DISPARQ: SELF-SUPERVISED PART CONCEPTS FOR INTERPRETABLE VISION FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

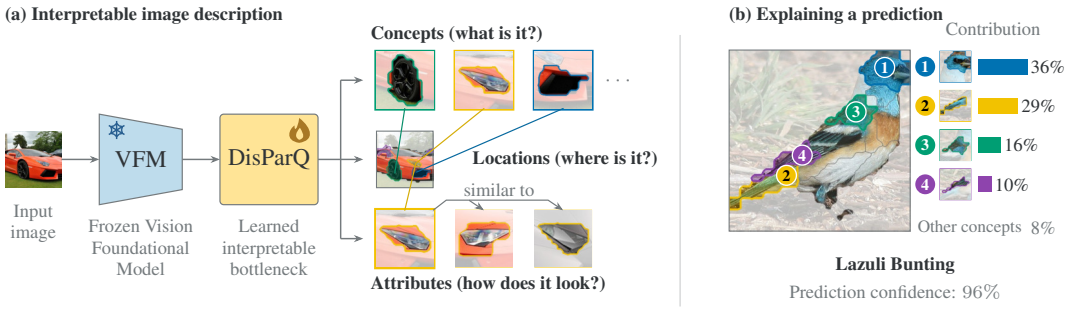


Figure 1: **DisParQ describes images through semantic visual concepts, their exclusive location assignment, and per-concept appearance attributes.** *Left:* a learned interpretable bottleneck on a frozen VFM identifies part concepts and assigns each location to one concept. Attribute examples illustrate similar appearances within the same learned concept. The concept representation is learned without class labels, part annotations, or language. *Right:* Top four concepts explain a prediction.

ABSTRACT

Concept-based vision models represent images through an intermediate layer of human-inspectable concepts, so what a model relies on can be traced to those concepts. However, those models are often limited to fixed categories or depend on language to define their concepts. We introduce **DisParQ** (**Dis**crete **Par**ts with **Q**uantized attributes), a method that learns spatially grounded, discrete concept representations from a powerful frozen vision-only self-supervised backbone. It requires no class labels and no language supervision. Each image patch is assigned to exactly one concept from a learnable prototype dictionary and only a sparse subset of concepts may activate per image. To capture how each concept varies across images (*e.g.,* the type of a "wheel"), we learn continuous residuals alongside the concepts, and then quantize them into discrete attributes. A spatial decoder reconstructs the backbone's representation from the concepts and attributes alone, *so successful reconstruction means that the discrete representation preserves the backbone's information.* We evaluate DisParQ across seven datasets, from general recognition (ImageNet, PartImageNet, Places) to fine-grained benchmarks (CUB, Cars, Dogs, Flowers). We show that DisParQ closely matches its frozen DINOv2 teacher on ImageNet linear probing **(83.2% top-1)**, achieves higher concept consistency than language-aligned models, remains competitive on fine-grained recognition, and enables cross-category part-based retrieval.

1 INTRODUCTION

Self-supervised vision transformers produce rich feature representations, serving as versatile foundation backbones for classification, segmentation, and retrieval (Caron et al., 2021; Oquab et al., 2024). As these models enter safety-critical domains, strong task performance alone is not sufficient, and we also need to understand ***what*** semantic concepts a model relies on and ***where*** it finds them during inference (Rudin, 2019; Rudin et al., 2022). This need has motivated research on inherently interpretable architectures. However, current paradigms make varying trade-offs in spatial grounding, concept representation, and supervision. Prototype-based networks (Chen et al., 2019; Janusz et al., 2026; Nauta et al., 2023; Rymarczyk et al., 2022) connect visual concepts to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

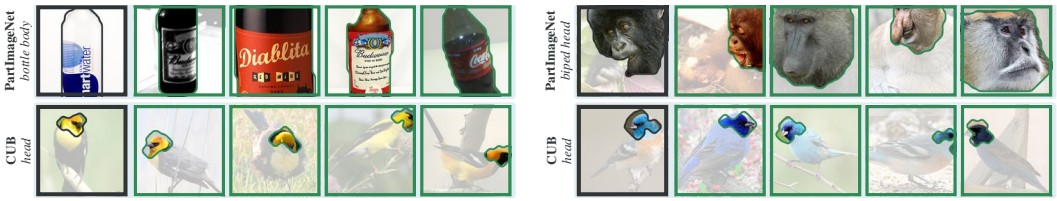


Figure 2: **Concepts retrieve the same part across categories.** A query region (black frame) and the four regions of other images with the closest DisParQ concept description (green frame).

fixed class labels and usually require fine-tuning the entire backbone. Concept Bottleneck Models (CBMs) (Koh et al., 2020; Oikarinen et al., 2023; Yang et al., 2023; Yuksekgonul et al., 2023; Rao et al., 2024) often rely on human-specified or language-derived concepts, but lack localized spatial grounding (Prasse et al., 2025). Recent variants such as CFM (Wittenmayer et al., 2026) introduce spatial grounding, yet remain reliant on language-aligned backbones. Sparse autoencoders (SAEs) have been applied to the patch tokens of vision models (Lim et al., 2025; Zaigrajew et al., 2025). They extract features without concept annotations and provide patch-wise spatial attributions, but their continuous, potentially overlapping activations do not assign exactly one concept to each location. Moreover, these methods capture which concepts appear and where, but none encodes ***how*** each concept varies across images (*e.g.,* the type of a "wheel") as a separate attribute. Together, these methods leave open the combination of (1) spatial grounding, (2) one concept assignment per location, (3) attributes that capture how each concept varies across images, and (4) no class labels or language supervision.

We ask an essential question: Can a representation with all four properties be learned entirely from frozen, vision-only self-supervised features, without sacrificing the expressive power of the underlying backbone?

To answer this, we introduce **DisParQ** (**Dis**crete **Par**ts with **Q**uantized attributes)[1], a framework that discovers spatially grounded visual concepts from frozen self-supervised representations, assigns exactly one concept to each patch, and captures how each concept varies across images with an attribute, without class labels or language supervision. DisParQ maintains a learnable dictionary of prototype vectors, each representing a global concept. Every patch is assigned to one concept, and only a sparse subset of concepts can activate per image. To preserve fine appearance differences across occurrences of the same concept, we learn a separate attribute codebook per concept. Training proceeds in two stages: we first learn the high-level concepts with continuous residuals, then quantize these residuals into discrete attributes, resulting in a fully discrete bottleneck at inference.

Figure 1 illustrates how DisParQ describes images with spatially grounded part concepts and explains a class prediction. Although only a small set of concepts (at most 16) are active per image, this description costs almost no accuracy: DisParQ matches its frozen backbone on ImageNet (Deng et al., 2009) linear probing (83.2% vs. 83.4%, Figure 5) and stays within 1-2.2 p.p. of it on four fine-grained datasets (Table 3). Among self-supervised methods, it ranks first on most part-quality metrics on PartImageNet (PIN) (He et al., 2022), and its concepts are more consistent across images than language-grounded ones (Table 2). We further confirm this in a user study (Section 4.3).

We also introduce a part-to-part retrieval benchmark, which tests whether a concept names the same part across categories: given a part region, retrieve the same part in other images (Figure 2). DisParQ consistently leads self-supervised baselines (Tables 4 and 19) on PIN and CUB (Wah et al., 2011).

To summarize, our **contributions** are as follows. **(C1)** We propose **DisParQ**, a framework that learns part concepts with all four properties on a frozen self-supervised vision foundation backbone: spatial grounding, one concept per patch, attributes that capture how each concept varies across images, and no class labels or language supervision required. **(C2)** We introduce a part-to-part retrieval benchmark on PartImageNet and CUB that tests whether a concept names the same part across images and categories. **(C3)** We study the design of DisParQ on six frozen backbones and across feature extraction, concept budgets and selection, appearance quantization, decoder depth, gradient routes, and training objectives.

[1]Concept visualizations and code: https://anonymous.4open.science/w/DisParQ-3E73/

Table 1: **Properties of interpretable concept representations.** DisParQ is the only method with all nine: it states *which* part concepts occur, *where* they are and *how* each one varies across images, without any labels. (✓ yes, (✓) partially, × no)

	Training			*Where*		*Which & how*		*Scope*	
Method	No class labels	No text	Frozen backbone	Concept maps	One per patch	Fully discrete	Within-concept attributes	Task-agnostic	Any backbone
ProtoPNet (Chen et al., 2019)	×	✓	×	✓	×	×	×	×	×
PIP-Net (Nauta et al., 2023)	(✓)	✓	×	✓	(✓)	×	×	×	×
ProtoQuant (Janusz et al., 2026)	(✓)	✓	✓	✓	(✓)	(✓)	×	×	✓
PDiscoFormer (Aniraj et al., 2024)	×	✓	(✓)	✓	(✓)	×	(✓)	×	×
LF-CBM (Oikarinen et al., 2023)	×	×	✓	×	×	×	×	×	✓
SALF-CBM (Benou & Riklin-Raviv, 2025)	×	×	✓	✓	×	×	×	×	✓
PatchSAE (Lim et al., 2025)	✓	✓	✓	✓	×	×	×	✓	✓
MSAE (Zaigrajew et al., 2025)	✓	✓	✓	✓	×	×	×	✓	✓
CFM (Wittenmayer et al., 2026)	✓	✓	✓	✓	×	×	×	✓	(✓)
DisParQ (ours)	✓	✓	✓	✓	✓	✓	✓	✓	✓

We make the following **empirical findings**, detailed in Section 4.2. **(F1) A discrete bottleneck costs almost no accuracy.** With only a small set of active concepts ($k \leq 16$) per image, DisParQ stays within 0.2 p.p. of its teacher on ImageNet and within 2.2 p.p. on four fine-grained datasets, using one configuration for all of them (Figure 5 and Table 3). **(F2) Vision-only self-supervision grounds a consistent part vocabulary.** Without any language supervision, the concepts of DisParQ are more consistent across images than language-grounded ones, and DisParQ leads self-supervised baselines on most part-quality metrics (Tables 2 and 3). **(F3) Concepts preserve their identity across categories.** In part-to-part retrieval, DisParQ leads the self-supervised baselines on most metrics, by up to **3.5** mAP on PartImageNet and **5.9** p.p. R@1 on CUB (Tables 4 and 19).

With **DisParQ**, we show that frozen vision foundation models can be interpreted through discrete parts and their attributes, without any human supervision and at almost no cost in accuracy.

2 Related Work

Prototype models classify images by matching patches to learned exemplars (Chen et al., 2019; Nauta et al., 2023), with newer ones using a VFM and quantized prototypes but still classifying through continuous similarity scores with class supervision (Janusz et al., 2026). **Part discovery models** instead learn a small, fixed set of parts, either without labels (Hung et al., 2019; Amir et al., 2023) or from class supervision (van der Klis et al., 2023; Aniraj et al., 2024). DisParQ learns a large shared dictionary without class labels or a task-specific classifier, letting each image select a few concepts to describe its parts. **Concept bottleneck** models predict through human-defined or language-generated concepts (Koh et al., 2020; Oikarinen et al., 2023), including spatial concept maps (Benou & Riklin-Raviv, 2025). Sparse autoencoders (SAEs) learn visual dictionaries from model activations (Lim et al., 2025; Zaigrajew et al., 2025), and CFM grounds concepts spatially and names them through a text vocabulary (Wittenmayer et al., 2026). These SAE dictionaries use continuous, overlapping codes that entangle concept identity with appearance. DisParQ learns concepts from visual features alone, assigns each patch exactly one concept, and encodes concept identity and within-concept attributes separately using integer indices alone (Table 1). **Existing evaluations** measure prototype purity, part alignment, or concept locality and consistency (Nauta et al., 2023; Aniraj et al., 2024; Wittenmayer et al., 2026). Our part-to-part retrieval benchmark tests whether concept descriptions preserve part identity across images and categories by retrieving the same part from all other images (Section 4.1). Appendix A gives the full related-work discussion.

3 DisParQ

DisParQ provides an interpretable, discrete, spatially grounded concept bottleneck on top of a Vision Foundation Model (VFM). An overview of DisParQ is presented in Figure 3. We first extract latent features (Section 3.1), and organize them into a small set of exclusive concept regions to ensure spatial grounding (Section 3.2). Then we encode each region's appearance to represent details of the concepts (Section 3.3). Lastly, a spatial decoder predicts the global representation from these concepts (Section 3.4). Training transforms continuous appearance attributes to discrete codes (Section 3.5). Additional details are provided in Appendix C.

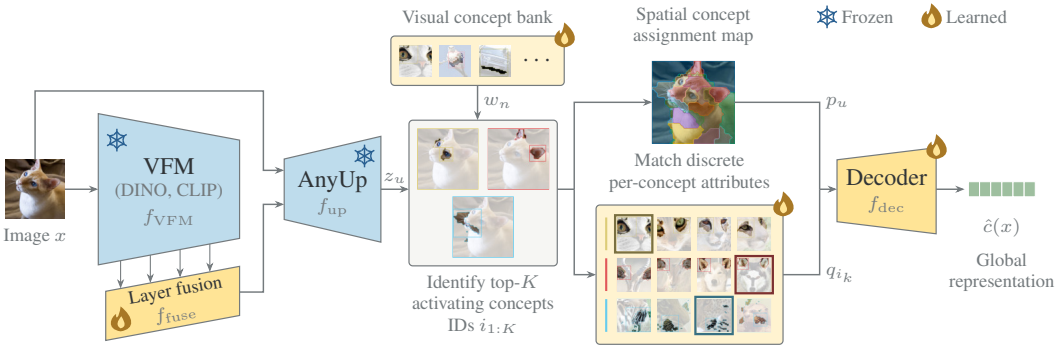


Figure 3: **DisParQ describes images with concepts and attributes.** Fused and upsampled VFM features z_u are matched against prototypes w_n to determine which concepts occur (top-K identities $i_{1:K}$) and where they occur (assignments p_u at positions u). Discrete attributes q_{i_k} encode how each concept appears in the image. The assignment map selects the learned concept embeddings e_n and attributes for the decoder, which predicts $\hat{c}(x)$ by distillation from the VFM's global representation.

3.1 Feature Extraction

We build on a pretrained, frozen VFM encoder f_{VFM}. An image x is passed through it, producing (1) encoded features (patch tokens) that serve as input to DisParQ, and (2) a global image representation $c(x)$ (typically the `cls` token), which is later used as a distillation target. These two outputs serve different purposes. Local features ground concepts in image regions, while the global target encourages their collective description to preserve the image's semantics.

As different details are encoded at different depths of the f_{VFM}, we introduce a lightweight shared-query attention module f_{fuse} that at each location combines features from different encoder blocks. Then, as the feature map is of low resolution, which results in low precision of concept activations, we upsample it with an AnyUp module f_{up} (Wimmer et al., 2026). AnyUp uses the original normalized image to guide feature upsampling, providing a finer grid for concept assignment. As a result we obtain spatial feature vectors z_u, where u is the patch position in the upsampled feature grid.

3.2 Concept Identification and Assignment

Concept identification proceeds in two steps: we select a small set of concepts from the shared bank for each image (Figure 3), and then assign each spatial location to one of those concepts (Figure 4).

Selecting concepts. The shared bank must accommodate diverse visual structures across images. At the same time, any single image should have a compact description. That is why we train a large bank of N concept prototypes, from which we select a small subset of K concepts per image ($N = 1024$, $K = 16$ by default). Each prototype w_n, with $n \in \{1, \dots, N\}$, is a learned reference vector. We compare it with the upsampled spatial features z_u to obtain a soft activation map A_n, and average that map to get concept scores χ_n:

$$\ell_{n,u} = -\|z_u - w_n\|_2^2, \qquad A_{n,u} = \text{softmax}_n(\ell_{n,u}), \qquad \chi_n = \frac{1}{HW} \sum_u A_{n,u}. \tag{1}$$

Here $H \times W$ is the upsampled feature grid size, and the softmax makes all N prototypes compete at each location. During training, Sinkhorn–Knopp balancing (Sinkhorn & Knopp, 1967) adjusts the scores χ_n across images in a batch to encourage balanced prototype usage. We select the K concepts with the highest balanced scores for each image. The bank index $i_k \in \{1, \dots, N\}$ identifies its kth selected concept, with $k \in \{1, \dots, K\}$.

Assigning locations. Having chosen *which* K concepts participate, we then determine *where* they occur. We recompute the softmax over only the selected concepts, so spatial competition is restricted to the image's chosen subset. This produces the post-selection soft maps $\bar{A}_{i_k}$ and hard identity map p shown in Figure 4:

$$\bar{A}_{i_k,u} = \text{softmax}_k(\ell_{i_k,u}), \qquad p_u = i_{\arg\max_k \bar{A}_{i_k,u}}. \tag{2}$$

The strongest response at each location determines p_u, making the concept assignment mutually exclusive. Training and inference details are specified in Appendix C.2.

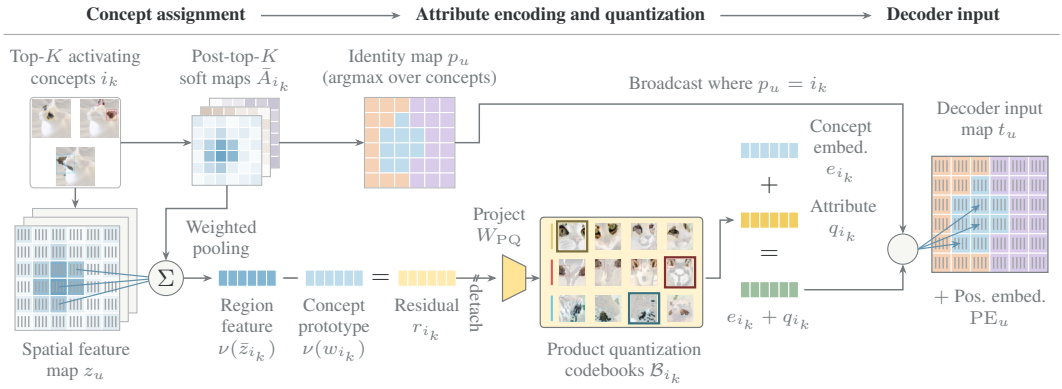


Figure 4: **Separating concept identity and appearance.** Selected-concept maps $\bar{A}_{i_k}$ determine exclusive spatial assignments p_u and pool each concept's region into $\bar{z}_{i_k}$. Its normalized deviation from the concept prototype, r_{i_k}, describes the occurrence relative to the concept. Product codebooks $\mathcal{B}_{i_k}$ quantize its projection. The resulting attribute q_{i_k} is added to the concept embedding e_{i_k} and broadcast to its assigned positions. Adding positional embeddings PE_u forms the decoder tokens t_u.

3.3 Attribute Encoding and Quantization

A concept such as a dog's nose can vary in appearance across images. Describing every appearance difference with a new prototype would fragment the concept bank, whereas ignoring those differences would discard information needed to match the VFM representation. We therefore learn a concept-specific *attribute* that captures *how* each occurrence differs from its concept reference.

The attribute pathway in Figure 4 follows three steps: pooling region features, computing their deviation from the prototype, and quantizing that residual. For each selected concept i_k, the soft map $\bar{A}_{i_k}$ weights the features z_u to form a region mean $\bar{z}_{i_k}$. We normalize both vectors with ν and subtract the prototype from the region mean to compute the residual r_{i_k}:

$$\bar{z}_{i_k} = \frac{\sum_u \bar{A}_{i_k,u} z_u}{\sum_u \bar{A}_{i_k,u}}, \qquad r_{i_k} = \nu(\bar{z}_{i_k}) - \nu(w_{i_k}). \tag{3}$$

Here ν denotes ℓ_2 normalization. The residual r_{i_k} describes the whole region relative to its concept reference, tying appearance information to that concept occurrence. We encode the residual with product quantization (PQ) (Jégou et al., 2011) using concept-specific codebooks $\mathcal{B}_{i_k}$. The result is a discrete appearance code and its quantized vector q_{i_k}, known as an *attribute*. Pooling and quantization details are given in Appendix C.3.

3.4 Concept Decoder

The decoder maps the interpretable concept description to a global image representation for downstream tasks. It combines concept identities, spatial assignments, and appearance attributes to predict the frozen VFM's representation through feature distillation.

The right portion of Figure 4 shows how each spatial token combines the assigned concept's embedding, its appearance attribute, and a positional embedding PE_u. A concept embedding e_n represents each concept in the decoder and is learned independently from the prototype used in previous steps. At each position $u \in \{1, \ldots, HW\}$, the identity p_u selects the concept embedding and attribute:

$$t_u = e_{p_u} + \mathrm{PE}_u + q_{p_u}, \qquad \hat{c}(x) = f_{\mathrm{dec}}\big([t_{\mathrm{CLS}}, t_1, \ldots, t_{HW}]\big). \tag{4}$$

A small transformer model integrates these tokens and predicts the VFM's global representation from the output of its learned decoder CLS token t_{CLS} (separate from the VFM's CLS token).

Every image-dependent input to this decoder passes through the concept description. In the quantized model, the selected identities $i_{1:K}$, assignment map p, and discrete appearance codes completely specify that description. Decoding requires only these discrete quantities, learned lookup tables, and the decoder, making explicit what information survives the bottleneck. The output $\hat{c}(x)$ has the same dimension as the VFM representation and can be used for downstream tasks such as classification and retrieval.

3.5 Training objectives and regime

We train through feature distillation without class labels, text descriptions, or region annotations. For an augmented image x and a sampled affine transformation T, each view $v \in \{x, T(x)\}$ predicts its corresponding frozen VFM target $c(v)$:

$$\mathcal{L}_{\text{global}} = \frac{1}{2} \sum_{v \in \{x, T(x)\}} \left[1 - \cos\big(\hat{c}(v), \text{stopgrad}(c(v))\big)\right], \tag{5}$$

The loss is averaged over images, cos denotes cosine similarity and stopgrad stops gradients. Following PDiscoFormer (Aniraj et al., 2024), we regularize assignments to encourage coherent, confident regions and distinct, active concepts. Affine equivariance encourages spatial responses to follow image content under geometric transformations. Appendix C.5 specifies the losses (Figure 7) and weights (Table 6). Appendix D.3 gives the augmentation and optimization recipe.

Stage 1: discovering concepts. We jointly train feature fusion, concept prototypes, the continuous appearance projection, and the decoder. Continuous attributes provide flexibility during concept discovery, while an auxiliary distillation head encourages pooled region features to preserve global image information. Assignments are hard in the forward pass. **Stage 2: quantizing attributes.** We freeze feature fusion and the prototypes, replace continuous attributes with product-quantized vectors, and train the quantization module, concept embeddings, and decoder through $\mathcal{L}_{\text{global}}$. The auxiliary head is disabled. The VFM and AnyUp remain frozen throughout both stages. Appendices C.3, C.5, and C.6 detail quantization, gradient handling, and the stage transition.

4 Experimental evaluation

In this section, we compare DisParQ with concept dictionaries and prototype models. Section 4.1 describes the evaluation setup, Section 4.2 the quantitative results, and Section 4.3 the qualitative results and how DisParQ explains a prediction. The implementation details for training and evaluating DisParQ as well as the baselines are in Appendices C.6 and D.

4.1 Experimental setup

Datasets. We train one DisParQ model per dataset, without labels. ImageNet-1k (Deng et al., 2009) and Places365 (Zhou et al., 2018) test classification at scale. PartImageNet (He et al., 2022), with masks for 40 part classes, and CUB-200-2011 (Wah et al., 2011), with 15 keypoints per bird, test the part concepts. Stanford Cars (Krause et al., 2013), Stanford Dogs (Khosla et al., 2011) and Oxford Flowers (Nilsback & Zisserman, 2008) test fine-grained classification (Appendix D.1).

Backbone and baselines. The main self-supervised comparison uses frozen DINOv2-B/14 with registers (Oquab et al., 2024; Darcet et al., 2024), except CFM (Wittenmayer et al., 2026), whose released dictionary is built on CLIP-DINOiser (Wysoczańska et al., 2024). We compare DisParQ to (i) sparse autoencoders trained on the same patch tokens: TopK SAE (Gao et al., 2025), PatchSAE (Lim et al., 2025) and MSAE (Zaigrajew et al., 2025), (ii) the concept dictionary of CFM, and (iii) the label-free first stage of the prototype models PIP-Net (Nauta et al., 2023) and ProtoQuant (Janusz et al., 2026). As unranked references, gray rows give PDiscoFormer (Aniraj et al., 2024) and the classifier stage of PIP-Net and ProtoQuant, which use class labels, and the frozen backbone without a bottleneck. For spatial evaluation and part retrieval, the main self-supervised comparison uses $K = 16$ concepts per image, with DisParQ selecting them from $N = 1024$. TopK SAE is trained with this budget. PatchSAE, MSAE and CFM keep their 16 strongest concepts for these evaluations, but use unbudgeted representations for classification and image retrieval. Appendix D states selection rules and protocol exceptions. Appendix F.2 discusses more settings.

Classification. We evaluate the representation that each method passes downstream, which for DisParQ is the CLS token its decoder predicts from the discrete description. For the baselines, the probes read the pooled reconstruction of the sparse autoencoders, the pooled concept activations of CFM, and the prototype scores of PIP-Net and ProtoQuant (see Appendix D.5). We report the top-1 accuracy of a linear probe (LP) and of a weighted 20-nearest-neighbor classifier (kNN).

Part concepts. On PartImageNet, we compare concept maps with part masks: foreground purity (Pur_{fg}), NMI and ARI after mapping each concept to its dominant part (NMI_m, ARI_m) (van der Klis et al., 2023; Aniraj et al., 2024), and the mIoU (He et al., 2022) and FG-ARI (Locatello et al., 2020)

Method	Spatial grounding	Discrete bottleneck	IN-1k ↑	Places ↑
ViT-B/16, supervised	∅	∅	81.1	—
ProtoQuant	(✓)	(✓)	80.1	—
CLIP ViT-B/16	∅	∅	80.2	55.1
DN-CBM	×	×	79.5	55.1
CDM	×	(✓)	79.3	52.6
CFM	(✓)	×	78.9	**55.4**
DINOv2-B/14	∅	∅	83.4	55.4
DisParQ (ours)	✓	✓	**83.2**	55.3

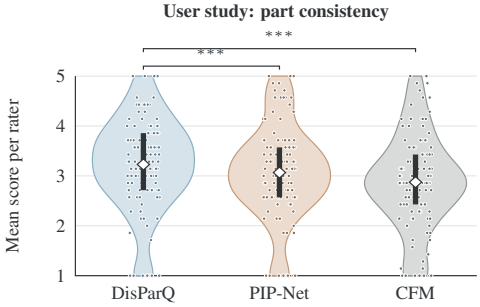


Figure 5: **Left:** Large-scale classification (linear probe, top-1). Marks follow Table 1. Gray rows are unranked references. All methods and their sources are in Table 15 in Appendix F.1. **Right:** DisParQ achieves the highest perceived part consistency in our user study. Diamonds mark means and bars show interquartile ranges. $^{***}p < 0.001$, paired Wilcoxon, Holm-corrected.

of the resulting part segmentation. We add the locality, consistency and impurity of CFM (Wittenmayer et al., 2026), which ask whether a concept covers one part and whether it covers the same part in every image. On CUB, we report PIP purity (Nauta et al., 2023) and the normalized mean error of keypoint regression (Kp NME) (Aniraj et al., 2024). Appendix D.6 defines all metrics in detail.

Part-to-part retrieval. We introduce a benchmark that asks whether a concept names the same part across images and categories. Each annotated part region, a mask on PartImageNet or a visible keypoint on CUB, is described by the histogram of the concepts it contains, and the regions of all other images are ranked by cosine similarity. Appendix D.6 gives the details on the benchmark.

4.2 Quantitative results

We first test whether the discrete bottleneck keeps the backbone's accuracy (F1), then the quality of its part concepts (F2), and finally whether a concept keeps its identity across categories (F3).

Classification. The discrete description costs almost no accuracy, although only concept identities, regions and quantized attributes reach the decoder. On ImageNet-1k, its linear probe reaches **83.2%**, the highest of all concept models in Figure 5 (left) (**+3.7** p.p. over DN-CBM) and only 0.2 p.p. below its DINOv2 backbone, while the vision-language concept models lose 0.7 to 1.3 p.p. to their CLIP backbone. On Places365, it reaches 55.3%, 0.1 p.p. below both its backbone and CFM. Against the self-supervised baselines of Tables 2 and 3, whose best rows use the same DINOv2 backbone, its LP is **3.6** p.p. higher on PartImageNet and **4.0** to **9.3** p.p. higher on CUB, Cars and Dogs. None of the other methods has concepts that are both exclusive and discrete (Appendix F.1). Quantized attributes retain appearance lost by concepts alone; removing them costs 12.2 p.p. in LP (Appendix E.3).

Part concepts on PartImageNet. Among self-supervised methods, DisParQ is best on **7 of the 10** metrics of Table 2 and second on two more, Pur_{fg} and FG-ARI, where the label-free stage of ProtoQuant leads. Its largest margins are on ARI_m (**62.3** vs. 56.3), locality (**36.4** vs. 30.5) and consistency, where its concepts are more consistent than those of the language-grounded CFM (**16.3** vs. 12.4). Its kNN accuracy is **4.8** p.p. above the best baseline, within 0.2 p.p. of the frozen backbone. It also exceeds the supervised PDiscoFormer on every part and grounding metric except FG-ARI.

Fine-grained datasets. On the four fine-grained datasets, DisParQ is best on **all 10** metrics of Table 3 among self-supervised methods (tied on Flowers LP). Its kNN accuracy is **16.7** to **19.8** p.p. above the best baseline on CUB, Cars and Dogs. On CUB, its concepts also localize the keypoints best, with PIP purity **54.5** vs. 37.3 and keypoint error **9.36** vs. 9.87 (Appendix D.6 explains how the purity window is placed on exclusive maps). The supervised PDiscoFormer[2] scores higher on the CUB clustering metrics and on keypoint error, and lower on PIP purity (Table 16 in Appendix F.3).

Cross-category part retrieval. The concepts of DisParQ keep their identity across categories. Among self-supervised methods, DisParQ is best on **7 of the 8** metrics of Table 4 and second on part-to-part R@5. On PartImageNet, it leads part-to-part retrieval with an R@1 of **79.7%** and an mAP of **30.5**, vs. 79.2% and 27.0 for the best baselines, and image-to-image retrieval with an mAP of **76.1** vs. 61.8. An oracle based only on part-label co-occurrence groups reaches an R@1 of 43.9%

[2]Checkpoints released only for CUB and Flowers; see Appendix F.4 for our reproduction on Cars and Dogs.

Table 2: **Part concepts on PartImageNet** (val, as % except Imp.). **Best** and second best among self-supervised methods. Subscripts: gap to the best. Gray: unranked references. K/N: concepts per image/dictionary size. Protocol: Appendix D.5. DisParQ shows superior performance on 7/10.

			Downstream		Part quality					Concept grounding		
Method	Backbone	K/N	LP↑	kNN↑	Pur_{fg}↑	NMI_m↑	ARI_m↑	mIoU↑	FG-ARI↑	Loc.↑	Cons.↑	Imp.↓
Supervised (reference)												
PDiscoFormer	DINOv2-B/14	50/50	89.8	88.1	54.9	48.4	37.9	29.1	28.8	34.9	2.0	0.505
$\text{PIP-Net}_{\text{stage 2}}$	DINOv2-B/14	16/768	87.5	86.8	59.3	37.3	11.5	24.9	18.9	25.2	6.4	0.437
$\text{ProtoQuant}_{\text{stage 2}}$	DINOv2-B/14	16/2048	87.8	89.5	71.7	55.6	30.0	26.5	19.5	13.1	4.8	0.513
Frozen backbone, no concept bottleneck (reference)												
CLS token	DINOv2-B/14	N/A	89.3	88.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Patch tokens, mean-pooled	DINOv2-B/14	N/A	85.7	79.3	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Self-supervised												
CFM	CLIP-DI-B/16	16/8192	72.6	69.2	54.7	51.4	24.5	22.7	12.5	22.9	12.4	0.378
TopK SAE	DINOv2-B/14	16/2048	80.5	73.5	68.8	58.0	36.0	24.8	16.1	20.6	9.8	0.462
PatchSAE	DINOv2-B/14	16/49152	84.1	77.2	76.2	60.1	34.9	25.0	17.0	24.7	4.4	0.455
MSAE	DINOv2-B/14	16/8192	85.6	79.5	59.0	50.3	23.5	20.5	12.1	18.0	8.1	0.460
$\text{PIP-Net}_{\text{stage 1}}$	DINOv2-B/14	16/768	76.9	71.5	68.5	54.2	34.3	29.0	19.1	30.5	10.2	**0.376**
$\text{ProtoQuant}_{\text{stage 1}}$	DINOv2-B/14	16/2048	85.2	83.9	**82.1**	68.7	56.3	33.7	**28.4**	18.1	7.3	0.514
DisParQ (ours)	DINOv2-B/14	16/1024	**89.2**$_{+3.6}$	**88.7**$_{+4.8}$	81.5$_{-0.6}$	**69.8**$_{+1.1}$	**62.3**$_{+6.0}$	**35.9**$_{+2.2}$	26.3$_{-2.1}$	**36.4**$_{+5.9}$	**16.3**$_{+3.9}$	0.385$_{+0.009}$

Table 3: **Fine-grained datasets** (test, as % except Kp NME; CFM on CLIP-DINOiser). Marks as in Table 2. For PDiscoFormer some scores are reproduced as in Appendix F.4. *4000 on CUB. Protocol: Appendix D.5. DisParQ consistently outperforms baseline methods on all but one metric.

		CUB-200-2011				Stanford Cars		Stanford Dogs		Oxford Flowers	
Method	K/N	LP↑	kNN↑	PIP pur.↑	Kp NME↓	LP↑	kNN↑	LP↑	kNN↑	LP↑	kNN↑
Supervised (reference)											
PDiscoFormer	K/K	86.8	74.8	40.4	2.87	N/A	N/A	N/A	N/A	99.6	99.7
$\text{PIP-Net}_{\text{stage 2}}$	16/768	88.9	88.9	47.0	10.16	91.4	87.3	87.1	85.2	99.6	99.7
$\text{ProtoQuant}_{\text{stage 2}}$	16/2048*	84.9	83.0	25.2	12.16	91.3	86.3	86.0	79.4	99.6	99.6
Frozen backbone, no concept bottleneck (reference)											
CLS token	N/A	90.4	86.7	N/A	N/A	93.2	85.6	88.6	86.8	99.7	99.7
Self-supervised											
CFM	16/8192	54.8	42.8	16.1	12.84	71.0	61.3	56.0	52.5	90.2	88.8
TopK SAE	16/2048	36.9	24.3	8.7	10.92	41.9	22.7	61.3	41.1	99.4	99.0
PatchSAE	16/49152	62.0	33.4	6.9	19.12	74.6	35.8	77.6	53.5	99.3	99.0
MSAE	16/6144	73.5	42.2	15.1	12.21	80.3	41.7	79.4	55.2	99.6	99.4
$\text{PIP-Net}_{\text{stage 1}}$	16/768	77.8	68.4	37.3	9.87	82.3	63.6	77.8	69.1	99.1	98.4
$\text{ProtoQuant}_{\text{stage 1}}$	16/2048*	80.1	67.7	24.6	11.59	87.0	60.2	82.2	66.9	**99.7**	99.5
DisParQ (ours)	16/1024	**89.4**$_{+9.3}$	**85.5**$_{+17.1}$	**54.5**$_{+17.2}$	**9.36**$_{-0.51}$	**91.0**$_{+4.0}$	**83.4**$_{+19.8}$	**87.8**$_{+5.6}$	**85.8**$_{+16.7}$	**99.7**	**99.7**$_{+0.2}$

(Appendix F.6). Against the supervised PDiscoFormer, DisParQ leads **4 of the 6** retrieval metrics at each of its three part counts, including part-to-part R@1 (**79.7%** vs. at most 71.8%) and image-to-image mAP (**76.1** vs. at most 70.2), while PDiscoFormer reaches a higher part-to-part mAP. On CUB, DisParQ also leads R@1 (**36.8%** vs. 30.9%; see Table 19 in Appendix F.6).

Backbones and design choices. DisParQ is not tied to one pretraining objective. Across seven configurations of six frozen backbones, self-supervised (DINOv2 and DINOv3 (Siméoni et al., 2025)) and vision-language (CLIP (Radford et al., 2021), SigLIP2 (Tschannen et al., 2025), TIPSv2 (Cao et al., 2026) and CLIP-DINOiser), it stays within 2 p.p. of each backbone's own linear probe on **6 of the 7** and within 3.5 p.p. on CLIP-DINOiser (see Figure 8 in Appendix E.1). At the same budget of 16/1024, DINOv2 gives the best concepts: it leads the other six configurations on **7 of the 8** concept metrics of Table 2, all but impurity, and on part-to-part retrieval. Appendix E presents our design choices for the bank size N, the per-image budget K and the losses in detail.

4.3 Qualitative results and interpretability

Part maps. Figure 6 compares the concept maps of DisParQ with those of ProtoQuant, CFM and the supervised PDiscoFormer. DisParQ follows the annotated parts and stops at the object boundary: its non-overlapping concept regions match distinct parts of the object, such as the head, body, wings, feet and tail of the kite (Figure 6a, first row), where the baselines cover larger regions with one concept or spread into the background. For a part such as a sail, one concept of DisParQ marks it in all four images, while a baseline's best-matching concept misses some of them or covers the whole object. Appendix G gives further examples on additional datasets, and failure cases.

Table 4: **Retrieval on PartImageNet** (val, as %). Marks as in Table 2. Dictionary methods appear at our N=1024 and their best N. ‡At the method's native resolution, on another query set (Appendix F.6). The controls and PCA are in Table 19. DisParQ is the best on all but one metric.

		Accuracy		Image → image			Part → part		
Method	K/N	LP↑	kNN↑	R@1↑	R@5↑	mAP↑	R@1↑	R@5↑	mAP↑
Supervised (reference)									
PDiscoFormer‡	50/50	89.8	88.1	78.4	93.4	69.3	71.8	89.0	45.9
PIP-Net$_{\text{stage 2}}$	16/768	87.5	86.8	77.6	91.9	64.2	65.4	85.2	21.1
ProtoQuant$_{\text{stage 2}}$	16/2048	87.8	89.5	82.3	93.8	76.6	62.6	89.6	15.5
Frozen backbone, no concept bottleneck (reference)									
CLS token	N/A	89.3	88.9	81.0	92.9	75.1	N/A	N/A	N/A
Patch tokens, mean-pooled	N/A	85.7	79.3	57.5	82.2	44.7	N/A	N/A	N/A
Self-supervised									
CFM‡	16/8192	72.6	69.2	47.0	75.4	34.6	31.6	68.8	16.0
TopK SAE	16/1024	78.1	73.0	52.6	79.2	42.7	56.2	86.5	17.2
TopK SAE	16/2048	80.5	73.5	53.8	77.5	43.3	58.3	86.9	18.1
PatchSAE	16/1024	82.9	76.5	56.8	80.8	44.6	41.9	83.6	11.8
PatchSAE	16/49152	84.1	77.2	56.8	81.1	44.4	48.6	82.9	8.0
MSAE	16/1024	85.2	79.9	58.9	81.7	46.1	41.4	82.2	12.5
MSAE	16/8192	85.6	79.5	58.3	82.3	45.4	41.2	79.9	12.6
PIP-Net$_{\text{stage 1}}$	16/768	76.9	71.5	49.1	78.6	36.3	66.6	87.7	27.0
ProtoQuant$_{\text{stage 1}}$	16/2048	85.2	83.9	72.5	90.4	61.8	79.2	**92.8**	22.7
DisParQ (ours)	16/1024	**89.2**$_{+3.6}$	**88.7**$_{+4.8}$	**80.6**$_{+8.1}$	**92.9**$_{+2.5}$	**76.1**$_{+14.3}$	**79.7**$_{+0.5}$	92.4$_{-0.4}$	**30.5**$_{+3.5}$

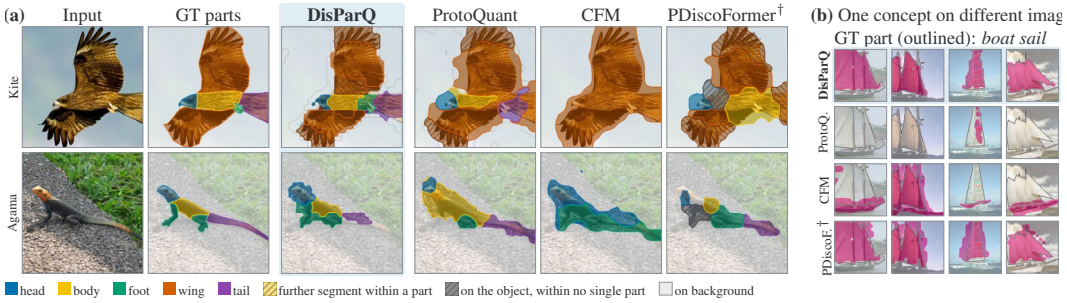


Figure 6: **Part concepts on PartImageNet.** *(a)* Part concept maps for DisParQ vs. baselines. DisParQ assigns each location exactly one concept, with no overlap, following object parts, while the baselines' overlapping concepts merge parts or spread into the background. *(b)* Each method's concept most consistent with the outlined sail, on four images. † trained with class labels.

Interpretability. DisParQ explains a prediction through its parts (Figure 1). Removing a concept from the discrete description and decoding the rest again measures how much the predicted class depends on that part, an occlusion test (Zeiler & Fergus, 2014) on parts instead of pixels. The four largest contributions account for 56-100% of the positive logit drops, and each retrieves the same part in other images, so a prediction reads as a short list of recognizable parts with their weights (details in Appendix G.5).

User study. We asked 120 participants to evaluate concepts of DisParQ, PIP-Net, and CFM on CUB, Cars, and PartImageNet. For each prototype, they rated how consistently it captures the same object part (1–5, where 5 is most consistent) and identified which of four regions in a new image it corresponds to. DisParQ prototypes were rated significantly more consistent than both baselines (Figure 5, right) and were matched to the correct part most often (69% vs. 60% for PIP-Net and 53% for CFM, random chance is 25%). Details are in Appendix F.8.

5 Conclusion

We presented DisParQ, which describes an image from a frozen self-supervised Vision Foundation Model (VFM) by a few discrete part concepts, one per location, each with a quantized attribute, learned without class labels or language. This description preserves the backbone's accuracy, gives part concepts that are more consistent across images than those of language-grounded and sparse-autoencoder dictionaries, and keeps a concept's identity across categories, so a prediction reads as a short list of parts and their contributions. In future work, the discrete concepts could support searching by part and appearance, and correcting a prediction by removing or replacing concepts.

ETHICS STATEMENT

We use existing public datasets and pretrained models for model training and benchmark evaluation, without using private datasets. To assess interpretation quality, we also conduct a user study on Clickworker that was compliant with our organization procedures and regulations. Participants were informed about the study and compensated for their participation, and we collected no identifying information.

DisParQ makes the representations of vision foundation models easier to inspect by expressing them through discrete part concepts. Although these concepts can reveal patterns used by a model, they do not by themselves establish that its predictions are fair, reliable, or causally grounded. The method inherits the biases and limitations of its frozen backbone and training data, so even visually coherent concepts may encode spurious correlations. Use in consequential applications would therefore require validation in the intended setting, including an assessment of errors and bias.

REPRODUCIBILITY STATEMENT

To support reproduction of our experiments, we describe the model and training objectives in Section 3 and provide a highly detailed, technical description in Appendix C. Appendix D details the datasets and splits, pretrained backbones, evaluation metrics, and baseline protocols. Throughout these descriptions, we distinguish label-free concept learning from the use of labels for downstream probes and evaluation. We provide an anonymous link for concept visualizations for our model and source code here: https://anonymous.4open.science/w/DisParQ-3E73/. We will publicly release the code, experiment configurations, and trained checkpoints upon acceptance.

AI USE STATEMENT

We used OpenAI Codex and Anthropic Claude Code to assist with language editing, literature searches, and experiment orchestration. These tools also supported code editing and debugging, as well as the preparation of figures from experimental results, under full human supervision. In particular, LLM-based agents were used to verify consistency between the paper and the experimental code, providing an additional layer of validation. We did not use AI image generation, and no AI-generated data were used in the experiments. The authors take responsibility for the final manuscript and code, as well as the experimental results and their interpretation.

REFERENCES

Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. On the effectiveness of ViT features as local semantic descriptors. In *ECCVW*, 2023.

Ananthu Aniraj, Cassio F. Dantas, Dino Ienco, and Diego Marcos. PDiscoFormer: Relaxing part discovery constraints with vision transformers. In *ECCV*, 2024.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Itay Benou and Tammy Riklin-Raviv. Show and tell: Visually explainable deep neural nets via spatially-aware concept bottleneck models. In *CVPR*, 2025.

Bingyi Cao, Koert Chen, Kevis-Kokitsi Maninis, Kaifeng Chen, Arjun Karpur, Ye Xia, Sahil Dua, Tanmaya Dabral, Guangxing Han, Bohyung Han, Joshua Ainslie, Alex Bewley, Mithun Jacob, René Wagner, Washington Ramos, Krzysztof Choromanski, Mojtaba Seyedhosseini, Howard Zhou, and André Araujo. TIPSv2: Advancing vision-language pretraining with enhanced patch-text alignment. In *CVPR*, 2026.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K. Su. This looks like that: Deep learning for interpretable image recognition. In *NeurIPS*, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023.

Subhabrata Choudhury, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Unsupervised part discovery from contrastive reconstruction. In *NeurIPS*, 2021.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable ProtoPNet: An interpretable image classifier using deformable prototypes. In *CVPR*, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Viktar Dubovik, Łukasz Struski, Jacek Tabor, and Dawid Rymarczyk. SIDE: Sparse information disentanglement for explainable artificial intelligence. *arXiv preprint arXiv:2507.19321*, 2025.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes challenge 2012 (VOC2012) results, 2012.

Thomas Fel, Ekdeep Singh Lubana, Jacob S. Prince, Matthew Kowal, Victor Boutin, Isabel Papadimitriou, Binxu Wang, Martin Wattenberg, Demba E. Ba, and Talia Konkle. Archetypal SAE: Adaptive and stable dictionary learning for concept extraction in large vision models. In *ICML*, 2025.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *ICLR*, 2025.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

Ju He, Shuo Yang, Shaokang Yang, Adam Kortylewski, Xiaoding Yuan, Jie-Neng Chen, Shuai Liu, Cheng Yang, Qihang Yu, and Alan Yuille. PartImageNet: A large, high-quality dataset of parts. In *ECCV*, 2022.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Zixuan Huang and Yin Li. Interpretable and accurate fine-grained recognition via region grouping. In *CVPR*, 2020.

Robert Huben, Hoagy Cunningham, Logan Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *ICLR*, 2024.

Wei-Chih Hung, Varun Jampani, Sifei Liu, Pavlo Molchanov, Ming-Hsuan Yang, and Jan Kautz. SCOPS: Self-supervised co-part segmentation. In *CVPR*, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *ICLR*, 2017.

Mikołaj Janusz, Adam Wróbel, Bartosz Zieliński, and Dawid Rymarczyk. ProtoQuant: Quantization of prototypical parts for general and fine-grained image classification. In *BMVC*, 2026.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 33(1), 2011.

Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford Dogs. In *CVPRW*, 2011.

Sunnie S. Y. Kim, Nicole Meister, Vikram V. Ramaswamy, Ruth Fong, and Olga Russakovsky. HIVE: Evaluating the human interpretability of visual explanations. In *ECCV*, 2022.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *ICML*, 2020.

Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *NeurIPS*, 2011.

Jonathan Krause, Michael Stark, Jia Deng, and Fei-Fei Li. 3D object representations for fine-grained categorization. In *ICCVW*, 2013.

Hyesu Lim, Jinho Choi, Jaegul Choo, and Steffen Schneider. Sparse autoencoders reveal selective remapping of visual concepts during adaptation. In *ICLR*, 2025.

Yoseph Linde, Andrés Buzo, and Robert M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1), 1980.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *NeurIPS*, 2020.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Chiyu Ma, Jon Donnelly, Wenjun Liu, Soroush Vosoughi, Cynthia Rudin, and Chaofan Chen. Interpretable image classification with adaptive prototype-based vision transformers. In *NeurIPS*, 2024.

Sachit Menon and Carl Vondrick. Visual classification via description from large language models. In *ICLR*, 2023.

Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. SPair-71k: A large-scale benchmark for semantic correspondence. *arXiv preprint arXiv:1908.10543*, 2019.

Meike Nauta, Ron van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *CVPR*, 2021.

Meike Nauta, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. PIP-Net: Patch-based intuitive prototypes for interpretable image classification. In *CVPR*, 2023.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *ICLR*, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024.

Mateusz Pach, Koryna Lewandowska, Jacek Tabor, Bartosz Zieliński, and Dawid Rymarczyk. LucidPPN: Unambiguous prototypical parts network for user-centric interpretable computer vision. In *ICLR*, 2025.

Konstantinos P. Panousis, Dino Ienco, and Diego Marcos. Coarse-to-fine concept bottleneck models. In *NeurIPS*, 2024.

Konstantinos Panagiotis Panousis, Dino Ienco, and Diego Marcos. Sparse linear concept discovery models. In *ICCVW*, 2023.

Katharina Prasse, Patrick Knab, Sascha Marton, Christian Bartelt, and Margret Keuper. DCBM: Data-efficient visual concept bottleneck models. In *ICML*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Sukrut Rao, Sweta Mahajan, Moritz Böhle, and Bernt Schiele. Discover-then-name: Task-agnostic concept bottlenecks via automated concept discovery. In *ECCV*, 2024.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), 2019.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16, 2022.

Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. ProtoPShare: Prototypical parts sharing for similarity discovery in interpretable image classification. In *KDD*, 2021.

Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. Interpretable image classification with differentiable prototypes assignment. In *ECCV*, 2022.

Mikołaj Sacha, Bartosz Jura, Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Interpretability benchmark for evaluating spatial misalignment of prototypical parts explanations. In *AAAI*, 2024.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2), 1967.

Łukasz Struski, Dawid Rymarczyk, and Jacek Tabor. InfoDisent: Explainability of image classification models by information disentanglement. *arXiv preprint arXiv:2409.10329*, 2024.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Hugues Turbé, Mina Bjelogrlic, Gianmarco Mengaldo, and Christian Lovis. ProtoS-ViT: Visual foundation models for sparse self-explainable classifications. In *NeurIPSW*, 2024.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2017.

Robert van der Klis, Stephan Alaniz, Massimiliano Mancini, Cassio F. Dantas, Dino Ienco, Zeynep Akata, and Diego Marcos. PDiscoNet: Semantically consistent part discovery for fine-grained recognition. In *ICCV*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Jiaqi Wang, Huafeng Liu, Xinyue Wang, and Liping Jing. Interpretable image recognition by constructing transparent embedding space. In *ICCV*, 2021.

Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. AnyUp: Universal feature upsampling. In *ICLR*, 2026.

Kai Wittenmayer, Sukrut Rao, Amin Parchami-Araghi, Bernt Schiele, and Jonas Fischer. CFM: Language-aligned concept foundation model for vision. In *ECCV*, 2026.

Monika Wysoczańska, Oriane Siméoni, Michaël Ramamonjisoa, Andrei Bursuc, Tomasz Trzciński, and Patrick Pérez. CLIP-DINOiser: Teaching CLIP a few DINO tricks for open-vocabulary semantic segmentation. In *ECCV*, 2024.

Mengqi Xue, Qihan Huang, Haofei Zhang, Jingwen Hu, Jie Song, Mingli Song, and Canghong Jin. ProtoPFormer: Concentrating on prototypical parts in vision transformers for interpretable image recognition. In *IJCAI*, 2024.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *CVPR*, 2023.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *ICLR*, 2023.

Vladimir Zaigrajew, Hubert Baniecki, and Przemysław Biecek. Interpreting CLIP with hierarchical sparse autoencoders. In *ICML*, 2025.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30, 2022.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE TPAMI*, 40(6), 2018.

APPENDIX

This appendix provides additional details and results for DisParQ. We first expand the discussion of related work (Appendix A) and discuss limitations (Appendix B). The architecture and training procedure are described in detail next (Appendix C). Then we present the experimental protocol, including the datasets, backbones, training procedures, and evaluation metrics used for DisParQ and the baselines (Appendix D). We examine each component through ablations (Appendix E) and provide extended results, including the full main-paper tables, a qualitative analysis of quantized attributes, and user-study details (Appendix F). Lastly, we show qualitative results across all five datasets and illustrate how DisParQ explains predictions (Appendix G), before describing how learned concepts can be named using a fixed vocabulary (Appendix H).

Table of contents

A Related work . . . 17

B Limitations . . . 17

C Architecture and training details . . . 18

C.1 Feature extraction . . . 18

C.2 Concept identification and assignment . . . 19

C.3 Attribute encoding and quantization . . . 20

C.4 Concept decoder . . . 21

C.5 Training objectives and regime . . . 21

C.6 Training stages and optimization . . . 24

D Experimental protocol . . . 25

D.1 Datasets and splits . . . 25

D.2 Backbones . . . 25

D.3 Training DisParQ . . . 26

D.4 Training the baselines . . . 27

D.5 Evaluation protocol . . . 29

D.6 Metrics . . . 30

E Ablation studies . . . 33

E.1 Feature extraction . . . 33

E.2 Concept bank, selection, and maintenance . . . 35

E.3 Appearance attributes and quantization . . . 37

E.4 Concept decoder . . . 39

E.5 Gradient routes . . . 40

E.6 Training objectives . . . 41

F Additional results . . . 43

F.1 Large-scale classification in full . . . 43

F.2 Full PartImageNet comparison 44
F.3 Full CUB concept metrics 47
F.4 PDiscoFormer on Cars and Dogs 47
F.5 ProtoQuant and PIP-Net: frozen vs. fine-tuned backbone 47
F.6 Retrieval results 48
F.7 What the quantized attributes keep 51
F.8 User study details and results 51

G Qualitative results 53
G.1 Interactive Concept Visualizer 53
G.2 PartImageNet 56
G.3 Fine-grained datasets 60
G.4 Part-to-part retrieval 62
G.5 Class evidence of DisParQ and interpretability 64
G.6 What quantization keeps 70
G.7 How one concept varies across images 76

H Automatic alignment of concept names 81
H.1 Concept descriptions and alignment targets 81
H.2 Concept descriptions to names 81
H.3 Evaluating the retrieved names 82
H.4 Results 83
H.5 Qualitative examples 83

A Related work

Prototypical parts and part discovery. ProtoPNet (Chen et al., 2019) introduced "this looks like that" reasoning: an image is classified by how closely its patches match prototypes learned for each class. Later work shares prototypes across classes to build compact vocabularies (Nauta et al., 2021; Rymarczyk et al., 2021; 2022; Nauta et al., 2023). Others refine the geometry of the prototype space (Wang et al., 2021; Donnelly et al., 2022) or separate color from other visual features (Pach et al., 2025). Recent variants build on Vision Transformers (Xue et al., 2024; Ma et al., 2024), including frozen foundation backbones (Struski et al., 2024; Turbé et al., 2024; Dubovik et al., 2025; Janusz et al., 2026). Among these, ProtoQuant (Janusz et al., 2026) also quantizes its prototypes into a codebook, but still classifies from continuous similarity scores. Part discovery methods segment objects into parts, either without any labels (Hung et al., 2019; Choudhury et al., 2021; Amir et al., 2023) or with class labels as the only signal (Huang & Li, 2020; van der Klis et al., 2023; Aniraj et al., 2024). Prototypes are learned to serve a classifier, and discovered parts form a small, fixed set per category or dataset. DisParQ instead learns, without class labels, a large shared dictionary from which each image selects a few concepts, assigns each patch exactly one of them, and captures how each of them varies across images with an attribute, making a frozen vision foundation model interpretable at the level of parts (Table 1).

Concept bottlenecks and concept dictionaries. Concept bottleneck models (CBMs) make predictions through a layer of human-defined concepts (Koh et al., 2020). Recent CBMs let a language model write the concept list and score it with a vision-language model (Yuksekgonul et al., 2023; Oikarinen et al., 2023; Yang et al., 2023), and spatial variants score concepts on image regions (Panousis et al., 2024) or as concept maps (Benou & Riklin-Raviv, 2025). Sparse autoencoders (SAEs) instead learn an unnamed, overcomplete dictionary from a model's activations. They have recently been used to interpret language models (Huben et al., 2024; Gao et al., 2025) and have since been applied to vision: PatchSAE (Lim et al., 2025) trains them on the patch tokens of a vision transformer, and later variants improve the dictionary itself (Zaigrajew et al., 2025; Fel et al., 2025). DN-CBM (Rao et al., 2024) names SAE concepts with CLIP, and the concept foundation model CFM (Wittenmayer et al., 2026) builds a spatially grounded concept dictionary on a language-aligned backbone and names its concepts from a predefined text vocabulary. Most CBMs name their concepts in advance, and SAE codes are continuous weights over concepts that overlap at each patch, which entangles a part's identity with its appearance. DisParQ learns its concepts from visual features alone, assigns each patch exactly one concept, separates a part's identity from how it varies across images (its attribute), and describes an image by at most 16 active concepts in integer indices alone, the budget at which we compare all dictionaries (Section 4.1).

Evaluating part concepts. Part concepts are usually scored by comparing each concept's map with part annotations: through the purity of prototypes on annotated keypoints (Nauta et al., 2023), keypoint regression and clustering agreement with annotated parts (van der Klis et al., 2023; Aniraj et al., 2024), or the locality and consistency of concept maps (Wittenmayer et al., 2026). A separate benchmark tests whether a prototype's activation map marks the image region that actually drives it (Sacha et al., 2024), and semantic correspondence matches keypoints between two given images, usually of the same category (Min et al., 2019; Amir et al., 2023). Our part-to-part retrieval benchmark instead starts from one annotated part region and asks whether its description finds the same part among the regions of all other images, across categories (Section 4.1 and Appendix D.6).

B Limitations

In this section we discuss the limitations in our proposed method and study. Part annotations may be coarser than the learned concepts, but over-segmentation and incorrect semantic grouping can also cause failures (Appendix G.2). The frozen backbone passes on its features and biases, including part of the class structure of the attributes (Appendix G.7). Contextual backbone features and soft pooling can carry information from outside a displayed hard region, so spatial assignments do not establish pixel-level causal locality. Concepts remain unnamed without examples or alignment (Appendix H). They cover the whole image, including background, with the same budget of $K{=}16$ concepts regardless of its number of parts (Appendix E.2).

C ARCHITECTURE AND TRAINING DETAILS

This section specifies details of DisParQ as introduced in Section 3, following the same narrative structure. It describes the canonical form used in the main experiments. Differences for studies with alternative configurations are described in the relevant sections.

The default DisParQ configuration uses frozen DINOv2-B/14 (Oquab et al., 2024) with registers (Darcet et al., 2024), a bank of $N = 1024$ concepts, $K = 16$ selected concepts per image, and private 16×64 product quantization. Every patch is assigned to one concept. Table 5 recaps the principal symbols and dimensions used throughout this section. Spatial positions are flattened in row-major order. Batch statistics refer to the local device batch unless stated otherwise.

Table 5: Notation and dimensions for the canonical model. Feature width D and decoder width d are distinct quantities, both equal to 768 here. K counts concepts and G counts attribute groups.

Symbol	Meaning	Value / shape
D, d	Feature width, decoder width	768, 768
B, b, u	Local batch size, image index, spatial index	16; $1 \le b \le B$; $1 \le u \le HW$
$z_u, c(x)$	Spatial features, global target	$\mathbb{R}^D$
w_n, e_n	Concept prototype, decoder value	$\mathbb{R}^D$, $\mathbb{R}^d$
N, K	Bank size, selected concepts	1024, 16
i_k, s_u, p_u	Global ID, local owner, spatial global ID	$p_u = i_{s_u}$
$\chi_{b,n}, \psi_n$	Selection score, full-bank usage distribution	Scalars
$A, \tilde{A}$	Pre- and post-top-K soft maps	$N \times HW$, $K \times HW$
$H \times W$	Shared assignment and decoder grid	32×32
$\bar{z}_{i_k}, r_{i_k}, q_{i_k}$	Pooled feature, residual, attribute	$\mathbb{R}^D$, $\mathbb{R}^D$, $\mathbb{R}^d$
$\mathcal{B}, j_{i_k,g}$	Private codebooks, codeword index	$N \times G \times M \times (d/G)$; $0 \le j < M$
G, M	PQ groups, entries per group	16, 64
ρ, η	Usage EMA, selection dual	$\mathbb{R}^N$ each
$\widetilde{Z}, V_m$	Area-scaled and modulated region features	$D \times K$
$W_{\text{cont}}, W_{\text{PQ}}$	Continuous and quantization projections	$d \times D$ each
PE, t_u	Position table, decoder token	$HW \times d$, $\mathbb{R}^d$

C.1 FEATURE EXTRACTION

Input images have size 224×224 and 3 RGB channels. The frozen DINOv2-B/14 encoder (Oquab et al., 2024) produces width-$D = 768$ patch features on a native 16×16 grid (14×14 patch size). Its final layer-normalized CLS token is the global distillation target $c(x) \in \mathbb{R}^D$. CLS and register tokens are removed from the spatial sequence. The backbone and the pretrained upsampler remain in evaluation mode throughout training. Backbone substitutions and their target definitions are described in Appendix E.1.

To combine information across depths while retaining spatial correspondence, the lightweight attention pooling operator f_{fuse} consumes raw patch outputs $h_{\lambda,\xi} \in \mathbb{R}^D$ from the last four blocks of the encoder. We restrict fusion to these later blocks to favor semantic features and reduce the opportunity for concepts to organize around positional cues from earlier layers. A shared bias-free key projection $W_{\text{key}} \in \mathbb{R}^{64 \times D}$ and a learned query $\omega^{\text{fuse}} \in \mathbb{R}^{64}$ form four 16-dimensional heads using scaled dot-product attention (Vaswani et al., 2017):

$$\alpha_{a,\lambda,\xi} = \text{softmax}_\lambda \left(\frac{(\omega_a^{\text{fuse}})^\top (W_{\text{key}} h_{\lambda,\xi})_a}{\sqrt{16}} \right), \qquad z_\xi^{\text{fuse}} = \frac{1}{4} \sum_{a=1}^{4} \sum_{\lambda=1}^{4} \alpha_{a,\lambda,\xi} h_{\lambda,\xi}. \tag{6}$$

Here ξ indexes the native patch grid, $\lambda \in \{1, \dots, 4\}$ indexes the selected encoder blocks, and $a \in \{1, \dots, 4\}$ indexes attention heads. Each head mixes full D-dimensional values. There is no value/output projection, extra normalization, residual connection, or positional encoding in fusion. It has $64D + 64$ parameters.

The query is initialized from a truncated normal with standard deviation 0.02. Throughout this appendix, truncated-normal initialization uses mean zero and absolute truncation bounds $[-2, 2]$,

as in the default PyTorch initializer. The key projection entries are uniform on $[-D^{-1/2}, D^{-1/2}]$. Removing fusion leaves only the final-block patch features, as studied in Appendix E.1.

Image-guided upsampling. To represent finer concept boundaries than the native 16×16 grid permits, we upsample the fused features with image guidance. The multi-backbone AnyUp model (Wimmer et al., 2026), using its second released pretrained weight set, receives the ImageNet-normalized image and the fused feature map. It mixes unprojected feature values and preserves width D, producing $z_u \in \mathbb{R}^D$ on a 32×32 grid. Its computation uses float32 before restoring the feature dtype. AnyUp's parameters are frozen, but stage 1 propagates gradients through its feature input to fusion. Stage 2 evaluates the frozen feature path without gradients, because fusion is also frozen in that stage.

A less computationally expensive alternative is bilinear interpolation of the fused features to the same grid, with pixel-center coordinates (zero-based output coordinate $o \in \{0, \dots, 31\}$ maps to source coordinate $(o + \frac{1}{2})/2 - \frac{1}{2}$ along each axis, using boundary replication). Appendix E.1 compares this alternative, native-resolution features, and mixed AnyUp/bilinear training. Bilinear and AnyUp scores are broadly comparable, with metric-dependent differences, while AnyUp yields slightly more detailed qualitative concept maps.

C.2 Concept identification and assignment

Given the upsampled features, we first score the full concept bank and then restrict spatial competition to the selected concepts. The squared-distance competition in Equation 1 follows the prototype assignment used in PDiscoNet and PDiscoFormer (van der Klis et al., 2023; Aniraj et al., 2024), here extended to a large bank with per-image selection. It uses unnormalized prototypes and features. Each prototype $w_n \in \mathbb{R}^D$, $n \in \{1, \dots, N\}$, is initialized from a truncated normal with standard deviation 0.02. Training adds independent standard Gumbel noise to each logit and uses temperature 1, with soft outputs (Jang et al., 2017). The mean activation score is $\chi_{b,n} = (HW)^{-1} \sum_u A_{b,n,u}$. A standard Gumbel sample is $-\log(-\log U)$ for U uniform on $(0, 1)$, sampled independently for each image, concept, and location. Let B be the local batch size and $b \in \{1, \dots, B\}$ index its images. From the $B \times N$ score matrix, form

$$\widetilde{\chi}_{b,n} = \chi_{b,n} - 0.5 \log(\rho_n + 10^{-8}), \qquad Q^{(0)} = \frac{\exp(\widetilde{\chi}/0.1)}{\sum_{b,n} \exp(\widetilde{\chi}_{b,n}/0.1)}, \tag{7}$$

where ρ_n is the exponential moving average (EMA) of selection frequency, updated as specified in Appendix C.6. We use Sinkhorn–Knopp normalization (Sinkhorn & Knopp, 1967), following the use of batch-balanced prototype assignments in SwAV (Caron et al., 2020). Here balancing selects a subset of concepts rather than providing a swapped-prediction target. Ten iterations alternately divide each column by N times its sum and each row by its sum, clamping denominators to 10^{-8}. Normalization is performed in float32, under stop-gradient. The largest K entries in each final row select distinct IDs, retained in ranking order. Stable sorting favors lower bank IDs at exact ties. A per-concept score bias that is constant across images cancels under ideal Sinkhorn normalization, so the coverage term is not an independent balancing force in exact arithmetic. Top-K truncation also does not guarantee exactly balanced hard selections.

The bank-size and per-image-budget studies in Appendix E.2 vary N and K separately. Appendix E.2 compares peak scoring, direct mass-based selection, and removal of the coverage bias.

Balancing is per local batch, without gathering score matrices across devices. A training batch of one degenerates to ties. The post-selection maps $\bar{A}$ are recomputed from clean logits with a second, independent Gumbel sample. Slicing the first sampled maps would change the stochastic computation. Each location has local owner $s_u = \arg\max_k \bar{A}_{i_k,u}$ and global concept identity $p_u = i_{s_u}$, with $k \in \{1, \dots, K\}$. The K selected concepts need not all own a location: a selected concept can have positive soft mass but an empty hard region. Its attribute then contributes no spatial decoder token. These discrete choices have no derivative. An identity surrogate in Appendix C.5 supplies a gradient to the soft maps.

Batch-independent evaluation. During training, a second balancing calculation on the original view's raw scores estimates a centered dual. If Q is its balanced matrix, set

$$\delta_{b,n} = \log \max(Q_{b,n}, 10^{-30}) - \chi_{b,n}/0.1, \qquad \eta_n^{\text{new}} = \text{center}_n \left[\frac{1}{B} \sum_b \text{center}_n(\delta_{b,n}) \right]. \tag{8}$$

Here center_n subtracts the mean over the N concept indices. In distributed training, we average this estimate across devices. We then update η by exponential moving average with decay 0.99 and recenter it, initializing it with the first estimate. Evaluation uses ordinary softmax and selects the largest K values of $\chi_n(x)/0.1 + \eta_n$. It omits the coverage term. The frozen PQ stage retains the dual rather than updating it.

C.3 Attribute encoding and quantization

Once the concept regions are defined, appearance encoding summarizes how each occurrence differs from its prototype. We first compute area-scaled pooled features, as in PDiscoNet (van der Klis et al., 2023), together with their soft region masses:

$$\widetilde{z}_{i_k} = (HW)^{-1} \sum_u \bar{A}_{i_k,u} z_u, \qquad \mu_{i_k} = (HW)^{-1} \sum_u \bar{A}_{i_k,u}. \tag{9}$$

The weighted region mean in Equation 3 is stabilized as $\bar{z}_{i_k} = \widetilde{z}_{i_k}/(\mu_{i_k} + 10^{-6})$. The normalization $\nu(y) = y/\max(\|y\|_2, 10^{-12})$ is evaluated in float32 for each vector y. The residual is then cast back to the feature dtype without further normalization. Both stages pass $\mathrm{stopgrad}(r_{i_k})$ to the appearance encoder, preventing the appearance route from changing the concept reference or region features to make its own prediction easier. Assignment learning instead receives the identity-surrogate, auxiliary, and regularization signals described below. In stage 1, $q_{i_k} = W_{\mathrm{cont}}\,\mathrm{stopgrad}(r_{i_k})$, where the shared bias-free projection $W_{\mathrm{cont}} \in \mathbb{R}^{d\times D}$ is initialized from a truncated normal with standard deviation 0.02. Appendix E.5 tests restoring residual gradients, while Appendix E.3 removes the appearance channel entirely.

Product quantization. To make this appearance channel discrete, we use product quantization (PQ) (Jégou et al., 2011). We project each residual to the decoder width with $y_{i_k} = W_{\mathrm{PQ}}\,\mathrm{stopgrad}(r_{i_k}) \in \mathbb{R}^d$ and split it into G channel groups indexed by $g \in \{1,\dots,G\}$. Each selected concept uses the codebooks of its bank identity i_k.

Each group is represented by one of M codewords:

$$j_{i_k,g} = \arg\min_{j\in\{0,\dots,M-1\}} \|y_{i_k}^{(g)} - \mathcal{B}_{i_k,g,j}\|_2^2, \qquad q_{i_k} = \bigoplus_{g=1}^{G} \mathcal{B}_{i_k,g,j_{i_k,g}}, \tag{10}$$

Here $\mathcal{B}_{i_k,g,j}$ is the jth codeword in group g of concept i_k, and $\bigoplus$ concatenates the selected codewords into the quantized attribute vector q_{i_k}.

The codebooks $\mathcal{B}_{i_k,g}$ are concept-specific, so appearance codes can specialize to each concept. Product quantization represents variation through combinations of codewords rather than a separate entry for every complete attribute vector. The indices $j_{i_k,1:G}$ form the discrete appearance code. Figure 4 illustrates the attribute path and its connection to the decoder.

PQ surrogate. Hard codeword selection requires a surrogate gradient during training. Following the straight-through principle (Bengio et al., 2013), we retain hard forward values while differentiating a soft codeword mixture. We use $G = 16$ channel groups and $M = 64$ entries in each concept-specific codebook, with group width d/G. For each group in Equation 10, training uses

$$\begin{aligned} \pi_{i_k,g,j} &= \mathrm{softmax}_j\left(-\|y_{i_k}^{(g)} - \mathcal{B}_{i_k,g,j}\|_2^2\right), \\ \bar{q}_{i_k}^{(g)} &= \sum_j \pi_{i_k,g,j}\mathcal{B}_{i_k,g,j}, \qquad q_{i_k}^{(g)} = \mathcal{B}_{i_k,g,j_{i_k,g}} + \bar{q}_{i_k}^{(g)} - \mathrm{stopgrad}(\bar{q}_{i_k}^{(g)}). \end{aligned} \tag{11}$$

The softmax temperature is fixed at 1. Here $\pi_{i_k,g,j}$ is the soft probability of codeword j for concept i_k and group g, and $\bar{q}_{i_k}^{(g)}$ is their weighted mixture. The backward pass includes the direct gradient to the chosen codeword and the gradient through the soft mixture, including its dependence on the codebooks. Thus, unlike the identity-gradient estimator in VQ-VAE (van den Oord et al., 2017), our surrogate differentiates the soft assignment probabilities as well as the codeword values. Evaluation performs hard lookup only. The shared bias-free projection and private tables are learned by global cosine distillation. There is no commitment loss, residual-reconstruction loss, codebook EMA, k-means initialization, or attribute-entropy penalty. Both projection and codebooks start from a truncated normal with standard deviation 0.02. Appendix E.3 compares continuous attributes, private and shared codebooks, and vector, residual, and product quantization at different bit rates.

C.4 CONCEPT DECODER

The decoder predicts the global VFM representation (its CLS token) from concept identities, spatial assignments, and attributes, as in Equation 4. Spatial self-attention (Vaswani et al., 2017) lets the readout combine concepts according to their arrangement, while learned positions and a decoder CLS token follow the ViT readout pattern (Dosovitskiy et al., 2021). Its N learnable concept embeddings $e_n \in \mathbb{R}^d$ are independent of the matching prototypes w_n, allowing the representation used for decoding to differ from the reference used for feature matching. Assignment and decoding share the $H \times W$ grid, so identities and attributes are broadcast directly using p_u, without any spatial pooling. Concept values, the $HW \times d$ positional table, and the learned decoder CLS token are initialized from a truncated normal with standard deviation 0.02.

The decoder has three pre-normalization blocks, four attention heads, and feed-forward width $4d$ with GELU (Hendrycks & Gimpel, 2016). Attention projections are bias-free, feed-forward layers have biases. LayerNorm (Ba et al., 2016) has learned affine parameters and $\epsilon = 10^{-5}$. There is no attention/transformer dropout, mask, or stochastic depth. A final $\mathrm{LayerNorm}(d) \to \mathrm{Linear}(d, D)$ maps the output CLS token to the predicted global representation $\hat{c}(x)$. The CLS input has no separate positional embedding. The output is not explicitly unit-normalized. Each block applies pre-normalized self-attention with a residual connection, then a pre-normalized two-layer feed-forward network with a second residual connection. LayerNorm scales and offsets start at one and zero. Unless initialized explicitly above, linear weights and biases are uniform on $[-d_{\text{in}}^{-1/2}, d_{\text{in}}^{-1/2}]$, where d_{in} is that layer's input width. The concatenated attention query/key/value projection instead uses Xavier-uniform initialization (Glorot & Bengio, 2010).

Given fixed model parameters, decoding requires only the selected bank IDs $i_{1:K}$, the $H \times W$ local-owner map s, and the $K \times G$ PQ indices. Its nominal fixed-width payload is

$$K \log_2 N + HW \log_2 K + KG \log_2 M, \tag{12}$$

or 2,720 bits at $H = W = 16$ and 5,792 bits at $H = W = 32$. These counts exclude model parameters and serialization overhead. Private PQ tables contain NMd parameters (25,165,824 at $d = 384$ and 50,331,648 at $d = 768$), plus Dd projection parameters. Compact per-image codes therefore do not imply a small decoder-side dictionary. Appendix E.3 separates attribute rate from sharing of the dictionary, and Appendix E.4 compares this decoder with pooling and an auxiliary-only readout, then varies transformer depth.

C.5 TRAINING OBJECTIVES AND REGIME

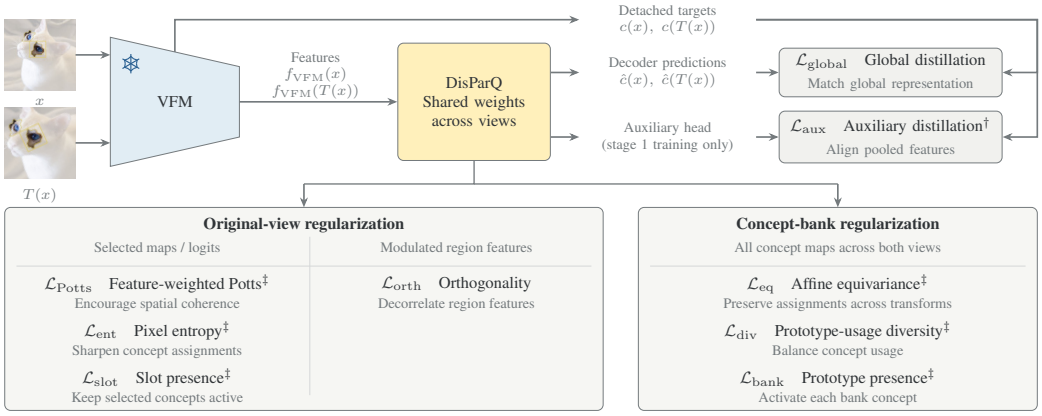


Figure 7: **Training objectives.** Two affine-related views share model parameters, and each prediction matches its own detached VFM target. Potts, entropy, slot presence, and orthogonality use the original view. Distillation and usage diversity average over views; bank presence takes maxima across both; equivariance compares aligned full-bank channels. †: auxiliary distillation is used only in continuous training. ‡: assignment regularizers train the front end in stage 1 but have no gradient to it during frozen PQ training. Orthogonality can still update region modulation in stage 2.

Table 6: Loss weights and view accounting. Frozen PQ still computes assignment regularizers, but they cannot train the frozen prototype-assignment path. Orthogonality can update the unfrozen modulation parameters.

Term	Stage 1	Stage 2	Views
$\mathcal{L}_{\text{global}}$	1	1	Mean of both
$\mathcal{L}_{\text{aux}}$	1	0	Mean of both
$\mathcal{L}_{\text{Potts}}$	1	1	Original
$\mathcal{L}_{\text{ent}}$	1	0.5	Original
$\mathcal{L}_{\text{orth}}$	1	1	Original
$\mathcal{L}_{\text{slot}}$	1	1	Original
$\mathcal{L}_{\text{eq}}$	1	1	Paired
$\mathcal{L}_{\text{div}}$	0.1	0.1	Mean of both
$\mathcal{L}_{\text{bank}}$	0.1	0.1	Maximum over both

Auxiliary region readout. The auxiliary head supervises pooled region content directly, complementing the decoder that sees concept identities and attributes. We adopt PDiscoFormer's normalization-based modulation (Aniraj et al., 2024), which develops the part modulation of PDiscoNet (van der Klis et al., 2023). A learned $\text{LayerNorm}([D, K])$ acts on the area-scaled $\widetilde{Z} = [\widetilde{z}_{i_1}, \ldots, \widetilde{z}_{i_K}] \in \mathbb{R}^{D\times K}$ to form modulated features V_m. It normalizes jointly over feature and slot axes. These features support orthogonality and the stage-1 auxiliary head f_{aux}. Following PDiscoNet's part dropout (van der Klis et al., 2023), we discourage reliance on a single informative region: the auxiliary branch zeros whole slots with slot dropout at probability 0.3 (scaling retained slots by $1/0.7$), averages over all K slot positions, and applies $\text{LayerNorm}(D) \to \text{Linear}(D, D) \to \text{GELU} \to \text{Linear}(D, D)$. Modulation and dropout do not enter the primary residual branch.

Identity surrogate in continuous training. To let global distillation shape assignments despite hard ownership, we use a straight-through construction (Bengio et al., 2013; Jang et al., 2017) based on the selected soft maps $\bar{A}$. During stage 1, we replace the identity contribution in Equation 4 by

$$t_u^{\text{id}} = e_{p_u} + \bar{e}_u - \text{stopgrad}(\bar{e}_u), \qquad \bar{e}_u = \sum_k \bar{A}_{i_k,u} e_{i_k}. \tag{13}$$

This has hard forward values but supplies gradients to soft spatial assignments and an additional gradient to concept values. It differentiates neither top-K selection nor hard attribute broadcasting. It connects global distillation to prototypes through $\bar{A}$ and to fusion through AnyUp's feature inputs, while the appearance residual remains detached. It is disabled in stage 2 and evaluation.

Removing the surrogate or enabling residual gradients is studied in Appendix E.5. The auxiliary-only alternative and the removal of auxiliary distillation are distinguished in Appendices E.4 and E.6.

Combined objective. Global and auxiliary distillation retain target information, while spatial and usage regularizers shape its organization into concepts. Table 6 specifies the weights of the nine terms shown in Figure 7. For each view $v \in \{x, T(x)\}$, the auxiliary prediction is $\hat{c}_{\text{aux}}(v) = f_{\text{aux}}(V_m(v))$ with the pooling and dropout above, and

$$\mathcal{L}_{\text{aux}} = \frac{1}{2} \sum_{v\in\{x,T(x)\}} [1 - \cos(\hat{c}_{\text{aux}}(v), \text{stopgrad}(c(v)))]. \tag{14}$$

The total objective is the sum of the nine losses multiplied by their table weights, averaged over batch images wherever an image index remains. Both cosine losses and usage diversity average their two per-view values. Potts, entropy, orthogonality, and slot presence use only the original view. Bank presence takes maxima over both views, and equivariance compares them. Cosine similarity uses $\epsilon = 10^{-8}$. The auxiliary head matches the same view-specific backbone target as the main decoder.

Feature-weighted Potts. We favor agreement between neighboring assignments when their features are similar, while weakening smoothing across feature discontinuities. This is a local soft Potts penalty with a Gaussian feature kernel, related to the appearance-dependent pairwise potentials of Krähenbühl & Koltun (2011). We use it in place of PDiscoFormer's unweighted total-variation prior (Aniraj et al., 2024). Our adaptive temperature keeps confident logits from saturating the smoothing gradient, and the bandwidth adapts to feature-distance scale. Specifically,

soften the clean selected logits without Gumbel noise using $C_{i_k,u} = \mathrm{softmax}_k(\ell_{i_k,u}/\tau_{\mathrm{P}})$, where $\tau_{\mathrm{P}} = \max(1, \mathrm{median}_{b,u}(\ell^{(1)}_{b,u} - \ell^{(2)}_{b,u})/2)$ and superscripts denote the two largest selected logits. Here C denotes the spatial probabilities used by the Potts loss. Image indices are suppressed in the spatial expressions below. For each horizontal/vertical neighbor pair (u, u'), counted once, let $\zeta_{uu'} = \|z_u - z_{u'}\|_2^2$ denote its squared feature distance and set $\sigma^2 = \max(10^{-8}, \mathrm{quantile}_{0.9}\{\zeta_{uu'}\})$. Then

$$\mathcal{L}_{\mathrm{Potts}} = \mathrm{mean}_{b,(u,u')} \exp[-\zeta_{uu'}/(2\sigma^2)] \left(1 - \sum_k C_{i_k,u} C_{i_k,u'}\right). \tag{15}$$

Features used for the kernel, temperature, and bandwidth are detached. Quantiles and medians are over the local batch. There is no normalization by total edge weight.

Entropy, orthogonality, and presence. The entropy term follows PDiscoFormer (Aniraj et al., 2024) and encourages confident spatial ownership. Orthogonality discourages redundant region features, following the part-decorrelation prior in PDiscoNet (van der Klis et al., 2023) and earlier SCOPS (Hung et al., 2019). We use squared Gram-matrix deviations, penalizing both positive and negative correlations, rather than the signed cosine sum written in PDiscoFormer. Selected-slot presence adapts PDiscoNet's batch-presence term: spatial averaging prevents an isolated activation from satisfying it. We additionally apply this criterion to full-bank identities to supply gradients to concepts outside the selected subset.

For batch-indexed maps, $i_{b,k}$ is the bank ID at selected-list position k in image b, and $k, l \in \{1, \ldots, K\}$ index selected-list positions. With $\widehat{V}_m$ denoting channel-normalized columns of V_m,

$$\mathcal{L}_{\mathrm{ent}} = -\,\mathrm{mean}_{b,u} \sum_k \bar{A}_{b,i_{b,k},u} \log(\bar{A}_{b,i_{b,k},u} + 10^{-8}), \tag{16}$$

$$\mathcal{L}_{\mathrm{orth}} = \mathrm{mean}_{b,k,l} \left[(\widehat{V}_{m,b}^{\top} \widehat{V}_{m,b})_{kl} - \mathbf{1}_{k=l}\right]^2, \tag{17}$$

$$\mathcal{L}_{\mathrm{slot}} = 1 - \frac{1}{K} \sum_k \max_{b,u} \mathrm{AvgPool}_{3\times3}(\bar{A}_{b,i_{b,k}})_u, \tag{18}$$

$$\mathcal{L}_{\mathrm{bank}} = 1 - \frac{1}{N} \sum_n \max_{b,u,v \in \{x, T(x)\}} \mathrm{AvgPool}_{3\times3}(A^v_{b,n})_u. \tag{19}$$

Pooling uses stride 1 without padding. Orthogonality averages over all BK^2 entries. Slot presence is indexed by local list position. Bank presence is indexed by globally shared identity and can train unselected prototypes. Presence maxima are not gathered across devices.

Prototype-usage diversity. Subset balancing does not directly penalize concentrated full-bank spatial responses. We therefore add a differentiable usage penalty toward a detached, Sinkhorn-balanced target, using the same balancing principle as above (Sinkhorn & Knopp, 1967; Caron et al., 2020). For each view, let $\psi_n = B^{-1} \sum_b \chi_{b,n}$ be the batch-mean full-bank activation, normalized to sum to one. To form a detached target ψ^{bal}, apply the same ten-iteration balancing routine to $\exp(\mathrm{stopgrad}(\chi))^{\top} \in \mathbb{R}^{N \times B}$ (with the column factor now B), transpose back, average over images, and normalize. The loss is

$$\mathcal{L}_{\mathrm{div}} = \sum_n \psi_n \left[\log(\psi_n + 10^{-8}) - \log(\psi^{\mathrm{bal}}_n + 10^{-8})\right]. \tag{20}$$

The exponent here has temperature 1, distinct from the selection temperature 0.1. The final row normalization of the $N \times B$ matrix makes each prototype row sum to one. Thus, when denominator clamps are inactive, averaging over images and normalizing gives $\psi^{\mathrm{bal}}_n = 1/N$ in exact arithmetic. The loss is therefore an epsilon-stabilized KL divergence toward uniform usage. Without the numerical stabilizers, it equals $\log N - \mathcal{H}(\psi)$, where $\mathcal{H}(\psi) = -\sum_n \psi_n \log \psi_n$. This term regularizes prototype usage rather than quantizer commitment.

Affine equivariance. To encourage concept identities to follow image content under changes of pose, we adapt the inverse-warp cosine objective of PDiscoNet and PDiscoFormer (van der Klis et al., 2023; Aniraj et al., 2024). Equivariant part maps were also encouraged in SCOPS (Hung et al., 2019). Our comparison uses active full-bank identities so that a change in the selected subset does not change which concepts are compared. For each image b, let $\mathcal{I}^{32}_b$ contain the 32 bank indices with the largest $\max(\chi_{b,n}(x), \chi_{b,n}(T(x)))$. We compare their full-bank maps after inverse warping:

$$\mathcal{L}_{\text{eq}} = 1 - \text{mean}_{b,n \in \mathcal{I}_b^{32}} \cos\big(A_{b,n}(x), \mathcal{W}_T^{-1} A_{b,n}(T(x))\big)\,. \tag{21}$$

Cosine similarity acts on flattened maps. The resampling operator $\mathcal{W}_T^{-1}$ first undoes translation, then rotation and scale, using two bilinear resampling operations. Map-space translations are rounded after scaling by the ratio of map to image resolution. No overlap mask removes filled borders. Using globally indexed maps avoids matching unrelated slot positions when the two views select different concepts.

The affine transforms T are sampled as described in Appendix D.3.

C.6 Training stages and optimization

Two-stage curriculum. The continuous first stage lets concepts form before imposing a discrete appearance bottleneck. Freezing assignment in the second stage then gives quantization a stable concept reference. Stage 1 jointly learns fusion, matching prototypes, continuous appearance projection, concept embeddings, decoder, region modulation, and the auxiliary head. It uses detached residuals and the identity surrogate. Stage 2 starts from the stage-1 checkpoint with the lowest validation global cosine distance among checkpoints saved every ten epochs. It retains fusion, prototypes, decoder embeddings, positional and CLS embeddings, transformer and output head, and region modulation. Fusion and prototypes are frozen. Usage and dual estimates are retained without updates, prototype resampling is disabled, and the auxiliary head is removed. The continuous projection is discarded. A new PQ projection and private codebooks are initialized as specified in Appendix C.3. These, the concept embeddings, and decoder learn through global distillation. Region modulation remains trainable through orthogonality but does not enter the decoder's appearance path. A fresh optimizer and learning-rate schedule are used for stage 2.

Freezing assignment parameters does not make the training forward deterministic: both stages use Gumbel noise and batch-balanced selection. Evaluation removes the noise, uses the stored dual for independent selection, and performs hard PQ lookup. Table 7 summarizes the gradient and state changes.

Table 7: Canonical stage transition. The backbone and AnyUp weights are frozen in both stages, residuals are detached in both.

Component or operation	Stage 1	Stage 2
Fusion and matching prototypes	Learned	Retained, frozen
Appearance encoder	Continuous projection	New private PQ
Concept embeddings and decoder	Learned	Retained, learned
Auxiliary distillation	Enabled	Disabled
Identity gradient surrogate	Enabled	Disabled
AnyUp feature-input gradients	Enabled	Disabled
Usage and selection dual	Updated	Retained, fixed
Dead-prototype resampling	Enabled	Disabled

Usage and prototype resampling. Balancing alone cannot restore a prototype that receives little useful learning signal. We therefore replace persistently underused prototypes with current region features. Stage 1 initializes $\rho_n = 1/N$. After computing both views, selected-ID counts are summed across views and devices and divided by the total number of selected IDs, forming a frequency vector. Usage is updated as 0.99 times its old value plus 0.01 times this frequency vector. After each optimizer step, prototypes with $\rho_n < 10^{-4}$ are replaced by normalized area-scaled region features drawn uniformly from the current batch's pooled features across both views (without replacement when the pool is large enough), scaled to the mean active-prototype norm, plus Gaussian noise of standard deviation 10^{-3}. If all prototypes are inactive, use the whole-bank mean norm. Reset the replaced prototype's usage to $1/N$, its dual to zero, and its value embedding from the replacement prototype. Resetting individual dual entries can temporarily change the vector's mean. It is recentered at the next dual update. A common additive shift does not affect selection rankings. Optimizer moments are not reset. Under distributed training, one device samples from its own feature pool and broadcasts the changes. Appendix E.2 measures the effect of disabling this replacement.

The augmentation, optimizer, schedules and checkpoint selection of DisParQ are given in Appendix D.3, together with the schedules of every dataset.

Order of operations. For each training batch, form the two views and compute their frozen targets and fused, upsampled features. Independently for each view, compute full-bank maps, balance mean scores, select concepts, recompute selected maps, pool residuals, and decode using continuous or quantized attributes. Sum the weighted losses with the view accounting in Table 6, backpropagate, clip gradients, and take one optimizer step. In stage 1, update usage from both views and estimate the selection dual from the original view during the forward computation. After the optimizer step, resample inactive prototypes. In stage 2 these assignment states remain fixed. At evaluation, a single view preprocessed according to Appendix D.5 supplies ordinary soft maps, dual-based selection, hard ownership, and hard attribute codes. Because balancing and several regularizers use local batches, distributed execution is not equivalent to a single solve on the concatenated batch.

The objective-removal studies in Appendix E.6 distinguish preservation of downstream information from spatial and semantic organization.

D Experimental protocol

This section specifies the datasets, backbones, training procedures and evaluation metrics used in Section 4 and Appendices E–G, including method-specific exceptions.

D.1 Datasets and splits

Table 8 lists ImageNet-1k (Deng et al., 2009), Places365-Standard (Zhou et al., 2018), PartImageNet (He et al., 2022), CUB-200-2011 (Wah et al., 2011), Stanford Cars (Krause et al., 2013), Stanford Dogs (Khosla et al., 2011) and Oxford Flowers (Nilsback & Zisserman, 2008). We train one DisParQ model per dataset without class labels or part annotations. We use class labels to fit classification probes and label the kNN reference set. Keypoint annotations train the keypoint regressors, and part annotations support concept evaluation. Supervised baseline training and selection among evaluated configurations are described in Section D.4.

We report results on the validation splits of ImageNet-1k, Places365 and PartImageNet and on the test splits of the other datasets. Appendix G.7 additionally uses the PartImageNet test split. On Oxford Flowers, we merge the training and validation splits, following PDiscoFormer (Aniraj et al., 2024).

Table 8: **Datasets.** Split sizes, available part annotations and corresponding results tables. Oxford Flowers uses the combined training and validation splits for training.

Dataset	Classes	Training	Evaluation	Part annotation	Tables
ImageNet-1k	1000	1,281,167	val 50,000	none	15
Places365	365	1,803,460	val 36,500	none	15
PartImageNet	158	20,466	val 1,206	40 part masks	2, 4, 19
CUB-200-2011	200	5,994	test 5,794	15 keypoints	3, 16, 19
Stanford Cars	196	8,144	test 8,041	none	3
Stanford Dogs	120	12,000	test 8,580	none	3
Oxford Flowers	102	2,040	test 6,149	none	3

D.2 Backbones

Table 9 lists DINOv2 (Oquab et al., 2024) with registers (Darcet et al., 2024), DINOv3 (Siméoni et al., 2025), TIPSv2 (Cao et al., 2026), CLIP (Radford et al., 2021), SigLIP2 (Tschannen et al., 2025) and CLIP-DINOiser (Wysoczańska et al., 2024). In DisParQ, these feature extractors are frozen and receive 224×224 images. The input wrapper converts ImageNet-normalized images to each encoder's pretrained normalization, while AnyUp retains ImageNet-normalized guidance. We fuse patch tokens from several layers as described in Appendix C.1, except for CLIP-DINOiser, which directly supplies its refined projected features. The global distillation targets are listed in the table.

AnyUp (Wimmer et al., 2026) increases the native 16×16 grid of patch-14 models to 32×32 and the 14×14 grid of patch-16 models to 28×28. Baseline input resolutions, trainable tokens and scoring grids have the exceptions described in Sections D.4 and D.5.

The main controlled comparison uses DINOv2-B/14, with CFM evaluated on its released CLIP-DINOiser dictionary. Figure 8 covers seven configurations of six feature extractors because SigLIP2 has two targets. We use CLIP ViT-L/14 for the MSAE reference on its original backbone (Figure 23 and Table 16).

CLIP-DINOiser combines a frozen CLIP encoder with the authors' released 1.8M-parameter adapter[3]. Two convolutions predict patch affinities that pool dense CLIP features into the denoised 512-dimensional map used by DisParQ. Its encoder is the listed OpenCLIP checkpoint (Cherti et al., 2023), trained on LAION-2B (Schuhmann et al., 2022). The separate CLIP ViT-B/16 row uses OpenAI weights.

Table 9: **Feature extractors and targets.** Checkpoints are Hugging Face Hub identifiers. The grid is the assignment grid of DisParQ, or the scoring grid of the CLIP-L/14 MSAE reference. Targets are the frozen global features distilled by DisParQ at 224×224. A dash indicates a baseline-only encoder.

Backbone	Checkpoint	Grid	Target
DINOv2-B/14, registers	`facebook/dinov2-with-registers-base`	32×32	CLS token
DINOv3-B/16	`timm/vit_base_patch16_dinov3.lvd1689m`	28×28	CLS token
TIPSv2-B/14	`google/tipsv2-b14`	32×32	CLS token
CLIP ViT-B/16	`openai/clip-vit-base-patch16`	28×28	CLS token after the last layer norm
SigLIP2-B/16	`google/siglip2-base-patch16-224`	28×28	attention-pooler output, or mean patch token
CLIP-DINOiser	`laion/CLIP-ViT-B-16-laion2B-s34B-b88K`	28×28	projected CLIP image embedding
CLIP ViT-L/14	`openai/clip-vit-large-patch14`	32×32	—

D.3 Training DisParQ

The main DINOv2 experiments use the architecture, losses and two-stage curriculum of Appendix C, with $N = 1024$ concepts, $K = 16$ selected concepts per image and concept-specific PQ with $G = 16$ groups of $M = 64$ codewords. Schedules and batch sizes vary across datasets, and Places365 uses mixed upsampling in stage 2.

Augmentation. Each training image produces two views. The first combines crop and photometric augmentations commonly used in self-supervised vision (Chen et al., 2020; Caron et al., 2021) to diversify appearance. The second applies a random affine transform to the normalized first view, following the affine view pairing used by PDiscoNet and PDiscoFormer (van der Klis et al., 2023; Aniraj et al., 2024). We draw one transform per local batch, providing known spatial correspondences for $\mathcal{L}_{\text{eq}}$. Table 10 specifies our recipe.

Table 10: **Augmentations of DisParQ.** The first-view transformations are applied in the listed order. The second view is an affine transform of the normalized first view.

Transformation	Probability	Parameters
First view		
Random resized crop	1	scale $[0.64, 1]$, ratio $[3/4, 4/3]$, 224×224, bicubic
Horizontal flip	0.5	
Color jitter	0.8	brightness and contrast 0.4, saturation 0.2, hue 0.1
Grayscale	0.2	
Gaussian blur	0.5	kernel 9, $\sigma \in [0.1, 2]$
Normalization	1	mean $(0.485, 0.456, 0.406)$, std $(0.229, 0.224, 0.225)$
Second view		
Affine transform	1	rotation $[-90°, 90°]$, translation $\pm 10\%$, scale $[0.7, 1.3]$, no shear, bilinear interpolation, zero fill in normalized units

Optimization. Tables 11 and 12 specify optimization and dataset-specific schedules. Both views are balanced separately within each local batch and jointly contribute to one optimizer update. Training

[3] https://github.com/wysoczanska/clip_dinoiser

loaders shuffle and drop incomplete batches. Learning rates are not automatically scaled with batch size.

Table 11: **Optimization and checkpoint selection of DisParQ.** Schedules and per-device batch sizes are in Table 12.

Setting	Value
Optimizer	AdamW (Loshchilov & Hutter, 2019), $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$
Weight decay	10^{-4}, on all trainable parameters
Learning-rate schedule	linear warm-up per epoch from $10^{-4}\times$ the peak, then cosine decay (Loshchilov & Hutter, 2017) to 10^{-6}
Devices and accumulation	one GPU, or four H100s for ImageNet-1k and Places365. No gradient accumulation
Precision and clipping	bfloat16 mixed precision and gradient-norm clipping at 1
Seed	42 for main experiments. Ablation replication is specified in Appendix E
Checkpoint selection	lowest validation $\mathcal{L}_{\text{global}}$ among checkpoints saved every 10 epochs (stage 1) or every epoch (stage 2). Stage 2 starts from the selected stage-1 checkpoint

Mixed upsampling on Places365. To reduce computation, Places365 uses AnyUp in a random 20% of stage-2 batches and bilinear interpolation otherwise. The other main training runs and downstream evaluation use AnyUp throughout. Appendix E.1 compares these alternatives.

D.4 Training the baselines

We train the dataset-specific baselines on the training data in Section D.1, using the authors' implementations where available. The main comparison uses DINOv2-B/14. Exceptions include the released CFM dictionary, released PDiscoFormer checkpoints and the MSAE reference on CLIP-L/14. Table 13 summarizes these models and their probe inputs. The vision-language and supervised reference entries in Figure 5 (left) and Table 15 reproduce results reported by CFM (Wittenmayer et al., 2026) and ProtoQuant (Janusz et al., 2026).

Sparse autoencoders. TopK SAE (Gao et al., 2025) is our re-implementation. PatchSAE (Lim et al., 2025)[4] and MSAE (Zaigrajew et al., 2025)[5] use the authors' code and hyperparameters (Table 14). These models are fitted to frozen DINOv2-B/14 patch tokens for the main comparison. Following the MSAE authors, we remove its TopK constraint at inference.

On PartImageNet, we sweep dictionary sizes and, for MSAE, both loss weightings and the three published expansion factors. The main table presents the best evaluated configuration of each SAE by mean rank over the reported columns of Table 2. For the fine-grained table, we select one MSAE configuration jointly across all four datasets, requiring an available LP result on each dataset and averaging ranks over the ten reported metrics. In both selections, metrics are ranked in their preferred direction on unrounded values, missing metrics rank last, and exact ties follow the fixed candidate order. Candidates without a foreground-purity result are excluded on PartImageNet. A tie in the aggregate rank is also resolved by candidate order. These are best-of-sweep references selected using the reported validation or test scores, including annotated-part metrics where available. Representation fitting remains label-free, but this configuration selection uses evaluation labels. The selected MSAE sizes are 8192 on PartImageNet and 6144 on the fine-grained datasets with reverse and uniform weighting respectively. Figures 22 and 23 show the evaluated alternatives.

CFM. CFM (Wittenmayer et al., 2026) is a Matryoshka sparse autoencoder trained by its authors on CC12M features of CLIP-DINOiser. We use its released dictionary[6] without refitting it to our datasets.

PIP-Net and ProtoQuant. Both use the authors' code[7] with the training settings in Table 14, adapted to frozen DINOv2-B/14 features. Stage 1 is label-free and appears with the self-supervised methods. Stage 2 adds supervised classification and is an unranked reference. ProtoQuant's original recipe first fine-tunes the backbone on the target dataset. Our adaptation keeps that backbone frozen to match the feature extractor used by the main self-supervised comparison.

[4] https://github.com/dynamical-inference/patchsae
[5] https://github.com/WolodjaZ/MSAE
[6] https://github.com/kawi19/CFM
[7] https://github.com/M-Nauta/PIPNet, https://github.com/gmum/protoquant.

Table 12: **Training schedules of DisParQ with DINOv2-B/14 and width 768.** The PartImageNet rows give the canonical schedule. Batch size is per device.

Stage	Epochs	Warm-up	Peak LR	Batch
PartImageNet				
Continuous concept discovery	150	10	$3 \cdot 10^{-4}$	16
Concept-specific PQ, 16×64	200	10	10^{-3}	16
CUB-200-2011, Stanford Cars, Stanford Dogs, Oxford Flowers				
Continuous concept discovery	200	40	$3 \cdot 10^{-4}$	16
Concept-specific PQ, 16×64	500	50	10^{-3}	16
ImageNet-1k, Places365				
Continuous concept discovery	50	10	$3 \cdot 10^{-4}$	64
Concept-specific PQ, 16×64	100	10	10^{-3}	64

Table 13: **Baselines in the main comparison.** Size denotes the global dictionary or prototype count, with PCA specified per image or class. Grid denotes the scoring resolution. The notation $16 \to 32$ indicates bilinear resampling from the native 16×16 maps to 32×32. Probe inputs are used for LP and kNN.

Method	Code	Class labels	Size	Grid	Probe input
TopK SAE	ours	no	2048 latents	$16 \to 32$	mean-pooled reconstruction
PatchSAE	authors'	no	49,152 latents	$16 \to 32$	mean-pooled reconstruction
MSAE	authors'	no	8192 latents (fine-grained: 6144)	$16 \to 32$	mean-pooled reconstruction
CFM	released	no	8192 latents	28×28	max- and mean-pooled activations
PIP-Net, stage 1 / 2	authors'	stage 2	768 prototypes	32×32 (AnyUp)	pooled prototype scores
ProtoQuant, stage 1 / 2	authors'	stage 2	2048 codes (CUB: 4000)	$16 \to 32$	pooled prototype scores
PDiscoFormer	released or retrained	yes	K parts + background	16×16 (CUB: $37 \to 32$)	mean part embedding
PCA, per image / class-wise	ours	class-wise only	16 per image or class	$16 \to 32$	none

Table 14: **Baseline training settings.** The table lists the recipes used for our retraining. MSAE runs for at least 21,000 steps, extending beyond 30 epochs on smaller datasets. CFM uses released weights and PCA is fitted directly to patch features.

Method	Optimizer	Learning rate	Batch	Length	Other settings
TopK SAE	Adam	$3 \cdot 10^{-4}$	64 images	20 epochs	8 latents per patch, 16 per image, auxiliary loss, weight $1/32$
PatchSAE	Adam	$4 \cdot 10^{-4}$, 500 warm-up steps	128 images	20,480 steps	L1 weight $8 \cdot 10^{-5}$, ghost gradients
MSAE	AdamW	10^{-4}	4096 tokens	30 epochs, $\geq 21{,}000$ steps	nested levels $64, 128, \ldots$ up to the dictionary size, uniform or reverse weighting
PIP-Net	AdamW	$5 \cdot 10^{-4}$, classifier 0.05	128 / 64	10 + 60 epochs	stage 1 / stage 2, features upsampled by AnyUp
ProtoQuant	AdamW	0.05	512	30 + 30 epochs	stage 1 / stage 2, temperature 0.1
PDiscoFormer, retrained	Adam	$1.414 \cdot 10^{-6}$ base, halved every 4 epochs	32 (PartImageNet: 4×32, accumulated)	28 epochs	position embedding, CLS and register tokens trained

PDiscoFormer. Tables 2 and 3 evaluate the authors' released checkpoints[8] for PartImageNet, CUB and Oxford Flowers. We additionally retrain the evaluated part counts on PartImageNet, including a frozen-token variant at $K = 25$ (marked † in Figure 21), and train $K = 16$ models on Stanford Cars and Stanford Dogs, which have no released checkpoints. The retraining recipe updates the positional embeddings, CLS and register tokens while keeping the encoder blocks frozen, except in the explicitly frozen-token variant. Relative to the base rate in Table 14, learning rates are multiplied by 10^3 for these tokens and by 10^4 for prototypes, modulation and classifier parameters. Following

[8] https://github.com/ananthu-aniraj/pdiscoformer

the retraining procedure, we select the snapshot with the highest test accuracy. These supervised references therefore use the evaluation labels for checkpoint selection.

PDiscoFormer's PartImageNet recipe combines the training and validation splits, so the validation images in Table 2 were also seen with their labels during representation learning. Its released CUB checkpoints retain their 518×518 input resolution. Appendix F.4 gives the retraining results, and Sections D.5 and D.6 describe how these models are scored.

PCA. We include two PCA references using 16 components per image, fitted to native 16×16 DINOv2-B/14 patch features. Each patch is assigned to the component with the largest absolute projection. Per-image PCA fits a new basis for each image, so its component identities have no shared meaning across images. Class-wise PCA fits one basis per training class and selects the basis using the evaluation image's ground-truth class. Each class has distinct global component IDs, giving 16 times the number of classes in the full bank. This ground-truth-conditioned representation makes a classification probe circular, so we report no downstream probe for it.

On PartImageNet, PCA also estimates foreground from the sign of the first component. We orient its sign so that the mean projection of the four corner patches is nonpositive and retain patches with positive projection. On CUB, all patches are retained. This heuristic changes PCA's spatial support and is part of the baseline definition.

D.5 Evaluation protocol

For locally evaluated models, we use the metric implementations in Section D.6 on the stated evaluation splits. The default budget is $K = 16$ concepts per image, with exceptions indicated by the K/N column. Published reference scores retain their original protocols. We disable the test-time Gumbel noise in PDiscoFormer's released implementation to make its concept maps deterministic.

Inputs. Our classification and image-to-image retrieval evaluations use a short-side resize to 256 followed by a 224×224 center crop. PIP-Net and ProtoQuant use the same crop-based probing pipeline. The dictionary baselines and PDiscoFormer are probed on cached features from square-resized images, preserving the protocol used to obtain their reported results. Thus using the same probe architecture does not imply identical input geometry or augmentation.

All reported mask and keypoint metrics use paired square resizing of images and annotations. The common input size is 224×224, with nearest-neighbor interpolation for masks and corresponding coordinate scaling for keypoints. This retains the full annotated object and aligns spatial evaluation across methods. It follows PIP-Net's full-image 224×224 protocol and 32×32 purity windows (Nauta et al., 2023). For the PDiscoFormer part-discovery and CFM grounding metrics (Aniraj et al., 2024; Wittenmayer et al., 2026), we adopt their metric definitions in this common geometry. In particular, our square-resized PartImageNet evaluation differs from PDiscoFormer's released center-crop evaluation.

Released PDiscoFormer CUB checkpoints retain the authors' 518×518 input size, preserving their learned spatial configuration. We scale the purity window to 74×74 to keep its relative size close to $32/224$. Keypoint regression uses image-normalized coordinates for every method, as defined in Section D.6.

Scoring grids. Table 13 lists the baseline grids. DisParQ and PIP-Net assign concepts on AnyUp's 32×32 DINOv2-B/14 feature grid. The sparse autoencoders, ProtoQuant and PCA produce native 16×16 maps, which we resample bilinearly to 32×32 for scoring. CFM retains a 28×28 grid, and PDiscoFormer retains 16×16 on PartImageNet. Its 37×37 CUB maps are resampled to 32×32.

Clustering metrics and part retrieval reduce annotations to these grids. Segmentation metrics instead upsample the concept maps to the image grid, and keypoint regression retains continuous target coordinates. The main tables therefore include resolution differences for CFM and PDiscoFormer, which can affect boundary-sensitive scores. Retrieval rows with a different region set are marked ‡ and accompanied by their own oracle.

Concept budget. TopK SAE is trained with at most 16 concepts per image, and its reconstruction retains this budget for downstream probing. PatchSAE, MSAE and CFM retain each image's 16 strongest concepts for spatial metrics and part retrieval. Their LP, kNN and image-to-image retrieval

instead use the unbudgeted representations listed in Table 13. Figure 22 includes other budgets and the unrestricted CFM dictionary used in its published evaluation.

Probing representations. Our probes use the predicted global representation $\hat{c}(x)$, which preserves the target definition for each backbone in Table 9. Baseline representations are listed in Table 13. We evaluate the representation passed downstream by each method.

1. **Linear probe (LP).** We apply batch normalization without affine parameters, followed by a linear classifier trained on the training split. Optimization uses SGD with momentum 0.9, no weight decay and learning rate $0.1 \cdot B_{\text{global}}/256$, where B_{global} is the global batch size. Training lasts 50 epochs, with 10 epochs of warm-up followed by cosine decay. For DisParQ, PIP-Net and ProtoQuant, training images are re-encoded under random resized crops and horizontal flips. Dictionary baselines and PDiscoFormer use precomputed vectors without probe-time augmentation. Cached dictionary probes use batches of 256. These choices preserve the recorded evaluation pipelines, so LP comparisons include differences in augmentation, image geometry and optimizer step counts.
2. **kNN.** We predict the class with the largest weighted vote among the 20 nearest training images under cosine similarity. A neighbor with similarity s contributes weight $\exp(s/0.07)$.

Fine-grained datasets. Table 3 reports LP and kNN accuracy on all four datasets, together with PIP purity and keypoint regression error on CUB. CUB provides annotations for 15 keypoints. Stanford Cars, Stanford Dogs and Oxford Flowers have no part annotations.

D.6 Metrics

LP and kNN. We report top-1 classification accuracy on the evaluation split using the probes in Section D.5.

Image retrieval. Images are represented by the same frozen vectors used for probing. Each evaluation image queries the other evaluation images by cosine similarity, and a retrieved image is relevant when it shares the query's class. Recall at rank r, denoted R@r, is the fraction of queries with at least one relevant result among the first r candidates. For each query, average precision (AP) averages the precision at ranks containing relevant results over the full ranking. Mean average precision (mAP) averages AP over queries. For image retrieval, queries with no other image of the same class are excluded from both recall and mAP.

Foreground purity. We reduce each mask to the scoring grid by a majority vote per cell. Let C_{nj} count cells assigned to global concept n with ground-truth label j, including the background label where it is annotated. Let $\mathcal{J}_{\text{fg}}$ be the foreground part labels and N_{fg} the number of annotated foreground cells in the evaluated split. Then

$$\text{Pur}_{\text{fg}} = \frac{1}{N_{\text{fg}}} \sum_{n} \max_{j \in \mathcal{J}_{\text{fg}}} C_{nj}.$$

A foreground cell left unassigned contributes to N_{fg} but to no concept count, so rejecting foreground cannot improve purity by reducing its denominator. On CUB, counts use the grid cells containing visible keypoints.

NMI, ARI and their mapped variants. We compare the ground-truth part labels y with concept identities c over the annotated foreground cells of the evaluated split, following part-discovery evaluation (Choudhury et al., 2021; van der Klis et al., 2023; Aniraj et al., 2024). Unassigned foreground cells receive a separate cluster label. NMI normalizes mutual information by the arithmetic mean of the two entropies,

$$\text{NMI} = \frac{2I(y;c)}{H(y) + H(c)}.$$

The Rand index (RI) measures the fraction of cell pairs that both labelings place together or both place apart. ARI corrects this agreement for chance,

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{1 - \mathbb{E}[\text{RI}]},$$

where the expectation assumes random assignments with fixed cluster sizes, giving chance an expected score of zero.

Raw NMI and ARI penalize splitting one part across several pure concepts. For NMI_m and ARI_m, we first map each concept to its dominant label on the evaluated split, $\arg\max_j C_{nj}$, including background where annotated. We then compute NMI and ARI on the mapped labels, retaining a separate cluster for unassigned cells. On PartImageNet, this mapping uses at most 41 semantic labels, comprising 40 parts and background, in addition to the unassigned cluster. Mapping removes the penalty for splitting a part into concepts that receive the same label. The mapping is fitted on the evaluated split, so these are descriptive clustering scores.

mIoU and FG-ARI. Our mIoU (He et al., 2022) is a per-image matched-part score. We upsample the K concept maps to input resolution, assign each pixel to its strongest concept, and use Hungarian one-to-one matching to annotated parts. IoU is averaged over matched pairs, then images, excluding unmatched parts. This differs from dataset-level semantic mIoU. FG-ARI (Locatello et al., 2020) averages per-image ARI over annotated foreground pixels, rather than the split-wide grid cells used above.

Locality, consistency and impurity. We follow CFM's metric implementation (Wittenmayer et al., 2026). We sort nonnegative map activations and find the cutoff where cumulative mass reaches 90%. All pixels at or above this cutoff are retained, including ties. Thus a uniform hard region retains its entire support. Part masks are dilated by a 3×3 maximum filter.

Locality takes the best IoU between each part and any concept support, averages over images containing that part, and then over the 40 PartImageNet part labels. Consistency first averages each concept–part IoU over images in which either the concept or the part is present. It then selects the best concept for each part and averages over parts. Impurity measures natural-log entropy of the retained activation mass over the dilated label masks, including background, after normalizing those label masses. We average entropy over images within each concept, then over concepts active in at least five images.

As an approximate implementation check under CFM's original center-crop/14×14 protocol, its unrestricted released dictionary gives locality 44.3, consistency 15.3 and impurity 0.333 (published: 44.3, 15.3, 0.330). This differs from our common square-resize/28×28 evaluation.

PIP purity. We adapt PIP-Net's top-image purity evaluation (Nauta et al., 2023) to each method's concept maps on the CUB test split, merging left and right versions of each part.

1. For each concept, retain up to 10 images with its highest peak activations.
2. In each image, center a 32×32-pixel window on the map's peak. If several cells share the maximum, choose the maximal cell nearest their centroid. Shift windows at image borders to retain their size. The released PDiscoFormer CUB models use the scaled 74×74 window described in Section D.5.
3. For each part, compute the fraction of retained images whose window contains its keypoint, using only images in which that part is visible.
4. Take the largest part-specific fraction as the concept's purity, then average over concepts active in at least one image.

Kp NME. Keypoint regression tests whether concept maps locate the 15 CUB keypoints, following PDiscoNet and PDiscoFormer (van der Klis et al., 2023; Aniraj et al., 2024). Each concept map supplies its two centroid coordinates, normalized by image size, and its mass. Features are indexed by global concept identity, with zeros for an absent concept. We rank concepts by training-set usage and select the basis size from $\{16, 32, 64, 128, 256, 512\}$ using a 20% holdout drawn from the training split with seed 0. Candidate sizes exceeding the available basis are skipped. If fewer than 16 concepts are available, we use all of them. After selection, the regressor is refitted on the full training split. For each keypoint, ridge regression with regularization 10^{-3} is fitted using training images where that keypoint is visible.

Kp NME is 100 times the mean Euclidean error over all visible test keypoints in image-normalized coordinates. Our PDiscoFormer entries in Tables 3 and 16 are recomputed with this evaluator. The original paper's bounding-box-normalized errors are a different quantity.

Part-to-part retrieval. The benchmark in Section 4.1 compares regions through globally indexed concept identities.

1. *Regions.* On PartImageNet, each part label covering at least two cells of the scoring grid defines one region per image, giving 3,524 regions at 32×32. On CUB, each visible keypoint defines a one-cell region. To bound the quadratic cost of ranking, we sample 8,000 regions uniformly without replacement using seed 0. The 224×224 CUB evaluation of DisParQ yields 68,224 regions before sampling. Counts can differ slightly for the 518×518 PDiscoFormer inputs.
2. *Descriptors.* A region is represented by a histogram of the global concept IDs assigned to its cells, normalized to sum to one. For DisParQ, the histogram counts the globally indexed identities p_u.
3. *Ranking.* Each region queries regions from other images by cosine similarity, with random tie-breaking using seed 0. The same region set supplies queries and gallery candidates.
4. *Scoring.* A retrieved region is relevant when it has the query's part label. R@r and mAP follow the definitions above. Queries without a relevant gallery candidate contribute zero to recall and are excluded from mAP.

Rows marked ‡ in Tables 4 and 19 use different grids and region sets with their own oracles. CFM's 28×28 grid gives 3,466 PartImageNet regions and oracle R@1 44.5. Part-to-image retrieval compares the region histogram with a whole-image concept histogram, also normalized to sum to one. It excludes the query image and treats an image as relevant when it contains the queried part.

Oracle and chance. For part retrieval, the category-only reference is implemented as a co-occurrence-group oracle. It ranks regions sharing the query's label group before other regions, with random ordering within both sets. We construct groups from the evaluated split by visiting part labels in label order. Each unassigned label starts a group and adds the remaining unassigned labels whose image-presence sets have Jaccard overlap of at least 0.6 with that seed label. This is a greedy grouping around the seed, without transitive merging.

For part-to-image retrieval, each gallery image receives the group of its lowest-index retained part label, or a separate sentinel group if it has no retained parts. Images sharing the query's group are ranked first, with random ordering within both sets.

On PartImageNet, these groups capture supercategory co-occurrence, with 38 of the 40 part labels in 12 groups. The construction also assigns groups to the remaining labels. This measures retrieval explained by co-occurrence without identifying the particular part. It does not use bird-species labels on CUB. Chance ranks all candidates randomly. Both reference scores average five random rankings. Classification and image-to-image oracle entries instead use the ground-truth image class directly.

Concept contribution. The qualitative explanations in Figure 1, Section 4.3 and Appendix G.5 use a separate linear classifier on $\hat{c}(x)$. It is fitted to features standardized using training-set means and standard deviations. Training uses AdamW for 200 epochs, learning rate 10^{-3} with cosine decay, weight decay 10^{-2}, batches of 1024 and label smoothing 0.1. Its predicted class defines the logit used for explanations, while the quantitative LP results use the probe in Section D.5.

We measure a concept's contribution by removing it from the discrete description and decoding the modified description again. This is an occlusion test (Zeiler & Fergus, 2014) at the concept level, related to concept interventions (Koh et al., 2020). The removed concept's cells are reassigned to the strongest remaining concept among the already selected IDs $i_{1:K}$. The concept embeddings and attribute codes remain fixed, so only the spatial identity map p changes. We measure the drop in the original predicted-class logit and normalize the positive drops across the selected concepts to sum to one. These weights describe relative positive support under the intervention, rather than an additive decomposition of the classifier's logit.

E ABLATION STUDIES

Shared protocol. We organize the ablations around the components of Section 3 and Appendix C.

To reduce computational cost, most component studies train the continuous first stage with bilinear feature upsampling instead of AnyUp.

Unless specified otherwise, the component control uses PartImageNet, frozen DINOv2-B/14 with registers, 224×224 inputs, a 32×32 assignment/decoder grid, $N = 1024$, $K = 16$, and a three-block width-768 decoder. It follows the 150-epoch stage-1 schedule with batch size 16 in Table 12. Each alternative changes the named component, retaining the remaining settings. Quantization uses a separate, equally trained continuation control, described in Appendix E.3. Backbone and concept-budget comparisons specify their own references below.

Reading the results. Unless stated otherwise, gray reference rows report absolute scores and colored entries report variant-minus-reference differences. Percentage-valued metrics are shown on a 0–100 scale, and their differences are in percentage points (pp). Backbone impurity and quantization loss use raw units. Attribute rates are absolute bits per selected concept in every row. The backbone figure's final panel reports each model's LP and kNN differences from its own frozen target, without subtracting the DINOv2 reference again. Blue indicates improvement and red degradation, with separate color scales whose ranges are shown below each panel.

The fusion, upsampling, selection, appearance-removal, auxiliary-readout, decoder-depth, gradient, objective, and quantization comparisons report means over seeds 42, 43, and 44. Smaller numbers give sample standard deviations across absolute reference scores or paired variant-minus-reference differences. The backbone, bank-size, and concept-budget sweeps report single runs. We interpret small differences descriptively and do not infer significance from mean rankings. LP denotes linear-probe accuracy, and mAP denotes full-ranking image-retrieval mAP. CFM locality and consistency in the component panels use energy-threshold maps. Metric definitions, evaluation procedures, and training schedules are given in Appendices D.6, D.5, and D.3, respectively.

E.1 FEATURE EXTRACTION

Backbone and global target. Figure 8 evaluates the quantized model across frozen feature extractors. The sweep follows the canonical two-stage settings, with the backbone-specific feature, target, and grid adaptations described in Appendix D.2 and Table 9. Decoder width remains 768, including for CLIP-DINOiser's 512-dimensional projected features. Boundary-sensitive scores should be compared within a grid.

DINOv2 retains LP accuracy within 0.1 pp of its frozen target and achieves the strongest foreground purity in its grid group. DINOv3 leads most part-quality measures in the patch-16 group, although CLIP-DINOiser has higher CFM consistency (-3.0 versus -3.7 pp relative to DINOv2). Thus the metric ordering is not universal. SigLIP2's pooler improves LP by 2.7 pp over GAP (-2.4 versus -5.1 pp relative to DINOv2), with mixed part-quality changes. The target definition therefore matters as well as the patch features. The final panel separates representation retention from each frozen target's own probe accuracy.

Layer fusion. Removing the shared-query fusion of Appendix C.1 uses the final-block patch features alone. Figure 9 shows essentially unchanged downstream performance ($+0.17$ pp LP and -0.05 pp mAP), but foreground purity falls by 3.16 pp and mIoU by 1.44 pp. Multi-layer fusion therefore contributes primarily to concept quality in this comparison, rather than increasing downstream accuracy.

Upsampling. Figure 10 uses continuous AnyUp training as its reference. Replacing image-guided upsampling with bilinear interpolation at the same 32×32 resolution changes LP by $+0.22$ pp and mAP by -0.03 pp. Bilinear improves foreground purity by 2.07 pp but lowers mIoU and locality by 1.39 and 1.47 pp. These mixed differences support its use as a cheaper component-ablation control without establishing equivalence on every metric. The canonical method retains AnyUp for its slightly more detailed qualitative maps.

Native resolution and mixed-route training. Keeping the native 16×16 grid removes upsampling and changes the decoder grid accordingly. Relative to AnyUp, LP changes by -0.04 pp, but

	Downstream		Part quality					Grounding		Impurity	vs. frozen target	
Backbone	LP ↑	kNN ↑	Pur_{fg} ↑	NMI_m ↑	ARI_m ↑	mIoU ↑	FG-ARI ↑	Loc. ↑	Cons. ↑	Imp. ↓	ΔLP	ΔkNN
First four panels: DINOv2 absolute, other rows minus DINOv2. Final panel: own-target gaps.												
DINOv2-B/14	89.2	88.7	81.5	69.8	62.3	35.9	26.3	36.4	16.3	0.385	-0.1	-0.2
DisParQ on the other backbones												
DINOv3-B/16	-2.2	-2.8	-5.4	-4.2	-4.9	-2.8	-3.7	-3.5	-3.7	-0.099	-1.9	-1.8
TIPSv2-B/14	-3.6	-5.9	-10.4	-9.4	-13.7	-6.1	-7.6	-7.5	-5.8	-0.118	-1.5	-1.4
CLIP-B/16	-6.3	-9.4	-12.8	-12.0	-17.8	-4.9	-5.7	-6.5	-4.3	-0.103	-1.4	-1.2
SigLIP2-B/16	-2.4	-4.0	-12.1	-16.8	-25.9	-6.1	-5.1	-7.8	-5.6	-0.069	-0.2	-0.2
SigLIP2-B/16 (GAP)	-5.1	-13.4	-11.6	-16.0	-27.9	-6.1	-5.2	-7.1	-5.4	-0.064	-0.5	-1.9
CLIP-DI-B/16	-9.4	-13.0	-20.6	-7.6	-17.0	-11.2	-12.6	-7.7	-3.0	+0.008	-3.5	-5.5
	±13.4 pp		±17.8 pp					±7.8 pp		±0.118	±5.5 pp	

Figure 8: **DisParQ across backbones.** Quantized models on PartImageNet validation across seven configurations of six frozen feature extractors, with $K = 16$ and $N = 1024$, using the canonical seed 42. In the first four panels, the gray DINOv2 row reports absolute scores and other rows report differences from that reference. Percentage-valued metrics use pp for differences, while impurity uses raw entropy units. The final panel reports each model's LP and kNN gaps relative to its corresponding frozen global target, without subtracting the DINOv2 row again. Blue indicates improvement and red degradation. Targets and scoring grids are specified in Table 9. Patch-14 and patch-16 models use 32×32 and 28×28 grids, respectively, which can affect boundary-sensitive scores.

	Representation		Spatial structure		Semantic quality		
	LP	mAP	FG-ARI	mIoU	FG-Purity	CFM Cons.	CFM Loc.
Bilinear baseline	89.49 (±0.15)	73.79 (±0.05)	20.92 (±0.57)	33.21 (±0.22)	79.14 (±0.51)	15.06 (±0.82)	33.24 (±0.66)
No layer fusion	+0.17 (±0.11)	-0.05 (±0.03)	-0.21 (±0.29)	-1.44 (±0.31)	-3.16 (±0.10)	-1.45 (±0.71)	-0.96 (±0.52)
	-10 -5 0 5 10		-8 -4 0 4 8				
	Representation: ±10 pp		Spatial and semantic metrics: ±8 pp				

Figure 9: Layer fusion versus final-block features under the bilinear stage-1 control. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

foreground purity falls by 6.55 pp, while FG-ARI rises by 2.42 pp, illustrating sensitivity to spatial granularity. The mixed alternative selects AnyUp with probability 0.2 and bilinear interpolation otherwise, sharing the draw across both views of a local batch, and uses full AnyUp for evaluation. It remains close in LP and mAP (+0.08 and −0.05 pp) and improves foreground purity by 2.17 pp in Figure 10. Expensive image-guided upsampling on every training batch is therefore not required to retain the measured downstream performance.

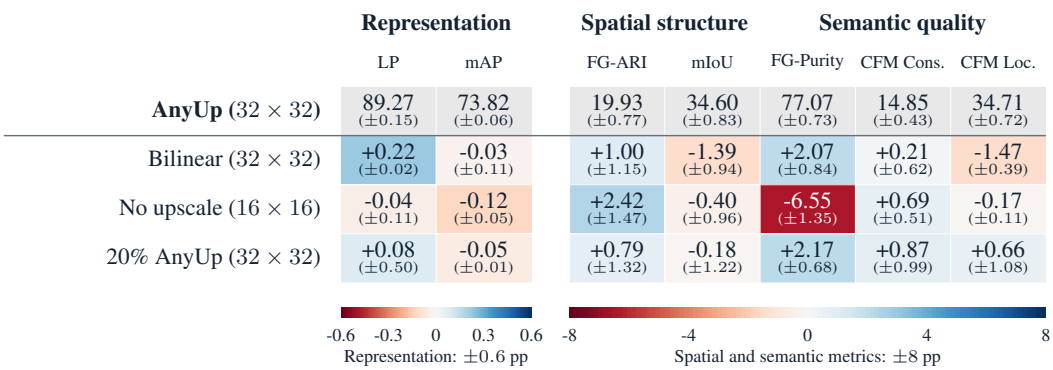


Figure 10: Feature upsampling with continuous AnyUp training as the reference. The 20% AnyUp arm mixes AnyUp and bilinear interpolation during training and uses full AnyUp at evaluation. Native resolution changes both the assignment and decoder grids to 16×16. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

E.2 Concept bank, selection, and maintenance

Bank size. Figure 11 varies the global vocabulary N from 1024 to 8192 at fixed $K = 16$. The budget sweeps use continuous DINOv2-B/14 discovery with AnyUp on a 32×32 grid and the 150-epoch stage-1 schedule in Table 12, sharing the $N = 1024, K = 16$ reference. Each setting is a single completed run. Increasing N to 8192 raises foreground purity by 4.05 pp, but FG-ARI and mIoU fall by 1.85 and 2.53 pp, while consistency and locality fall by 10.47 and 3.23 pp. LP increases by 0.41 pp. Thus higher purity with a larger vocabulary does not translate into better spatial segmentation or more consistent concepts across images. The canonical bank balances these competing properties. Interpret purity alongside consistency using the metric definitions in Appendix D.6, since finer fragmentation can increase purity.

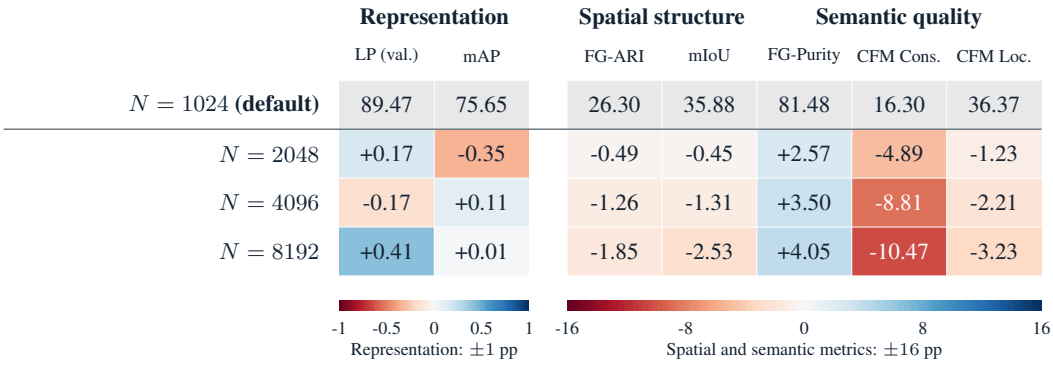


Figure 11: Concept-bank size N at fixed $K = 16$ during continuous DINOv2-B/14 discovery with AnyUp, using the sweep setup in Appendix E.2. The gray $N = 1024$ row gives absolute reference scores, and colored entries give differences in pp. Each row is one completed run.

Concepts per image. Figure 12 varies K from 4 to 64 at fixed $N = 1024$. The $K = 32$ and $K = 64$ configurations also increase the equivariance-map budget to 64 and 128, respectively, so these comparisons vary both budgets. The treatment of unmatched parts in mIoU follows Appendix D.6. With $K = 4$, LP remains close to the reference (+0.50 pp), but FG-ARI and mIoU fall by 14.37 and 15.31 pp, and consistency and locality fall by 12.49 and 16.00 pp. Increasing K beyond 16 also fails to improve these concept metrics. At $K = 32$, FG-ARI, mIoU, and foreground purity fall by 1.83, 2.17, and 3.84 pp. The per-image budget controls the spatial decomposition, and downstream accuracy alone does not identify a useful concept budget. At $N = 1024$, $K = 16$ gives the strongest reported spatial and semantic scores among the tested values.

	Representation		Spatial structure		Semantic quality		
	LP (val.)	mAP	FG-ARI	mIoU	FG-Purity	CFM Cons.	CFM Loc.
$K = 4$	+0.50	-0.14	-14.37	-15.31	-11.80	-12.49	-16.00
$K = 8$	+0.41	-0.22	-7.16	-9.54	-7.48	-8.74	-11.15
$K = 16$ **(default)**	89.47	75.65	26.30	35.88	81.48	16.30	36.37
$K = 32$	-0.25	+0.13	-1.83	-2.17	-3.84	-0.19	-1.32
$K = 64$	+0.33	+0.04	-2.57	-2.99	-4.73	-0.37	-2.13

-1 -0.5 0 0.5 1
Representation: ±1 pp

-16 -8 0 8 16
Spatial and semantic metrics: ±16 pp

Figure 12: Selected concept count K at fixed $N = 1024$ during continuous DINOv2-B/14 discovery with AnyUp. The sweep setup and accompanying equivariance budgets are described in Appendix E.2. The gray $K = 16$ row gives absolute reference scores, and colored entries give differences in pp. Each row is one completed run.

Joint bank and concept-budget sweep. Figure 13 extends these two slices to all 20 combinations of $K \in \{4, 8, 16, 32, 64\}$ and $N \in \{1024, 2048, 4096, 8192\}$. LP varies from 88.14% to 90.22%, while foreground purity and CFM consistency vary more strongly. The highest observed purity is 86.60% at $(K, N) = (32, 8192)$, whereas the default $(16, 1024)$ has the highest observed consistency (16.30%). Increasing the bank size therefore does not uniformly improve concept quality, and the effect depends on the selected concept budget. These single-run comparisons are descriptive.

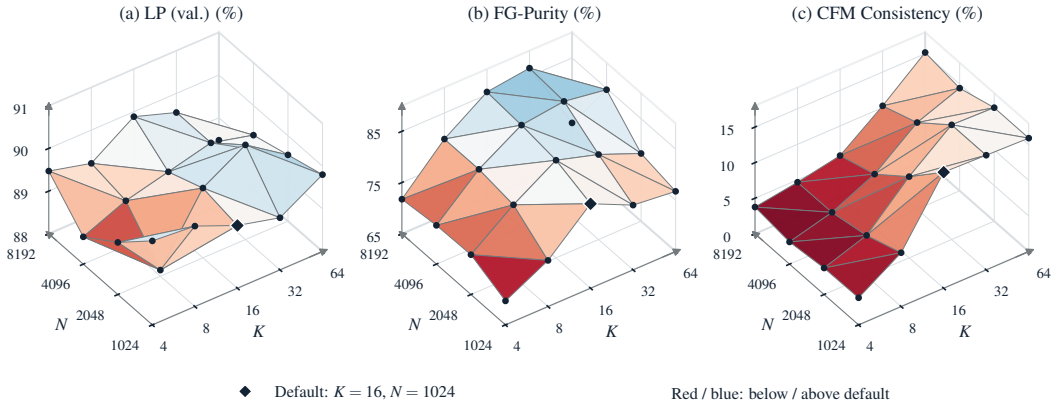


Figure 13: Joint sweep of selected concept count K and bank size N on PartImageNet with DINOv2-B/14, using the continuous AnyUp setup and equivariance budgets in Appendix E.2. Heights give absolute LP validation accuracy, foreground purity, and CFM consistency in percent. Vertical ranges differ across panels. Both horizontal axes use logarithmic (base-2) spacing. Dots mark all 20 measured configurations and the diamond marks the default $(K, N) = (16, 1024)$. Surfaces connect adjacent observations without smoothing. Red and blue indicate scores below and above the default, respectively, with color intensity scaled separately for each metric. Each configuration is a single completed run.

Peak rather than mean scoring. The canonical selector averages each full-bank map over space (Appendix C.2). The peak alternative instead takes its maximum, retaining Sinkhorn balancing and all other settings. Figure 14 shows small downstream changes ($+0.06$ pp LP, $+0.05$ pp mAP) but a 3.45 pp decrease in foreground purity. Selecting concepts by their most active cell does not improve the overall region semantics in this setting.

Coverage bias and direct mass selection. Setting the coverage-bias coefficient to zero retains Sinkhorn balancing and changes foreground purity by -0.79 ± 1.46 pp, with little mean change in FG-ARI or mIoU (Figure 14). These small differences relative to seed variation do not establish an independent benefit of the coverage term. Appendix C.2 explains its cancellation under ideal balancing. To compare the selectors, direct top-K selection by mean activation is contrasted with the zero-bias Sinkhorn row. LP differs by $+0.03$ pp, while consistency and locality fall by 1.38 and 1.63 pp.

Prototype resampling. Disabling the inactive-prototype replacement of Appendix C.6, while retaining usage tracking and selection, lowers mAP by 7.51 pp, foreground purity by 13.43 pp, and consistency by 8.52 pp (Figure 14). These are substantially larger changes than those from the scoring alternatives. Prototype replacement supports both retrieval and concept quality.

	Representation		Spatial structure		Semantic quality		
	LP	mAP	FG-ARI	mIoU	FG-Purity	CFM Cons.	CFM Loc.
Bilinear baseline	89.49 (±0.15)	73.79 (±0.05)	20.92 (±0.57)	33.21 (±0.22)	79.14 (±0.51)	15.06 (±0.82)	33.24 (±0.66)
Peak selector scoring	+0.06 (±0.31)	+0.05 (±0.08)	+0.18 (±0.63)	-0.20 (±0.42)	-3.45 (±0.59)	-1.26 (±1.48)	-0.26 (±0.65)
Zero coverage bias	-0.19 (±0.31)	+0.04 (±0.15)	+0.01 (±0.66)	-0.03 (±0.57)	-0.79 (±1.46)	-0.10 (±1.35)	+0.05 (±1.13)
Mass top-K, zero bias	-0.17 (±0.21)	-0.01 (±0.03)	+0.06 (±0.45)	-0.31 (±0.24)	-0.06 (±0.54)	-1.47 (±0.72)	-1.58 (±1.09)
No bank resampling	-2.31 (±0.06)	-7.51 (±0.45)	-3.81 (±0.49)	-6.91 (±0.09)	-13.43 (±1.27)	-8.52 (±1.25)	-6.41 (±0.96)

-10 -5 0 5 10 Representation: ±10 pp

-14 -7 0 7 14 Spatial and semantic metrics: ±14 pp

Figure 14: Selection rules and prototype maintenance under the bilinear stage-1 control. The zero-bias Sinkhorn and zero-bias mass rows form the direct selector comparison. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

E.3 Appearance attributes and quantization

Attributes during concept discovery. Removing the continuous appearance contribution sets $q_{i_k} = 0$ in the decoder of Appendix C.4, while learning concepts and retaining identity, geometry, and the auxiliary branch. Figure 15 shows losses of 8.98 pp LP and 6.22 pp mAP, although FG-ARI and mIoU rise slightly. Concept identities and geometry alone do not retain the same downstream information. This discovery-stage experiment differs from removing attributes after the concept bank has been learned, considered next.

	Representation		Spatial structure		Semantic quality		
	LP	mAP	FG-ARI	mIoU	FG-Purity	CFM Cons.	CFM Loc.
Bilinear baseline	89.49 (±0.15)	73.79 (±0.05)	20.92 (±0.57)	33.21 (±0.22)	79.14 (±0.51)	15.06 (±0.82)	33.24 (±0.66)
No attributes	-8.98 (±1.62)	-6.22 (±1.23)	+0.53 (±1.68)	+0.66 (±0.49)	-2.35 (±0.70)	-0.50 (±1.85)	+1.35 (±1.42)

-10 -5 0 5 10 Representation: ±10 pp

-8 -4 0 4 8 Spatial and semantic metrics: ±8 pp

Figure 15: Continuous appearance attributes during concept discovery under the bilinear stage-1 control. This differs from removing attributes after freezing concept assignment. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

Quantization protocol and rate. Figure 16 compares attribute encoders with the same seed-matched bilinear stage-1 parent. Each arm freezes concept assignment, retains the decoder, and receives the same 200-epoch continuation under the PartImageNet stage-2 schedule in Table 12. The continuous arm retains its learned projection, while the discrete arms initialize fresh quantization modules. The reference is private PQ with $G = 16$ groups and $M = 64$ entries per group, as defined in Appendix C.3. A private codebook is indexed by concept identity. A shared codebook serves all concepts. Rates count only attribute indices per selected concept, excluding identities, geometry, and model parameters. All discrete alternatives use nearest-codeword forward values and differentiable soft-mixture surrogates. Their codebooks and input projections are learned through distillation. These arms compare codebook structures, using our common training objective and surrogate rather than the original VQ-VAE or SoundStream training losses.

Continuous and absent attributes. Continuing with continuous attributes instead of PQ improves LP and kNN by 0.46 and 0.71 pp, while mAP falls by 0.31 pp (Figure 16). Both arms receive the same continuation, controlling for additional training time. Removing attributes from this frozen-assignment bilinear control lowers LP by 12.24 pp and mAP by 11.14 pp relative to private PQ. Both comparisons support retaining appearance information.

Private low-rate quantizers. Vector quantization (VQ) represents a full vector by a single codeword (Linde et al., 1980; van den Oord et al., 2017). Private VQ64 selects one full-width vector from 64 entries (6 bits per concept). Private PQ2×8 (Jégou et al., 2011) splits the projected residual into two channel groups with eight entries each, also using 6 bits. Relative to private PQ16×64, they lose 1.69 and 2.05 pp LP, respectively, and both increase distillation loss by approximately 0.05 (Figure 16). Very compact attributes retain much more information than removing attributes, but incur a larger cost than the 96-bit reference. These comparisons jointly change the attribute-index rate and quantizer structure and capacity, rather than isolating bit rate alone.

Shared 12-bit quantizers. Shared VQ4096 selects a single full-width vector. Residual vector quantization (RVQ), as used in SoundStream (Zeghidour et al., 2022), instead quantizes successive residuals. Shared RVQ64+64 sums a first 64-entry codeword and a second 64-entry codeword fitted to the remaining residual. Shared PQ2×64 concatenates two channel-group codewords (Jégou et al., 2011). Each uses 12 bits per concept. Their LP changes are -1.84, -2.23, and -2.41 pp, respectively, with mAP changes of -0.40, -0.89, and -0.94 pp (Figure 16). At this rate, full-vector VQ has the highest mean LP and mAP among these three alternatives, but all trail the higher-rate reference.

Shared versus private product codebooks. Shared PQ16×64 matches the reference's 96-bit attribute-index rate while sharing its codebooks across concepts. It changes LP by -0.17 pp, mAP by $+0.03$ pp, and kNN by $+0.07$ pp (Figure 16). Thus private codebooks do not provide a uniform downstream advantage in this comparison. They allow concept-specific appearance dictionaries, at the parameter cost specified in Appendix C.4. Sharing is a competitive alternative when reducing codebook storage is important.

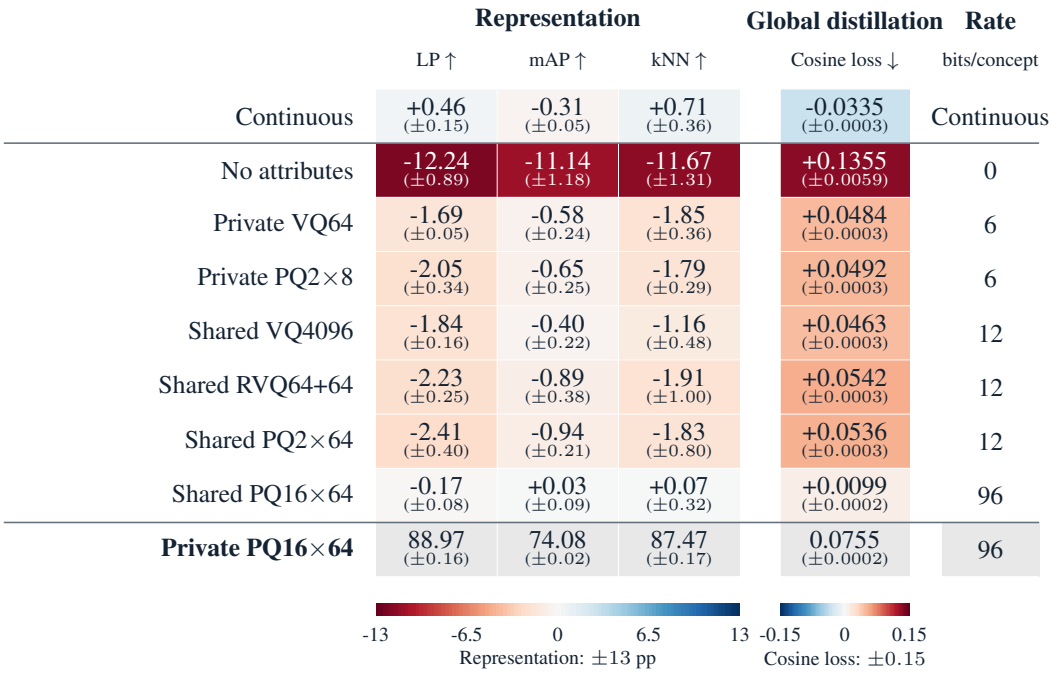


Figure 16: Attribute quantizers after equal 200-epoch continuation from a seed-matched, frozen bilinear concept model. Entries are means over three seeds. The gray private PQ16×64 row gives absolute reference scores, and other score entries give paired differences, in pp for LP, mAP, and kNN. Sample SDs appear beneath scores, using paired-difference SDs outside the reference row. Cosine loss is the single-view evaluation cosine distance between the decoder output and frozen global target, in raw units. Rates are absolute attribute-index bits per selected concept in every row and exclude model parameters.

E.4 Concept decoder

Pooling instead of spatial attention. The zero-block alternative mean-pools the spatial tokens t_u and applies the same normalization and output projection, omitting the CLS token and all attention blocks. Figure 17 shows losses of 1.26 pp LP and 2.06 pp mAP relative to the three-block decoder, with smaller concept-quality changes. This is the pooling-only alternative to Appendix C.4. The attention blocks improve integration of the spatial concept description.

Transformer depth. Figure 17 varies depth from zero to six blocks at fixed width 768. Moving from one to three blocks improves LP by 0.51 pp and mAP by 0.31 pp. Increasing from three to six roughly doubles decoder parameters (23.42M to 44.68M) for 0.11 pp higher mean scores in each downstream metric, while consistency falls by 2.07 pp. Three blocks offer a useful capacity trade-off. Deeper decoding does not uniformly improve concept quality.

Auxiliary-only readout. Removing the spatial decoder also removes its global loss and appearance branch. The pooled-region auxiliary head becomes the downstream representation, and checkpoints are selected by auxiliary validation distillation loss. Figure 18 shows -0.75 pp LP and $+0.35$ pp mAP, with relatively small concept-quality changes. This system comparison changes both training and readout. Comparing the same auxiliary readout in both systems instead gives approximately 0.00 ± 0.82 pp LP and -0.05 ± 0.15 pp mAP. Concepts can therefore be learned through auxiliary distillation and the assignment objectives alone. The auxiliary readout consumes continuous pooled region features. The stage-1 spatial decoder instead consumes concept identities, spatial assignments, and continuous appearance attributes, which are quantized in the final model.

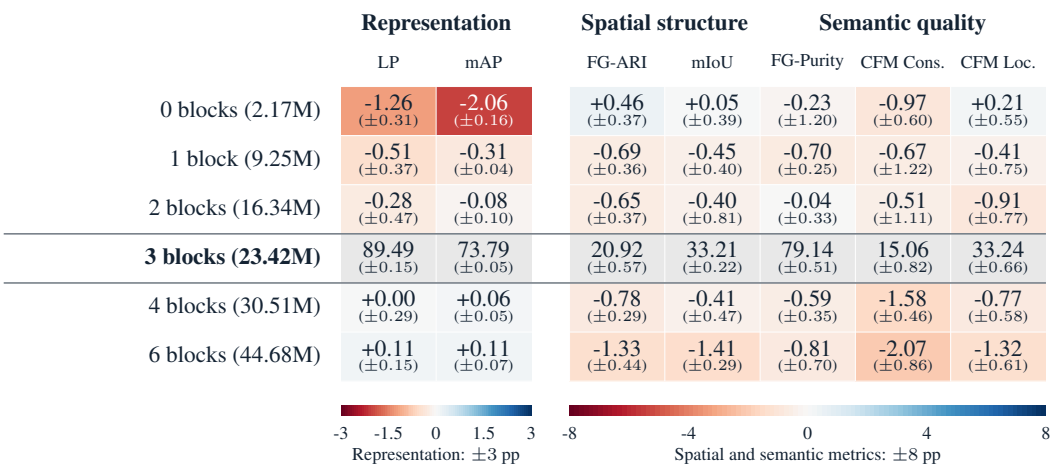


Figure 17: Decoder depth at fixed width 768 under the bilinear stage-1 control. Zero blocks denotes mean pooling, and labels include decoder parameter counts. The gray three-block row gives absolute reference scores. Colored entries are differences in pp, with SDs as defined in Appendix E.

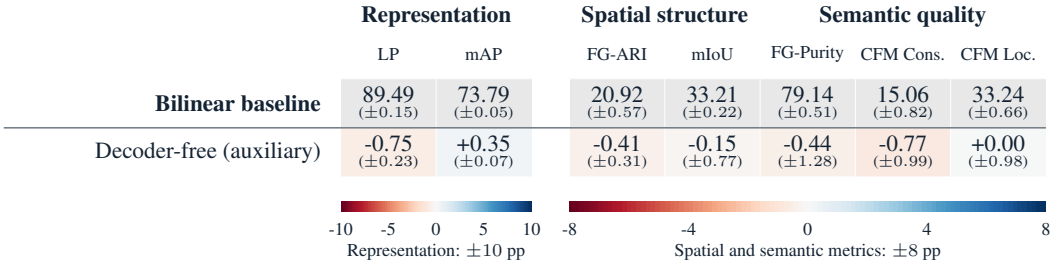


Figure 18: Spatial concept decoder versus the auxiliary-only region readout under the bilinear stage-1 control. The auxiliary readout consumes continuous pooled region features. The stage-1 decoder consumes concept identities, spatial assignments, and continuous appearance attributes. The comparison changes both training and readout, as discussed in Appendix E.4. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

E.5 Gradient routes

Identity surrogate. Disabling Equation 13 preserves hard forward identities but removes the decoder's gradient to soft assignments through identity embeddings. Other training objectives remain active. Figure 19 shows -0.28 pp LP, -0.79 pp FG-ARI, and -1.00 pp consistency. The surrogate has modestly higher mean scores under this control, alongside the other sources of concept-learning gradients.

Residual gradients. The canonical residual in Appendix C.3 is detached. Restoring its derivative lets global distillation update pooled features and matching prototypes through the appearance projection as well as through the identity surrogate. LP and mAP change by -0.08 and $+0.04$ pp, while FG-ARI, mIoU, and foreground purity fall by 1.10, 0.67, and 0.69 pp (Figure 19). These results do not motivate adding this gradient route to the canonical model.

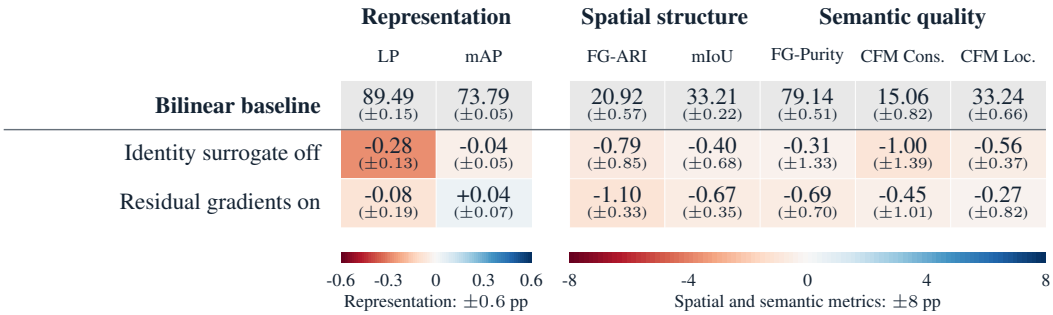


Figure 19: Gradient routes under the bilinear stage-1 control. The identity-surrogate removal preserves hard forward identities, while the residual-gradient arm enables derivatives through the appearance residual. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

E.6 Training objectives

We study the contribution of each auxiliary or regularization objective by setting one non-global loss weight in Table 6 to zero during continuous training, while retaining the global decoder loss and all remaining weights. Figure 20 reports the resulting changes. The objectives and weights are defined in Appendix C.5. Across these ablations, several objectives affect region quality more strongly than downstream performance. When auxiliary distillation ($\mathcal{L}_{\text{aux}}$) is removed, LP and mAP change little, but foreground purity and locality fall by 1.38 and 1.51 pp. The auxiliary signal supports region quality even when the retained spatial decoder preserves downstream information. Removing feature-weighted Potts regularization ($\mathcal{L}_{\text{Potts}}$) lowers LP by only 0.12 pp, but mIoU and consistency fall by 6.94 and 7.91 pp. Feature-aware spatial regularization chiefly supports concept organization rather than downstream accuracy. Removing $\mathcal{L}_{\text{orth}}$ lowers LP by 0.11 pp and both mIoU and consistency by 1.25 pp. Its effects are modest relative to Potts, but generally favor retaining distinct modulated region features.

The assignment and cross-view objectives reveal trade-offs between foreground purity and spatial and grounding metrics. Without assignment entropy regularization ($\mathcal{L}_{\text{ent}}$), FG-ARI and mIoU rise by 2.45 and 1.65 pp, and consistency and locality rise by 1.28 and 2.16 pp, while foreground purity falls by 0.87 pp. The regularizer favors purity in this comparison, while several spatial and grounding metrics favor its removal. Similarly, without affine equivariance ($\mathcal{L}_{\text{eq}}$), mAP is unchanged and FG-ARI rises by 0.34 pp, but foreground purity and consistency fall by 4.76 and 2.14 pp. Cross-view agreement supports semantic stability beyond the quality of a single-image partition. Removing selected-region presence ($\mathcal{L}_{\text{slot}}$) raises FG-ARI by 1.28 pp and mIoU by 0.66 pp but lowers foreground purity by 2.70 pp. Local activation of selected regions favors purity, with mixed effects on partition metrics.

When usage diversity ($\mathcal{L}_{\text{div}}$) is removed, LP changes by +0.01 pp, but foreground purity and locality fall by 7.04 and 4.82 pp. Regularizing full-bank activation therefore remains useful alongside balancing the subset-selection scores. In comparison, removing bank presence ($\mathcal{L}_{\text{bank}}$) lowers foreground purity by 0.68 pp and locality by 0.19 pp.

	Representation		Spatial structure		Semantic quality		
	LP	mAP	FG-ARI	mIoU	FG-Purity	CFM Cons.	CFM Loc.
Bilinear baseline	89.49 (±0.15)	73.79 (±0.05)	20.92 (±0.57)	33.21 (±0.22)	79.14 (±0.51)	15.06 (±0.82)	33.24 (±0.66)
No auxiliary distillation	-0.03 (±0.28)	+0.00 (±0.06)	-0.57 (±0.48)	-0.53 (±0.41)	-1.38 (±0.80)	-0.93 (±1.09)	-1.51 (±0.89)
No Potts	-0.12 (±0.52)	+0.03 (±0.15)	-3.35 (±0.91)	-6.94 (±0.25)	-6.96 (±0.99)	-7.91 (±0.81)	-6.85 (±0.78)
No assignment entropy	-0.28 (±0.35)	-0.03 (±0.10)	+2.45 (±1.28)	+1.65 (±1.10)	-0.87 (±0.12)	+1.28 (±0.28)	+2.16 (±1.29)
No orthogonality	-0.11 (±0.05)	-0.01 (±0.09)	-0.87 (±0.27)	-1.25 (±0.44)	-0.50 (±0.87)	-1.25 (±1.24)	-1.20 (±0.83)
No equivariance	-0.28 (±0.28)	+0.00 (±0.07)	+0.34 (±0.72)	-0.85 (±1.11)	-4.76 (±1.38)	-2.14 (±1.14)	-1.07 (±0.75)
No selected-region presence	-0.06 (±0.54)	+0.07 (±0.03)	+1.28 (±0.64)	+0.66 (±0.44)	-2.70 (±0.91)	-0.58 (±1.62)	+0.44 (±1.24)
No usage diversity	+0.01 (±0.31)	+0.11 (±0.05)	+0.55 (±0.94)	-3.60 (±0.46)	-7.04 (±1.25)	-3.50 (±0.90)	-4.82 (±0.50)
No bank presence	-0.12 (±0.11)	-0.03 (±0.07)	-0.16 (±0.65)	-0.16 (±0.20)	-0.68 (±1.65)	-0.69 (±0.66)	-0.19 (±0.59)



Figure 20: Individual auxiliary or regularization loss removals under the bilinear stage-1 control, retaining the global decoder loss. Objectives are defined in Appendix C.5. Gray entries are absolute reference scores, and colored entries are differences in pp, with SDs as defined in Appendix E.

F ADDITIONAL RESULTS

This section gives the full versions of the main-paper tables, in their order (Sections F.1–F.6), explains how we visualize quantized attributes (Section F.7), and reports the user-study details and results (Section F.8).

F.1 LARGE-SCALE CLASSIFICATION IN FULL

Table 15 lists every method of the comparison summarized in Figure 5 (left) of the main paper, with the same properties and the same ranking.

Figure 5 (left) keeps one supervised method and the three vision-language methods with the highest ImageNet accuracy, preferring methods that are also baselines elsewhere in the paper; Table 15 lists all of them. In the supervised block, InfoDisent and ProtoQuant train their heads on a frozen, label-supervised ViT-B/16. The other rows report a linear probe on the backbone or on the concept representation, except DCLIP, which scores classes by their descriptions (Menon & Vondrick, 2023). Following CFM (Wittenmayer et al., 2026), LF-CBM (Oikarinen et al., 2023), LaBo (Yang et al., 2023), CDM (Panousis et al., 2023), DCLIP, DN-CBM (Rao et al., 2024) and D-CBM (Prasse et al., 2025) learn global concepts, CF-CBM (Panousis et al., 2024) region-level ones, and InfoDisent (Struski et al., 2024), SALF-CBM (Benou & Riklin-Raviv, 2025), ProtoQuant (Janusz et al., 2026) and CFM spatial ones.

Table 15: **Large-scale classification, all methods** (linear probe, top-1). ✓ yes, (✓) partially, × no, as in Table 1; ∅: not applicable; —: not reported. **Best** and second best among self-supervised methods; gray rows are unranked references. Vision-language results follow Fig. 6 of CFM (Wittenmayer et al., 2026); supervised ViT-B/16 results are from Table 4 of ProtoQuant (Janusz et al., 2026), using the CLS backbone baseline. †DCLIP uses description-based scoring, not a linear probe.

Method	Concept representation: Spatial grounding	Concept representation: Discrete bottleneck	Top-1 accuracy: ImageNet-1k ↑	Top-1 accuracy: Places ↑
Fully supervised, ViT-B/16 (reference)				
Backbone	∅	∅	81.1	—
InfoDisent (Struski et al., 2024)	(✓)	×	79.2	—
ProtoQuant (Janusz et al., 2026)	(✓)	(✓)	80.1	—
Vision-language self-supervised, CLIP ViT-B/16				
Backbone	∅	∅	80.2	55.1
LF-CBM (Oikarinen et al., 2023)	×	×	75.4	50.6
LaBo (Yang et al., 2023)	×	×	78.9	—
CDM (Panousis et al., 2023)	×	(✓)	79.3	52.6
DCLIP† (Menon & Vondrick, 2023)	×	×	68.0	40.3
DN-CBM (Rao et al., 2024)	×	×	79.5	55.1
D-CBM (Prasse et al., 2025)	×	×	70.5	50.9
CF-CBM (Panousis et al., 2024)	(✓)	(✓)	78.5	—
SALF-CBM (Benou & Riklin-Raviv, 2025)	(✓)	×	76.3	49.4
CFM (Wittenmayer et al., 2026)	(✓)	×	78.9	**55.4**
Vision-only self-supervised, DINOv2-B/14				
Backbone	∅	∅	83.4	55.4
DisParQ (ours)	✓	✓	**83.2**	55.3

Properties. The two property columns follow Table 1. *Spatial grounding* is full when every location carries exactly one concept, partial for region-level or overlapping concepts, and absent for global concepts. A *discrete bottleneck* is full when discrete indices alone define what the predictor reads, and partial when a discrete selection or codebook is combined with continuous scores, as in CDM, CF-CBM and ProtoQuant. DisParQ is the only method with both. The columns describe how each method is built, not how well it localizes.

F.2 Full PartImageNet comparison

We extend Table 2 by reporting every evaluated configuration under the same protocol, as differences to DisParQ on DINOv2-B/14. Figure 21 covers the **supervised methods**: all part counts K of PDiscoFormer, released and retrained, and PIP-Net and ProtoQuant with their classifier stage on every backbone we trained them on. Figure 22 covers the **self-supervised methods**: every concept budget of CFM including its own unbudgeted setting, the sparse-autoencoder baselines across dictionary sizes, budgets and backbones, and the label-free first stage of PIP-Net and ProtoQuant. Figure 23 shows the Matryoshka SAE across dictionary sizes, concept budgets, loss weightings and its own backbone. DisParQ on the other backbones in Figure 8, with the backbone study of Appendix E.1.

		Downstream		Part quality					Grounding		Impurity
Backbone	K/N	LP ↑	kNN ↑	Pur_{fg} ↑	NMI_m ↑	ARI_m ↑	mIoU ↑	FG-ARI ↑	Loc. ↑	Cons. ↑	Imp. ↓
DisParQ (ours): absolute values. Every row below is that row minus this one.											
DINOv2-B/14	**16/1024**	89.2	88.7	81.5	69.8	62.3	35.9	26.3	36.4	16.3	0.385
PDiscoFormer, released checkpoints											
DINOv2-B/14	8/8	-0.9	-0.5	-65.6	-56.0	-57.5	-12.3	-3.0	-11.7	-14.6	+0.163
DINOv2-B/14	16/16	+0.4	-0.2	-41.7	-28.9	-37.2	-7.7	+4.0	-4.5	-14.3	+0.156
DINOv2-B/14	25/25	+0.3	-1.2	-31.8	-26.8	-29.8	-7.8	+3.1	-2.7	-14.2	+0.114
DINOv2-B/14	50/50	+0.6	-0.6	-26.6	-21.4	-24.4	-6.8	+2.5	-1.5	-14.3	+0.120
PDiscoFormer†, our retraining of the authors' code											
DINOv2-B/14	8/8	-0.6	-0.6	-60.1	-56.6	-56.4	-9.8	-1.5	-7.3	-14.4	+0.198
DINOv2-B/14	16/16	-1.0	-1.1	-50.6	-51.7	-53.9	-7.7	+0.7	-4.0	-14.3	+0.178
DINOv2-B/14	25/25	-1.6	-1.2	-54.4	-52.7	-54.0	-6.8	-0.6	-2.8	-14.4	+0.151
DINOv2-B/14	50/50	-0.6	-0.7	-46.0	-43.4	-49.0	-6.2	+5.7	-0.3	-14.0	+0.075
PDiscoFormer†, with the position, CLS and register tokens frozen as well											
DINOv2-B/14	25/25	-1.1	-1.4	-54.1	-57.3	-57.0	-8.7	-2.4	-4.0	-14.6	+0.156
PIP-Net, stage 2 (self-supervised pretraining, then its classifier)											
DINOv2-B/14	16/768	-1.7	-1.9	-22.2	-32.5	-50.8	-11.0	-7.4	-11.2	-9.9	+0.052
DINOv2-B/14	16/1024	-1.4	-1.6	-25.6	-30.9	-50.4	-10.6	-6.9	-9.8	-9.6	+0.047
DINOv2-S/14	16/384	-3.8	-5.4	-44.5	-42.4	-57.1	-14.2	-8.2	-11.2	-11.2	-0.012
DINOv2-S/14	16/1024	-4.3	-4.1	-37.1	-37.7	-55.3	-12.9	-7.4	-9.4	-9.7	+0.020
ConvNeXt-T	16/768	-12.3	-12.8	-63.8	-63.8	-62.3	-25.0	-22.2	-21.3	-13.4	-0.087
ResNet-50	16/2048	-27.8	-69.0	-61.1	-66.6	-62.3	-19.9	-15.2	-21.3	-14.8	-0.108
ProtoQuant, stage 2 (codebook, then its classifier), frozen backbone											
DINOv2-B/14	16/1024	-1.3	+0.0	-11.3	-17.7	-38.8	-10.7	-8.2	-23.2	-10.8	+0.108
DINOv2-B/14	16/2048	-1.4	+0.8	-9.8	-14.2	-32.3	-9.4	-6.8	-23.3	-11.5	+0.128
DINOv2-B/14	16/4096	-1.6	+0.0	-7.8	-11.5	-27.5	-8.3	-6.1	-23.3	-12.3	+0.166
DINOv2-S/14	16/2048	-4.1	-4.5	-8.8	-12.1	-27.2	-8.1	-5.2	-22.4	-10.9	+0.145
DeiT-S/16	16/2048	-2.1	-3.9	-16.6	-18.8	-38.3	-10.3	-10.3	-23.3	-11.6	+0.264
ProtoQuant^{ft}, stage 1, on a backbone first fine-tuned on the labels (its own protocol)											
DINOv2-B/14	16/2048	-0.1	+0.0	-20.3	-61.7	-62.3	-26.6	-21.7	-27.7	-13.5	+0.222
DINOv2-S/14	16/2048	-3.5	-4.0	-14.8	-37.5	-59.8	-18.9	-14.2	-27.6	-12.6	+0.214
DeiT-S/16	16/2048	+0.8	+1.3	-18.2	-29.1	-55.6	-16.3	-15.1	-23.9	-11.6	+0.226
ProtoQuant^{ft}, stage 2											
DINOv2-B/14	16/2048	-1.1	+0.3	-18.7	-49.4	-61.2	-22.6	-18.6	-27.3	-12.8	+0.197
DINOv2-S/14	16/2048	-3.6	-2.0	-19.7	-58.1	-62.3	-24.6	-20.2	-27.5	-12.4	+0.204
DeiT-S/16	16/2048	+1.0	+1.8	-18.7	-35.8	-59.4	-17.4	-15.6	-26.6	-12.7	+0.211
PCA, class-wise (the ground-truth label selects one PCA per class)											
DINOv2-B/14	4/632	∅	∅	-26.3	-18.8	-30.8	-24.5	-17.9	-16.6	-12.6	+0.275
DINOv2-B/14	8/1264	∅	∅	-24.7	-17.3	-28.9	-22.9	-15.8	-15.0	-12.4	+0.260
DINOv2-B/14	16/2528	∅	∅	-23.6	-16.4	-27.9	-21.7	-14.1	-13.5	-12.0	+0.243
DINOv2-B/14	32/5056	∅	∅	-23.1	-16.0	-27.0	-21.5	-14.0	-13.3	-12.1	+0.240
DINOv2-B/14	64/10112	∅	∅	-23.0	-15.8	-26.9	-21.5	-14.0	-13.1	-12.0	+0.241
		±4.3 pp		±57.3 pp					±23.3 pp		±0.243

Figure 21: **PartImageNet: supervised methods against DisParQ.** The gray row gives DisParQ on DINOv2-B/14 in absolute numbers (protocol and columns of Table 2); every other row is that method *minus* DisParQ, in percentage points (raw units for Imp.). Blue: better than DisParQ; red: worse, so for Imp. (↓) a negative difference is blue. Color scales are per panel and clipped at the 90th percentile of $|\Delta|$; ∅ is undefined. Every method here trains with class labels. † is our retraining of the authors' code. ConvNeXt-T and ResNet-50 are PIP-Net's original CNNs under its published schedule. ProtoQuant^{ft} first fine-tunes the backbone on the labels, which is why its stage 1 is listed here. For PCA, K/N is components per class over components in total.

		Downstream		Part quality					Grounding		Impurity
Backbone	K/N	LP↑	kNN↑	$\mathrm{Pur}_{\mathrm{fg}}$↑	NMI_m↑	ARI_m↑	mIoU↑	FG-ARI↑	Loc.↑	Cons.↑	Imp.↓
DisParQ (ours): absolute values. Every row below is that row minus this one.											
DINOv2-B/14	**16/1024**	89.2	88.7	81.5	69.8	62.3	35.9	26.3	36.4	16.3	0.385
PCA, per image											
DINOv2-B/14	16/16	∅	∅	-66.6	-69.3	-62.1	-22.3	-15.7	-17.1	-15.0	-0.056
CFM, released 8192-latent dictionary											
CLIP-DI-B/16	16/8192	-16.6	-19.5	-26.8	-18.4	-37.8	-13.2	-13.8	-13.5	-3.9	-0.007
CLIP-DI-B/16	32/8192	-16.3	-19.5	-26.4	-17.5	-36.4	-12.1	-13.1	-6.5	-2.8	+0.016
CLIP-DI-B/16	64/8192	-17.3	-19.5	-26.4	-17.5	-36.4	-11.8	-13.1	+2.4	-2.1	+0.010
CLIP-DI-B/16	128/8192	-15.7	-19.5	-26.4	-17.4	-36.2	-11.8	-13.1	+5.3	-2.4	-0.020
CLIP-DI-B/16	∼85/8192	-16.0	-19.5	-26.4	-17.4	-36.2	-11.8	-13.1	+5.3	-2.3	-0.025
TopK SAE on DINOv2-B/14: dictionary size, then concepts per image											
DINOv2-B/14	16/1024	-11.1	-15.7	-13.3	-14.3	-31.0	-11.7	-12.0	-16.0	-6.8	+0.063
DINOv2-B/14	16/2048	-8.7	-15.2	-12.7	-11.8	-26.3	-11.1	-10.2	-15.8	-6.5	+0.077
DINOv2-B/14	16/4096	-8.2	-15.2	-12.4	-13.8	-30.4	-11.0	-10.7	-15.8	-7.5	+0.074
DINOv2-B/14	16/8192	-9.3	-15.0	-12.3	-15.1	-33.4	-11.1	-10.4	-15.9	-7.5	+0.075
DINOv2-B/14	32/1024	-7.4	-14.4	-11.1	-13.3	-31.0	-10.0	-11.1	-12.7	-4.1	+0.075
DINOv2-B/14	64/1024	-8.0	-13.1	-11.8	-14.4	-32.3	-10.2	-11.8	-9.0	-2.7	+0.078
DINOv2-B/14	128/1024	-5.9	-13.3	-12.6	-16.7	-36.3	-11.5	-12.4	-5.1	-5.4	+0.075
TopK SAE on the other backbones											
DINOv3-B/16	16/1024	-13.6	-22.7	-11.2	-3.6	-10.4	-8.7	-8.4	-14.6	-5.9	+0.112
TIPSv2-B/14	16/1024	-12.3	-22.2	-14.1	-9.1	-21.1	-10.2	-11.1	-15.4	-5.7	+0.063
CLIP-B/16	16/1024	-23.3	-36.0	-19.1	-15.6	-31.6	-13.0	-14.3	-17.6	-6.0	+0.102
SigLIP2-B/16	16/1024	-12.3	-20.6	-23.2	-31.7	-52.0	-15.2	-13.5	-20.5	-7.9	+0.199
CLIP-DI-B/16	16/1024	-21.1	-36.7	-18.9	-6.7	-15.4	-13.2	-13.1	-14.5	-6.5	+0.065
PatchSAE (authors' code and hyperparameters, our features): dictionary size											
DINOv2-B/14	16/1024	-6.3	-12.2	-15.8	-18.9	-41.5	-15.6	-15.3	-14.9	-7.3	-0.009
DINOv2-B/14	16/2048	-6.4	-12.2	-13.0	-16.3	-39.4	-14.4	-14.2	-13.8	-8.7	-0.042
DINOv2-B/14	16/4096	-5.7	-11.7	-11.8	-15.9	-37.7	-13.6	-14.2	-13.5	-10.2	-0.052
DINOv2-B/14	16/8192	-5.2	-11.4	-9.8	-13.5	-34.5	-12.6	-12.3	-12.7	-11.2	-0.027
DINOv2-B/14	16/49152	-5.1	-11.5	-5.3	-9.7	-27.4	-10.9	-9.3	-11.7	-11.9	+0.070
MSAE (authors' code and hyperparameters, our features), default size											
DINOv2-B/14	16/6144	-4.0	-9.7	-22.1	-21.4	-42.4	-16.0	-15.1	-18.8	-8.6	+0.085
DINOv2-B/14	16/6144	-3.8	-9.3	-23.0	-26.8	-50.0	-16.6	-16.0	-19.9	-8.8	+0.056
PIP-Net, stage 1 (self-supervised pretraining only)											
DINOv2-B/14	16/768	-12.3	-17.2	-13.0	-15.6	-28.0	-6.9	-7.2	-5.9	-6.1	-0.009
DINOv2-B/14	16/1024	-8.2	-16.1	-13.3	-14.2	-24.4	-5.2	-4.8	-3.6	-4.8	-0.009
DINOv2-S/14	16/384	-22.5	-32.0	-25.5	-26.1	-49.1	-12.2	-12.1	-8.3	-6.5	-0.035
DINOv2-S/14	16/1024	-15.3	-22.8	-18.5	-16.6	-28.6	-7.8	-7.2	-5.1	-4.5	-0.011
ConvNeXt-T	16/768	-71.7	-85.5	-65.7	-69.6	-62.3	-28.0	-24.8	-23.1	-15.3	-0.170
ResNet-50	16/2048	-73.1	-81.2	-54.5	-63.6	-62.4	-18.6	-13.1	-20.8	-14.6	+0.252
ProtoQuant, stage 1 (codebook only), frozen backbone											
DINOv2-B/14	16/1024	-5.4	-8.4	-2.3	-3.2	-9.0	-3.3	+1.6	-18.1	-8.0	+0.102
DINOv2-B/14	16/2048	-4.0	-4.8	+0.6	-1.1	-6.0	-2.2	+2.1	-18.3	-9.0	+0.129
DINOv2-B/14	16/4096	-3.5	-4.0	+2.9	+0.3	-5.4	-2.7	+0.8	-19.2	-10.7	+0.174
DINOv2-S/14	16/2048	-8.8	-12.8	+0.8	+0.8	-2.0	-1.9	+2.5	-17.5	-8.6	+0.111
DeiT-S/16	16/2048	-5.1	-8.8	-12.0	-23.3	-49.6	-12.6	-8.8	-23.6	-11.9	+0.282
Frozen backbone without any bottleneck (the ceilings for LP and kNN)											
DINOv2-B/14	CLS token	+0.1	+0.2	∅	∅	∅	∅	∅	∅	∅	∅
DINOv2-B/14	patch mean	-3.5	-9.4	∅	∅	∅	∅	∅	∅	∅	∅
CLIP-DI-B/16	CLS token	-5.9	-7.5	∅	∅	∅	∅	∅	∅	∅	∅
CLIP-DI-B/16	patch mean	-15.4	-30.9	∅	∅	∅	∅	∅	∅	∅	∅
		±23.3 pp		±36.4 pp					±18.1 pp		±0.170

Figure 22: **PartImageNet: self-supervised methods against DisParQ.** Reading as in Figure 21. $\sim 85/8192$ is CFM without a per-image budget, its own setting (every firing concept is kept). TopK SAE, PatchSAE and MSAE are trained on the same frozen patch tokens as DisParQ; PatchSAE and MSAE run their authors' code and hyperparameters (49152 and 6144 are their published widths; MSAE's other settings are in Figure 23) and, like CFM, receive the 16-concept budget only at scoring time. LP/kNN read what each method hands downstream: the pooled reconstruction for the SAEs, CFM's $[\max|\mathrm{mean}]$-pooled concept activations. The last block gives the frozen backbone itself: its CLS token, which DisParQ predicts, and its mean patch token, the ceiling for a pooled-reconstruction probe.

		Downstream		Part quality					Grounding		Impurity
Backbone	K/N	LP ↑	kNN ↑	Pur_{fg} ↑	NMI_m ↑	ARI_m ↑	mIoU ↑	FG-ARI ↑	Loc. ↑	Cons. ↑	Imp. ↓
DisParQ (ours): absolute values. Every row below is that row minus this one.											
DINOv2-B/14	**16/1024**	89.2	88.7	81.5	69.8	62.3	35.9	26.3	36.4	16.3	0.385
Reverse weighting (RW): dictionary size, then concepts per image											
DINOv2-B/14	16/1024	-4.0	-8.8	-20.4	-20.6	-42.1	-15.5	-15.1	-21.8	-9.5	+0.113
DINOv2-B/14	16/2048	-4.0	-9.3	-21.5	-21.2	-42.8	-15.7	-14.9	-19.5	-8.9	+0.064
DINOv2-B/14	16/4096	-3.9	-9.7	-22.1	-23.4	-44.8	-16.1	-15.1	-18.8	-8.6	+0.089
DINOv2-B/14	16/6144	-4.0	-9.7	-22.1	-21.4	-42.4	-16.0	-15.1	-18.8	-8.6	+0.085
DINOv2-B/14	16/8192	-3.6	-9.2	-22.5	-19.5	-38.8	-15.4	-14.2	-18.4	-8.2	+0.075
DINOv2-B/14	16/12288	-4.1	-9.8	-22.2	-20.2	-40.2	-15.9	-14.9	-18.6	-8.6	+0.089
DINOv2-B/14	16/24576	-3.8	-9.3	-21.8	-19.9	-39.6	-15.3	-14.3	-18.6	-8.4	+0.092
DINOv2-B/14	32/6144	-4.0	-9.7	-20.8	-20.9	-40.2	-14.9	-14.6	-15.5	-6.5	+0.061
DINOv2-B/14	64/6144	-4.0	-9.7	-20.1	-20.9	-40.1	-13.8	-14.0	-12.1	-4.4	+0.031
DINOv2-B/14	128/6144	-4.0	-9.7	-19.6	-20.8	-40.1	-13.4	-13.7	-9.2	-2.8	+0.002
Uniform weighting (UW)											
DINOv2-B/14	16/1024	-4.0	-9.1	-21.7	-23.3	-44.3	-15.8	-15.4	-22.9	-9.9	+0.122
DINOv2-B/14	16/2048	-3.5	-9.7	-23.0	-20.7	-41.5	-14.4	-12.4	-21.0	-9.8	+0.113
DINOv2-B/14	16/4096	-4.1	-9.2	-22.3	-25.4	-48.1	-16.8	-16.4	-20.0	-8.8	+0.061
DINOv2-B/14	16/6144	-3.8	-9.3	-23.0	-26.8	-50.0	-16.6	-16.0	-19.9	-8.8	+0.056
DINOv2-B/14	16/8192	-3.4	-9.3	-22.6	-23.4	-45.6	-16.4	-15.2	-19.8	-8.9	+0.069
DINOv2-B/14	16/12288	-4.2	-9.4	-22.0	-23.7	-45.3	-16.1	-14.9	-19.8	-8.9	+0.088
DINOv2-B/14	16/24576	-3.9	-9.3	-21.9	-21.5	-42.3	-16.5	-15.6	-19.5	-8.4	+0.083
DINOv2-B/14	32/6144	-3.8	-9.3	-21.5	-26.0	-48.6	-15.3	-15.0	-17.0	-7.3	+0.072
DINOv2-B/14	64/6144	-3.8	-9.3	-20.8	-26.0	-48.5	-14.5	-14.7	-14.0	-6.1	+0.076
DINOv2-B/14	128/6144	-3.8	-9.3	-20.4	-25.9	-48.1	-14.0	-14.4	-11.9	-5.1	+0.070
On its own model, CLIP ViT-L/14 (expansion 8)											
CLIP-L/14	16/8192	-6.4	-16.8	-39.1	-34.7	-56.9	-14.8	-13.4	-19.7	-8.2	+0.143
CLIP-L/14	16/8192	-6.1	-16.8	-37.7	-34.6	-56.7	-15.0	-13.6	-19.4	-7.5	+0.156
		±9.7 pp		±42.8 pp					±19.8 pp		±0.113

Figure 23: **PartImageNet: the Matryoshka SAE across its settings, against DisParQ.** Reading as in Figure 21. The authors' code and hyperparameters, trained on the frozen patch tokens. 6144, 12288 and 24576 latents are its published expansion factors 8, 16 and 32 on the 768-d DINOv2-B/14 tokens (8192 is expansion 8 on CLIP ViT-L/14, the model it was published on); 1024 to 8192 are the sizes the other sparse autoencoders were swept over. The per-image concept budget is imposed at scoring time, so the 32-, 64- and 128-concept rows read the dictionary of the $16/6144$ row.

F.3 Full CUB concept metrics

Table 16 lists every concept metric we measure on CUB. Since CUB marks parts with keypoints, the clustering metrics are computed on the cells that hold a visible keypoint, and mIoU, FG-ARI and the grounding metrics are undefined (Appendix D.6). DisParQ leads the self-supervised methods on every column, by 3.5 to 8.5 points on the clustering metrics and by 17.2 on PIP purity. The supervised PDiscoFormer, trained with class labels and evaluated at 518×518, scores higher on the clustering metrics for $K \geq 8$. Its CUB checkpoints read 518×518 images, so we resample their 37×37 part maps to the 32×32 grid and scale the PIP purity window from 32 to 74 pixels (Appendix D.6). It also has a far lower keypoint error at every K (4.44, 2.93 and 2.87 for K=4, 8 and 16, against 9.36 for DisParQ), since its background channel concentrates its parts on the bird and it reads 518×518 images. Its PIP purity is 64.3 at K=4 and 8 and 40.4 at K=16, against 54.5 for DisParQ with 16 concepts.

Table 16: **All concept metrics on CUB-200-2011** (test split, frozen DINOv2-B/14; CFM: CLIP-DINOiser; $\text{MSAE}_{\text{CLIP-L/14}}$: MSAE on the model it was published on; PDiscoFormer: the released checkpoints, at 518×518); $\times 100$ except Kp NME. CUB is annotated with keypoints, not part masks, so mIoU, FG-ARI and the grounding triple of Table 2 are undefined. Marks, subscripts and gray rows as in Table 3; ∅: undefined for that method, —: not measured.

		Downstream		Concept quality				
Method	K/N	LP ↑	kNN ↑	Pur_{fg} ↑	NMI_m ↑	ARI_m ↑	PIP pur. ↑	Kp NME ↓
Supervised (reference)								
PDiscoFormer	4/4	—	—	28.6	55.1	27.3	64.3	4.44
PDiscoFormer	8/8	—	—	51.9	67.2	42.5	64.3	2.93
PDiscoFormer	16/16	86.8	74.8	53.6	65.6	46.0	40.4	2.87
PCA	16/3200	∅	∅	14.3	3.5	0.4	—	—
$\text{PIP-Net}_{\text{stage 2}}$	16/768	88.9	88.9	42.0	37.0	21.3	47.0	10.16
$\text{ProtoQuant}_{\text{stage 2}}$	16/4000	84.9	83.0	41.0	33.3	21.2	25.2	12.16
Frozen backbone, no concept bottleneck (reference)								
CLS token	—	90.4	86.7	—	—	—	—	—
Self-supervised								
CFM	16/8192	54.8	42.8	12.5	4.8	0.6	16.1	12.84
TopK SAE	16/2048	36.9	24.3	21.2	17.0	7.4	8.7	10.92
PatchSAE	16/49152	62.0	33.4	18.3	5.0	2.3	6.9	19.12
MSAE	16/6144	73.5	42.2	13.8	3.2	1.1	15.1	12.21
$\text{MSAE}_{\text{CLIP-L/14}}$	16/8192	71.5	36.5	18.3	15.2	4.4	8.8	11.44
$\text{PIP-Net}_{\text{stage 1}}$	16/768	77.8	68.4	39.7	32.7	16.4	37.3	9.87
$\text{ProtoQuant}_{\text{stage 1}}$	16/4000	80.1	67.7	44.0	36.4	24.3	24.6	11.59
DisParQ (ours)	16/1024	**89.4**$_{+9.3}$	**85.5**$_{+17.1}$	**47.5**$_{+3.5}$	**44.9**$_{+8.5}$	**29.6**$_{+5.3}$	**54.5**$_{+17.2}$	**9.36**$_{-0.51}$

F.4 PDiscoFormer on Cars and Dogs

PDiscoFormer releases no checkpoint for Stanford Cars or Stanford Dogs, so Table 3 leaves those cells empty. Table 17 gives our retraining with the authors' code and recipe (Appendix D.4), at 224×224 with K=16 parts, on the same training splits as DisParQ. As in Table 2, the probes read the mean of its modulated part embeddings. Trained with class labels, PDiscoFormer matches DisParQ on the linear probe on Cars (91.1 vs. 91.0) and trails it by 2.3 to 4.9 p.p. on the other three columns. Its recipe keeps the snapshot with the best test accuracy, which is that of epoch 27 of 28 on both datasets, so this selection hardly touches the test split.

F.5 ProtoQuant and PIP-Net: frozen vs. fine-tuned backbone

Two of our baselines fine-tune the backbone in their own protocols. ProtoQuant trains it for classification on the target dataset and then freezes it, so the per-backbone accuracies its Table 3 reports are

Table 17: **PDiscoFormer reproduced on Cars and Dogs** (test, $\times 100$): the authors' code retrained at 224×224 with $K{=}16$. The other rows are from Table 3.

		Stanford Cars		Stanford Dogs	
Method	K/N	LP↑	kNN↑	LP↑	kNN↑
PDiscoFormer, reproduced	16/16	91.1	79.9	85.5	80.9
CLS token	—	93.2	85.6	88.6	86.8
DisParQ (ours)	16/1024	91.0	83.4	87.8	85.8

those of a fine-tuned network, and PIP-Net's recipe trains its backbone end to end. We report both on the frozen DINOv2-B/14 that every other method reads (Appendix D.4), and we also ran their protocols as published. Table 18 gives the pair, for completeness and as a control on that choice.

Fine-tuning improves classification accuracy and hurts part quality. ProtoQuant's stage-1 linear probe on PartImageNet rises from 85.2 to 89.1 and its kNN from 83.9 to 88.7, while its foreground purity falls from 82.1 to 61.2, its part mIoU from 33.7 to 9.3 and its part-to-part R@1 from 79.2 to 30.7. On CUB its purity falls from 44.0 to 16.3 and its keypoint error rises from 11.59 to 14.47, against the 15.50 of predicting the mean keypoint. PIP-Net loses the same part columns and, unlike ProtoQuant, loses accuracy with them: its stage-1 probe drops from 77.8 to 57.6 and its kNN from 70.6 to 31.8. What the backbone is fine-tuned *on* accounts for the difference. ProtoQuant's first stage trains it with class labels, so its features are class-discriminative before any codebook is learned, while PIP-Net trains it through its own label-free objective, which supplies no such signal and degrades the pretrained features in this configuration.

One column improves: PIP-Net's keypoint error on CUB, from 9.91 to 9.07. Kp NME regresses keypoints from part centroids, so it rewards parts that lie on the bird even when those parts are less pure, and it is the one column in the table that does not read a part as a region. Further arms are left out of the table because they move more than one thing at a time. Using PIP-Net's official protocol, which unfreezes the last two blocks after a frozen warm-up and scores a bare softmax over the backbone's channels, yields lower part scores on PartImageNet: foreground purity 20.8 and mIoU 12.5 at stage 1 on DINOv2-B/14, against 50.7 and 21.8 frozen, and 15.8 and 7.9 on the ConvNeXt and 27.0 and 17.3 on the ResNet from its own paper. Of the fine-tuned ProtoQuant arms we ran with three seeds, on their DeiT-S, accuracy varies by at most 0.6, which is why the table carries one seed per cell.

Thus, the frozen rows of the main tables are not a weakened form of either method. They are the stronger arm on the columns the paper measures, and they keep the comparison to one variable, since a fine-tuned arm sees the target dataset's labels through its backbone as well as through its classifier. The fine-tuned runs are usable as a control because they reproduce their authors' reported accuracy: ProtoQuant's fine-tuned DeiT-S reaches a top-1 of 87.0 on bounding-box-cropped CUB, against the 86.9 of their Table 3.

F.6 Retrieval results

Table 19 is Table 4 with CUB and the part → image setting added, plus two rows for: PCA (per-image and class-wise), and DisParQ described by something other than its concept identities. The attribute head quantizes a *residual*, the normalized region feature minus its normalized prototype, and each concept owns a private codebook, so an attribute vector carries no concept identity by construction, and its codes are only comparable between occurrences of the same concept. The three descriptor rows show what follows: the codes alone retrieve a same-part neighbor about as often as the concept ids do (R@1 78.9 against 79.7 on PartImageNet) but rank the rest of the list far worse (mAP 16.3 against 30.5); concatenating the two changes nothing; and using the attribute to order candidates that already share a concept raises mAP to 57.3 on PartImageNet and from 9.3 to 28.1 on CUB. Appearance is orthogonal to part identity, and only useful once identity has selected the candidates. No baseline has an attribute, so these rows are ours alone and unranked.

Table 19 also holds the rows that Table 4 leaves out. The category-only oracle and uniform chance are two reference rankers that use no concepts (Appendix D.6). The oracle uses ground-truth classes for accuracy and image-to-image retrieval, solving those columns by construction. For part retrieval,

Table 18: **A frozen against a fine-tuned backbone**, for the two baselines whose own protocols fine-tune it (DINOv2-B/14; $\times 100$ except Kp NME; *stage 2* rows are gray, as they use class labels). The main paper reports the frozen arms, so that every method reads the same features; these rows reproduce them. Fine-tuning raises classification accuracy on every dataset and lowers part quality throughout, with one exception: PIP-Net's keypoint error on CUB, which improves. PIP-Net's fine-tuned arms exist only at the native 16×16 grid, so its pair is the second and third rows of its block and the first row is the AnyUp arm of Tables 2 and 3. CUB carries Pur_{fg} and Kp NME rather than PIP purity, whose window rule changed after the fine-tuned arms were scored. —: not measured.

			PartImageNet					CUB			Cars	Dogs	Flowers
Method	Backbone	Stage	LP↑	kNN↑	Pur_{fg}↑	mIoU↑	p2p R@1↑	LP↑	Pur_{fg}↑	Kp NME↓	LP↑	LP↑	LP↑
PIP-Net	frozen, AnyUp	1	76.9	71.5	68.5	29.0	66.6	77.8	39.7	9.87	82.3	77.8	99.1
PIP-Net	frozen, AnyUp	2	87.5	86.8	59.3	24.9	65.4	88.9	42.0	10.16	91.4	87.1	99.6
PIP-Net	frozen	1	77.8	70.6	50.7	21.8	61.3	79.0	38.8	9.91	85.3	76.4	99.0
PIP-Net	frozen	2	87.2	87.8	37.1	20.5	51.5	88.8	39.4	10.23	91.5	86.8	99.6
PIP-Net	fine-tuned	1	57.6	31.8	26.4	17.2	31.7	67.6	33.5	9.07	70.8	51.8	91.4
PIP-Net	fine-tuned	2	88.2	85.6	22.1	15.3	29.1	90.0	37.2	8.42	93.2	86.0	99.5
ProtoQuant	frozen	1	85.2	83.9	82.1	33.7	79.2	80.1	44.0	11.59	87.0	82.2	99.7
ProtoQuant	frozen	2	87.8	89.5	71.7	26.5	62.6	84.9	41.0	12.16	91.3	86.0	99.6
ProtoQuant	fine-tuned	1	89.1	88.7	61.2	9.3	30.7	89.1	16.3	14.47	93.8	87.2	99.5
ProtoQuant	fine-tuned	2	88.1	89.0	62.8	13.3	35.2	89.4	21.2	14.85	94.1	86.5	99.5

it uses only part-label co-occurrence groups and reaches an R@1 of 43.9, against 79.7 for DisParQ. The components of per-image PCA (Appendix D.4) differ from image to image, so part-to-part retrieval with them is close to chance (R@1 5.3, against 4.6). Class-wise PCA, which selects its components with the class label and is therefore a supervised reference, shows why R@1 must be read beside mAP: ranking by category, it reaches an R@1 of 52.4 on its own query set at an mAP of only 8.9.

PDiscoFormer is scored at its native feature resolution, whose 16×16 grid gives a query set of 3,111 regions and an oracle R@1 of 46.2, against 3,524 regions and 43.9 on the 32×32 grid (Appendix D.6). Resampling its part maps to our 32×32 grid, as we do for its CUB models, puts it on our query set and changes its scores by at most 3.5: part-to-part R@1 is 45.8, 63.5 and 71.8 for K=16, 25 and 50 (natively 49.3, 63.6 and 71.8), and mAP is 32.1, 40.8 and 43.9 (natively 34.2, 41.9 and 45.9). On either grid, PDiscoFormer trails DisParQ on R@1 (79.7) and exceeds it on mAP (30.5).

On CUB, DisParQ leads part-to-part R@1 (36.8) and R@5 (73.7), against an R@1 of 22.7 for the oracle and 7.1 for chance, and trails the TopK SAE by 0.2 on mAP. Part-to-image retrieval is at chance on CUB, since nearly every image contains nearly every keypoint, so it is omitted (∅). Appendix D.6 defines the regions, the oracle and the query sets.

MSAE is trained under two loss weightings. The MSAE rows of Table 19 use the reverse-weighted dictionary, the configuration selected on PartImageNet (Table 2), which is the only one scored on this benchmark. Table 3 and Table 16 instead report the uniform-weighted dictionary selected jointly across the four fine-grained datasets, whose CUB linear probe and k-NN read 73.5 and 42.2 against the 72.5 and 41.7 reported there.

Table 19: **Retrieval on PartImageNet and CUB** (×100). Among self-supervised methods, DisParQ has the best part-to-part R@1 on both datasets. Conventions follow Table 2. A double dagger marks a different grid and oracle (Section F.6).

		Accuracy		Image → image			Part → image			Part → part		
Method	K/N	LP↑	kNN↑	R@1↑	R@5↑	mAP↑	R@1↑	R@5↑	mAP↑	R@1↑	R@5↑	mAP↑
PartImageNet												
Supervised (reference)												
PDiscoFormer‡	16/16	89.6	88.5	78.8	92.9	70.2	38.8	63.5	36.3	49.3	78.3	34.2
PDiscoFormer‡	25/25	89.5	87.5	78.4	93.4	69.1	52.0	74.6	42.5	63.6	86.2	41.9
PDiscoFormer‡	50/50	89.8	88.1	78.4	93.4	69.3	61.9	80.4	51.4	71.8	89.0	45.9
PCA, class-wise‡	16/2528	∅	∅	∅	∅	∅	83.0	98.4	20.1	52.4	88.1	8.9
PIP-Net$_{\text{stage 2}}$	16/768	87.5	86.8	77.6	91.9	64.2	77.5	93.8	45.9	65.4	85.2	21.1
ProtoQuant$_{\text{stage 2}}$	16/2048	87.8	89.5	82.3	93.8	76.6	86.2	98.5	33.0	62.6	89.6	15.5
Frozen backbone, no concept bottleneck (reference)												
CLS token	—	89.3	88.9	81.0	92.9	75.1	—	—	—	—	—	—
Patch tokens, mean-pooled	—	85.7	79.3	57.5	82.2	44.7	—	—	—	—	—	—
Ground-truth controls												
category-only oracle	—	100.0	100.0	100.0	100.0	100.0	82.2	91.5	81.5	43.9	89.0	44.7
uniform chance	—	—	—	0.6	2.9	0.6	13.7	47.7	14.1	4.6	20.1	4.9
Self-supervised												
PCA, per-image	16/16	∅	∅	∅	∅	∅	11.0	39.0	14.5	5.3	23.6	5.2
CFM‡	16/8192	72.6	69.2	47.0	75.4	34.6	63.4	74.3	40.8	31.6	68.8	16.0
TopK SAE	16/1024	78.1	73.0	52.6	79.2	42.7	84.7	97.4	34.9	56.2	86.5	17.2
TopK SAE	16/2048	80.5	73.5	53.8	77.5	43.3	85.6	98.0	35.7	58.3	86.9	18.1
PatchSAE	16/1024	82.9	76.5	56.8	80.8	44.6	83.3	97.4	29.3	41.9	83.6	11.8
PatchSAE	16/49152	84.1	77.2	56.8	81.1	44.4	85.0	95.5	20.6	48.6	82.9	8.0
MSAE	16/1024	85.2	79.9	58.9	81.7	46.1	82.9	95.8	30.9	41.4	82.2	12.5
MSAE	16/8192	85.6	79.5	58.3	82.3	45.4	79.8	93.4	30.3	41.2	79.9	12.6
PIP-Net$_{\text{stage 1}}$	16/768	76.9	71.5	49.1	78.6	36.3	77.9	94.6	47.6	66.6	87.7	27.0
ProtoQuant$_{\text{stage 1}}$	16/2048	85.2	83.9	72.5	90.4	61.8	**89.8**	**98.8**	40.3	79.2	**92.8**	22.7
DisParQ (ours)	16/1024	**89.2**$_{+3.6}$	**88.7**$_{+4.8}$	**80.6**$_{+8.1}$	**92.9**$_{+2.5}$	**76.1**$_{+14.3}$	86.9$_{-2.9}$	97.8$_{-1.0}$	**48.5**$_{+0.9}$	**79.7**$_{+0.5}$	92.4$_{-0.4}$	**30.5**$_{+3.5}$
DisParQ, the same model described differently (ours only, unranked)												
attribute codes only	16/1024	89.2	88.7	80.6	92.9	76.1	89.8	98.1	32.1	78.9	93.8	16.3
concept id + attribute	16/1024	89.2	88.7	80.6	92.9	76.1	86.9	97.8	48.5	79.9	92.4	30.5
concept id + same-concept appearance	16/1024	89.2	88.7	80.6	92.9	76.1	74.8	91.9	37.7	82.4	93.2	57.3
CUB-200-2011												
Supervised (reference)												
PDiscoFormer	16/16	86.8	74.8	69.8	89.5	38.2	∅	∅	∅	45.8	84.6	38.6
PCA, class-wise	16/3200	∅	∅	∅	∅	∅	∅	∅	∅	9.0	34.0	7.1
PIP-Net$_{\text{stage 2}}$	16/768	88.9	88.9	85.6	94.9	75.0	∅	∅	∅	32.3	69.8	8.5
ProtoQuant$_{\text{stage 2}}$	16/4000	84.9	83.0	76.8	93.5	52.5	∅	∅	∅	30.1	65.9	8.4
Frozen backbone, no concept bottleneck (reference)												
CLS token	—	90.4	86.7	85.1	95.4	65.9	∅	∅	∅	—	—	—
Ground-truth controls												
category-only oracle	—	100.0	100.0	100.0	100.0	100.0	∅	∅	∅	22.7	60.2	22.9
uniform chance	—	—	—	0.5	2.4	0.5	∅	∅	∅	7.1	30.2	7.1
Self-supervised												
PCA, per-image	16/16	∅	∅	∅	∅	∅	∅	∅	∅	7.6	32.6	7.2
CFM	16/8192	54.8	42.8	35.9	63.3	11.9	∅	∅	∅	8.9	35.1	7.4
TopK SAE	16/1024	36.6	24.5	22.0	44.0	10.0	∅	∅	∅	15.5	42.9	**9.5**
TopK SAE	16/2048	36.9	24.3	21.9	43.4	10.0	∅	∅	∅	15.9	47.2	9.4
PatchSAE	16/1024	56.6	33.5	30.1	55.1	12.6	∅	∅	∅	9.3	34.3	7.2
PatchSAE	16/49152	62.0	33.4	30.2	54.1	11.4	∅	∅	∅	10.9	37.2	7.2
MSAE$_{\text{RW}}$	16/1024	71.0	41.3	35.9	61.9	14.4	∅	∅	∅	10.5	37.1	7.7
MSAE$_{\text{RW}}$	16/6144	72.5	41.7	36.3	62.2	14.4	∅	∅	∅	9.5	34.1	7.5
PIP-Net$_{\text{stage 1}}$	16/768	77.8	68.4	60.7	86.4	35.2	∅	∅	∅	29.8	64.3	9.4
ProtoQuant$_{\text{stage 1}}$	16/4000	80.1	67.7	59.4	84.7	31.5	∅	∅	∅	30.9	67.5	8.2
DisParQ (ours)	16/1024	**89.4**$_{+9.3}$	**85.5**$_{+17.1}$	**84.5**$_{+23.8}$	**95.1**$_{+8.7}$	**67.8**$_{+32.6}$	∅	∅	∅	**36.8**$_{+5.9}$	**73.7**$_{+6.2}$	9.3$_{-0.2}$
DisParQ, the same model described differently (ours only, unranked)												
attribute codes only	16/1024	89.4	85.5	84.5	95.1	67.8	∅	∅	∅	37.6	71.5	7.9
concept id + attribute	16/1024	89.4	85.5	84.5	95.1	67.8	∅	∅	∅	38.6	74.0	9.3
concept id + same-concept appearance	16/1024	89.4	85.5	84.5	95.1	67.8	∅	∅	∅	39.6	76.5	28.1



Figure 24: Example questions shown in the survey briefing for Q2: part retrieval.

F.7 What the quantized attributes keep

For a selected concept i_k in an image, the attribute encoder sees the residual $r_{i_k} = \nu(\bar{z}_{i_k}) - \nu(w_{i_k})$ between the normalized region mean $\bar{z}_{i_k}$ and the normalized concept prototype w_{i_k} (Section 3.3), so $\nu(w_{i_k}) + r_{i_k}$ is the normalized region mean. Stage 2 projects the residual to $y_{i_k} = W_{\mathrm{PQ}}\, r_{i_k}$ and encodes it as G=16 indices $j_{i_k,1:G}$ into the concept-specific codebooks $\mathcal{B}_{i_k}$, with M=64 codewords per group (Appendix C.3). The attribute q_{i_k} that the decoder receives is learned for decoding and is not trained to reconstruct y_{i_k}.

To see what the code keeps, we therefore decode it back into feature space: each index $j_{i_k,g}$ is replaced by the mean of group g of y_{i_k} over the training occurrences of concept i_k that received it, and a ridge regression fitted on the training split maps the result back to feature space, giving $r^{\mathrm{dec}}_{i_k}$. The images of Figures 44–48 (Appendix G.6) are held out from both.

The quantized attribute also keeps which class an occurrence of a concept comes from: in Appendix G.7, the occurrences of one concept in the evaluation images group by the class of their image (Figures 49–52).

F.8 User study details and results

Setup. We recruited 120 native English speakers through the `clickworker.com` platform, balanced in terms of gender and age (18-64). The study was fully anonymous, no personal data was collected, and it followed our institution's guidelines for research involving human participants. The median completion time was 8 minutes 42 seconds. Before starting, participants read a short briefing describing both question types, with an annotated example for each (Fig. 25 and Fig. 24). Each participant then answered 42 questions: 21 part-retrieval questions (Q2) and 21 part-consistency questions (Q1). For both question types, the 21 questions contained 7 prototypes from each of DisParQ, PIP-Net, and CFM, drawn from all three datasets, presented in random order and with the method identity hidden. We used three survey versions with different prototype sets, each completed by 40 participants. For Q1, the per-method CUB/Cars/PartImageNet counts were 3/3/1, 3/1/3 and 1/3/3 across versions. Prototypes were selected by sampling random test images and taking one of the four most important prototypes for the image's prediction. Here, four specifies the candidate set for stimulus selection, not the model's concept budget. The same images were used for all three methods, and all prototypes were visualized in the same way, by their top activating image patches. To ensure data quality, we screened responses for inconsistent answering and excluded 4% of participants, recruiting replacements until 120 valid responses were collected.

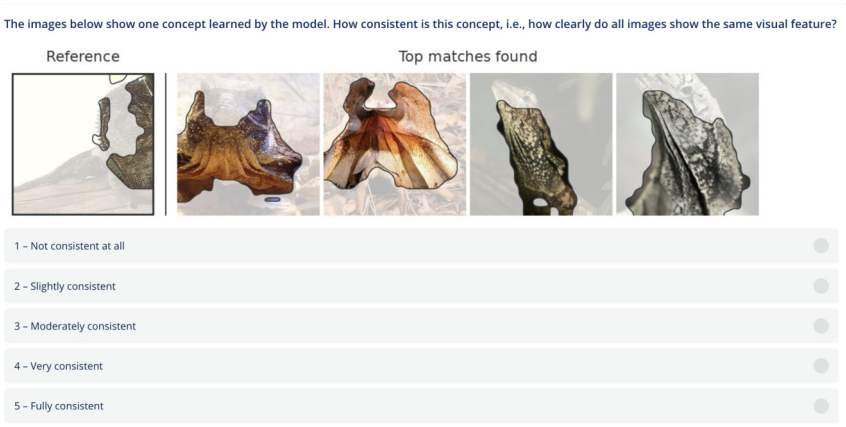


Figure 25: Example questions shown in the survey briefing for Q1: part consistency.

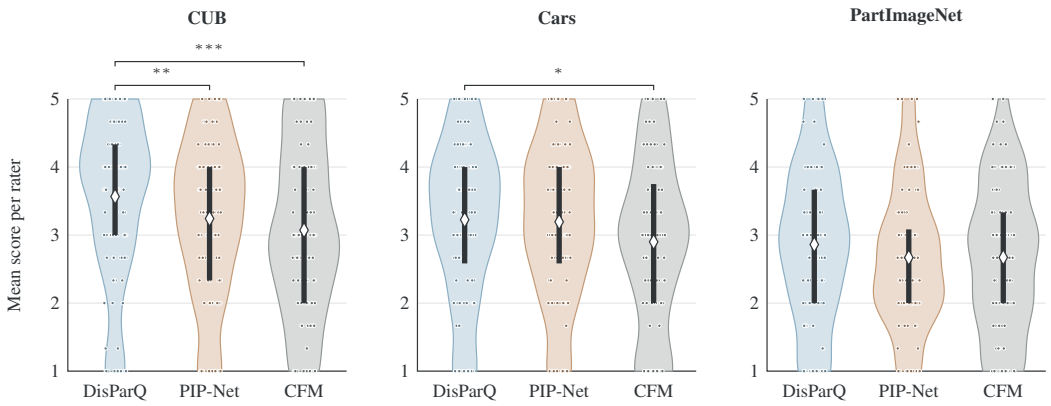


Figure 26: **Part consistency ratings per dataset.** Each point is one participant's mean rating (1: not consistent at all, 5: fully consistent) over the prototypes of a given method shown from that dataset (n=120). Violins show the distribution, diamonds the mean, and black bars the interquartile range. Brackets mark comparisons in which DisParQ is rated significantly higher than a baseline (two-sided paired Wilcoxon signed-rank test, Holm-corrected): $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$; unmarked comparisons are not significant.

Q1: Part consistency. Participants were shown the visualization of a single prototype and asked: *"The images below show one concept learned by the model. How consistent is this concept, i.e., how clearly do all images show the same visual feature?"* They answered on a 5-point scale (1: not consistent at all, 2: slightly, 3: moderately, 4: very, 5: fully consistent) and were told that there are no right or wrong answers. For each participant, we average ratings per method, overall and per dataset. We compare DisParQ with each baseline using two-sided paired Wilcoxon signed-rank tests, with Holm correction across all eight comparisons (two baselines, overall and three datasets).

Participants rated DisParQ prototypes as the most consistent overall, with a significant advantage over both PIP-Net ($p < 0.001$) and CFM ($p < 0.001$). Per dataset (Fig. 26), the gap is largest on CUB, where DisParQ is significantly better than both baselines. On Cars it significantly outperforms CFM, while on PartImageNet all three methods are rated similarly low, suggesting that consistent parts are harder to judge on this more diverse dataset.

Table 20: Part retrieval accuracy (%). Participants chose which of four model-identified regions matches the visualized concept, or "None of the above" (chance: 25% or lower); 120 participants. All methods are significantly above chance ($p < 10^{-4}$, one-sided Wilcoxon). Stars mark baselines that DisParQ significantly outperforms (paired Wilcoxon signed-rank test on per-participant accuracy, Holm-corrected): $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Method	All	CUB	Cars	PartImageNet
Random	25.00	25.00	25.00	25.00
CFM	51.50***	50.00**	56.67***	48.57**
PIP-Net	60.12***	63.57	63.57*	53.21***
DisParQ	**69.05**	**65.00**	**72.86**	**69.29**

Q2: Part retrieval. Inspired by HIVE (Kim et al., 2022), this task measures how well humans can predict the model's reasoning from its explanations. Participants were shown the visualization of a prototype on the left and a new image with four lettered regions on the right, all identified by the model, and asked: *"The images on the left show one concept learned by the model. Which region of the image on the right does this concept correspond to?"* Participants could select one of the four regions or "None of the above". The correct answer is the region in which the model localizes the prototype. For each participant, we compute the accuracy per method, overall and per dataset, and compare DisParQ with each baseline using a two-sided paired Wilcoxon signed-rank test with Holm correction. We test each method against a chance level of 25% with a one-sided Wilcoxon signed-rank test; this is conservative, as five answer options were available.

All methods lead to above-chance accuracy, but participants were most accurate with DisParQ prototypes on every dataset, reaching 69% overall compared to 60% for PIP-Net and 52% for CFM (Tab. 20). The advantage is significant against both baselines overall and on every dataset except CUB, where DisParQ and PIP-Net perform comparably. It is most pronounced on PartImageNet, where DisParQ outperforms PIP-Net by 16 percentage points. A logistic regression with standard errors clustered by participant confirms the result: with DisParQ, the odds of a correct answer are 1.5 and 2.1 times higher than with PIP-Net and CFM, respectively ($p < 10^{-4}$).

G Qualitative Results

This section first describes our interactive concept visualizer to browse concepts from DisParQ (Sections G.1). It then compares the concept maps of DisParQ with those of the baselines (Sections G.2–G.4), and shows what the description of DisParQ contains: the concepts behind a prediction (Section G.5) and their attributes (Sections G.6 and G.7).

G.1 Interactive Concept Visualizer

We provide an interactive concept visualizer at `https://anonymous.4open.science/w/DisParQ-3E73` that allows browsing concepts from DisParQ. The visualizer includes concept banks for five datasets (PartImageNet, CUB, Dogs, Cars, and Flowers), along with an image gallery that shows what concepts activate for an image and where. We describe the two galleries below:

Image gallery. This consists of three images per class (Figure 27) for each dataset. The gallery view (Figure 27, top) shows images with a concept overlay, and can be filtered by class or concept ID. Each image can be selected to open the per-image view, that shows the top-16 concepts, where they activate, and what they are, characterized by other exemplars from the dataset that contain the same concept (Figure 27, bottom). The visualizer also contains an option to view the per-image view of a random image from the selected dataset.

Concept gallery. This shows the full set of concepts that activate on at least one image in the dataset (Figure 28). The gallery view (Figure 28, top) shows all concepts from the dataset that are found in at least one image. Each concept card shows exemplars of images and where the concept activates on it, along with the number of images the concept activates on the primary classes they belong to. The per-concept view (Figure 28, bottom) shows the images and regions within them that correspond to

the strongest, typical, and weakest activations for the concept in the dataset. It also shows the top classes the concept appears in, and other concepts that most frequently co-occur with the concept.

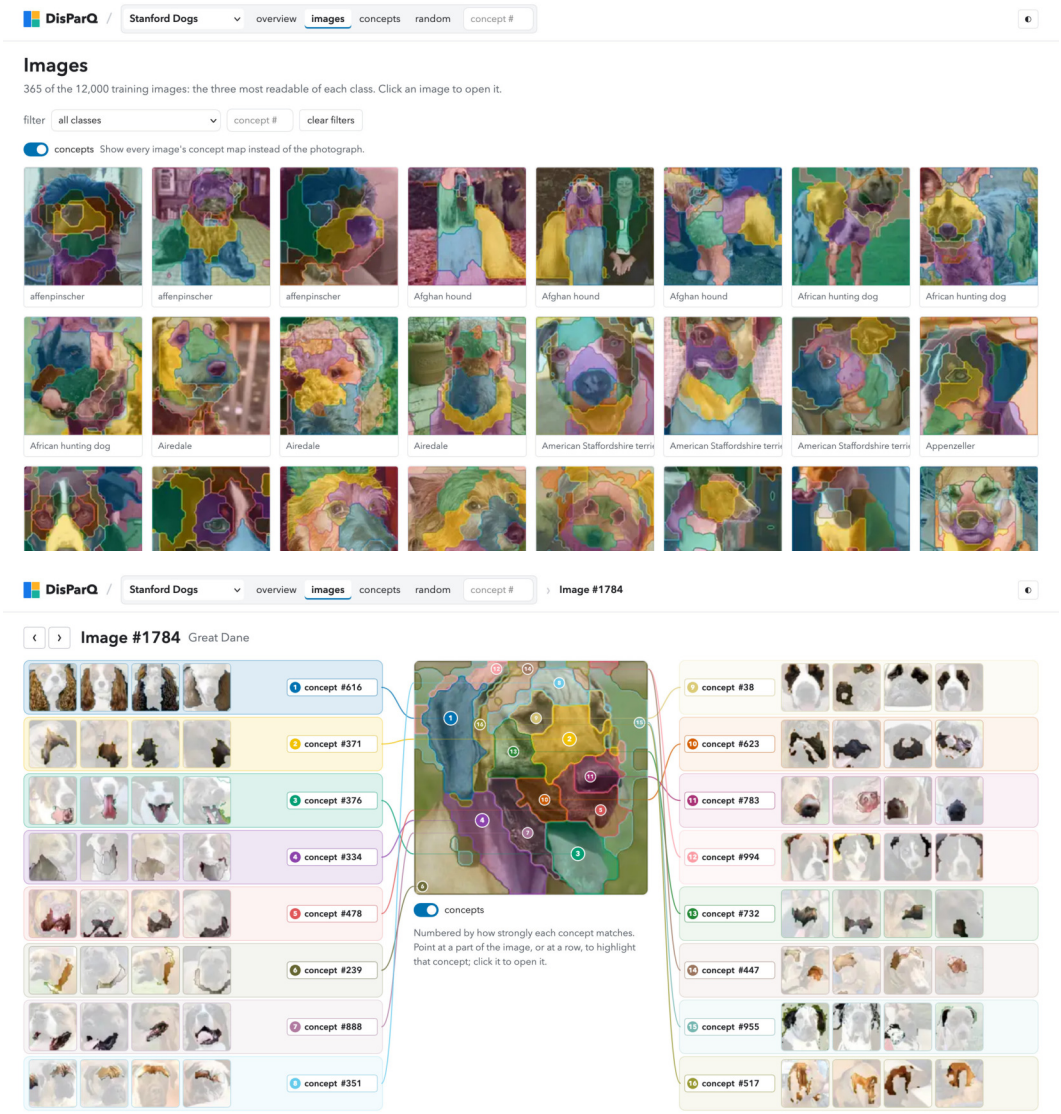


Figure 27: **Visualizer: the image gallery and one image.** *Top:* The training images of one dataset, filtered by class or by concept, with an overlay of each image's concept map. *Bottom:* One image, opened from the gallery: its concept map in the middle, its concepts numbered by match strength. Each concept's row shows the training images where it matches most strongly, and an arrow joins the row to the concept's region.

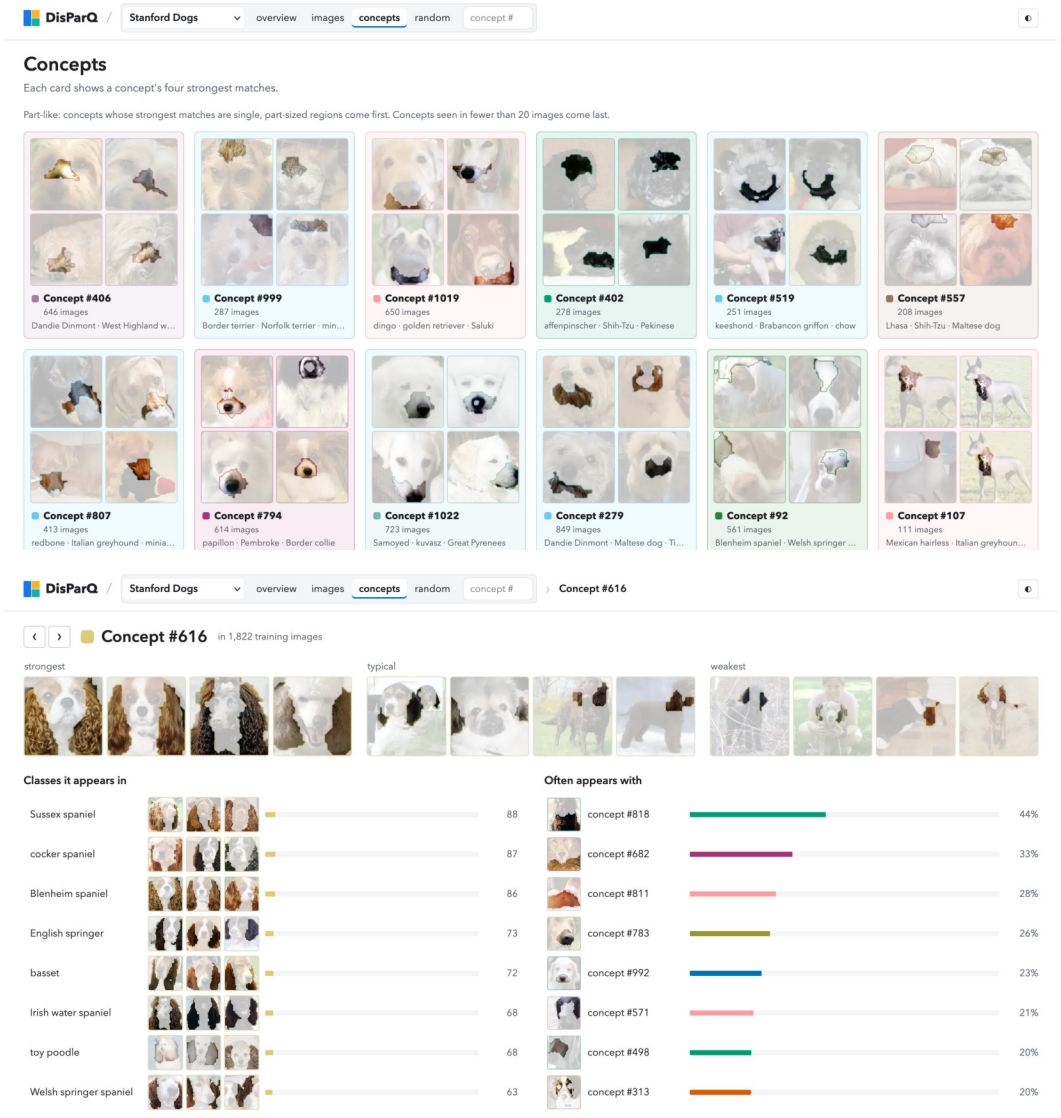


Figure 28: **Visualizer: the concept gallery and one concept.** *Top:* Every concept used on a dataset, each shown by its four strongest matches. *Bottom:* One concept, opened from the gallery: its strongest, typical and weakest matches, the classes it appears in with examples, and the concepts it most often appears with.

G.2 PARTIMAGENET

This subsection extends Figure 6. All figures use the PartImageNet validation split and the models of Table 2; every method sees the same center-cropped 224×224 input, and the ground-truth mask is cropped identically. Colors follow the rule of Figure 6. Figure 29 shows five further images and, for two parts, the one concept of each method that is most consistent with that part over the validation split.

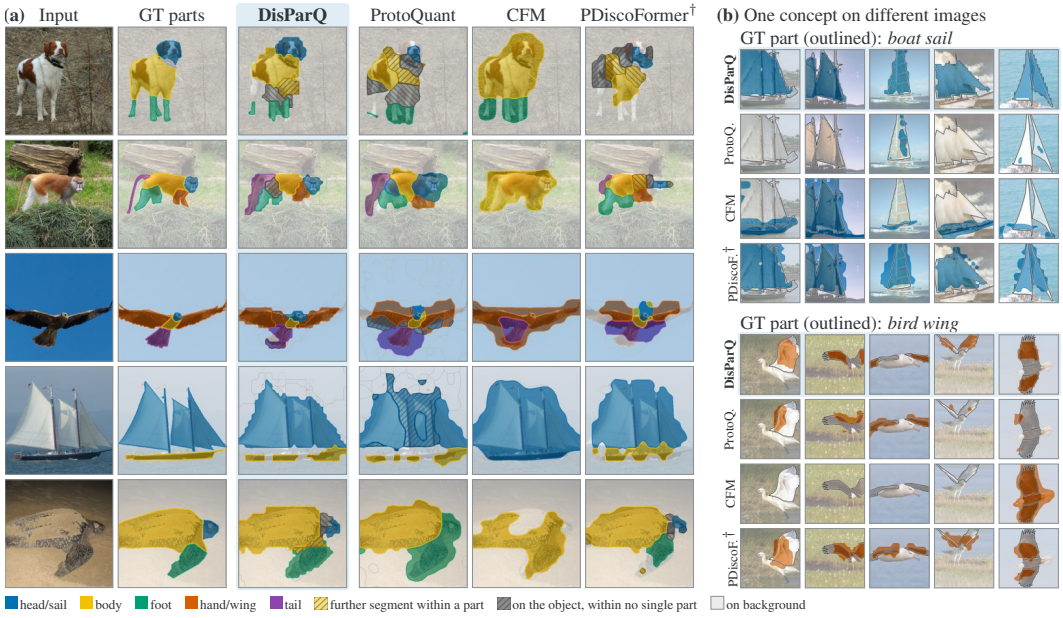


Figure 29: **Part maps, and one concept across images**, on PartImageNet. *(a)* Each method's hard concept map is cut into segments, the segments are matched one-to-one to the ground-truth parts of that image, and a matched segment takes its part's color, so agreement with the *GT parts* column reads as matching colors. Hatched segments lie on the object but match no part (over-segmentation); they take a part's color when at least 60% of the segment lies within that part, and are gray otherwise. Pale outlined segments lie on the background. † trained with class labels. *(b)* For a ground-truth part (outlined), the one concept of each method that is most consistent with that part over the validation split, shown on the same five images. The concept of DisParQ marks the part in every image; a baseline's best concept is absent from most of them, or covers the whole object.

Figures 31 and 32 compare DisParQ with five baselines: ProtoQuant (Janusz et al., 2026), MSAE (Zaigrajew et al., 2025), PDiscoFormer (Aniraj et al., 2024), PIP-Net (Nauta et al., 2023) and CFM (Wittenmayer et al., 2026). DisParQ follows the annotated parts and stops at the object boundary: it separates the hull of a sailboat from its sails, and the head and flippers of a turtle from its shell. The baselines merge parts into one concept (MSAE, PIP-Net and CFM cover a turtle with the color of its shell) or spread into the background (MSAE and PIP-Net fill the sea and sky around a sailboat). On random images (Figure 32), DisParQ still recovers small parts such as a snake's head, while MSAE marks the background and CFM covers the object with a single concept. Figure 30 follows one concept of each method across five images: the concept of DisParQ marks the sail, the hull and the bottle mouth in all five images and the tire, the wing and the fish head in most, whereas the best-matching concept of a baseline is missing from several images or covers the whole object.

Where DisParQ fails. Figure 33 shows the images with the largest per-image deficit against the best baseline. They share one cause: close-ups in which a single annotated part (almost always *head*) fills the frame. DisParQ describes such an image with many finer concepts like snout, eyes, ears, fur, of which only one can be matched to the single ground-truth part, while a method with one coarse concept per object matches it trivially. We see this as a limitation of the annotation granularity of human-labeled datasets rather than of the concepts: PartImageNet labels a head as one part, so a finer decomposition that is consistent across images is scored as over-segmentation.



Figure 30: **One concept, different images.** Each row shows, for the outlined ground-truth part, the concept of that method most consistent with it over the validation split. † trained with class labels.

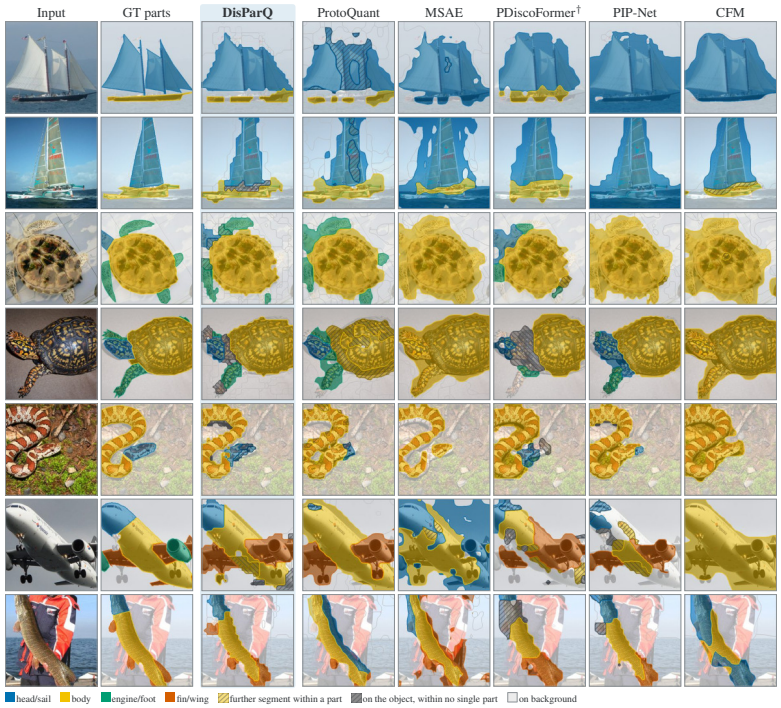


Figure 31: **Further examples** on PartImageNet, all five baselines. Maps and colors as in Figure 6. † trained with class labels.

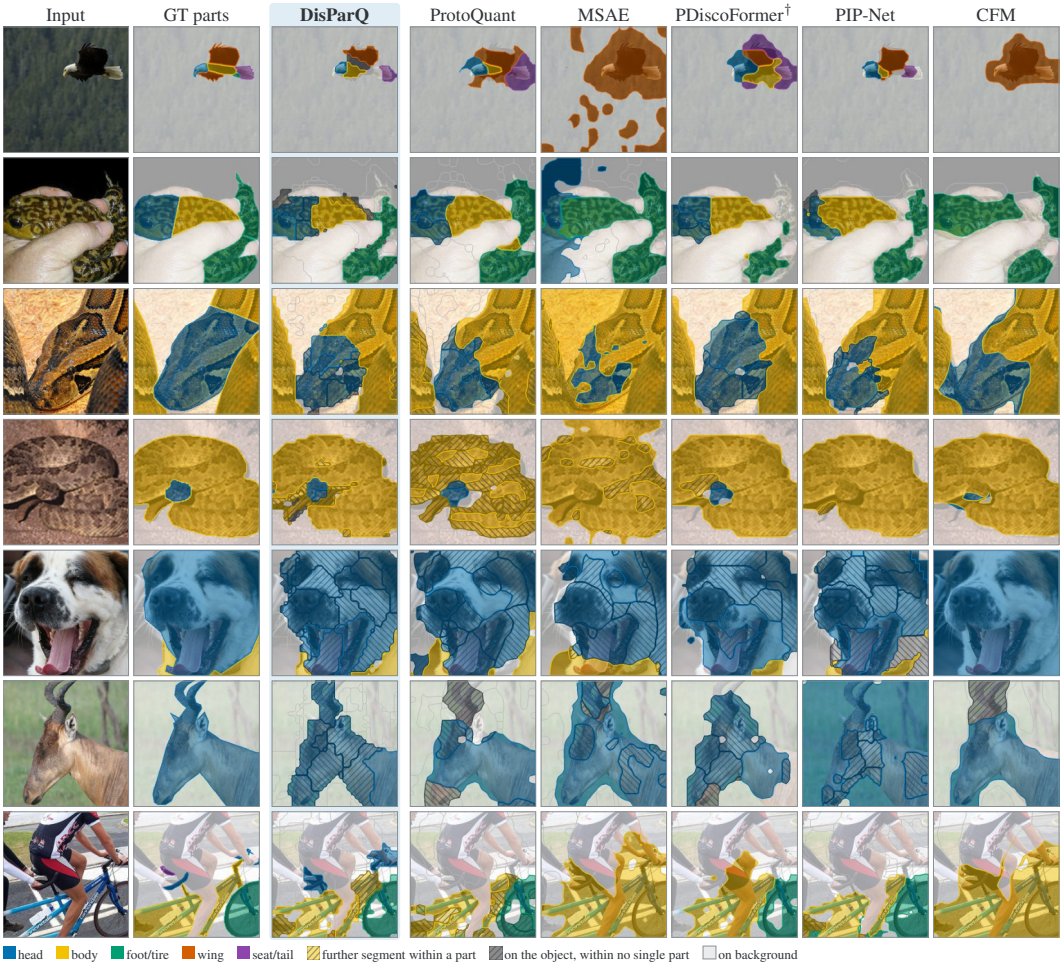


Figure 32: **Random examples** from the PartImageNet validation split. Maps and colors as in Figure 6. † trained with class labels.

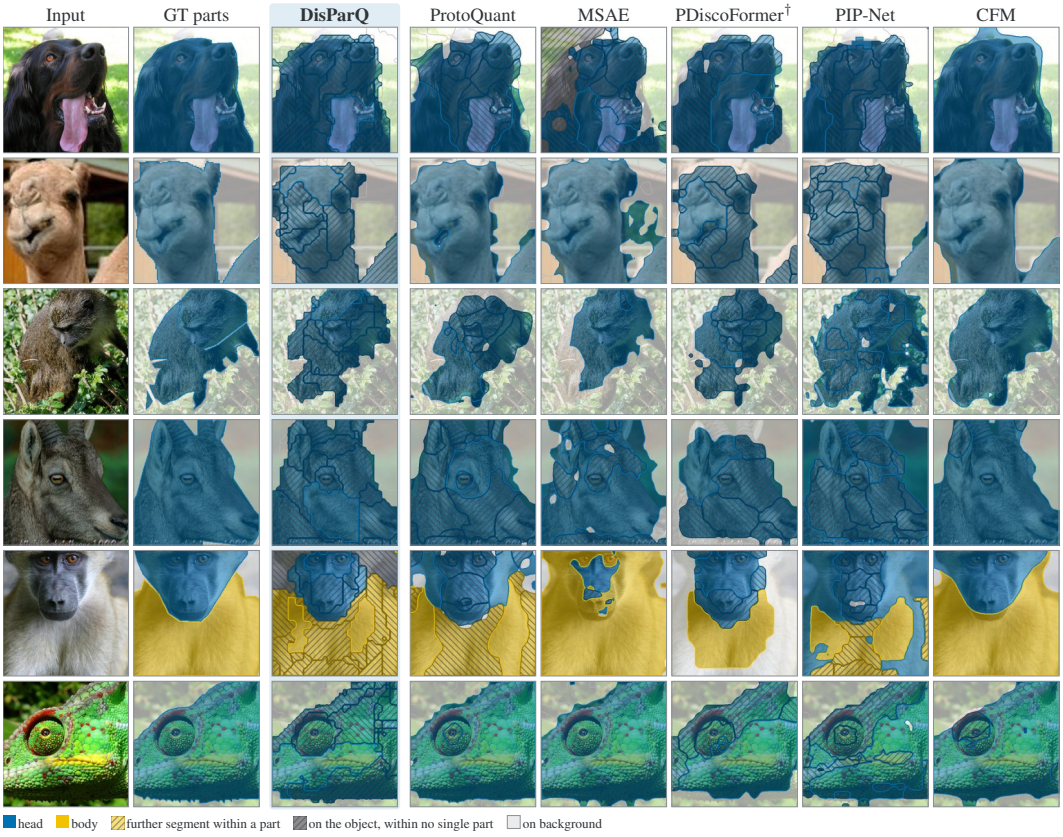


Figure 33: **Failure cases**: images on which a baseline reaches a higher per-image mIoU than DisParQ. Maps and colors as in Figure 6. † trained with class labels.

G.3 FINE-GRAINED DATASETS

This subsection repeats Figure 6 on four fine-grained datasets: CUB (Wah et al., 2011), Stanford Cars (Krause et al., 2013), Stanford Dogs (Khosla et al., 2011) and Oxford Flowers (Nilsback & Zisserman, 2008), with the baselines trained on each dataset (Figures 34–37). All images come from the test split of each dataset. These datasets have no part masks, so a segment is colored by where it sits on the object, by the same rule in every column; the reference column shows the foreground and, on CUB, the annotated keypoints. Panel (b) follows two concepts of DisParQ, each across five images, beside each baseline's best-matching concept. The maps of DisParQ stay on the object and divide it into regions that recur across images, while MSAE also covers the background. The concept of DisParQ marks the same part in all five images, for example the tail or the head of a bird, while the best-matching concept of MSAE covers whole birds. PDiscoFormer is trained with class labels and shown for reference.

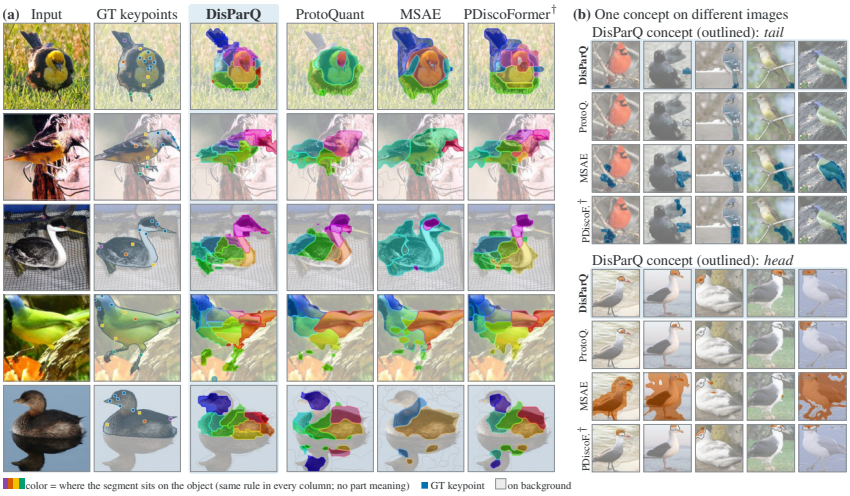


Figure 34: **CUB: DisParQ and baselines.** *(a)* Hard concept maps, colored by where a segment sits on the object (same rule in every column, no part meaning); pale outlined segments lie on the background. *(b)* Two concepts of DisParQ (outlined), each beside the concept of every baseline that agrees best with it. † trained with class labels.

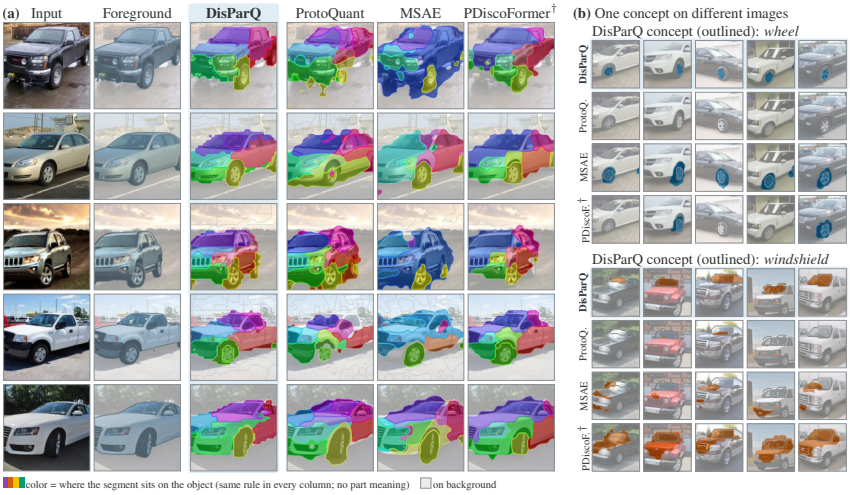


Figure 35: **Stanford Cars: DisParQ and baselines.** *(a)* Hard concept maps, colored by where a segment sits on the object (same rule in every column, no part meaning); pale outlined segments lie on the background. *(b)* Two concepts of DisParQ (outlined), each beside the concept of every baseline that agrees best with it. † trained with class labels.

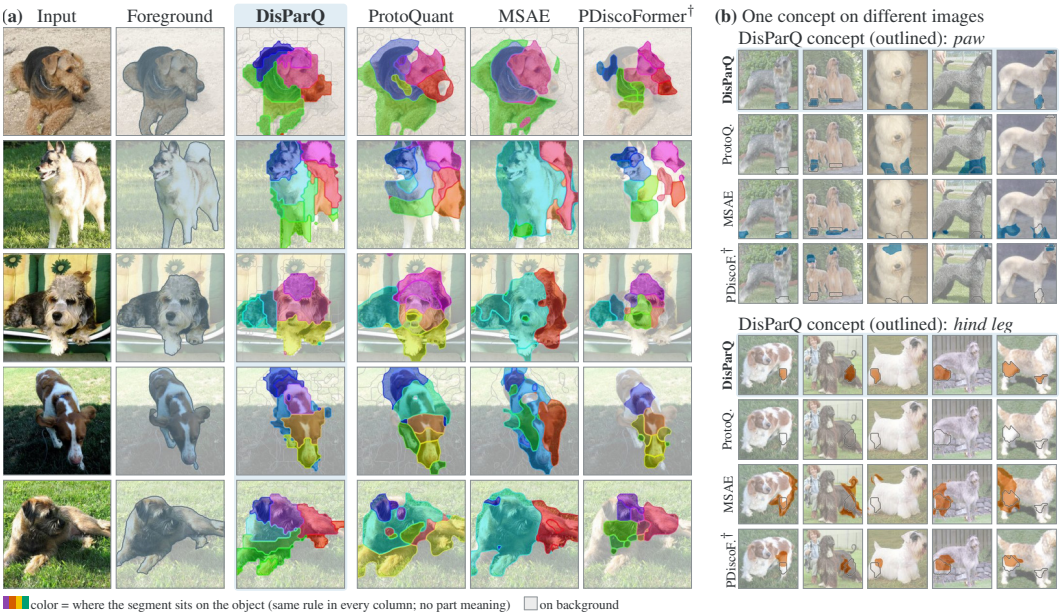


Figure 36: **Stanford Dogs: DisParQ and baselines.** *(a)* Hard concept maps, colored by where a segment sits on the object (same rule in every column, no part meaning); pale outlined segments lie on the background. *(b)* Two concepts of DisParQ (outlined), each beside the concept of every baseline that agrees best with it. † trained with class labels.

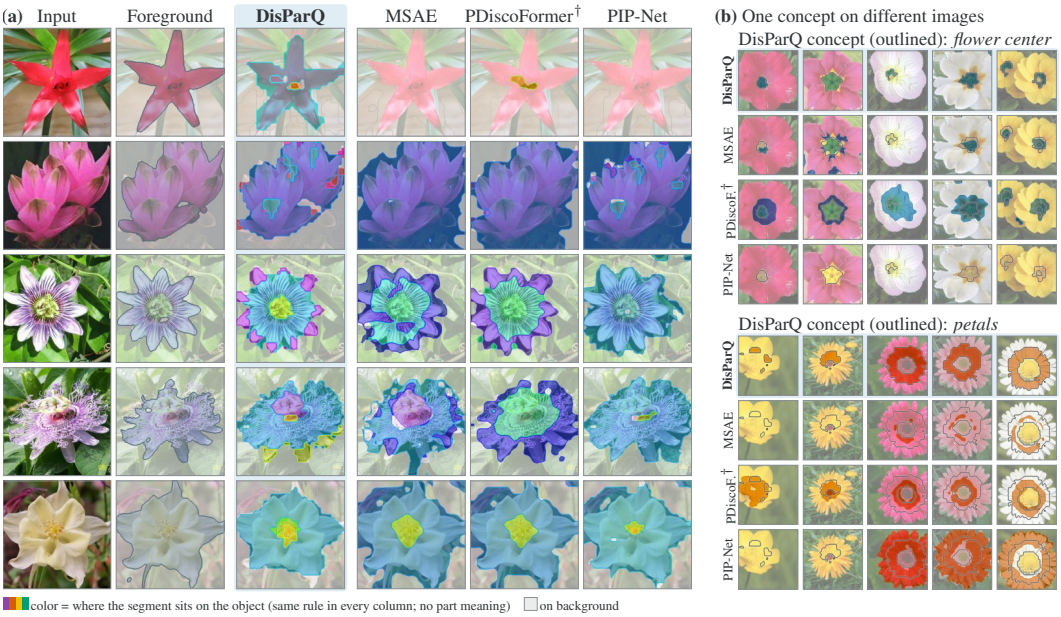


Figure 37: **Oxford Flowers: DisParQ and baselines.** *(a)* Hard concept maps, colored by where a segment sits on the object (same rule in every column, no part meaning); pale outlined segments lie on the background. *(b)* Two concepts of DisParQ (outlined), each beside the concept of every baseline that agrees best with it. † trained with class labels.

G.4 PART-TO-PART RETRIEVAL

This subsection shows part-to-part retrieval, the benchmark of Table 4, defined in Appendix D.6, on all five datasets (Figure 38). For a query region (left), each row gives the four regions of other images that a method ranks highest by the concept histogram of the region. On PartImageNet and CUB, a green border marks a retrieval that carries the query's annotated part and a red border a miss. Queries and retrievals come from held-out images (the PartImageNet validation split and the test splits of the other datasets), and all methods describe the same regions. A region is an annotated part mask on PartImageNet and a keypoint group on CUB; on Cars, Dogs and Flowers, which have no part annotation, it is a concept of DisParQ.

All four retrievals of DisParQ carry the query's part in each of the six scored panels, while at most two of the four retrievals of ProtoQuant or MSAE do, as for the snake head and for the head and leg of a bird. PDiscoFormer, trained with class labels, also retrieves the bird parts. On Cars, Dogs and Flowers, which have no part annotation, DisParQ retrieves windshields, bumpers, grilles, muzzles and petals like the query, while the baselines return regions of mixed appearance.



Figure 38: **Part-to-part retrieval on five datasets.** Each panel is one query: the region on the left (dark border) and, per method (rows, DisParQ first), the four highest-ranked regions of other images. Green or red border: the retrieval does or does not carry the query's annotated part (PartImageNet and CUB); Cars, Dogs and Flowers are unscored and their region names are ours. † trained with class labels.

G.5 Class evidence of DisParQ and interpretability

This subsection extends Figure 1 to more images, for DisParQ on DINOv2-B/14 (Oquab et al., 2024) (Figures 39–43). Predictions use the separate explanation classifier in Appendix D.6. The left panels show the four concepts with the largest positive drops in the predicted-class logit. The right panels show their nearest training occurrences of the same concept, ranked by cosine similarity between attributes. The images on the left come from the PartImageNet validation split and the test splits of the other datasets.

On PartImageNet (Figure 39), the concepts for a Jeep, a warplane or a Komodo dragon retrieve the same part in other images and classes. On CUB (Figure 40), the green head of a mallard retrieves the heads of other mallards, the head and wing of a blue jay account for 81% of its positive logit drops, and the yellow wing bar of a European goldfinch retrieves wings with the same bar. On Stanford Cars (Figure 41), the evidence lies on parts a person would name: the number plate and tail lights of a BMW X5 retrieve number plates and tail lights of other cars, and the headlight and wheel of a Volkswagen Beetle retrieve round headlights and wheel rims. On Stanford Dogs (Figure 42), it lies on the face and the coat, and the face concept of a Blenheim spaniel retrieves the faces of other spaniels. On Oxford Flowers (Figure 43), the corona and stamens of a passion flower account for 79% of its positive logit drops, and the dark disc of a sunflower retrieves other sunflower discs.

Interpretability. Two regularities hold across the 25 examples. Positive logit drops are concentrated in a few concepts. The four largest contributions account for 56 to 100% of their total and the largest alone for 17 to 52%. These normalized occlusion effects measure relative support under the intervention. The concepts are also shared across images, as each concept retrieves the same part in other images, often of other classes, so predictions for different classes are explained with one vocabulary of parts, and a prediction reads as a short list of recognizable parts with their weights. Appendix D.6 (Concept contribution) defines how the contribution is measured.

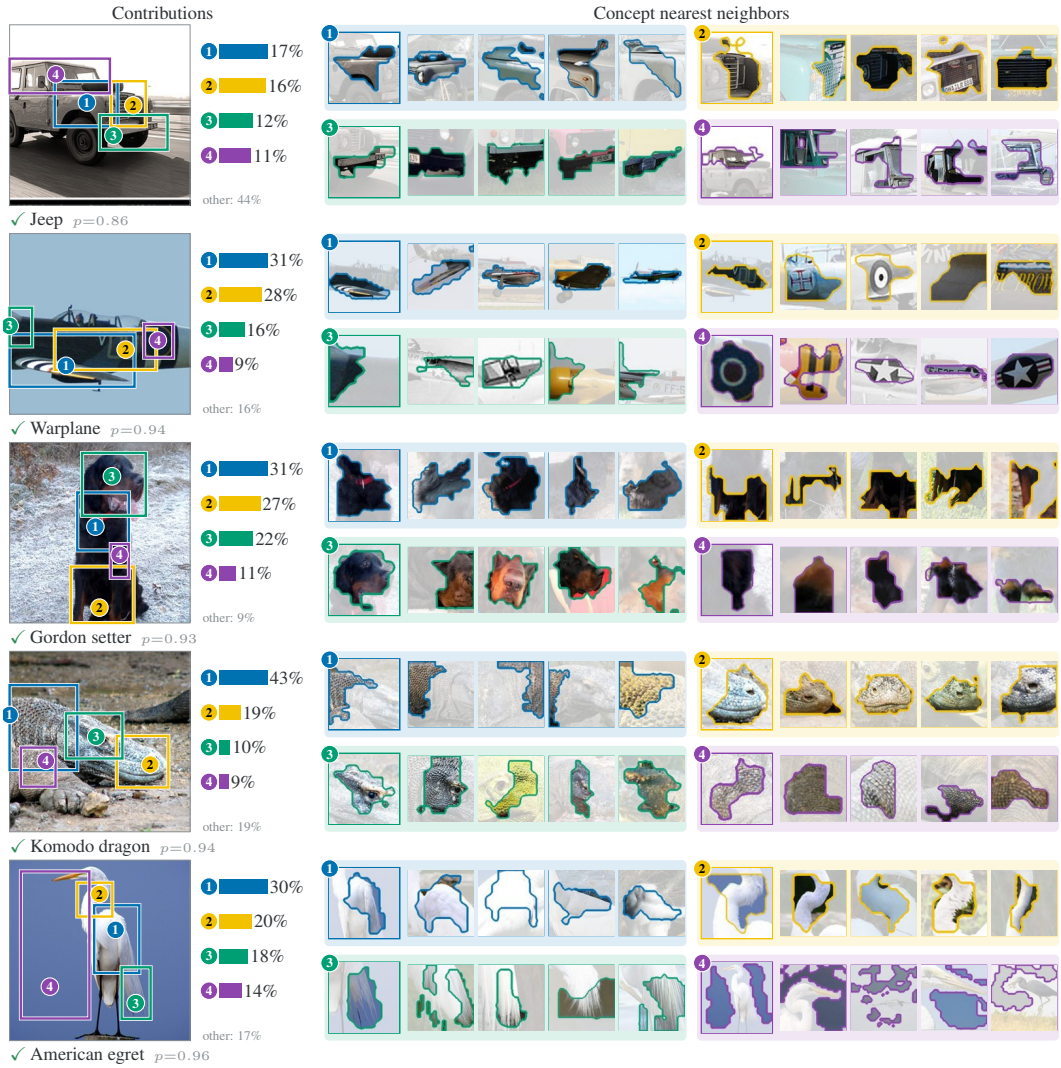


Figure 39: **PartImageNet: evidence concepts and what they retrieve.** *Left:* the four concepts with the largest contribution to the predicted-class logit. *Right:* each one cropped from this image (thick border) beside its nearest training occurrences.

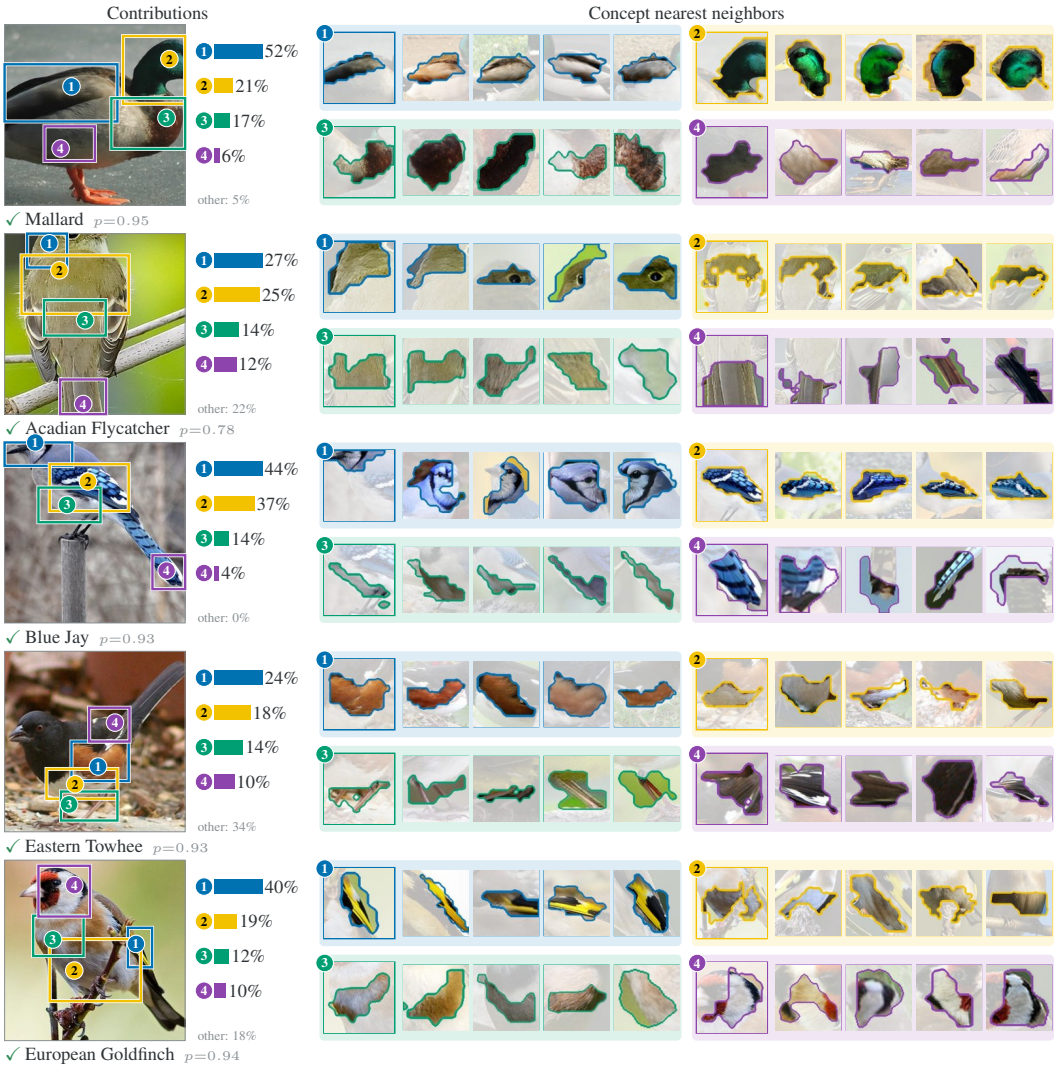


Figure 40: **CUB: evidence concepts and what they retrieve.** *Left:* the four concepts with the largest contribution to the predicted-class logit. *Right:* each one cropped from this image (thick border) beside its nearest training occurrences.



Figure 41: **Stanford Cars: evidence concepts and what they retrieve.** *Left:* the four concepts with the largest contribution to the predicted-class logit. *Right:* each one cropped from this image (thick border) beside its nearest training occurrences.

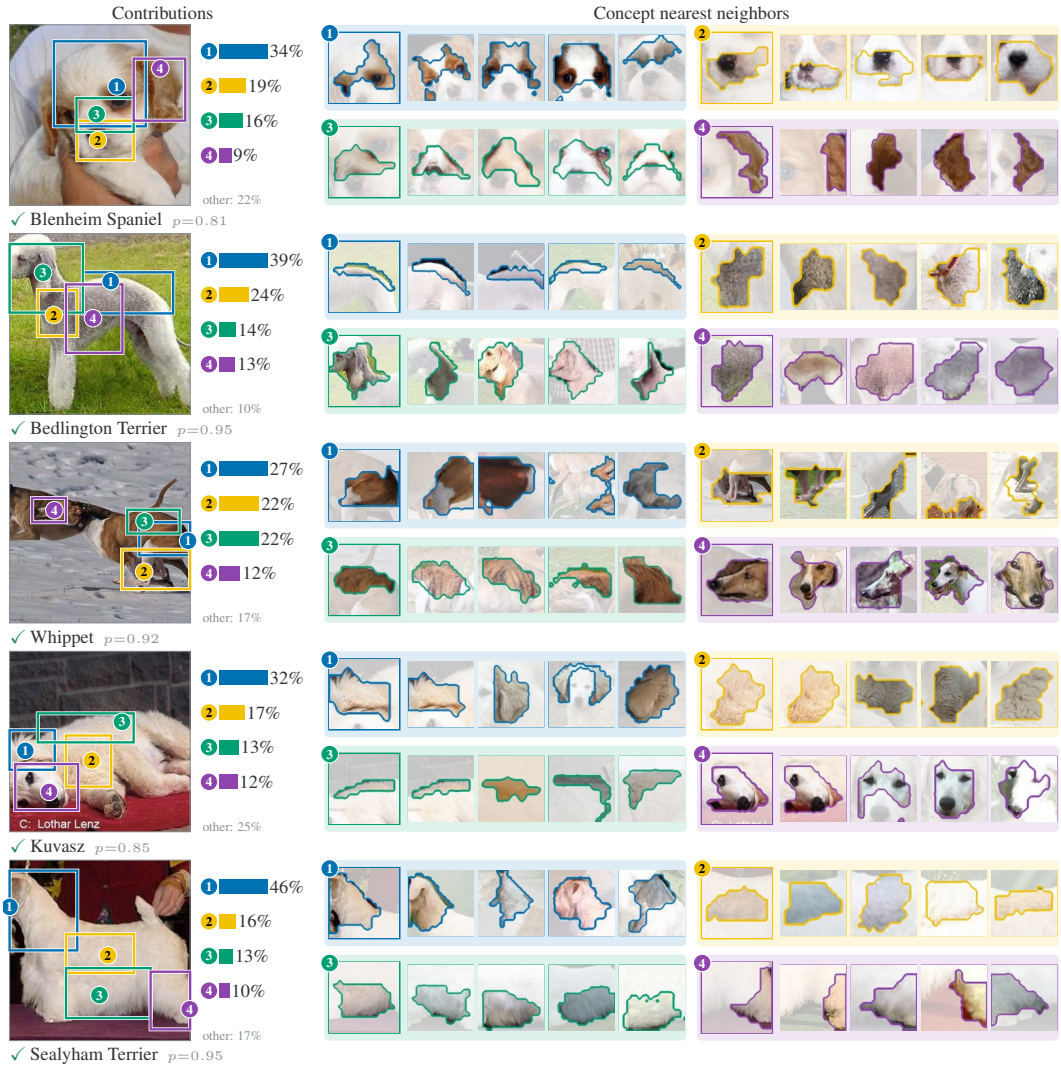


Figure 42: **Stanford Dogs: evidence concepts and what they retrieve.** *Left:* the four concepts with the largest contribution to the predicted-class logit. *Right:* each one cropped from this image (thick border) beside its nearest training occurrences.



Figure 43: **Oxford Flowers: evidence concepts and what they retrieve.** *Left:* the four concepts with the largest contribution to the predicted-class logit. *Right:* each one cropped from this image (thick border) beside its nearest training occurrences.

G.6 WHAT QUANTIZATION KEEPS

This subsection shows what the product-quantized attribute (Jégou et al., 2011) keeps of the continuous one (Figures 44–48). To draw a code, we decode it back into feature space as $r_{i_k}^{\mathrm{dec}}$ using statistics of the training split (Appendix F.7). The images shown come from the PartImageNet validation split and the test splits of the other datasets. *Top:* one PCA basis, fitted on the backbone's foreground patch features, maps features to colors, so colors compare across columns. The columns show the backbone features and each concept region drawn in three ways: as its concept key $\nu(w_{i_k})$, as key plus continuous attribute $\nu(w_{i_k}) + r_{i_k}$, and as key plus quantized attribute $\nu(w_{i_k}) + r_{i_k}^{\mathrm{dec}}$. *Bottom:* for two concepts, the attributes of their occurrences before (left) and after (right) quantization, drawn as image crops at their coordinates in one PCA plane. The two panels are scaled independently, so what matters is which crops stay next to each other.

Top rows. Drawn as its concept key alone, a region has the same color in every image: the key says which concept a region belongs to, not how the region looks. Adding the continuous attribute gives each region back the color of its own backbone features, and the quantized attribute gives nearly the same colors. *Bottom panels.* Quantization keeps the neighborhoods of the attribute space: 55 to 78% of the 5-nearest-neighbor relations (the share in each panel title) are the same before and after. The code therefore keeps which occurrences of a concept look alike, which Section G.7 examines over all occurrences of a concept.

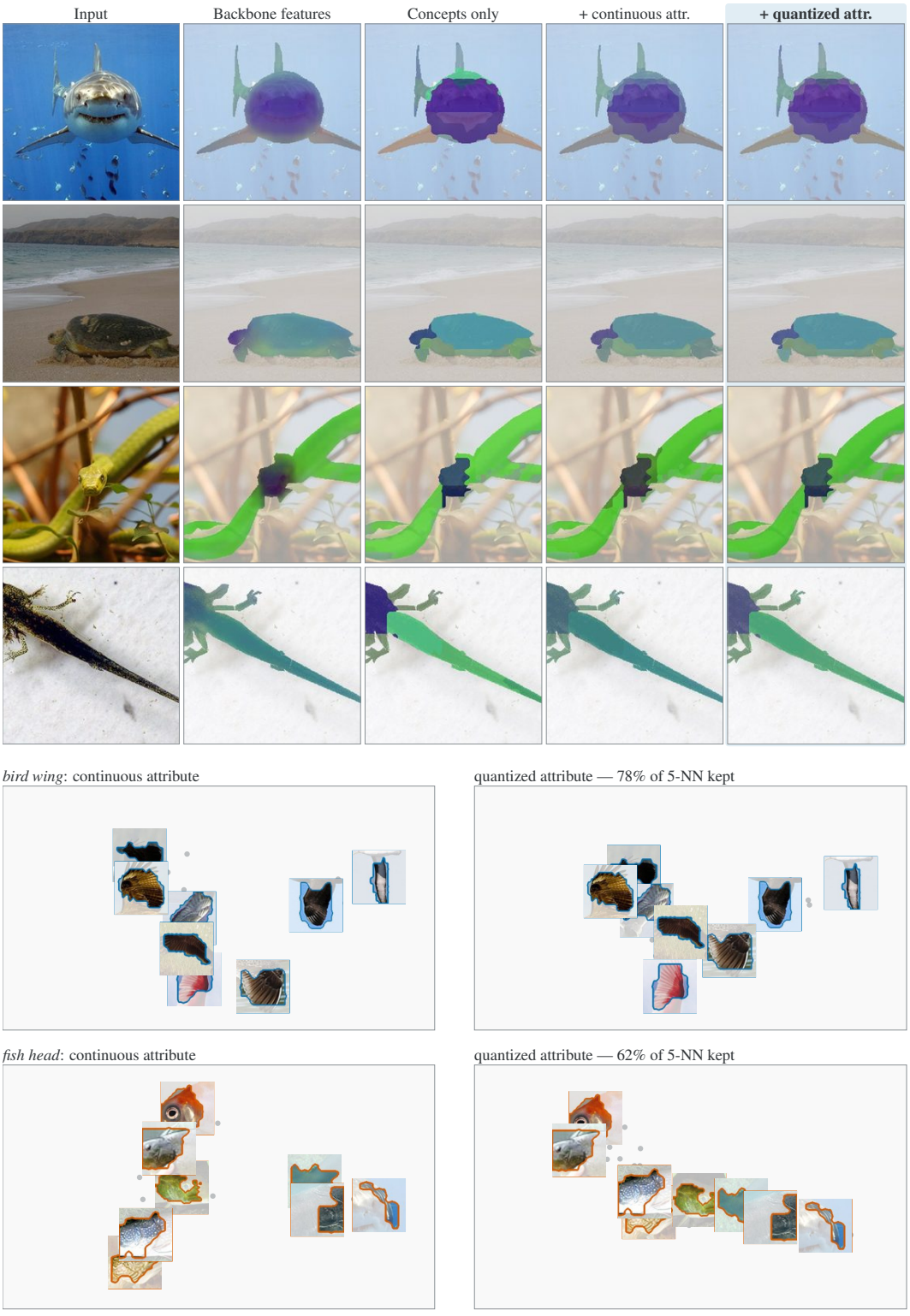


Figure 44: **PartImageNet: what quantization keeps.** *Top:* backbone features, and each concept region as its key, key + continuous attribute and key + quantized attribute, in one PCA basis. *Bottom:* two concepts' attribute spaces before and after quantization (same crops and basis; panels scaled independently).

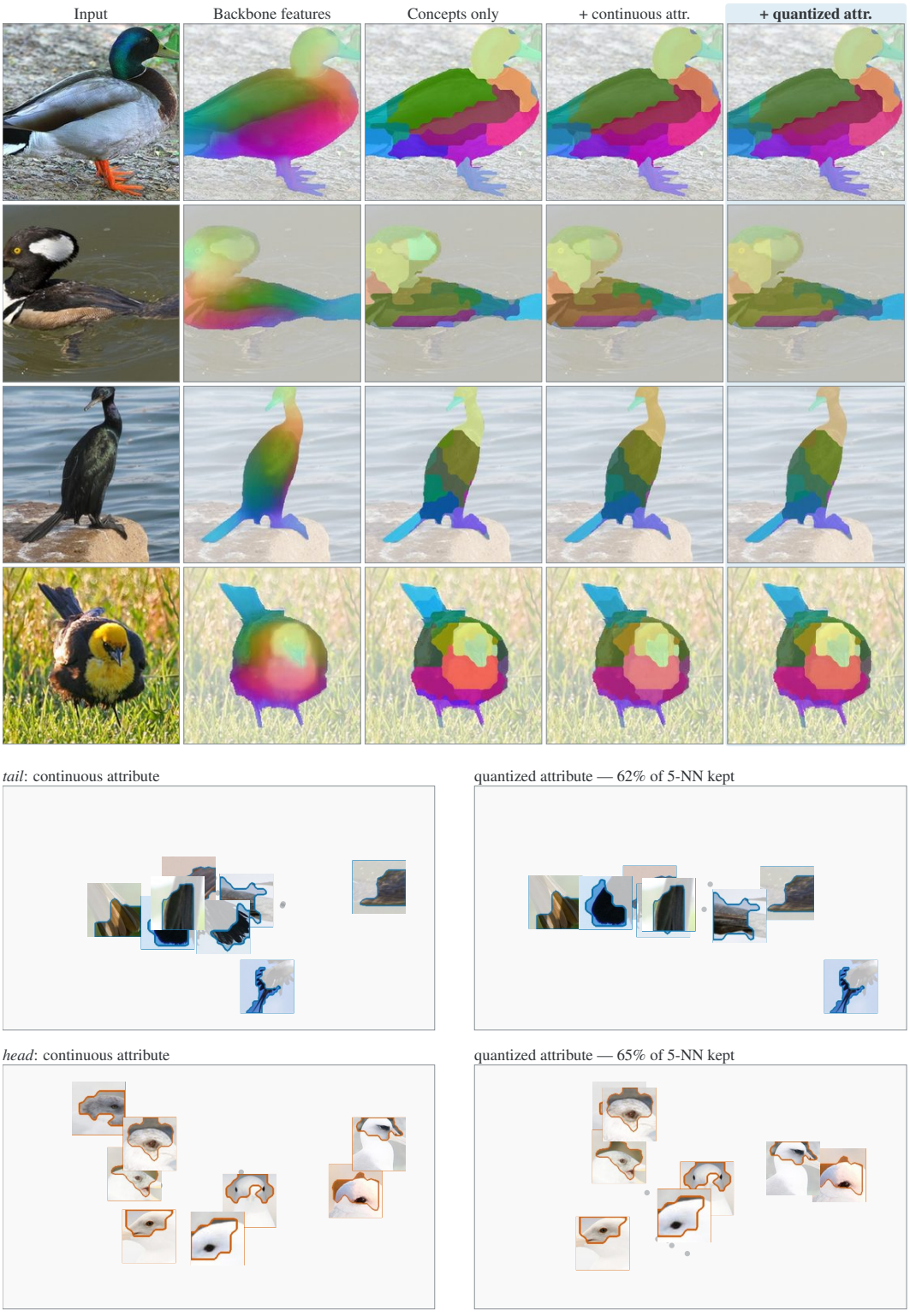
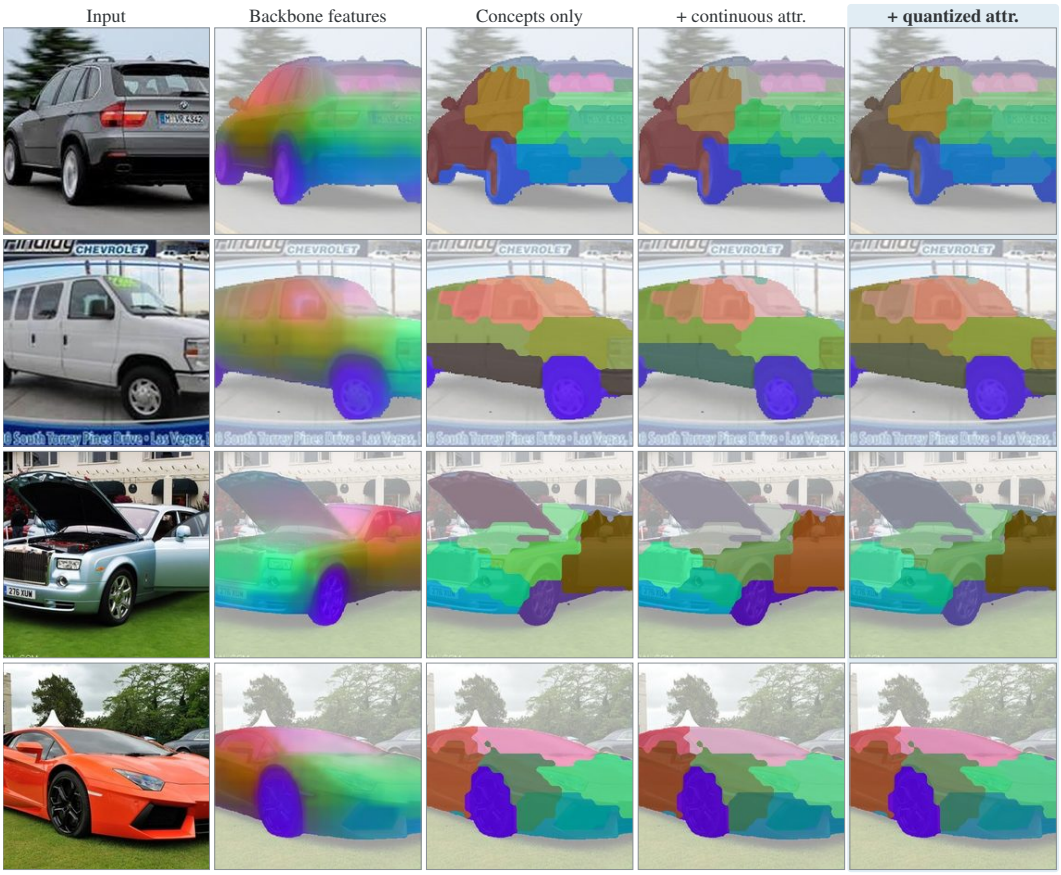


Figure 45: **CUB: what quantization keeps.** *Top:* backbone features, and each concept region as its key, key + continuous attribute and key + quantized attribute, in one PCA basis. *Bottom:* two concepts' attribute spaces before and after quantization (same crops and basis; panels scaled independently).



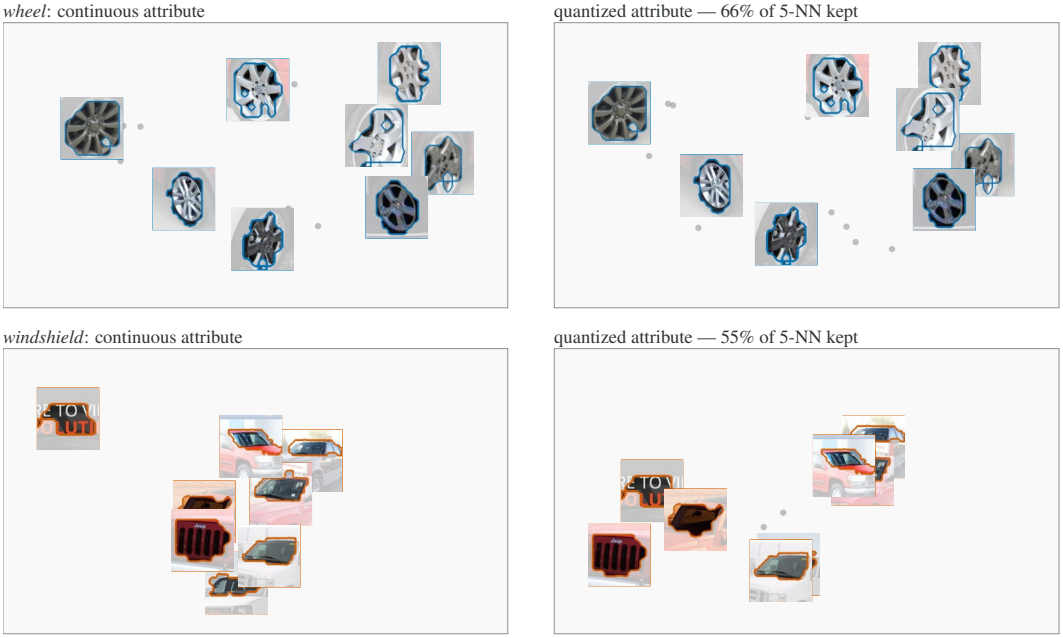


Figure 46: **Stanford Cars: what quantization keeps.** *Top:* backbone features, and each concept region as its key, key + continuous attribute and key + quantized attribute, in one PCA basis. *Bottom:* two concepts' attribute spaces before and after quantization (same crops and basis; panels scaled independently).

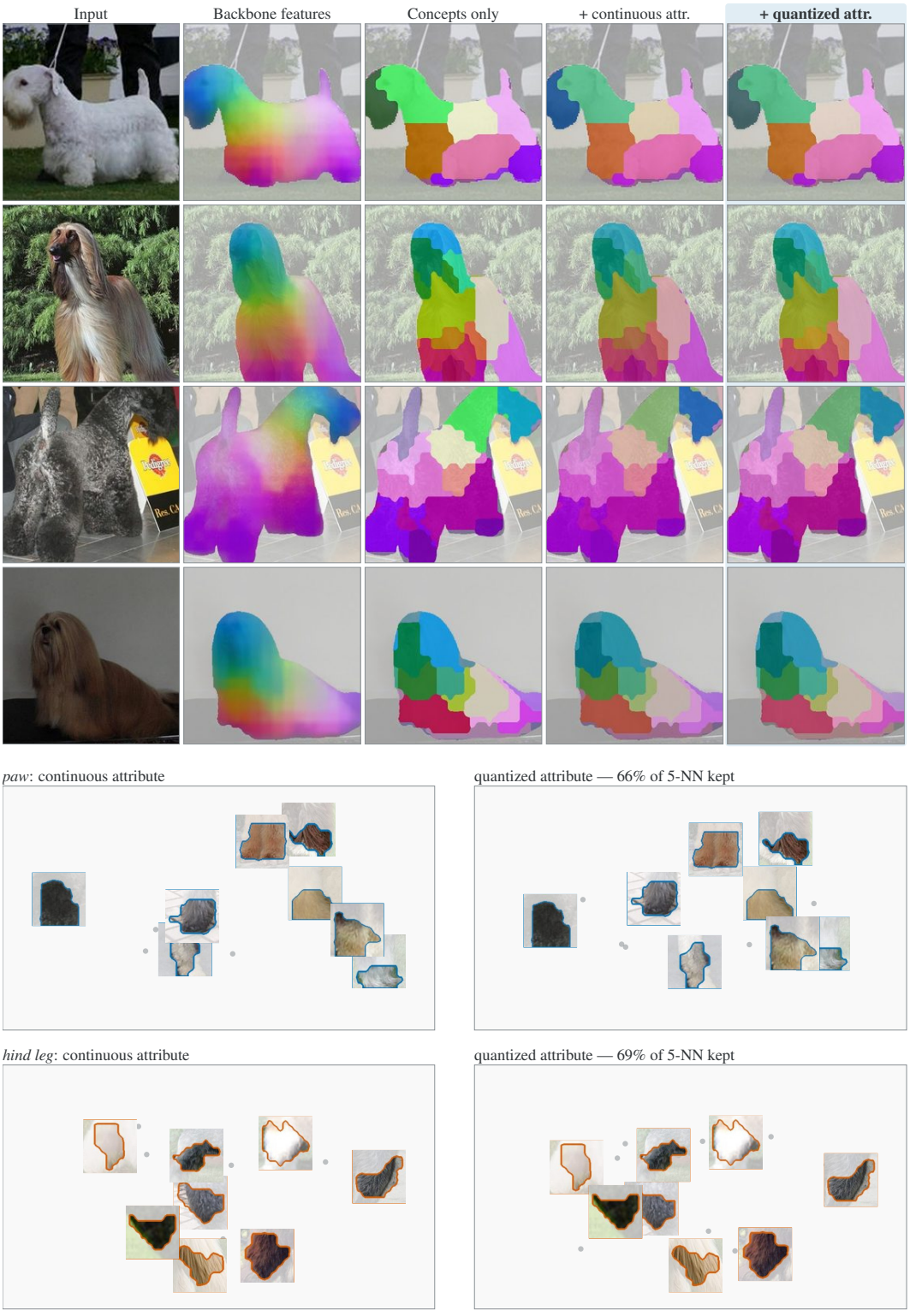


Figure 47: **Stanford Dogs: what quantization keeps.** *Top:* backbone features, and each concept region as its key, key + continuous attribute and key + quantized attribute, in one PCA basis. *Bottom:* two concepts' attribute spaces before and after quantization (same crops and basis; panels scaled independently).

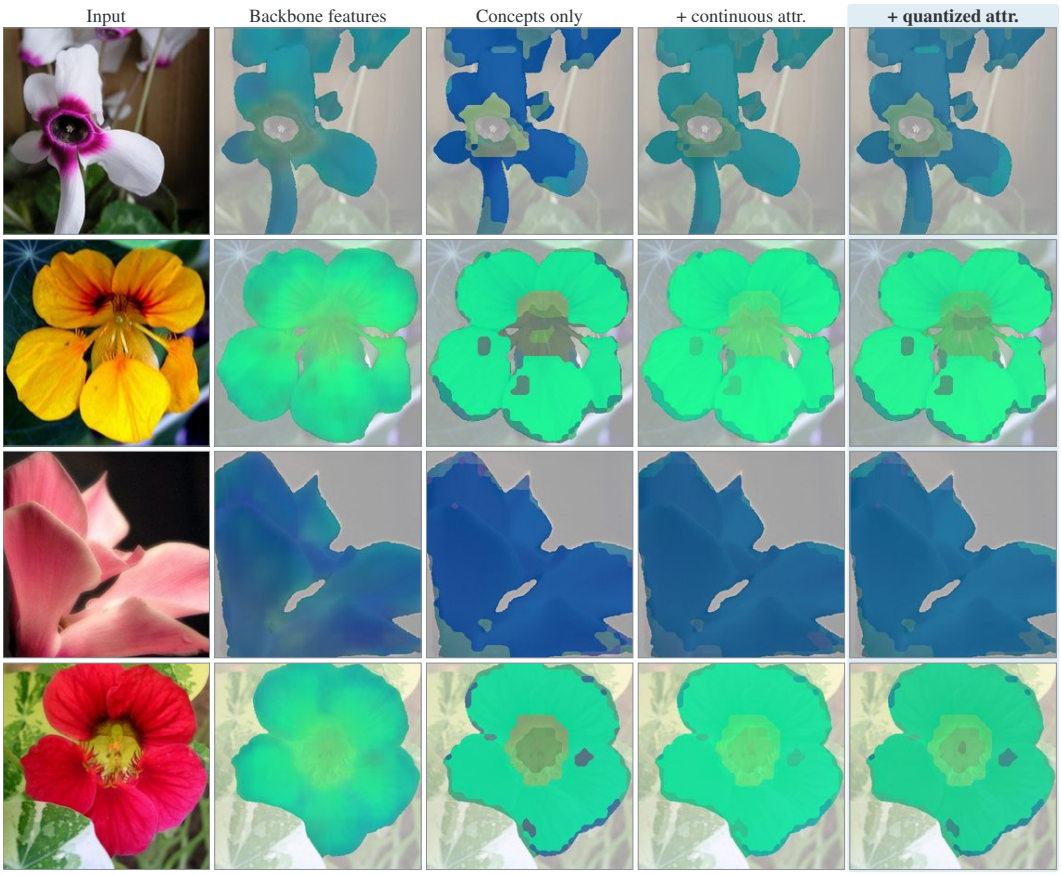




Figure 48: **Oxford Flowers: what quantization keeps.** *Top:* backbone features, and each concept region as its key, key + continuous attribute and key + quantized attribute, in one PCA basis. *Bottom:* two concepts' attribute spaces before and after quantization (same crops and basis; panels scaled independently).

G.7 How one concept varies across images

This subsection shows how one concept varies across images (Figures 49–52). Two occurrences of a concept share its prototype and differ in their attribute codes (Section 3.3). A concept occurs in an image when the model selects it among the $K{=}16$ concepts of that image and gives it at least 1.2% of the 32×32 grid. Each map holds the quantized attributes q_{i_k} (Equation 10) of all occurrences of one concept in images not used in training: the PartImageNet validation and test splits ($3{,}614$ images) and the test splits of CUB ($5{,}794$), Stanford Cars ($8{,}041$) and Stanford Dogs ($8{,}580$). At evaluation, q_{i_k} is the concatenation of the $G{=}16$ selected codewords, so it follows from the discrete code alone.

We fit a PCA to these vectors and draw each occurrence at its coordinates, as a crop with the concept's region outlined or as a dot. The plane holds 12 to 20% of the variance, so positions are approximate; the row below each map gives the exact neighbors of one occurrence (the input, the large crop), its four nearest and four farthest by cosine over the full 768-dimensional attribute. Unlike the bottom panels of Figures 44–48, which compare continuous and quantized attributes on a small gallery, these maps show the quantized attribute of every occurrence.

The frame of a crop gives the class of its image (the ImageNet class on PartImageNet, the species on CUB, the make on Stanford Cars and the breed on Stanford Dogs), with the five most frequent classes of the concept colored and the others gray; the outline color gives the distance to the input, from vermilion (nearest) to blue (farthest). The model never sees these labels during training. They are used here only to color the frames.

Within a concept, occurrences group by the kind of object, although no class label is used in training. The PartImageNet concept on the *quadruped head* separates tigers, leopards, cougars, domestic cats and cheetahs, and the four nearest occurrences of the input leopard are leopards; the four nearest beaks to an evening grosbeak's belong to evening grosbeaks, the nearest grilles to a Mercedes-Benz C-Class come from the same model, and the nearest muzzles to a bull mastiff's are those of bull mastiffs.

This is the division of labor the model is built for. The concept identity is shared across classes and says *which* part is present, such as a grille; the attribute says *which kind* of that part it is, down to the species, the make or the breed. A description by concepts and attributes can therefore separate fine-grained classes that share all their parts, which is consistent with the small gap to the backbone on the fine-grained datasets (Table 3).

What this does not show. The attribute is computed from a frozen DINOv2 region feature, which already separates classes, so part of this structure comes from the backbone. The figures show that the discrete code preserves it within a single concept; they do not show that quantization adds it.

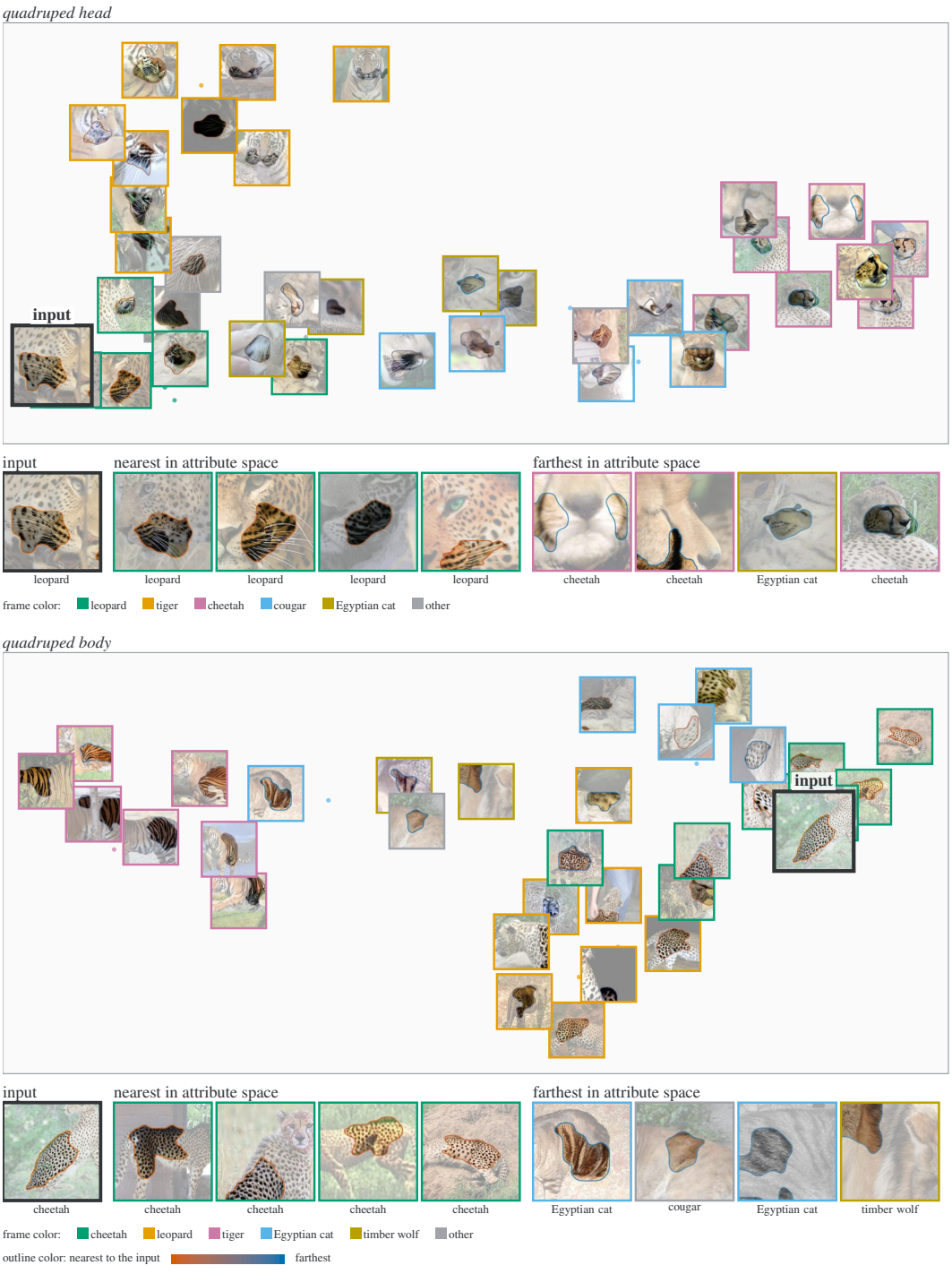


Figure 49: **PartImageNet, *quadruped head*** (64 **images) and *quadruped body*** (60 **images): how a concept of DisParQ varies across images.** *Map:* PCA of the quantized attributes of all occurrences of the concept; each is a crop with its region outlined, or a dot; the dark-framed crop is the input. Frame color: the ImageNet class of the image; outline color: distance to the input. *Row:* the input, its four nearest and four farthest occurrences. On PartImageNet and CUB, a concept is named after the annotated part it falls on.

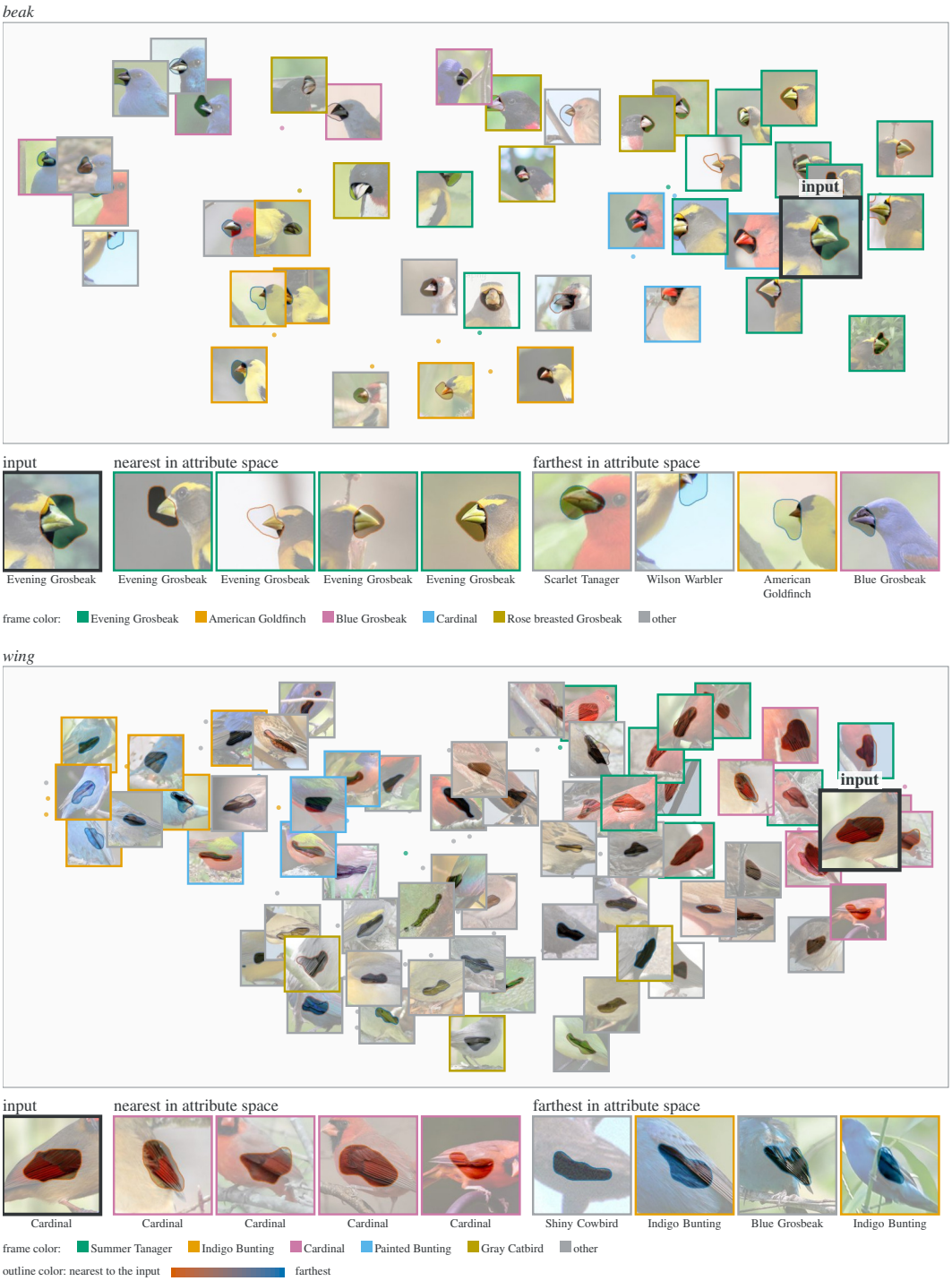


Figure 50: **CUB, *beak* (73 images) and *wing* (191 images): how a concept of DisParQ varies across images.** *Map:* PCA of the quantized attributes of all occurrences of the concept; each is a crop with its region outlined, or a dot; the dark-framed crop is the input. Frame color: the species of the image; outline color: distance to the input. *Row:* the input, its four nearest and four farthest occurrences. On PartImageNet and CUB, a concept is named after the annotated part it falls on.

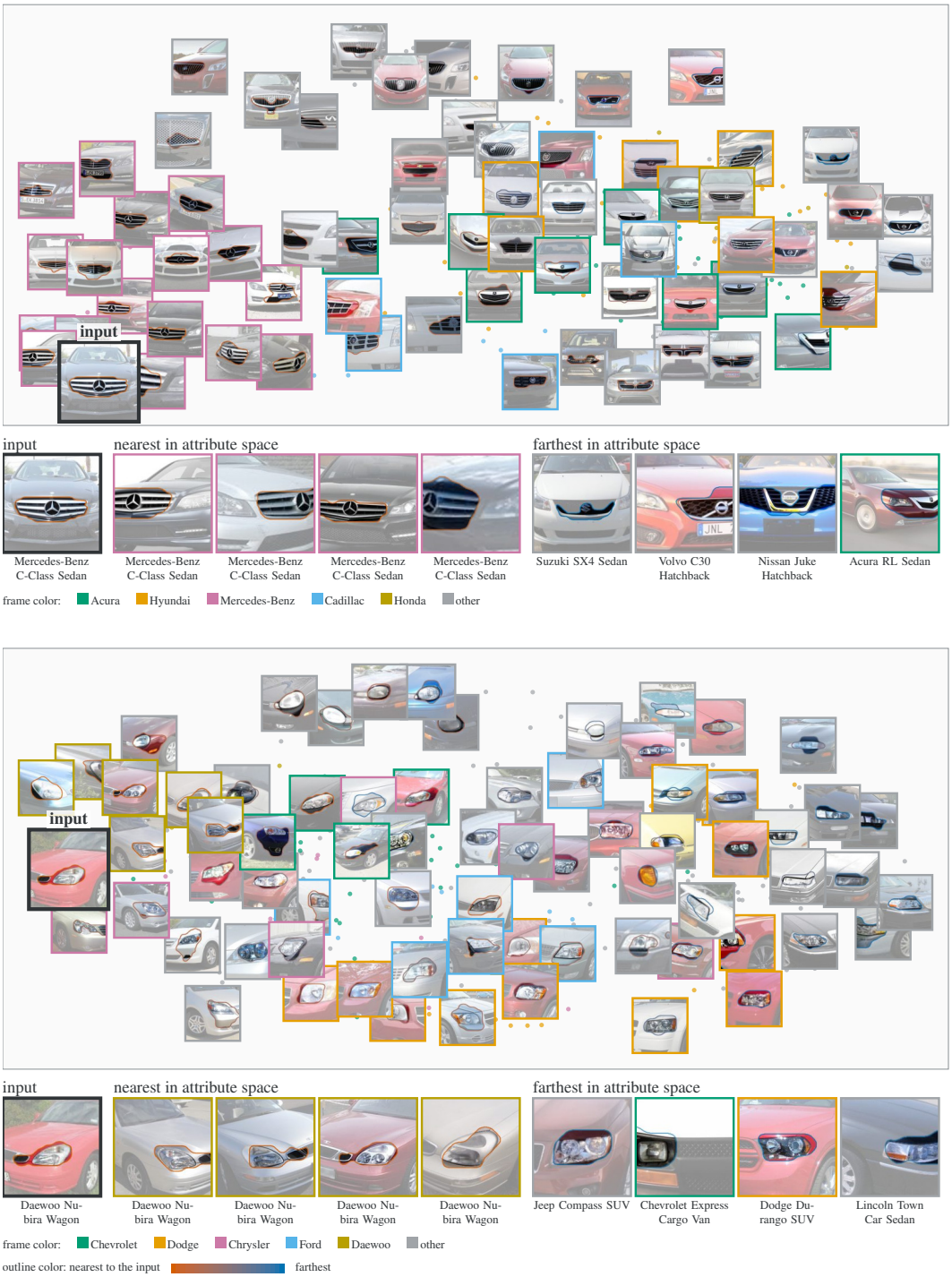


Figure 51: **Stanford Cars, two concepts** (288 **and** 500 **images): how a concept of DisParQ varies across images.** *Map:* PCA of the quantized attributes of all occurrences of the concept; each is a crop with its region outlined, or a dot; the dark-framed crop is the input. Frame color: the make of the image; outline color: distance to the input. *Row:* the input, its four nearest and four farthest occurrences. On PartImageNet and CUB, a concept is named after the annotated part it falls on.

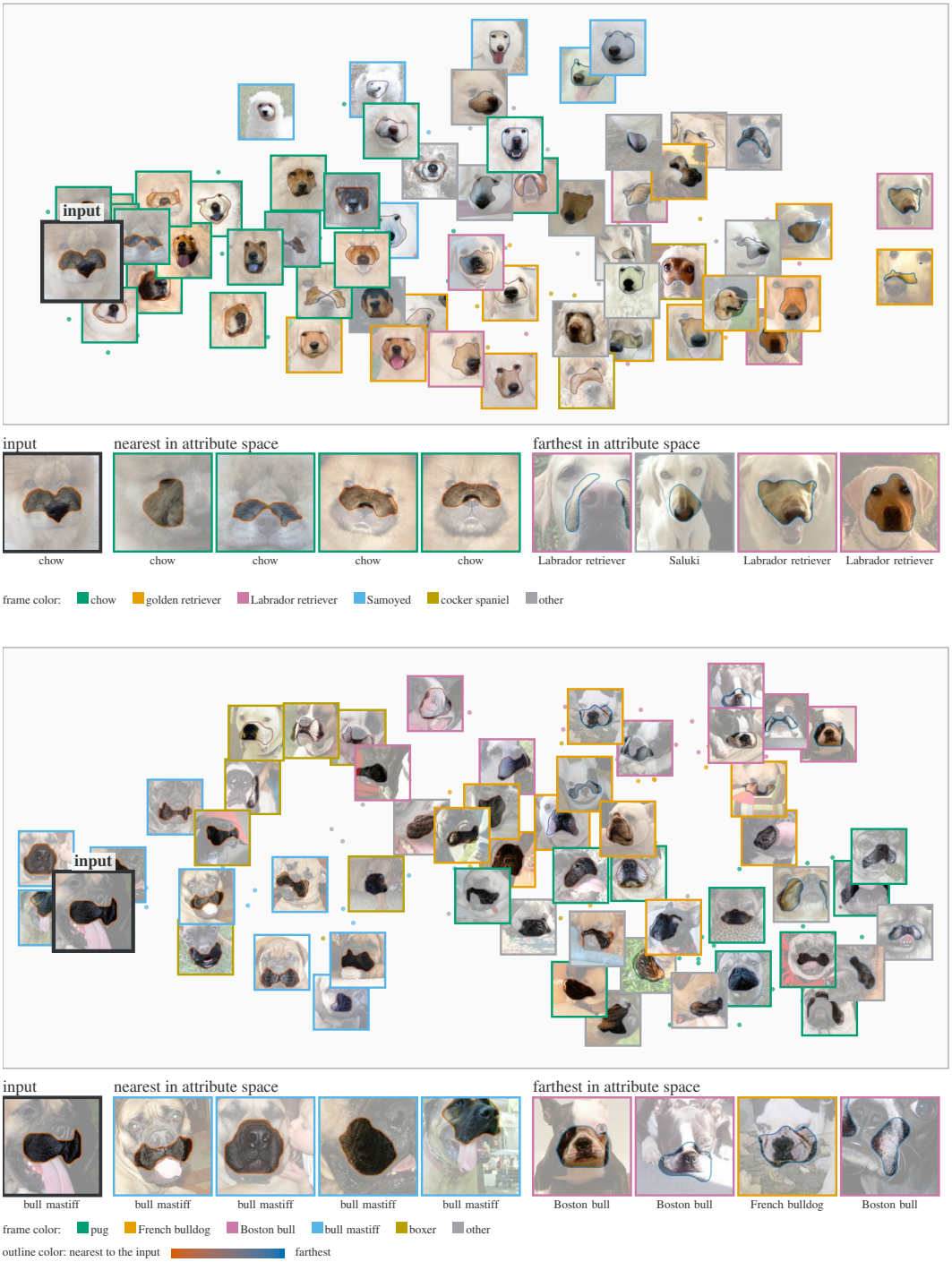


Figure 52: **Stanford Dogs, two concepts** (197 **and** 212 **images): how a concept of DisParQ varies across images.** *Map:* PCA of the quantized attributes of all occurrences of the concept; each is a crop with its region outlined, or a dot; the dark-framed crop is the input. Frame color: the breed of the image; outline color: distance to the input. *Row:* the input, its four nearest and four farthest occurrences. On PartImageNet and CUB, a concept is named after the annotated part it falls on.

H AUTOMATIC ALIGNMENT OF CONCEPT NAMES

DisParQ learns spatially grounded concepts without text supervision (Section 3). In this section, we investigate whether these concepts can be named after training using language-aligned features and a fixed vocabulary, and whether their quantized attributes help describe individual occurrences. We first align the frozen concept descriptions with a language-aligned visual representation, then retrieve names from a fixed vocabulary. Finally, we assess whether those names recover the object or part labels of their regions. All naming methods use the same regions within each model and evaluation dataset, so their comparison isolates the contribution of the naming rule.

H.1 CONCEPT DESCRIPTIONS AND ALIGNMENT TARGETS

Frozen concept models. We use three frozen DisParQ models, two based on DINOv2 and trained on ImageNet-1k and PartImageNet (PIN), respectively, and one based on CLIP-DINOiser and trained on PartImageNet. Following the backbone interfaces in Appendix E.1, the DINOv2 models use $H = W = 32$, while CLIP-DINOiser uses $H = W = 28$ and omits layer fusion.

For each image x, we extract the selected concepts and hard identity map p of Section 3.2. A concept occurrence consists of the locations assigned to a selected bank identity $n = i_k$, namely those with $p_u = n$. Its description combines the fixed prototype w_n with the image-dependent attribute q_n from Equation 10. Naming therefore uses the same quantized attributes as the decoder. As in the method section, dependence on the current image is implicit.

Language-aligned targets. To relate these descriptions to text, we use a frozen CLIP-DINOiser (Wysoczańska et al., 2024) teacher. Its projected patch features follow the language-aligned interface described in Appendix E.1. We bilinearly resample them to the concept model's $H \times W$ grid, average over locations with $p_u = n$, and normalize the resulting vector to unit length. The resulting target $\varphi_n \in \mathbb{R}^{512}$ describes the image content assigned to concept n. Every comparison uses the same teacher and its OpenCLIP ViT-B/16 text encoder (Cherti et al., 2023), trained on LAION-2B (Schuhmann et al., 2022).

Data and fitting split. For each concept model, we fit alignment independently on PartImageNet (He et al., 2022) and VOC2012 (Everingham et al., 2012). We split the 20,466 PartImageNet training images or 5,717 VOC2012 Main training images into 90% for fitting and 10% for adapter selection, using image-level split seed 45. Only the fitting subset determines adapter parameters, input statistics, and mean target vectors. No segmentation labels are used in fitting or selection. Evaluation uses the 1,206 PartImageNet validation images or 1,449 VOC2012 validation images. Each image undergoes a single shorter-side resize and center crop to 224×224, without sliding windows or multiple crops. Annotation masks undergo the paired transformation with nearest-neighbor interpolation. An occurrence used for fitting or adapter selection must occupy at least four assignment-grid cells.

H.2 CONCEPT DESCRIPTIONS TO NAMES

A fixed name can describe what occurrences of a concept have in common, while an attribute-dependent name can also reflect differences between occurrences. We compare both approaches in the teacher's 512-dimensional space using the same vocabulary.

Fixed concept descriptions. The *region mean* averages φ_n over fitting images for each concept and normalizes the mean to unit length. It provides a fixed description without learning an adapter. The *linear key* and *MLP key* methods instead learn a mapping from the prototype w_n to the teacher space. Here "key" refers to the matching prototype w_n. The MLP has one hidden layer of width 512 with GELU. Both mappings produce one normalized vector per concept and therefore one fixed name. We also evaluate the normalized prototype directly (*direct key*) for CLIP-DINOiser, whose prototypes occupy the compatible projected CLIP space. DINOv2 prototypes have no such direct correspondence to text.

Attribute-dependent descriptions. To let names reflect individual occurrences, we add q_n to the adapter input. Before fitting the adapters, we compute a mean and standard deviation for each input dimension across the occurrences in the fitting split, separately for prototypes and attributes. We subtract the corresponding mean and divide by the standard deviation for each adapter input. The

additive adapter applies a learned linear projection to each standardized input, adds the two outputs and a learned bias, and normalizes the result to unit length. The *joint MLP* instead concatenates the standardized inputs and passes them through a 512-unit hidden layer with GELU, followed by a projection to the teacher space and unit-length normalization. This allows the attribute's contribution to depend on the prototype.

Adapter fitting. Adapters minimize cosine distance to φ_n, with occurrences weighted inversely by their concept's fitting frequency. We use AdamW with learning rate 10^{-3}, weight decay 10^{-4}, batch size 512, and at most 100 epochs. We select the adapter with the highest holdout cosine similarity, averaged equally over concepts with at least five qualifying fitting-image occurrences. These are the supported concepts used below. Early stopping uses patience 15 and minimum improvement 10^{-5}.

The *pooled teacher* reference names each validation occurrence directly from φ_n, which is computed by running the teacher on that validation image and pooling its features over the concept region. The adapters instead predict a vector in the teacher's space from w_n and, for the attribute-dependent methods, q_n. The reference therefore shows what names can be retrieved from the teacher's features on the same regions.

Vocabulary retrieval. We use the filtered vocabulary supplied by the authors of CFM (Wittenmayer et al., 2026), retaining 102,171 distinct strings after removing the malformed entry `(#`. Each string has three normalized text embeddings, obtained by averaging the supplied noun, adjective, and verb prompt sets separately. Given a normalized visual description, we select the string with the largest cosine similarity across all strings and prompt groups. Every method uses this same vocabulary and prompt recipe, without CFM's hierarchy-specific naming correction. Dataset class names do not restrict the choice of a name.

H.3 Evaluating the retrieved names

Name-to-class retrieval. We assess the selected name by asking whether it retrieves the annotated label of its region. We re-encode the name and the dataset class names with a shared prompt ensemble: the teacher's 80 ImageNet templates for VOC, and "a photo of {}." / "a close-up photo of {}." for PartImageNet. We then rank class embeddings by cosine similarity to the name embedding. The candidates are the 20 VOC object classes or 40 PartImageNet part classes, plus background. This evaluates the retrieved text through its embedding, rather than matching strings exactly or classifying directly from the adapter output.

Eligible regions and metrics. We evaluate validation occurrences of supported concepts that occupy at least four assignment-grid cells. We upsample each hard region to the 224×224 annotation grid by nearest neighbor and ignore void pixels. For inclusion in the primary metric, the dominant ground-truth label must be foreground and cover at least 75% of the region's valid pixels. This focuses evaluation on regions with a clear annotated label.

Let $R_n(x)$ be the rank of that label when retrieved from the predicted name, and let $\mathcal{O}_n$ contain eligible validation images for concept n. We average first over occurrences of each concept and then over the set $\mathcal{C}$ of concepts with at least one eligible occurrence:

$$\begin{aligned} \mathrm{Acc}_{\mathrm{name}} &= \frac{1}{|\mathcal{C}|} \sum_{n \in \mathcal{C}} \frac{1}{|\mathcal{O}_n|} \sum_{x \in \mathcal{O}_n} \mathbb{1}[R_n(x) = 1], \\ \mathrm{MRR} &= \frac{1}{|\mathcal{C}|} \sum_{n \in \mathcal{C}} \frac{1}{|\mathcal{O}_n|} \sum_{x \in \mathcal{O}_n} \frac{1}{R_n(x)}. \end{aligned} \tag{22}$$

Top-1 accuracy measures how often the correct label ranks first, while mean reciprocal rank (MRR) also credits labels retrieved below first place. Tables 21 and 22 report $100\mathrm{Acc}_{\mathrm{name}}$ and MRR, together with the eligible region and concept counts. Those subsets differ across concept models, including for the pooled-teacher reference. No human or language-model judge is used. The text encoder both selects names and scores their correspondence to class labels, so the metric is not an independent semantic judgment. The confidence intervals measure variation across sampled concepts, not across training seeds.

	ImageNet / DINOv2		PIN / DINOv2		PIN / CLIP-DINOiser	
Method	Top-1	MRR	Top-1	MRR	Top-1	MRR
Region mean	15.28	0.2764	23.18	0.3623	26.88	0.4133
Direct key	—	—	—	—	31.95	0.4486
Linear key	12.29	0.2468	22.95	0.3598	27.40	0.4245
MLP key	14.82	0.2722	22.29	0.3544	26.30	0.4074
Additive	16.04	0.2905	22.53	0.3651	30.71	0.4534
Joint MLP	23.74	0.3691	21.74	0.3553	26.86	0.4136
Pooled teacher (reference)	29.75	0.4314	26.02	0.3959	30.97	0.4544

Table 21: PartImageNet name-to-class retrieval on validation regions. Top-1 is a percentage, while MRR lies in $[0, 1]$. Both are macro-averaged over concepts. Column groups identify concept-training dataset / backbone. Adapters are fitted independently on PartImageNet training images. Eligible region / concept counts are 5,541 / 467, 4,758 / 416, 1,208 / 171, respectively. The teacher reference pools features from each model's own regions.

	ImageNet / DINOv2		PIN / DINOv2		PIN / CLIP-DINOiser	
Method	Top-1	MRR	Top-1	MRR	Top-1	MRR
Region mean	30.09	0.4643	41.28	0.5490	39.96	0.5531
Direct key	—	—	—	—	44.97	0.5878
Linear key	27.90	0.4384	41.45	0.5515	44.32	0.5794
MLP key	29.13	0.4537	41.54	0.5512	39.40	0.5481
Additive	41.56	0.5515	54.07	0.6396	58.50	0.6895
Joint MLP	53.14	0.6438	61.54	0.7004	58.17	0.6936
Pooled teacher (reference)	73.82	0.8128	80.21	0.8630	73.33	0.8103

Table 22: VOC2012 name-to-class retrieval on validation regions. Top-1 is a percentage, while MRR lies in $[0, 1]$. Both are macro-averaged over concepts. Column groups identify concept-training dataset / backbone. Adapters are fitted independently on VOC training images. Eligible region / concept counts are 7,567 / 633, 7,088 / 368, 3,576 / 190, respectively. The teacher reference pools features from each model's own regions.

H.4 Results

Object names on VOC. The joint MLP improves top-1 retrieval over region means by 23.05, 20.26, and 18.21 percentage points for ImageNet/DINOv2, PartImageNet/DINOv2, and PartImageNet/CLIP-DINOiser, respectively (Table 22). Paired concept-bootstrap 95% intervals are $[20.35, 25.48]$, $[16.86, 23.56]$, and $[13.75, 22.33]$ points, using 1,000 resamples. The additive adapter performs slightly better than the joint MLP for PartImageNet/CLIP-DINOiser (58.50 versus 58.17). Attribute-dependent adapters therefore improve object-name retrieval over fixed region-mean names in all three settings.

Part names on PartImageNet. The gains do not extend uniformly to part labels (Table 21). The joint MLP improves over region means by 8.46 points for ImageNet/DINOv2 (95% interval $[5.96, 10.83]$), but changes PartImageNet/DINOv2 and PartImageNet/CLIP-DINOiser by -1.44 and -0.02 points, with intervals $[-3.22, 0.40]$ and $[-3.53, 2.98]$. Direct CLIP-DINOiser prototypes remain competitive. Better agreement with the teacher also need not yield better part names: for PartImageNet/DINOv2, joint-MLP cosine similarity to the pooled teacher increases from the region mean's 0.8169 to 0.8615, while top-1 retrieval falls from 23.18 to 21.74. The pooled teacher itself achieves only $26 - -31\%$ top-1 on PartImageNet part labels.

H.5 Qualitative examples

Gallery construction. For each supported concept, the gallery contains its closest qualifying validation image and up to nine additional, distinct images sampled reproducibly. A displayed occurrence must be among the image's K selected concepts and occupy at least 1% of the assignment grid. The closest image minimizes the prototype distance $\|z_u - w_n\|_2^2$ from Equation 1 over image-grid

locations. The gallery separates fixed names from attribute-dependent names and includes pooled-teacher predictions, with five candidate names available per method. The 1% threshold applies only to visualization. Quantitative evaluation retains the four-cell threshold above.

Selection for the figures. Figures 53 and 54 each show ten concepts. To select them, we exclude background-associated concepts by requiring at least 85% mean foreground overlap across their saved gallery examples and at least 75% in each of the five displayed examples. We then inspect contact sheets to select concepts with diverse classes, excluding rows with potentially inappropriate displayed labels. Each row retains the closest example and four random examples. The figures illustrate differences between methods rather than estimating naming accuracy. All displayed names are unedited top-one predictions.

Additional selected examples. Figures 55 and 56 apply the same support, foreground-overlap, label-screening rules to two additional model–dataset pairs. Each row retains the closest saved occurrence and four visually similar examples chosen manually from the remaining saved gallery images.

Concept-level names	Closest	Random 1	Random 2	Random 3	Random 4
Concept 437 **Direct:** N/A **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** car images **T:** alloy wheels	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** alloy wheels	 **A:** car images **J:** car images **T:** alloy wheels	 **A:** car images **J:** motorcycle wheel **T:** wheel motorcycle
Concept 600 **Direct:** N/A **Mean:** bus service **Linear:** bus route **MLP:** bus service	 **A:** courtesy bus **J:** courtesy bus **T:** courtesy bus	 **A:** city bus **J:** city bus **T:** trolley bus	 **A:** train at **J:** train at **T:** train at	 **A:** city bus **J:** bus service **T:** city bus	 **A:** train at **J:** train at **T:** train at
Concept 467 **Direct:** N/A **Mean:** train at **Linear:** train at **MLP:** train at	 **A:** train at **J:** railroad train **T:** rail vehicle	 **A:** train at **J:** railroad train **T:** railroad train	 **A:** train at **J:** railroad train **T:** railroad company	 **A:** train at **J:** train at **T:** train at	 **A:** train at **J:** railroad train **T:** railroad train
Concept 313 **Direct:** N/A **Mean:** these **Linear:** these **MLP:** these	 **A:** recent **J:** latest **T:** bottle label	 **A:** these **J:** orig **T:** bottle out of	 **A:** bottle out of **J:** bottle out of **T:** drink driver	 **A:** these **J:** these **T:** bottle out of	 **A:** these **J:** bottle out of **T:** bottle out of
Concept 608 **Direct:** N/A **Mean:** recent **Linear:** haired **MLP:** recent	 **A:** baby hat **J:** recent **T:** asian couple	 **A:** baby hat **J:** recent **T:** frat	 **A:** baby hat **J:** recent **T:** two sitter sofa	 **A:** baby hat **J:** recent **T:** baby bed	 **A:** baby hat **J:** recent **T:** toddler girl
Concept 559 **Direct:** N/A **Mean:** house cat **Linear:** house cat **MLP:** house cat	 **A:** cat **J:** house cat **T:** burmese cat	 **A:** house cat **J:** house cat **T:** house cat	 **A:** house cat **J:** house cat **T:** house cat	 **A:** house cat **J:** house cat **T:** house cat	 **A:** house cat **J:** house cat **T:** house cat
Concept 562 **Direct:** N/A **Mean:** dog breed **Linear:** dog breed **MLP:** dog breed	 **A:** dog breed **J:** french bulldog **T:** pug	 **A:** dog breed **J:** dog breeding **T:** dog breeding	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** dog breeding **J:** dog breeding **T:** english bulldog	 **A:** dog breed **J:** dog breeding **T:** shih
Concept 659 **Direct:** N/A **Mean:** farm animal **Linear:** dog breeding **MLP:** farm animal	 **A:** cattle breeding **J:** farm horse **T:** horse's mouth	 **A:** bovine **J:** farm animal **T:** cow head	 **A:** dog breeding **J:** dog breeding **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** farm animal **J:** farm animal **T:** farm animal
Concept 1022 **Direct:** N/A **Mean:** farm animals **Linear:** cattle breeding **MLP:** farm animal	 **A:** farm horse **J:** farm horse **T:** horse breeding	 **A:** farm horse **J:** farm horse **T:** pony trekking	 **A:** farm horse **J:** farm horse **T:** farm horse	 **A:** farm animals **J:** baby sheep **T:** sheep pen	 **A:** sheep **J:** lamb wool **T:** sheep
Concept 984 **Direct:** N/A **Mean:** avian **Linear:** avian **MLP:** avian	 **A:** avian **J:** cute birds **T:** cute birds	 **A:** avian **J:** avian **T:** red-throated diver	 **A:** avian **J:** avian **T:** mourning dove	 **A:** waterfowl **J:** waterfowl **T:** freckled duck	 **A:** avian **J:** bird watching **T:** rubythroated

Figure 53: **PartImageNet-trained DINOv2 concepts named on VOC (selected examples).** Each row shows the closest occurrence and four saved random examples. Left: fixed direct-key, region-mean, linear-key, and MLP-key names. Direct CLIP naming is undefined for DINOv2 (N/A). Below each image: **A**, additive, **J**, joint MLP, gray **T**, teacher.

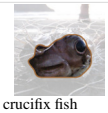
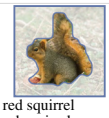
Concept-level names	Closest	Random 1	Random 2	Random 3	Random 4
Concept 676 **Direct:** googly eyes **Mean:** googly eyes **Linear:** photo shows **MLP:** googly eyes	 **A:** googly eyes **J:** googly eyes **T:** googly eyes	 **A:** googly eyes **J:** googly eyes **T:** set eyes on	 **A:** with an eye to **J:** with an eye to **T:** googly eyes	 **A:** have my eye on **J:** hairy eyeball **T:** with an eye to	 **A:** with an eye to **J:** rare bird **T:** game bird
Concept 653 **Direct:** loud mouth **Mean:** loud mouth **Linear:** loud mouth **MLP:** loud mouth	 **A:** mouth watering **J:** mouth-watering **T:** loud mouth	 **A:** wolf dog **J:** crying wolf **T:** loud mouth	 **A:** mouth watering **J:** loud mouth **T:** dog breed	 **A:** mouth watering **J:** mouth-watering **T:** bigmouthed	 **A:** mouth watering **J:** mouth watering **T:** dog breed
Concept 419 **Direct:** puffer fish **Mean:** fish species **Linear:** fish species **MLP:** fish species	 **A:** fish species **J:** fish species **T:** fish species	 **A:** fish species **J:** fish species **T:** fish species	 **A:** fish species **J:** fish species **T:** sea toad	 **A:** crucifix fish **J:** fish for **T:** fish meal	 **A:** crucifix fish **J:** fish for **T:** fish for
Concept 884 **Direct:** food fish **Mean:** sport fish **Linear:** sport fish **MLP:** sport fish	 **A:** fish for **J:** sport fish **T:** sport fish	 **A:** sport fish **J:** fish for **T:** rainbow trout	 **A:** freshwater fish **J:** freshwater fish **T:** trout	 **A:** freshwater fish **J:** freshwater fish **T:** fish for	 **A:** freshwater fish **J:** fish for **T:** lake trout
Concept 445 **Direct:** tortoiseshell turtle **Mean:** tortoiseshell turtle **Linear:** turtles **MLP:** tortoiseshell turtle	 **A:** tortoiseshell turtle **J:** tortoiseshell turtle **T:** turtles	 **A:** tortoiseshell turtle **J:** tortoiseshell turtle **T:** soft-shelled turtle	 **A:** turtles **J:** water turtle **T:** soft-shelled turtle	 **A:** leatherback turtle **J:** sea turtle **T:** leatherback turtle	 **A:** soft-shelled turtle **J:** sea turtle **T:** female of the species is more deadly than the male
Concept 615 **Direct:** bald eagle **Mean:** american eagle **Linear:** american eagle **MLP:** american eagle	 **A:** bald eagle **J:** american eagle **T:** american eagle	 **A:** american eagle **J:** american eagle **T:** american eagle	 **A:** bird of prey **J:** eagle nest **T:** fish eagle	 **A:** bald eagle **J:** bald eagle **T:** bald eagle	 **A:** american eagle **J:** american eagle **T:** american eagle
Concept 109 **Direct:** squirrel **Mean:** squirrel **Linear:** squirrel **MLP:** squirrel	 **A:** red squirrel **J:** red squirrel **T:** squirrel	 **A:** fox squirrel **J:** squirrel **T:** squirrel	 **A:** squirrel **J:** squirrel **T:** tree squirrel	 **A:** squirrel **J:** tree squirrel **T:** squirrel	 **A:** squirrel **J:** squirrel **T:** squirrel
Concept 190 **Direct:** leopardprint **Mean:** leopardprint **Linear:** leopardprint **MLP:** leopardprint	 **A:** leopardprint **J:** leopardprint **T:** leopardprint	 **A:** leopardprint **J:** leopardprint **T:** leopard print	 **A:** big cat **J:** leopardprint **T:** big cat	 **A:** leopard **J:** leopard **T:** leopard	 **A:** big cat **J:** big cat **T:** leopardprint
Concept 166 **Direct:** tire rim **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** car images **T:** car	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** suv	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** alloy wheels
Concept 76 **Direct:** farm machinery **Mean:** tractor **Linear:** tractor **MLP:** tractor	 **A:** tractor **J:** tractor **T:** tractor	 **A:** truck farming **J:** truck wiring **T:** plow blade	 **A:** truck farming **J:** truck farming **T:** truck farming	 **A:** truck farming **J:** truck farming **T:** truck farming	 **A:** two-wheeled tractor **J:** two-wheeled tractor **T:** two-wheeled tractor

Figure 54: **PartImageNet-trained CLIP-DINOiser concepts named on PartImageNet (selected examples).** Each row shows the closest occurrence and four saved random examples. Direct names use the compatible projected CLIP prototype space. Labels denote **A**, additive, **J**, joint MLP, gray **T**, teacher.

Concept-level names	Closest	Similar 1	Similar 2	Similar 3	Similar 4
Concept 166 **Direct:** tire rim **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** car images **T:** car images	 **A:** suvs **J:** sport utility vehicle **T:** suv	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** alloy wheels	 **A:** car images **J:** car images **T:** car images
Concept 432 **Direct:** trolley bus **Mean:** trolley bus **Linear:** trolley bus **MLP:** trolley bus	 **A:** buses **J:** bus service **T:** bus service	 **A:** trolley bus **J:** city bus **T:** articulated bus	 **A:** bus service **J:** bus service **T:** city bus	 **A:** trolley bus **J:** city bus **T:** city bus	 **A:** bus service **J:** bus service **T:** bus service
Concept 67 **Direct:** bull nose **Mean:** dog breeding **Linear:** dog breed **MLP:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breeding	 **A:** peta **J:** dog breeding **T:** dairy cow	 **A:** dog breed **J:** dog breed **T:** dogfriendly	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breed
Concept 75 **Direct:** house cat **Mean:** house cat **Linear:** house cat **MLP:** house cat	 **A:** house cat **J:** house cat **T:** cat food	 **A:** house cat **J:** house cat **T:** house cat	 **A:** house cat **J:** house cat **T:** house cat	 **A:** burmese cat **J:** house cat **T:** burmese cat	 **A:** bear cat **J:** cat **T:** cute cat
Concept 261 **Direct:** cattle breeding **Mean:** cattle breeding **Linear:** cattle breeding **MLP:** cattle breeding	 **A:** cattle breeding **J:** cattle breeding **T:** cattle breeding	 **A:** cattle farm **J:** cattle farm **T:** cattle farm	 **A:** farm horse **J:** farm horse **T:** farm horse	 **A:** beef cattle **J:** bovine **T:** beef plant	 **A:** beef cattle **J:** beef cattle **T:** beef cattle
Concept 92 **Direct:** sweatshirt **Mean:** outerwear **Linear:** outerwear **MLP:** outerwear	 **A:** outerwear **J:** winter jacket **T:** winter jacket	 **A:** outerwear **J:** outerwear **T:** body warmer	 **A:** winter jacket **J:** outerwear **T:** sweatshirt	 **A:** winter jacket **J:** outerwear **T:** waterproof jacket	 **A:** winter jacket **J:** outerwear **T:** toddler boy
Concept 374 **Direct:** avian **Mean:** avian **Linear:** avian **MLP:** avian	 **A:** avian **J:** avian **T:** myna bird	 **A:** bird food **J:** avian **T:** mallard	 **A:** rare bird **J:** macaw **T:** macaws	 **A:** rare bird **J:** parrots **T:** cute birds	 **A:** rare bird **J:** rare bird **T:** avian
Concept 138 **Direct:** baby sheep **Mean:** sheep **Linear:** sheep **MLP:** sheep	 **A:** baby sheep **J:** sheep **T:** baby sheep	 **A:** lamb wool **J:** lamb wool **T:** lamb wool	 **A:** sheep **J:** sheep **T:** merino sheep	 **A:** sheeps **J:** sheeps **T:** sheeps	 **A:** baby sheep **J:** sheep **T:** lamb wool
Concept 23 **Direct:** trail riding **Mean:** horse riding **Linear:** horse riding **MLP:** horse riding	 **A:** horseback riding **J:** horse riding **T:** ride horseback	 **A:** ride horseback **J:** horse riding **T:** cattle ranch	 **A:** horseback riding **J:** horse riding **T:** race horse	 **A:** horse riding **J:** horse riding **T:** horseback riding	 **A:** horse riding **J:** horse riding **T:** race horse
Concept 1009 **Direct:** cycling jersey **Mean:** maillot **Linear:** maillot **MLP:** maillot	 **A:** cycling jersey **J:** maillot **T:** cycling jersey	 **A:** riding school **J:** pack riding **T:** livery driver	 **A:** cycling jersey **J:** these **T:** cycling jersey	 **A:** cycling jersey **J:** cycling jersey **T:** cycling jersey	 **A:** jerseys **J:** jerseys **T:** cycling jersey

Figure 55: **PartImageNet-trained CLIP-DINOiser concepts named on VOC (curated examples).** Each row shows the closest occurrence and four visually similar saved examples. Fixed names appear at left; per-image labels are **A**, additive, **J**, joint MLP, and gray **T**, teacher.

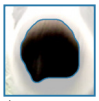
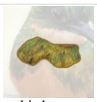
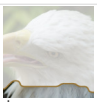
Concept-level names	Closest	Similar 1	Similar 2	Similar 3	Similar 4
Concept 82 **Direct:** N/A **Mean:** dog breed **Linear:** dog breed **MLP:** dog breed	 **A:** cute **J:** cute **T:** dog breeding	 **A:** dog breed **J:** dog breeding **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breeding	 **A:** dog breed **J:** dog breeding **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dogfriendly
Concept 231 **Direct:** N/A **Mean:** dog breed **Linear:** dog breed **MLP:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** companion dog	 **A:** dog breed **J:** dog breed **T:** dog breed	 **A:** dog breed **J:** dog breed **T:** dog breed
Concept 672 **Direct:** N/A **Mean:** monkey **Linear:** monkey **MLP:** monkey	 **A:** animal lover **J:** monkey **T:** howler monkey	 **A:** monkey **J:** monkey jacket **T:** baboon	 **A:** monkey **J:** monkey **T:** endangered species	 **A:** howler monkey **J:** howler monkey **T:** howler monkey	 **A:** monkey **J:** monkey jacket **T:** monkey jacket
Concept 400 **Direct:** N/A **Mean:** tortoiseshell turtle **Linear:** tortoiseshell turtle **MLP:** tortoiseshell turtle	 **A:** tortoise shell **J:** turtle shell **T:** tortoise shell	 **A:** tortoiseshell turtle **J:** tortoiseshell turtle **T:** tortoiseshell turtle	 **A:** sea turtle **J:** sea turtle **T:** sea turtle	 **A:** tortoiseshell turtle **J:** tortoiseshell turtle **T:** tortoiseshell turtle	 **A:** tortoiseshell turtle **J:** tortoiseshell turtle **T:** tortoiseshell turtle
Concept 507 **Direct:** N/A **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** truck wiring **T:** veteran car	 **A:** car images **J:** truck wiring **T:** ford f	 **A:** courtesy car **J:** car images **T:** muscle car	 **A:** car images **J:** autos post **T:** palace car	 **A:** car images **J:** truck wiring **T:** truck wiring
Concept 965 **Direct:** N/A **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** camper van	 **A:** car images **J:** car images **T:** police van
Concept 623 **Direct:** N/A **Mean:** car images **Linear:** car images **MLP:** car images	 **A:** car images **J:** car images **T:** headlight	 **A:** car images **J:** car manufacturer **T:** sports car	 **A:** car images **J:** car images **T:** car manufacturer	 **A:** car images **J:** car images **T:** car images	 **A:** car images **J:** car images **T:** car images
Concept 617 **Direct:** N/A **Mean:** avian **Linear:** avian **MLP:** avian	 **A:** rare bird **J:** rare bird **T:** avian	 **A:** avian **J:** bird of prey **T:** maori hen	 **A:** avian **J:** golden eagle **T:** bird of prey	 **A:** rare bird **J:** rubythroated **T:** bird poop	 **A:** avian **J:** bald eagle **T:** bald eagle
Concept 478 **Direct:** N/A **Mean:** avian **Linear:** avian **MLP:** avian	 **A:** bird watching **J:** rare bird **T:** bird watching	 **A:** game bird **J:** shorebird **T:** game bird	 **A:** bird of prey **J:** golden eagle **T:** golden eagle	 **A:** bird watching **J:** shorebird **T:** sandpiper	 **A:** avian **J:** rare bird **T:** rubythroated
Concept 1006 **Direct:** N/A **Mean:** reptile **Linear:** reptile **MLP:** reptile	 **A:** snakeskin **J:** snakeskin **T:** snakeskin	 **A:** snakeskin **J:** snakeskin **T:** conch shell	 **A:** reptile **J:** reptile **T:** snakeskin	 **A:** edible frog **J:** reptile **T:** reptile	 **A:** reptile **J:** snakeskin **T:** snakeskin

Figure 56: **ImageNet-trained DINOv2 concepts named on PartImageNet (curated examples).** Each row shows the closest occurrence and four visually similar examples selected from the saved gallery. Fixed names appear at left, and the names below each image are **A**, additive, **J**, joint MLP, and gray **T**, teacher.