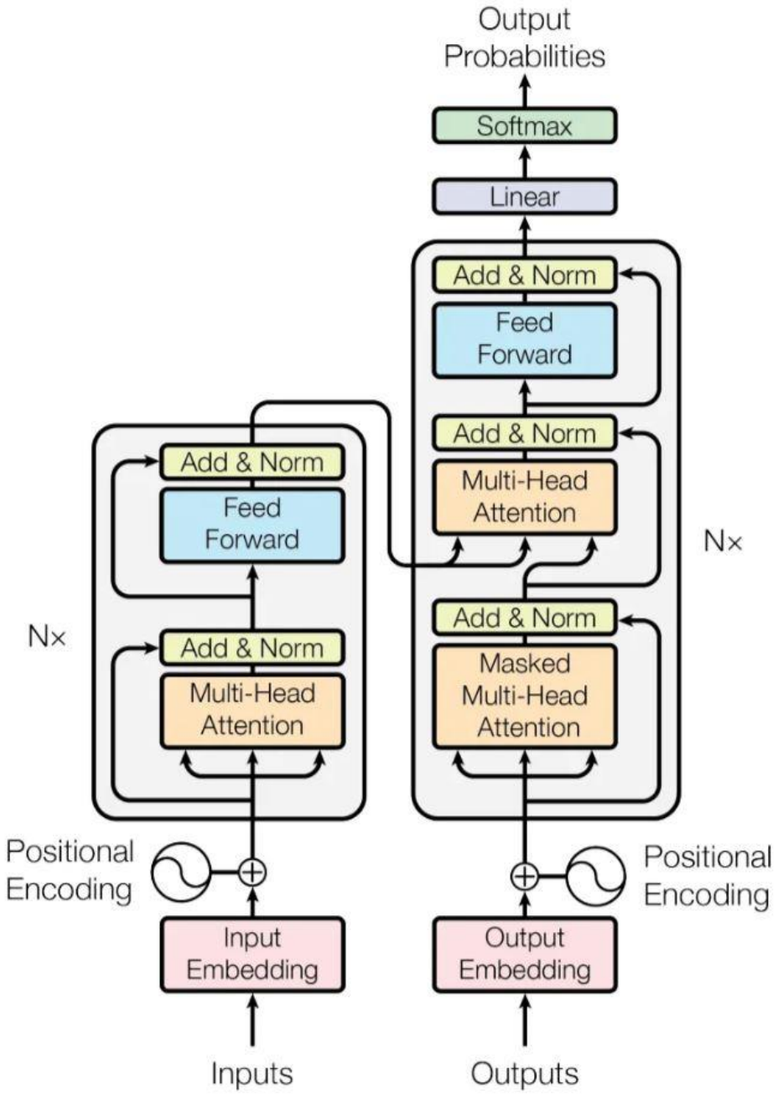
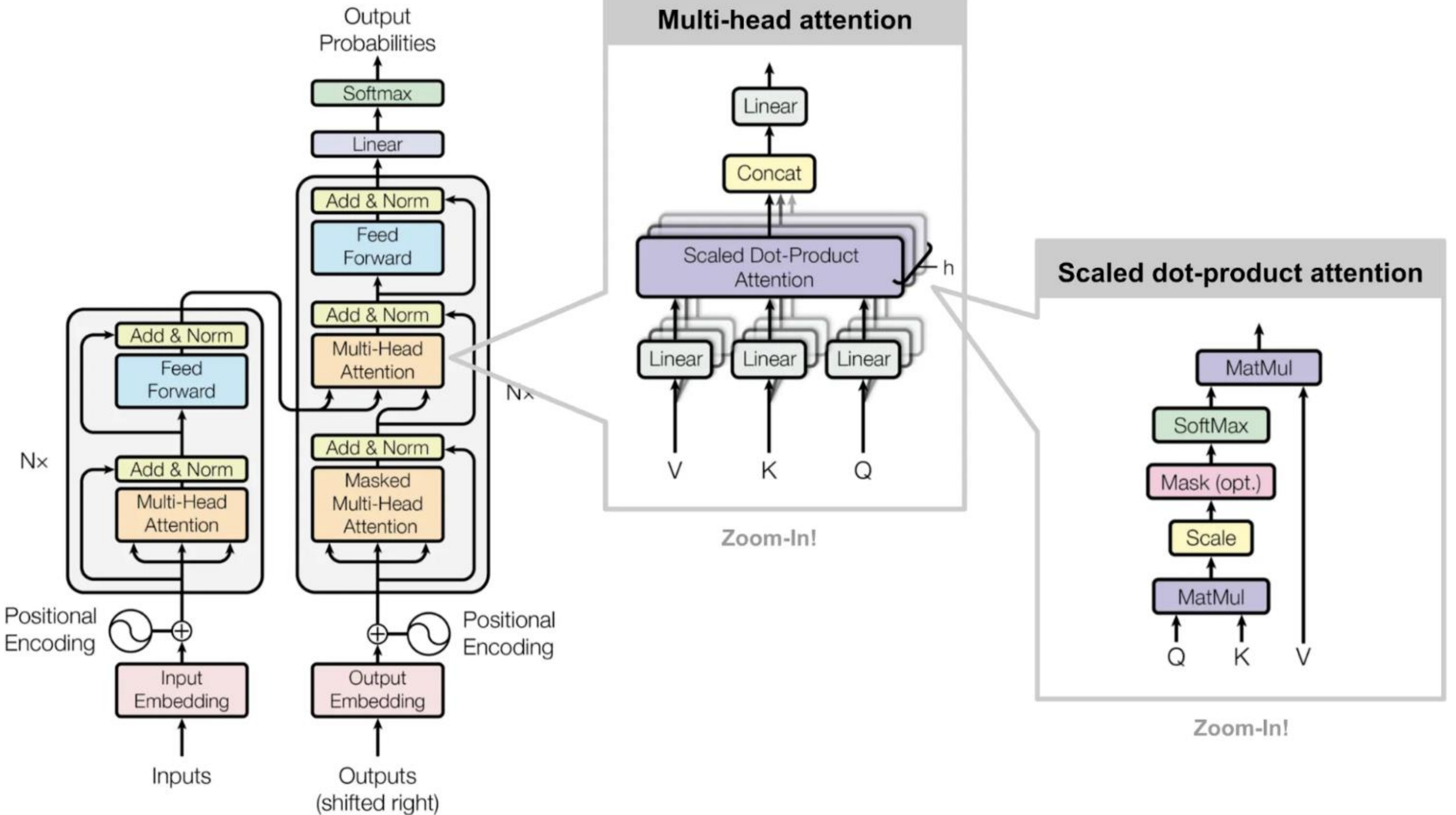
BERT

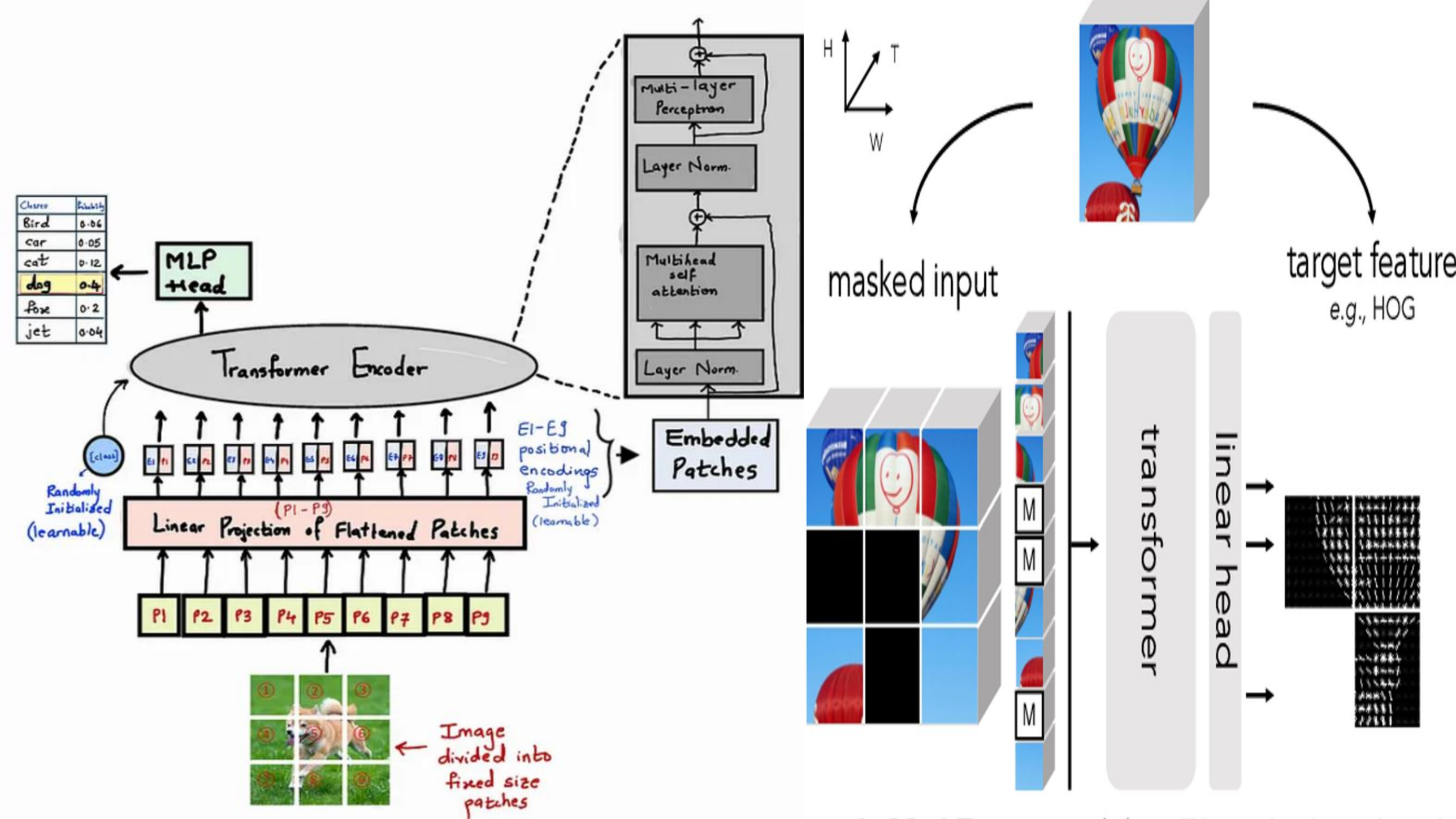
Encoder



GPT

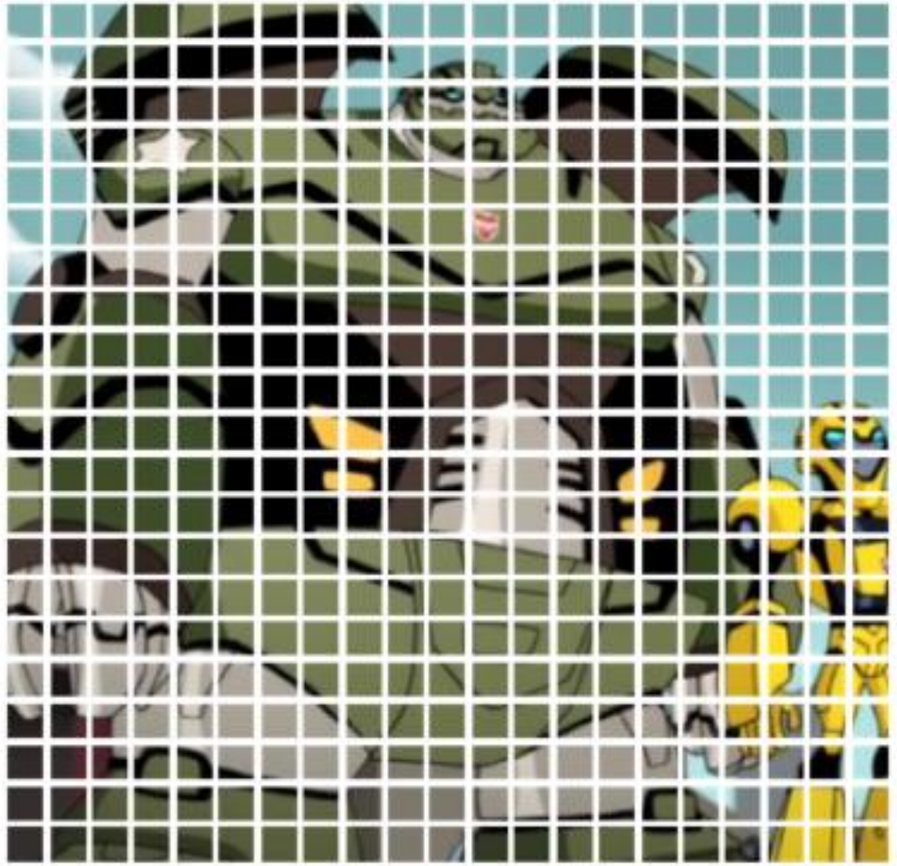
Decoder


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Multi-head attention
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q
Zoom-In!
Scaled dot-product attention
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Zoom-In!

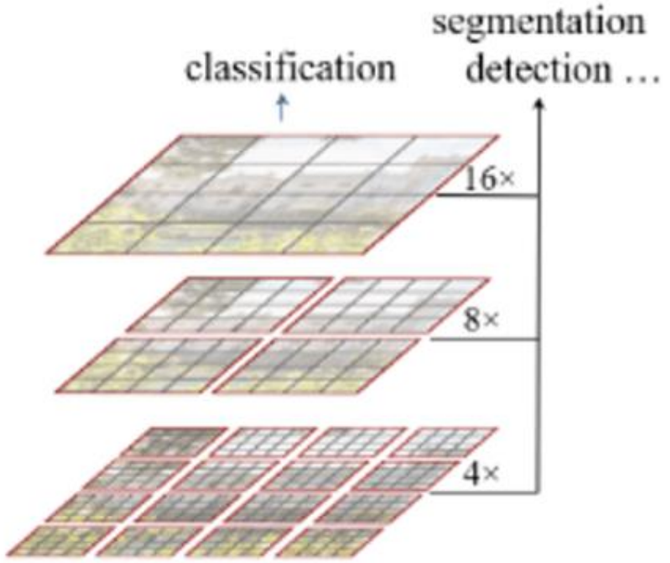

Classes
Bird
car
cat
dog
fox
jet
Probability
0.06
0.05
0.12
0.4
0.2
0.04
MLP Head
Transformer Encoder
[class]
Randomly Initialised (learnable)
(P1 - P9)
Linear Projection of Flattened Patches
E1-E9 positional encodings
Randomly Initialized (learnable)
Embedded Patches
Multi-layer Perceptron
Layer Norm.
Multihead self attention
Layer Norm.
P1
P2
P3
P4
P5
P6
P7
P8
P9
Image divided into fixed size patches
H
T
W
masked input
target feature
e.g., HOG
M
M
M
transformer
linear head

Cropped Image

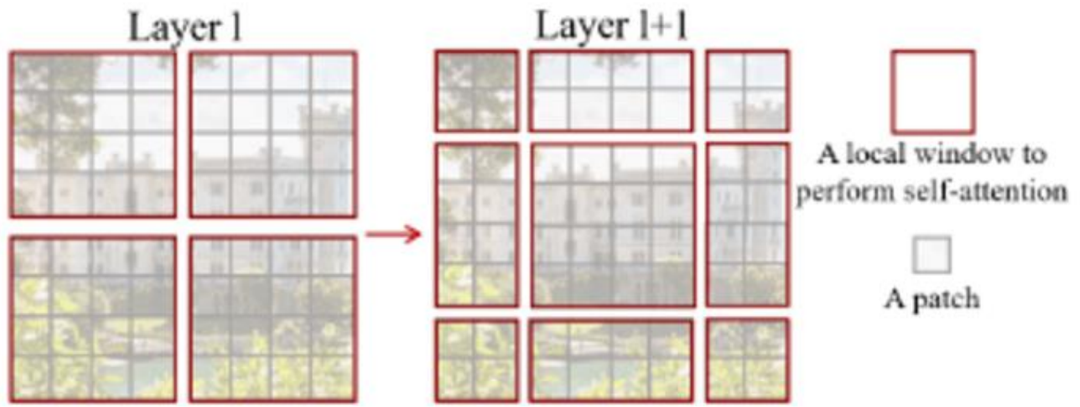
Image Patches



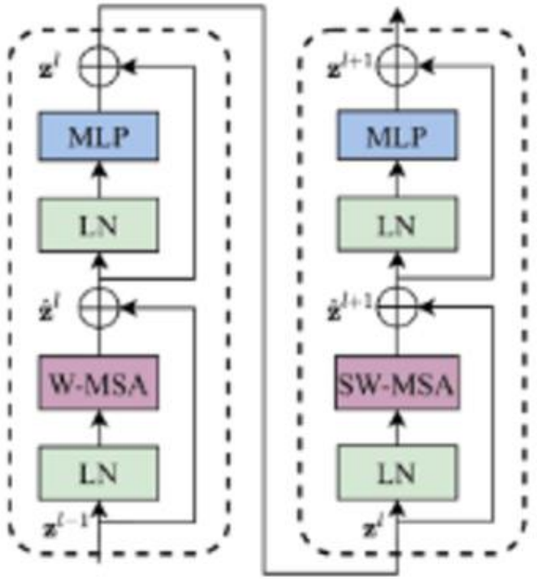
Flattened Image Patches



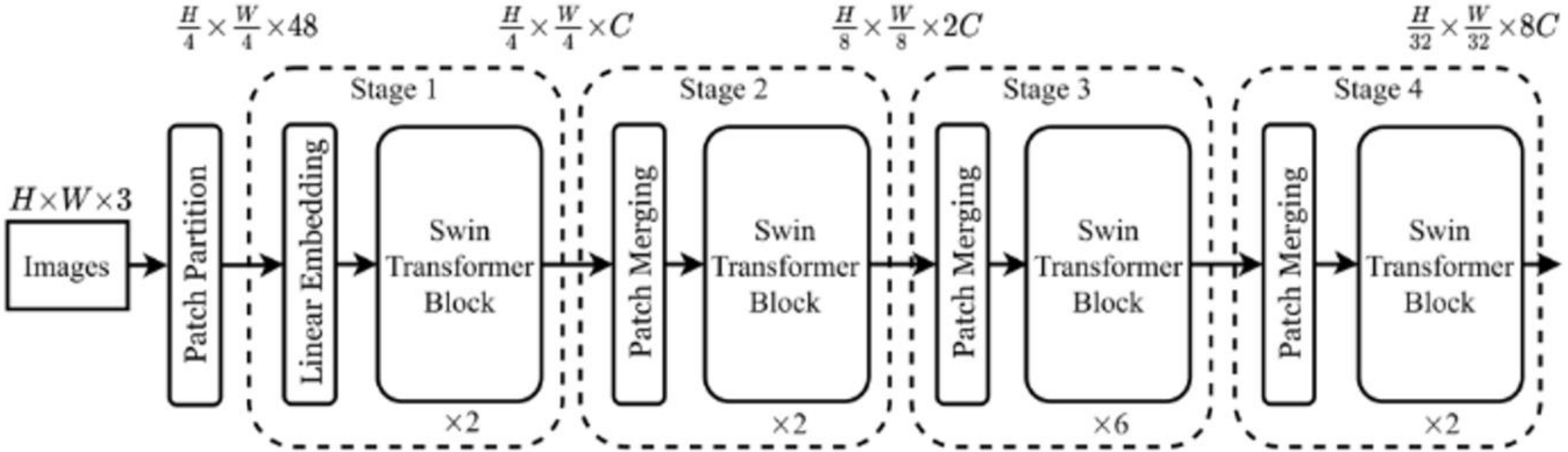
(a) Swin Transformer



(b) Shifted Window



(c) Two Successive Swin Transformer Blocks



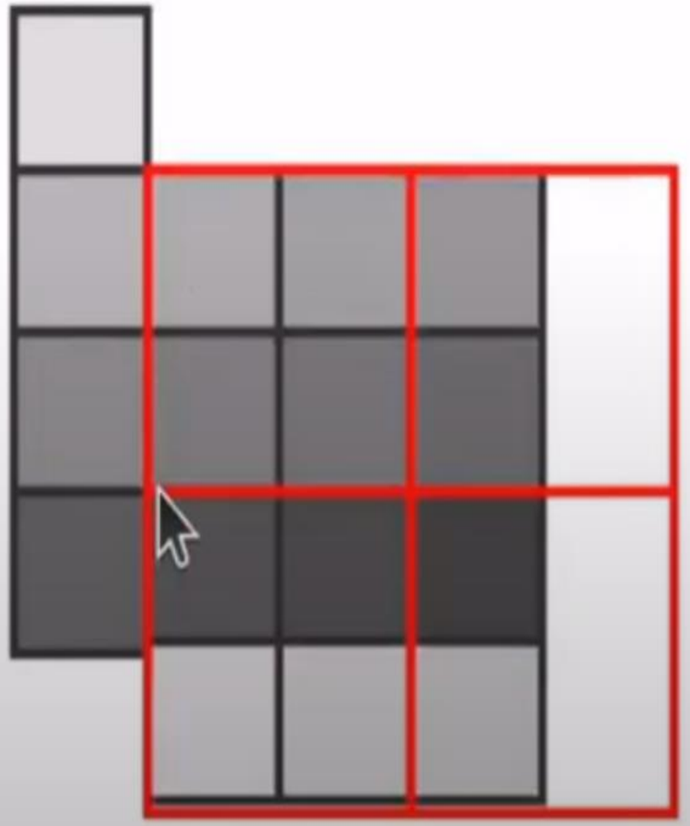
(d) Architecture

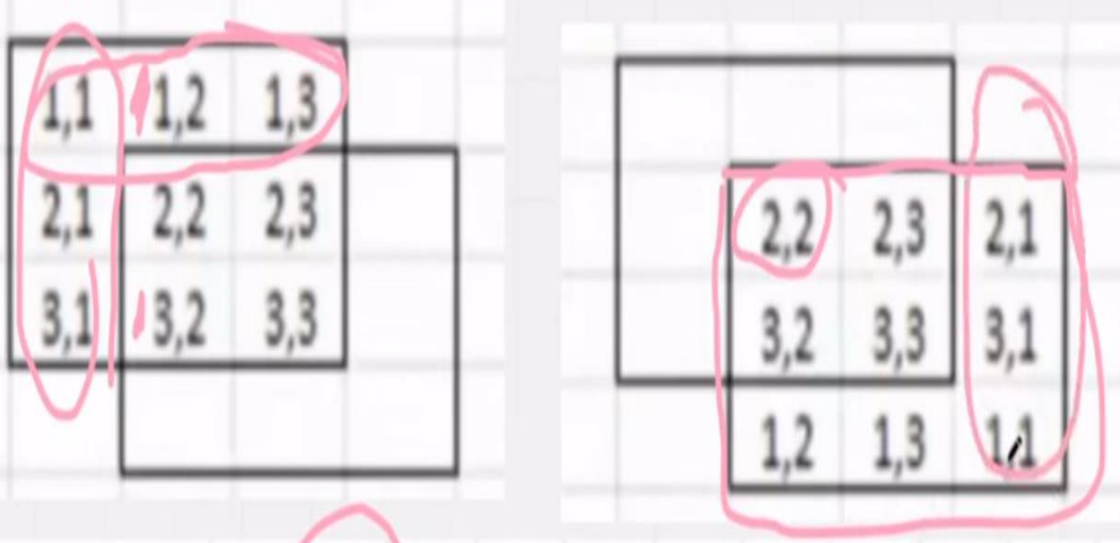
Shifted Window MSA

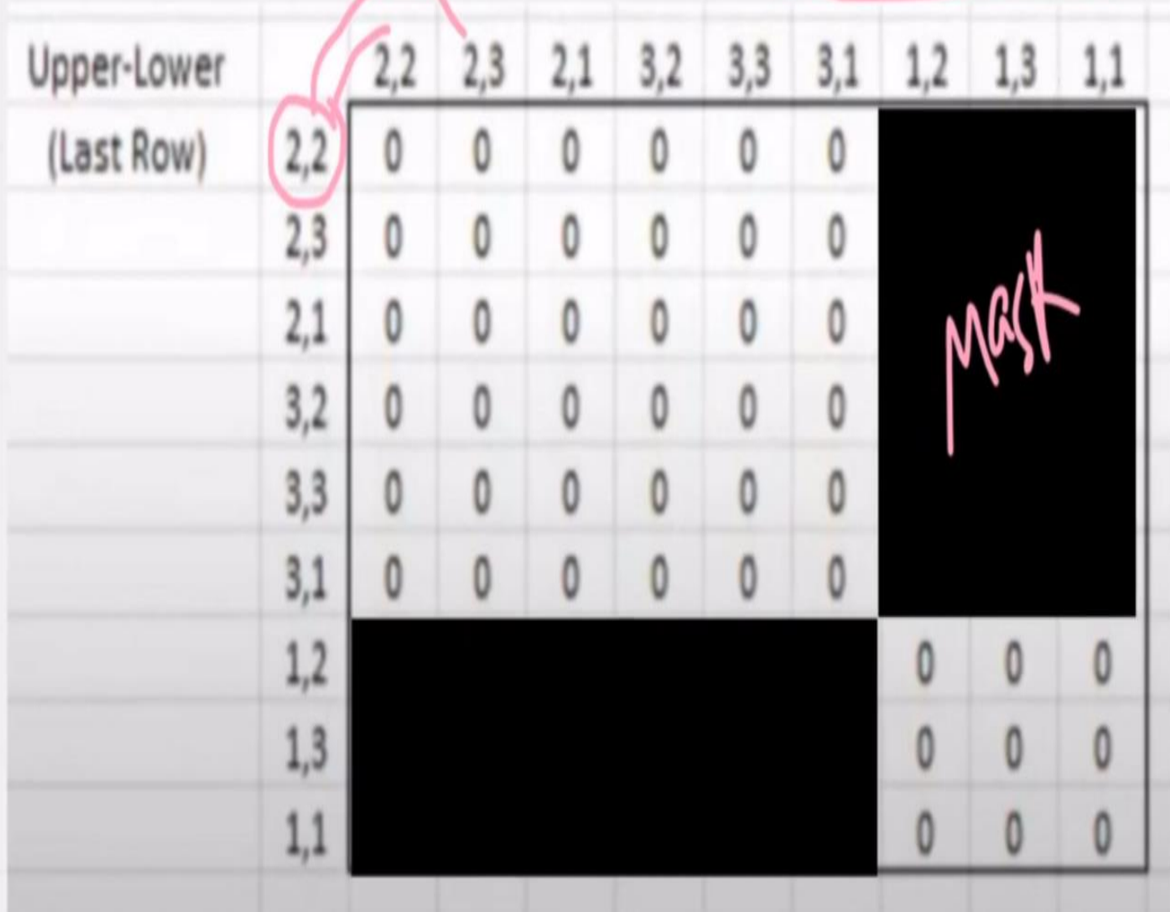
Step 1: Shift window by a factor of M/2, where M = window size

Step 2: For efficient batch computation, move patches into empty slots to create a complete window.

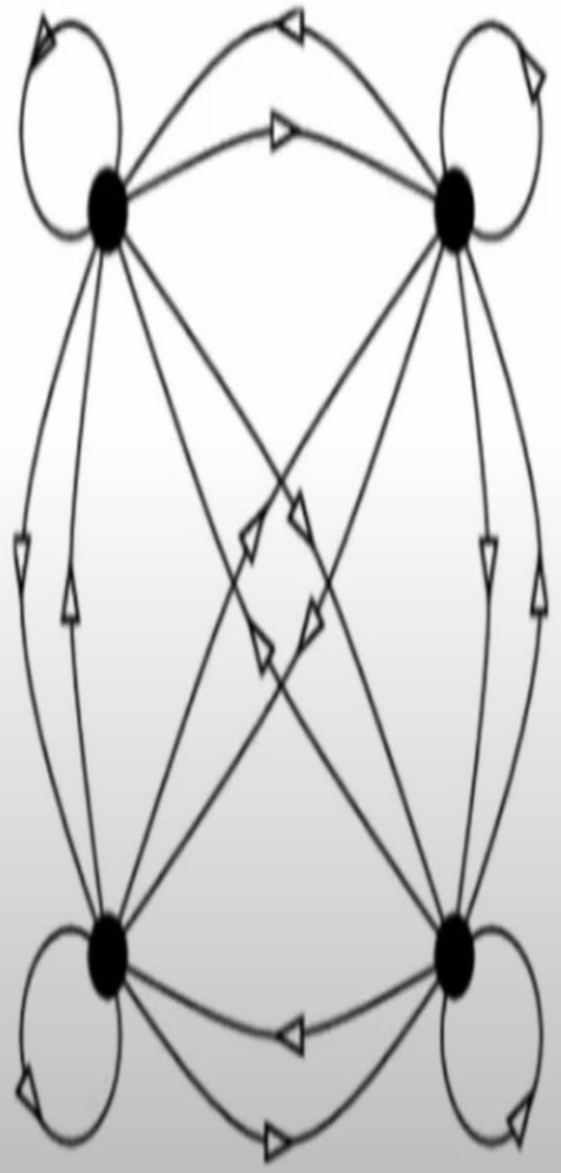
This is known as 'cyclic shift' in the paper.







-1, -1	-1, 0	-1, 1	-1, 2
0, -1	0, 0	0, 1	0, 2
1, -1	1, 0	1, 1	1, 2
2, -1	2, 0	2, 1	2, 2



Patch Merging

Assuming that n=2, and each group consists of 2x2 neighboring patches

Step 1: Split input image into groups of 2x2
Step 2: In each group, stack the patches depth-wise
Step 3: Combine the stacked groups