Pre-trained Models Can Count (Almost): Exploring Quantitative Structure in Visual Representations

Supplementary Material

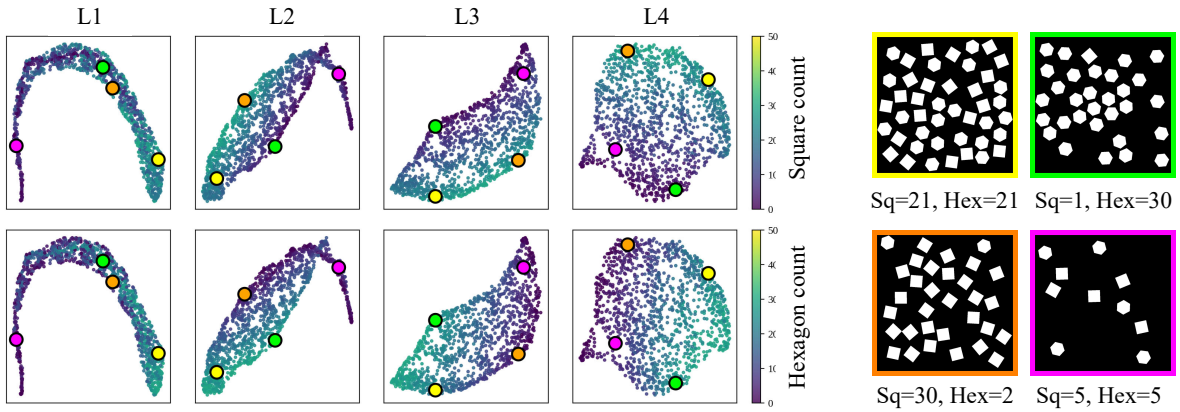

Figure 1. UMAP visualization of the feature space for images containing both squares and hexagons (used in the shape-selective counting task). Left panel: UMAP projections of feature vectors from each layer, with each point (image) color-coded by its object count. The top row is colored by the number of squares, and the bottom row by the number of hexagons. Right panel: sample images (the numbers of squares and hexagons are shown beneath each image), where the border color corresponds to the color of the matching point in the left-panel UMAP.

A. Additional Experimental Results

A.1. Feature Space Visualization: More Results

In Sec. 6.1 of the main paper, we presented UMAP visualizations of the feature space for two real-image datasets. In this section, we provide additional visualizations using further datasets.

We first consider the selective counting tasks from Sec. 4.2, where each image contains a mixture of squares and hexagons on a black background (see Fig. 4(b) in the main paper). The task is to count only the specified shape.

As described in the main paper, each image is processed by the ImageNet-pretrained ResNet-50, and we extract feature maps from each layer. These feature maps are vectorized using global average pooling (GAP), resulting in 256-, 512-, 1024-, and 2048-dimensional feature vectors for Layers 1–4. For each feature space, we apply UMAP to the combined set of 1,300 training and test images and visualize the resulting 2D embeddings by color-coding each point according to the object count. The results are shown in Fig. 1.

The Layer 3 visualization (third row of the left panel, labeled "L3"), which corresponds to the features yielding the highest counting accuracy, is particularly informative. As in the real-image datasets, the embeddings vary almost monotonically with object count. Moreover, in this dataset, the embeddings that reflect the number of squares (top row of the left panel) and those that reflect the number of hexagons (bottom row) are clearly separated, with distinct embedding directions. This separation provides insight into why selective counting is achievable using pretrained features.

We also present the Layer 3 visualizations for all training and test images from ShanghaiTech A and B (1,198 images in total) in Fig. 2. Despite the significant variation in the scale of people both within and across images—making these datasets particularly challenging for counting—the embeddings again show a clear structure aligned with object count.

Taken together, these results further reinforce our central finding: even without any task-specific training for counting, ImageNet-pretrained features inherently encode object count information within their feature space.

A.2. Color-Selective Counting Task

This section presents the experimental results for the color-selective counting task, *which were omitted from the main paper* due to space limitations. The task is defined on images containing red and blue discs of identical size placed on a black background, and consists of three subtasks: (1) counting only blue discs, (2) counting only red discs, and (3) counting the total number of discs of both colors. The

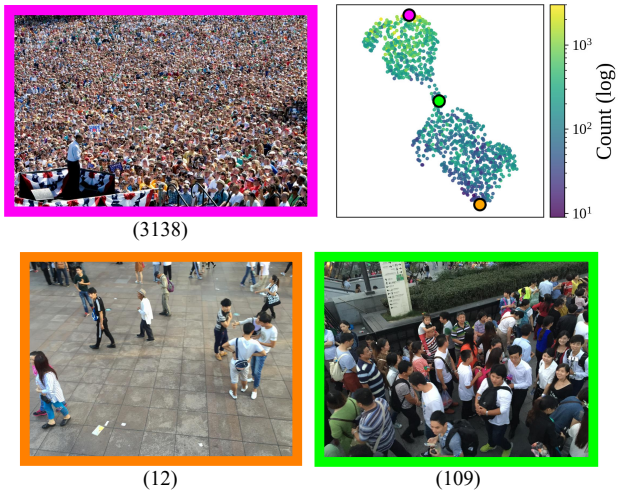

Figure 2. UMAP visualization for the counting task on ShanghaiTech A/B. The border color of each sample image corresponds to the color of the associated point in the UMAP plot. The numbers in parentheses indicate the object counts.

image generation procedure is described in Sec. B.1. The scatter plots for the three subtasks are shown in Fig. 3, and the corresponding prediction errors (MAE) are summarized in Table 1.

As with the size-selective and shape-selective counting tasks reported in the main paper, the pretrained model achieves high accuracy when using features from Layers 2 and 3, while the randomly initialized model fails to perform the task reliably.

A.3. Object Arrangement and Local Pooling

In the analysis of the effect of object arrangement in Sec. 4.3, we evaluated feature aggregation using GAP (Fig. 1(a) of the main paper). As described in the main paper, when the spatial distribution of objects is consistent between training and testing (uniform during training and uniform during testing; U/U), the counting accuracy is highest. When the distributions differ (U/C: uniform for training and center-concentrated for testing, or C/U: center-concentrated for training and uniform for testing), the accuracy decreases to some extent.

Here, we additionally evaluate spatial aggregation using local pooling (Fig. 1(b) in the main paper), varying the pooling kernel size as $k = 3$, 5, and 7. The results are presented in Table 2.

The overall trend is similar to that observed with GAP: the U/C and C/U settings yield lower accuracy than the U/U setting. However, using local pooling improves the overall performance, resulting in accuracy that even surpasses the U/U case with GAP. This indicates that local pooling provides greater robustness to changes in object arrangement.

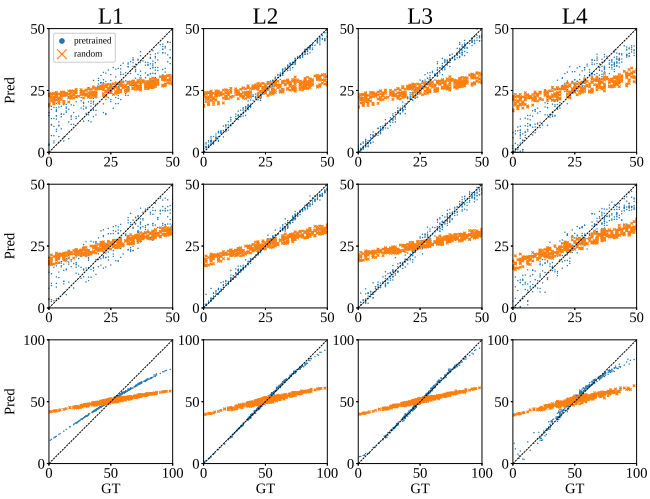

Figure 3. Selective-counting results for discs of different colors (see Fig. 4(c) in the main paper for example images). The regression model uses an MLP with 128 hidden units. Rows (top to bottom) correspond to counting red discs, counting blue discs, and counting the total number of discs. Columns (left to right) correspond to Layers 1–4. The ImageNet-pretrained ResNet-50 is shown with circles (o), and the randomly initialized (untrained) model with crosses (x).

Table 1. Accuracy (MAE) for the color-selective counting task.

Counting Task	Pretrained				Random Weight			
	L1	L2	L3	L4	L1	L2	L3	L4
Red	6.63	1.10	1.33	4.09	10.84	10.90	10.53	10.04
Blue	6.56	0.98	1.47	3.73	9.62	9.34	10.25	8.85
Both	5.94	0.92	0.91	2.65	12.47	11.78	11.81	11.67

Table 2. Counting accuracy (MAE) when the object arrangement differs between training and testing. The task is to selectively count hexagons in images containing both squares and hexagons. Results are shown for different kernel sizes of the local pooling method.

Train / Test	Kernel	L1	L2	L3	L4
U / U	3	3.28	0.68	0.51	0.57
	5	4.22	0.39	0.45	0.61
	7	4.07	0.48	0.43	0.68
	GAP	5.34	1.86	0.78	1.65
U / C	3	3.15	0.64	0.61	0.70
	5	4.38	0.48	0.69	0.84
	7	4.21	0.90	0.64	0.91
	GAP	5.44	1.96	1.03	2.06
C / U	3	3.38	0.84	0.65	0.95
	5	4.16	0.51	0.63	1.07
	7	3.97	0.88	0.65	0.98
	GAP	5.35	1.90	1.09	2.29

A.4. Evaluation Results of ViT Models

In the experiments of Sec. 4, we primarily used ResNet-50, and the evaluation of ViT models was limited to the object-counting task with varying backgrounds presented in Sec. 4.5. In this section, we report additional results for four ViT models—AugReg [38], CLIP [30], DINOv2 [28], and MAE (Masked Auto-Encoder) [10]—on the other tasks described in Sec. 4. For completeness, we also include the results of randomly initialized models; however, there should be no substantial theoretical differences among these models.

Details of Vision Transformer Models We employed four Vision Transformer (ViT) models sourced from the `timm` library. The specifications of these models are summarized in Table 3. Input images were resized to the resolution required by each model. For the ViT models, GAP features were computed by averaging the patch tokens while excluding the CLS token.

Simple Case Table 4 presents the results of the four ViT models on the simple case in Sec. 4.1—counting discs of a single size on a black background. The overall trends are consistent with those reported for the ImageNet-pretrained ResNet-50. Even with random weights, the models can accurately count large discs but fail on smaller ones. Pretrained models achieve stable accuracy across disc sizes, with the best performance generally appearing in the mid-to-late blocks (B7–B11). Among the models, MAE (Masked Auto-Encoder) performed best, whereas DINOv2 showed the lowest accuracy.

Table 4. Accuracy (MAE) for counting single-size discs with radius r pixels, using intermediate features from various ViT models. B0–B11 denote the blocks of each ViT model, ordered from the input side toward the output side.

Model		r	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	5	1.09	1.39	1.36	1.31	1.64	1.38	1.21	0.95	1.01	1.10	1.27	1.24
		15	1.05	1.09	1.13	1.17	1.20	1.24	1.02	0.82	0.86	0.94	0.92	1.00
		25	0.80	0.86	0.89	0.88	1.05	0.97	0.80	0.73	0.65	0.64	0.69	0.70
		50	0.29	0.31	0.36	0.35	0.35	0.33	0.30	0.24	0.23	0.23	0.26	0.23
	Random Weight	5	12.41	12.36	12.31	12.24	12.20	12.13	12.07	12.06	12.08	12.04	12.09	12.01
		15	8.15	2.27	0.80	0.59	0.56	0.50	0.41	0.37	0.34	0.31	0.32	0.32
		25	0.34	0.25	0.24	0.26	0.24	0.22	0.23	0.28	0.24	0.24	0.29	0.29
		50	0.07	0.08	0.07	0.06	0.07	0.07	0.08	0.08	0.08	0.07	0.07	0.08
CLIP	Pretrained	5	9.18	6.28	1.98	2.24	2.23	2.46	2.19	2.65	2.83	2.01	1.92	1.47
		15	1.75	1.61	1.68	1.76	2.09	2.43	2.30	1.93	1.89	1.59	1.44	1.28
		25	1.42	1.42	1.32	1.38	1.12	1.31	1.53	1.60	1.61	1.40	1.64	1.28
		50	0.40	0.45	0.53	0.53	0.35	0.32	0.39	0.39	0.50	0.41	0.42	0.45
	Random Weight	5	12.45	12.39	12.37	12.35	12.31	12.24	12.24	12.27	12.17	12.14	12.11	12.07
		15	7.94	4.63	0.68	0.64	0.71	0.62	0.47	0.56	0.48	0.49	0.52	0.42
		25	0.43	0.37	0.31	0.33	0.21	0.23	0.25	0.26	0.26	0.24	0.24	0.25
		50	0.08	0.08	0.07	0.06	0.05	0.07	0.06	0.07	0.08	0.06	0.06	0.08
DINOv2	Pretrained	5	11.68	11.56	11.61	10.64	9.74	4.81	3.64	3.06	2.03	1.60	1.73	1.51
		15	10.44	10.88	10.50	7.74	6.71	2.55	2.28	2.25	1.62	1.13	1.22	1.11
		25	5.21	6.34	5.84	3.18	1.86	1.65	1.58	1.24	1.02	0.72	0.72	0.56
		50	0.95	0.94	1.07	0.80	0.52	0.44	0.36	0.37	0.36	0.26	0.23	0.18
	Random Weight	5	12.46	12.41	12.40	12.37	12.37	12.23	12.28	12.22	12.16	12.04	12.13	11.90
		15	6.54	2.25	1.28	0.67	0.41	0.45	0.31	0.35	0.24	0.25	0.22	0.20
		25	0.09	0.09	0.09	0.09	0.11	0.10	0.09	0.10	0.09	0.09	0.09	0.10
		50	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.04	0.03	0.03
MAE	Pretrained	5	0.87	1.15	1.20	1.21	1.16	1.07	1.06	1.11	1.11	1.02	0.80	0.86
		15	0.76	0.79	0.86	0.83	0.76	0.85	0.85	0.87	0.74	0.56	0.64	0.70
		25	0.40	0.43	0.43	0.46	0.49	0.51	0.51	0.43	0.37	0.39	0.37	0.40
		50	0.12	0.13	0.14	0.15	0.18	0.17	0.14	0.14	0.13	0.13	0.13	0.16
	Random Weight	5	12.46	12.41	12.40	12.36	12.38	12.26	12.25	12.23	12.25	12.18	12.14	12.17
		15	7.11	1.79	0.72	0.96	0.69	0.52	0.64	0.49	0.56	0.52	0.52	0.52
		25	0.35	0.29	0.24	0.31	0.25	0.26	0.23	0.21	0.20	0.21	0.21	0.23
		50	0.06	0.06	0.05	0.06	0.06	0.06	0.06	0.06	0.06	0.05	0.06	0.05

Size-Selective Counting Table 5 reports the MAE results for the Size-Selective Counting task described in Sec. 4.2.

Table 5. Accuracy (MAE) for size-selective count using intermediate features from different ViT models.

Model		r	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	20	9.74	9.40	7.83	8.08	6.63	7.44	5.77	6.64	4.69	4.85	4.77	4.22
		25	7.65	7.34	7.00	6.56	4.73	5.45	4.97	4.41	3.84	3.48	3.33	2.96
		Both	2.90	2.69	2.36	2.57	3.01	2.98	3.14	3.22	3.26	3.12	2.88	2.28
	Random Weight	20	10.71	10.54	10.41	10.32	10.21	10.28	10.14	10.12	9.88	9.82	9.92	9.91
		25	6.56	6.58	6.51	6.39	6.33	6.36	6.19	6.24	6.18	6.17	6.21	6.23
		Both	3.95	3.86	3.81	3.88	3.88	3.86	3.86	3.85	3.89	3.80	3.78	3.78
CLIP	Pretrained	20	10.21	10.08	9.51	9.25	9.23	9.45	9.55	9.89	9.65	9.64	9.34	8.69
		25	7.93	7.91	7.78	7.52	7.36	7.02	7.15	7.01	6.68	6.69	6.59	6.32
		Both	4.21	4.26	3.94	3.43	2.91	3.45	3.73	4.16	4.00	3.96	3.81	3.66
	Random Weight	20	10.52	10.19	10.23	10.10	10.19	10.24	10.07	9.92	10.01	9.82	9.77	9.78
		25	6.51	6.41	6.44	6.44	6.41	6.37	6.36	6.32	6.36	6.32	6.33	6.14
		Both	3.83	3.68	3.77	3.70	3.68	3.66	3.59	3.60	3.64	3.62	3.62	3.65
DINOv2	Pretrained	20	10.40	10.84	11.23	10.44	10.08	10.07	9.99	9.81	9.71	6.74	7.31	4.80
		25	8.59	8.38	8.61	8.41	7.80	7.46	7.44	7.16	6.47	4.71	4.63	3.32
		Both	9.72	10.27	11.51	10.58	4.84	3.90	3.85	3.70	3.78	3.63	3.63	2.28
	Random Weight	20	10.81	10.83	10.85	10.85	10.81	10.81	10.81	10.80	10.79	10.82	10.78	10.79
		25	6.64	6.62	6.60	6.62	6.68	6.61	6.67	6.65	6.64	6.66	6.62	6.64
		Both	4.02	4.01	4.00	4.04	3.97	3.98	3.96	3.94	3.96	3.97	3.97	3.99
MAE	Pretrained	20	9.19	8.46	7.78	5.96	3.09	2.08	2.08	1.50	1.56	1.82	1.93	2.41
		25	6.41	5.44	5.35	4.16	1.99	1.57	1.40	1.24	1.34	1.35	1.59	1.97
		Both	2.89	2.77	2.69	2.36	1.84	1.57	1.83	1.17	1.13	0.94	0.95	1.24
	Random Weight	20	10.55	10.15	10.07	10.05	9.88	9.93	9.97	9.90	9.67	9.71	9.77	9.72
		25	6.48	6.38	6.31	6.20	6.25	6.12	6.17	6.21	6.06	6.02	5.88	5.84
		Both	3.81	3.82	3.63	3.69	3.58	3.63	3.68	3.60	3.58	3.60	3.59	3.62

In this task, the performance varies considerably across models, and their behavior differs from that of ResNet-50 reported in the main text. First, the models with random weights (common to all ViTs) perform reasonably well for total counting but fail entirely at size-selective counting. Among the pretrained models, MAE performs best: its mid-to-upper blocks exhibit the highest accuracy, reaching a level close to that of ResNet-50. In contrast, AugReg and DINOv2 show only modest accuracy near their top blocks, and overall performance remains low. CLIP, in particular, performs no better than random weights.

For reference, Table 6 shows the results obtained by increasing the number of hidden units in the MLP from 128 to 1,024. All models exhibit some improvement, and the lower mid-level blocks of AugReg achieve moderate accuracy. These results suggest that certain intermediate layers of these models do contain the information required for size-selective counting, but that this information is encoded in a more complex (more nonlinear) manner than in ResNet-50 or MAE.

Table 3. Vision Transformer models used in our experiments. All models are taken from the `timm` library.

Model	Backbone	Pretraining	Input size	`timm` ID
AugReg [38]	ViT-B/16	Self-supervised (AugReg)	224×224	`vit_base_patch16_224.augreg_in21k_ft_in1k`
CLIP [30]	ViT-B/16	Image–text (CLIP)	224×224	`vit_base_patch16_clip_224.openai`
DINOv2 [28]	ViT-B/14	Self-supervised (DINOv2)	518×518	`vit_base_patch14_dinov2.lvd142m`
MAE [10]	ViT-B/16	Self-supervised (MAE)	224×224	`vit_base_patch16_224.mae`

Table 6. Accuracy (MAE) for size-selective counting using intermediate features from different ViT models when an MLP with a hidden dimension of 1024 is employed.

Model		r	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	20	7.32	3.86	1.93	1.79	1.89	2.16	2.39	2.52	2.92	3.21	3.37	3.23
		25	5.39	3.37	1.56	1.49	1.60	1.86	2.02	1.96	2.16	2.29	2.34	2.26
		Both	2.40	1.86	1.20	1.10	1.06	1.42	1.26	1.38	1.39	1.51	1.74	1.47
	Random Weight	20	9.66	9.31	8.68	8.11	7.62	7.62	7.41	7.23	7.11	7.02	6.79	6.86
		25	6.08	6.08	5.62	5.30	5.03	5.05	5.01	4.80	4.72	4.64	4.45	4.45
		Both	3.47	3.49	3.42	3.38	3.38	3.36	3.28	3.26	3.30	3.29	3.11	3.15
CLIP	Pretrained	20	9.84	9.19	6.73	4.52	4.47	5.23	6.39	7.15	6.14	6.06	5.52	5.57
		25	7.36	7.11	5.87	4.21	3.61	4.19	4.65	5.05	4.64	4.55	4.15	4.31
		Both	3.35	2.91	2.41	1.96	2.10	2.39	2.68	3.00	2.83	2.93	2.65	2.42
	Random Weight	20	9.47	9.21	8.43	8.41	8.04	7.80	7.70	7.53	7.39	7.31	7.19	7.13
		25	6.12	5.77	5.41	5.31	5.06	4.95	4.91	4.84	4.65	4.68	4.59	4.65
		Both	3.43	3.45	3.34	3.38	3.44	3.33	3.30	3.31	3.27	3.21	3.20	3.17
DINOv2	Pretrained	20	10.06	10.03	10.08	9.78	9.49	8.61	8.13	6.97	5.13	3.66	4.03	3.34
		25	7.61	7.74	7.71	7.50	7.03	6.15	5.72	4.79	3.45	2.54	2.75	2.53
		Both	2.64	3.14	3.25	3.49	3.17	2.96	2.80	2.69	2.86	1.91	2.06	1.55
	Random Weight	20	10.47	10.42	10.54	10.33	10.22	10.16	10.15	10.01	9.94	9.73	9.80	9.66
		25	6.47	6.48	6.42	6.36	6.34	6.32	6.29	6.22	6.17	6.03	6.15	6.02
		Both	3.87	3.85	3.85	3.87	3.88	3.84	3.82	3.77	3.75	3.71	3.74	3.70
MAE	Pretrained	20	4.18	2.40	2.06	1.68	1.62	1.49	1.24	1.16	1.31	1.38	1.53	1.84
		25	3.30	1.78	1.63	1.31	1.22	1.16	0.97	0.89	1.02	1.07	1.20	1.43
		Both	2.51	1.94	2.02	1.53	0.93	0.84	0.71	0.51	0.58	0.62	0.67	0.81
	Random Weight	20	9.51	8.80	8.28	7.90	7.72	7.51	7.35	7.29	7.43	7.35	7.25	6.98
		25	5.84	5.39	5.21	4.99	4.87	4.74	4.70	4.63	4.72	4.71	4.65	4.48
		Both	3.50	3.35	3.37	3.28	3.28	3.34	3.26	3.35	3.24	3.32	3.27	3.19

Shape-Selective Counting Table 7 presents the MAE results for the Shape-Selective Counting task described in Sec. 4.2. In this task, all models show results comparable to those of ResNet-50. Models with random weights fail to perform the task, while pretrained models achieve relatively high accuracy in the mid-to-late blocks (approximately B7–B11), with some variation across models.

Table 7. Accuracy(MAE) for shape selective count using intermediate features from different ViT models.

Model		Task	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	Square	4.91	4.50	2.32	0.88	0.80	0.74	0.83	0.82	0.85	0.84	0.85	0.83
		Hexagon	5.29	4.84	2.85	1.00	0.92	0.86	0.88	0.81	0.86	0.96	0.91	0.86
		Both	0.82	0.86	0.86	0.84	1.07	0.92	1.12	1.03	0.88	1.00	0.91	0.86
	Random Weight	Square	5.08	5.07	5.04	5.04	5.04	5.02	5.03	5.04	5.03	5.02	5.02	5.02
		Hexagon	5.13	5.15	5.12	5.13	5.12	5.11	5.11	5.13	5.12	5.13	5.11	5.10
		Both	0.44	0.30	0.31	0.33	0.29	0.31	0.26	0.24	0.23	0.20	0.24	0.23
CLIP	Pretrained	Square	4.97	4.98	4.76	3.47	1.60	1.09	0.95	1.15	0.98	0.94	0.91	0.95
		Hexagon	5.33	5.28	5.05	3.67	1.55	1.08	1.16	1.07	1.06	1.02	0.98	0.95
		Both	1.45	1.41	1.25	1.36	1.09	1.25	1.35	1.52	1.31	1.42	1.37	1.31
	Random Weight	Square	5.06	5.06	5.04	5.05	5.03	5.03	5.03	5.04	5.04	5.04	5.05	5.04
		Hexagon	5.13	5.15	5.13	5.12	5.13	5.13	5.14	5.13	5.14	5.15	5.14	5.12
		Both	0.40	0.26	0.26	0.28	0.27	0.25	0.28	0.26	0.24	0.26	0.25	0.23
DINOv2	Pretrained	Square	5.24	5.14	5.38	4.97	4.60	3.09	1.38	1.20	0.96	0.98	0.97	0.94
		Hexagon	5.78	5.78	5.83	5.30	4.98	3.57	1.72	1.41	1.16	0.94	1.15	0.98
		Both	5.90	5.82	7.09	4.76	2.42	1.76	1.56	1.38	1.11	1.00	0.99	0.76
	Random Weight	Square	5.10	5.10	5.10	5.10	5.10	5.10	5.09	5.10	5.10	5.10	5.09	5.09
		Hexagon	5.13	5.11	5.11	5.11	5.12	5.11	5.13	5.13	5.13	5.13	5.12	5.11
		Both	0.15	0.13	0.13	0.12	0.13	0.10	0.10	0.10	0.08	0.10	0.11	0.12
MAE	Pretrained	Square	4.90	4.64	4.14	2.75	1.18	0.97	0.76	0.73	0.72	0.73	0.78	0.93
		Hexagon	5.18	4.90	4.06	3.17	1.22	1.01	0.81	0.72	0.74	0.80	0.86	1.03
		Both	0.53	0.50	0.41	0.52	0.51	0.45	0.42	0.40	0.39	0.43	0.46	0.56
	Random Weight	Square	5.05	5.06	5.06	5.05	5.04	5.03	5.03	5.03	5.03	5.03	5.05	5.03
		Hexagon	5.13	5.11	5.09	5.10	5.11	5.10	5.11	5.10	5.10	5.09	5.08	5.10
		Both	0.36	0.41	0.32	0.29	0.31	0.29	0.27	0.27	0.26	0.25	0.23	0.25

Color-Selective Counting Table 8 presents the results for the Color-Selective Counting task (the same task used to evaluate ResNet-50 in Sec. A.2, where the model selectively counts red and blue discs).

Table 8. Accuracy (MAE) for color-selective counting using intermediate features from different ViT models.

Model		Task	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	Red	1.24	1.20	1.31	1.34	1.91	1.90	1.62	1.64	1.74	1.64	1.82	1.78
		Blue	1.27	1.08	1.19	1.26	1.95	1.58	1.80	1.63	1.80	1.55	1.73	1.63
		Both	2.44	2.21	2.13	2.41	2.41	2.40	2.34	2.18	1.66	1.64	1.63	1.58
	Random Weight	Red	0.94	0.82	0.77	0.76	0.73	0.75	0.74	0.77	0.75	0.79	0.78	0.77
		Blue	1.00	0.90	0.91	0.94	0.86	0.89	0.83	0.85	0.85	0.85	0.84	0.82
		Both	3.54	2.73	1.49	1.51	1.40	1.39	1.33	1.29	1.32	1.35	1.31	1.34
CLIP	Pretrained	Red	2.46	2.41	3.35	4.60	6.02	6.96	5.95	5.35	5.93	4.14	3.43	2.88
		Blue	2.38	1.99	2.53	4.29	6.24	6.36	5.08	5.18	4.64	4.44	3.76	2.20
		Both	4.14	3.29	2.82	2.97	2.70	3.41	3.55	3.44	4.13	4.08	3.64	3.92
	Random Weight	Red	1.18	1.15	1.11	1.10	1.12	1.06	1.06	1.03	1.04	1.03	1.00	1.01
		Blue	1.09	1.04	1.05	1.05	0.97	0.94	0.90	0.90	0.91	0.90	0.88	0.90
		Both	2.16	2.11	2.01	1.99	1.98	1.87	1.82	1.79	1.78	1.77	1.70	1.73
DINOv2	Pretrained	Red	9.09	10.25	10.24	9.45	9.02	9.10	8.83	8.59	8.48	7.26	7.61	6.03
		Blue	8.88	9.88	9.85	8.70	8.56	8.56	8.47	8.57	8.21	7.92	7.95	6.32
		Both	10.60	12.85	11.46	9.48	6.46	3.40	3.06	3.14	2.45	2.22	2.42	1.70
	Random Weight	Red	1.31	1.31	1.31	1.31	1.31	1.31	1.31	1.31	1.31	1.31	1.31	1.32
		Blue	1.11	1.11	1.12	1.11	1.11	1.11	1.12	1.11	1.11	1.11	1.11	1.11
		Both	2.39	2.48	2.37	2.36	2.42	2.37	2.42	2.38	2.47	2.41	2.40	2.39
MAE	Pretrained	Red	0.74	0.83	0.83	0.78	0.88	0.89	0.84	0.85	0.86	0.85	0.93	1.01
		Blue	0.66	0.68	0.75	0.72	0.64	0.77	0.86	0.78	0.77	0.83	0.86	1.00
		Both	1.06	1.09	1.25	1.15	1.23	1.06	1.02	0.93	0.81	0.75	0.82	0.88
	Random Weight	Red	1.12	1.02	0.96	0.91	0.91	0.89	0.89	0.88	0.88	0.84	0.84	0.79
		Blue	1.01	0.95	0.92	0.88	0.85	0.85	0.86	0.85	0.87	0.83	0.84	0.82
		Both	2.05	1.86	1.79	1.66	1.64	1.58	1.57	1.54	1.61	1.49	1.50	1.43

This task exhibits trends that differ from the other tasks. First, the random-weight models achieve surprisingly high accuracy—surpassing even the pre-trained ResNet-50. Moreover, for most models except MAE, the random-weight version performs better than the pre-trained one. For CLIP and DINOv2, the pre-trained models not only underperform their random-weight counterparts but fail almost entirely at the task. In contrast, the pre-trained AugReg model shows a drop in performance but still maintains accuracy comparable to ResNet-50. MAE (Masked Autoencoder) consistently achieves high accuracy across tasks; it is the only model that outperforms its random-weight counterpart, and thus exceeds the performance of ResNet-50. Table 9 shows the results obtained when increasing the hidden dimension to 1024. As in the previous tasks, all models exhibit substantial improvement, while the relative ranking among models remains unchanged.

These findings offer several insights into how these models preserve color information in their feature representations. First, the strong performance of random-weight models suggests that the architectural structure itself plays some role in representing color information. Second, the severe degradation observed in some pre-trained models indicates that the way color information is encoded in the learned features differs significantly across models.

Table 9. Accuracy (MAE) for color-selective counting using intermediate features from different ViT models when an MLP with a hidden dimension of 1024 is employed.

Model		Task	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	Red	0.86	0.73	0.83	0.85	0.90	0.87	0.99	1.07	1.08	1.03	1.21	1.17
		Blue	0.82	0.69	0.79	0.83	0.86	0.89	0.97	1.03	1.15	1.07	1.29	1.24
		Both	1.16	1.05	1.21	0.93	0.99	1.01	1.09	1.29	1.17	1.15	1.31	1.23
	Random Weight	Red	0.83	0.77	0.76	0.73	0.77	0.74	0.75	0.74	0.74	0.73	0.73	0.74
		Blue	0.83	0.83	0.76	0.74	0.75	0.71	0.70	0.72	0.73	0.75	0.75	0.74
		Both	1.51	1.45	1.33	1.22	1.31	1.20	1.26	1.22	1.23	1.23	1.24	1.18
CLIP	Pretrained	Red	1.29	0.97	1.17	1.13	1.12	1.46	1.21	1.39	1.47	1.48	1.44	1.80
		Blue	1.30	1.00	1.00	1.06	1.20	1.25	1.36	1.51	1.49	1.63	1.46	1.55
		Both	2.30	1.82	1.58	1.48	1.46	1.50	1.75	1.92	2.10	1.98	1.93	2.07
	Random Weight	Red	1.16	1.07	1.06	1.01	0.99	0.98	0.95	0.92	0.93	0.91	0.91	0.90
		Blue	1.11	1.05	1.02	1.01	1.00	0.98	0.95	0.91	0.90	0.86	0.86	0.84
		Both	2.18	2.02	1.99	1.92	1.89	1.85	1.79	1.73	1.65	1.64	1.64	1.61
DINOv2	Pretrained	Red	0.66	3.03	5.88	5.67	5.16	5.36	5.04	3.54	2.88	2.77	3.34	3.42
		Blue	0.65	3.32	5.39	5.36	5.37	5.39	4.58	3.13	3.00	2.54	3.42	3.14
		Both	1.45	1.99	2.11	2.21	1.93	1.51	1.44	1.41	1.17	0.97	1.02	0.93
	Random Weight	Red	1.31	1.32	1.32	1.31	1.31	1.32	1.31	1.32	1.31	1.31	1.31	1.31
		Blue	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11
		Both	2.36	2.36	2.36	2.36	2.36	2.36	2.36	2.36	2.36	2.36	2.36	2.36
MAE	Pretrained	Red	0.50	0.53	0.55	0.63	0.65	0.62	0.59	0.63	0.59	0.61	0.66	0.70
		Blue	0.53	0.50	0.51	0.48	0.54	0.54	0.56	0.56	0.58	0.58	0.66	0.75
		Both	0.85	0.78	0.76	0.79	0.68	0.72	0.67	0.64	0.59	0.55	0.64	0.70
	Random Weight	Red	1.12	1.01	0.90	0.84	0.83	0.81	0.79	0.78	0.74	0.69	0.68	0.72
		Blue	0.96	0.92	0.88	0.83	0.79	0.76	0.74	0.72	0.68	0.65	0.65	0.68
		Both	2.01	1.84	1.67	1.59	1.49	1.42	1.38	1.34	1.23	1.13	1.14	1.18

Robustness to Background Variation Table 10 presents the results for the Robustness to Background Variation task described in Sec. 4.4. While Table 7 in Sec. 4.5 showed a subset of these results, this table provides the complete version. For all ViT models, robustness to background variation is observed, consistent with the behavior of ResNet. Although the exact performance varies across models, accuracy is generally highest in the mid-to-late blocks (approximately B7–B11).

Table 10. Accuracy (MAE) for counting colored discs in varying backgrounds using intermediate features from different ViT models.

Model		Task	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
AugReg	Pretrained	Black	1.58	1.37	1.32	1.41	1.46	1.30	1.45	1.36	1.40	1.42	1.42	1.34
		Fixed	4.27	3.26	2.06	1.93	2.02	1.94	2.18	2.15	2.03	2.06	2.08	2.14
		Random	6.91	4.16	3.15	2.98	3.01	2.87	3.05	3.09	3.19	3.43	4.33	4.69
	Random Weight	Black	9.92	6.34	5.45	4.21	2.66	2.55	2.37	2.27	2.63	2.40	2.42	2.36
		Fixed	11.53	9.44	9.18	6.65	4.93	7.16	4.50	4.61	4.98	4.33	4.19	4.24
		Random	13.49	13.30	13.08	12.98	12.86	12.88	12.84	12.82	12.69	12.53	12.65	12.50
CLIP	Pretrained	Black	2.04	1.95	1.88	1.99	1.91	2.16	2.36	2.33	2.15	2.17	2.12	1.79
		Fixed	9.74	9.84	7.65	6.61	2.91	3.59	2.39	2.08	2.36	2.32	2.34	2.32
		Random	9.17	6.90	4.31	3.80	3.29	3.10	3.02	3.03	3.28	3.45	3.45	3.92
	Random Weight	Black	12.27	11.81	11.19	10.56	9.20	8.18	6.65	8.46	5.95	5.31	4.81	5.56
		Fixed	11.31	9.04	6.38	6.19	5.31	5.22	6.22	5.15	5.32	5.44	5.12	5.06
		Random	13.38	12.98	12.84	12.75	12.68	12.68	12.53	12.61	12.40	12.39	12.38	12.34
DINOv2	Pretrained	Black	10.28	9.35	8.93	7.28	3.44	3.07	2.18	2.02	1.53	1.35	1.39	1.24
		Fixed	12.03	12.13	11.76	10.45	9.89	7.49	4.86	1.88	1.70	1.62	1.60	1.39
		Random	12.61	11.91	11.76	10.26	8.30	5.42	3.72	3.37	3.09	3.04	3.23	3.71
	Random Weight	Black	11.64	10.39	6.93	5.97	4.43	2.99	3.18	2.39	2.38	2.74	2.39	2.36
		Fixed	8.13	6.39	4.99	6.07	4.81	4.43	4.35	4.38	4.34	4.36	4.39	4.43
		Random	13.59	13.38	13.20	13.17	13.03	12.96	12.83	12.66	12.73	12.82	12.64	12.58
MAE	Pretrained	Black	1.64	1.38	1.42	1.34	1.29	1.23	1.26	1.21	0.86	0.92	0.92	0.95
		Fixed	2.64	2.41	2.30	2.10	1.80	1.70	1.70	1.64	1.65	1.73	1.85	1.82
		Random	5.72	5.20	4.32	3.50	2.94	2.72	2.72	2.59	2.64	2.70	2.88	2.91
	Random Weight	Black	11.57	7.92	6.39	4.19	3.55	3.21	3.33	2.60	2.27	2.22	2.38	2.21
		Fixed	9.03	7.18	5.98	6.15	5.02	4.98	4.70	4.87	4.83	4.52	4.51	4.43
		Random	13.53	13.25	13.22	13.08	13.03	13.06	13.01	12.90	12.80	12.79	12.68	12.71

A.5. Scatter Plots Omitted from the Main Paper

In the main paper, we presented both the prediction–ground-truth scatter plots and the MAE tables only for the initial simple case using a linear regressor. For all subsequent experiments, only the MAE tables were shown. In this section, we present the corresponding scatter plots for each experiment. Note that all plots shown here are for the ImageNet-pretrained ResNet-50 model.

Figure 4 shows the scatter plots for the single-size disc counting task in Sec. 4.1 when using the MLP regressor ($m = 128$). These plots correspond to Table 2 and visually support the analysis discussed in the main paper.

Figure 5 shows the results of the selective counting task for disc sizes described in Sec. 4.2, and Figure 6 shows the results for the selective counting of polygonal shapes. These correspond to Tables 3 and 4 in the main paper, respectively.

Figure 7 shows the scatter plots for the colored-disc counting task under varying background conditions described in Sec. 4.4. They correspond to Table 6.

Figure 8 shows the results when the spatial distribution of discs differs between training and inference. The discs are either placed uniformly at random over the image or concentrated near the center. An example is shown in Fig. 4(d) of the main paper. This figure corresponds to Table 5 in the main paper.

Figure 9 shows the results of the various ViT models on the same task—counting colored discs under varying background conditions—described in Sec. 4.5. This corresponds to Table 7 in the main paper.

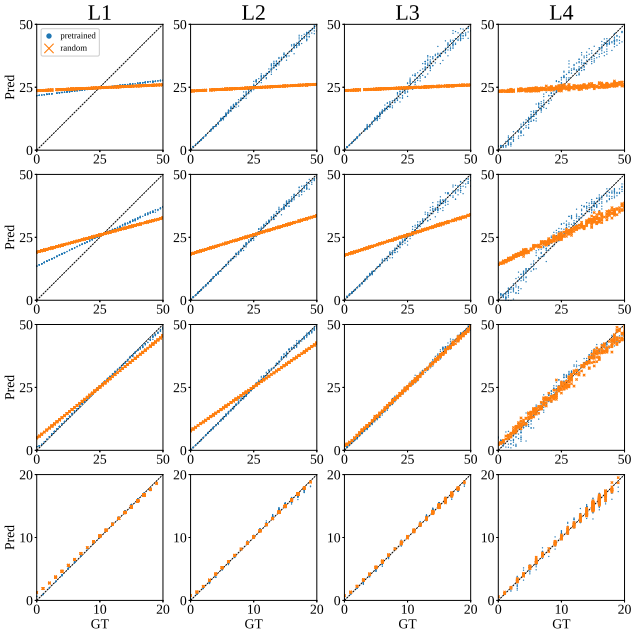

Figure 4. Counting results for single-size discs (plot version of Table 2 in the main paper). The regression model uses an MLP with 128 hidden units. Rows (top to bottom) correspond to disc radii r = 5, 15, 25, and 50 pixels, and columns (left to right) correspond to Layers 1–4. The ImageNet-pretrained ResNet-50 is shown in dark blue circles, and the random-weight (untrained) model in orange crosses.

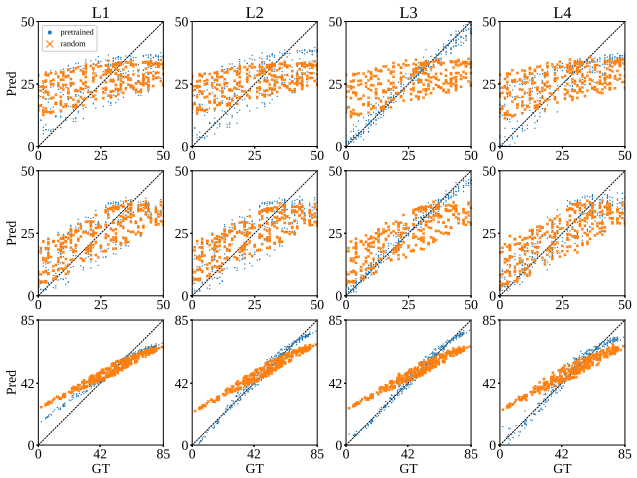

Figure 5. Selective-counting results for target disc sizes in images containing multiple disc sizes (plot version of Table 3 in the main paper). The regression model uses an MLP with 128 hidden units. Rows (top to bottom) correspond to the cases where only discs with $r = 20$ are counted, only those with $r = 25$ are counted, and both are counted, respectively. Columns (left to right) correspond to Layers 1–4.

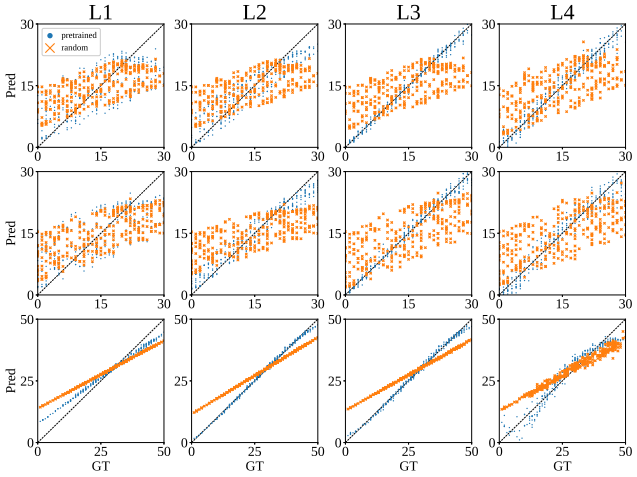

Figure 6. Selective-counting results for target shapes in images containing both squares and hexagons Counting results for single-size discs (plot version of Table 4 in the main paper). The regression model uses an MLP with 128 hidden units. Rows (top to bottom) correspond to counting only squares, only hexagons, and both shapes, respectively. Columns (left to right) correspond to Layers 1–4.

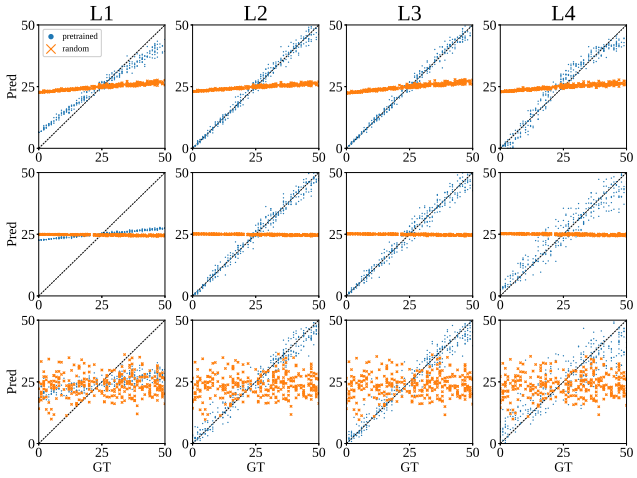

Figure 7. Counting results for disc images with different backgrounds (the plot version of Table 6 in the main paper). The regression model uses an MLP with 128 hidden units. Rows (top to bottom) correspond to: black background, fixed background shared between training and inference, and random background for both training and inference. Columns (left to right) correspond to Layers 1–4.

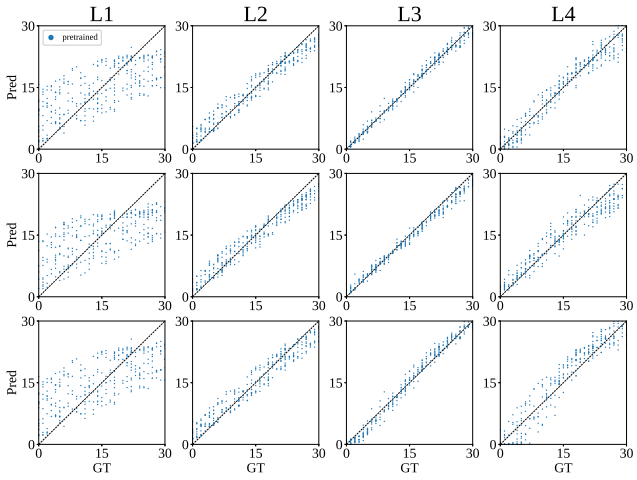

Figure 8. Results of disc counting when the spatial distribution of discs differs between training and testing. Rows correspond to Train/Test settings in the order U/U, U/C, and C/U from top to bottom, and columns correspond to Layers 1, 2, 3, and 4 from left to right.

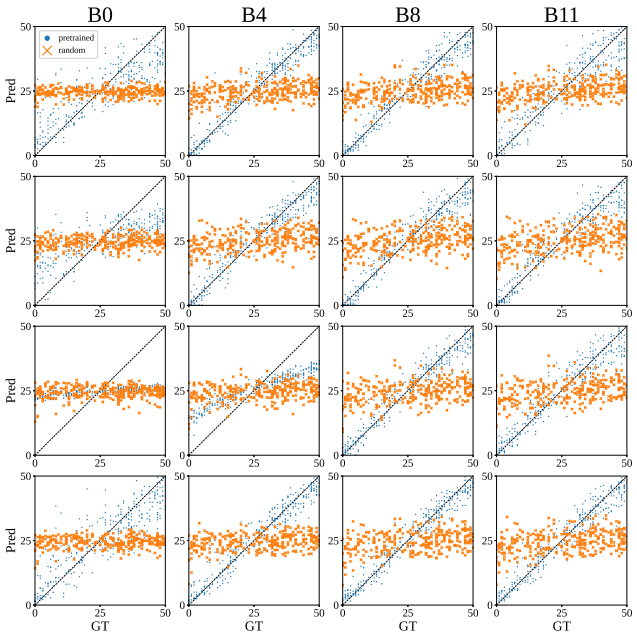

Figure 9. Counting results for discs over random natural-image backgrounds, showing the performance of various ViT models (plot version of Table 7 in the main paper). A 128-unit MLP is used as the regression head. Rows (top to bottom) correspond to AugReg, CLIP, DINOv2, and MAE, respectively, and columns (left to right) correspond to Blocks 0, 4, 8, and 11.

B. Details of Experimental Settings

B.1. Synthetic Data Generation Procedures

This section describes the synthetic image generation procedures used in the experiments in Sec. 4 of the main paper.

Single-Size Disc Images (Sec. 4.1) Each image is a 512×512 black canvas onto which we draw n white discs of radius r, where n is sampled uniformly from $[0, 50]$. Disc centers are sampled uniformly within the region that ensures the entire disc lies inside the image (i.e., with a margin of r pixels). If a sampled center causes overlap with an existing disc, it is resampled up to a fixed retry limit. Rendering stops when either n discs have been placed or the retry limit is exceeded. In the latter case, the number of successfully drawn discs is taken as the ground-truth count, resulting in a quasi-uniform spatial arrangement. For each radius $r \in \{5, 15, 25, 50\}$, we generated 1,000 training images and 300 test images.

Disc Images with Multiple Radii (Sec. 4.2) Using the same canvas configuration, we draw discs of two radii, $r = 20$ and $r = 25$. For each image, the number of discs of each radius is independently sampled from $[0, 50]$. Disc centers are sampled uniformly within the valid region and resampled when overlapping, up to a retry limit. The two disc types are rendered in alternating order. As before, rendering terminates when the target counts are reached or when retries exceed the limit; in the latter case, the number of placed discs becomes the ground truth. We generated 1,000 training and 300 test images.

Images with Different Shapes (Squares and Hexagons; Sec. 4.2) We render squares and regular hexagons on a 512×512 black background. The size of each shape is chosen so that its area matches that of a disc with radius $r = 25$. For each shape type, the target count is sampled uniformly from $[0, 30]$. Shapes are positioned by sampling center locations uniformly within the valid, non-overlapping region. Overlap is determined using the sum of the circumradii. Shape orientation is randomized. The two shape types are rendered alternately. Rendering for each type stops when the target count is reached or the retry limit is exceeded; the number of successfully placed shapes then becomes the ground truth. To reduce aliasing, we first render at $1{,}024 \times 1{,}024$ resolution and downsample to 512×512 using LANCZOS interpolation. We generated 1,000 training and 300 evaluation images.

Disc Images with Different Colors To evaluate color-selective counting (not included in the main paper), we generate images containing red and blue discs (radius $r = 20$) on a 512×512 black background. The number of discs of each color is independently sampled from $[0, 50]$. As with previous settings, disc centers are sampled to avoid boundary overflow and overlap, and placement attempts alternate between the two colors. Rendering stops when the target counts are reached or when retries exceed the limit, and the number of successfully placed discs is used as the ground truth. We generated 1,000 training and 300 evaluation images.

Object Arrangement Images in which objects are arranged with a central bias rather than uniformly at random are generated by extending the setting used for evaluating shape-selective counting. First, for discs with a radius $r = 25$, we determine the sizes of squares and regular hexagons so that their areas match those of the corresponding discs. For each image, the target numbers of squares and hexagons are sampled uniformly from the range $[0, 30]$, and the two shape types are placed alternately. During placement, the center of each shape is sampled from an isotropic Gaussian distribution whose mean is the image center and whose variance is set to 30% of the image radius. If a sampled shape overlaps an existing one or extends outside the image boundary, the placement is retried. Rendering stops when the target number of shapes has been placed or when the retry limit is exceeded, and the number of successfully placed shapes at that point is used as the ground-truth count.

We generated 1,000 training images and 300 test images. Each image is first created at a resolution of 1024×1024 and then downsampled to 512×512. The task is selective counting of hexagons.

B.2. Details of the Real-Image Experiments (Sec. 5)

This section describes the details of the experimental setup for real-image evaluation presented in Sec. 5 of the main paper.

The model architecture is as follows. We use ResNet-50 (timm: `resnet50.a1_in1k`) as the backbone. Based on the findings from the synthetic-image experiments, we use the output feature map from Layer 3 and perform spatial aggregation using local pooling, as illustrated in Fig. 1(b) of the main paper. The kernel size is set to 3×3. The regression head is a two-layer MLP with a hidden dimension of 128. The input images undergo the following preprocessing: the short side is padded with black pixels to make the image square, after which it is resized to 512×512 and normalized using the ImageNet mean and standard deviation.

During training, we use MSE as the loss function and AdamW for optimization, with a learning rate of 1×10^{-3} and a weight decay of 0.05. The learning rate is scheduled using CosineAnnealingLR ($T_{\max} = 60$). The model is trained for 60 epochs with a batch size of 16.

C. Details of Methods

C.1. Local Pooling

As described in Sec. 3, our primary approach aggregates intermediate feature maps using global average pooling (GAP), feeds the resulting vector into an MLP, and regresses the object count. In addition to this baseline, we also experimented with an alternative method that applies local average pooling of size $k \times k$, feeds each pooled feature vector into an MLP, and sums the outputs to obtain the final count. A brief overview was provided at the end of Sec. 3; here, we describe the method in detail.

Given an intermediate feature map $F \in \mathbb{R}^{w \times h \times c}$, we apply local average pooling of size $k \times k$ with non-overlapping windows, effectively performing $k : 1$ downsampling. We denote this operation by AP^k. Applying it to F yields

$$F' = \mathrm{AP}^k(F) \in \mathbb{R}^{\lceil w/k \rceil \times \lceil h/k \rceil \times c}.$$

Let (i, j) denote a spatial location in the pooled feature map F'. At each location, we apply the same regression model independently, and the final prediction is obtained by summing all outputs. Let $\mathrm{AP}^k_{i,j}(F) \in \mathbb{R}^c$ be the feature vector at location (i, j). The regression is then computed as

$$y = \sum_{i,j} g(\mathrm{AP}^k_{i,j}(F_l); \theta), \tag{3}$$

where g is a two-layer MLP, identical to the one used in the GAP-based method.

C.2. Visualization of Feature-map Attributions

In Sec. 6.2, we visualize feature-map attribution—that is, how each element of the feature map contributes to the prediction y for a given image I—using the $\mathrm{Gradient} \times \mathrm{Input}$ method [13,33]. The details are described below.

For each feature map $F \in \mathbb{R}^{w \times h \times c}$ obtained from a given layer of ResNet–50, we compute the attribution at each spatial position (i, j) as the inner product between the channel-wise gradient $\partial y / \partial f_{ij} \in \mathbb{R}^c$ and the feature vector $f_{ij} \in \mathbb{R}^c$:

$$\mathrm{Attribution}(i, j) = \left(\frac{\partial y}{\partial f_{ij}} \right)^\top f_{ij}. \tag{4}$$

Note that the attribution obtained using $\mathrm{Gradient} \times \mathrm{Input}$ is known to be equivalent to the first-order relevance term in Layerwise Relevance Propagation (LRP) [2].