

# Cost Aggregation with 4D Convolutional Swin Transformer for Few-Shot Segmentation

Sunghwan Hong<sup>1,\*</sup>, Seokju Cho<sup>1,\*</sup>, Jisu Nam<sup>1</sup>, Stephen Lin<sup>2</sup>, Seungryong Kim<sup>1</sup> (\* Equal Contribution)

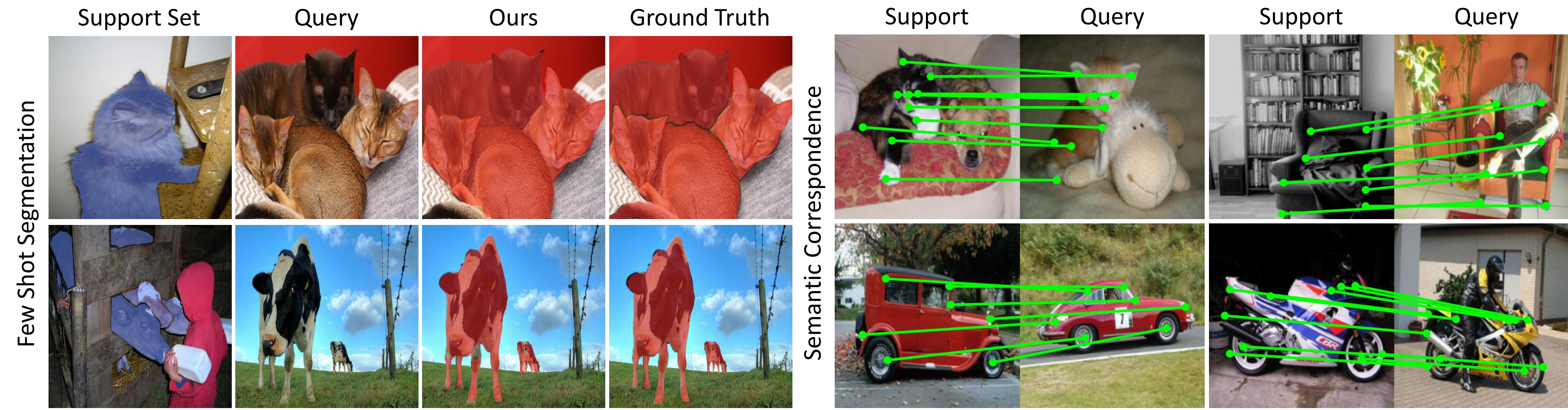
<sup>1</sup>Korea University, Seoul, Korea <sup>2</sup>Microsoft Research Asia, Beijing, China



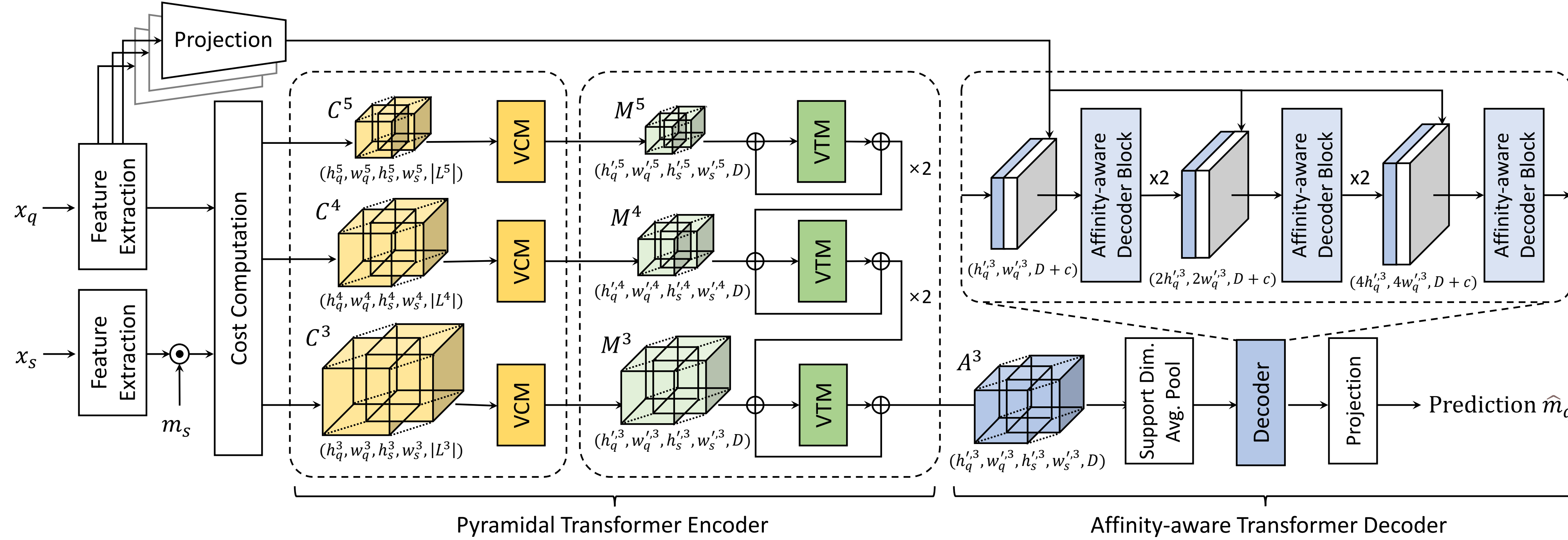
This paper presents a novel cost aggregation network, called **Volumetric Aggregation with Transformers (VAT)**, for few-shot segmentation.

- The use of transformers can benefit correlation map aggregation through self-attention over a global receptive field.
- However, the tokenization of a correlation map for transformer processing can be detrimental, because the discontinuity at token boundaries reduces the local context available near the token edges and decreases inductive bias.
- To address this problem, **we propose a 4D Convolutional Swin Transformer**, where a high-dimensional Swin Transformer is preceded by a series of small-kernel convolutions that impart local context to all pixels and introduce convolutional inductive bias.
- We additionally boost aggregation performance by **applying transformers within a pyramidal structure**, where aggregation at a coarser level guides aggregation at a finer level.
- Noise in the transformer output is then filtered in **the subsequent decoder with the help of the query's appearance embedding**.

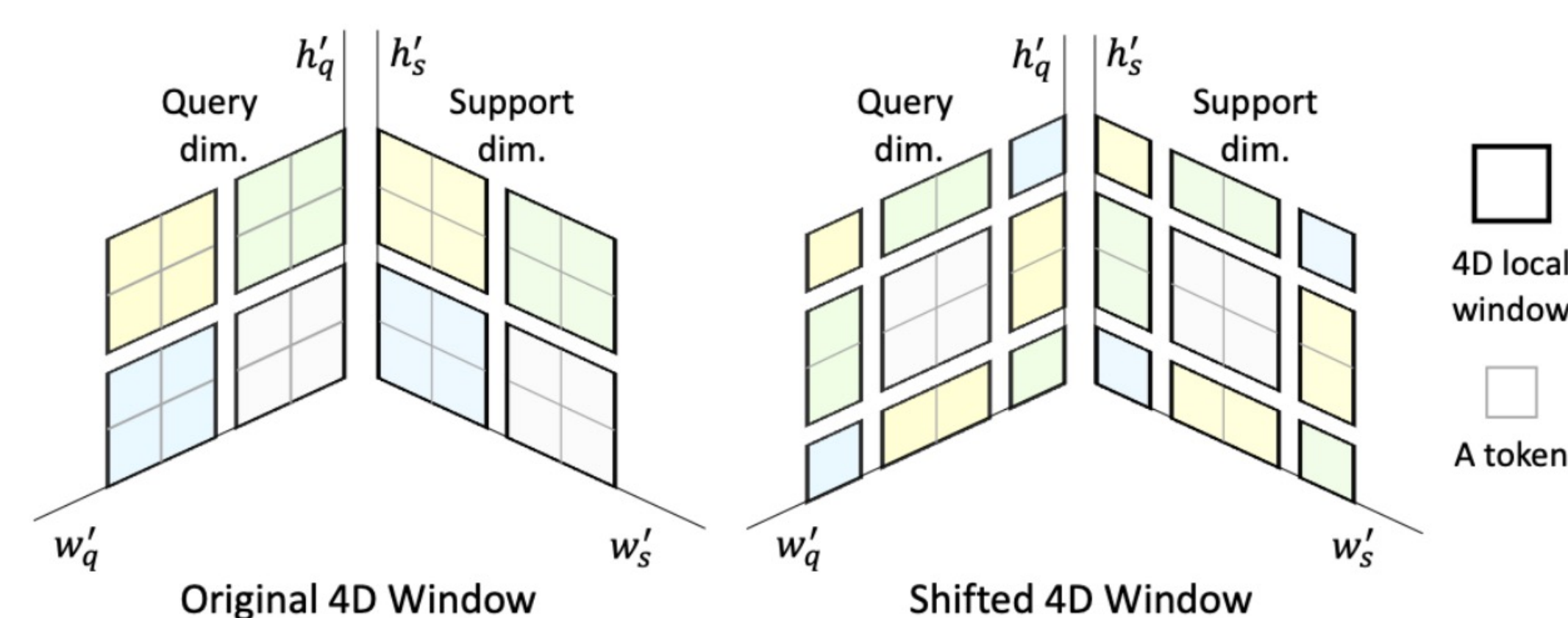
With this model, a new state-of-the-art is set for all the standard benchmarks in few-shot segmentation. It is shown that VAT attains state-of-the-art performance for semantic correspondence as well, where cost aggregation also plays a central role.



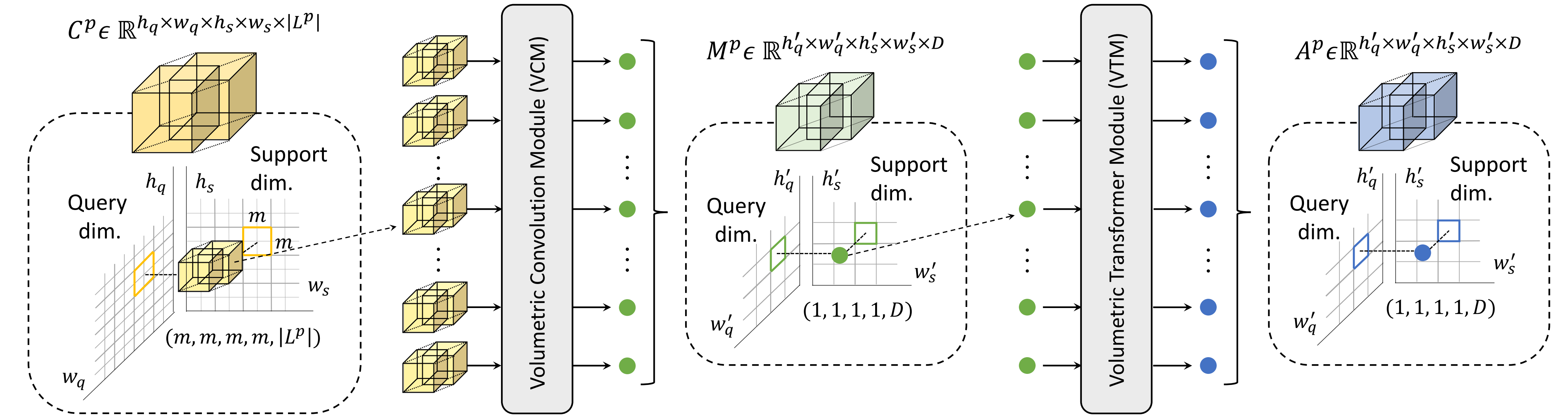
**Fig. 1: Our VAT reformulates few-shot segmentation as semantic correspondence.** VAT sets a new state-of-the-art in few-shot segmentation, and attains state-of-the-art performance for semantic correspondence as well.



**Fig. 2: Overall network architecture.** Our network consists of feature extraction and cost computation, a pyramidal transformer encoder, and an affinity-aware transformer decoder.



**Fig. 3: Illustration of shifted 4D windows in VTM.** It computes self-attention within the partitioned windows and considers inter-window interactions by shifting the windows.



**Fig. 4: Overview of 4D Convolutional Swin Transformer.** We replace the VEM with VCM and the output undergoes VTM for cost aggregation.

Backbone network	Methods	1-shot							5-shot							# learnable params
		5 <sup>0</sup>	5 <sup>1</sup>	5 <sup>2</sup>	5 <sup>3</sup>	mIoU	FB-IoU	mBA	5 <sup>0</sup>	5 <sup>1</sup>	5 <sup>2</sup>	5 <sup>3</sup>	mIoU	FB-IoU	mBA	
ResNet50	HSNet	64.3	70.7	60.3	<b>60.5</b>	<b>64.0</b>	<b>76.7</b>	<b>53.9</b>	70.3	73.2	67.4	<b>67.1</b>	69.5	<b>80.6</b>	<b>54.5</b>	<b>2.6M</b>
	CyCTR	<b>65.7</b>	<b>71.0</b>	59.5	59.7	<b>64.0</b>	-	-	<b>69.3</b>	<b>73.5</b>	63.8	63.5	67.5	-	-	-
	VAT (ours)	<b>67.6</b>	<b>72.0</b>	<b>62.3</b>	<b>60.1</b>	<b>65.5</b>	<b>77.8</b>	<b>54.4</b>	<b>72.4</b>	<b>73.6</b>	<b>68.6</b>	<b>65.7</b>	<b>70.1</b>	<b>80.9</b>	<b>54.8</b>	<b>3.2M</b>
ResNet101	HSNet	67.3	72.3	<b>62.0</b>	<b>63.1</b>	<b>66.2</b>	<b>77.6</b>	<b>53.9</b>	71.8	74.4	67.0	<b>68.3</b>	<b>70.4</b>	<b>80.6</b>	<b>54.4</b>	<b>2.6M</b>
	CyCTR	<b>67.2</b>	<b>71.1</b>	57.6	59.0	63.7	73.0	-	<b>71.0</b>	75.0	58.5	65.0	67.4	75.4	-	-
	VAT (ours)	<b>70.0</b>	<b>72.5</b>	<b>64.8</b>	<b>64.2</b>	<b>67.9</b>	<b>79.6</b>	<b>54.7</b>	<b>75.0</b>	<b>75.2</b>	<b>68.4</b>	<b>69.5</b>	<b>72.0</b>	<b>83.2</b>	<b>54.8</b>	<b>3.3M</b>

**Table 1: Performance comparison on PASCAL-5<sup>i</sup>.**

Backbone feature	Methods	1-shot						5-shot					
		20 <sup>0</sup>	20 <sup>1</sup>	20 <sup>2</sup>	20 <sup>3</sup>	mean	FB-IoU	mBA	20 <sup>0</sup>	20 <sup>1</sup>	20 <sup>2</sup>	20 <sup>3</sup>	mean
ResNet50	HSNet	36.3	<b>43.1</b>	38.7	38.7	39.2	<b>68.2</b>	<b>53.0</b>	43.3	<b>51.3</b>	48.2	45.0	46.9
	CyCTR	<b>38.9</b>	43.0	<b>39.6</b>	<b>39.8</b>	<b>40.3</b>	-	-	41.1	48.9	45.2	<b>47.0</b>	45.6
	VAT (ours)	<b>39.0</b>	<b>43.8</b>	<b>42.6</b>	<b>39.7</b>	<b>41.3</b>	<b>68.8</b>	<b>54.2</b>	<b>44.1</b>	<b>51.1</b>	<b>50.2</b>	<b>46.1</b>	<b>47.9</b>

**Table 2: Performance comparison on COCO-20<sup>i</sup>.**

Methods	F.T. Feat.	Data Aug.	Cost Aggregation	SPair-71k PCK @ $\alpha_{\text{bbox}}$				PF-PASCAL PCK @ $\alpha_{\text{img}}$				PF-WILLOW PCK @ $\alpha_{\text{bbox}}$		
				0.03	0.05	0.1	0.15	0.03	0.05	0.1	0.15	0.05	0.1	0.15
CATs	✓	✗	Transformer	10.2	21.6	43.5	55.0	41.6	67.5	89.1	94.9	46.6	75.6	87.5
	✓	✓	Transformer	13.8	27.7	49.9	<b>61.7</b>	<b>49.9</b>	<b>75.4</b>	<b>92.6</b>	<b>96.4</b>	<b>50.3</b>	<b>79.2</b>	90.3
VAT	✓	✗	Transformer	14.9	28.3	48.4	59.1	54.6	72.9	91.1	95.6	46.0	78.8	91.3
	✓	✓	Transformer	<b>19.6</b>	<b>35.0</b>	<b>55.5</b>	<b>65.1</b>	<b>62.7</b>	<b>78.2</b>	<b>92.3</b>	<b>96.2</b>	<b>52.8</b>	<b>81.6</b>	<b>91.4</b>

**Table 3: Quantitative results on SPair-71k, PF-PASCAL and PF-WILLOW.**

