Introduction of Transformer

Presenter:

Prof. Pau-Choo Chung

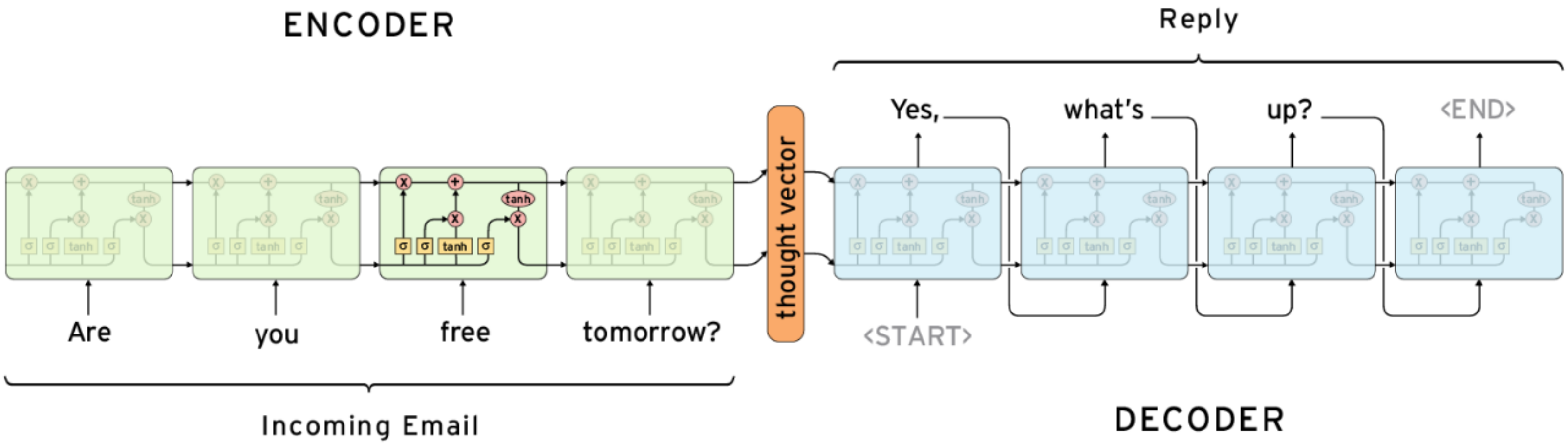
Outline

- Introduction
- Architecture
- Applications

Introduction

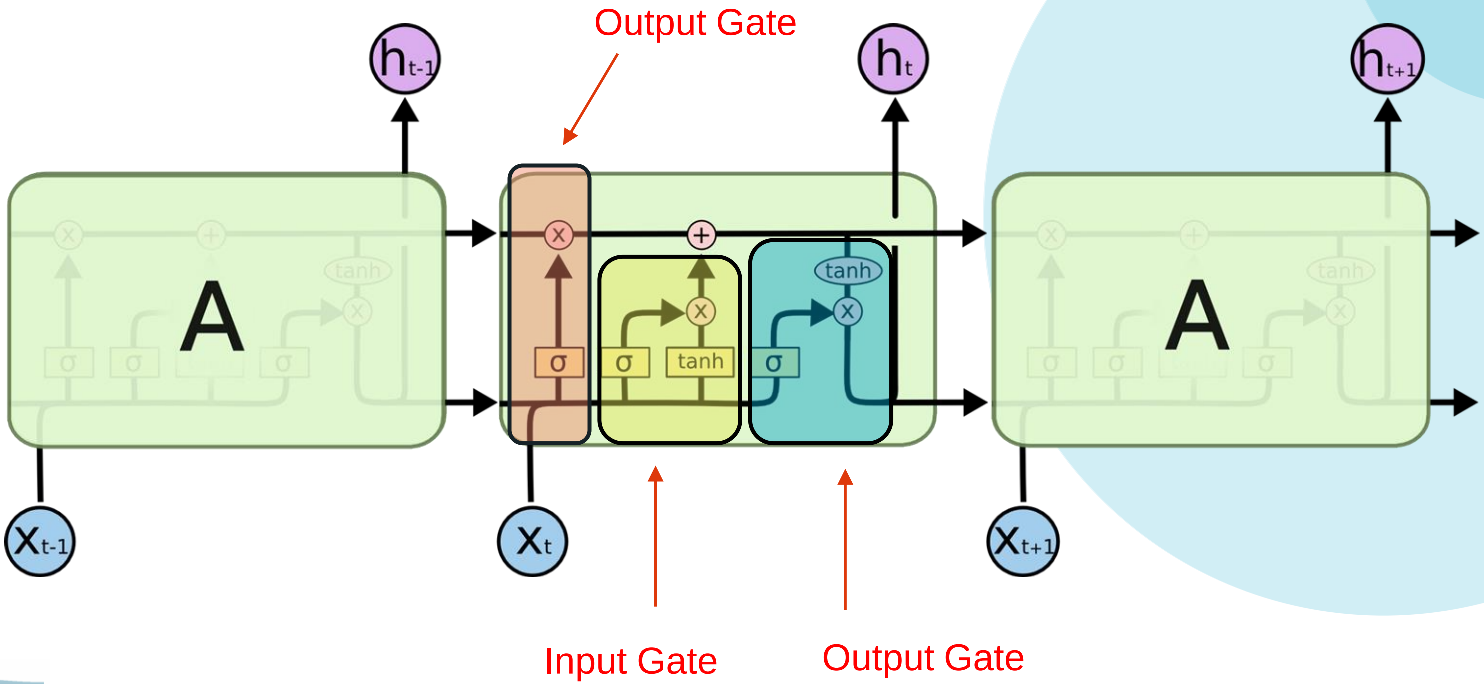
Seq2Seq

- Encoder-Decoder model

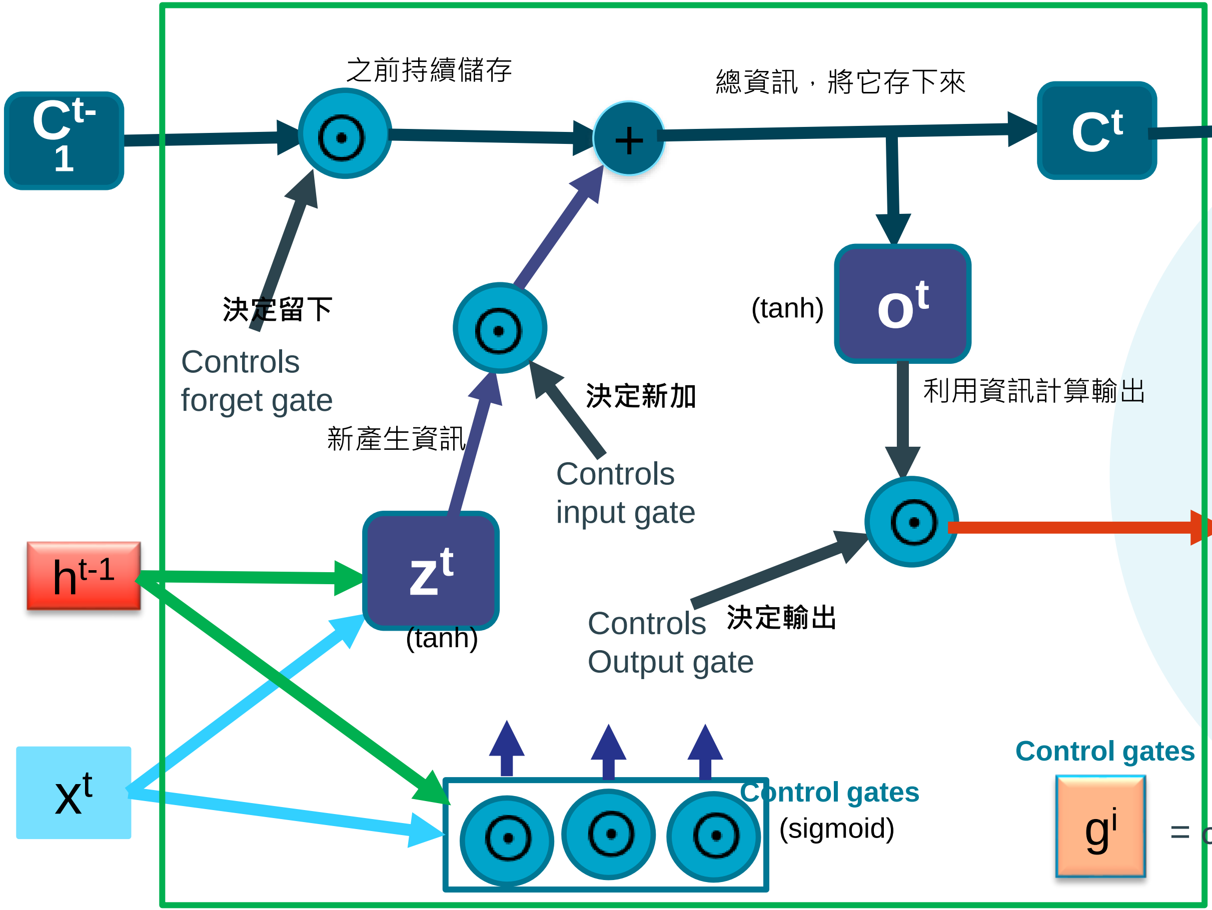


LSTM

- Long Short-Term Memory



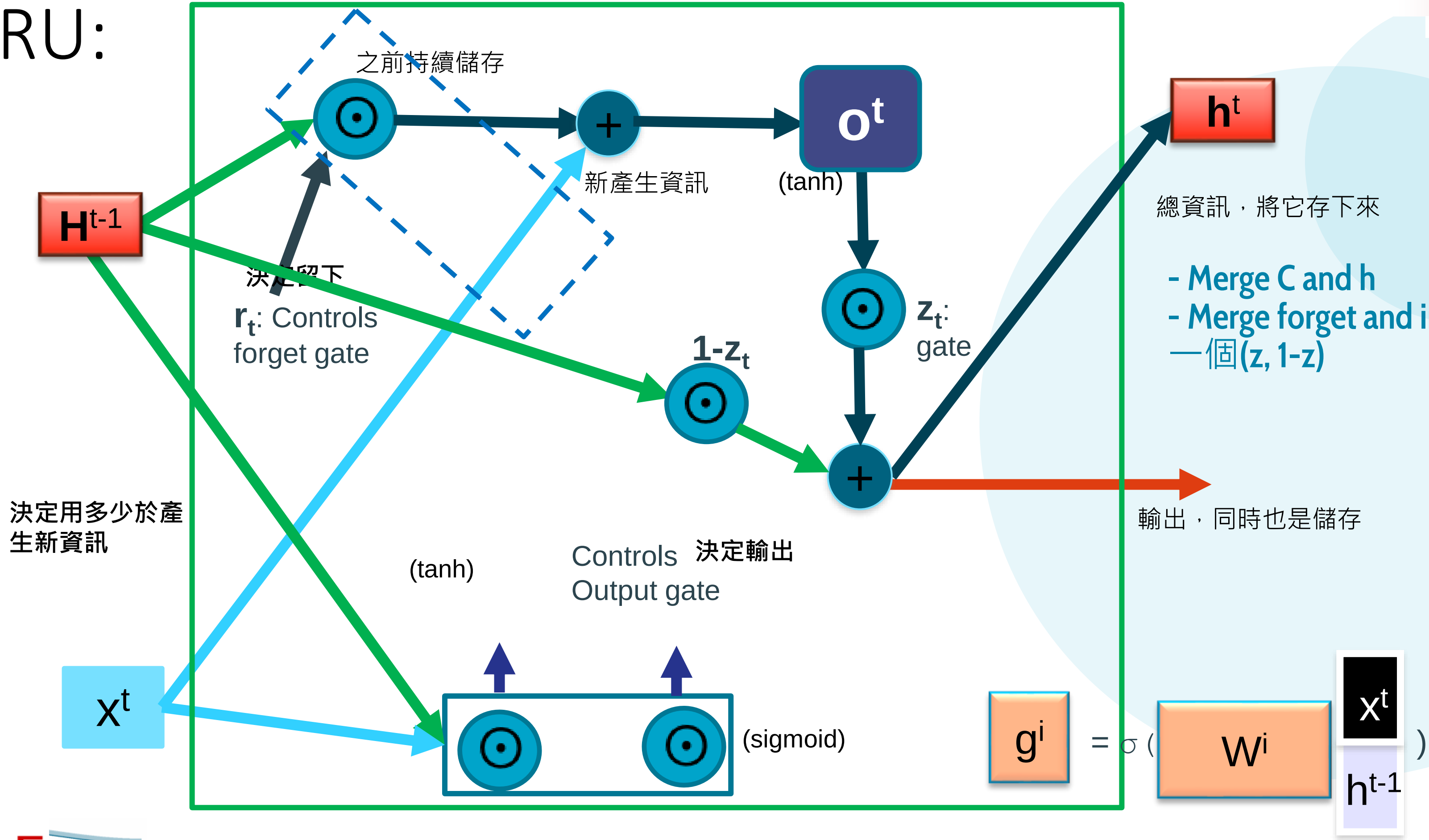
LSTM



Control gates

$g^i = \sigma\left(W^i \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$

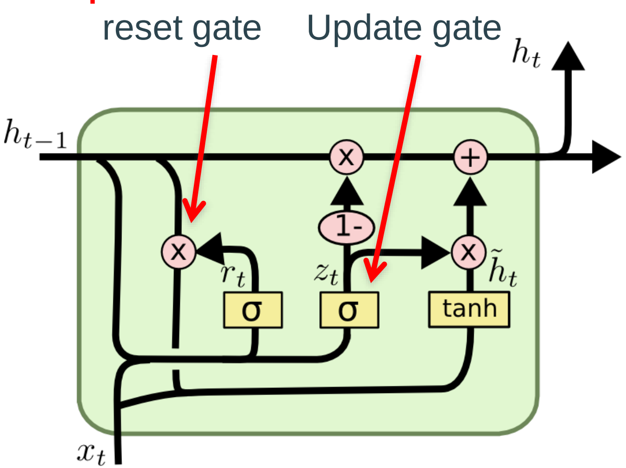
GRU:



- Merge C and h
- Merge forget and input成為一個(z, 1-z)

GRU – gated recurrent unit

(more compression)

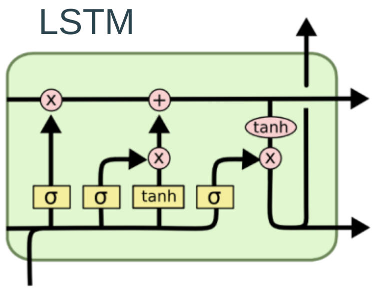


$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

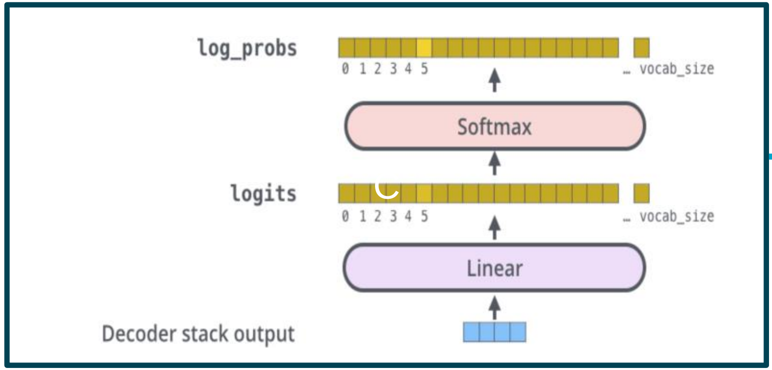
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

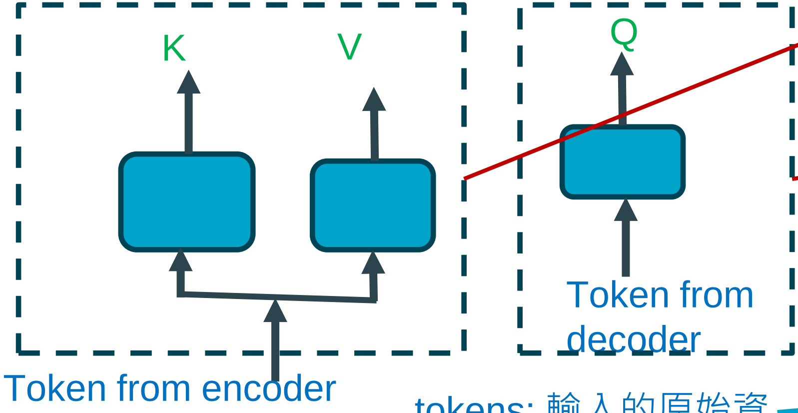


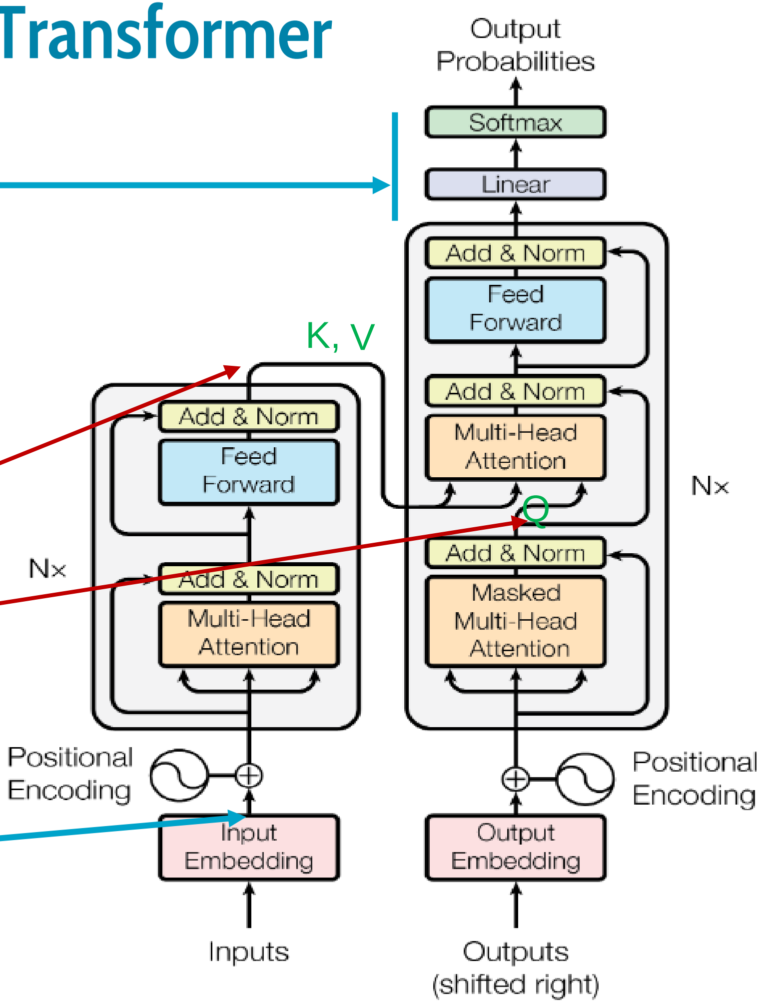
It combines the forget and input into a single update gate.
It also merges the cell state and hidden state. This is simpler than LSTM. There are many other variants too.

X,*: element-wise multiply

Transformer



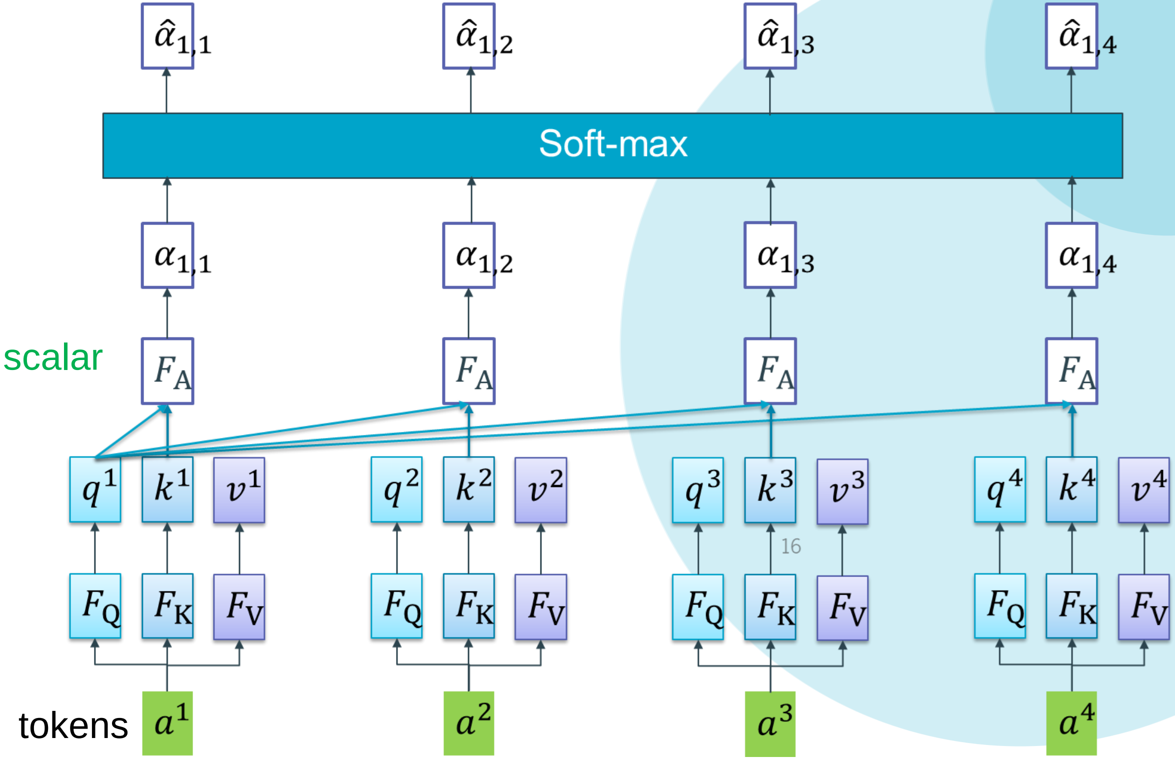




tokens: 輸入的原始資料經過一個網路，產生embedding features, 為tokens

Self-attention

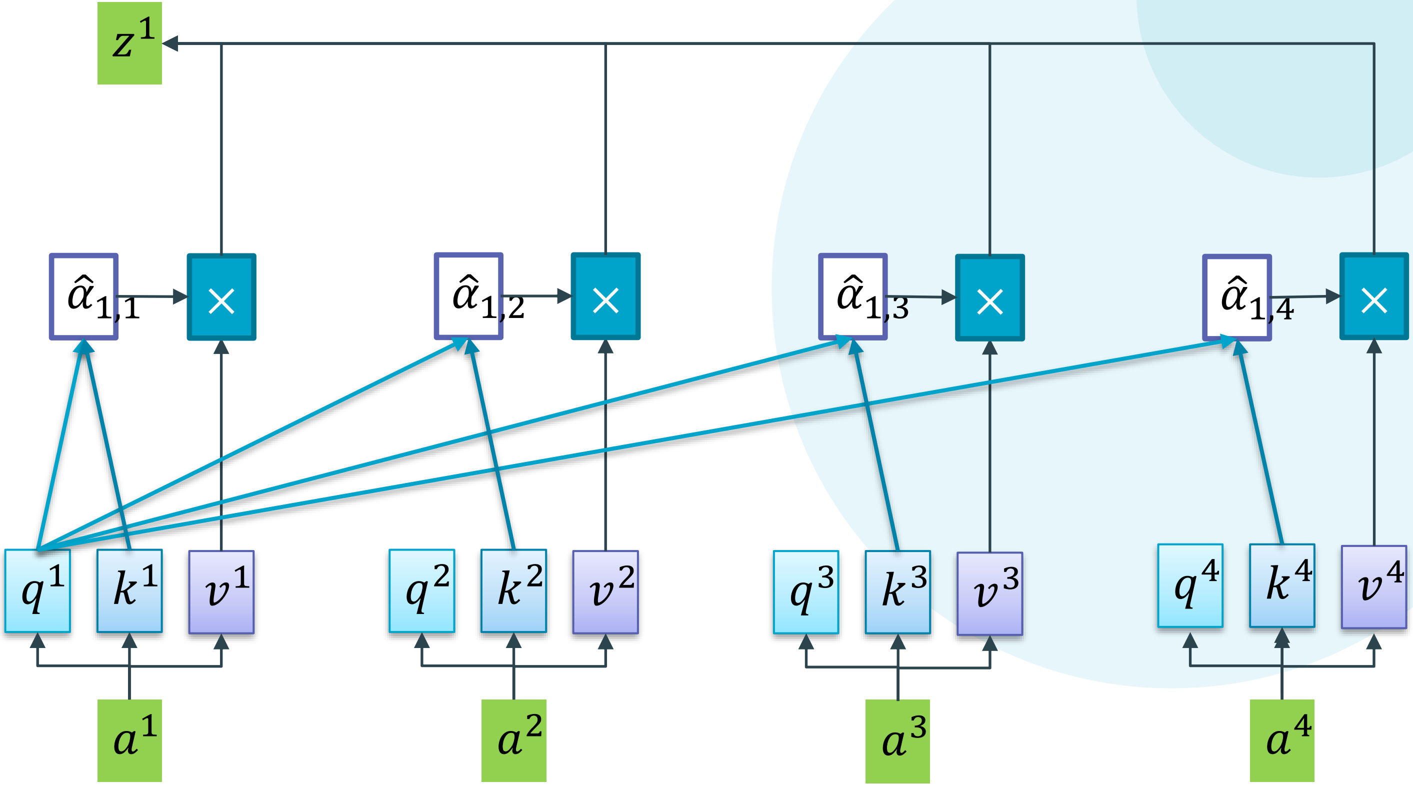
- For fast computation, use scaled dot-product attention
- $q^i = F_\mathrm{Q}(a^i) = W_\mathrm{Q} a^i$
- $k^i = F_\mathrm{K}(a^i) = W_\mathrm{K} a^i$ vectors
- $v^i = F_\mathrm{V}(a^i) = W_\mathrm{V} a^i$
- $\alpha_{i,j} = F_\mathrm{A}(q^i, k^j) = q^i \cdot k^j / \sqrt{d}$ scalar
- $\hat{\alpha}_{i,j} = \exp \alpha_{i,j} / \sum_m \exp \alpha_{i,m}$



Self-attention

- $z^i = \sum_m \hat{\alpha}_{i,m} \, v^m$

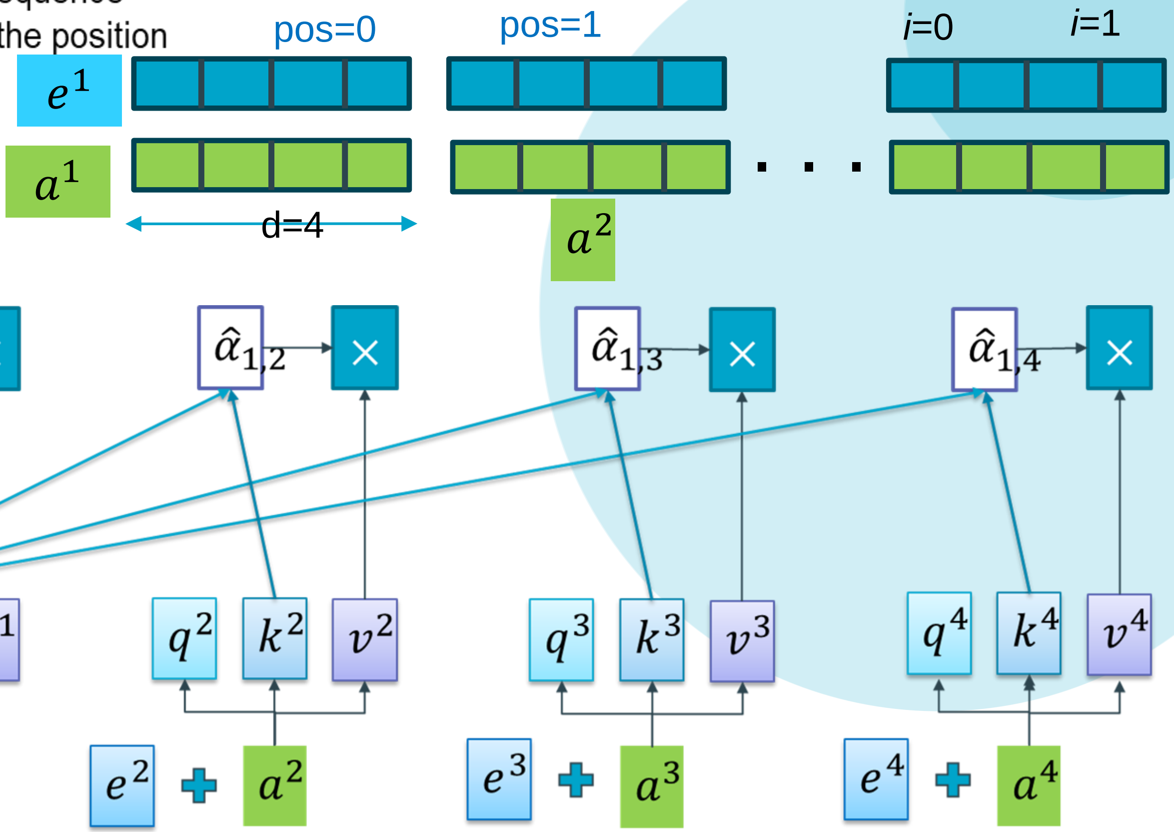
Vectors,每個token
有對應的 feature
vector z



Positional Encoding

- In self-attention, the order of the tokens is not considered
- The order is important in (language) sequence
- Add an unique encoding to represent the position
- $e_{2i}^{pos} = \sin(pos/10{,}000^{2i/d_{\text{model}}})$
  $e_{2i+1}^{pos} = \cos(pos/10{,}000^{2i/d_{\text{model}}})$

token維度

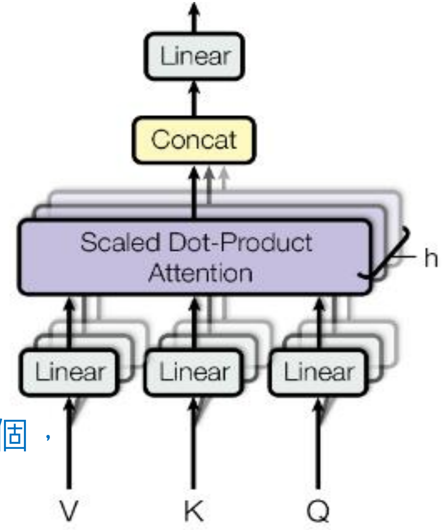


直接數字相加

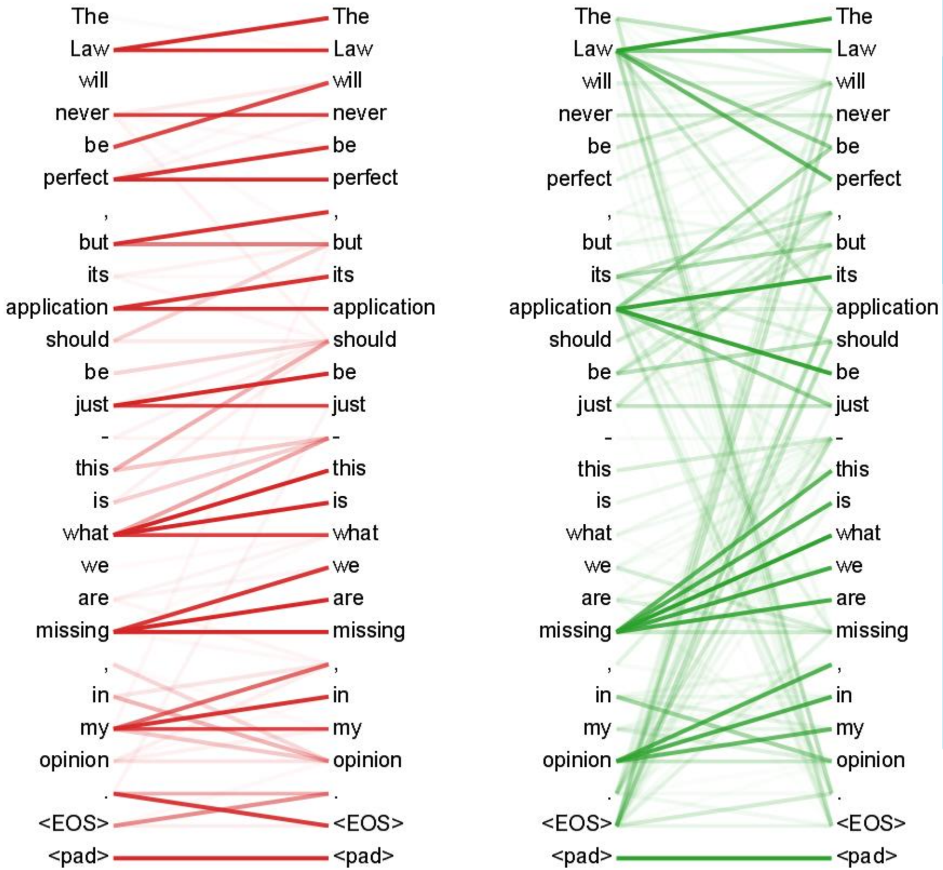
Multi-head Attention

- A single word may have multiple meanings
- Use multiple representations to model it

前面之transformer 架構複製多份，結果 concatenate 後，經過 linear model 進行 fusion mapping



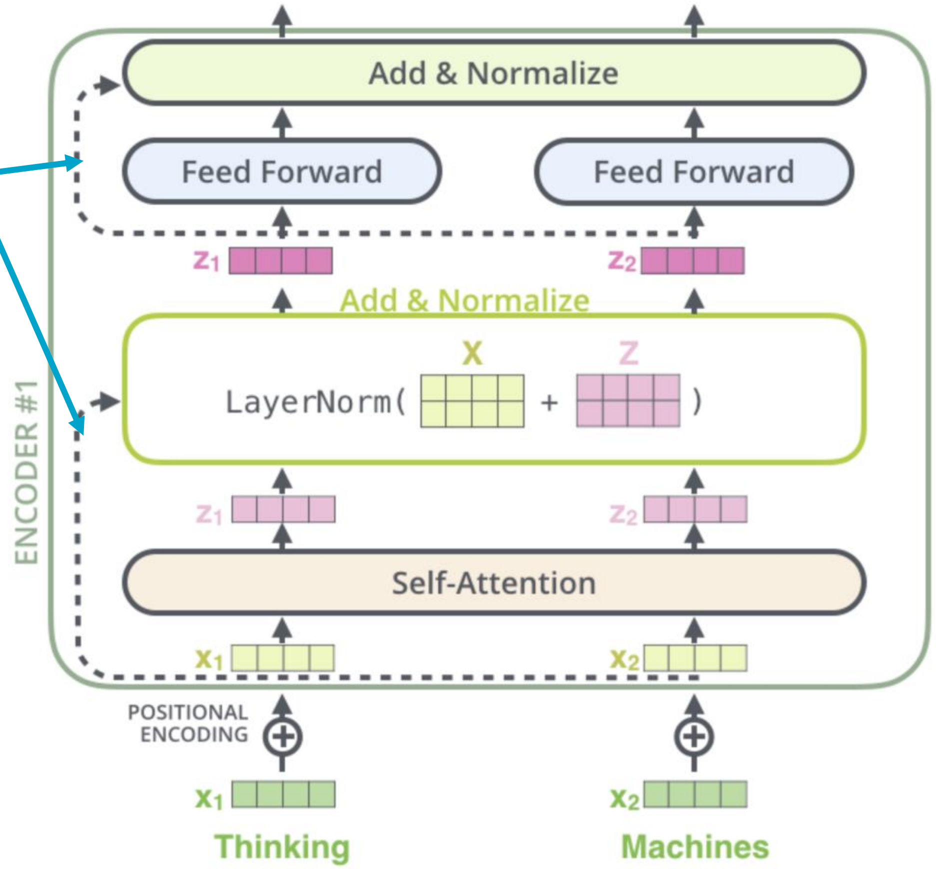
NOTE: Q, K, V為同一個，Embedding 只有一份

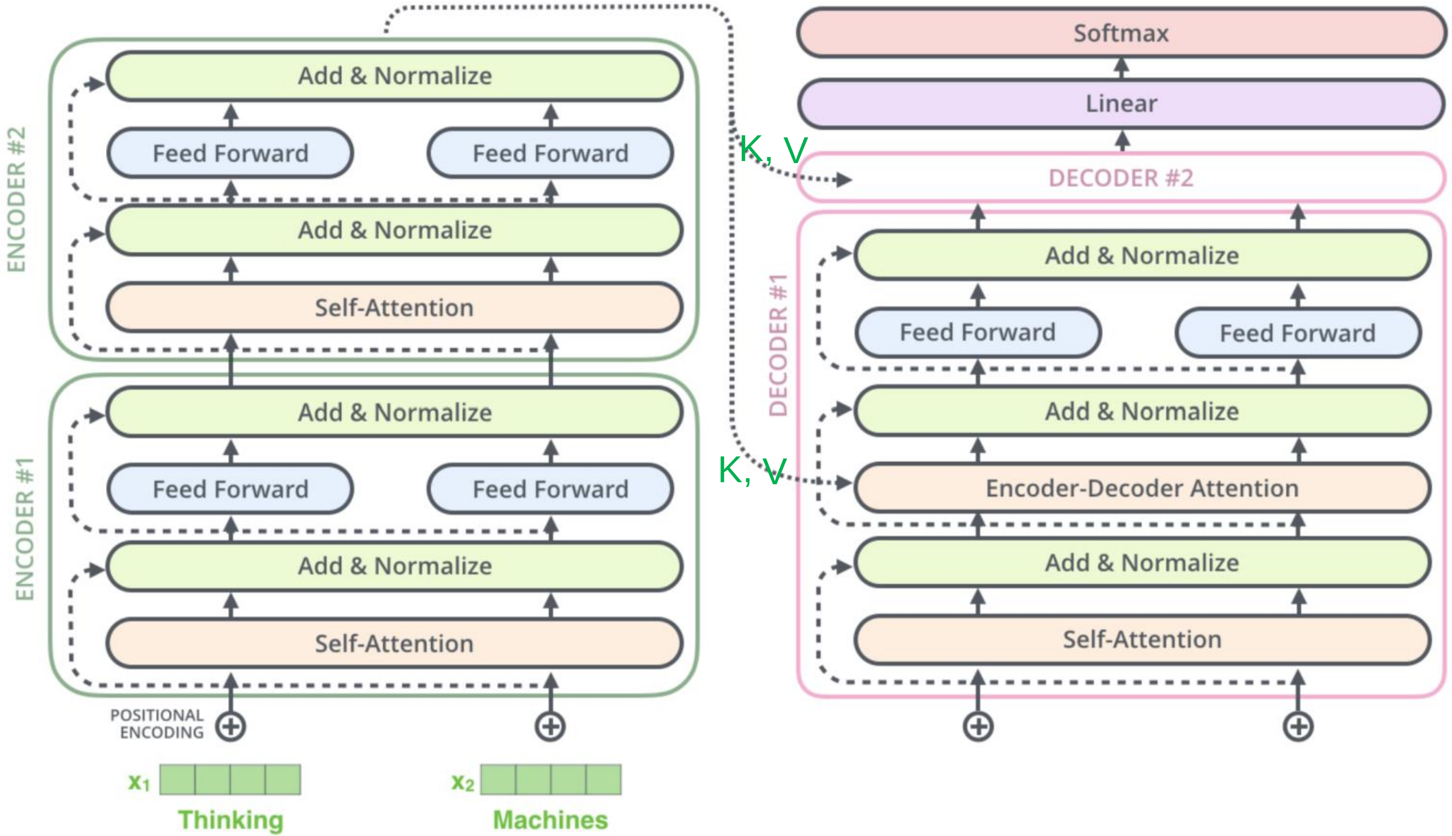


Encoder details

residual link

不同 token 間除了attention,
各自獨立前進，沒有融合

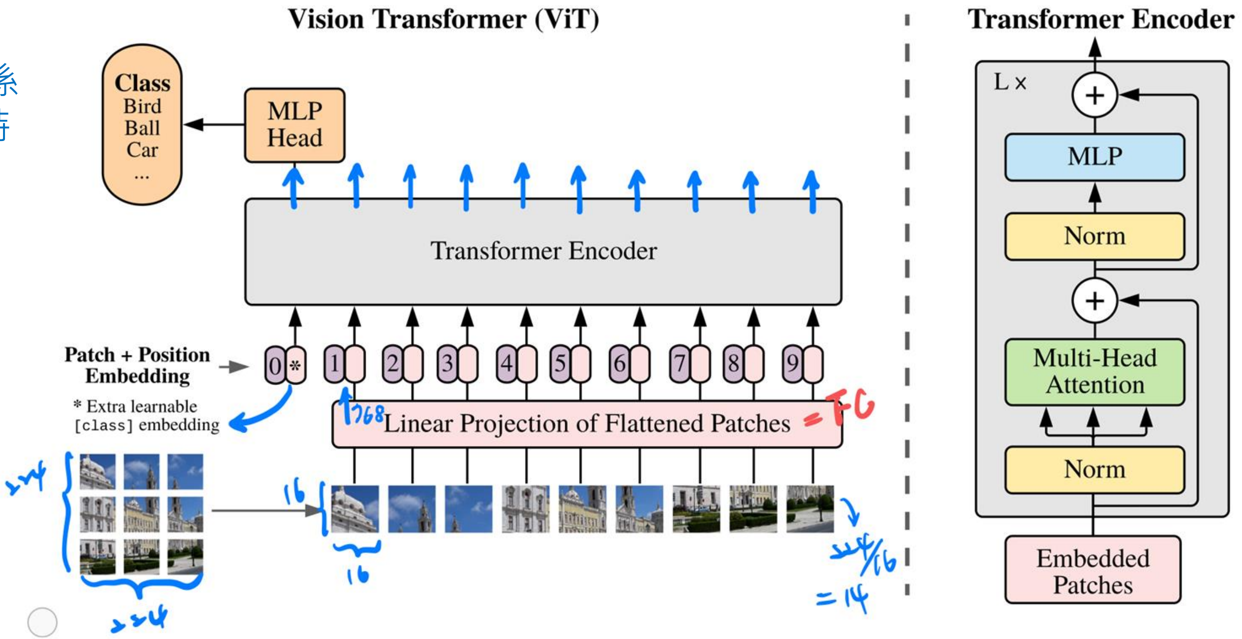




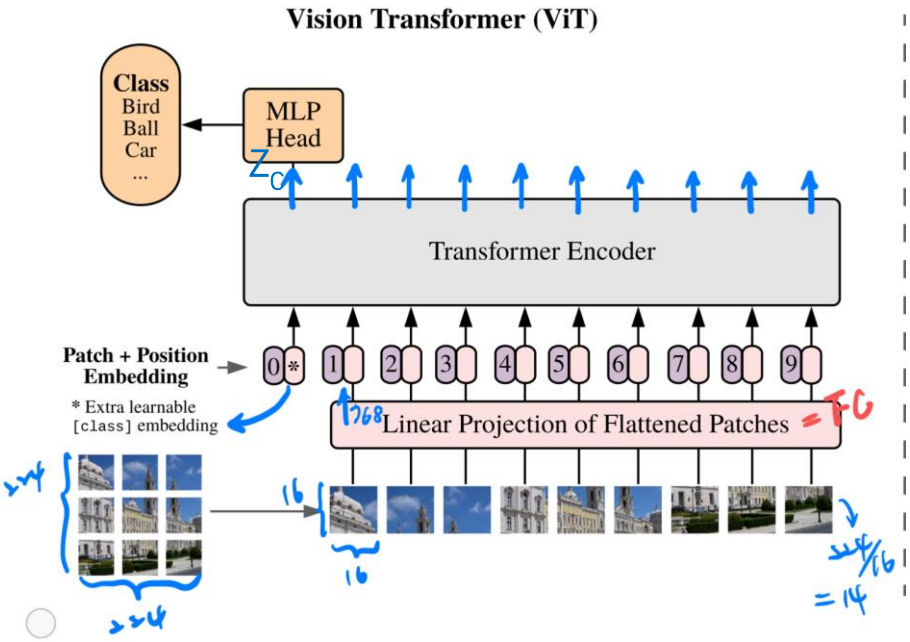
Encoder 提供 K, V 供decoder

Vision Transformer - main framework

- 將影像切成patches
- 利用class embedding 來連結各個token 與類別關係
- 最後用class embedding特徵，經MLP分類



Vision Transformer - main framework

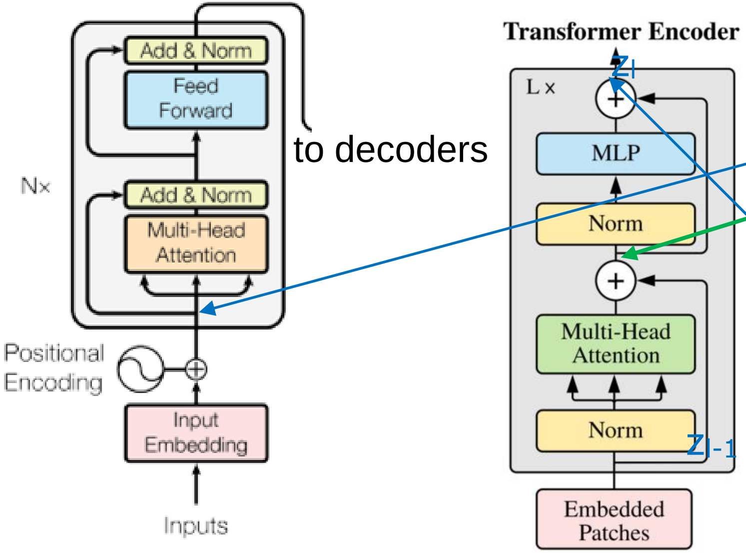


embedded patch

position embedding

prepend learnable embedding
(給固定class token， 一樣學F 涵式，轉成K, Q, V，和其他影像token 算出其值z_c)

Vision Transformer - main framework



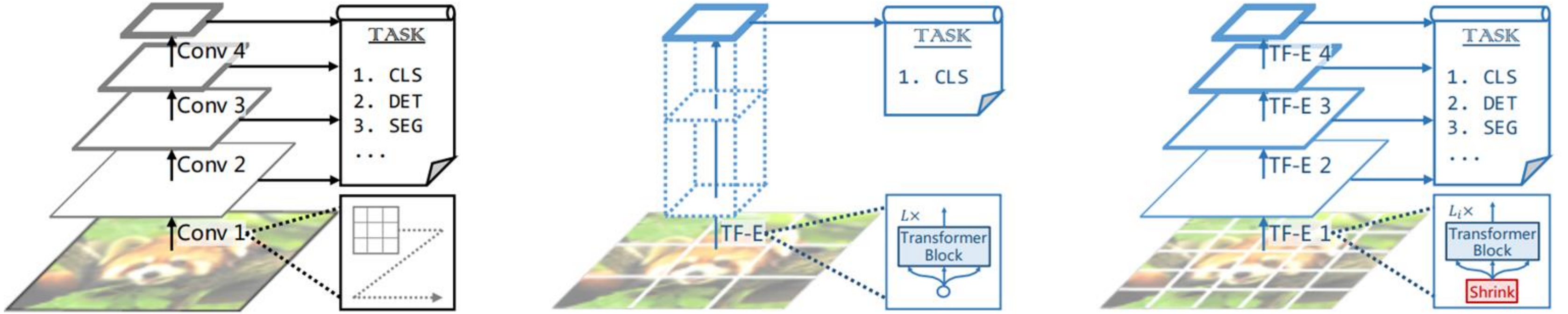
E: embedding

$$
\begin{aligned}
\mathbf{z}_0 &= [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\, \cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, && \mathbf{E} \in \mathbb{R}^{(P^2\cdot C)\times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1)\times D} \\
\mathbf{z}'_\ell &= \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, && \ell = 1\ldots L \\
\mathbf{z}_\ell &= \mathrm{MLP}(\mathrm{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, && \ell = 1\ldots L \\
\mathbf{y} &= \mathrm{LN}(\mathbf{z}_L^0)
\end{aligned}
$$

LN: layer normalization (用所有plane中之所有pixel 相加做正規化)

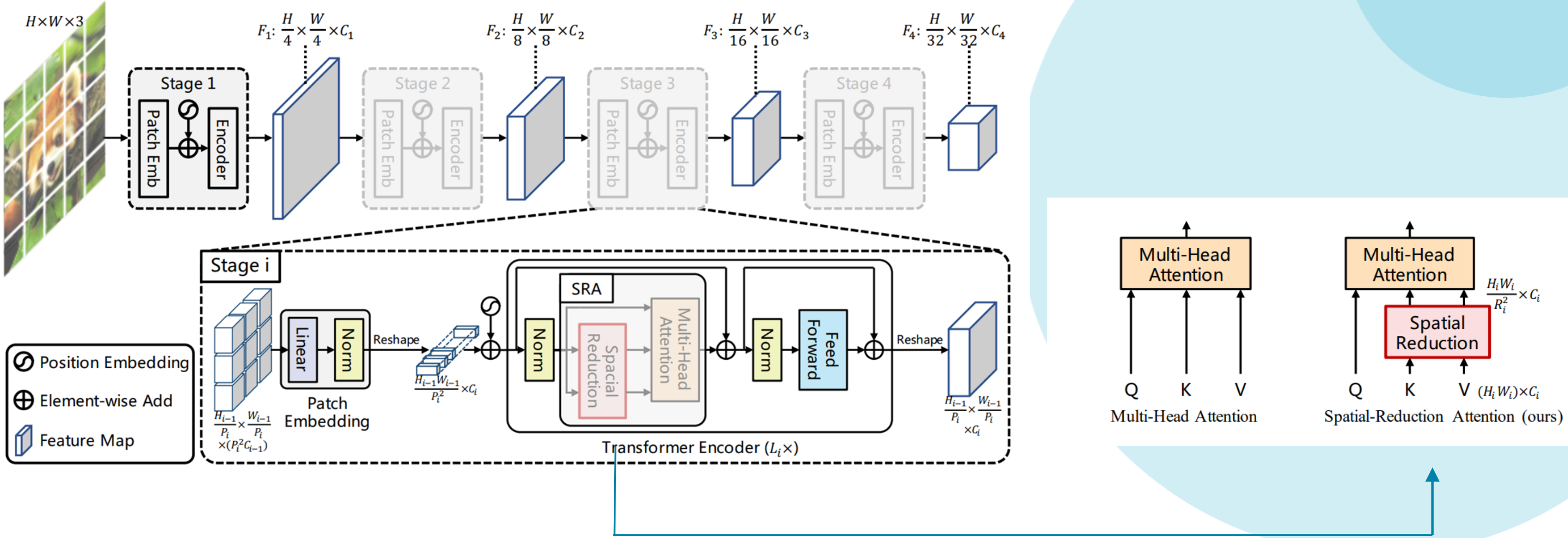
Pyramid Vision Transformer

A pure Transformer model (convolution-free) used to generate multi-scale feature maps for dense prediction tasks



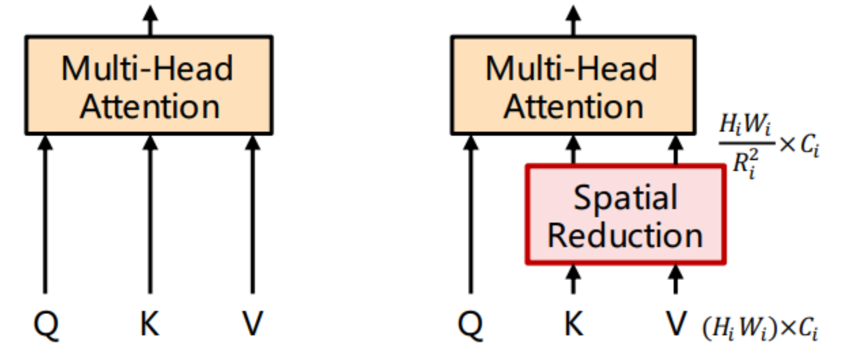
Pyramid CNN | Vision Transformer | Pyramid Vision Transformer

Pyramid Vision Transformer - Main Framework



Pyramid Vision Transformer - Spatial Reduction Attention

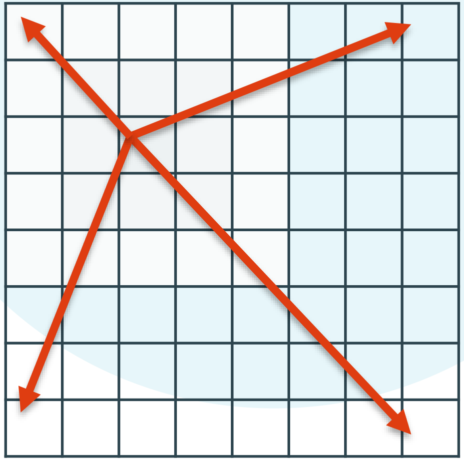
Spatial Reduction Attention (SRA) reduces the dimension of the key (K) and value (V) matrices by a factor of Ri2, where i indicates the stage in the Transformer model.,



Multi-Head Attention   Spatial-Reduction Attention (ours)

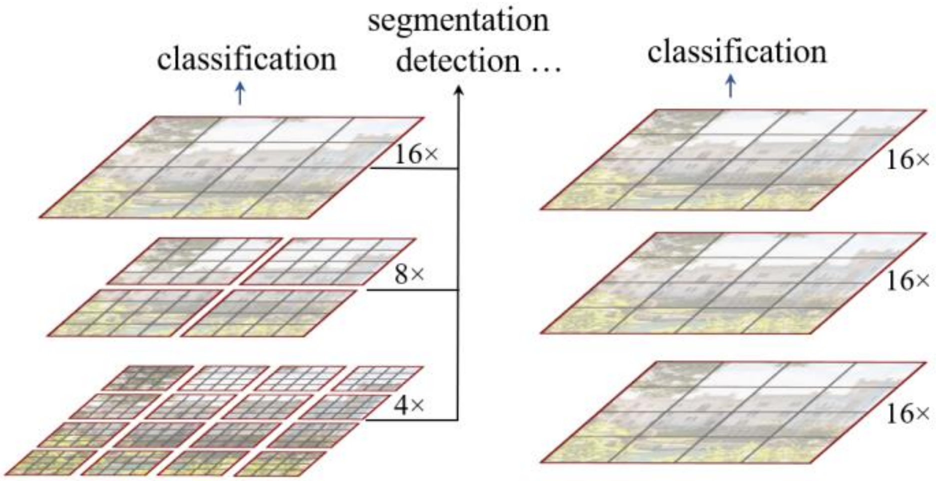
Is Global Attention the Best Solution?

- Global self-attention is EXPENSIVE in images
  - Recall that each element computes attention to ALL elements, $O(n^2)$
- Local regions may be more important at low level in images
  - Recall convolutional network
- Can we reduce computation while still collecting important information?



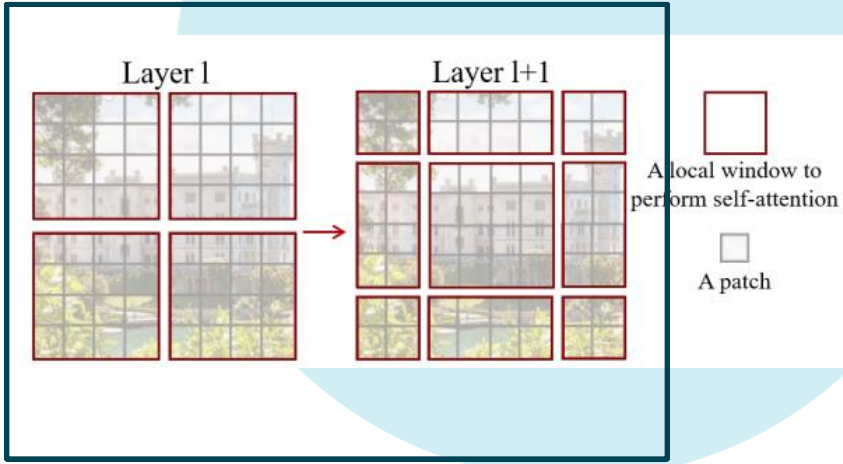
Swin Transformer

Progressively produce feature maps with a smaller resolution while increasing the number of channels in the feature maps.



(a) Swin Transformer (ours)

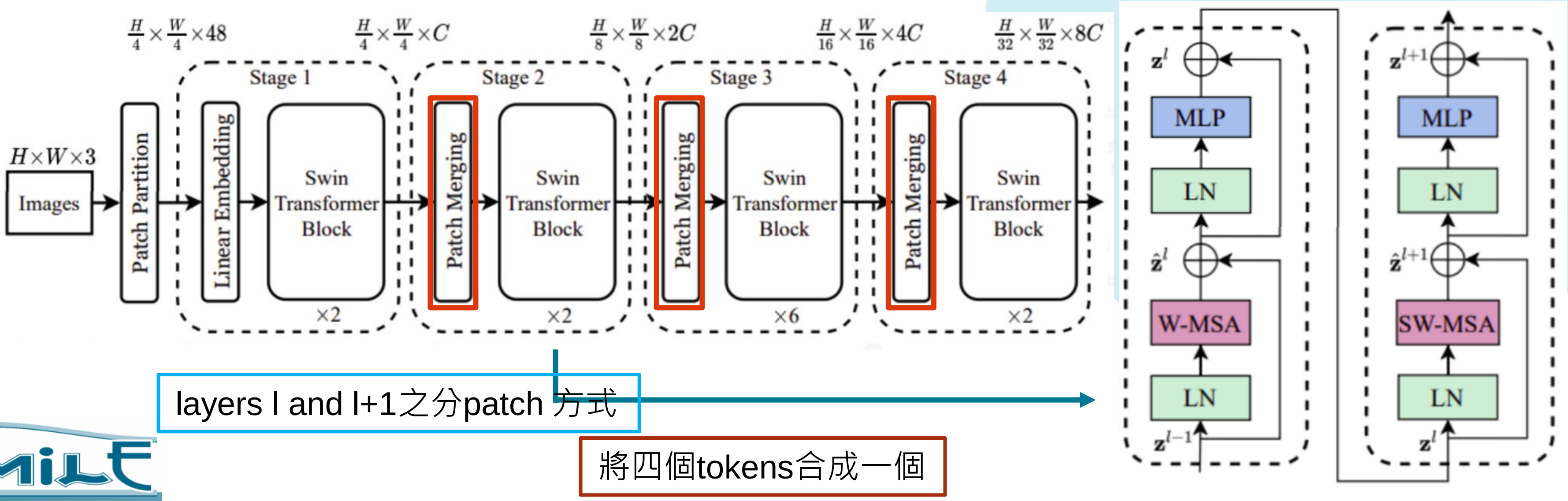
(b) ViT



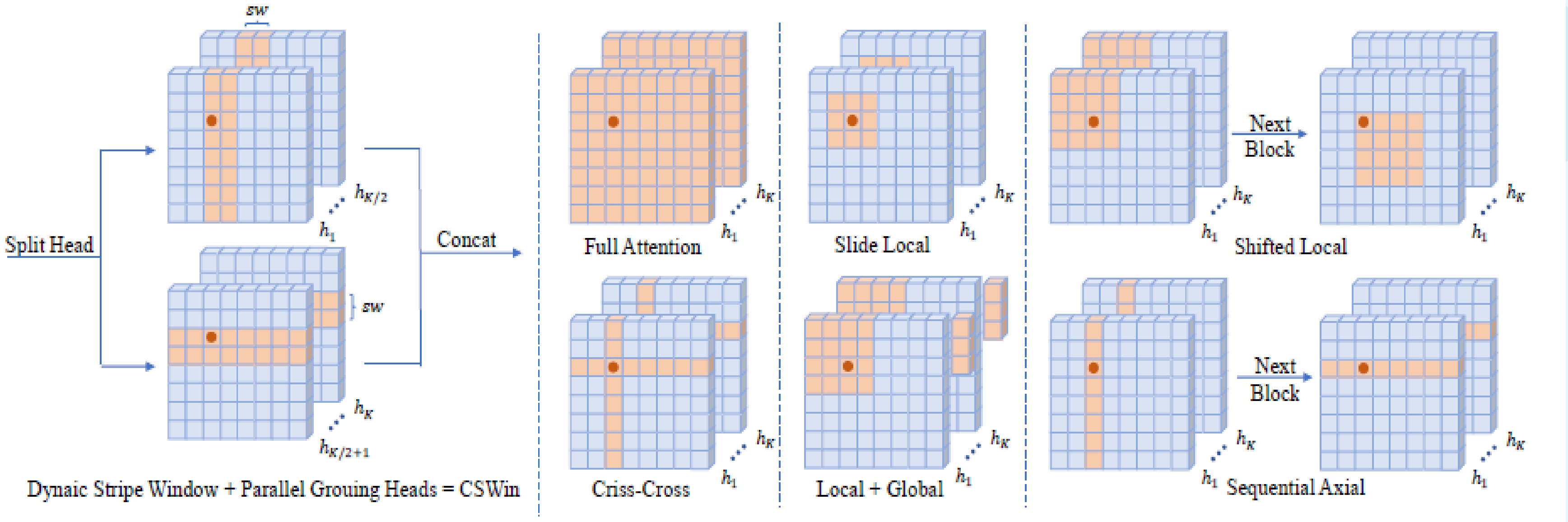
只和同window 內做attention，

Swin Transformer

smaller patch size of 4 x 4 pixels (16 x 16 in Vit)
use patch merging to changes the resolution at the beginning of each stage
alternating between W-MSA and SW-MSA in two consecutive layers,
allowing the information to propagate to a larger area when going deeper

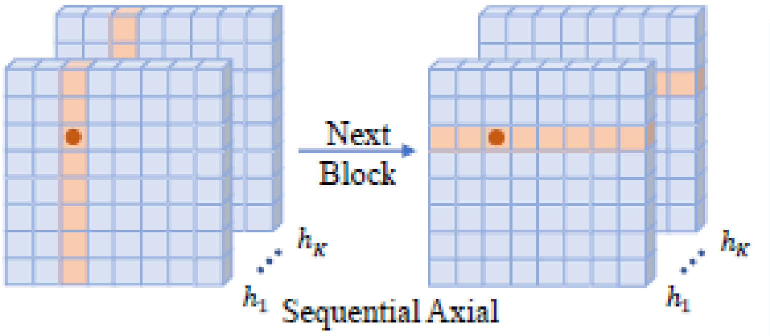


CSWin Transformer [4]

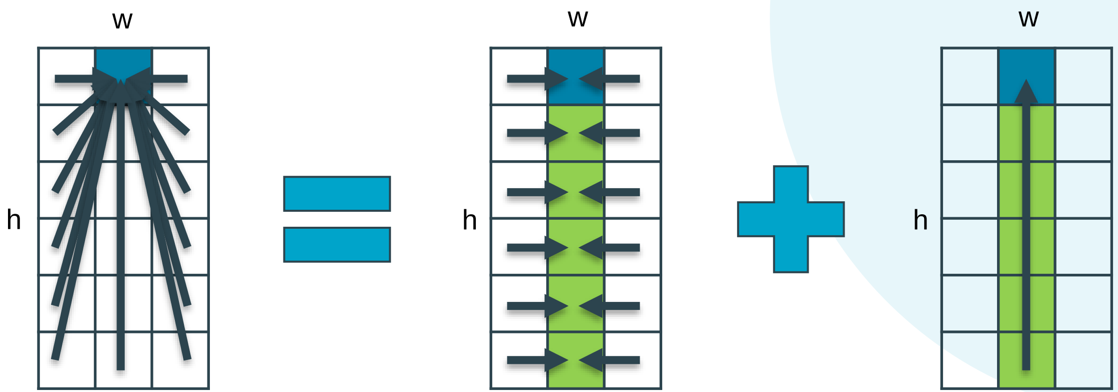


- Vision of local window attention is limited and need more blocks to achieve global receptive field
- CSWin Transformer uses Cross-Shaped Window Self-Attention to efficiently pass information globally

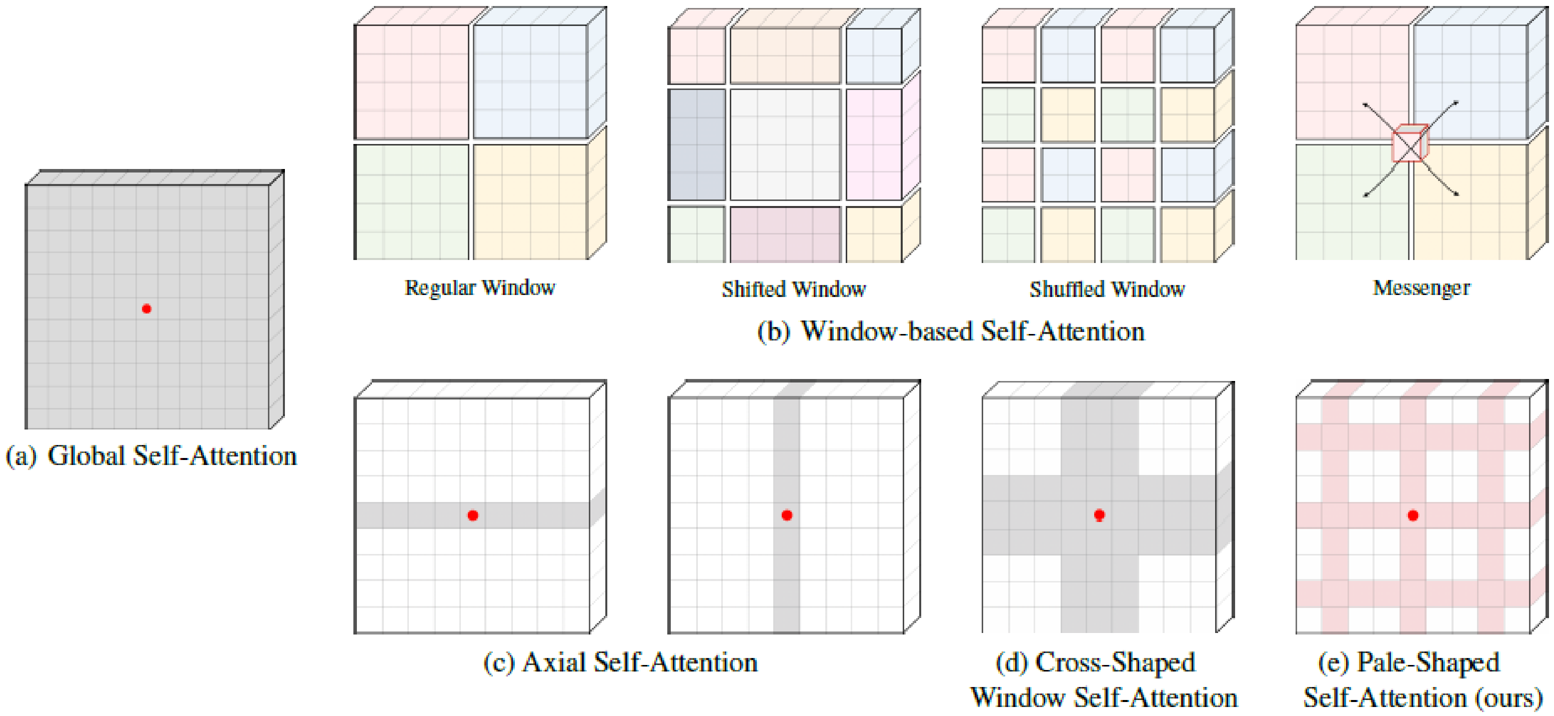
Axial Attention



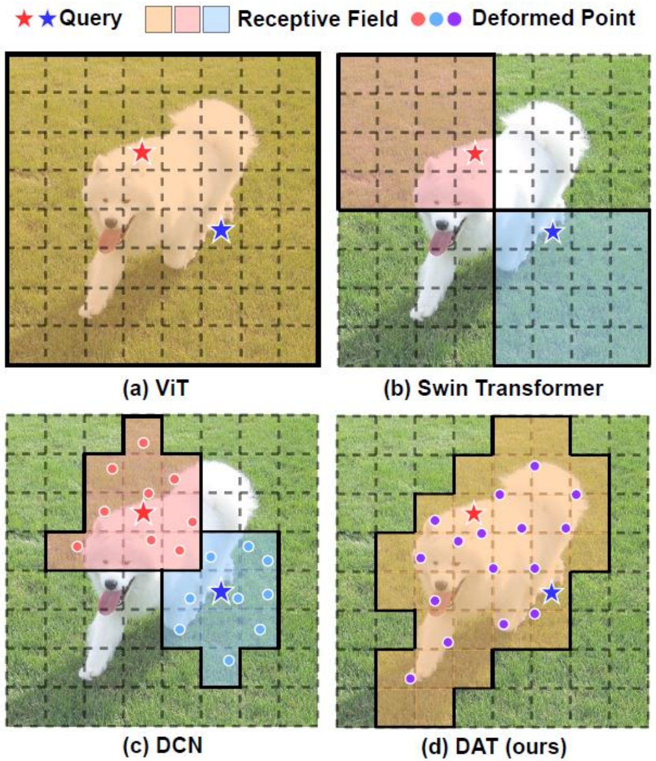
[4]

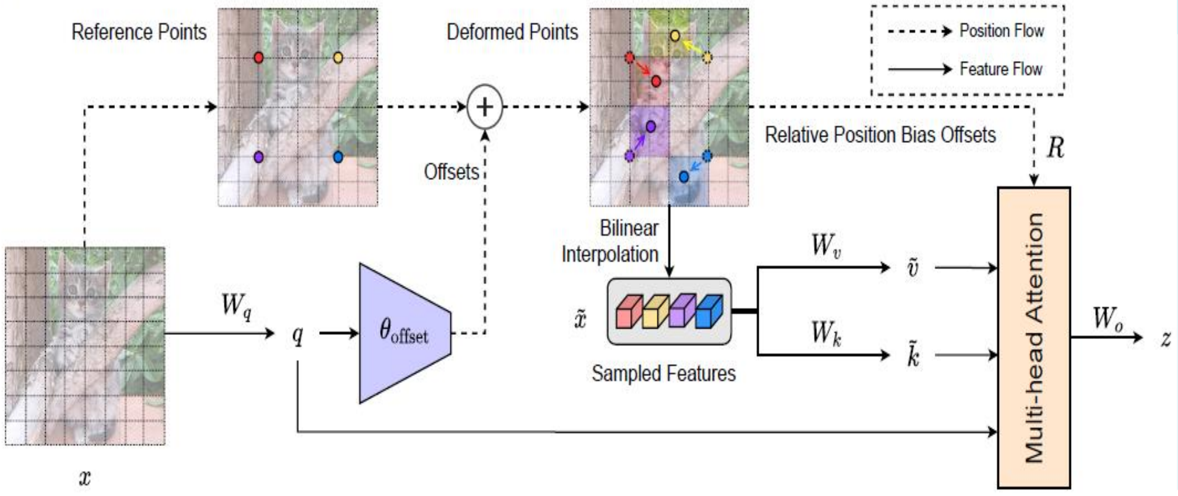


Pale Transformer [5]



Deformable Attention [6]





(a) Deformable attention module

DeiT
(Data-efficient image transformers)

- Training Data-Efficient Image Transformers & Distillation Through Attention, DeiT, Data-Efficient Image Transformers, by Facebook AI, Sorbonne University (2021, ICML)

1. No hundreds of millions of images pre-trained required.
2. More efficient than ViT.
3. A teacher-student strategy is introduced with distillation token.

Distillation Through Attention

What's difference between DeiT and ViT?

- DeiT has the same architecture as ViT except the input token part that having an additional distillation token.

Why Distillation?

- When trained on insufficient amounts of data, distillation is a way to help training.
- Using a distillation token ensuring that the student learns from the teacher through attention.

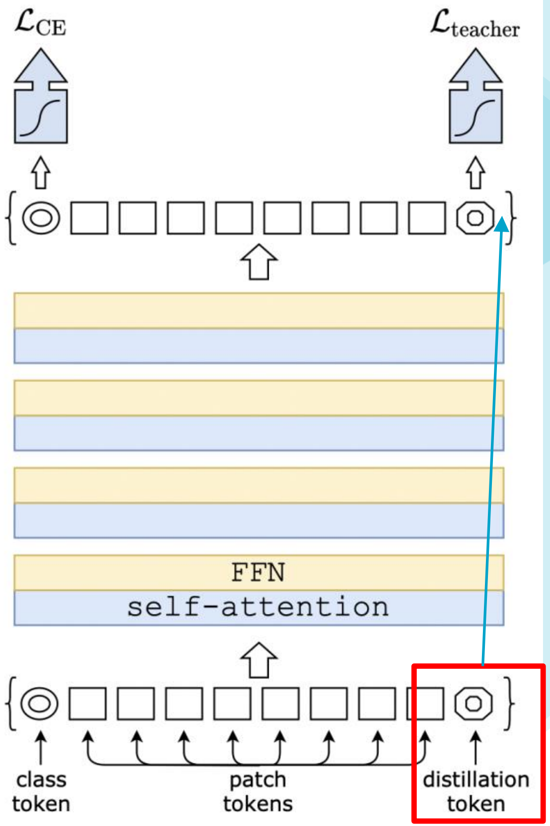
Distillation Token

What is Distillation Token?

- A new distillation token is included.
  It interacts with the class and patch tokens through the self-attention layers.

How to implement?

- Employed in a similar fashion as the class token, except that on output of the network its objective is to reproduce the hard label predicted by the teacher, instead of true label.



經過teacher classifier，結果用於student做為學習之label

Soft label distillation

- Student model has similar output distribution as the teacher model.

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{\text{CE}}(\psi(Z_{\text{s}}), y) + \boxed{\lambda\tau^2 \text{KL}(\psi(Z_{\text{s}}/\tau), \psi(Z_{\text{t}}/\tau))}$$

Hard label distillation

- Output of the teacher model is used as true label.

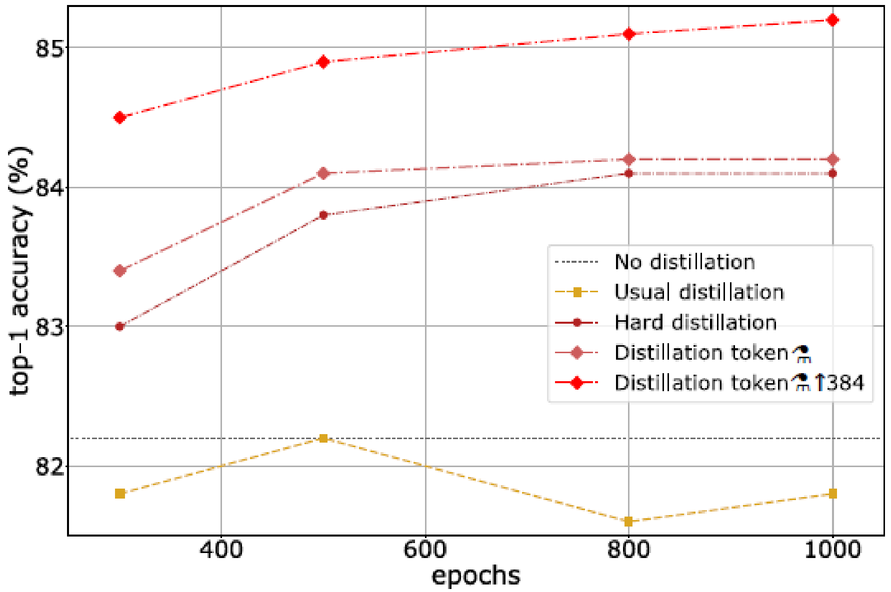
$$\mathcal{L}_{\text{global}}^{\text{hardDistill}} = \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \boxed{\frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y_{\text{t}}).}$$

Z_s = Output of the student model
y_t = Output of the teacher model
y = Ground Truth

Hard Label Distillation: Calculate cross entropy between y_t and Z_s

Experiment

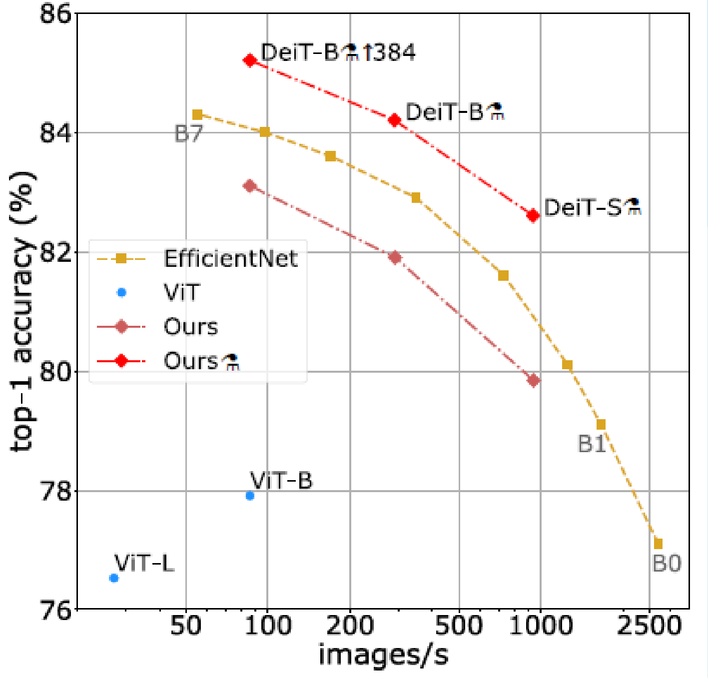
- With higher Precision but



With proposed distillation token
perform better than with other distillation methods

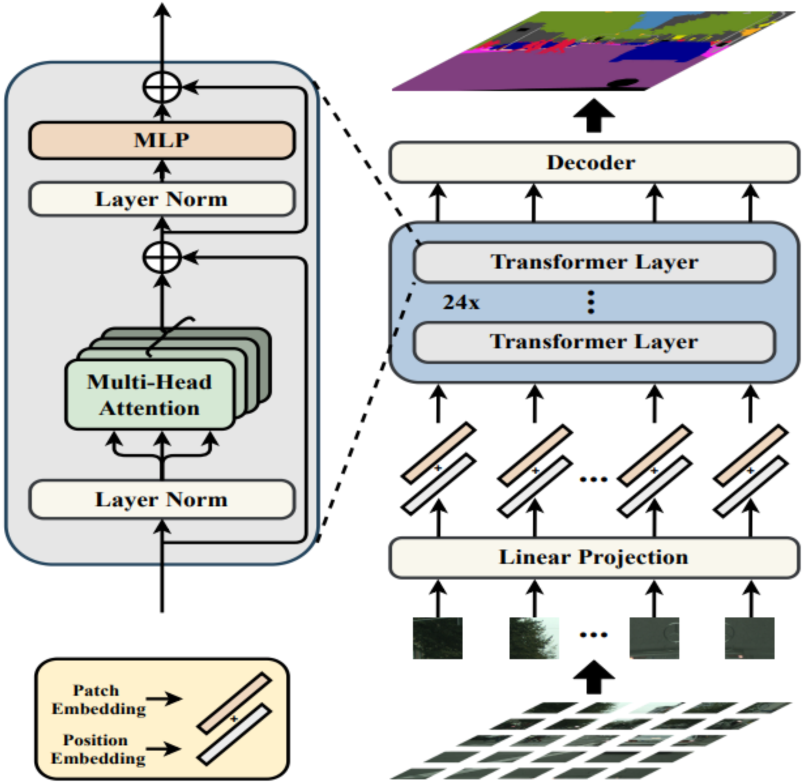
↑384 With Input Image size – 384 x 384

⚗ With Hard Label Distillation



With proposed distillation token
perform better than others.

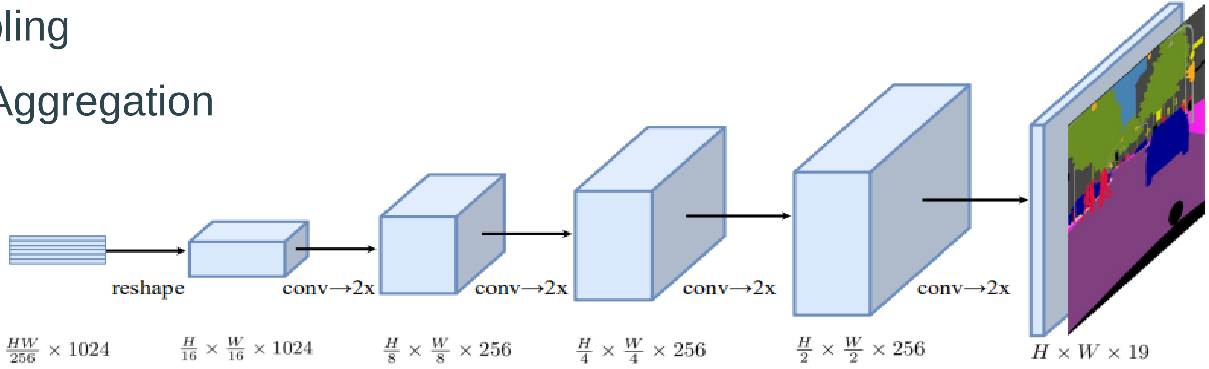
SEgmentation TRansformer (SETR)

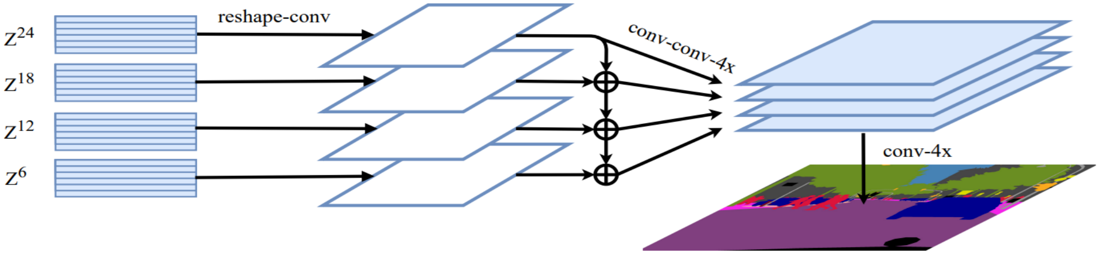


Ref. : Sixiao Zheng et al., 2021

SETR - Decoder

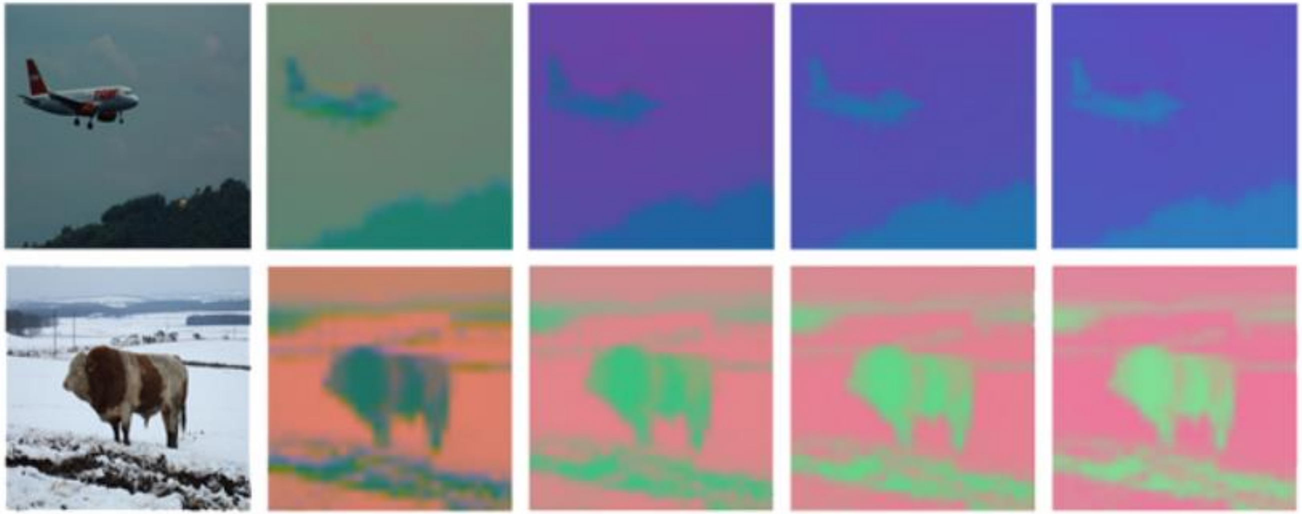
- Navie upsampling
  - 1x1 Conv + Batch Normalization + 1x1 Conv
- Progressive Upsampling
- Multi-Level Feature Aggregation





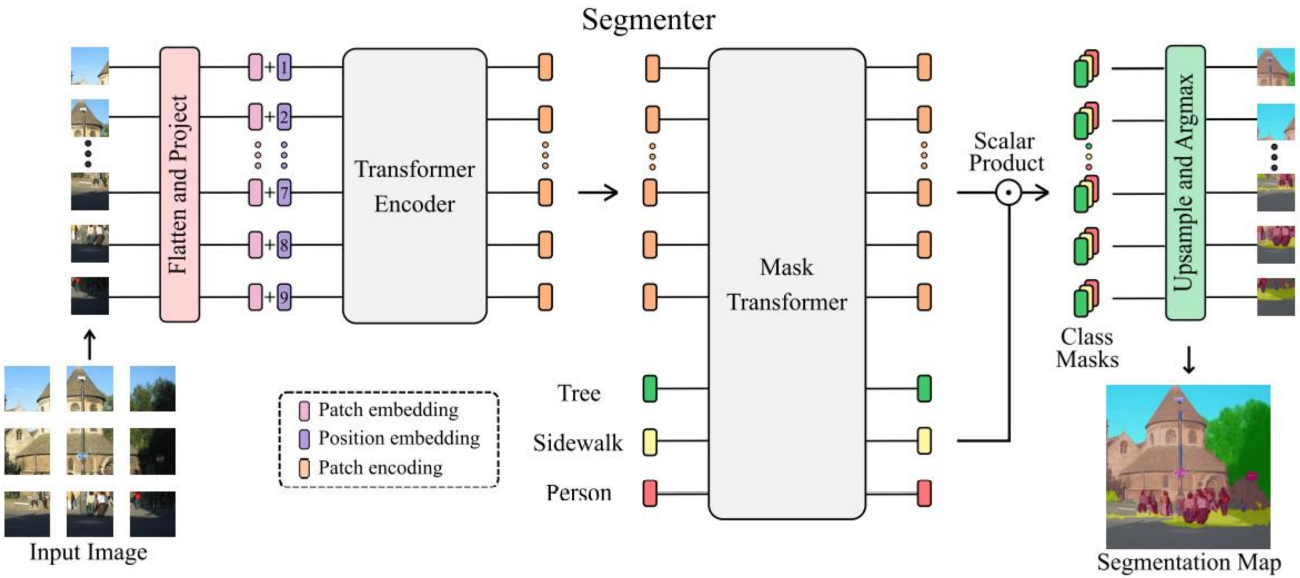
SETR - Result

- Visualization of the output feature of layer Z^1, Z^9, Z^{17}, Z^{24}

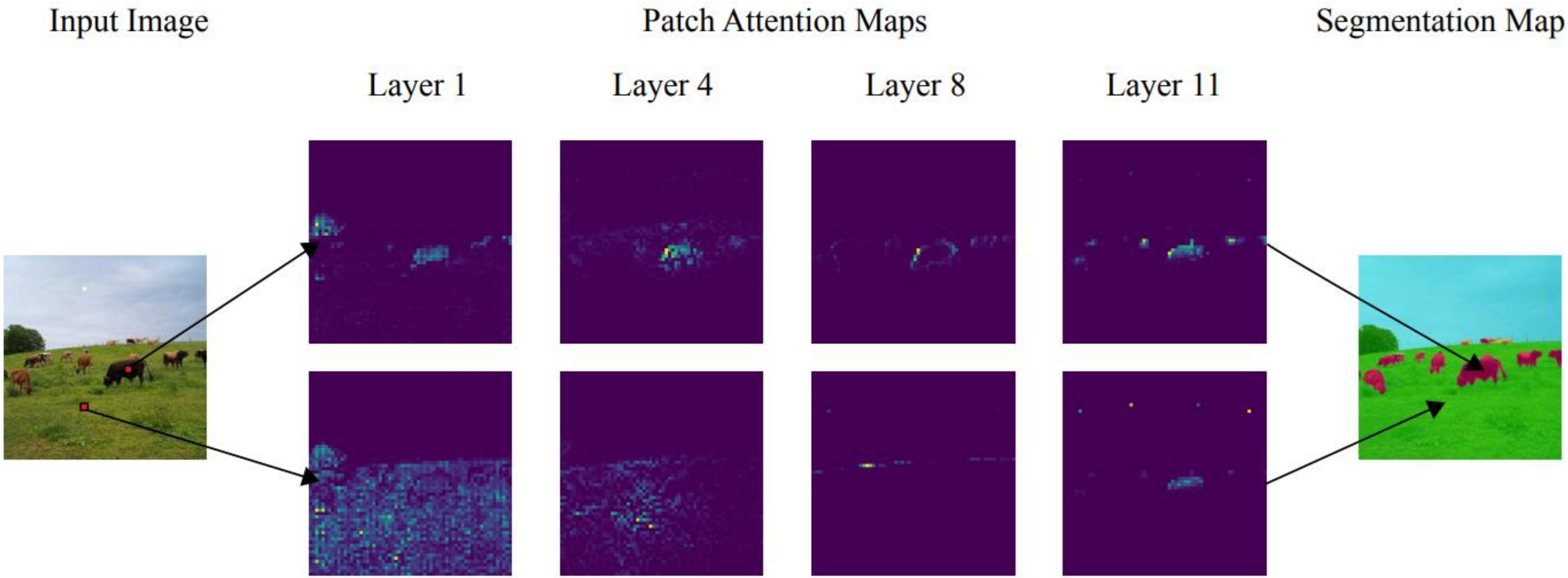


Segmentor

Pure transformer-based segmentation model with an encoder as used in ViT, and add **learnable class embeddings** and **transformer-based decoder** for generating class masks

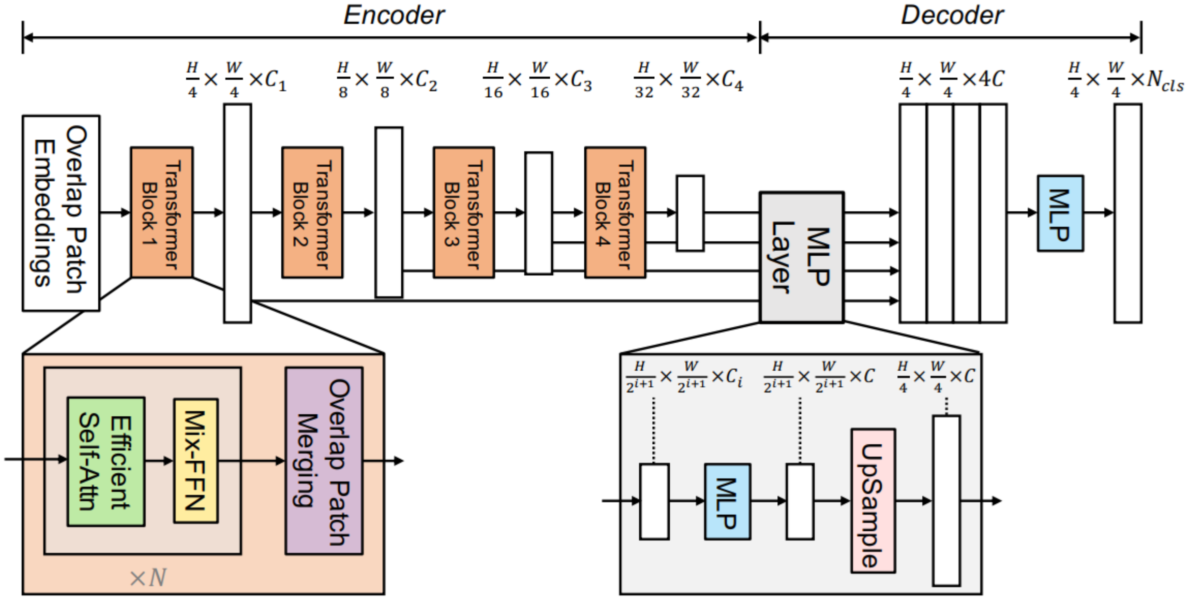


Segmentor - attention map results



SegFormer

- Unified transformers with lightweight MLP decoders



Ref. : Enze Xie al., 2021