Transformers for Vision

Transformer

- The de facto neural architecture of many applications
  - GPT: generative pre-training Transformer
  - Gemini: multi-modal LLMs
  - Computer vision
  - Speech recognition

ChatGPT   Gemini   Claude   Llama

Attention

- Attention compute a weighted average of hidden state vectors.
- Formally:
  - Given N k-dimensional input vector $x^1 \ldots x^N$and output vector $y^1 \ldots y^N$
  - Output y^i is a weighted average of input vector; $\boldsymbol{w}_{ij}$ is the attention weight from position i over x^j
    - $y^i = \sum_{j=1}^{N} \boldsymbol{w}_{ij} x^j$

Formulation of Self-attention

- Given N k-dimensional input vector $x^1 \dots x^N$ and output vector $y^1 \dots y^N$
- Conventional projection layer:
  - Output y^i is a weighted average of input vector
    - $y^i = \sum_{j=1}^{N} \boldsymbol{w}_{ij} x^j$
- Self-attention: self-dependent
  - The weight w_{ij} depends on input **itself** x^j
    - $w'_{ij} = \sum_k x_k^i x_k^j$
  - Followed by Softmax normalization: $w_{ij} = \frac{\exp(w'_{ij})}{\sum_j \exp(w'_{ij})}$

Self-attention with Query, Key, Value

- Self-attention: The weight w_{ij} depends on input itself
  - $w'_{ij} = \sum_k x_k^i x_k^j$
  - the input x^i is to matched the input x^j
- More flexible version: add learnable parameters
  - Linear projection layer with learnable weight matrix $W \in \mathbb{R}^{k \times k}$
  - Query: $Q^i = W_q x^i$, to match others
  - Key: $K^i = W_k x^i$, to be matched
  - Value: $V^i = W_v x^i$

Self-attention with Query, Key, Value

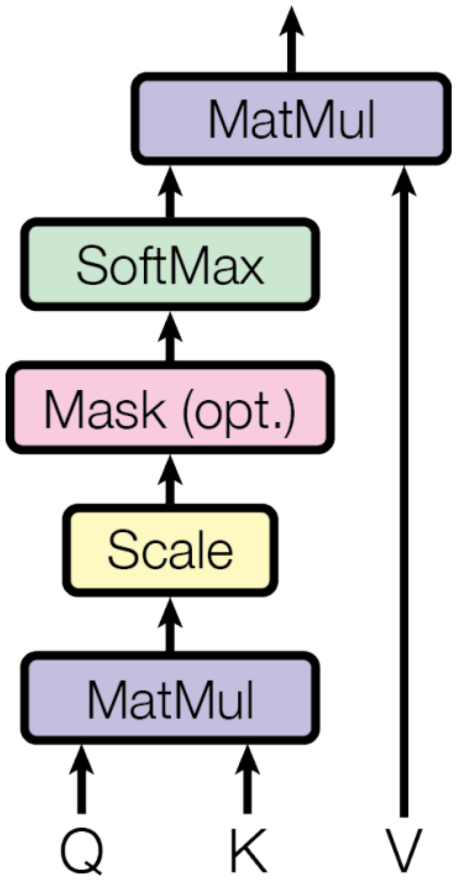
- Q, K, V are linearly projected
  - Query: $Q^i = W_q x^i$
  - Key: $K^i = W_k x^i$
  - Value: $V^i = W_v x^i$
- Compute attention weight based on Q^i and K
  - $A_{ij} = \frac{\exp(Q^i K^i)}{\sum_j \exp(Q^i K^j)}$

Self-attention with Query, Key, Value

- Compute attention weight based on Q^i and K
  - $A_{ij} = \frac{\exp(Q^i K^i)}{\sum_j \exp(Q^i K^j)}$
- Aggregated with value vector V
  - The new output for i-th position depends on the attention weights and value vectors of all input positions j
  - $y^i = \sum_{j=1}^{T} A_j^i V^j$

Scaled Dot-product Attention

- Scale factor $\sqrt{d}$ normalizes the dot product values QK^T, preventing their variance from becoming overly large.
- Scale dot-product: $A = Softmax(\frac{QK^T}{\sqrt{d}})$
- Matrix Multiplication: $y = AV$



[1] Vaswani A, et al. Attention is all you need[J]. NeurIPS, 2017

$$scaled\ dot-product:$$

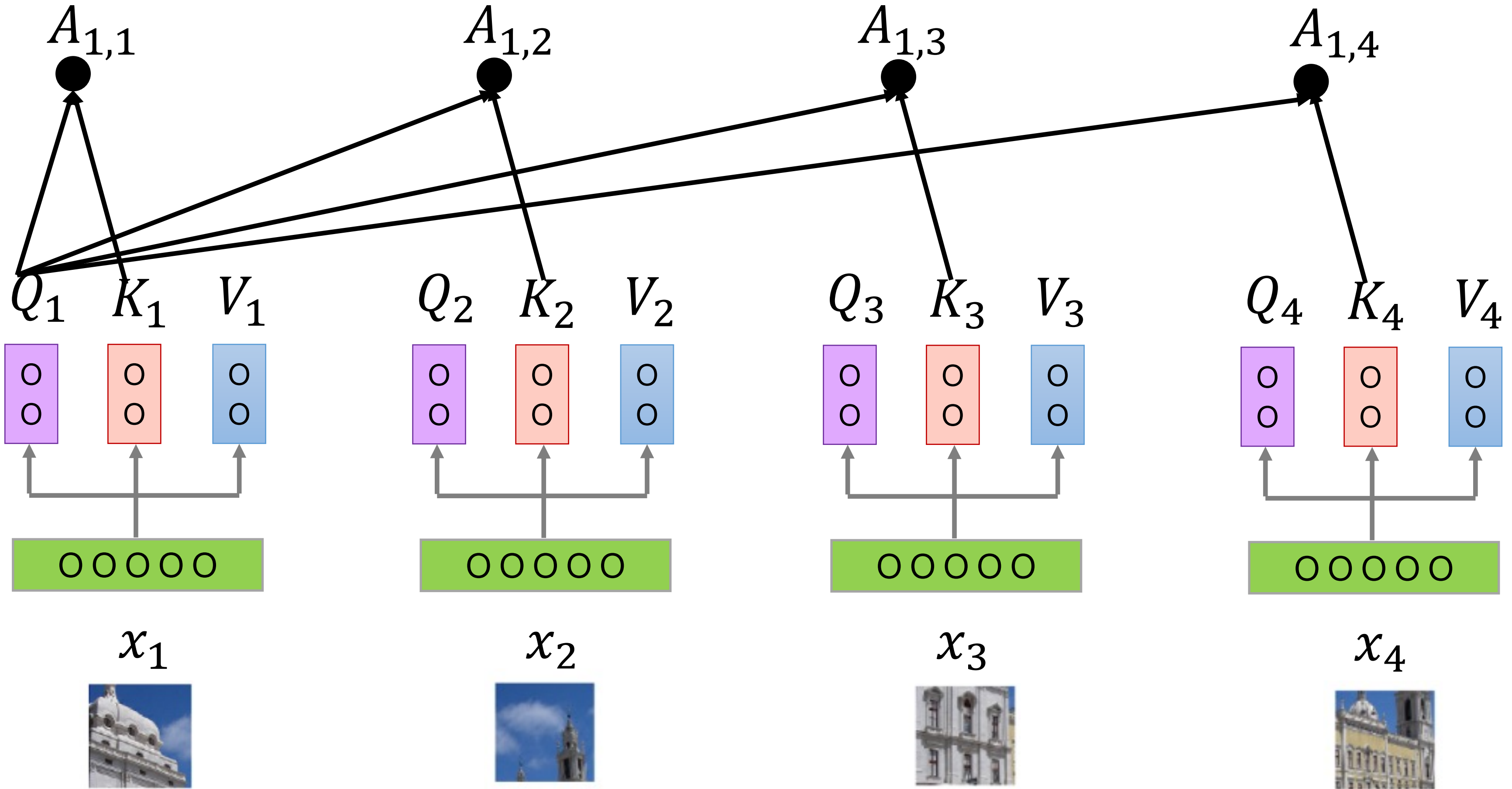
$$A_{1,i} = \frac{QK^T}{\sqrt{d}}$$

Q: query (to match others)

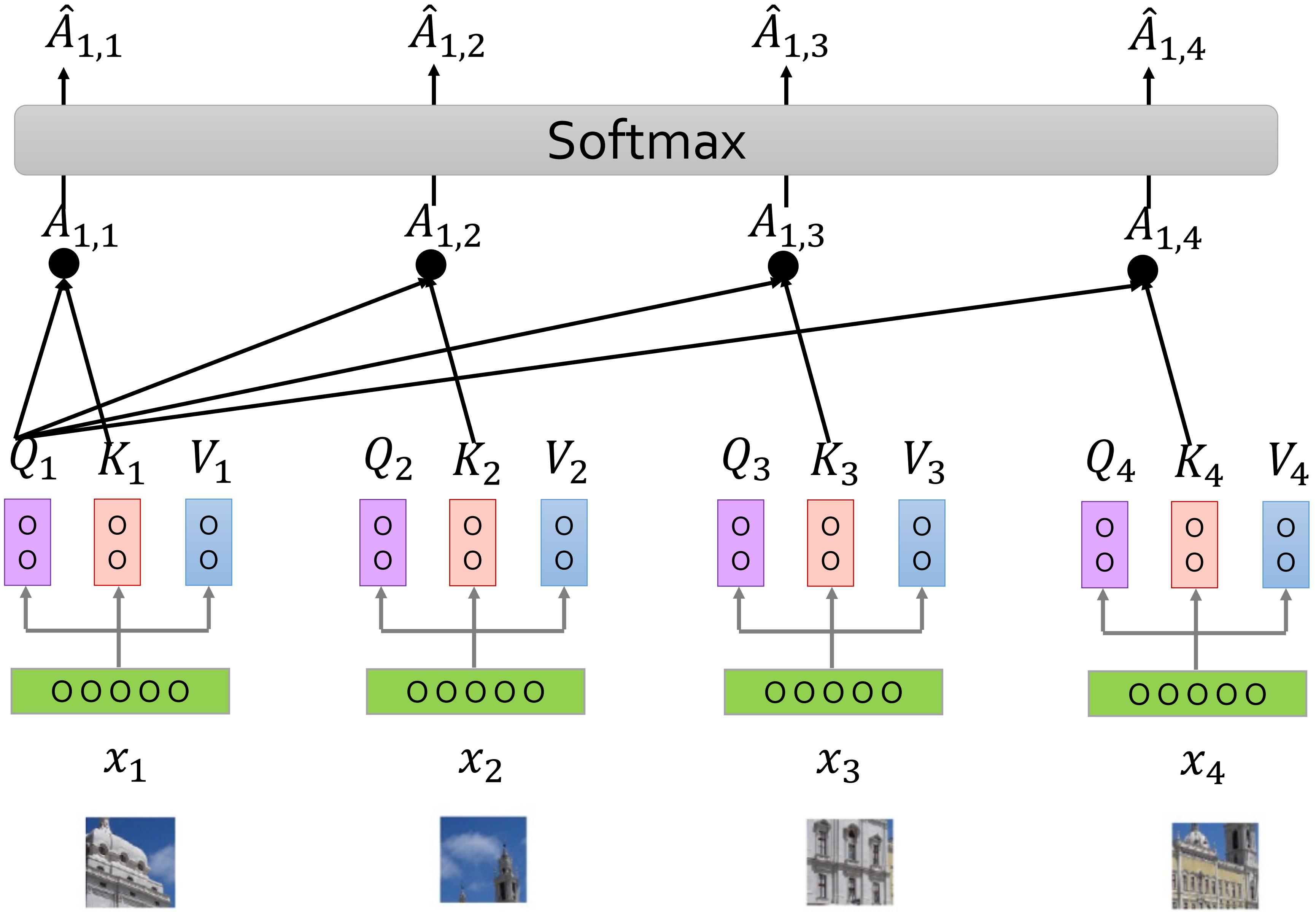
K: key (to be matched)

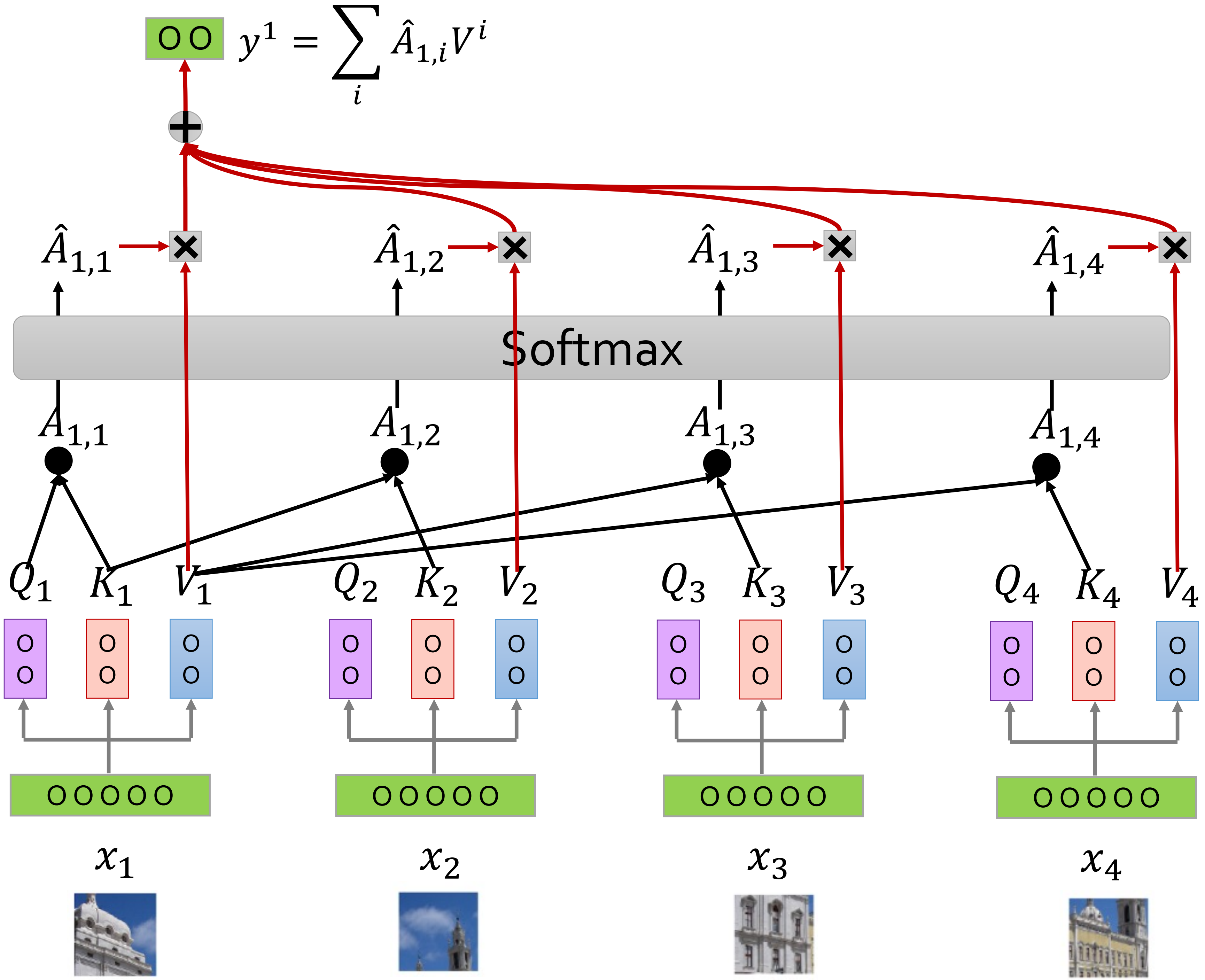
V: value

x: input vector before projection



$$\sigma(z)_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$




$y^1 = \sum_i \hat{A}_{1,i} V^i$
$\hat{A}_{1,1}$
$\hat{A}_{1,2}$
$\hat{A}_{1,3}$
$\hat{A}_{1,4}$
Softmax
$A_{1,1}$
$A_{1,2}$
$A_{1,3}$
$A_{1,4}$
Q_1 K_1 V_1
Q_2 K_2 V_2
Q_3 K_3 V_3
Q_4 K_4 V_4
x_1
x_2
x_3
x_4

Quadratic complexity

- Attention(Q, K, V) = $Softmax\left(\frac{QK^T}{\sqrt{d}}\right)V$
- Complexity: $O(n^2 \cdot d)$
- *n* = sequence length, *d* = hidden dimension

Cross-attention

- Given two different input vector x and m
- Input to Cross-attention
  - Query: $Q^i = W_q x^i$
  - Key: $K^i = W_k m^i$
  - Value: $V^i = W_v m^i$
- Source vector x is attended to target vector m
- Compute cross-attention weight based on Q^i and K
  - $A_{ij} = \frac{\exp(Q^i K^i)}{\sum_j \exp(Q^i K^j)}$

Rethink Cross-attention *as* Feature Clustering

- (not exactly the same)
- input vector $x \in \mathbb{R}^{N \times d}$ ($Q^i = W_q x^i$) is the feature centroids with N centroids
- input vector $m \in \mathbb{R}^{HW \times d}$ ($K^j = W_k m^j$, $V^i = W_v m^i$) is the pixel features
- Cross-attention weight *v.s.* Soft assignment based on feature distance
  - $A_{ij} = \frac{\exp(Q^i K^j)}{\sum_l \exp(Q^i K^l)}$
- Value aggregation *v.s.* Centroid update:
  - $O^i = \sum_{j=1}^{M} A_j^i V^j$

Positional encoding

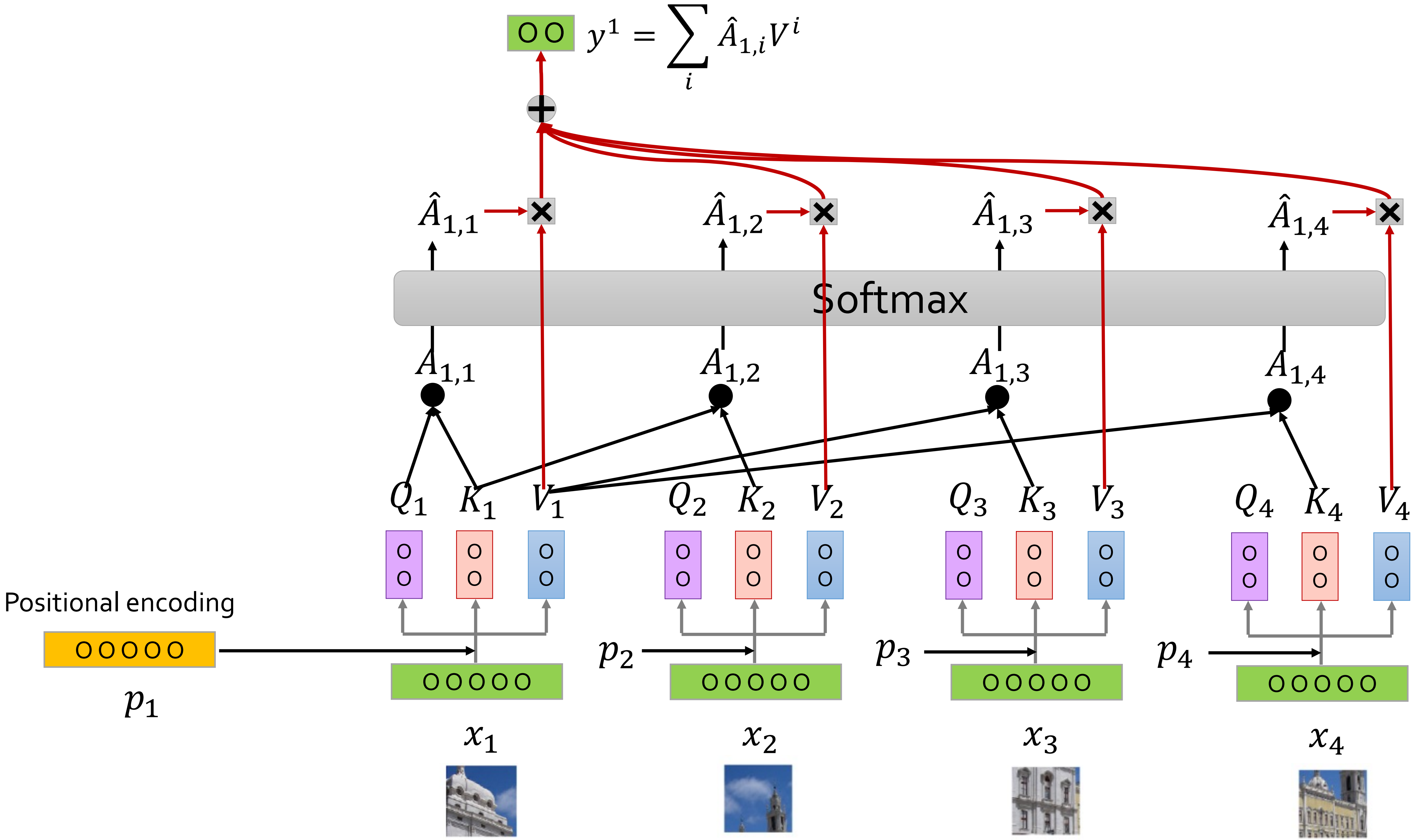
- Motivation:
  - While the attention mechanism is powerful, it doesn't inherently capture the **order of elements** within the sequence.
  - In language, the order of words is crucial for meaning. For example, "The cat sat on the mat" has a different meaning than "The mat sat on the cat.”
  - Similarly, in computer vision, the permutation of image pixels matters.

Positional encoding

$$\bar{x}_t = \left[\begin{array}{c} x_t \\ t \end{array} \right]$$

- append t to the input
  - This is not a great idea, because absolute position is less important than relative position
  - we want to represent position in a way that tokens with similar relative position have similar positional encoding

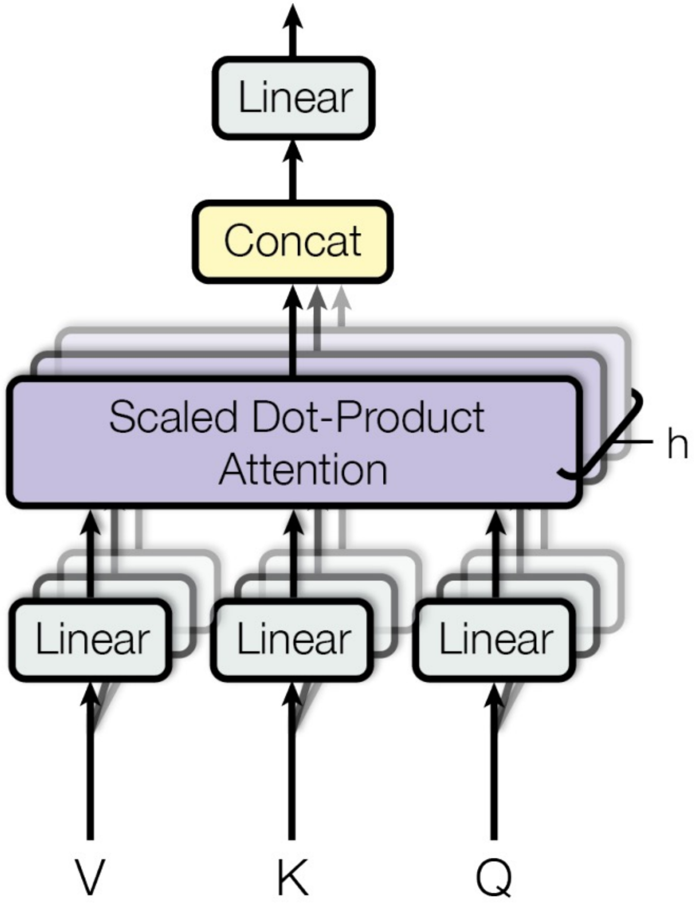
- More advanced positional encoding (e.g., frequency-based, learning-based, relative positional encoding) are not discussed in this lecture.



$y^1 = \sum_i \hat{A}_{1,i} V^i$
$\hat{A}_{1,1}$ $\hat{A}_{1,2}$ $\hat{A}_{1,3}$ $\hat{A}_{1,4}$
Softmax
$A_{1,1}$ $A_{1,2}$ $A_{1,3}$ $A_{1,4}$
Q_1 K_1 V_1 Q_2 K_2 V_2 Q_3 K_3 V_3 Q_4 K_4 V_4
Positional encoding
p_1 p_2 p_3 p_4
x_1 x_2 x_3 x_4

Multi-head self-attention

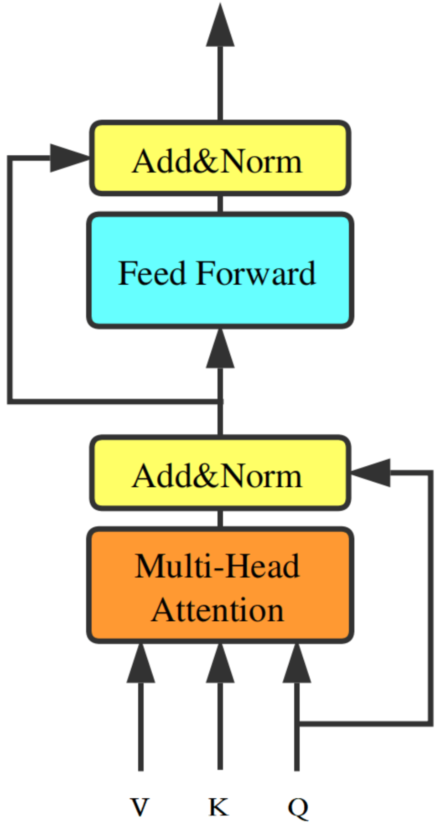
- Run **h** attention models in parallel on top of different linearly projected versions of Q, K, V; concatenate and linearly project the results
- Intuition: enables model to attend to different kinds of information at different positions



[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

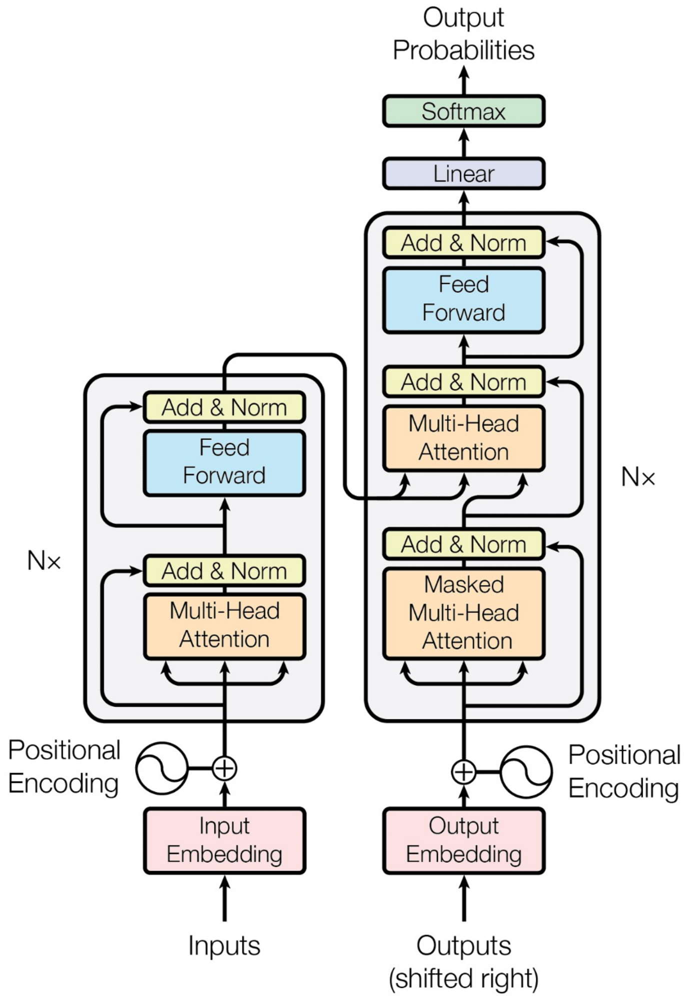
Transformer block

- A Transformer is a sequence of transformer blocks
  - Vaswani et al.:
    - 12 blocks, 512 embedding dimension, 6 attention heads
  - **Multi-Head Attention**: introduced before
  - **Add & Norm**: residual connection followed by layer normalization
  - **Feedforward (Multi-layer perceptron)**: two linear layers with ReLUs in between, applied independently to each vector
- Attention is the only interaction between inputs



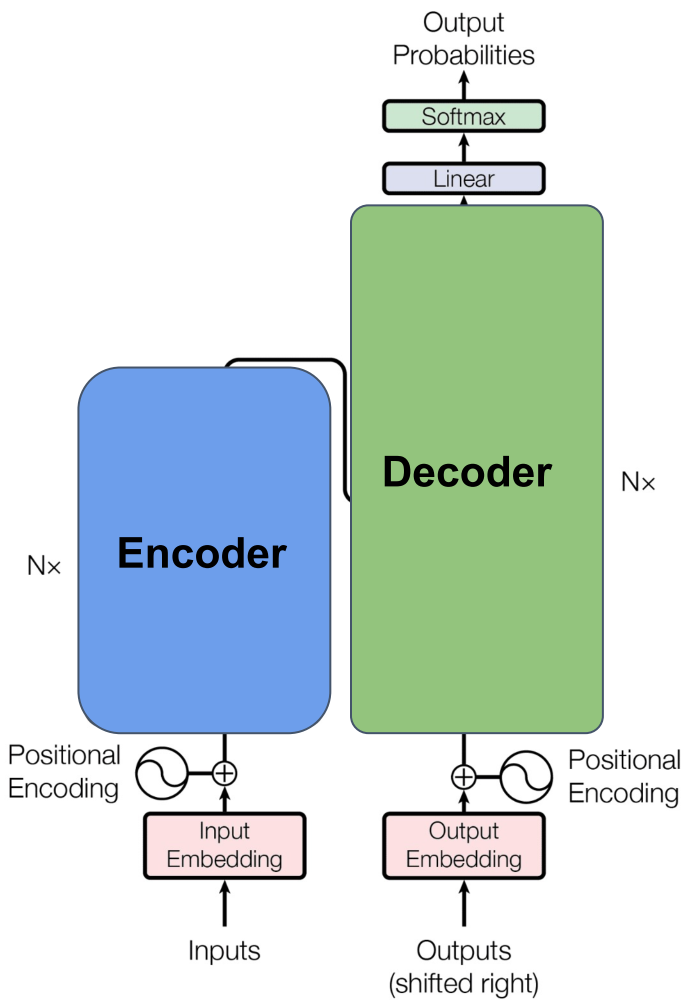
[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers



[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers

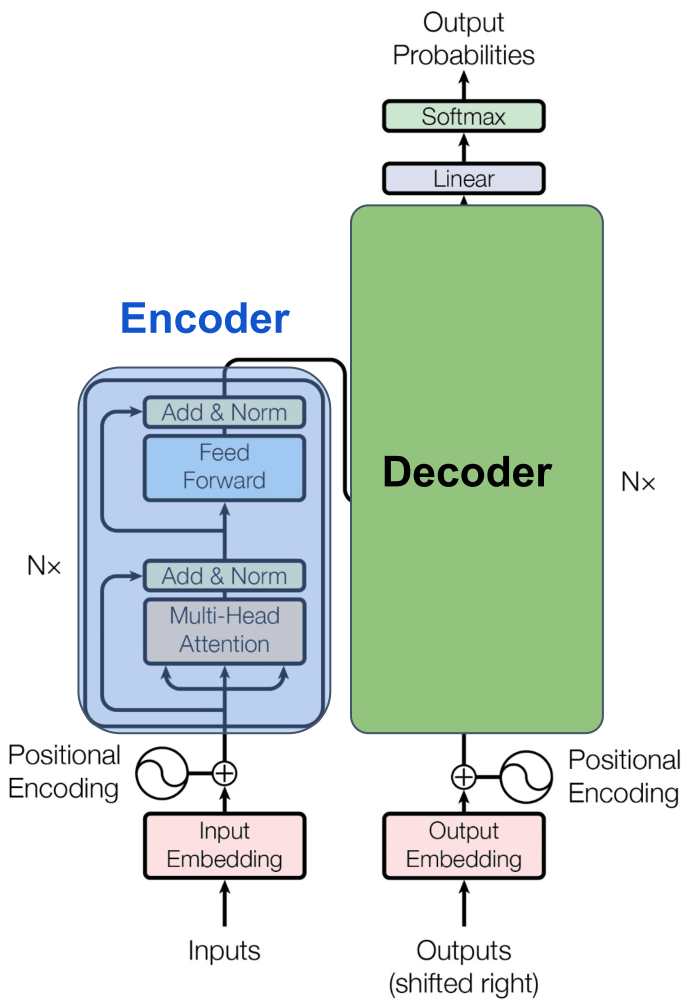


[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers

Transformer Encoder:

- ❑ Self-attention
- ❑ Serves as strong feature extracting/enhancing modules.

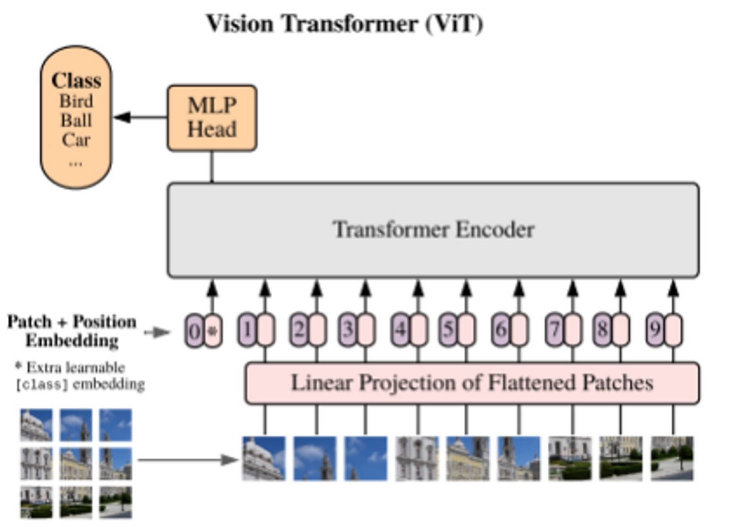


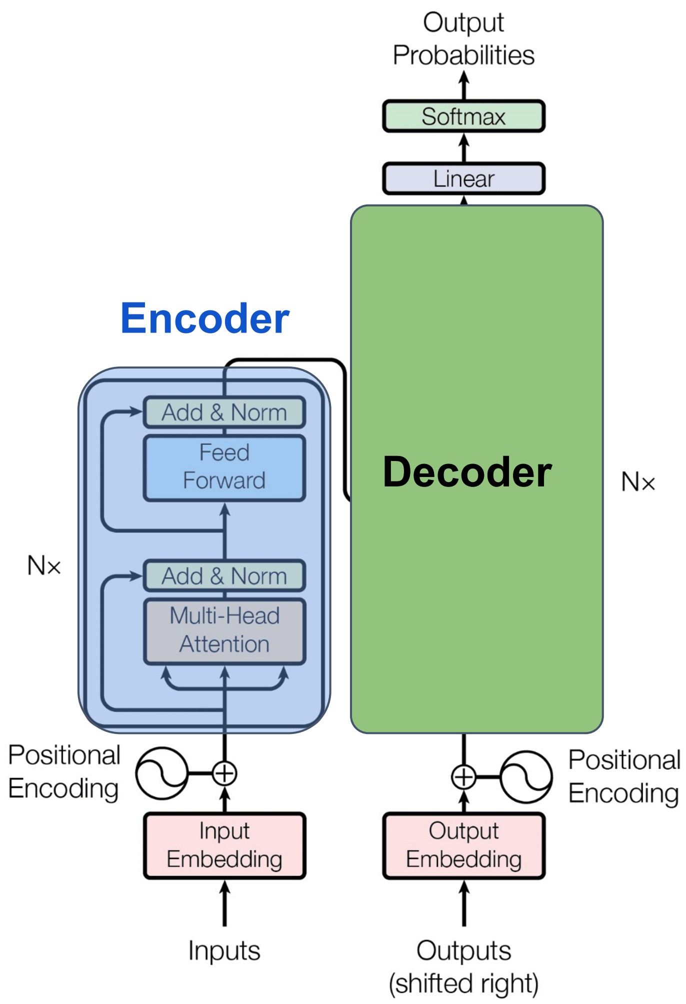
[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers

Transformer Encoder:

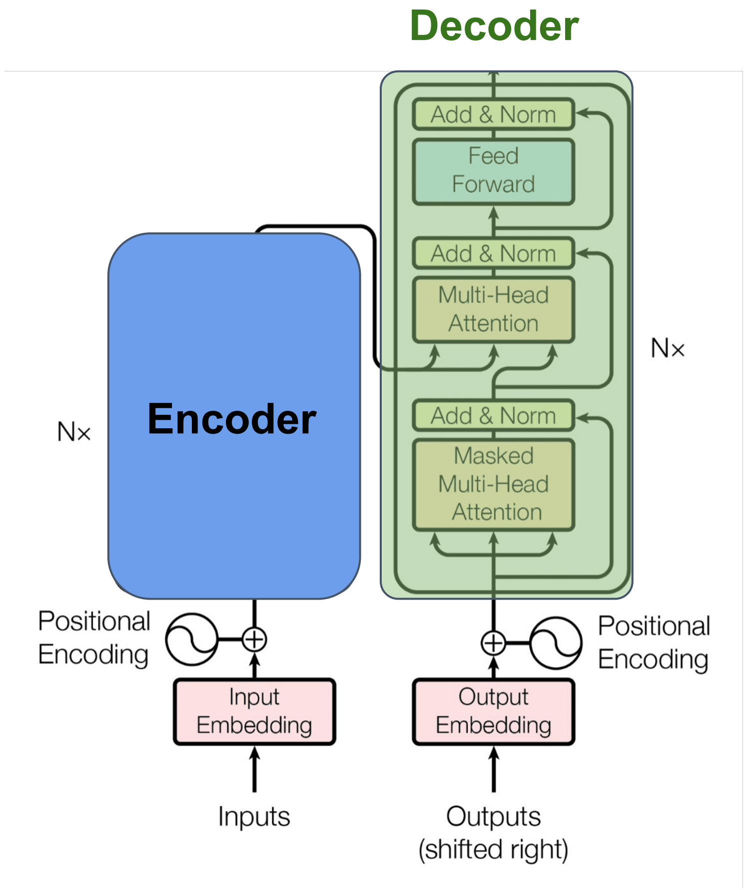
- Vision Transformer (ViT)





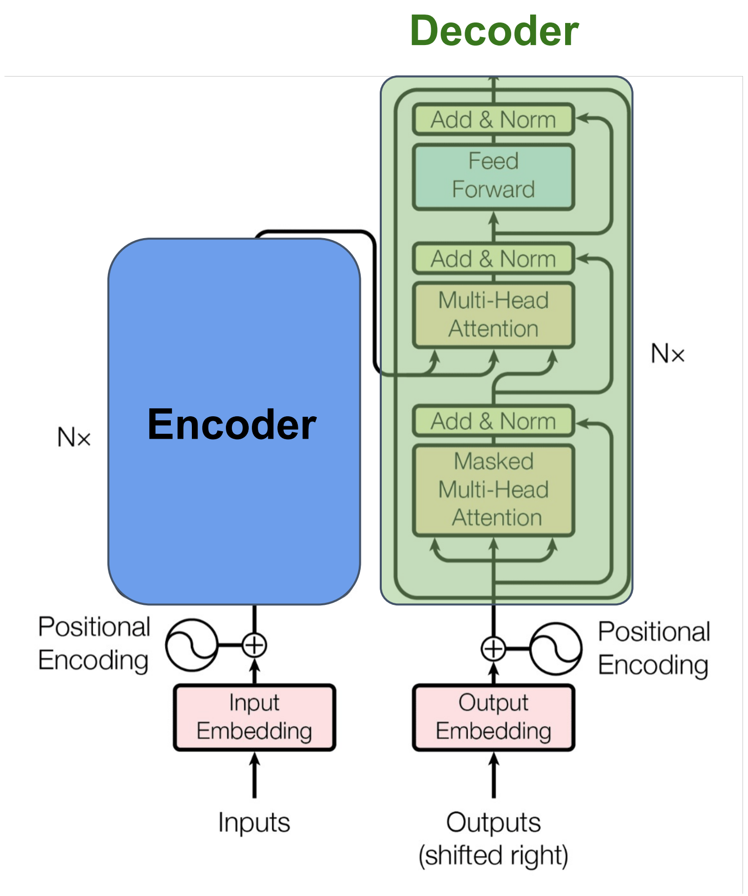
[1] Attention Is All You Need. NeurIPS 2017.
[2] ViT: Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. ICLR 2021.

Transformers



[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers



Transformer Decoder:

- Cross-attention
- Serves as a bridge for sequence-to-sequence translation

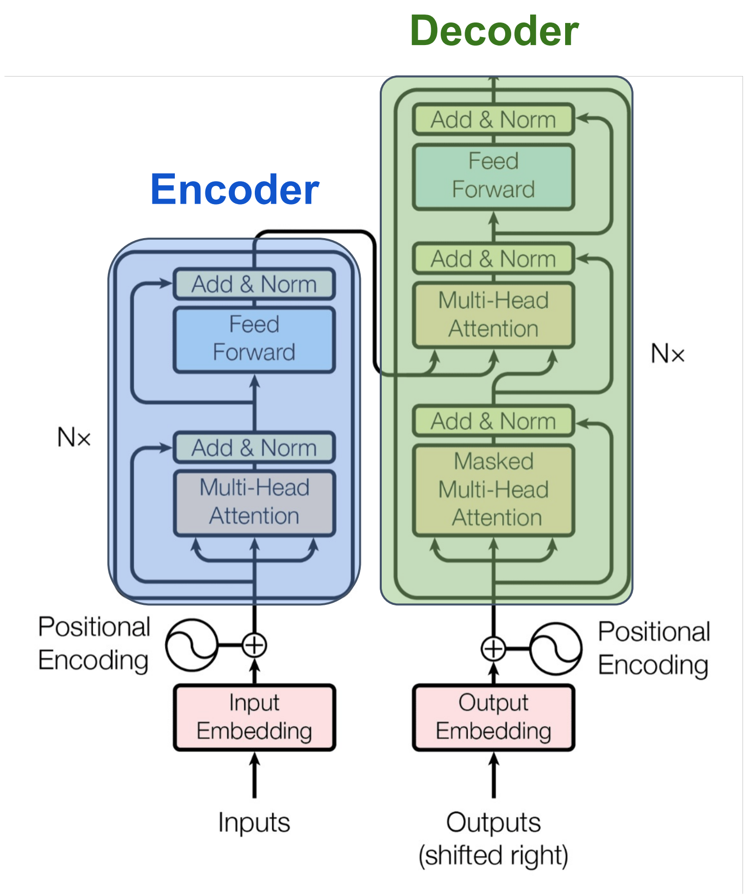
[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Transformers

- ➢ Extracting pixel-level representation
- ➢ Converting to object-level representation

Transformer Encoder:

- ❑ Self-attention
- ❑ Serves as strong feature extracting/enhancing modules.



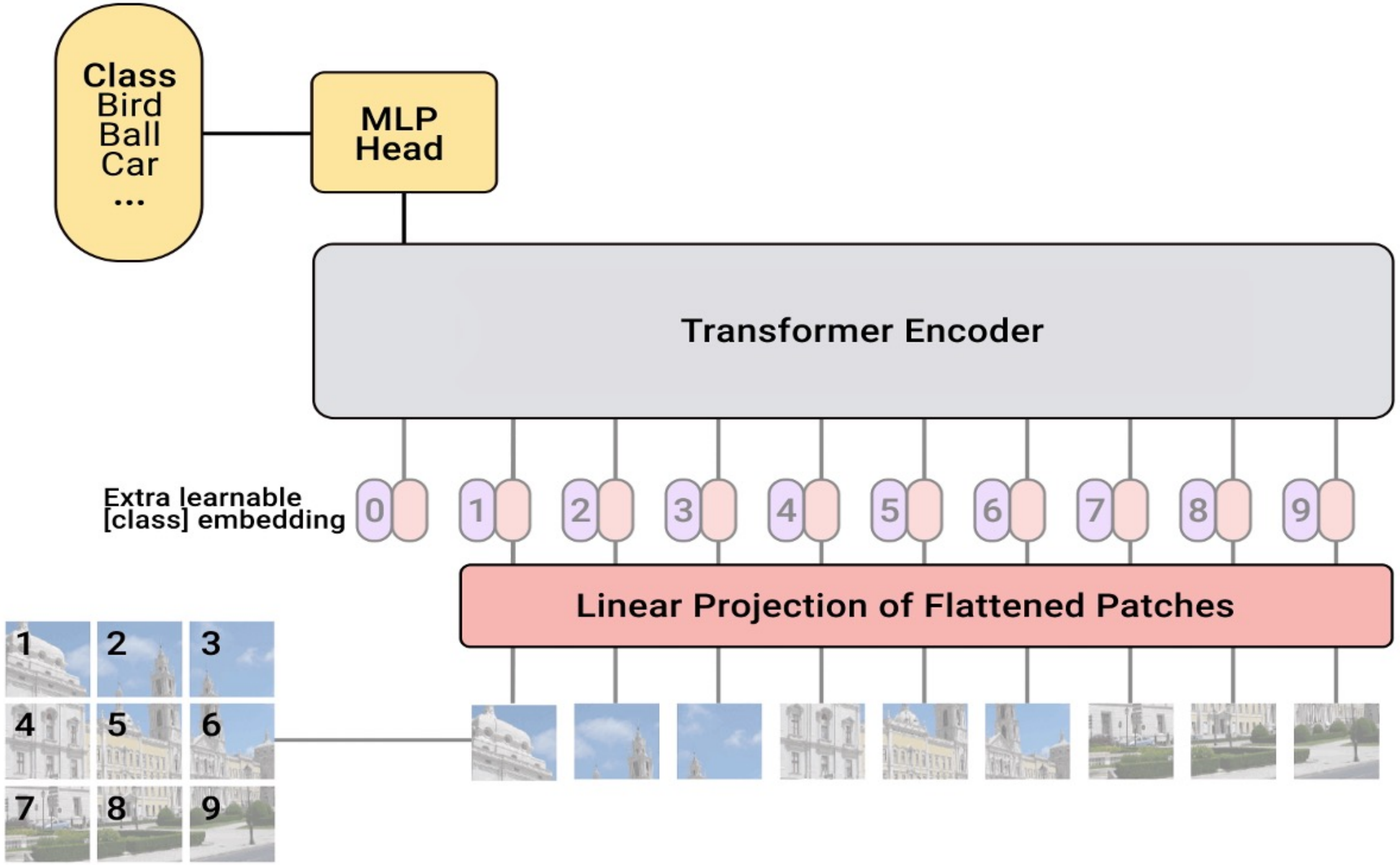
Transformer Decoder:

- ❑ Cross-attention
- ❑ Serves as a bridge for sequence-to-sequence translation

[1] Vaswani, et al. Attention is all you need. NeurIPS, 2017

Vision Transformers (ViTs)

ViT: the pioneer of Transformer architectures for vision



[1] Dosovitskiy, et al. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021

Vision Transformers (ViTs)

ViT: overview

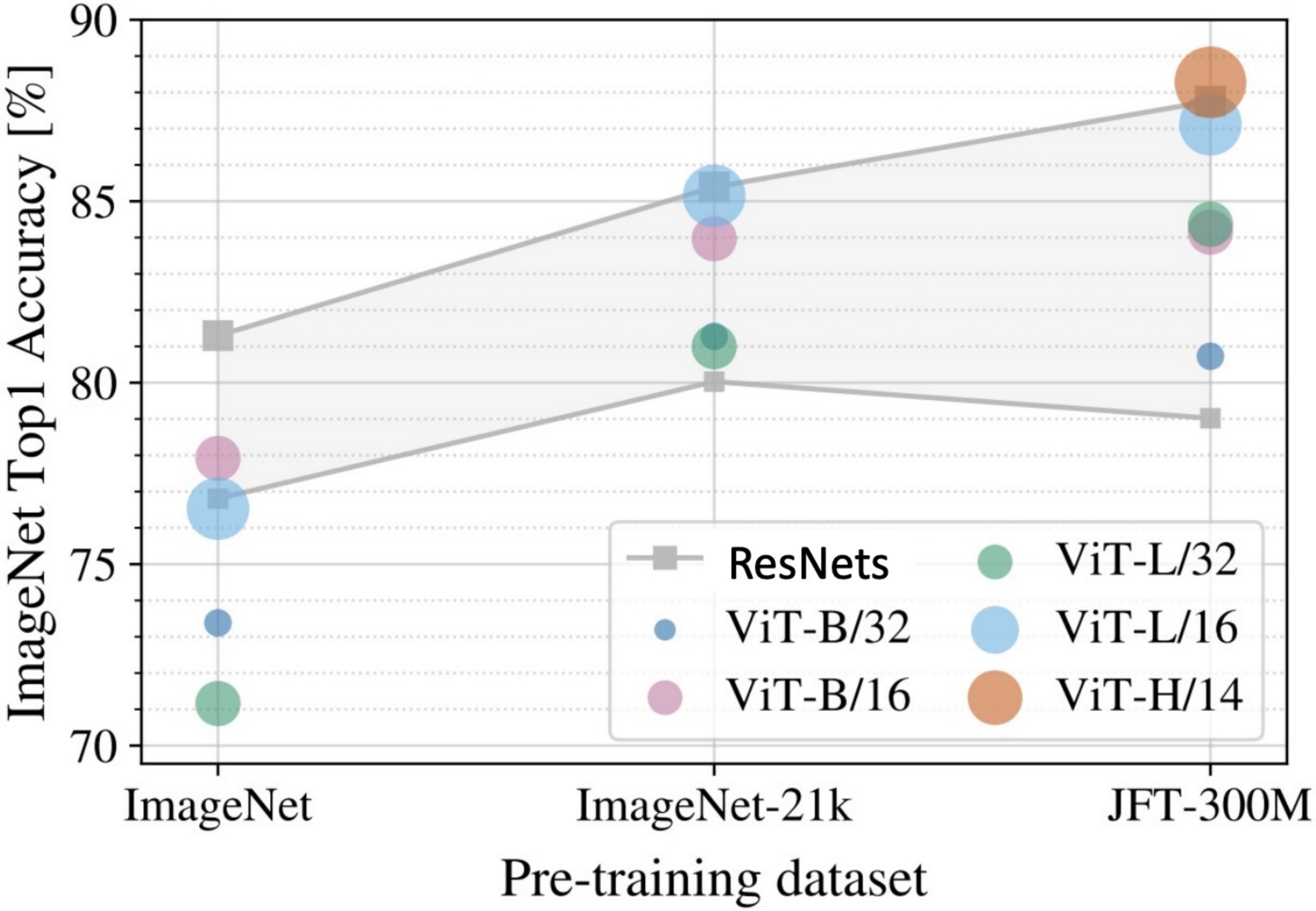
- Divide an input image into 196 (14x14) small images of size (16x16)
- Treat it as embedding in NLP
- Use it as an input for traditional transformer encoder (like in BERT)
- Use 12 transformer layers (Norm, Multi-head attention, etc.)
- the last output, use it as input for Dense Layer with 1000
- you have a classification model

[1] Dosovitskiy, et al. ViT: An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021

ViT: performance comparison to CNN

JFT-300M is an internal Google dataset with 300M labeled images

If you pretrain on JFT and finetune on ImageNet, large ViTs outperform large ResNets



B = Base
L = Large
H = Huge

/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

ViT: strength and weakness

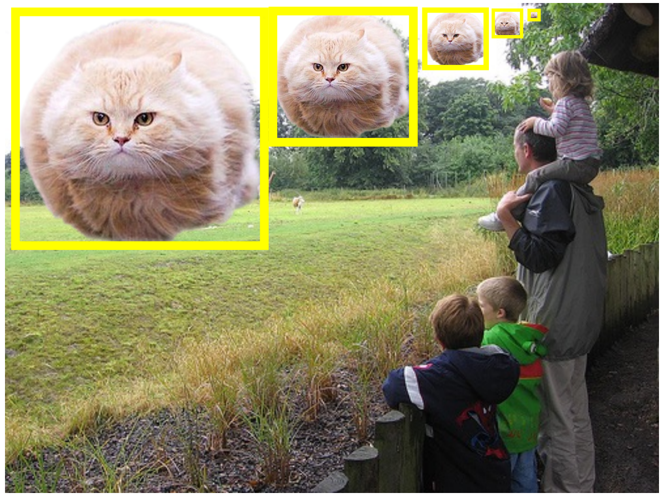
- Strength:
  - Long-range interaction via global attention
  - Strong performance/scalability with large scale of data
- Weakness:
  - Quadratic computational complexity
  - Lacking locality
  - Lacking low-level details due to large patch size (e.g., 16x16)

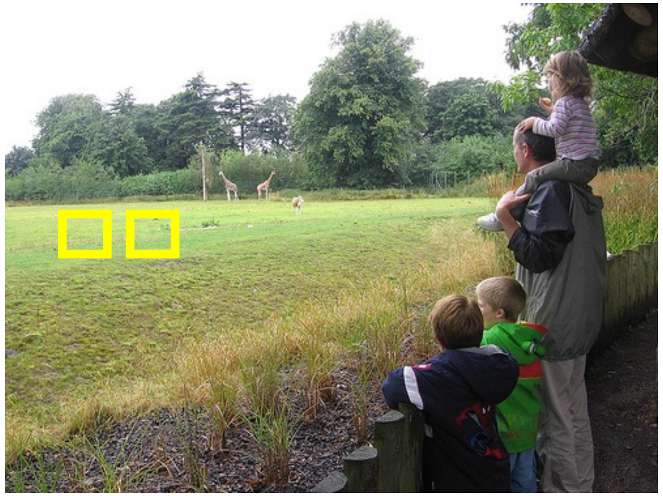
Key difference between Visual and Text Signals

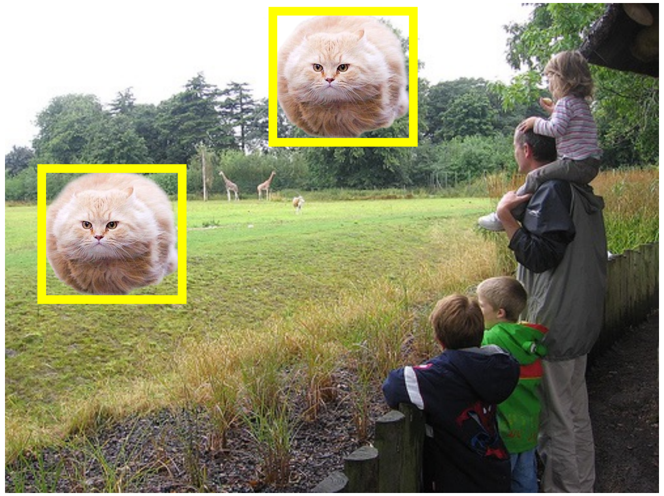
multi-scale
(scale invariance)

locality
(spatial smoothness)

translation invariance







I am a **fat cat.**
I am a **fat fat cat cat.** (invalid)

No scale variation

I like the **green grass.**

No spatial smoothness

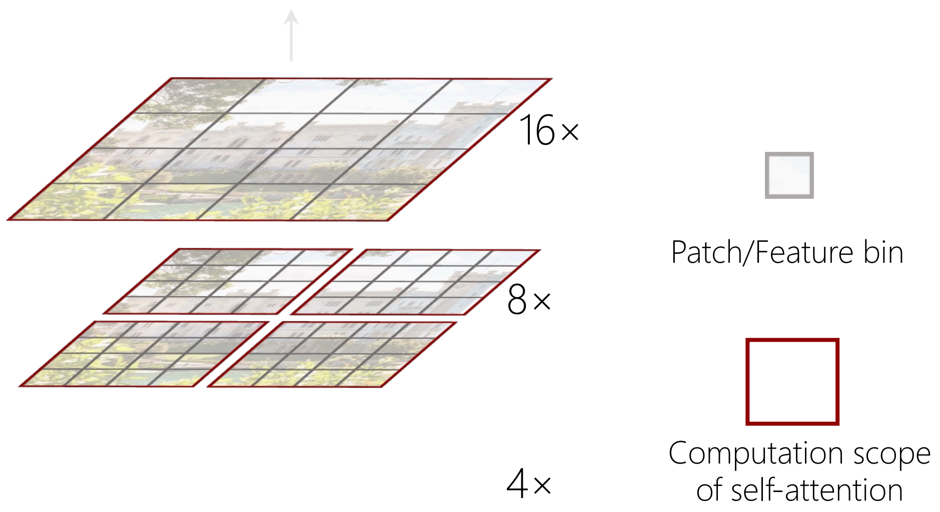
I am a **fat cat.**
Fat cat is me.

Sensitive to absolute locations

Example adapted from Ross Girshick and Ze Liu

Hierarchical Vision Transformer

- Swin Transformer: Hierarchical Vision Transformer using Shifted Windows
  - Multi-scale (hierarchical) feature map
  - Use shifted window to get image patches



More friendly to visual signals

multi-scale / locality / translation invariance

Hybrid CNN-Transformer Model

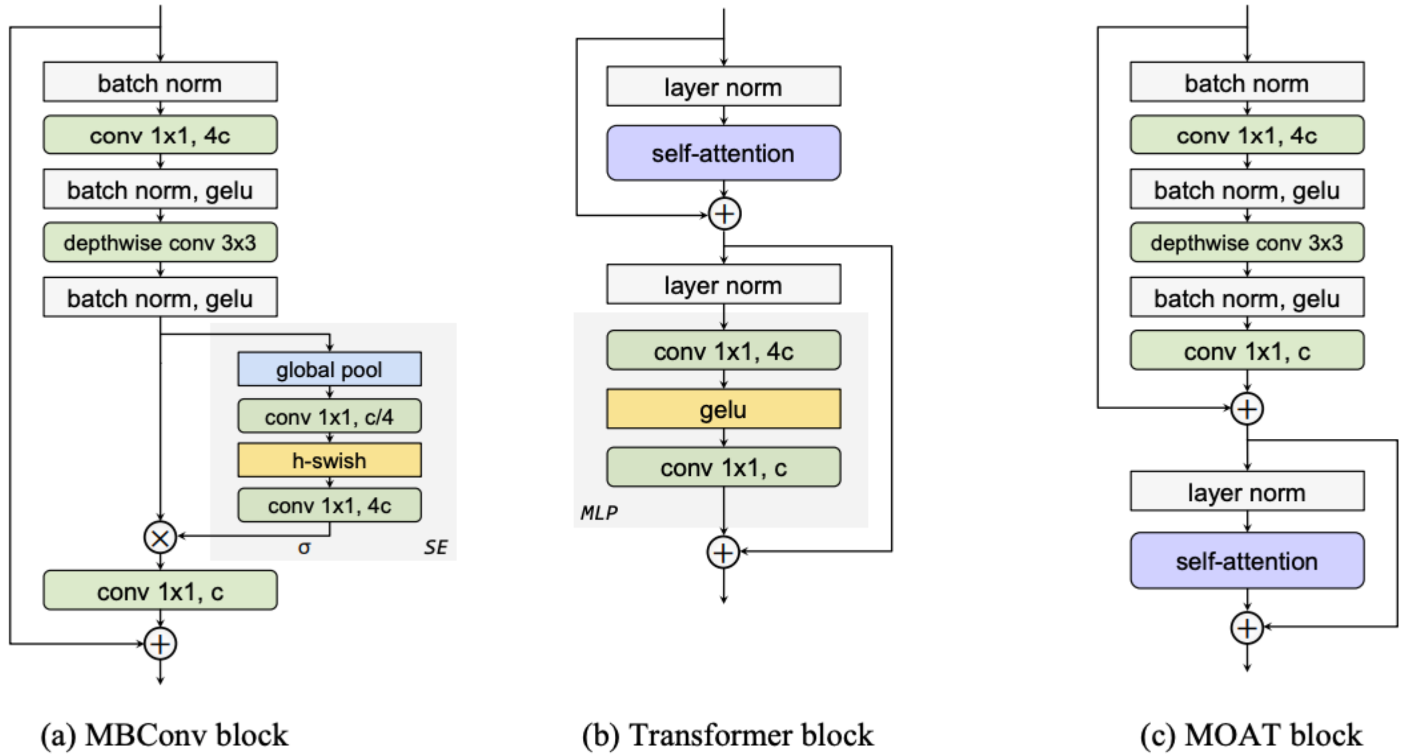
- Formulation:
  - Micro-level Hybrid Model
    - Each basic block is designed with convolutional layer and self-attention layers
    - The model stack with several basic blocks repeatedly
  - Macro-level Hybrid Model
    - Each basic block is either pure CNN block or pure Transformer block
    - The model starts with few CNN blocks
    - The model further stacks several Transformer blocks
- Strength
  - Preserve local information due to CNN
  - Preserve translation invariance due to CNN
  - Efficient learning of spatial hierarchy due to CNN
  - Enable global interaction due to Transformer

Examples

- Hybrid Transformer for visual recognition: Yang et al., MOAT, ICLR 2023
- Hybrid Transformer for medical AI: Chen et al., TransUNet, arXiv 2021
- Hybrid Transformer for vision-language: Chen et al., ViTamin, CVPR 2024

[1] Yang, et al. Moat: Alternating mobile convolution and attention brings strong vision models ICLR 2023
[2] Chen, et al. Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306, 2021.
[3] Chen, et al. Design scalable vision models in the vision-language era, CVPR 2024

Hybrid Transformer for Visual Recognition



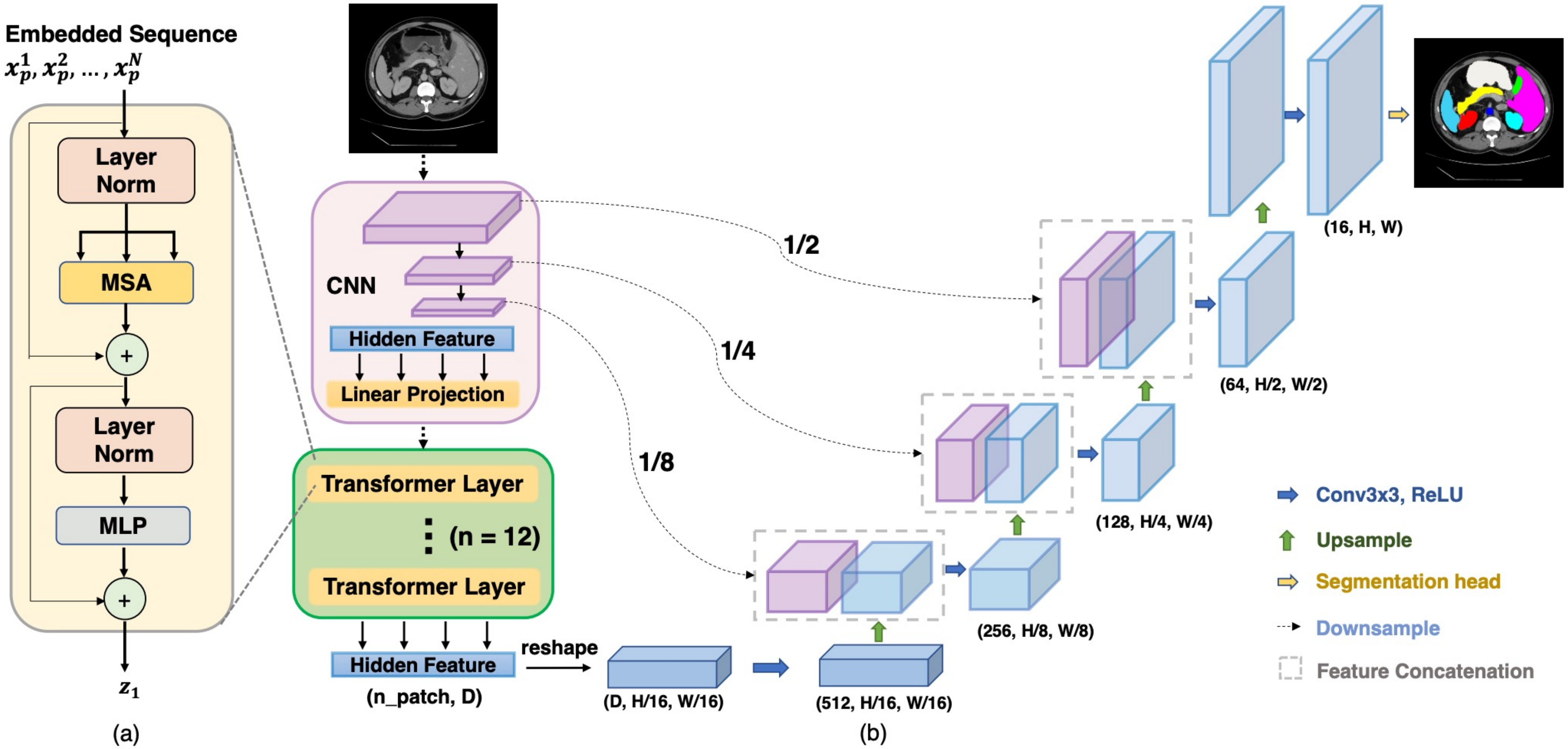
- Yang et al. MOAT ICLR2023

Figure 1: **Block comparison.** (a) The MBConv block (Sandler et al., 2018) employs the inverted bottleneck design with depthwise convolution and squeeze-and-excitation (Hu et al., 2018) applied to the expanded features. (b) The Transformer block (Vaswani et al., 2017) consists of a self-attention module and a MLP module. (c) The proposed MOAT block effectively combines them. The illustration assumes the input tensor has channels *c*.

[1] Yang, et al. Moat: Alternating mobile convolution and attention brings strong vision models ICLR 2023

Hybrid Transformer + UNet: TransUNet

- Chen et al. 2021



[2] Chen, et al. Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306, 2021.

Hybrid Transformer for Vision-Language

- Chen et al. CVPR 2024

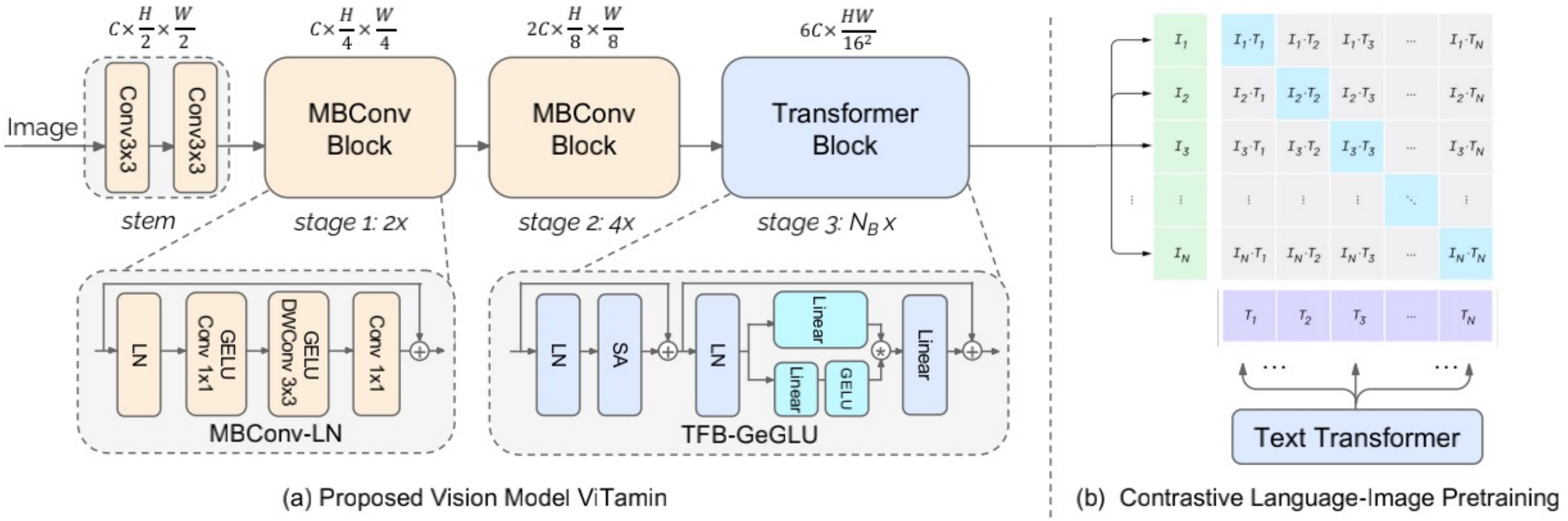


Figure 3. **Overview of ViTamin architecture**. (a) ViTamin begins with a convolutional stem, followed by Mobile Convolution Blocks (MBConv) in stage 1 and 2, and Transformer Blocks (TFB) in stage 3. The 2D input to the stage 3 is flatten to 1D. For the *macro-level* designs, the three-stage layout generates the final feature map with output stride 16, similar to ViT/16 [31]. We set channels sizes for the three stages to be (C, $2C$, $6C$). For the *micro-level* designs, the employed MBConv-LN modifies MBConv [115] by using a single LayerNorm [4]. TFB-GeGLU upgrades TFB's FFNs [131] (Feed-Forward Networks) with GELU Gated Linear Units [117]. (b) In the CLIP framework, given N image-text pairs, the vision model's output I_i is learned to align with its corresponding text Transformer's output T_i. Our text Transformers are the same as OpenCLIP [62]. +: Addition. *: Multiplication.

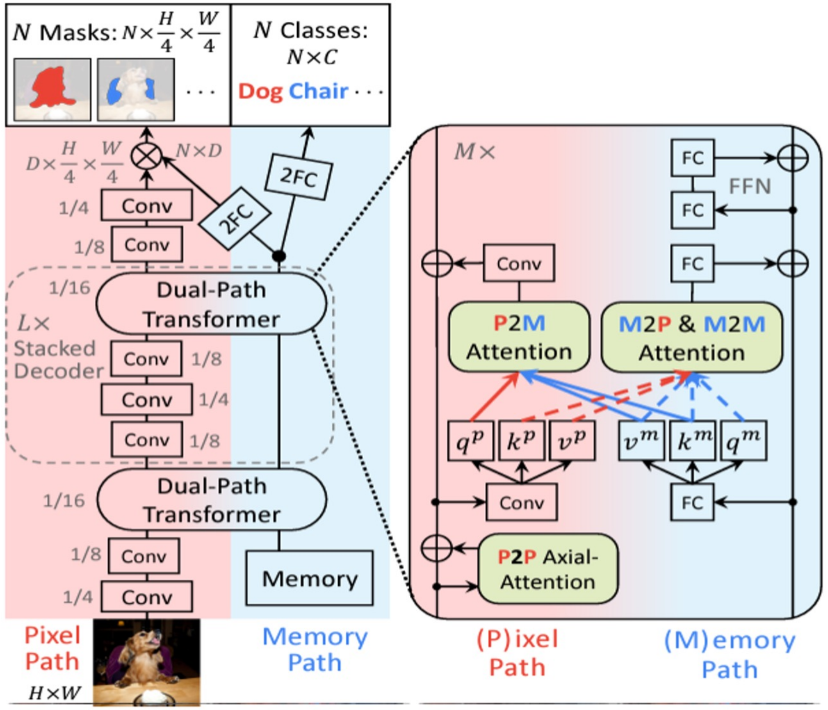
[1] Chen, et al. Design scalable vision models in the vision-language era, CVPR 2024

Summary

- Transformers consist of several Transformer blocks with multi-head self-attention layer and feedforward layers.
- It is highly scalable and highly parallelizable
- Faster training, larger models, better performance across vision and language tasks
- Good capability in vision tasks.

Dual-Path Transformer: MaX-DeepLab

- Dual Path: Pixel Path + Memory Path
  - Memory to store global information
- Self-attention:
  - P2P: Pixel-to-pixel self-attention
    - Both query and key are pixel vector itself
  - M2M: Memory-to-memory self-attention
    - Both query and key are memory itself
- Cross-attention
  - P2M: Pixel-to-memory cross-attention
    - Query is pixel, key is memory
  - M2P: Memory-to-pixel cross-attention
    - Query is memory, key is pixel

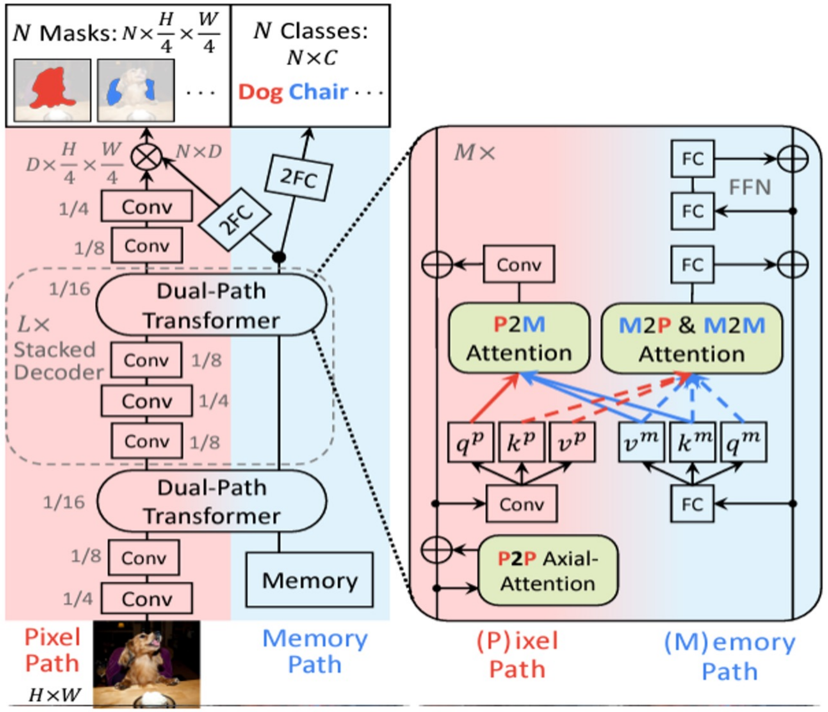


(a) Overview of MaX-DeepLab (b) Dual-path transformer block

Wang, et al. MaX-DeepLab: End-to-End Panoptic Segmentation With Mask Transformers. CVPR 2021.

Dual-Path Transformer: MaX-DeepLab

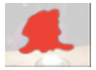
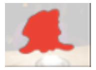
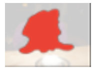
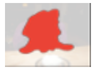
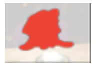
- Dual Path: Pixel Path + Memory Path
- Prediction: N pairs of object class $p_j^{class} \in \mathbb{R}^1$ and mask $p_j^{mask} \in \mathbb{R}^{HW}$
  - Memory is decoded to N object class
  - The dot product of Memory and Pixel result in N object masks



(a) Overview of MaX-DeepLab (b) Dual-path transformer block

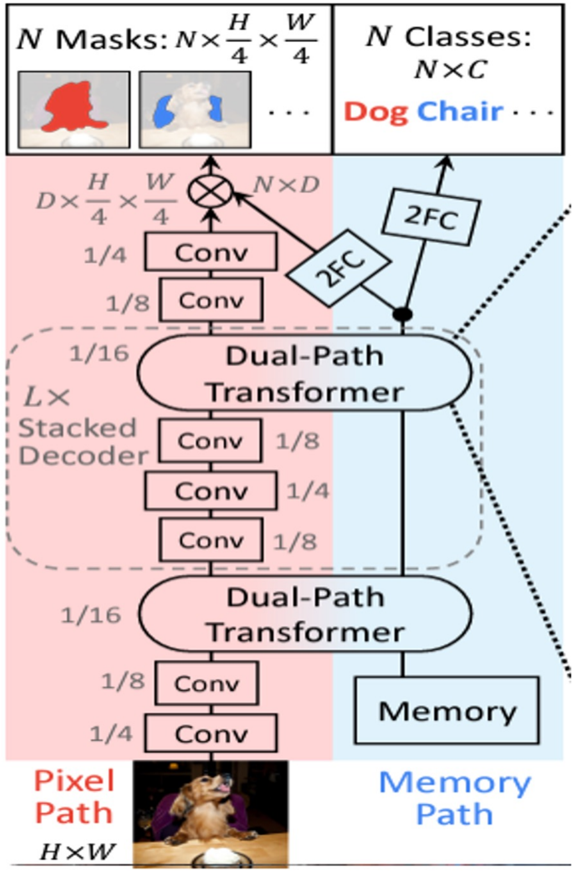
Wang, et al. MaX-DeepLab: End-to-End Panoptic Segmentation With Mask Transformers. CVPR 2021.

Hungarian Matching for Prediction-Groundtruth Association

- Goal: match the grountruth y_i to a prediction p_j
  - Ground-truth: M pairs of object class $y_i^{class} \in \mathbb{R}^1$ and mask $y_i^{mask} \in \mathbb{R}^{HW}$
  - Prediction: N pairs of object class $p_j^{class} \in \mathbb{R}^1$ and mask $p_j^{mask} \in \mathbb{R}^{HW}$
    - Memory is decoded to N object class
    - The dot product of Memory and Pixel result in N object masks
- Example:
  - Successful match: y_i = {dog,  } ; p_j = {dog,  }
    - Classification accuracy = 1, and mask IoU = 1
  - Failed match: y_i = {dog,  } ; p_j = {dog,  } -> low mask IoU
    - Classification accuracy = 1, but mask IoU = 0.1
  - Failed match: y_i = {dog,  } ; p_j = {cat,  } -> wrong classification
    - Mask IoU=1, but classification accuracy=0

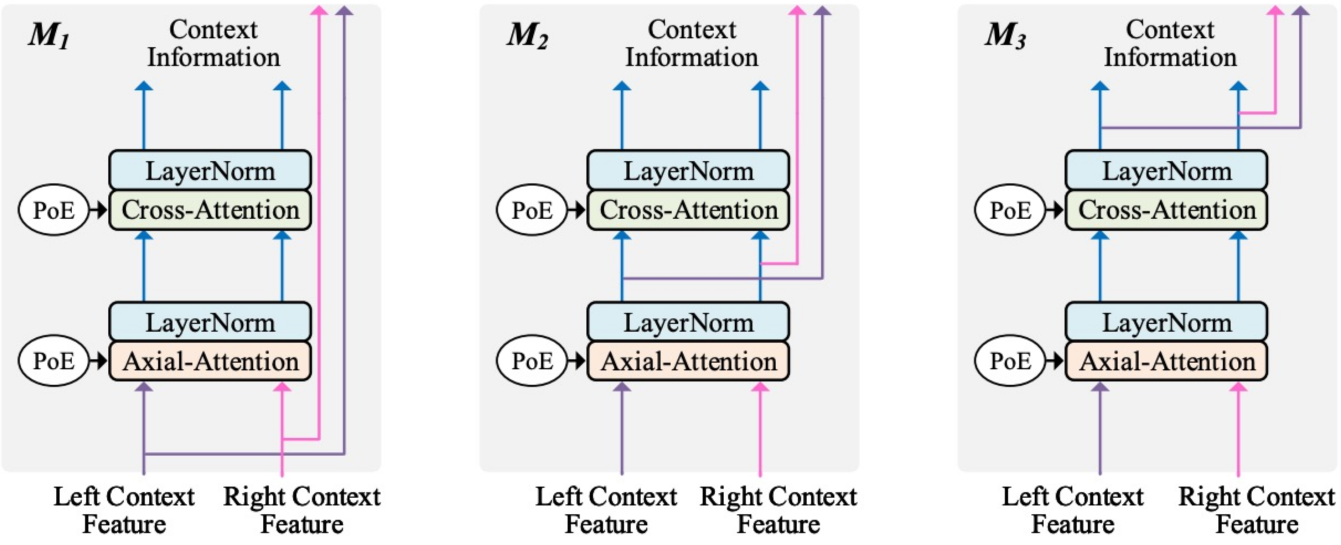
Hungarian Matching for Prediction-Groundtruth Association

- Goal: match the grountruth y_i to a prediction p_j
  - Ground-truth: M pairs of object class y_i^{class} and mask y_j^{mask}
  - Prediction: N pairs of object class p_i^{class} and mask p_j^{mask}
    - Memory is decoded to N object class
    - The dot product of Memory and Pixel result in N object masks
- Define cost matrix C of size $M \times N$
  - $C_{i,j} = -accuracy\left(y_i^{class}, p_j^{class}\right) - IoU\left(y_i^{mask}, p_j^{mask}\right)$
- Linear assignment problem to do association
  - Formally, the task is to find a injection $f: \{1,2,..m\} \rightarrow: \{1,2,..n\}$ that minimizes the total assignment cost: Minimize $\sum_{i=1}^{M} C_{i,f(i)}$



Cross-attention for Stereo Matching / Fusion

- The context feature of left and right images are fused with cross-attention
- Query is left context features and key is the right context feature, or vice verse
- Reference to context-enhanced stereo Transformer [1] to estimate disparity



(a) M1 (b) M2 (c) M3

[1] Guo, et al. Context-enhanced stereo transformer ECCV 2022

Context Enhanced Stereo Transformer

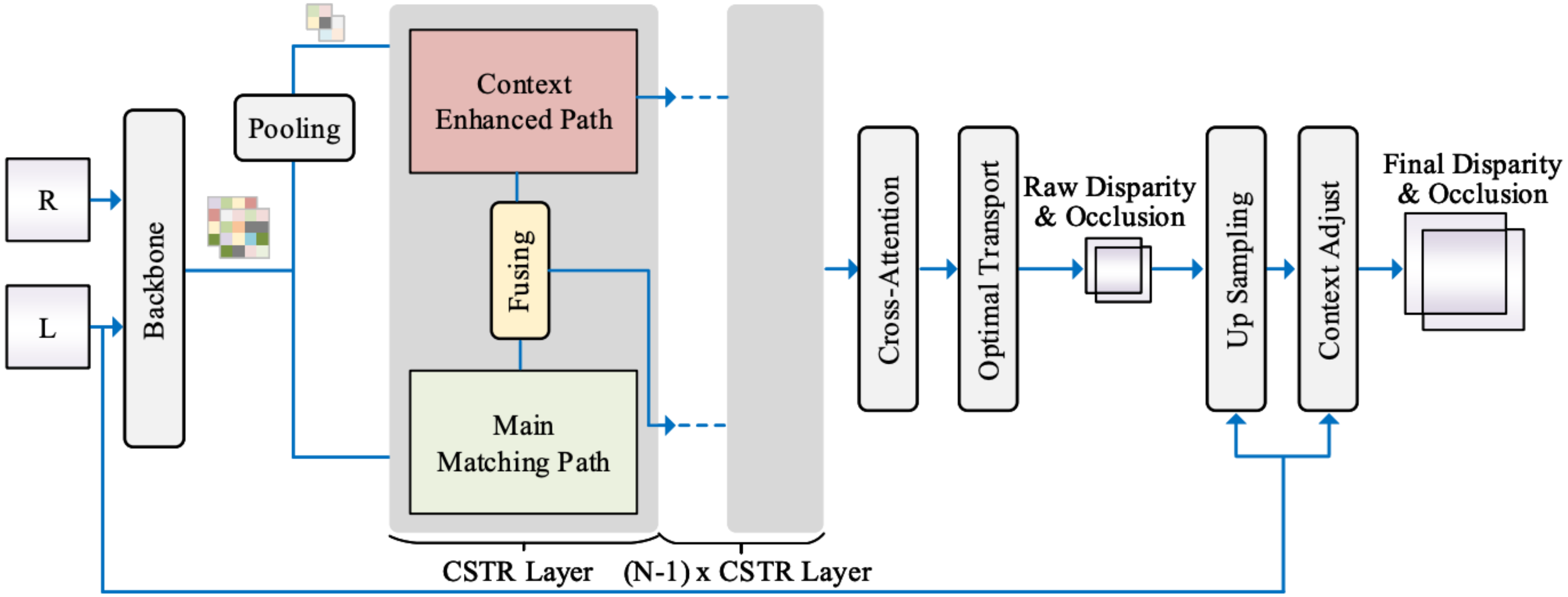
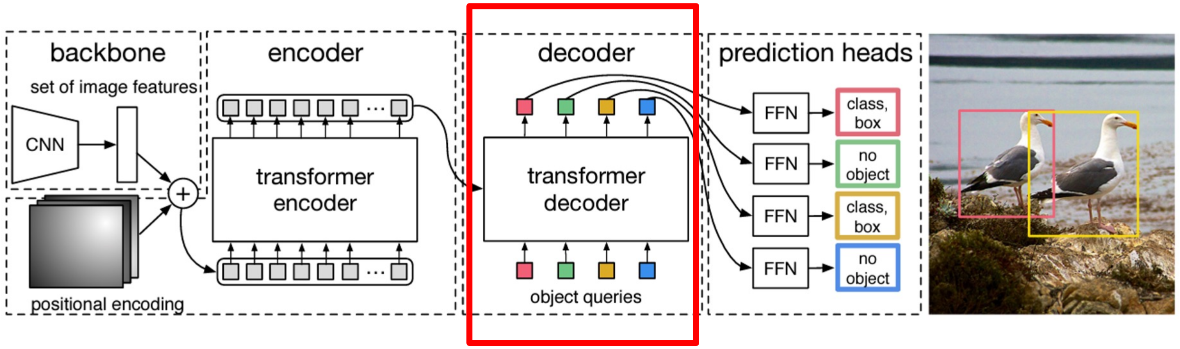


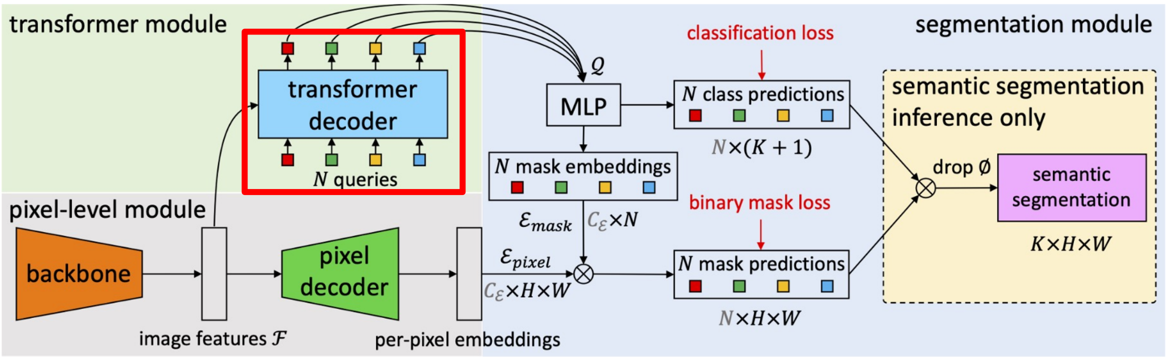
Fig. 3. CSTR consists of two main components:(1) Context Enhanced Path that extracts long-range context information in low resolution feature. (2) Main Matching Path that use Axial-Attention to enhance context and Cross-Attention to compute raw disparity. Then a learnable Up Sampling block up restore the original scale of disparity and Context Adjustment block refines the disparity with context information across epipolar lines conditioned on the left image.

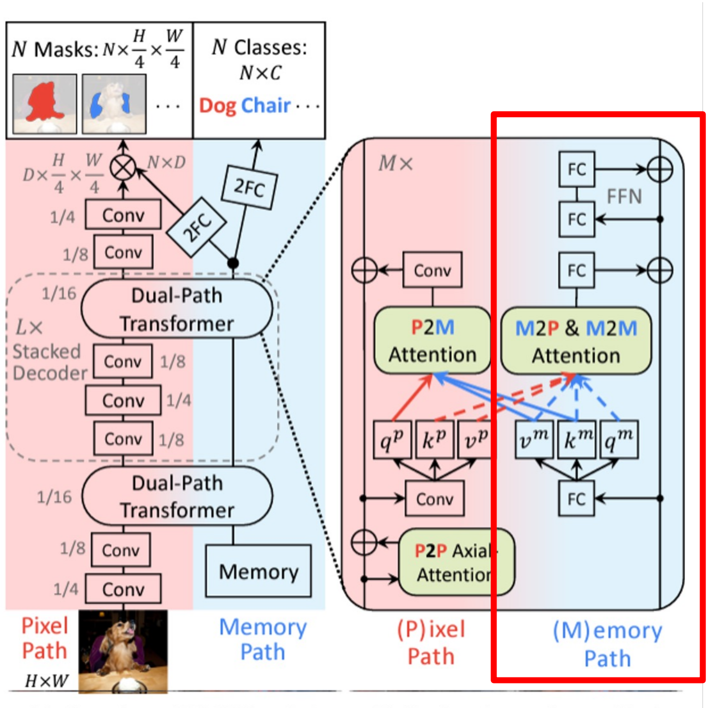
[1] Guo, et al. Context-enhanced stereo transformer ECCV 2022

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Transformer decoder (cross-attention) is crucial in terms of an end-to-end problem formulation:







(a) Overview of MaX-DeepLab (b) Dual-path transformer block

Carion, et al. End-to-end object detection with transformers. ECCV 2020.
Wang, et al. MaX-DeepLab: End-to-End Panoptic Segmentation With Mask Transformers. CVPR 2021.
Cheng., et al. Per-Pixel Classification is Not All You Need for Semantic Segmentation. NeurIPS 2021.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

Yu., et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone
- Image feature enhancement through transformer encoder (self-attention)

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone
- Image feature enhancement through transformer encoder (self-attention)
- learnable object queries aggregate pixel features through transformer decoder (cross-attention) -> box/mask embedding

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone
- Image feature enhancement through transformer encoder (self-attention)
- learnable object queries aggregate pixel features through transformer decoder (cross-attention) -> box/mask embedding
- huangarian matching for supervision assignments

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Common pipeline:

- Image feature extraction through a backbone
- Image feature enhancement through transformer encoder (self-attention)
- **learnable object queries aggreate pixel features through transformer decoder (cross-attention) -> box/mask embedding**
- huangarian matching for supervision assignments

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)
- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights
- The attentior xel features, with linear project

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

- The update of queries is added in a residual manner.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)

- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights

- The attentior … xel features, with linear project

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}\left(\boxed{\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}}\right) \times \mathbf{V}^p,$$

- The update of queries is added in a residual manner.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)

- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights

- The attentior … xel features, with linear project…

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\operatorname{softmax}}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

- The update of queries is added in a residual manner.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)

- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights

- The attention … xel features, with linear project…

$$\hat{\mathbf{C}} = \mathbf{C} + \operatorname*{softmax}_{HW}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \boxed{\times \mathbf{V}^p,}$$

- The update of queries is added in a residual manner.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)
- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights
- The attentior … xel features, with linear project

$$\hat{\mathbf{C}} = \boxed{\mathbf{C} +} \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

- The update of queries is added in a residual manner.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

Cross-Attention:

- Affinity logits are computed, with linear projections, between queries (Q^c) and pixels (K^p)

- A spatial-wise (HW) softmax is applied to convert the affinity logits map to attention weights

- The attentior … xel features, with linear project

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

- The update of queries is added in a residual manner

As a result, the queries will be updated and converted to correspond to a specific object in prediction.

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

- Inconsistency of using object queries for mask prediction and updating object queries

- Sparse attention map

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

- Inconsistency of using object queries for mask prediction and updating object queries

mask prediction:

$$\underbrace{\mathbf{Z}}_{\text{mask probs}} = \operatorname{softmax}_{N}(\underbrace{\mathbf{F} \times \mathbf{C}^{\mathrm{T}}}_{\text{affinity logits}}),$$

- Sparse attention map

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

- Inconsistency of using object queries for mask prediction and updating object queries

mask prediction:

$$\mathbf{Z} = \boxed{\underset{N}{\mathrm{softmax}}}(\mathbf{F} \times \mathbf{C}^{\mathrm{T}}),$$

updating object queries:

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\mathrm{softmax}}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

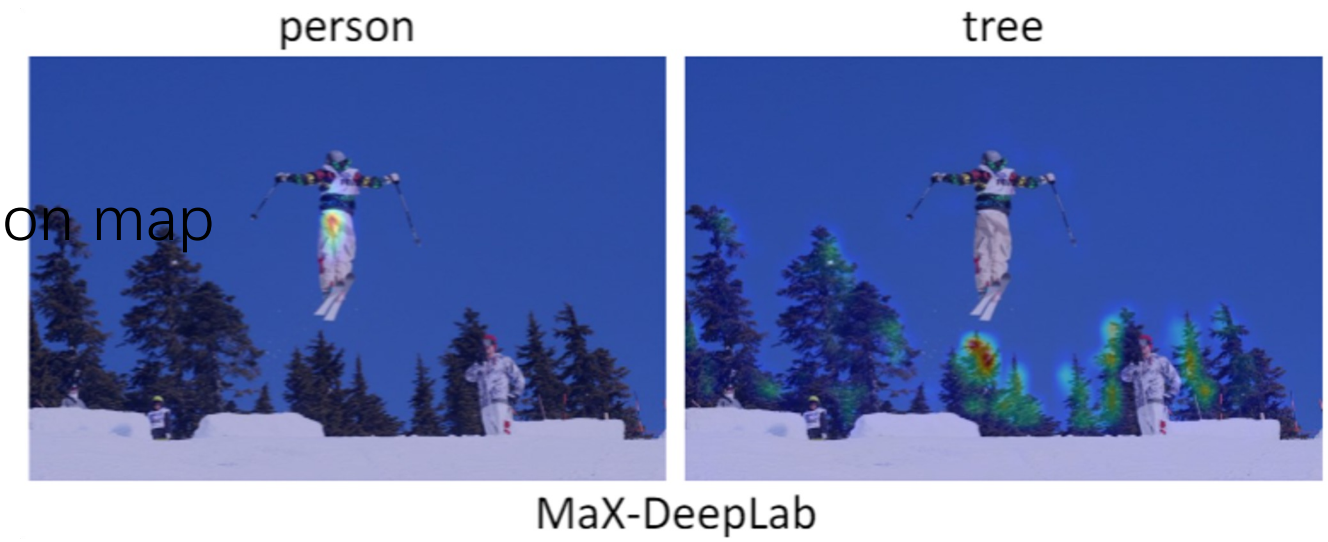
- Sparse attention map

Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

- Inconsistency of using memory queries for mask prediction and updating memory queries

- Sparse attention map



Yu, et al. CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation. CVPR 2022, Oral.

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \boxed{\mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,}$$

Cross-attention -> a clustering process

object queries -> cluster centers

attention map -> clustering assignment

updating object queries -> updating clustering center

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\boxed{\hat{\mathbf{C}}} = \boxed{\mathbf{C}} + \underset{HW}{\mathrm{softmax}}(\boxed{\mathbf{Q}^c} \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Cross-attention -> a clustering process

object queries -> cluster centers

attention map -> clustering assignment

updating object queries -> updating clustering center

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}})} \times \mathbf{V}^p,$$

Cross-attention -> a clustering process

object queries -> cluster centers

attention map -> clustering assignment

updating object queries -> updating clustering center

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \boxed{\times \mathbf{V}^p,}$$

Cross-attention -> a clustering process

object queries -> cluster centers

attention map -> clustering assignment

updating object queries -> updating clustering center

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Machine Translation: Each object query corresponds to a word in target language, and it will be assigned to one most affiliated word in source language as its update

Panoptic Segmentation: Each object query corresponds to an object in prediction, and it will be assigned to one most affiliated pixel as its update

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Machine Translation: Each object query corresponds to a word in target language, and it will be assigned to one most affiliated word in source language as its update

Panoptic Segmentation: Each object query corresponds to an object in prediction, and it will be assigned to one most affiliated pixel as its update

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Machine Translation: Each object query corresponds to a word in target language, and it will be assigned to one most affiliated word in source language as its update

Panoptic Segmentation: Each object query corresponds to an object in prediction, and it will be assigned to one most affiliated pixel as its update

objects assigned to pixels?

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\operatorname{softmax}}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Machine Translation: Each object query corresponds to a word in target language, and it will be assigned to one most affiliated word in source language as its update

Panoptic Segmentation: Each object query corresponds to an object in prediction, and it will be assigned to one most affiliated pixel as its update

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

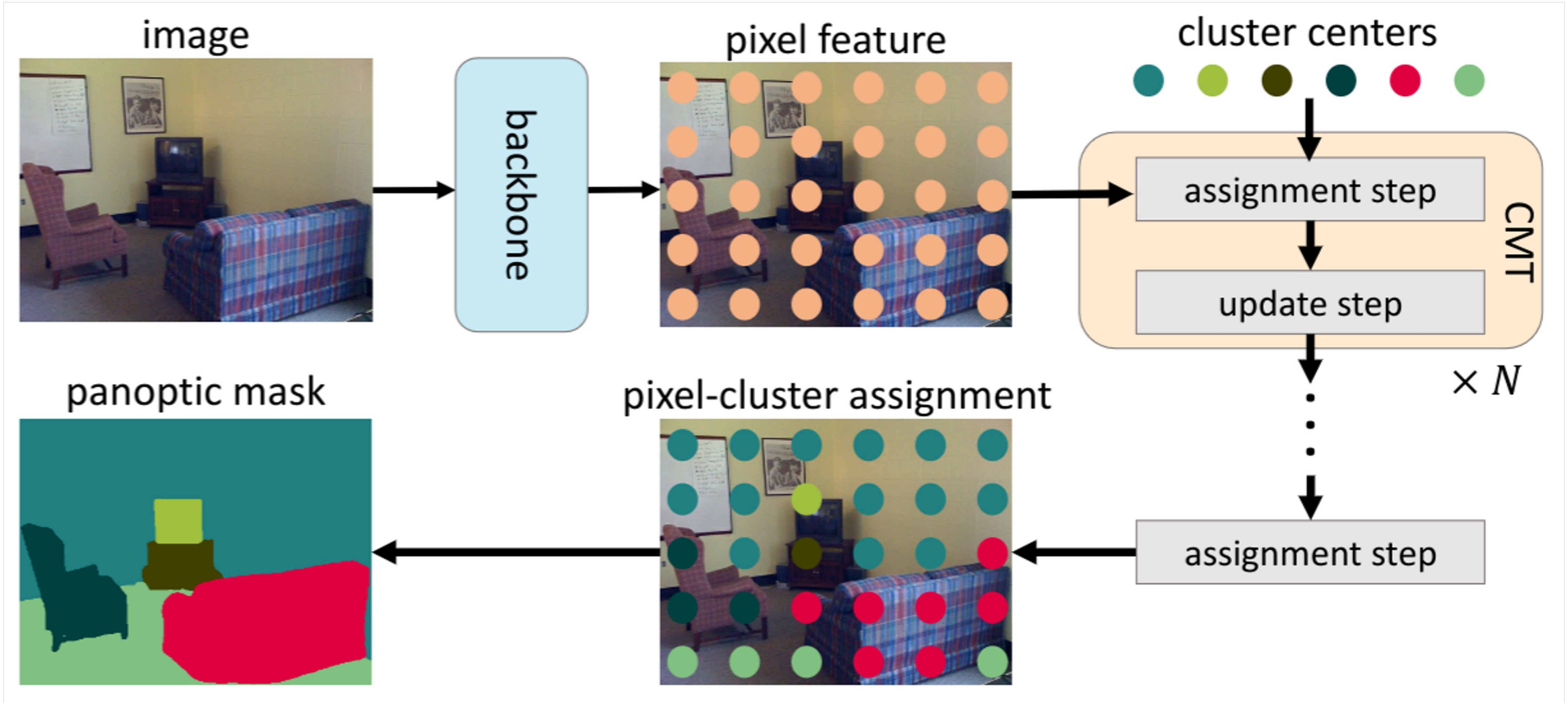
$$\cancel{\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\operatorname{softmax}}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,}$$

$$\hat{\mathbf{C}} = \mathbf{C} + (\boxed{\underset{N}{\operatorname{softmax}}}(\tilde{\mathbf{K}}^p \times (\tilde{\mathbf{Q}}^c)^{\mathrm{T}}))^{\mathrm{T}} \times \mathbf{V}^p.$$

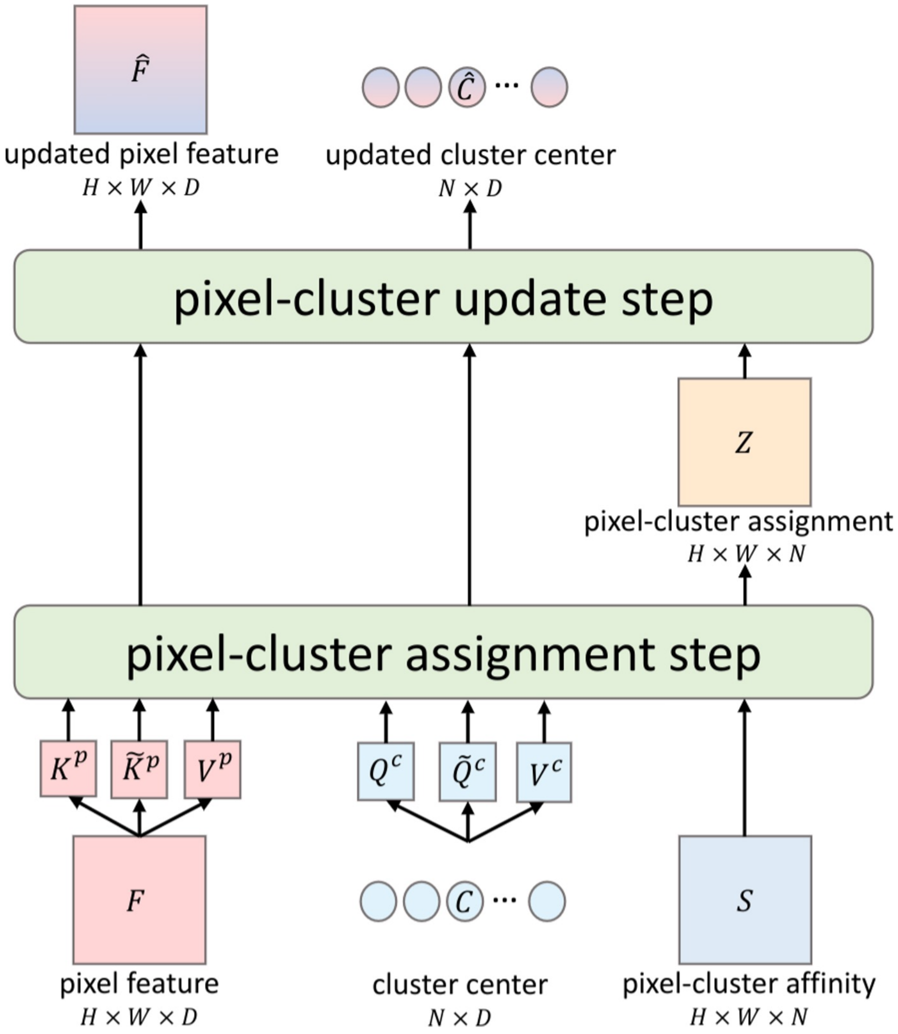
~~*Panoptic Segmentation:* Each object query corresponds to an object in prediction, and it will be assigned to one most affiliated pixel as its update~~

Panoptic Segmentation: Each pixel will choose one most affiliated object, all assigned pixels will serve as an update to corresponding object query

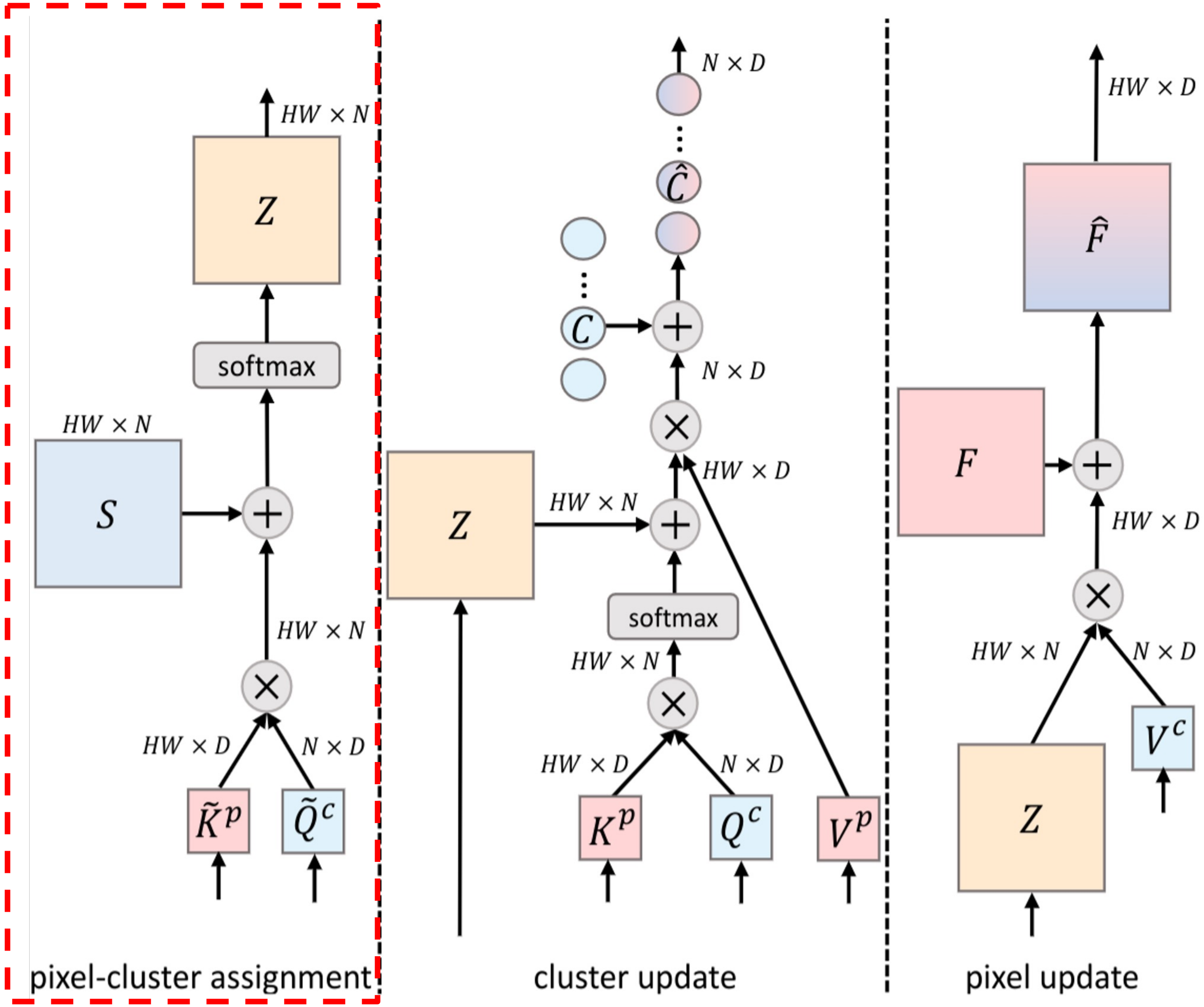
CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



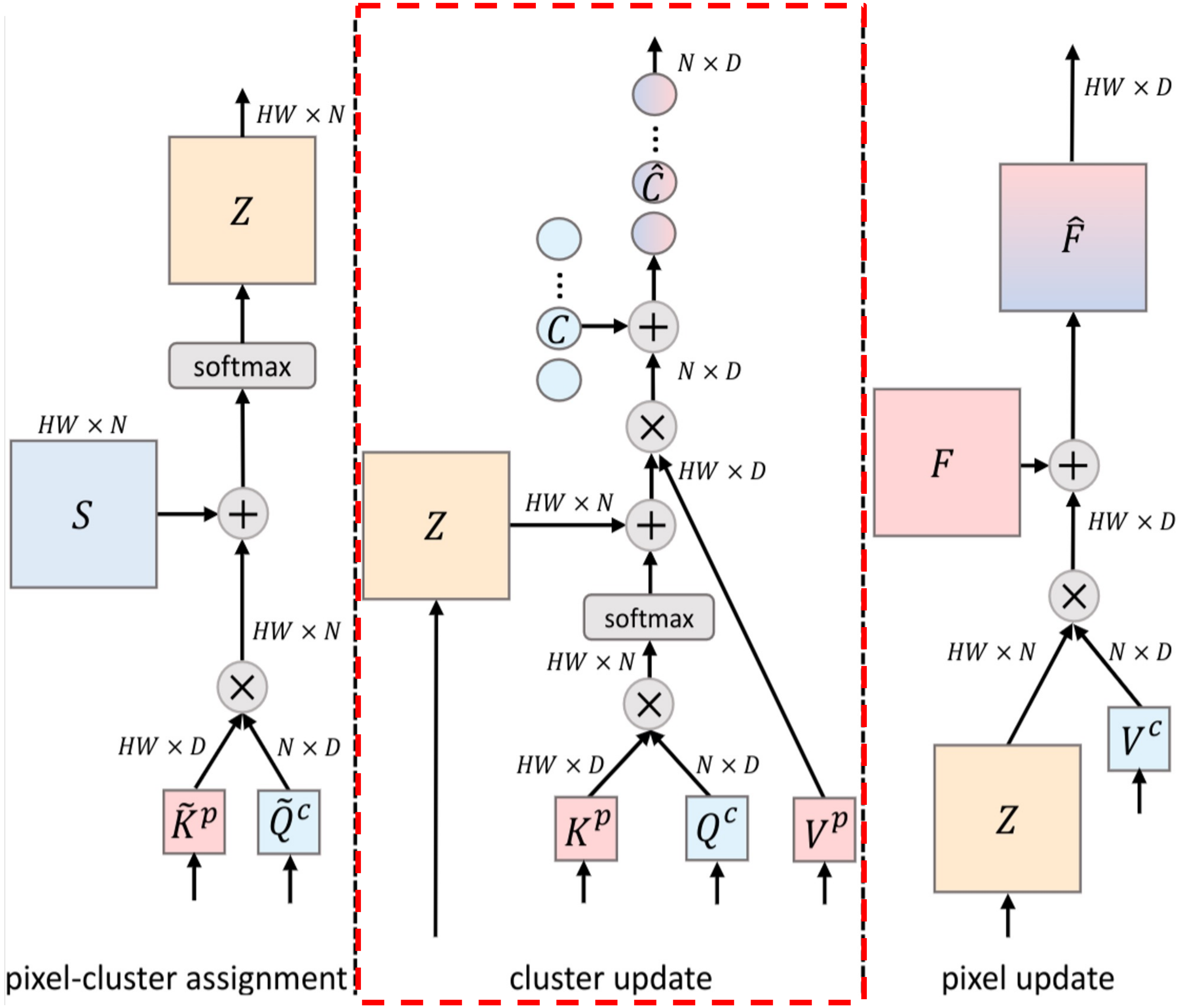
CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



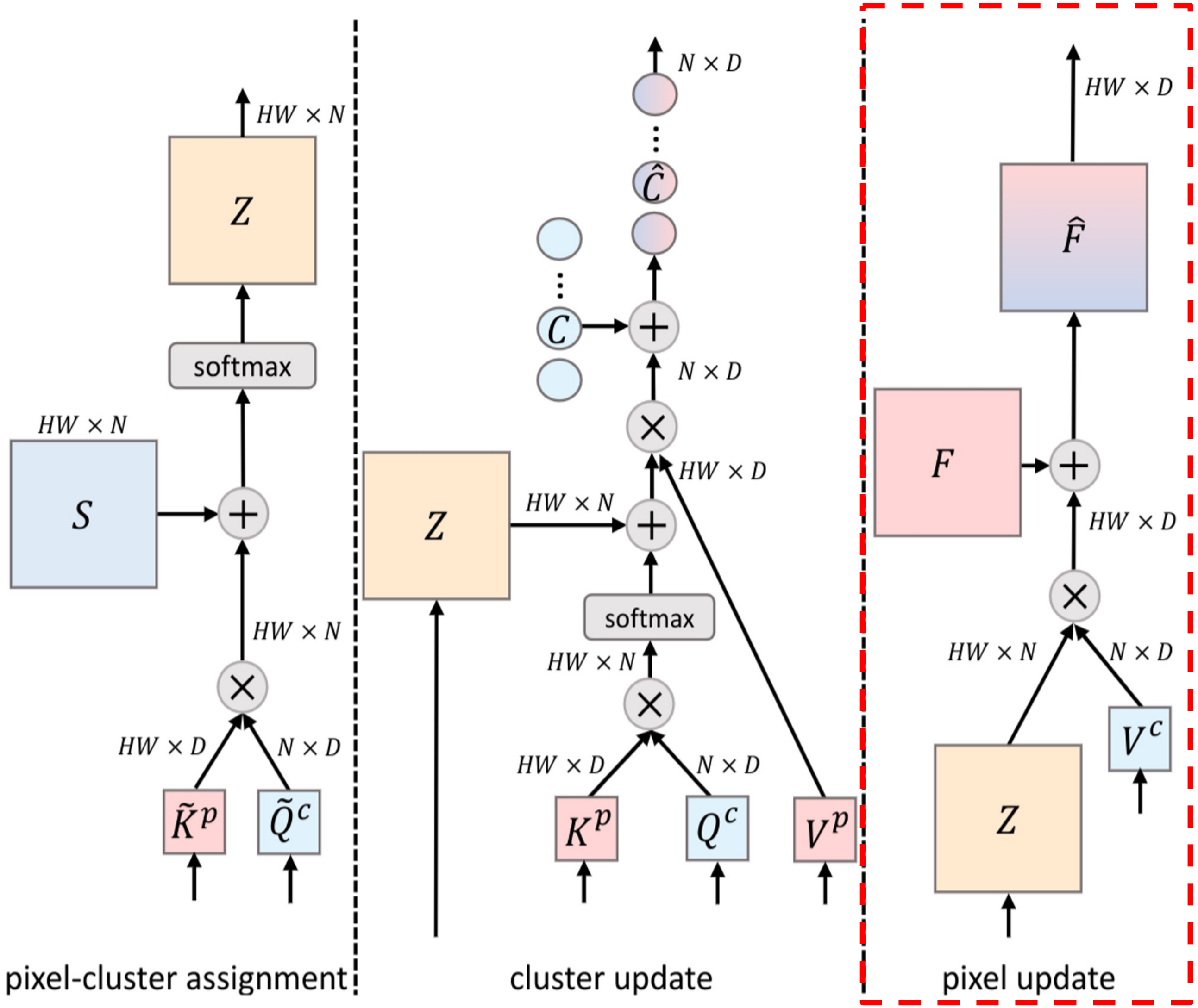
CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

- Include coordinates into the clustering

- Improve instance discrimination loss in MaX-DeepLab to pixel-wise contrastive loss

Achanta., et al. SLIC Superpixels Compared to State-of-the-Art Superpixel Method. TPAMI.
Wang., et al. MaX-DeepLab: End-to-End Panoptic Segmentation With Mask Transformers. CVPR 2021.

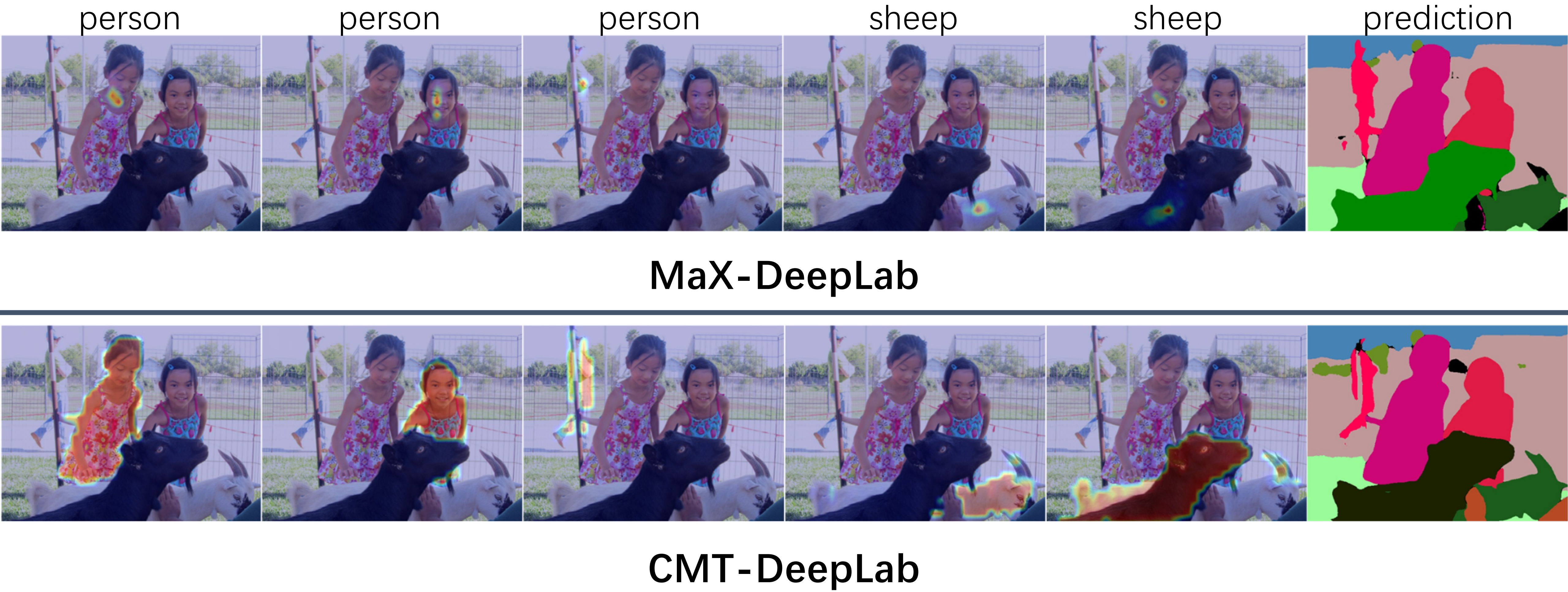
CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation

We build CMT-DeepLab upon previous SOTA method MaX-DeepLab:

COCO Panoptic:
+**4.2**% *val* PQ
+**4.4**% *test* PQ

Cityscapes Panoptic:
+**2.9**% *val* PQ

CMT-DeepLab: Clustering Mask Transformers for Panoptic Segmentation



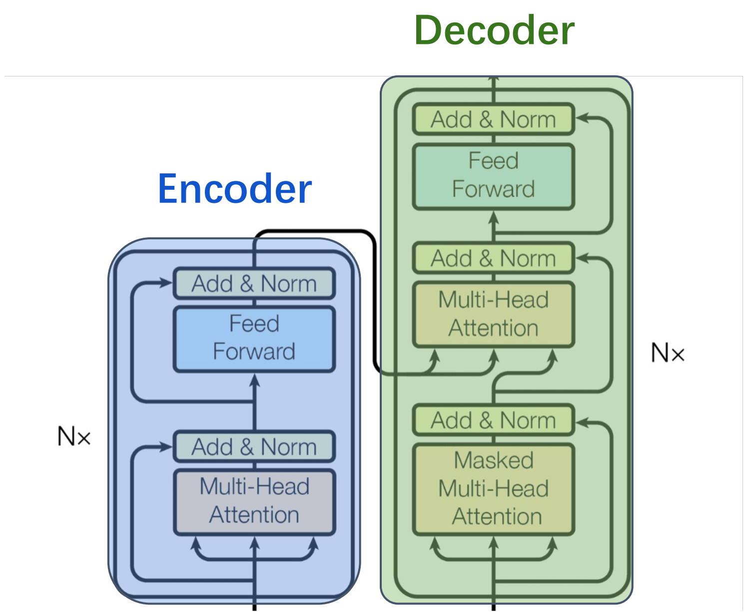
Summary

- A novel clustering view to better understand and design Transformer modules for object-centric representation learning
- We introduce CMT-DeepLab, which unfies cross-attention and panoptic segmentation from a clustering perspective

Overview

Extracting pixel-level representation:

- ❑ Vision Transformer (ViT)



Converting to object-level representation:

- ❑ CMT-DeepLab [2] (CVPR 22 ***Oral***)
- ❑ **kMaX-DeepLab [3] (ECCV 22)**

k-means Mask Transformer

Cross-attention:

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}) \times \mathbf{V}^p,$$

Yu., et al. k-means Mask Transformer. ECCV 2022.

k-means Mask Transformer

Cross-attention:

$$\hat{\mathbf{C}} = \mathbf{C} + \underset{HW}{\mathrm{softmax}}(\boxed{\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}}) \times \mathbf{V}^p,$$

Affinity logits with linear projections

k-means Mask Transformer

Cross-attention:

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\operatorname{softmax}}}\left(\boxed{\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}}\right) \times \mathbf{V}^p,$$

Affinity logits with linear projections

Spatial-wise softmax for attention weight

k-means Mask Transformer

Cross-attention:

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\mathrm{softmax}}}\left(\boxed{\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}}\right)\boxed{\times \mathbf{V}^p,}$$

Affinity logits with linear projections → Spatial-wise softmax for attention weight → Retrieve update values with linear projections

k-means Mask Transformer

Cross-attention:

$$\hat{\mathbf{C}} = \boxed{\mathbf{C}} + \boxed{\underset{HW}{\operatorname{softmax}}}\left(\boxed{\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}}\right) \boxed{\times \mathbf{V}^p},$$

Affinity logits with linear projections → Spatial-wise softmax for attention weight → Retrieve update values with linear projections → Residual update of object query

k-means Mask Transformer

k-means clustering algorithm:

$$\mathbf{A} = \underset{N}{\operatorname{argmax}}(\mathbf{C} \times \mathbf{P}^{\mathrm{T}}),$$

$$\hat{\mathbf{C}} = \mathbf{A} \times \mathbf{P},$$

k-means Mask Transformer

k-means clustering algorithm:

$$\mathbf{A} = \underset{N}{\operatorname{argmax}}(\boxed{\mathbf{C} \times \mathbf{P}^{\mathrm{T}}}),$$

$$\hat{\mathbf{C}} = \mathbf{A} \times \mathbf{P},$$

Affinity logits

k-means Mask Transformer

k-means clustering algorithm:

$$\mathbf{A} = \boxed{\underset{N}{\operatorname{argmax}}}\left(\boxed{\mathbf{C} \times \mathbf{P}^{\mathrm{T}}}\right),$$

$$\hat{\mathbf{C}} = \mathbf{A} \times \mathbf{P},$$

Affinity logits → Cluster-wise argmax for clustering assignment

k-means Mask Transformer

k-means clustering algorithm:

$$\mathbf{A} = \boxed{\underset{N}{\operatorname{argmax}}}\left(\boxed{\mathbf{C} \times \mathbf{P}^{\mathrm{T}}}\right),$$

$$\hat{\mathbf{C}} = \boxed{\mathbf{A} \times \mathbf{P},}$$

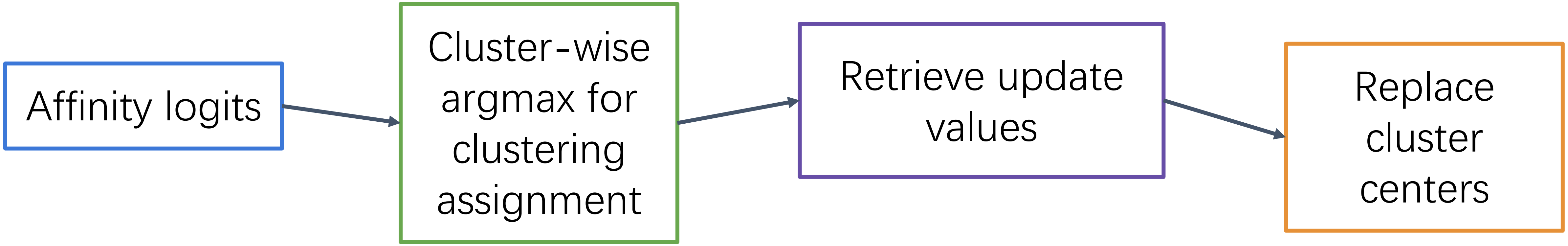
Affinity logits → Cluster-wise argmax for clustering assignment → Retrieve update values

k-means Mask Transformer

k-means clustering algorithm:

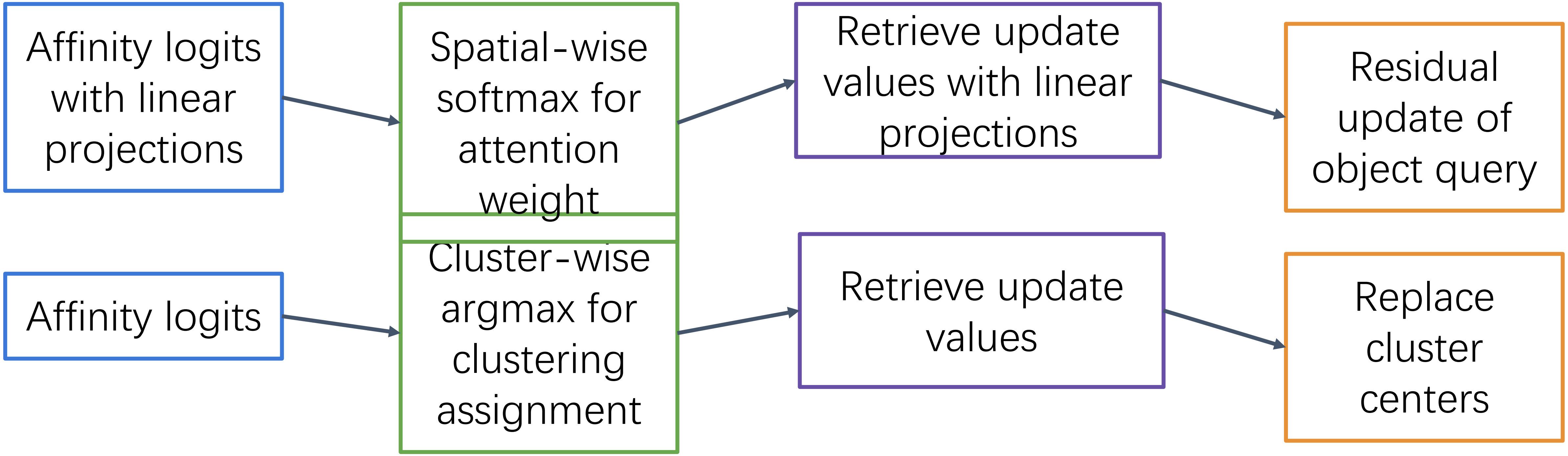
$$\mathbf{A} = \boxed{\underset{N}{\operatorname{argmax}}}\left(\boxed{\mathbf{C} \times \mathbf{P}^{\mathrm{T}}}\right),$$

$$\boxed{\hat{\mathbf{C}}} = \boxed{\mathbf{A} \times \mathbf{P},}$$



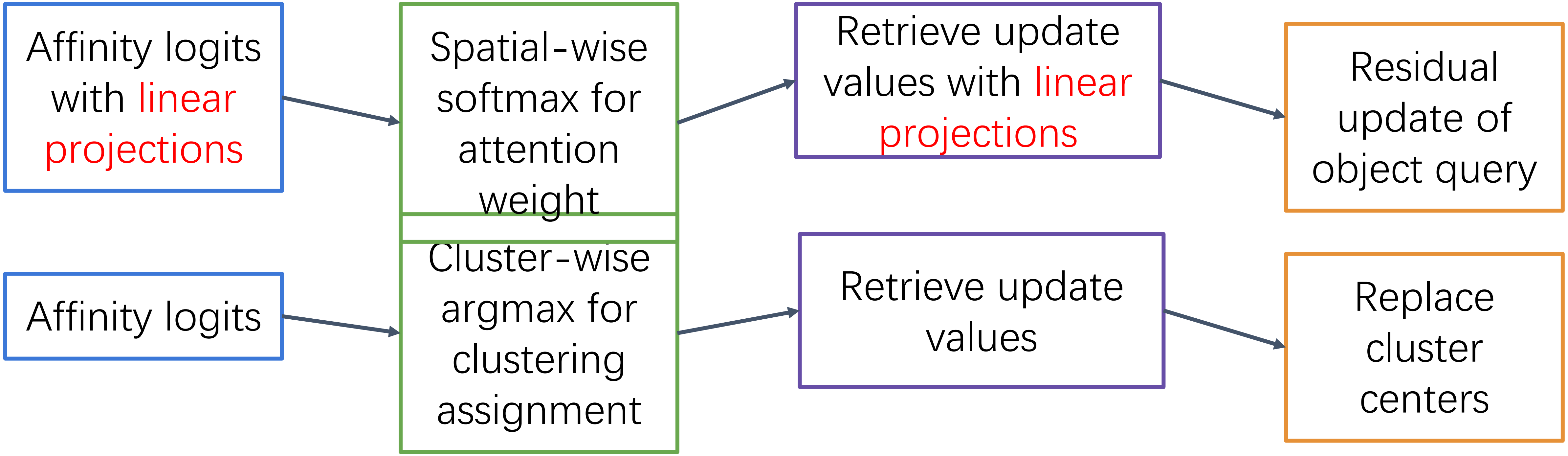
k-means Mask Transformer

cross-attention v.s. k-means clustering algorithm:



k-means Mask Transformer

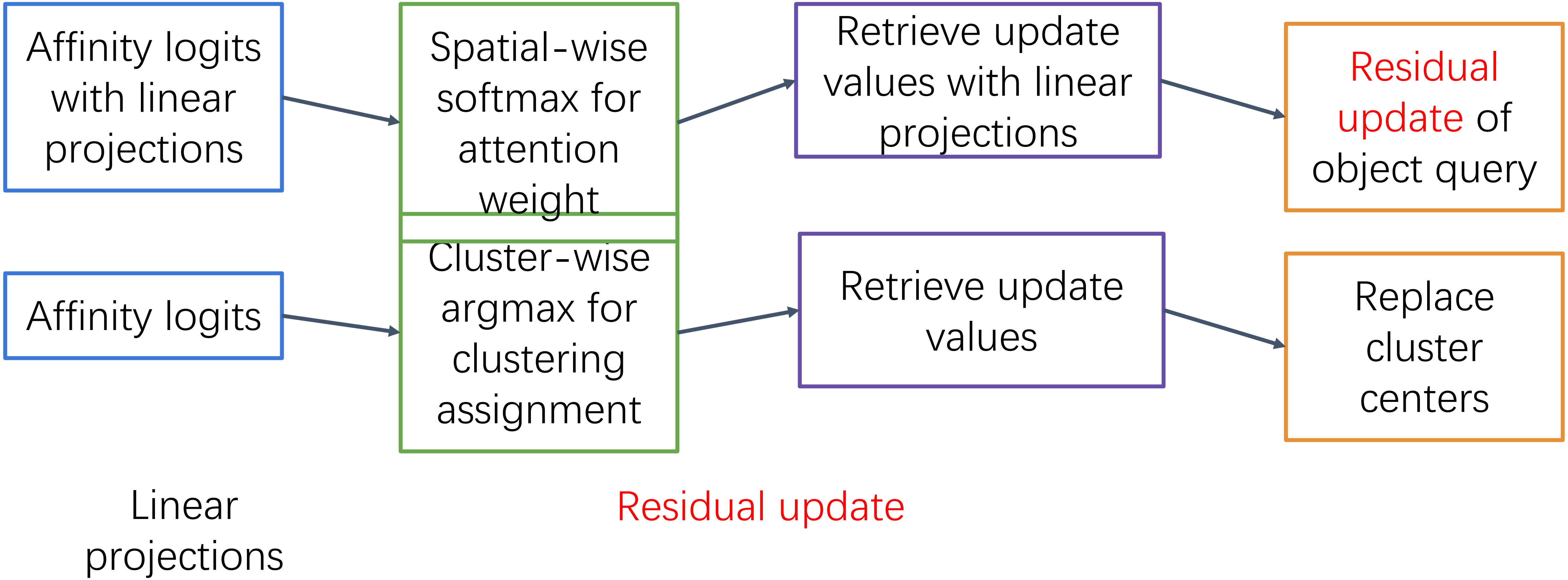
cross-attention v.s. k-means clustering algorithm:



Linear projections

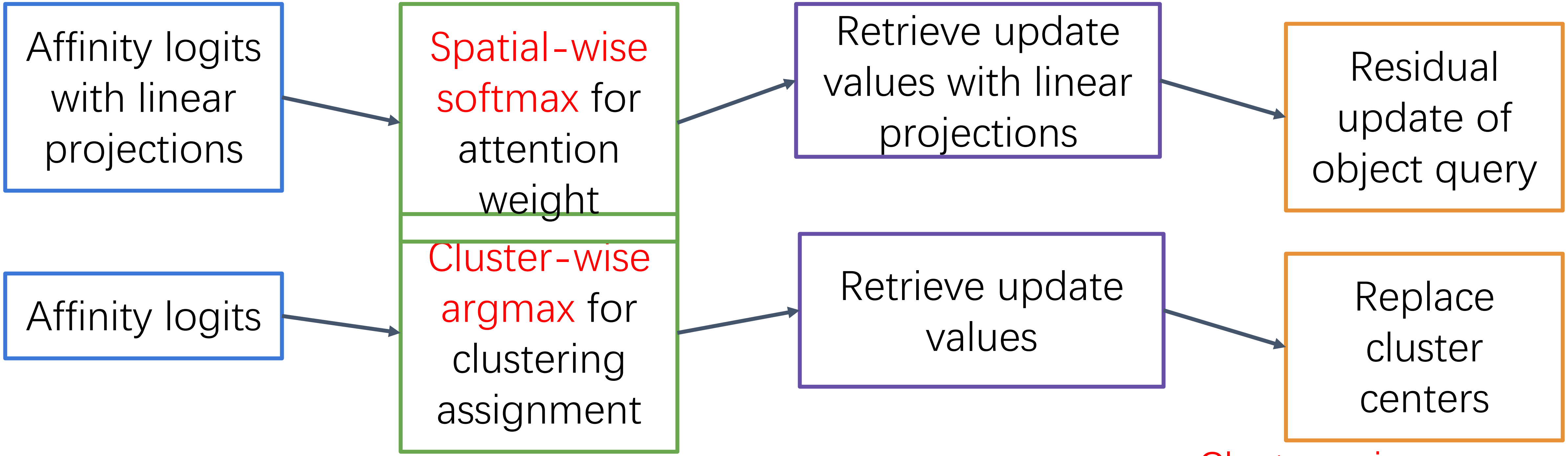
k-means Mask Transformer

Cross-attention v.s. *k-means* clustering algorithm:



k-means Mask Transformer

cross-attention v.s. k-means clustering algorithm:



Linear projections

Residual update

Cluster-wise argmax
v.s.
Spatial-wise softmax

k-means Mask Transformer

cross-attention v.s. k-means clustering algorithm:

Affinity logits with linear projections → Spatial-wise softmax for attention weight → Retrieve update values with linear projections → Residual update of object query

Affinity logits → Cluster-wise argmax for clustering assignment → Retrieve update values → Replace cluster centers

Linear projections

Residual update

Cluster-wise argmax v.s. Spatial-wise

k-means Mask Transformer

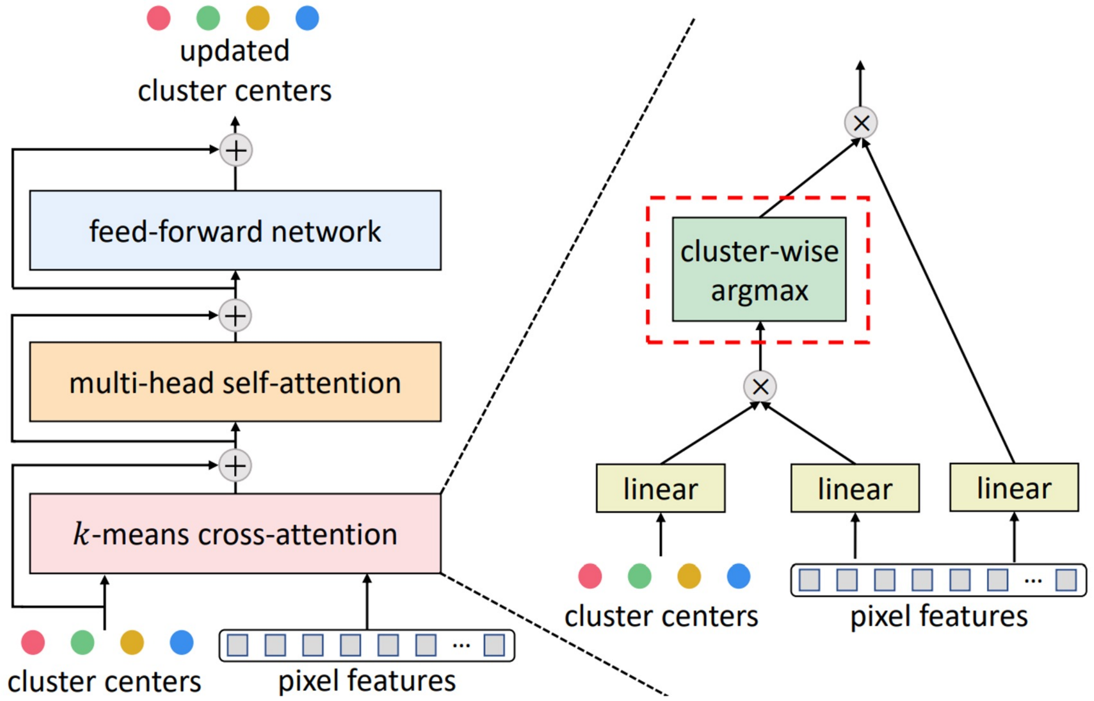
A simple ***change*** for k-means cross-attention:

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{HW}{\operatorname{softmax}}}\left(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}\right) \times \mathbf{V}^p,$$

$$\hat{\mathbf{C}} = \mathbf{C} + \boxed{\underset{N}{\operatorname{argmax}}}\left(\mathbf{Q}^c \times (\mathbf{K}^p)^{\mathrm{T}}\right) \times \mathbf{V}^p.$$

k-means Mask Transformer

A simple ***change*** for k-means cross-attention:



k-means Mask Transformer

A simple ***change*** for k-means cross-attention:

pixel-cluster interaction module	ResNet-50 params	ResNet-50 FLOPs	ResNet-50 PQ	MaX-S params	MaX-S FLOPs	MaX-S PQ
cross-attention [89]	56M	165G	47.5	73M	237G	52.0
dual-path cross-attention [92]	58M	175G	48.0	75M	247G	52.3
k-means cross-attention	57M	168G	52.7	74M	240G	56.1
dual-path k-means cross-attention	59M	176G	53.0	76M	248G	56.2

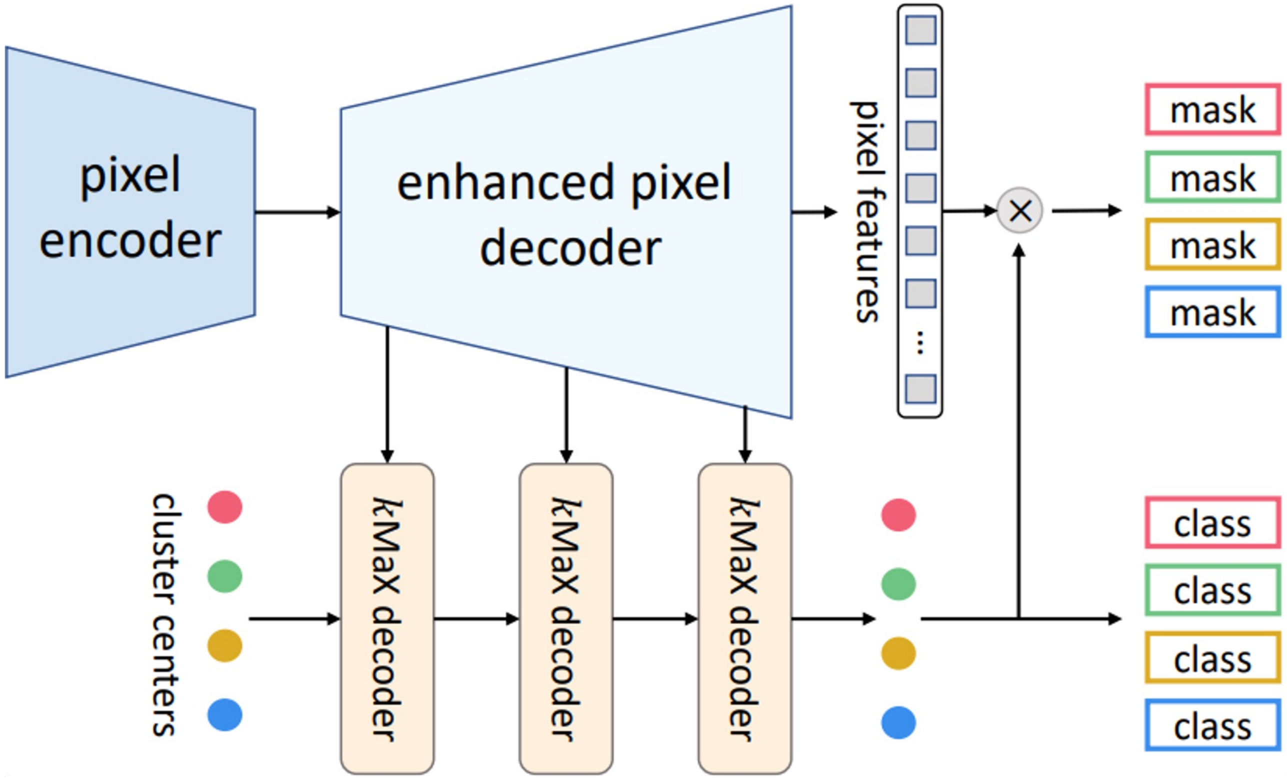
k-means Mask Transformer

A simple change for k-means cross-attention:

pixel-cluster interaction module	ResNet-50 params	ResNet-50 FLOPs	ResNet-50 PQ	MaX-S params	MaX-S FLOPs	MaX-S PQ
cross-attention [89]	56M	165G	47.5	73M	237G	52.0
dual-path cross-attention [92]	58M	175G	48.0	75M	247G	52.3
k-means cross-attention	57M	168G	52.7	74M	240G	56.1
dual-path k-means cross-attention	59M	176G	53.0	76M	248G	56.2

5.2% (47.5% -> 52.7%) PQ improvement with one change and negelectable extra cost

k-means Mask Transformer



k-means Mask Transformer

Method	params	FLOPs	FPS	PQ
MaX-DeepLab	451M	3692G	-	51.1% (-6.9%)
MaskFormer	212M	792G	5.2	52.7% (-5.3%)
K-Net	-	-	-	54.6% (-3.4%)
CMT-DeepLab	270M	1114G	3.2	55.3% (-2.7%)
kMaX-DeepLab	**232M**	**749G**	**6.6**	**58.0%**

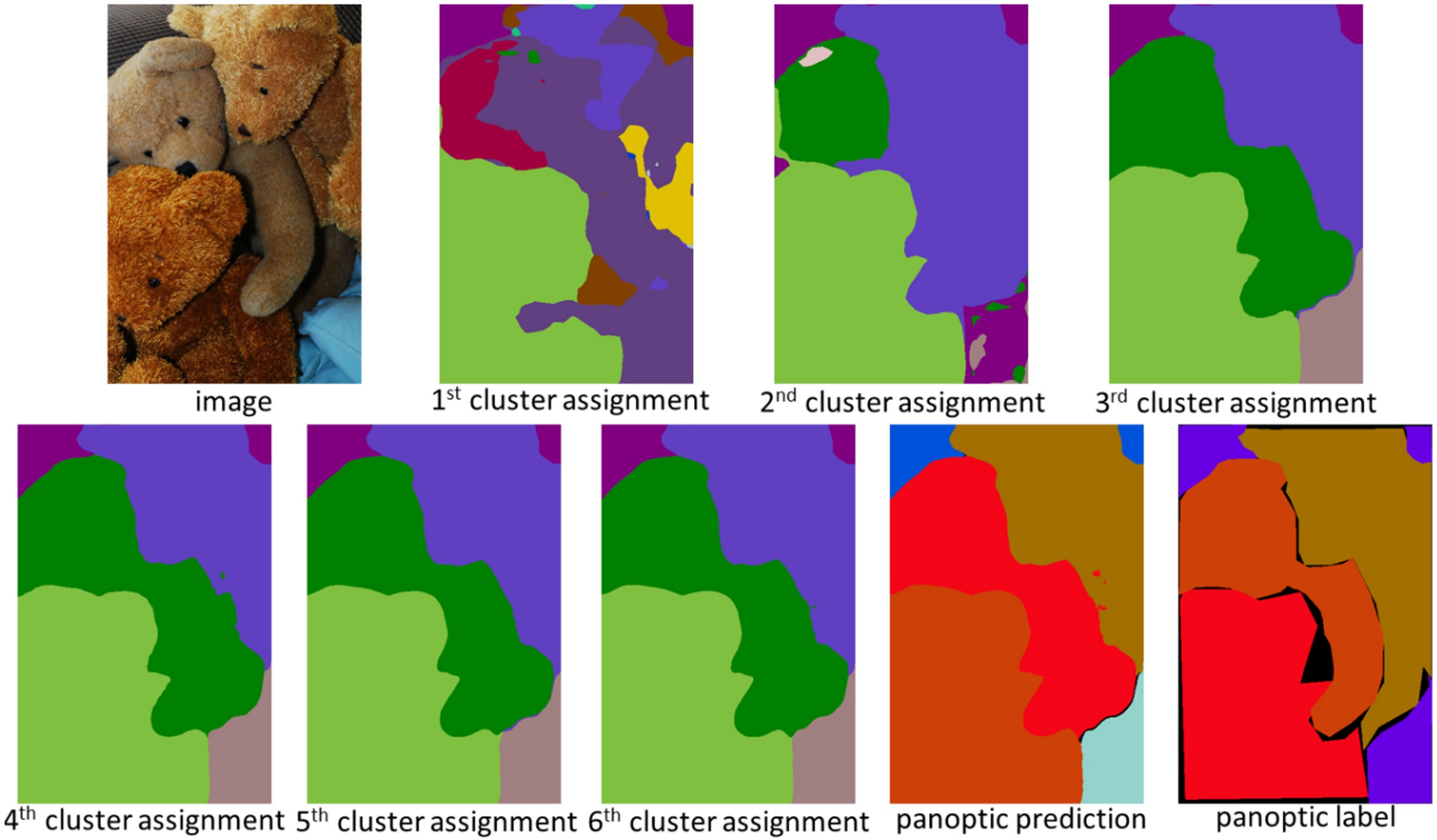
COCO *val* set

k-means Mask Transformer

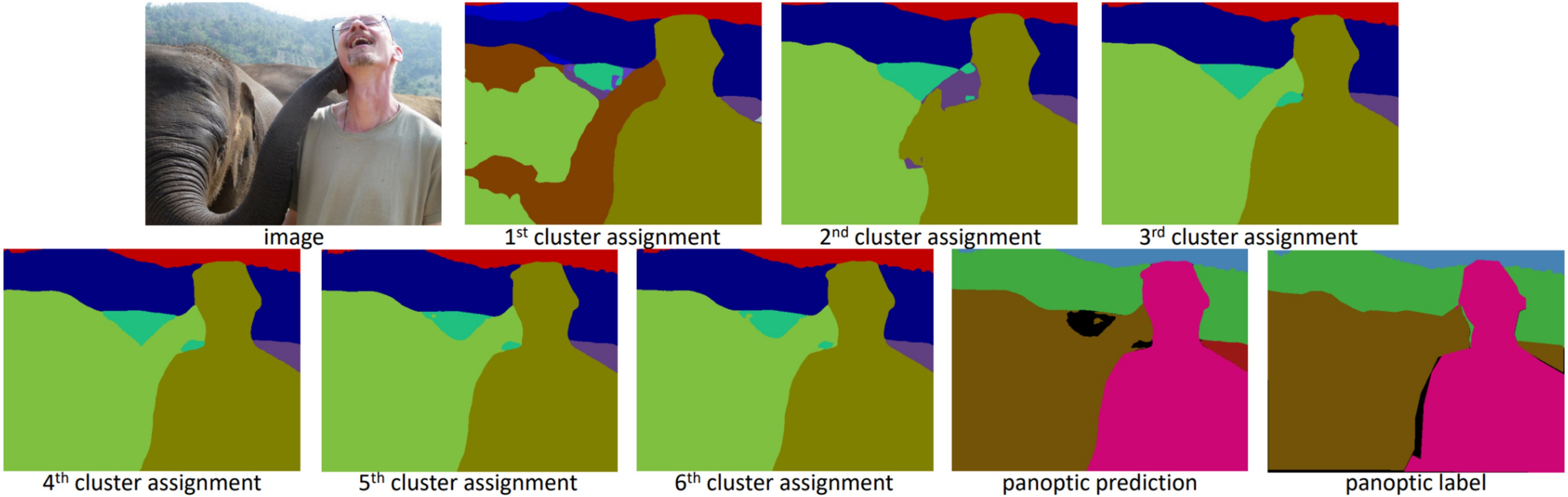
Method	params	FLOPs	FPS	PQ	AP^{mask}	mIoU
Panoptic-DeepLab	47M	548G	5.7	63.0% (-5.4%)	35.3% (-8.7%)	80.5% (-3.0%)
Axial-DeepLab	173M	2447G	-	64.4% (-4.0%)	36.7% (-7.3%)	80.6% (-2.9%)
SWideRNet	536M	10365G	1.0	66.4% (-2.0%)	40.1% (-3.9%)	82.2% (-1.3%)
kMaX-DeepLab	**232M**	**1673G**	**3.1**	**68.4%**	**44.0%**	**83.5%**

Cityscapes *val* set

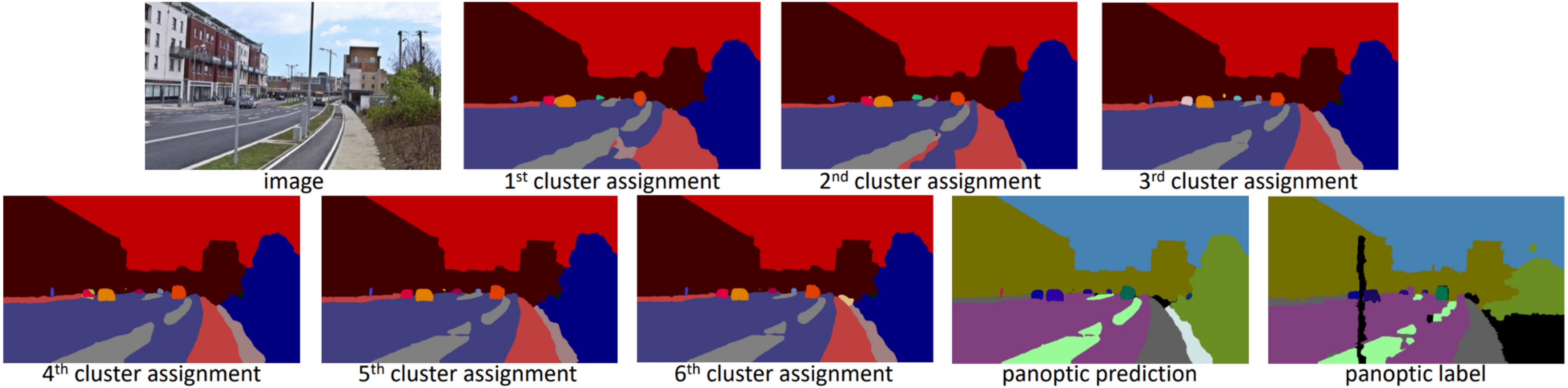
k-means Mask Transformer



k-means Mask Transformer



k-means Mask Transformer



Summary

- Discuss the unlerlying similarity between ***cross-attention*** and ***k-means clustering*** algorithm.

- Propose ***k-means cross-attention***, which designs cross-attention as a k-means clustering module, leading to better ***object-centric representation.***

- A simple change on ***activation function*** with ***SOTA*** performance.