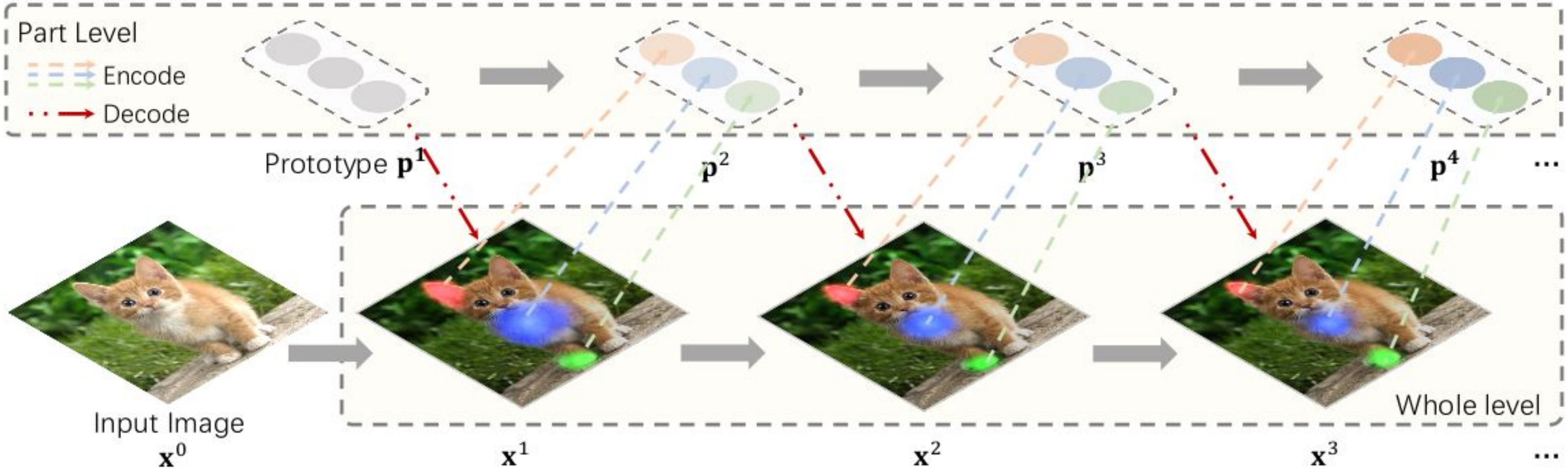
Visual Parser (ViP): Representing Part-Whole Relations with Transformers

Presenter: Mert Kilickaya

Visual Parser: Overview

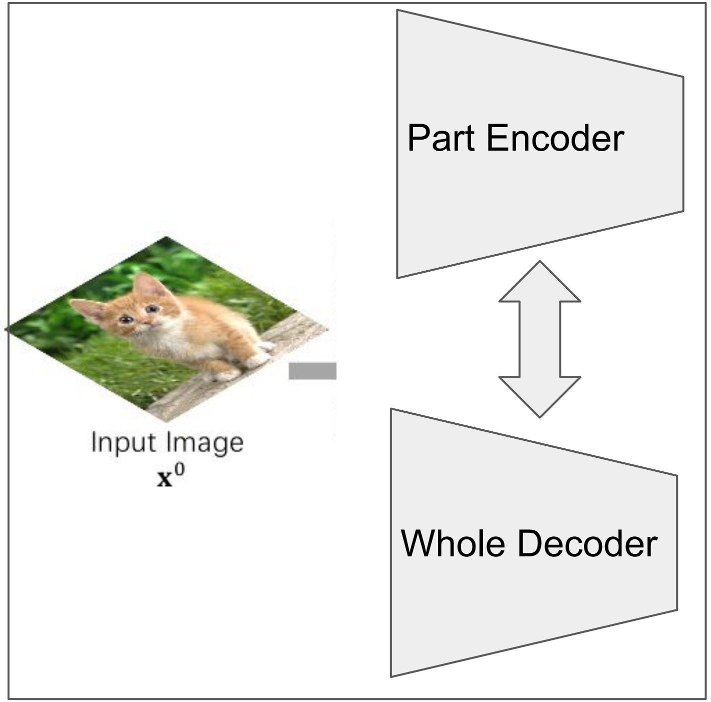


ViT: Represents the parts (local patches) only

CNN: Represents the whole (global feature map) only

This work: Combines parts and whole to improve discriminative ability

Visual Parser: Overview



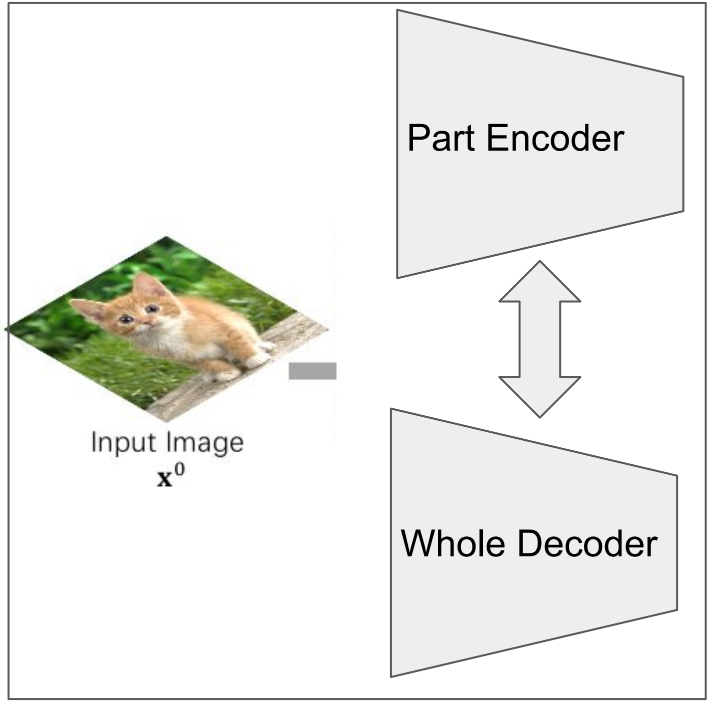
Part Encoder: Encodes N parts into C dimensional features

Whole Encoder: Encodes L pixels into C dimensional features

PE attends on pixels for feature extraction for each patch

WE in return decodes each part representation to pixels

Visual Parser: Part Encoder



Part Feature Extraction: from whole feature map

Part-to-Part Encoding: each part is a function of all others

Part Selection: Soft attention on part-level for discrimination

Visual Parser: Part Encoder: Feature Extraction

Part Encoder attends on global feature map pixels for feature extraction

Query: Part embeddings (N x C) | **Key, Value**: Whole feature map (L x C) | L = width x height

whole + position

$$\hat{\mathbf{p}}^{i-1} = \mathbf{p}^{i-1} + \text{Attention}(\mathbf{p}^{i-1} + \mathbf{d}_e, \mathbf{x}^{i-1} + \mathbf{d}_w, \mathbf{x}^{i-1}),$$

part + position

whole

M: Query x Key: (N x L) dimensional matrix that assigns per-pixel importance for each part

d_e: plays a key role in M, by forcing each part to focus on different regions (parts)

Visual Parser: Part Encoder: Part-to-Part Encoding & Selection

Part-to-Part Encoding: Parts also interact with each other with a simple learned (N x N) matrix W_p

$$\hat{\mathbf{p}}_r^{i-1} = \hat{\mathbf{p}}^{i-1} + \mathbf{W}_p \cdot \mathrm{LN}(\hat{\mathbf{p}}^{i-1}), \tag{5}$$

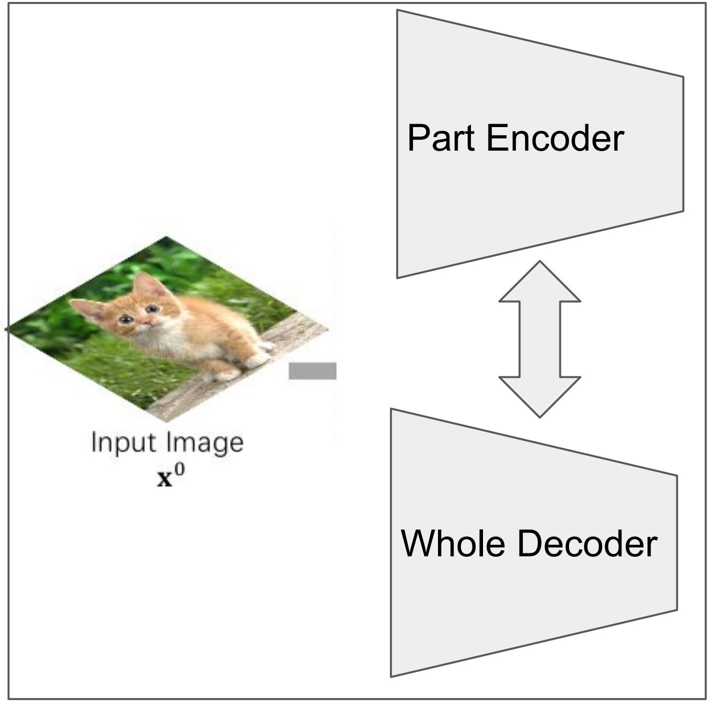
Part Selection: Not all parts matter for each image and objects

Part Selection: The authors learn a simple soft (sigmoid) attention to re-weigh each part feature

Activating the part representations. The part representation learnt above may not be all meaningful since different objects may have different numbers of parts describing themselves. We thereby further apply a Multi-Layer Perceptron (MLP) that has two linear mappings with weight $\mathbf{W}_{f1}, \mathbf{W}_{f2} \in \mathbb{R}^{C\times C}$ and an activation function (GELU [26]) $\sigma(\cdot)$ in its module. The activation function will only keep the useful parts to be active, while those identified to be less helpful will be squashed. In this way, we obtain the part representation $\mathbf{p}^i$ for block i by:

$$\begin{aligned}\mathbf{p}^i &= \hat{\mathbf{p}}_r^{i-1} + \mathrm{MLP}(\hat{\mathbf{p}}_r^{i-1}), \\ \mathrm{MLP}(\hat{\mathbf{p}}_r^{i-1}) &= \sigma(\mathrm{LN}(\hat{\mathbf{p}}_r^{i-1}) \cdot \mathbf{W}_{f1}) \cdot \mathbf{W}_{f2}.\end{aligned} \tag{6}$$

Visual Parser: Whole Decoder



Part-to-Whole Interaction

Each pixel attends to all the other parts

Patch-based Local Attention

Each local patch attends to all other patches

Visual Parser: Whole Decoder

Part-to-Whole Interaction

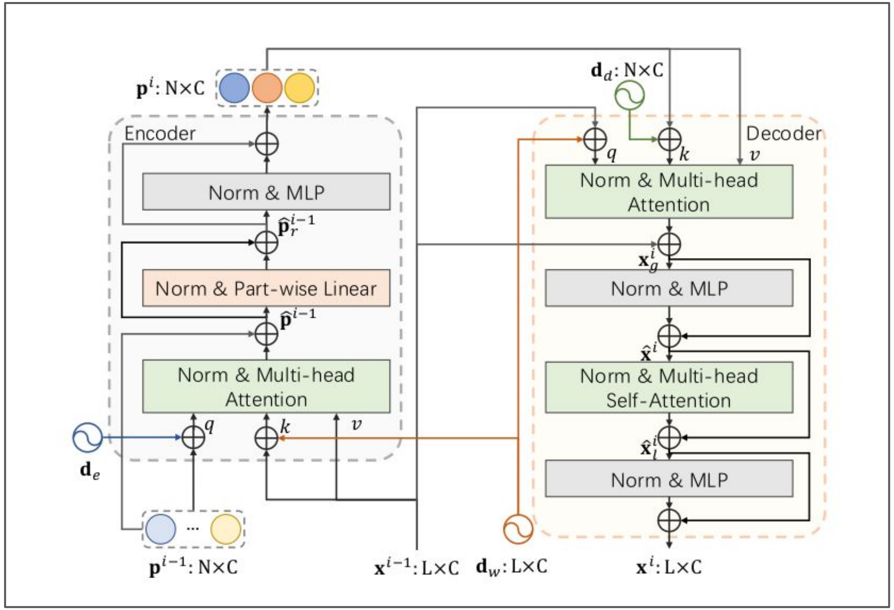
Query: Whole map (L x C) | **Key, Value**: Part embeddings (N x C) | L = width x height

$$\mathbf{x}_g^i = \mathbf{x}^{i-1} + \text{Attention}(\mathbf{x}^{i-1} + \mathbf{d}_w, \mathbf{p}^i + \mathbf{d}_d, \mathbf{p}^i),$$
$$\hat{\mathbf{x}}^i = \mathbf{x}_g^i + \text{MLP}(\mathbf{x}_g^i),$$

Patch-based Local Attention

$$\hat{\mathbf{x}}_t^i = \mathbf{x}_t^i + \text{Attention}(\mathbf{x}_t^i, \mathbf{x}_t^i + \mathbf{r}^i, \mathbf{x}_t^i),$$
$$\hat{\mathbf{x}}_l^i = \{\hat{\mathbf{x}}_1^i, ..., \hat{\mathbf{x}}_t^i, ..., \hat{\mathbf{x}}_{N_p}^i\},$$
$$\mathbf{x}^i = \hat{\mathbf{x}}_l^i + \text{MLP}(\hat{\mathbf{x}}_l^i),$$

Visual Parser: Summary



Part-to-Whole Multi-head attention

Part-to-Part Self attention (with selection)

Whole-to-Part Multi-head attention

Whole-to-Whole (local patch) Self attention

Analysis-1: Faster and better than HaloNet (ImageNet SOTA)

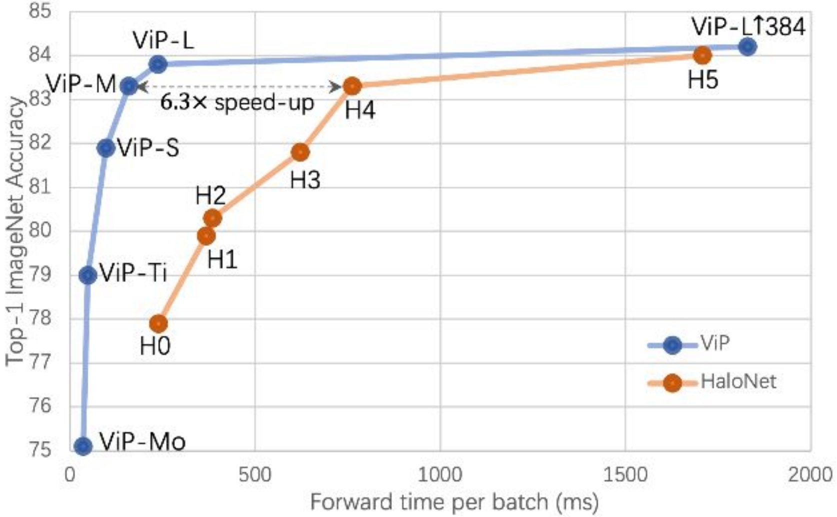


Figure 4: Speed-Accuracy comparison with HaloNet.

Analysis-2

	N	FLOPS (G)	Top-1 (%)
ViP-Ti	8	1.6	77.6
	16	1.6	78.1
	32	1.7	79.0
	64	1.8	79.1

Table 3: Effect of number of parts for ViP-Ti.

Scales to a higher number of parts (i.e. 64)

	Predict on parts	Predict on wholes	Top-1 (%)
ViP-Ti	✓		**79.0**
		✓	78.3
ViP-S	✓		**81.9**
		✓	81.5
ViP-M	✓		82.7
		✓	**83.3**

Table 4: Effect of predicting on part/whole level.

Parts are more discriminative than whole (mostly)

ImageNet and MS-COCO

Model	Input Size	Params (M)	FLOPS (G)	Top-1 Acc (%)
CNN Architectures				
BoTNet-T3 [54]	224^2	33.5	7.3	81.7
BoTNet-T4 [54]	224^2	54.7	10.9	82.8
BoTNet-T5 [54]	256^2	75.1	19.3	83.5
RegNetY-4G [49]	224^2	20.6	4.0	80.0
RegNetY-8G [49]	224^2	39.2	8.0	81.7
RegNetY-16G [49]	224^2	83.6	15.9	82.9
Transformer Architectures				
ViT-B [18]	384^2	86.4	55.4	77.9
ViT-L [59]	384^2	307	190.7	76.5
DeiT-Ti [59]	224^2	5.7	1.6	72.2
DeiT-S [59]	224^2	22.1	4.6	79.8
DeiT-B [59]	224^2	86.6	17.6	81.8
DeiT-B↑384 [59]	384^2	86.6	55.4	83.1
PVT-Tiny [64]	224^2	13.2	1.9	75.1
PVT-Small [64]	224^2	24.5	3.8	79.8
PVT-Medium [64]	224^2	44.2	6.7	81.2
PVT-Large [64]	224^2	61.4	9.8	81.7
T2T-ViT-14 [71]	224^2	21.5	5.2	81.5
T2T-ViT-19 [71]	224^2	39.2	8.9	81.9
T2T-ViT-24 [71]	224^2	64.1	14.1	82.3
TNT-S [23]	224^2	23.8	5.2	81.5
TNT-B [23]	224^2	65.6	14.1	82.9
Swin-T [46]	224^2	29	4.5	81.3
Swin-S [46]	224^2	50	8.7	83.0
Swin-B [46]	224^2	88	15.4	83.3
ViP-Mo	224^2	5.3	0.8	75.1
ViP-Ti	224^2	12.8	1.7	79.0
ViP-S	224^2	32.1	4.5	81.9
ViP-M	224^2	49.6	8.0	83.3
ViP-B	224^2	87.8	15.0	83.8
ViP-B↑384	384^2	87.8	39.1	84.2

Table 2: Results on ImageNet-1K.

Backbone	RetinaNet 1×						RetinaNet 3×						Params (M)	FLOPS (G)
	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L		
ViP-Mo	**36.5**	56.7	38.6	23.4	39.7	48.4	**39.2**	59.7	41.4	25.5	42.3	51.7	5.3 (15.2)	15 (166)
R18 [44]	31.8	49.6	33.6	16.3	34.3	43.2	35.4	53.9	37.6	19.5	38.2	46.8	11.0 (21.3)	37 (189)
PVT-T [64]	36.7(+4.9)	56.9	38.9	22.6	38.8	50.0	39.4(+4.0)	59.8	42.0	25.5	42.0	52.1	12.3 (23.0)	70 (221)
ViP-Ti	**39.7**(+7.9)	60.6	42.2	23.9	42.9	53.0	**41.6**(+6.2)	62.6	44.0	27.2	45.1	54.2	11.2 (21.4)	29 (181)
R50 [44]	36.5	55.4	39.1	20.4	40.3	48.1	39.0	58.4	41.8	22.4	42.8	51.6	23.3 (37.7)	84 (239)
PVT-S [64]	40.4(+3.9)	61.3	43.0	25.0	42.9	55.7	42.2(+3.2)	62.7	45.0	26.2	45.2	57.2	23.6 (34.2)	134 (286)
ViP-S	**43.0**(+6.5)	64.0	45.9	28.9	46.7	56.3	**44.0**(+5.0)	65.1	47.2	28.8	47.3	57.2	29.0 (39.9)	75 (227)
R101 [44]	38.5	57.8	41.2	21.4	42.6	51.1	40.9	60.1	44.0	23.7	45.0	53.8	42.3 (56.7)	160 (315)
X101-32 [44]	39.9(+1.4)	59.6	42.7	22.3	44.2	52.5	41.4(+0.5)	61.0	44.3	23.9	45.5	53.7	41.9 (56.4)	164 (319)
PVT-M [64]	41.9(+3.4)	63.1	44.3	25.0	44.9	57.6	43.2(+2.3)	63.8	46.1	27.3	46.3	58.9	43.7 (54.3)	222 (374)
ViP-M	**44.3**(+5.8)	65.9	47.4	30.7	48.0	57.9	**45.3**(+4.4)	66.4	48.5	29.7	48.6	59.3	48.8 (59.8)	135 (287)
X101-64 [44]	41.0	60.9	44.0	23.9	45.2	54.0	41.8	61.5	44.4	25.2	45.4	54.6	81.0 (95.5)	317 (473)
PVT-L [64]	42.6	63.7	45.4	25.8	46.0	58.4	43.4	63.6	46.1	26.1	46.0	59.5	60.9 (71.5)	324 (476)

Table 5: Various backbones with RetinaNet. Here R and X are abbreviations for ResNet and ResNeXt. Parameters and FLOPS in black are for backbones, while those in (gray) are for the whole frameworks.

Efficient & Accurate both on ImageNet and MS-COCO

Qualitative Part Attention

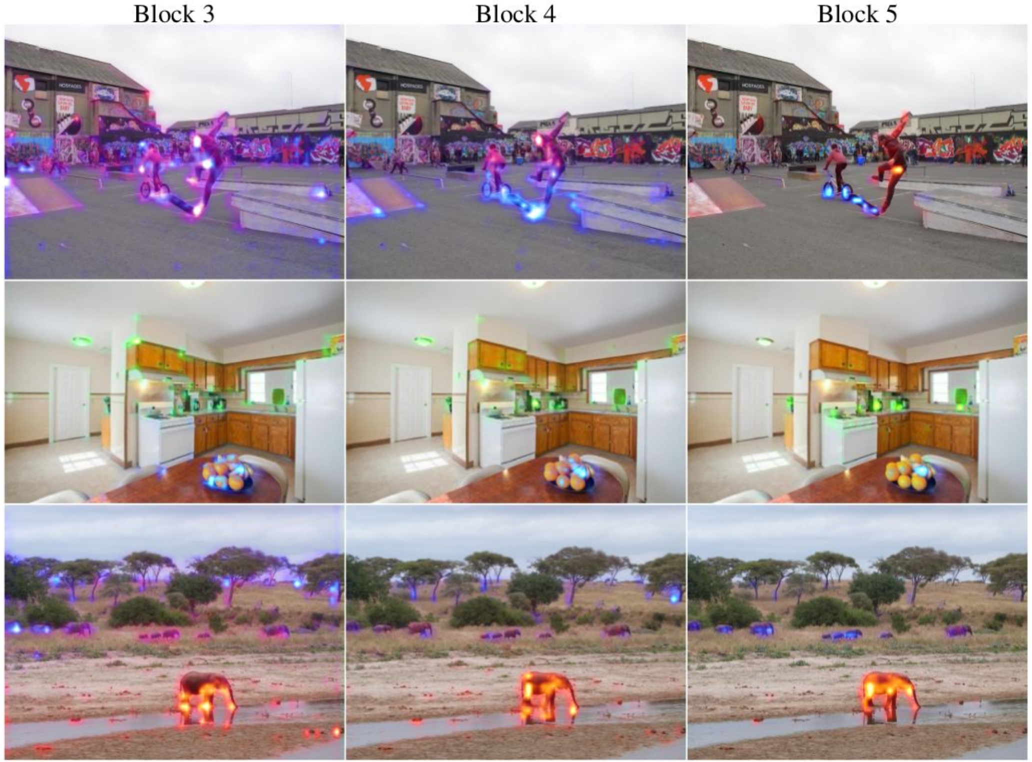


Figure 5: Visualization results about where the part representations attend on. Pixels rendered in different colors are associated to different parts. Best viewed in color.

Sparser attention in deeper blocks

Selective on body parts

Discussion

Simple idea to combine globality and locality

DETR with Whole-to-Part Attention?

DETR aggregates local information via query embeddings (which is then mapped to object class/box)

Part Selection

Contribution of part selection?

Hard selection of parts (instead of soft)? Is it really selective (sigmoid saturates easily)?

Lack of Part Constraints

No inductive bias on part structure? (i.e. parts should be focused, rather than spread across the image)