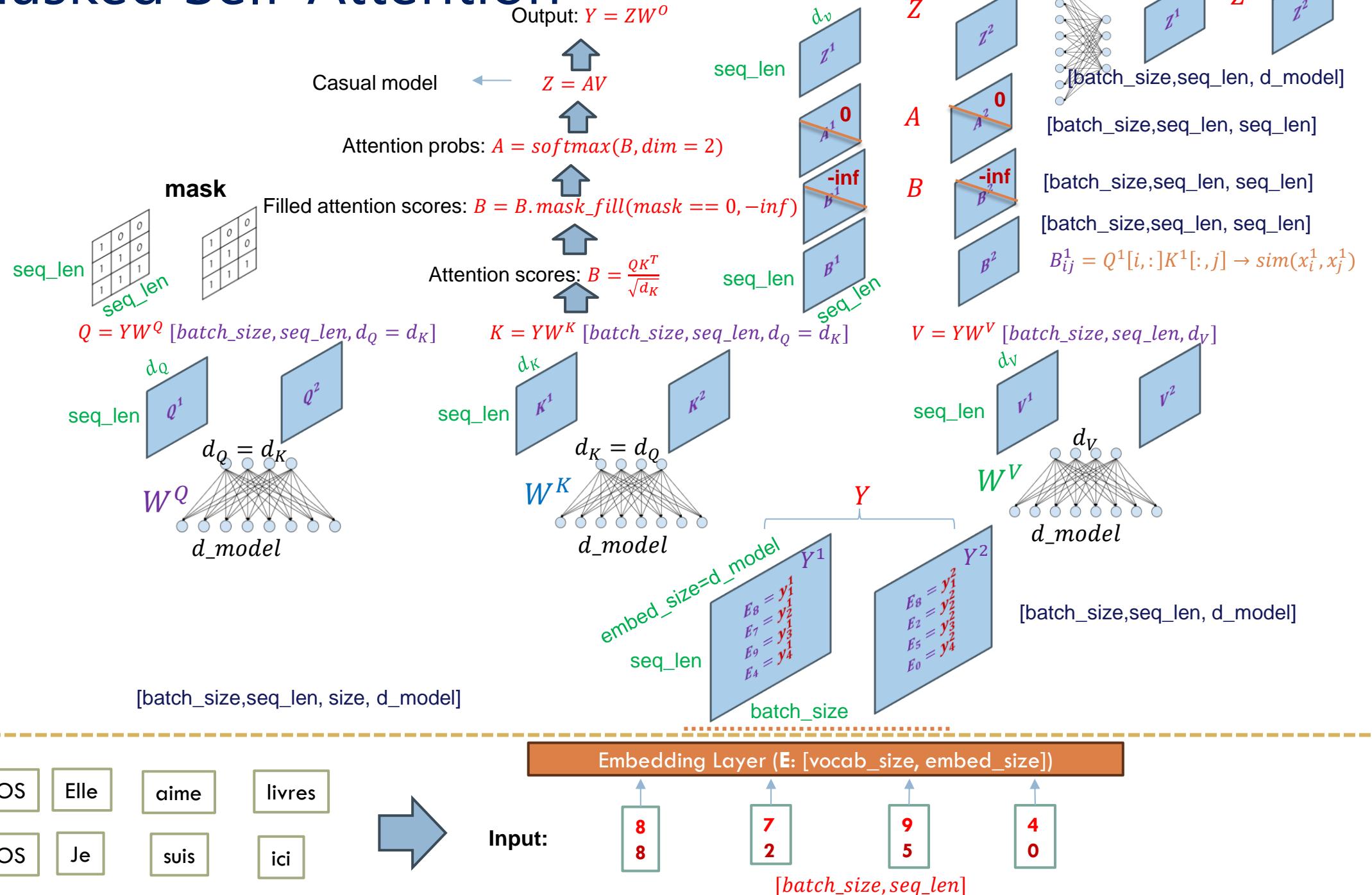


# Self-Attention

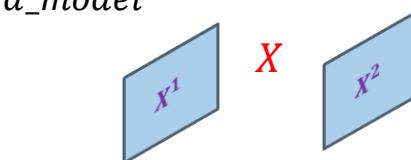
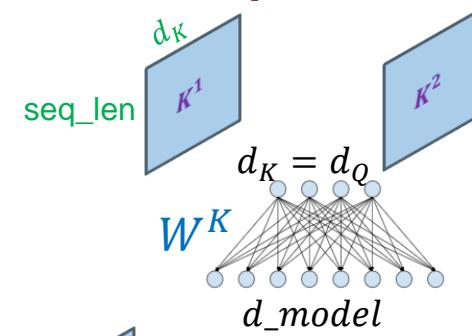
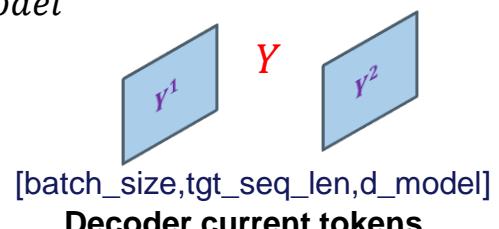
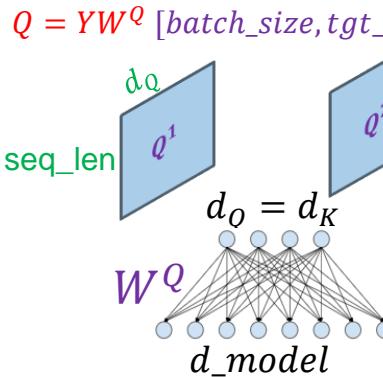
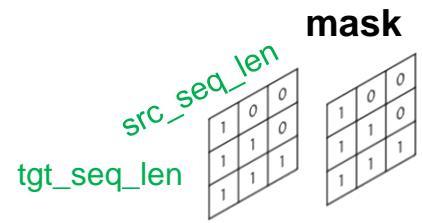
*Attention calculation is  $O(n^2)$*



# Masked Self-Attention

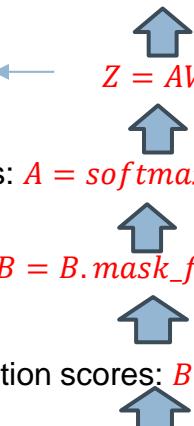


# Cross-Attention



[batch\_size, tgt\_seq\_len,  $d_V$ ]

Output:  $Y = ZW^O$



Attention probs:  $A = \text{softmax}(B, \text{dim} = 2)$

Filled attention scores:  $B = B.\text{mask\_fill}(\text{mask} == 0, -\text{inf})$

Attention scores:  $B = \frac{QK^T}{\sqrt{d_K}}$

tgt\_seq\_len  
 $d_V$   
 $B^1$   
 $B^2$

$Z$

$Z^1$

$Z^2$

$A$

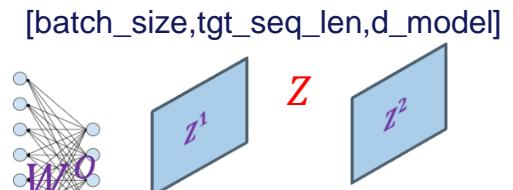
$A^1$

$A^2$

$B$

$B^1$

$B^2$

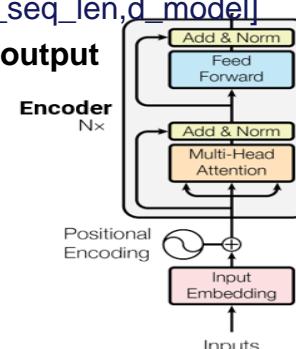
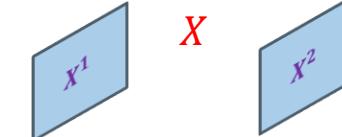
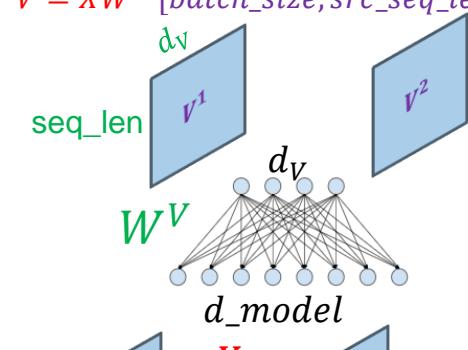


[batch\_size, tgt\_seq\_len, src\_seq\_len]

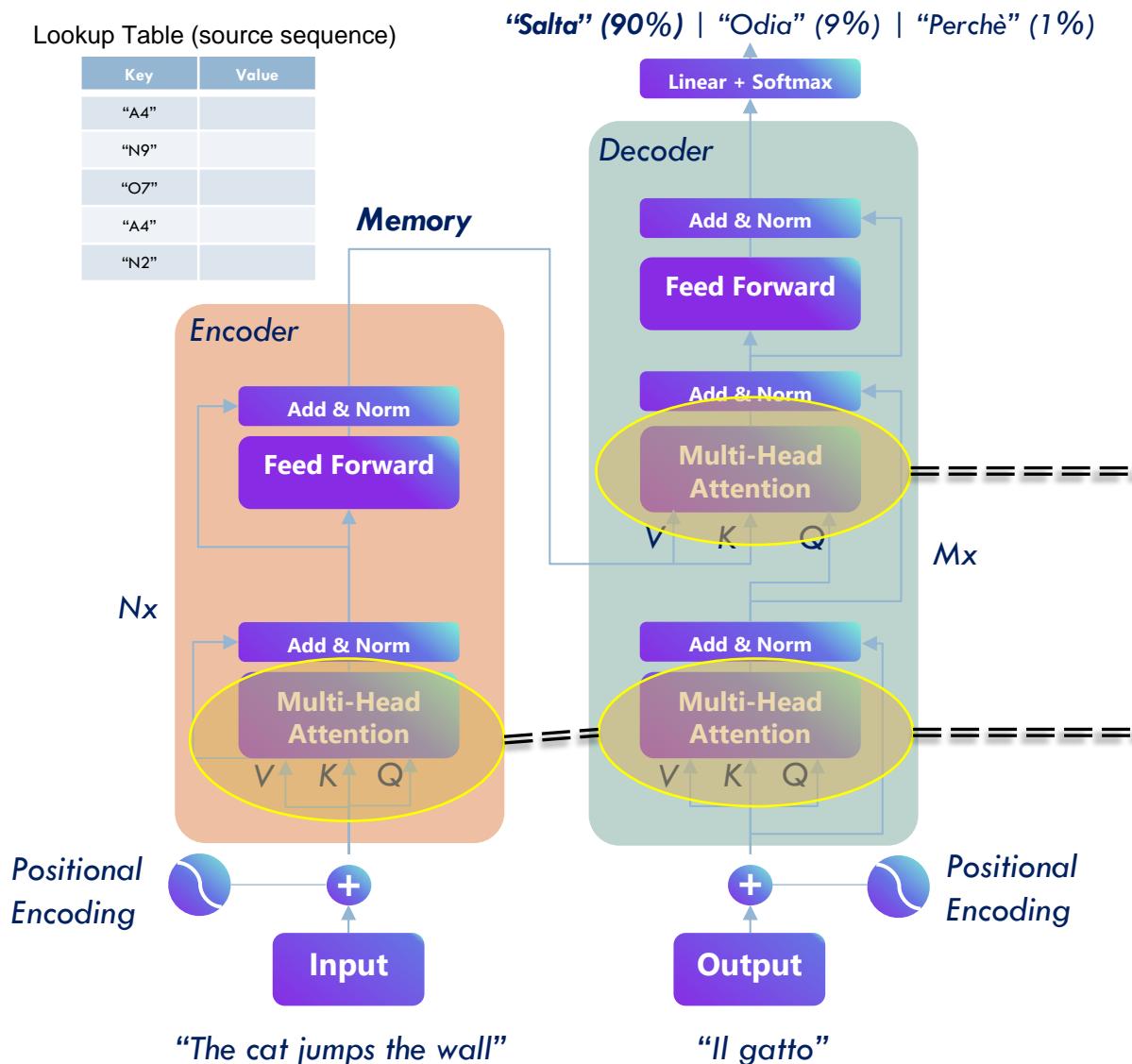
[batch\_size, tgt\_seq\_len, src\_seq\_len]

[batch\_size, tgt\_seq\_len, src\_seq\_len]

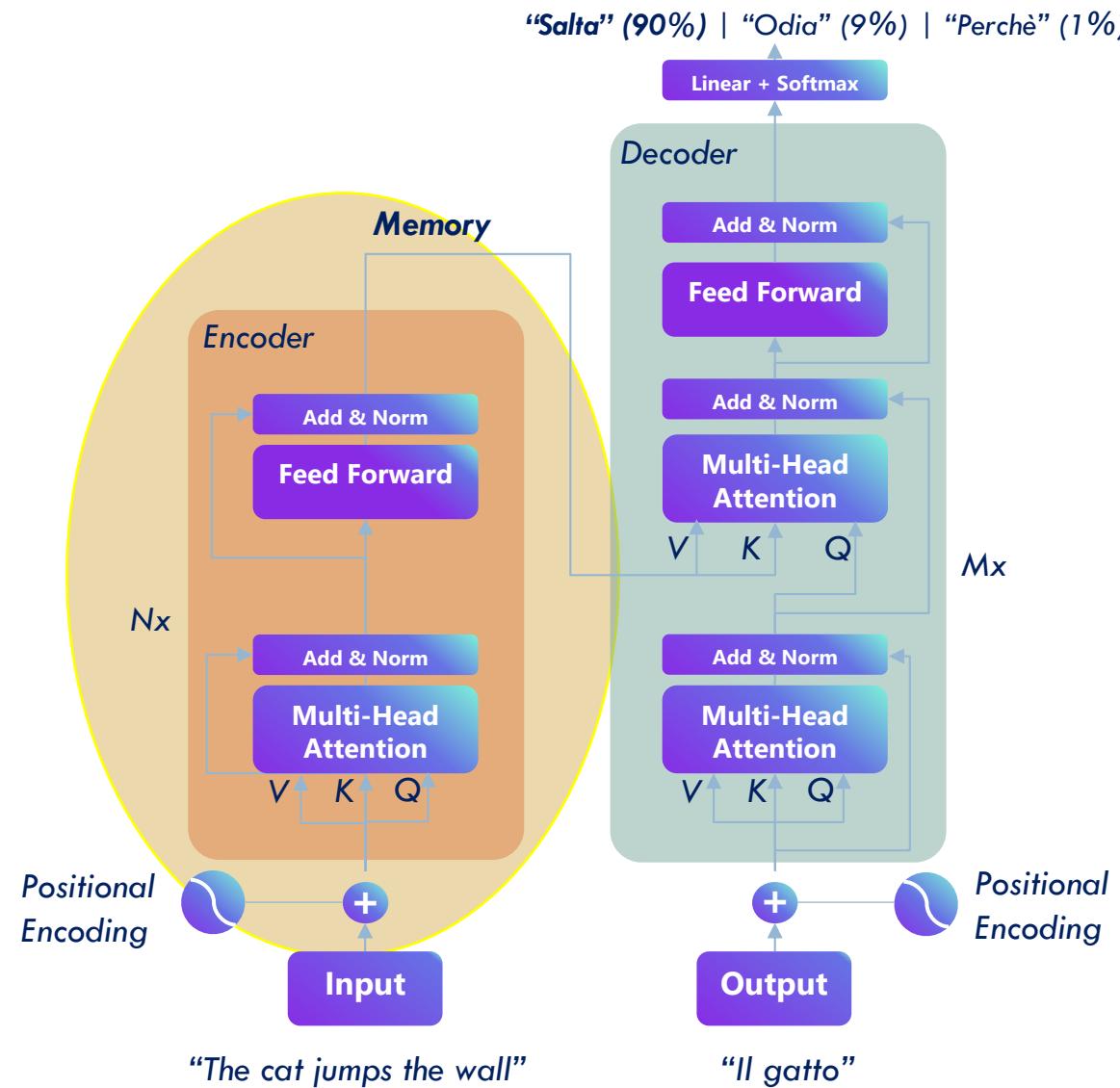
$B_{ij}^1 = Q^1[i, :]K^1[:, j] \rightarrow \text{sim}(x_i^1, x_j^1)$



# Full Transformer Architecture



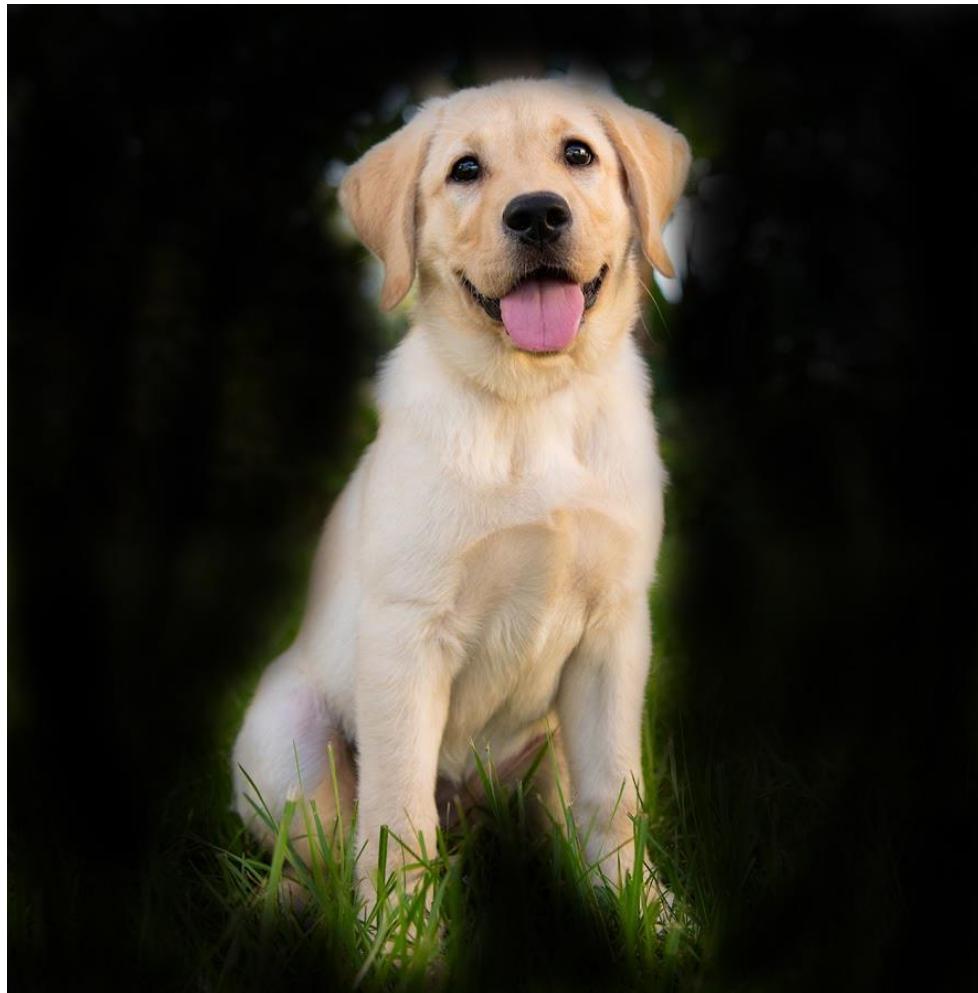
# Full Transformer Architecture



# Vision Transformer

# Transformers in Computer Vision

*Can we use the self-attention mechanism in images?*



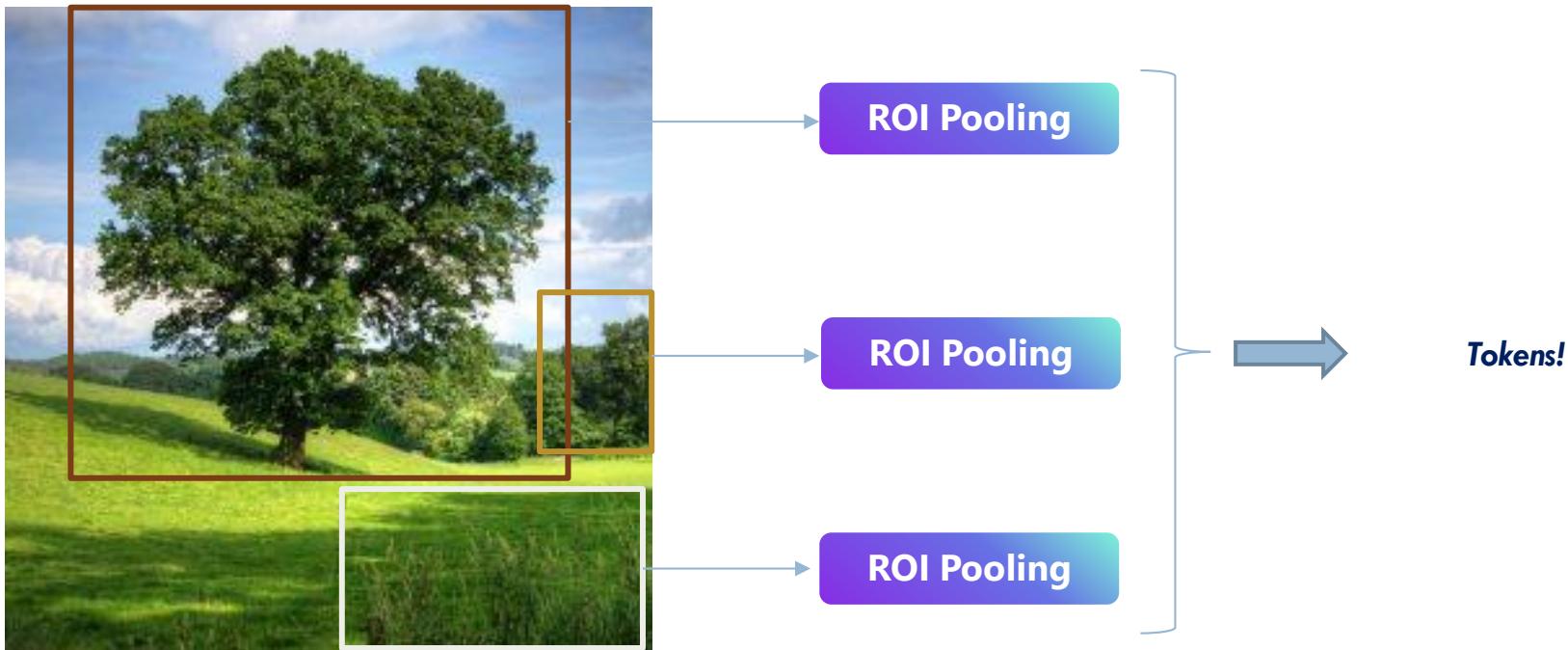
# Transformers in Computer Vision

- The transformer works with a set of tokens
- What are **tokens/visual words** in images?

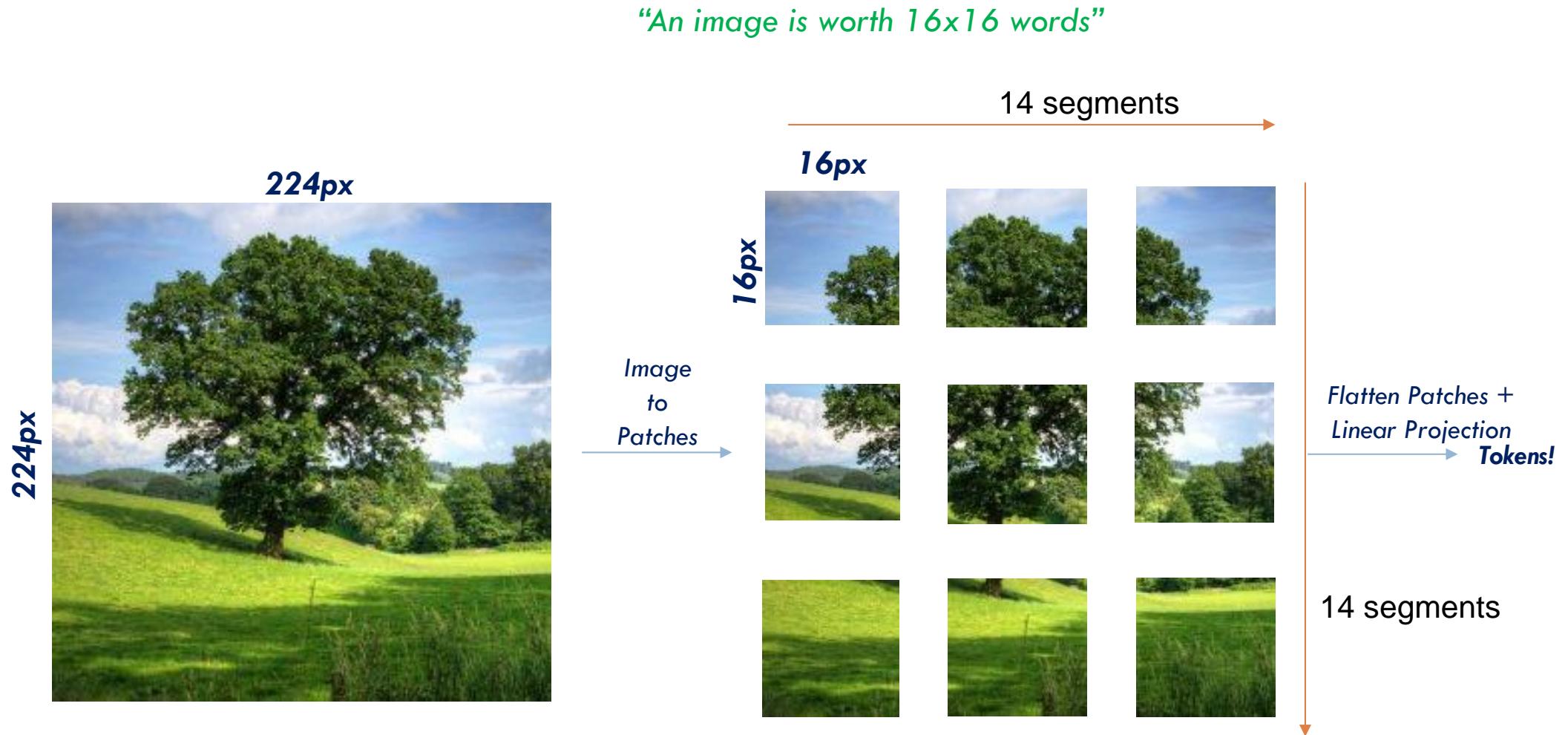


# Transformers in Computer Vision

- Tokens as the features from an object detector

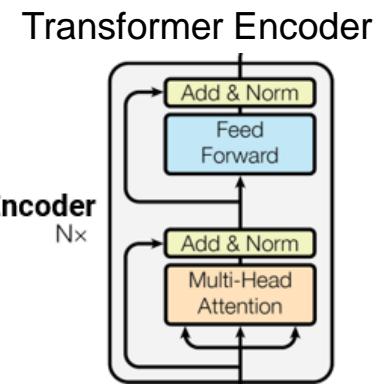
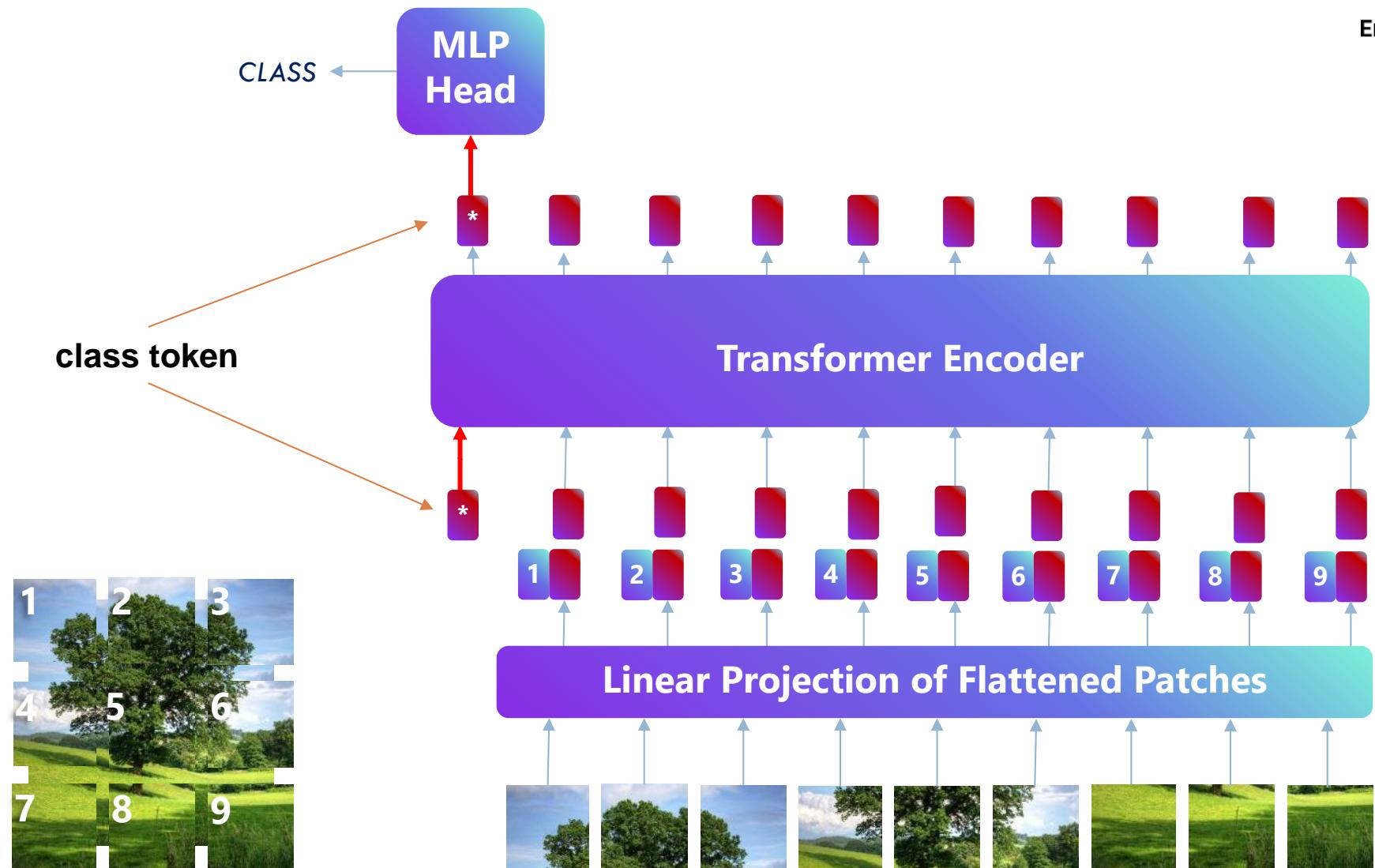


# Vision Transformers (ViTs)



“An image is worth 16x16 words” | Dosovitskiy et al., 2020

# Vision Transformers (ViTs)



# Vision Transformers (ViTs)

In practice: take 224x224 input image,  
divide into 14x14 grid of 16x16 pixel  
patches (or 16x16 grid of 14x14 patches)

Output vectors



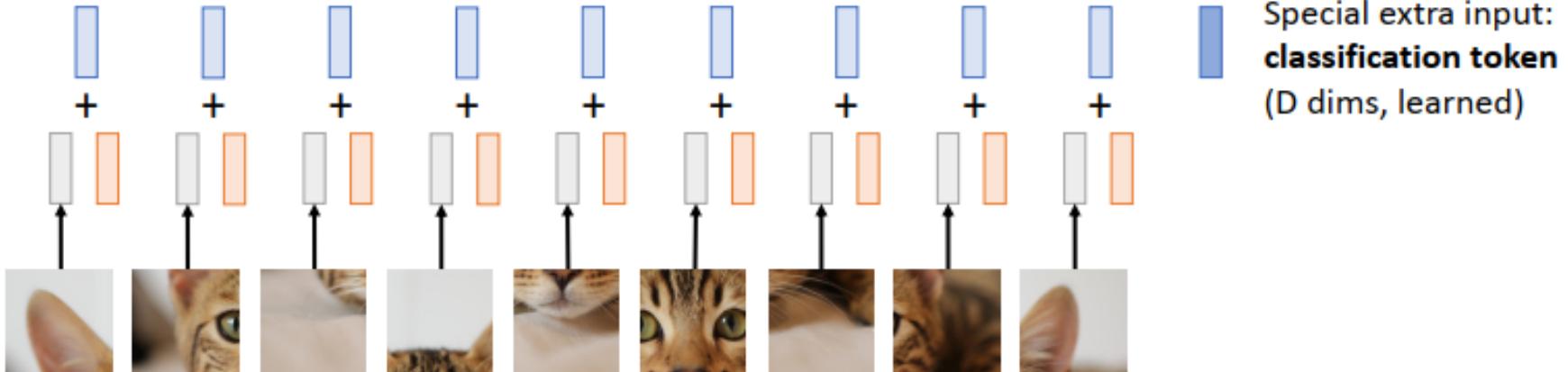
With 48 layers, 16 heads per  
layer, all attention matrices  
take 112 MB (or 192MB)

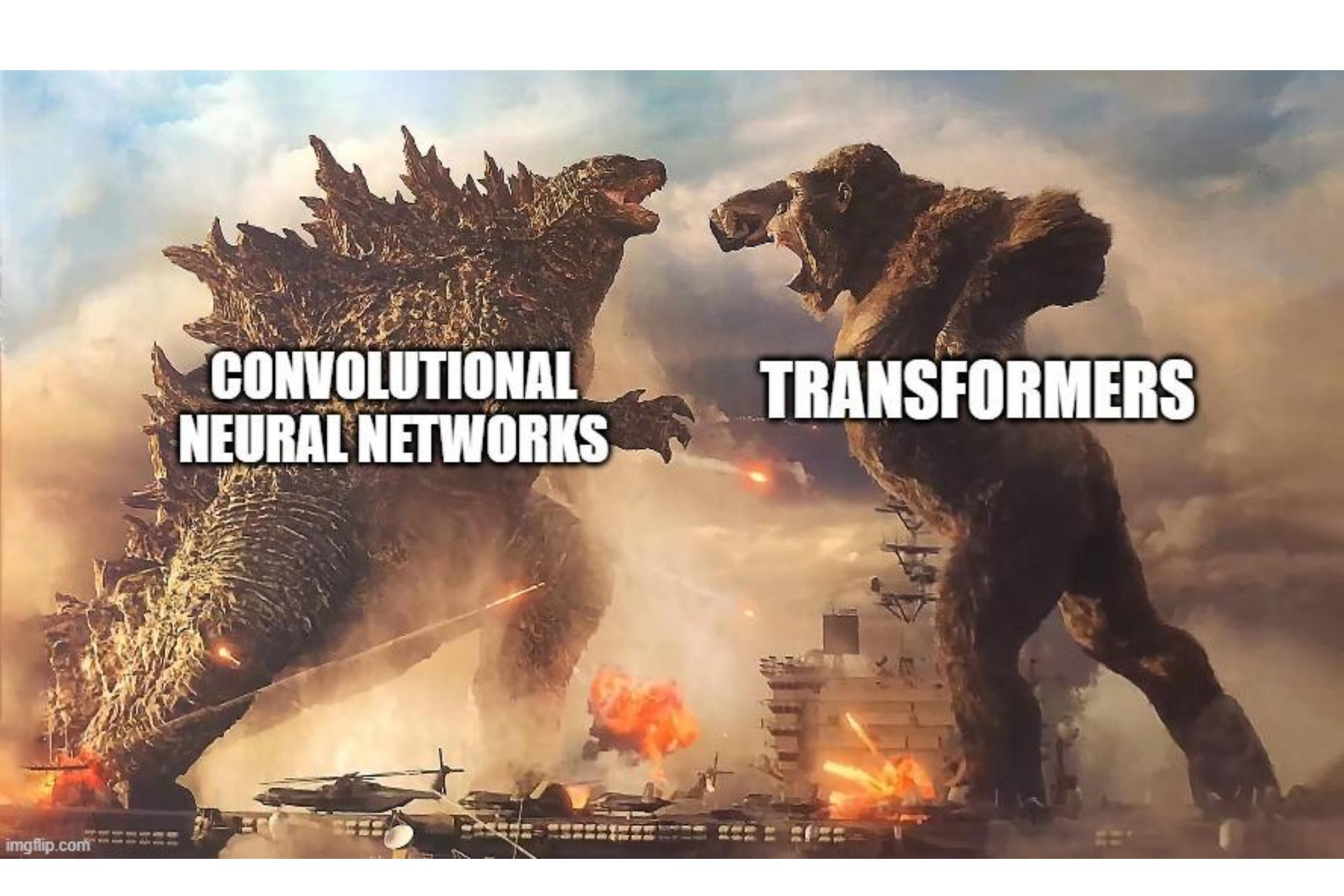
Exact same as  
NLP Transformer!

Add positional  
embedding: learned D-  
dim vector per position

Linear projection to  
D-dimensional vector

N input patches, each  
of shape 3x16x16

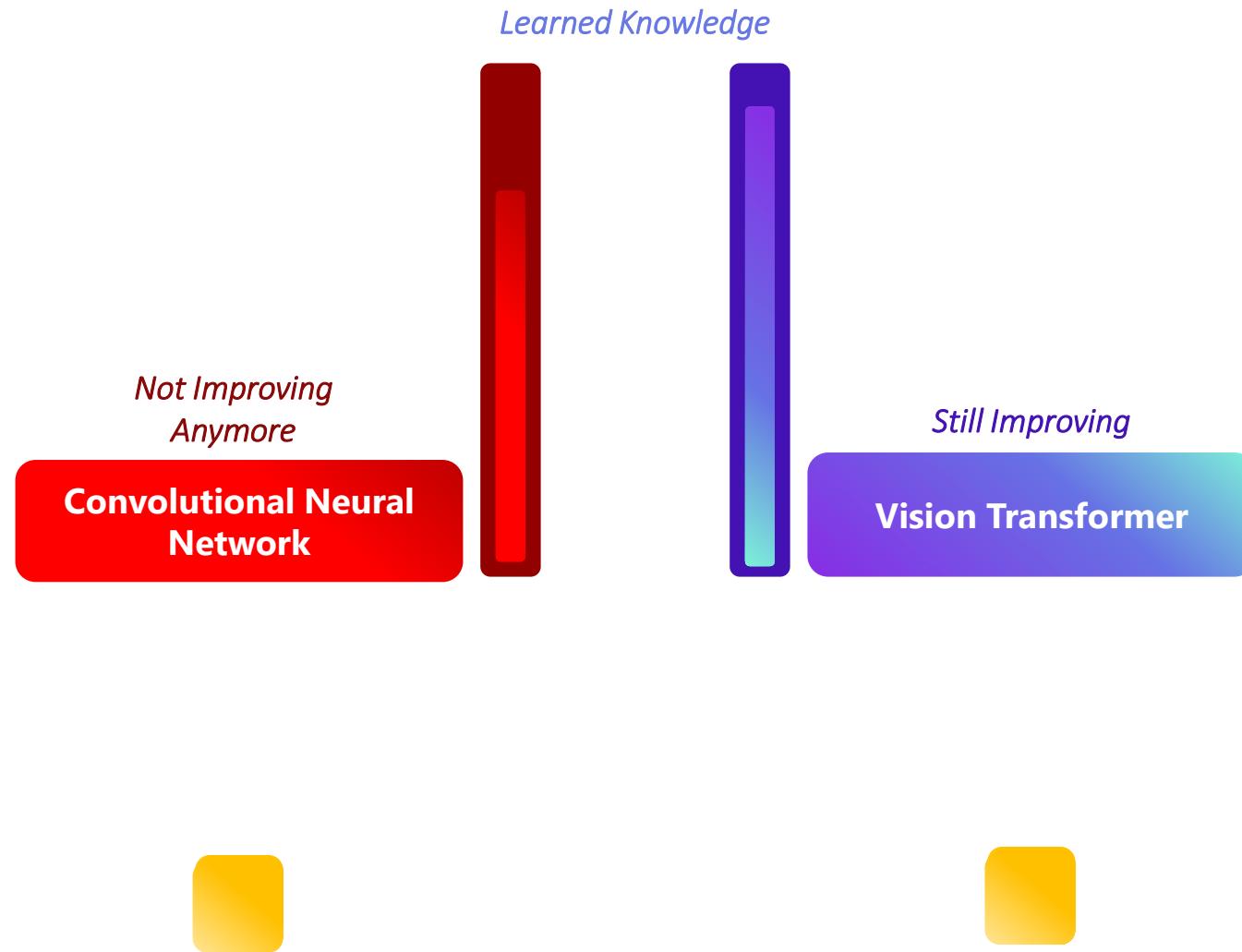




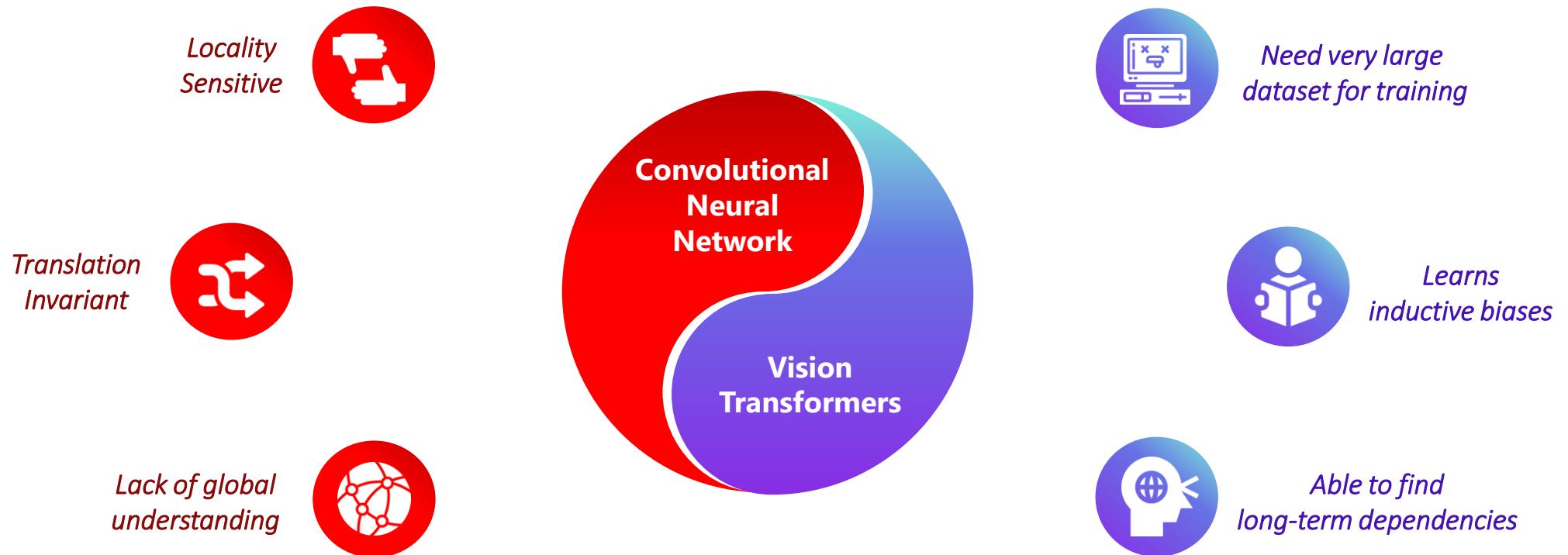
**CONVOLUTIONAL  
NEURAL NETWORKS**

**TRANSFORMERS**

# What happens during training?

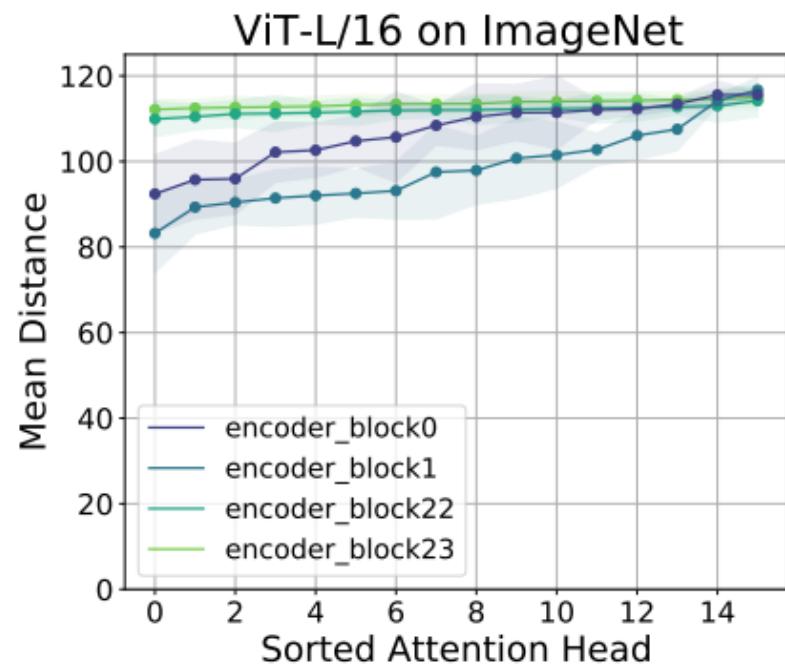


# Why are they different?

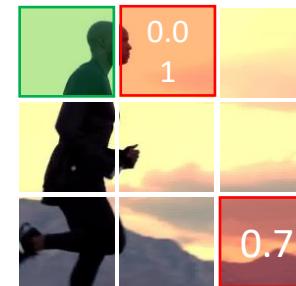


# A different point of view

*ViTs are both local and global!*



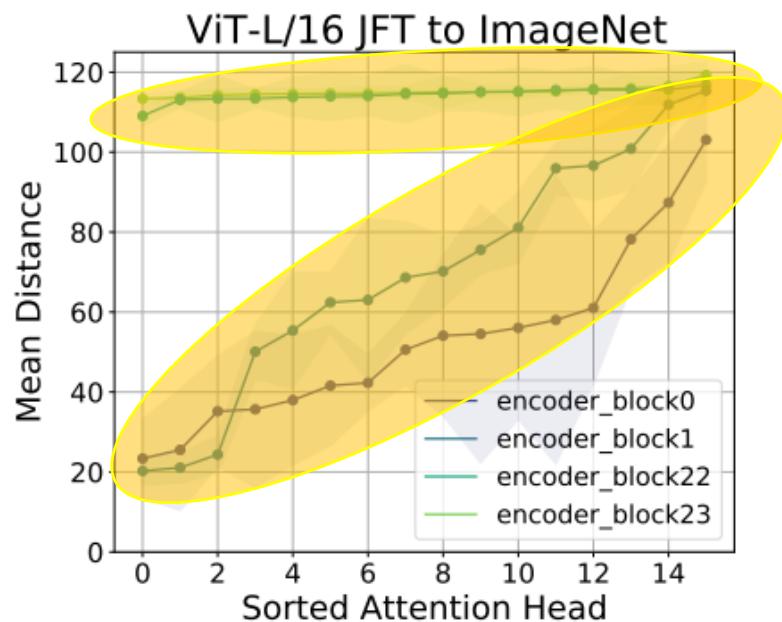
*Heads focus on farther patches*



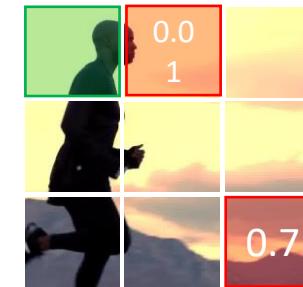
*The ViT learns only global information  
with low amount of data*

# A different point of view

*ViTs are both local and global!*



*Higher layers heads still focus on farther patches*



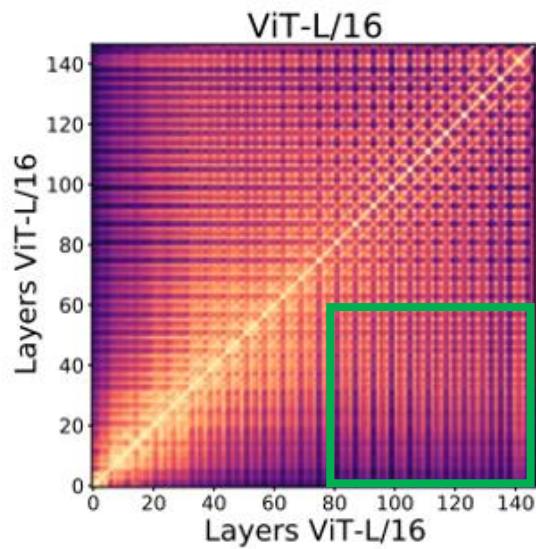
*Lower layers heads focus on both farther and closer patches*



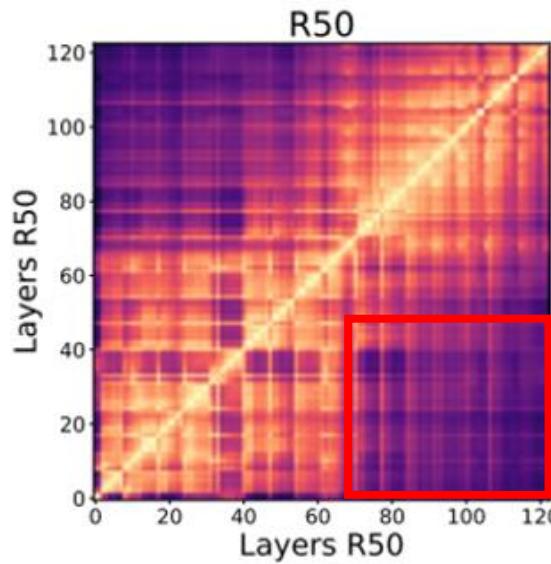
*The ViT learns also local information with more data*

# A different point of view

*They learn different representations!*



*Similar representations  
through the layers*

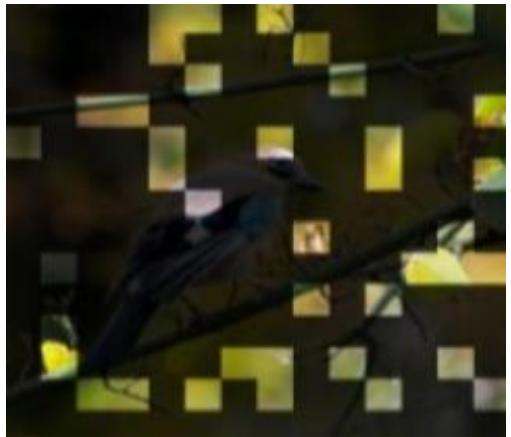


*Different representations  
through the layers*

# A different point of view

*Vision Transformers are very robust!*

*Occlusion*



*Distribution Shift*



*Adversarial Perturbation*



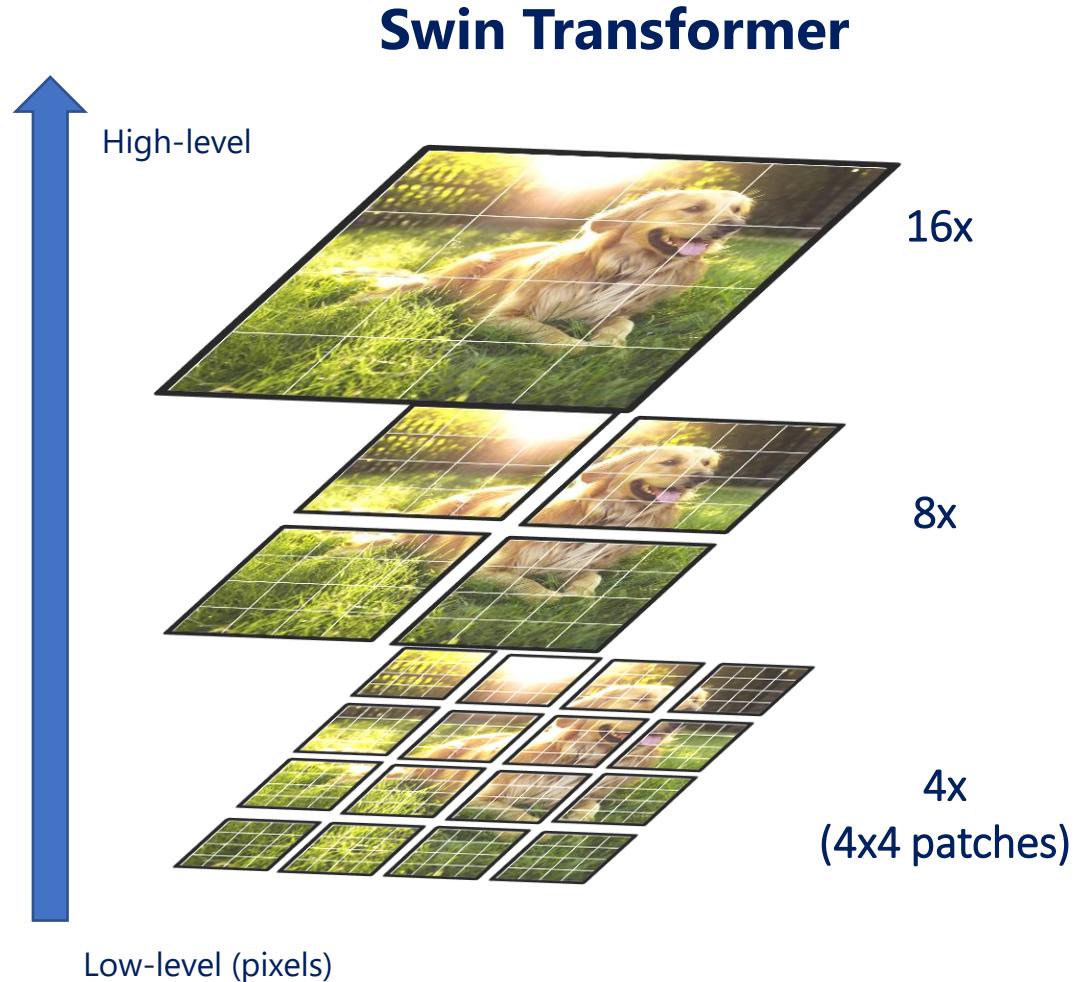
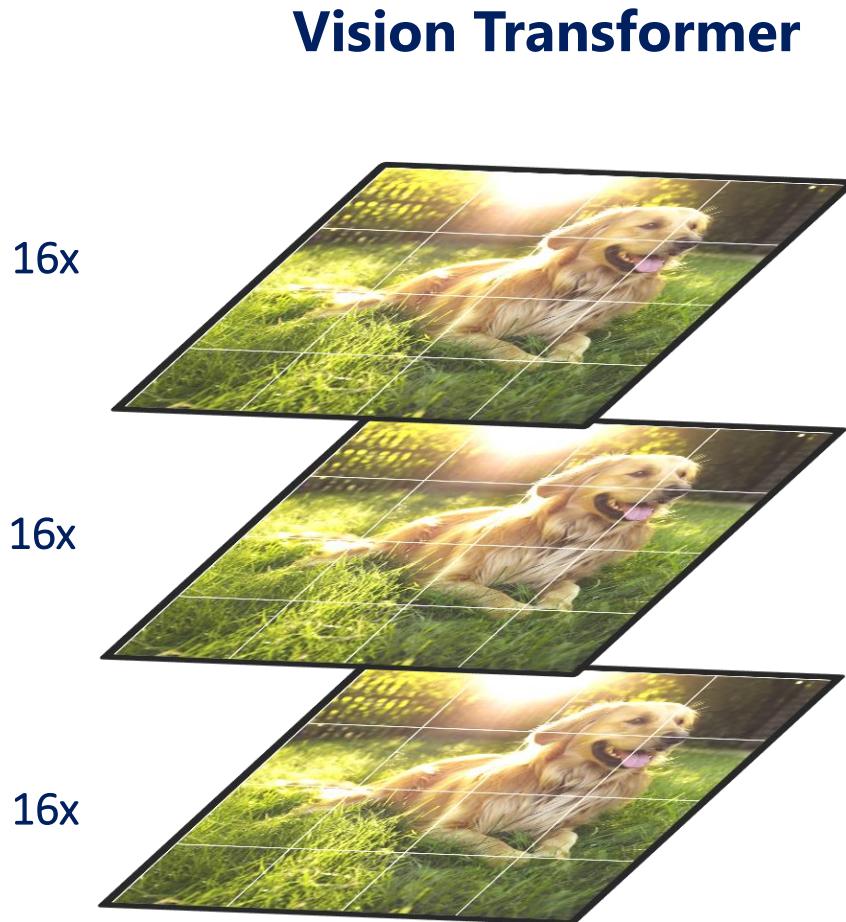
*Permutation*



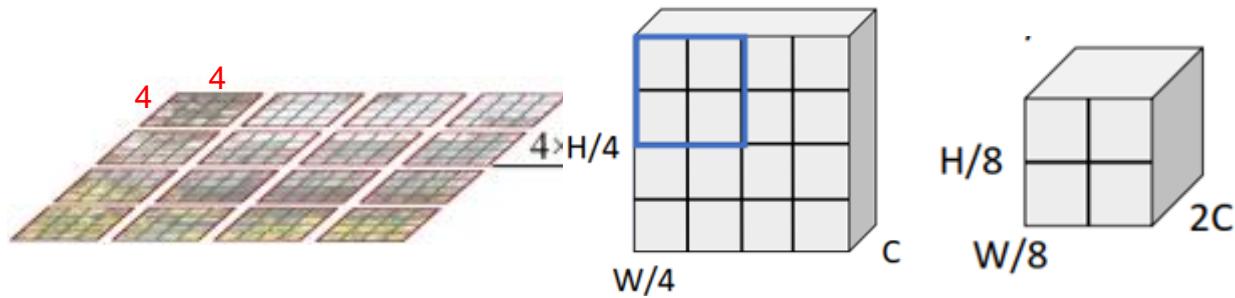
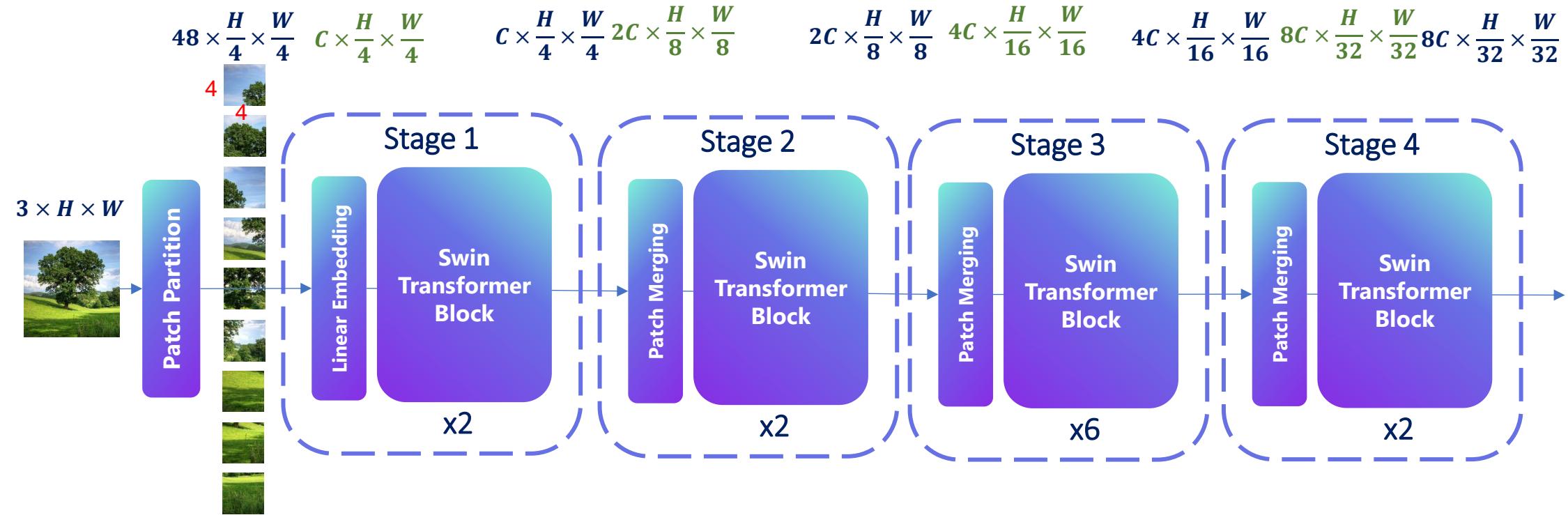
# Swin Transformer

# Swin Transformers

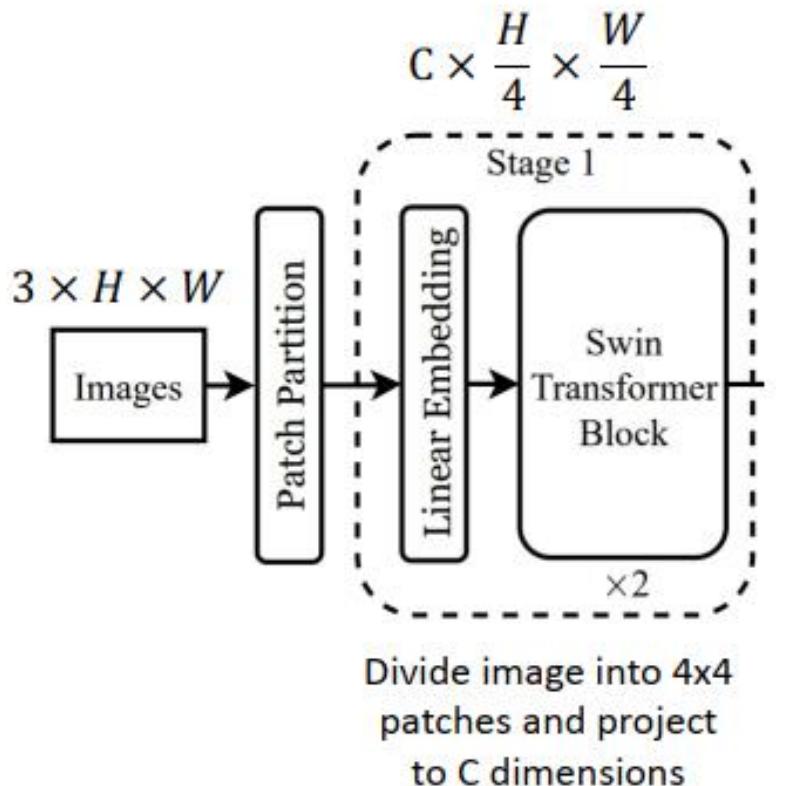
*Shifted Window based Self-Attention*



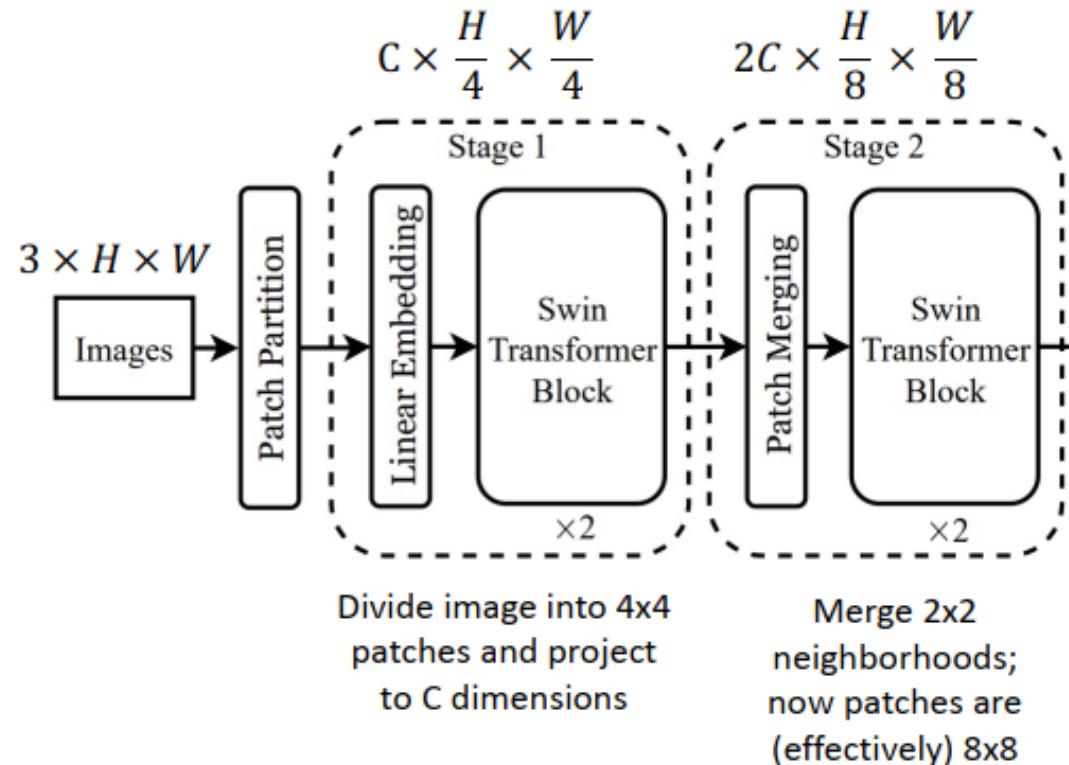
# Swin Transformers



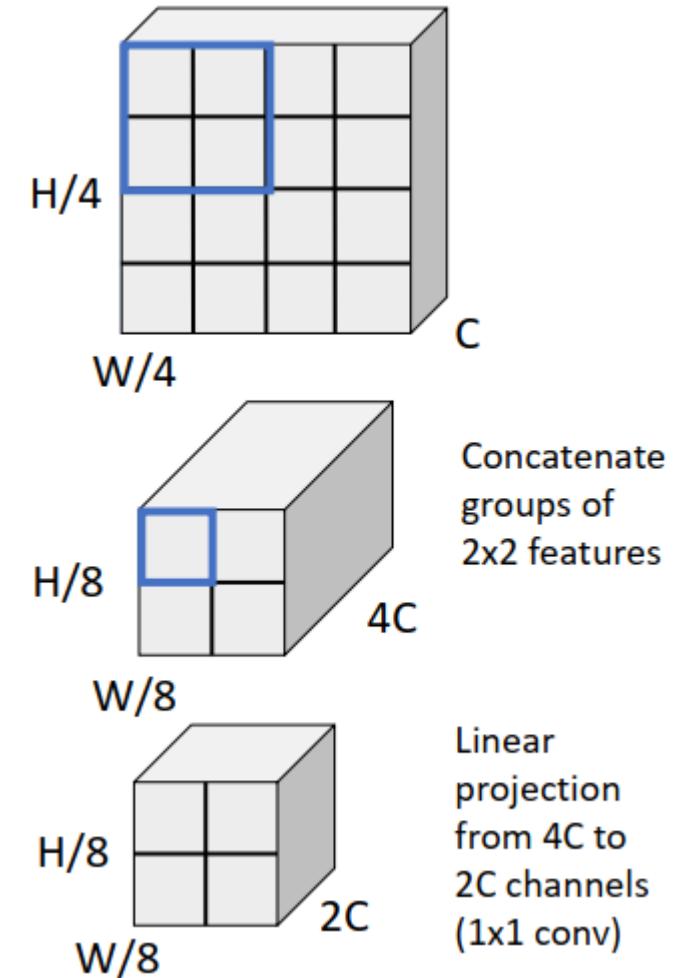
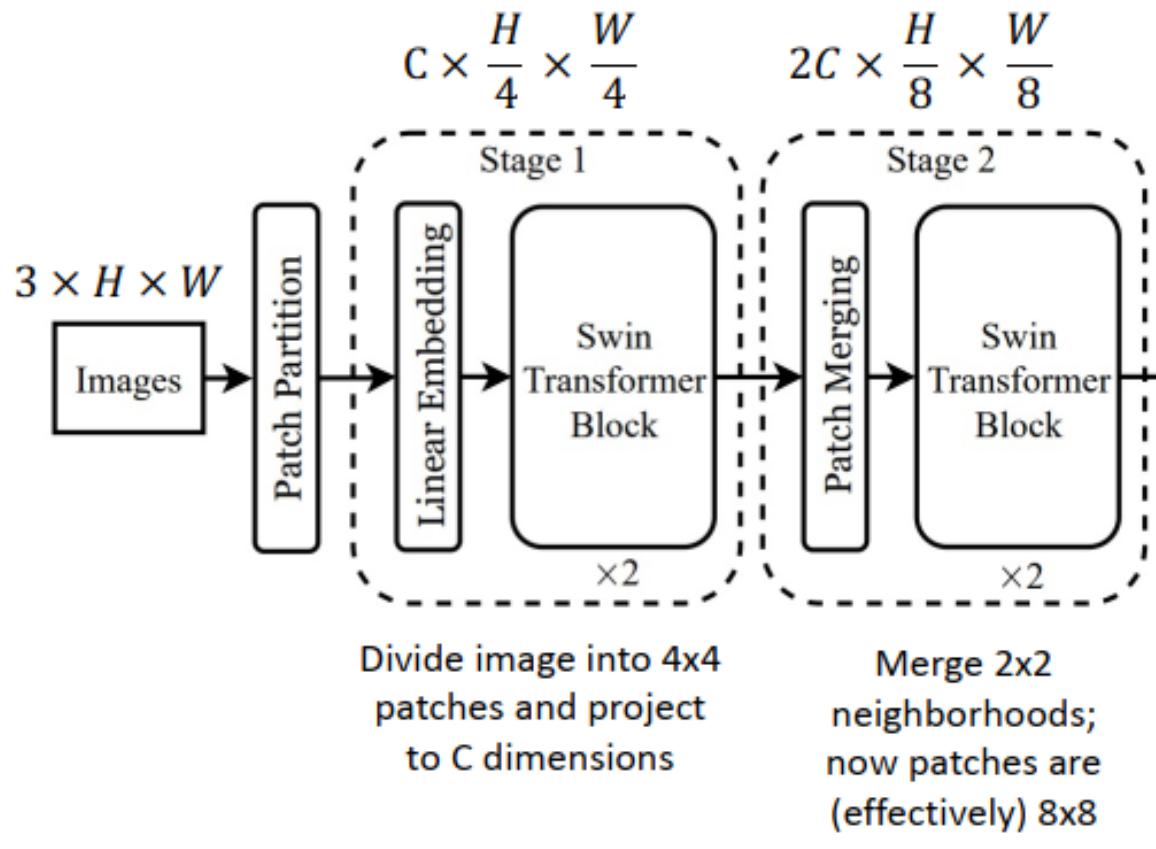
# Swin Transformers



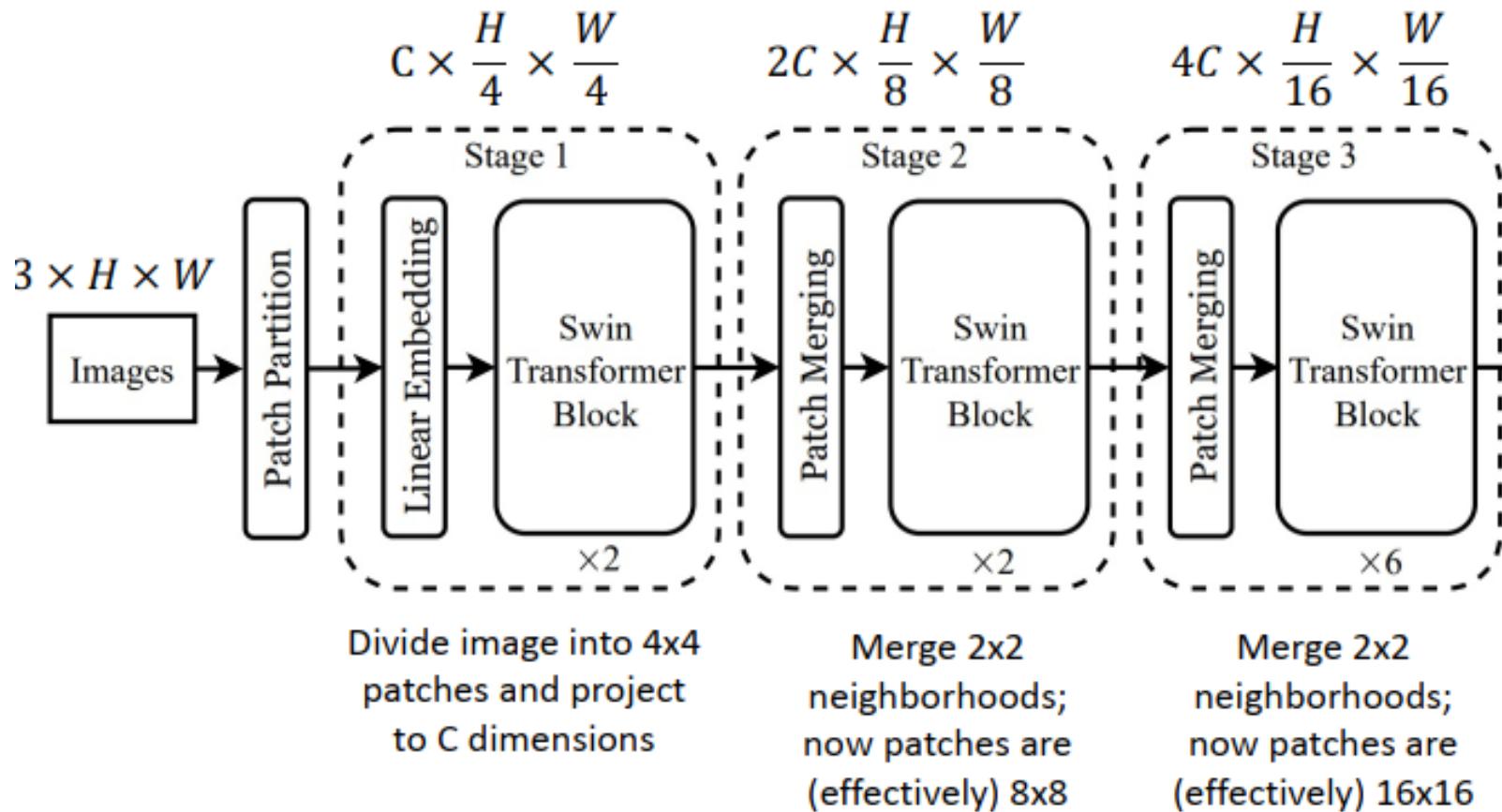
# Swin Transformers



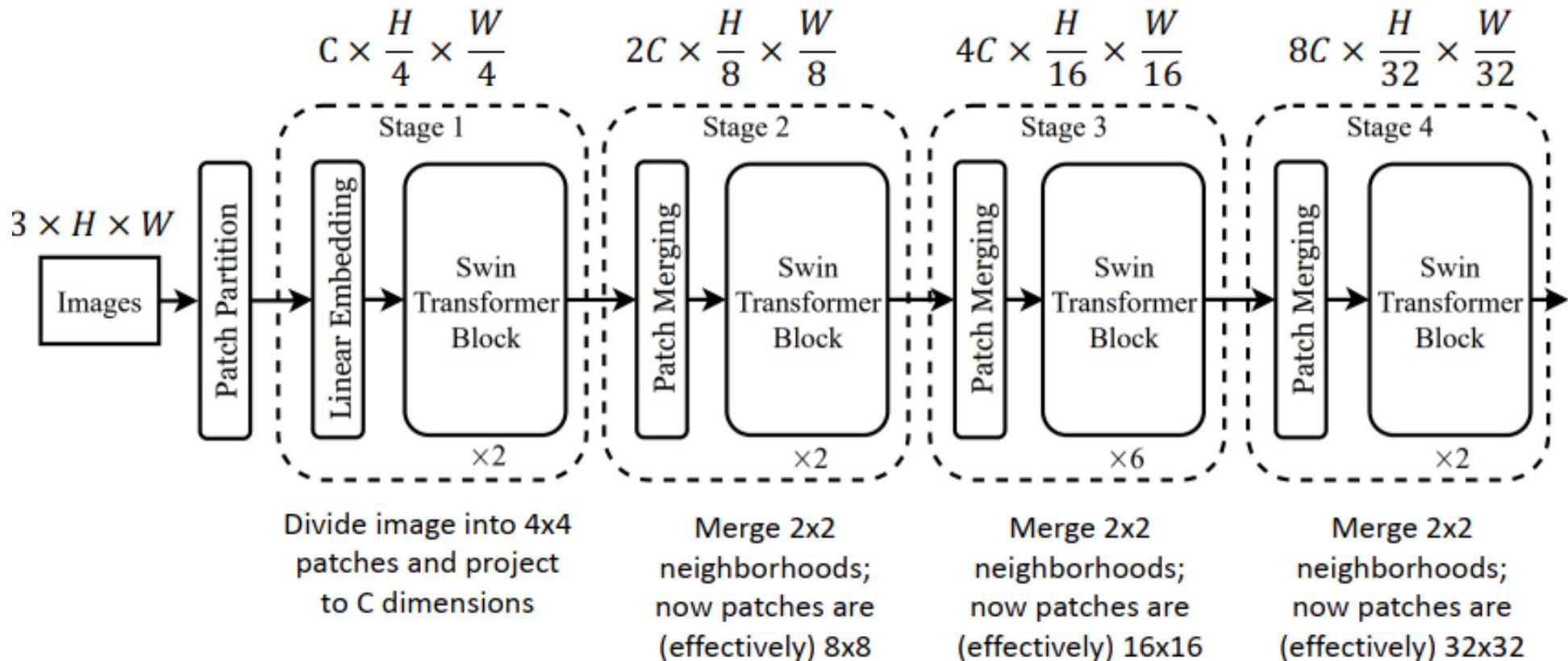
# Swin Transformers



# Swin Transformers

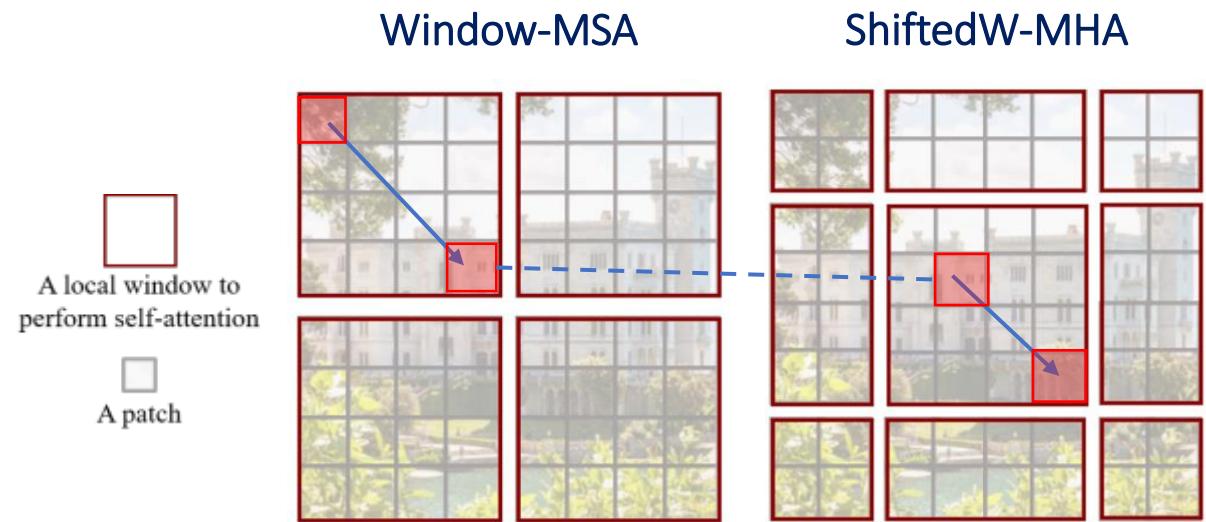
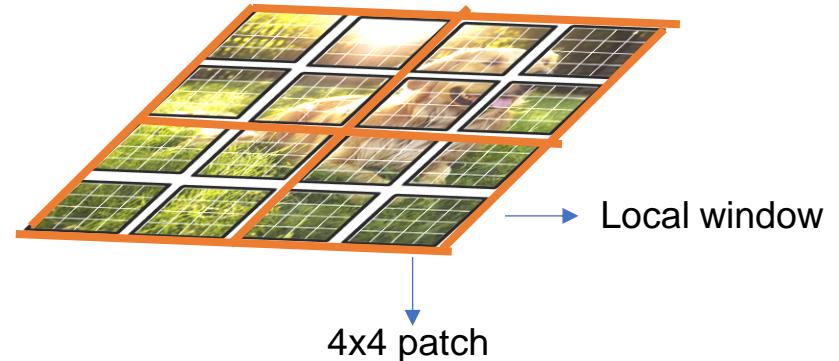
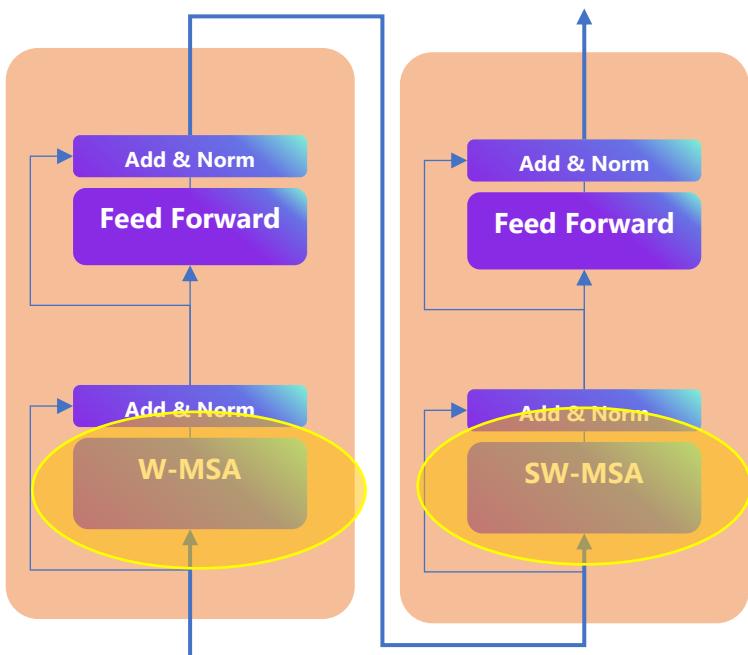


# Swin Transformers



# Swin Transformer Block

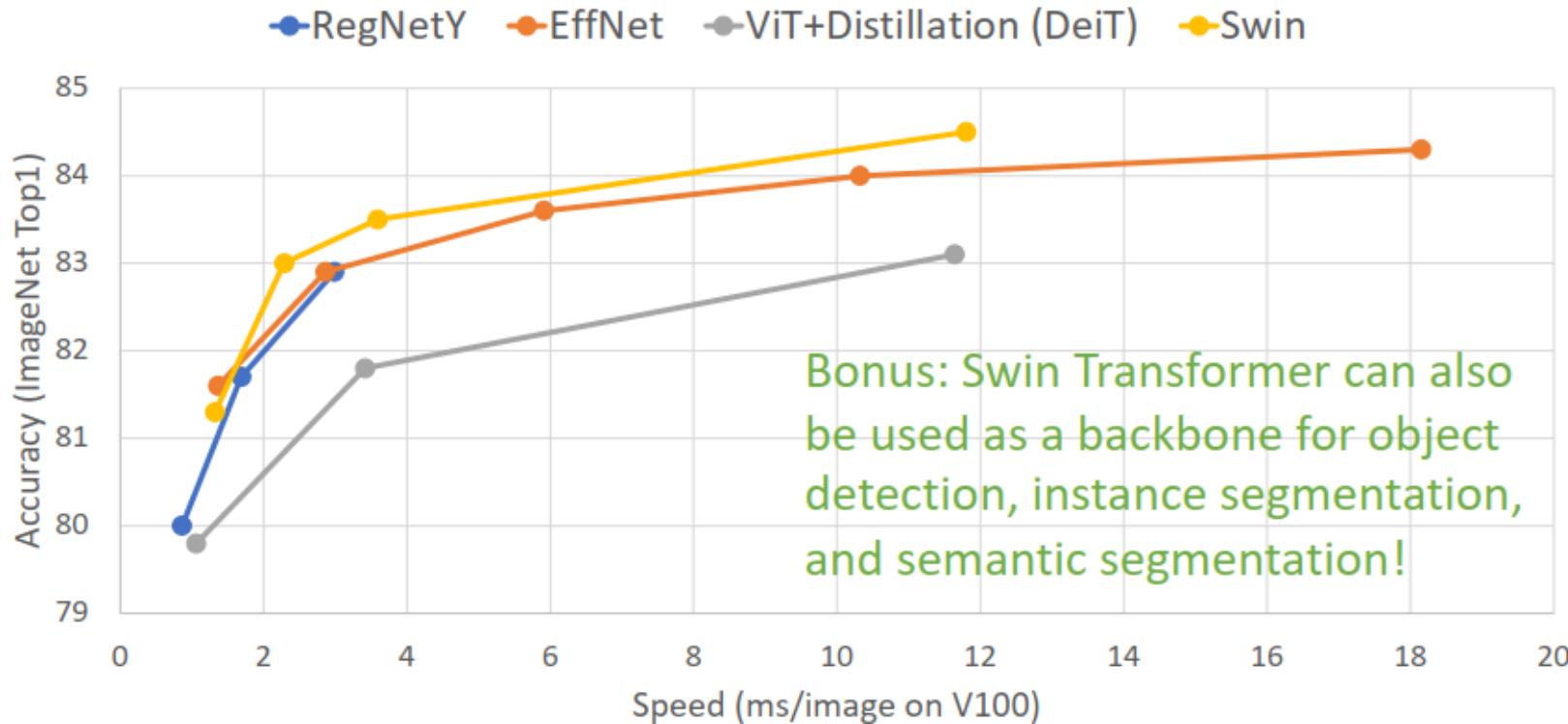
- ✓ Two cascading Transformer Encoder Blocks:
  - ✓ Window Self Attention
  - ✓ Shifted Window Self Attention



**Problem:** tokens only interact with other tokens within the same window; no communication across windows

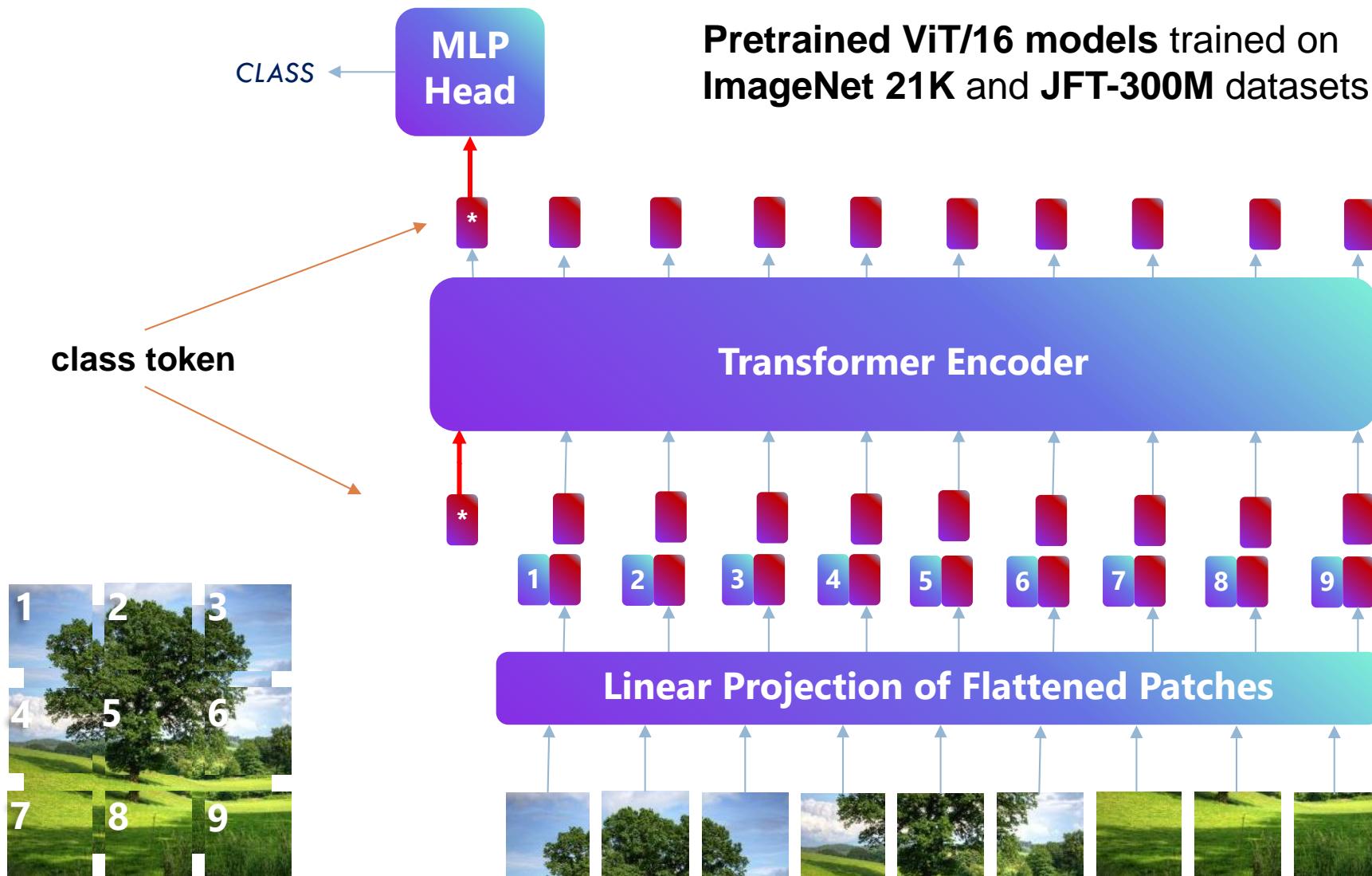
**Solution:** Alternate between normal windows and shifted windows in successive Transformer blocks

# Swin Transformer: Speed and Accuracy

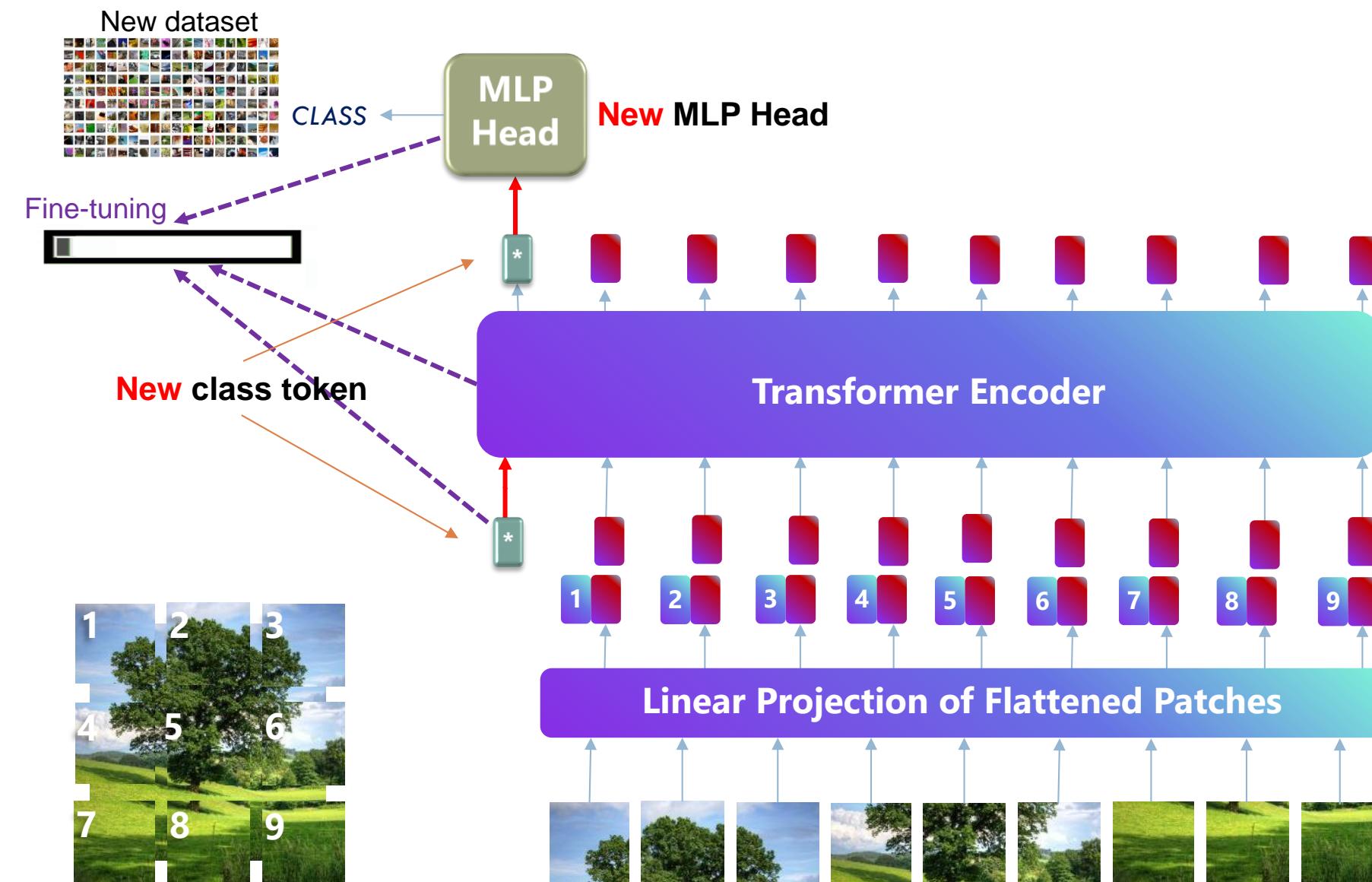


# Model Fine-Tuning

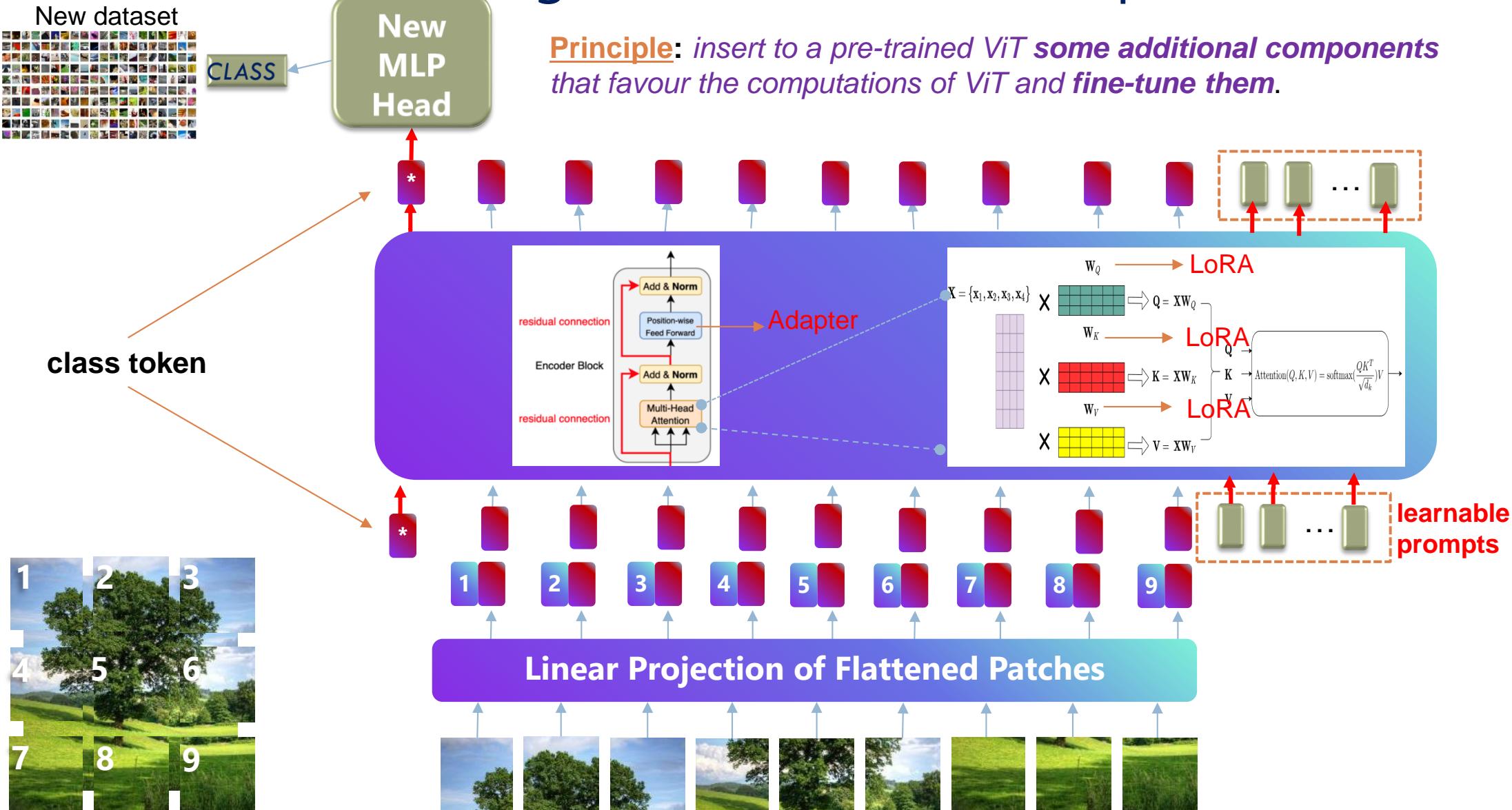
# ViT: Model Fine-Tuning



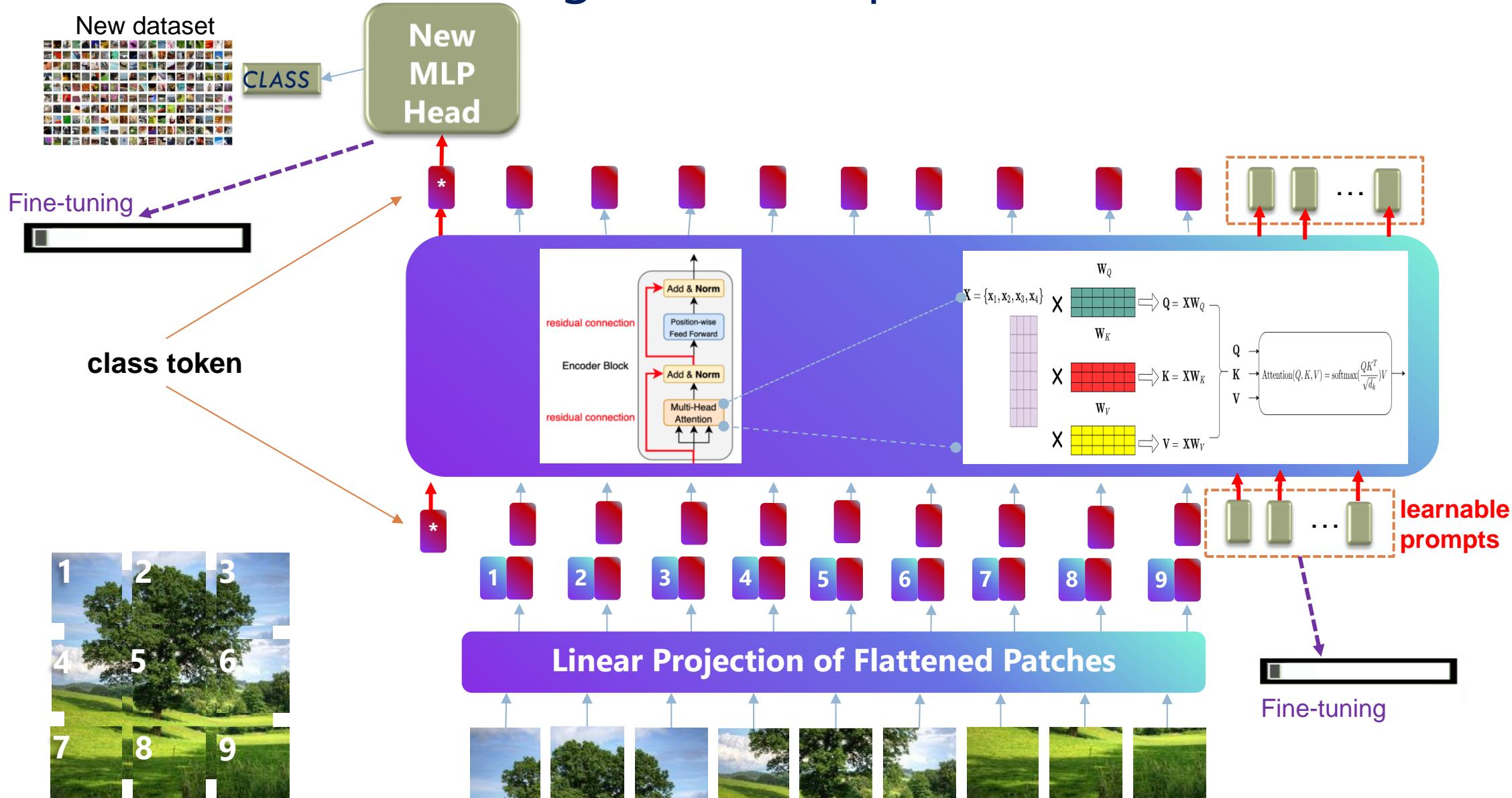
# ViT: Model Fine-Tuning



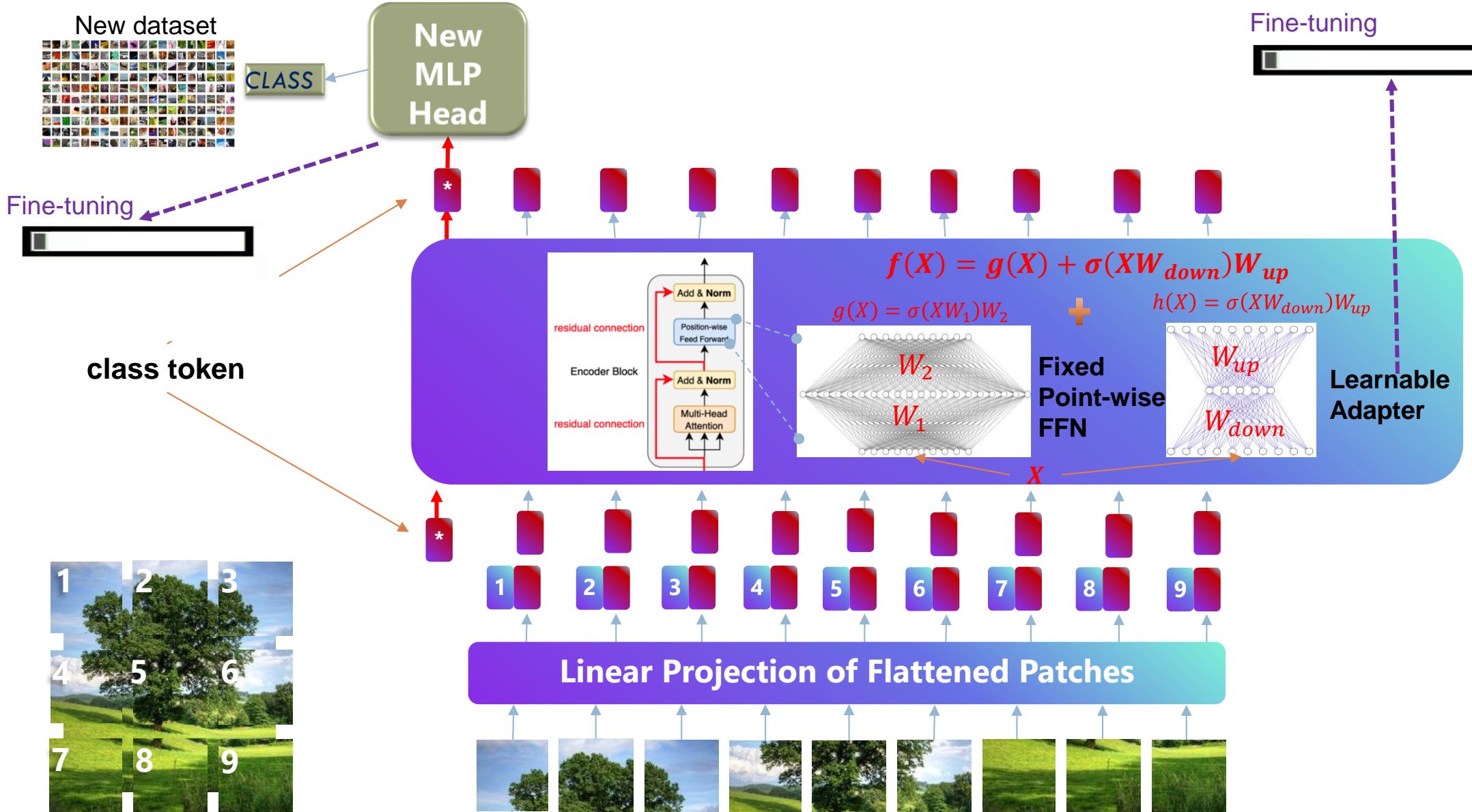
# ViT: Model Fine-Tuning with Additional Components



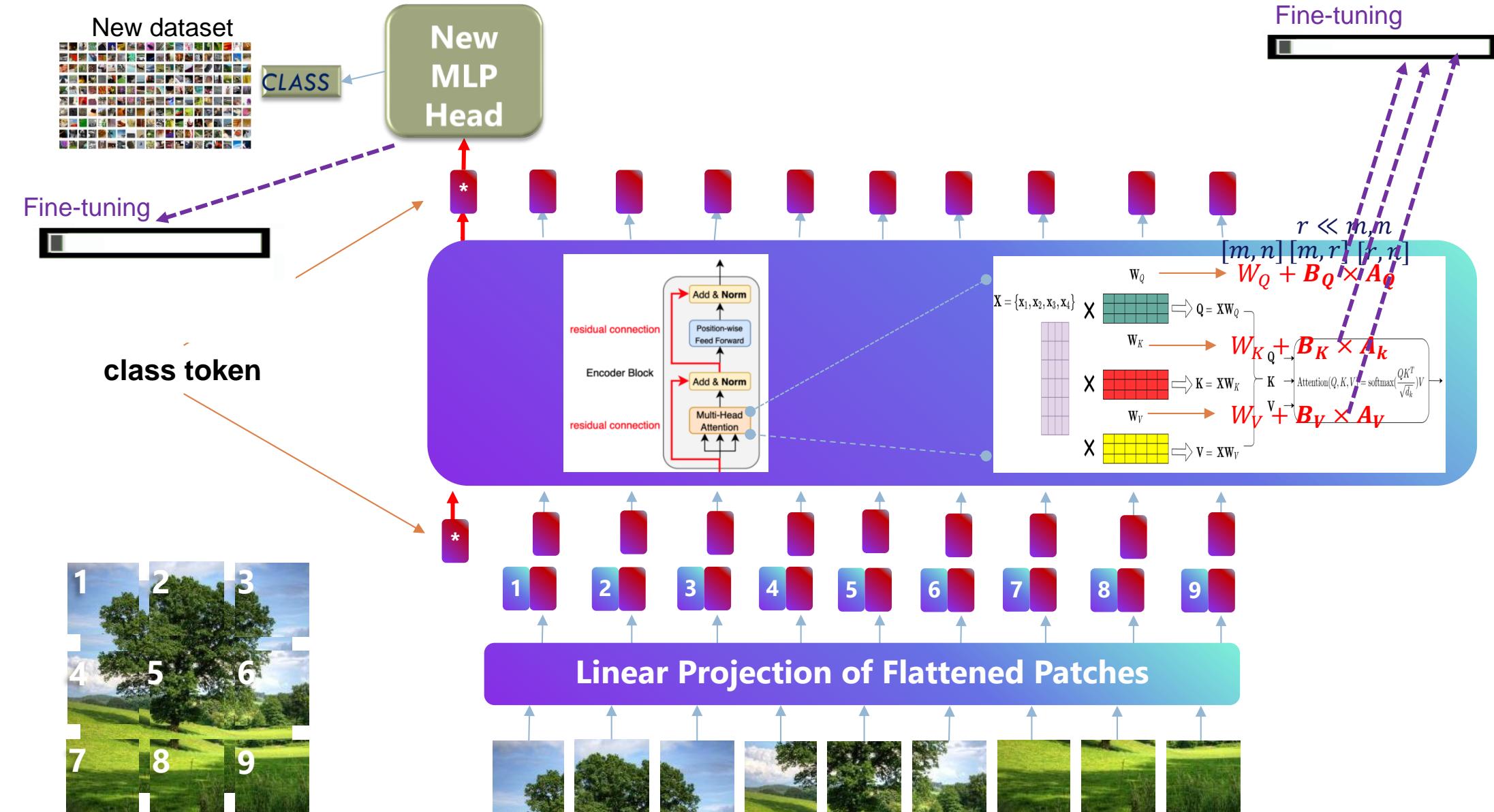
# ViT: Model Fine-Tuning with Prompts



# ViT: Model Fine-Tuning with Adapters



# ViT: Model Fine-Tuning with Additional LoRA



# Summary

- Revision of Transformers
- Vision Transformer
- Swin Vision Transformer
- Model Fine-Tuning for Transformer
  - Prompt-Tuning
  - LoRA
  - Adapter

Thanks for your attention!  
Question time

