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

Sunghwan Hong[1,*], Seokju Cho[1,*], Jisu Nam[1], Stephen Lin[2], Seungryong Kim[1] (* Equal Contribution)
[1]Korea University, Seoul, Korea [2]Microsoft Research Asia, Beijing, China

ECCV TEL AVIV 2022

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.

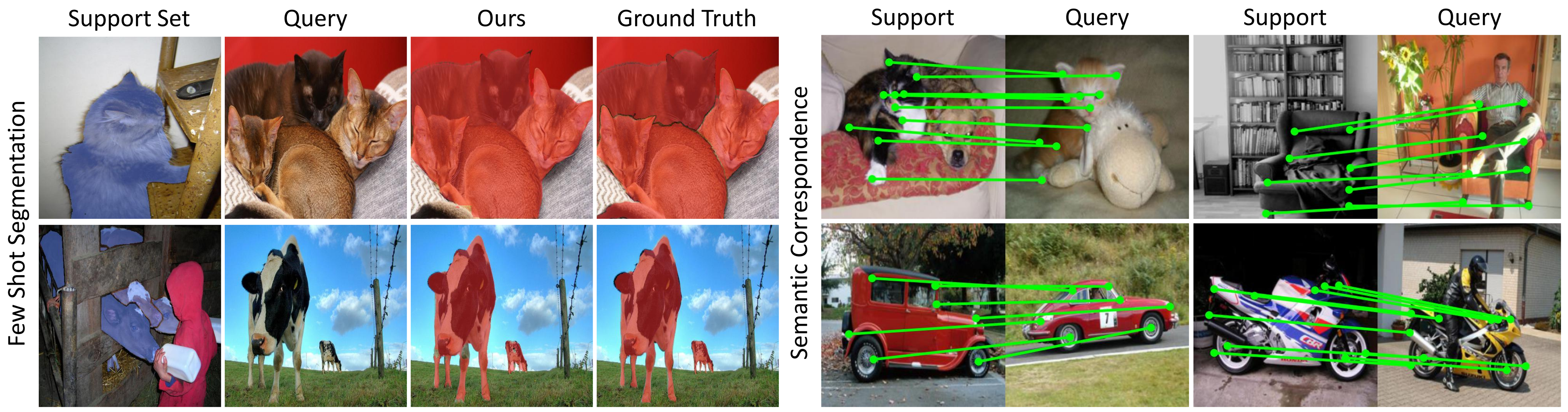

Fig. 1: Our VAT reformulates few-shot segmentation as semantic correspondence. VAT sets a new state-of-the-art in few-shot segmentation, and attains state-of-the-art performance for semantic correspondence as well.

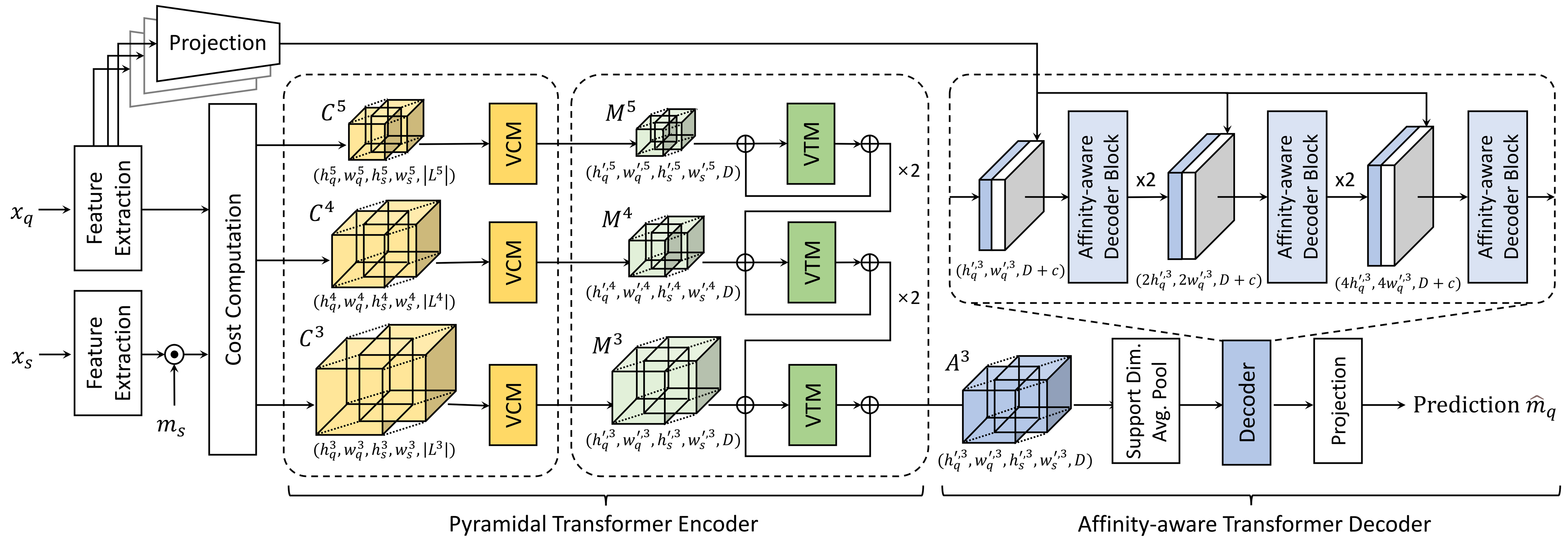

Fig. 2: Overall network architecture. Our network consists of feature extraction and cost computation, a pyramidal transformer encoder, and an affinity-aware transformer decoder.

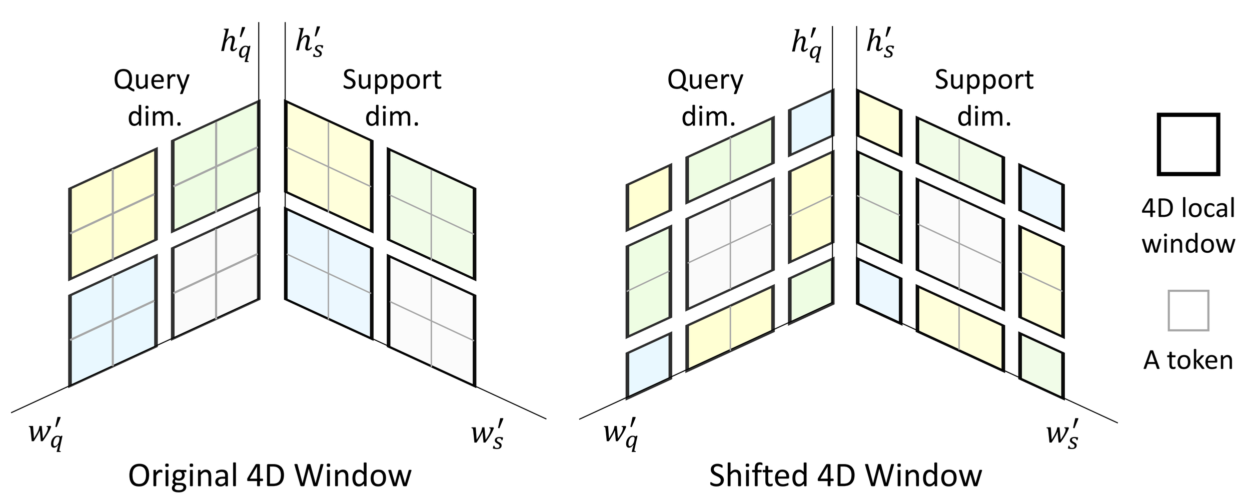

Fig. 3: Illustration of shifted 4D windows in VTM. It computes self-attention within the partitioned windows and considers inter-window interactions by shifting the windows.

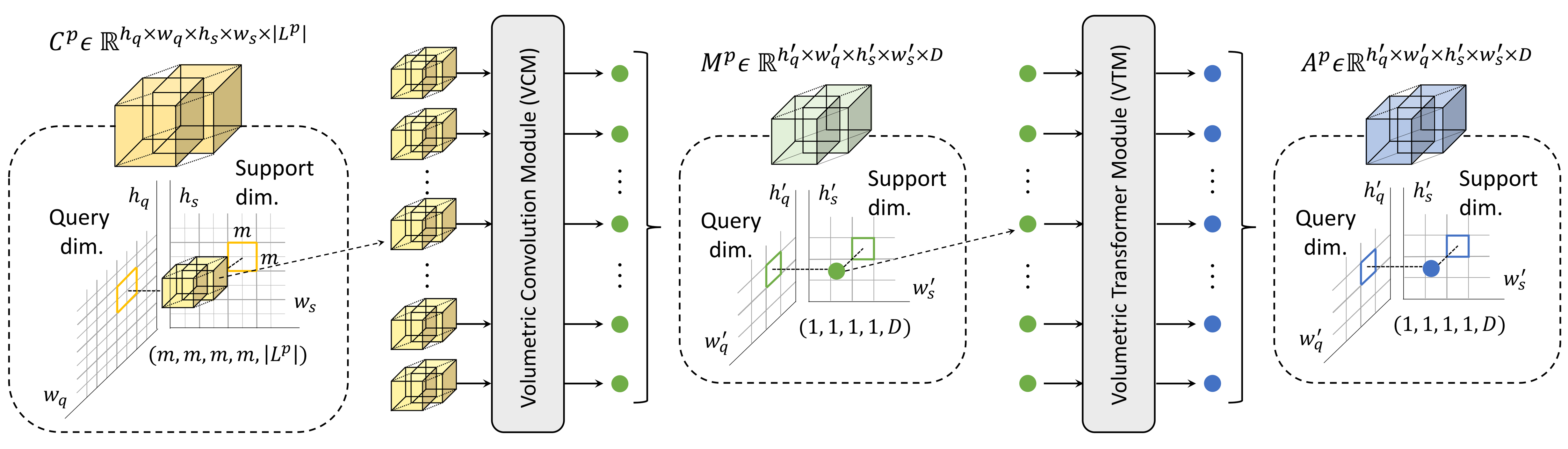

Fig. 4: Overview of 4D Convolutional Swin Transformer. We replace the VEM with VCM and the output undergoes VTM for cost aggregation.

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

Table 1: Performance comparison on **PASCAL-5^i.**

Backbone feature	Methods	1-shot 20^0	20^1	20^2	20^3	mean	FB-IoU	mBA	5-shot 20^0	20^1	20^2	20^3	mean	FB-IoU	mBA
ResNet50	HSNet	36.3	43.1	38.7	38.7	39.2	68.2	53.0	43.3	**51.3**	48.2	45.0	46.9	70.7	53.8
	CyCTR	38.9	43.0	39.6	**39.8**	40.3	-	-	41.1	48.9	45.2	**47.0**	45.6	-	-
	VAT (ours)	**39.0**	**43.8**	**42.6**	39.7	**41.3**	**68.8**	**54.2**	**44.1**	51.1	**50.2**	46.1	**47.9**	**72.4**	**54.9**

Table 2: Performance comparison on **COCO-20^i.**

Methods	F.T. Feat.	Data Aug.	Cost Aggregation	SPair-71k PCK @ α_{bbox} 0.03	0.05	0.1	0.15	PF-PASCAL PCK @ α_{img} 0.03	0.05	0.1	0.15	PF-WILLOW PCK @ α_{bbox} 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	61.7	49.9	75.4	**92.6**	**96.4**	50.3	79.2	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	**19.6**	**35.0**	**55.5**	**65.1**	**62.7**	**78.2**	92.3	96.2	**52.8**	**81.6**	**91.4**

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