SUB-PATCH SEMANTICS IN DINOV3: GEOMETRY, RECOVERABILITY, AND FEATURE UPSAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Foundational models pretrained on large pool of unlabelled data has brought strong image representation across a wide range of domains. However, their semantically rich but low spatial resolution poses a challenge when working with pixel level dense prediction task such as segmentation or depth prediction. Existing work such as JAFAR, AnyUp try to recover this sub-patch information using the input image as a guide to upsample the ViT feature map. The upsamplers are trained to produce feature map are closer to the feature map as generated from a higher resolution image under the mean absolute error objective. However, a systematic study of what sub-patch information is already encoded within a patch token, how this information is geometrically organized, and to what extent it can be recovered directly from the coarser feature representation remains largely unexplored. In this work, we carry out a study of the geometric and semantic nature of the patch vectors. We find that patch embeddings are anisotropic, with distinct object representations occupying a shared, cone-like subspace where recovering the identity of objects in a mixed patch is challenging. We show that when different objects are present in the same patch, the resulting patch vector can't be explained as a linear decomposition of the mixed tokens. We show that the identity of the foreground object in such cases is still recoverable in a low-dimensional subspace. A Fisher-style criterion identifies such a subspace by selecting directions that separate foreground classes while remaining relatively stable across changes in background. Based on our observation, we propose an upsampler pre-trained on a labelled dataset that learns class prototypes and uses them to extract class-discriminative sub-patch information for coarse DINOv3 patch tokens without requiring an image-based guide that are used by models such as JAFAR or AnyUp. On semantic segmentation, our method improves linear-probe performance by approximately 2 mIoU over the raw DINOv3 tokens, by 1.2 mIoU over existing image-guided learned feature-upsampling baselines and outperforms transformer and DPT-based decoders.

1 INTRODUCTION

Self-supervised models that are pretrained on large pool of unlabeled data are now able to produce consistent representation that are able to produce state-of-the-art accuracy across various downstream tasks such as zero-shot or one-shot segmentation, depth estimation classification etc. However, to handle high-resolution image and large-scale training, ViTs divide images into smaller patches, producing one representation vector per patch. Most ViTs use patch size of 16, which means we get one feature vector for every 16×16 pixel region. This produces representations that are semantically rich but spatially downsampled by $16\times$, which is a practical constraint when pixel-level accuracy is required. Most methods solve for this issue by training deep CNN or transformer based decoder that mix neighbouring tokens and performs bilinear upsampling of feature maps to produce higher resolution segmentation maps. Another strategy is to bilinearly upsample the image before feeding to the model (Cuttano et al., 2026). While this produces a semantically rich, higher resolution feature maps, it increases the computation cost due to quadratic complexity of self-attention mechanism in vision transformers. Recently, a new class of feature upsamplers such as JAFAR (Couairon et al., 2026), AnyUp (Wimmer et al., 2026) try to recover this sub-patch finer information using the image as a guide to upsample the ViT feature map. The upsamplers

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.

are trained to produce feature maps that are closer to the feature map as generated from a higher resolution image under the mean absolute error objective. While such methods show that feature upsampling is possible such that the upsampled feature map is closer to the one obtained from encoders run on higher resolution image, a systematic study of what sub-patch information is already encoded within a patch token, how this information is geometrically organized, and to what extent it can be recovered directly from the coarser feature representation remains largely unexplored.

In this work, we carry out a study of the geometric and semantic nature of the patch vectors, largely focused on DINOv3 model (Siméoni et al., 2025) given its state-of-the-art accuracy on various downstream tasks. We define the majority class present in a patch as background class and thin or smaller objects present in a patch as foreground object. We study this question through ADE20K images and custom synthetic dataset. We first characterize how object identity and spatial position are represented across DINOv3 layers. We measure object discriminability and augmentation invariance via cosine similarity and subspace removal across different layers of the model. We move on to synthetic dataset where we have control of different object shapes and the patch vectors that uniquely identify a given shape and color. We verify our conclusions from the first experiment. Then, for the sub-patch extraction experiment, we simulate mixed patches where we set one shape as background object that occupies a few patches, and then we start mixing a foreground object occupying a small sub-patch (20 to 30%) of one of the patches. We try to recover the identity and location of the object under different occupancy levels and different object shapes. Lastly, based on these experiments, we build an hypothesis and propose an upsampler that achieves state-of-the-art segmentation accuracy numbers and we validate it on public segmentation dataset from different domains: ADE20K dataset (Zhou et al., 2017) and OpenEarthMap dataset (Xia et al., 2023).

Our analysis gives three main findings. First, we find that patch embeddings of DINOv3 are anisotropic, with distinct object representations occupying a shared, cone-like subspace, while class information is primarily encoded in deviations around it. We also find that the direction is more important than the magnitude of the vector itself. Second, through our study on synthetic objects, where vectors corresponding to different shapes, colors are known in prior, we see that when a small foreground object is inserted into a patch dominated by other class object, the resulting patch vector can't be explained as a linear combination of foreground and background patch vectors. Due to the presence of large shared subspace between different objects, the addition of foreground object changes the representation in a way that direct subtraction of background token or removal of a subspace of background-derived PCA directions doesn't recover the foreground and instead hurts the representation of both the foreground and background object. This entanglement of the foreground and background objects means that the image-guided upsamplers that rely on a linear combination of neighborhood patch vectors, can't reliably produce sub-patch vectors that can reveal the foreground objects present in that sub-patch. Third, we find that the identity of the foreground object added is recoverable in a low-dimensional subspace. This subspace is the same as that identified by Fisher's criterion where directions has high variance between different foreground objects and the variation in background classes are low.

Based on our learning, we propose an upsampler pre-trained on a labeled dataset that learns class prototypes and uses them to extract class-discriminative sub-patch information for coarse DINOv3 patch tokens without requiring an image-based guide that are used by models such as JAFAR or AnyUp. On semantic segmentation, our method improves linear-probe performance by approximately 2 mIoU over the raw DINOv3 tokens, by 1.2 mIoU over existing image-guided learned feature-upsampling baselines and further closes the gap with deep decoders such as DPT (Ranftl et al., 2021) or transformer based decoders while using a fraction of the parameters.

2 Related Work

There has been substantial work in understanding the representation learnt by self-supervised ViT models. For example, Cuttano et al. (2026) show that there are positional bias present in the representation of DINOv3 patch tokens that can limit accuracy of single-shot segmentation across different image. Their method estimates this positional structure in a low-dimensional subspace positional bias and remove its projection in feature space to achieve better cross image matching. Kerssies et al. (2025), who introduced the Encoder-only Mask Transformer (EoMT) model, found that ViT carry strong representation power and higher segmentation accuracy can be achieved by simply scaling

the ViT rather than adding architectural complexity such as those present in Mask2Former models. Zhang et al. (2023) and Zhang et al. (2024) find that although DINO models carry strong semantic features across different images, they are not very good at differentiating parts of the body such as left leg from right leg or left ear from right ear when compared to that of a diffusion model. However, there is a lack of such studies on the representation power of DINO based models on sub-patch objects which is a crucial requirement for fine-grained dense prediction tasks. Related analyses of representation geometry in contrastive models also show that learned spaces can contain structured offsets beyond task semantics (Liang et al., 2022).

Previous work in this field try to recover finer spatial information either via task specific decoder that learns how to upsample the image and mix nearby patch tokens or using a pre-trained upsampler whose objective is to create a higher resolution representation of the feature map. Another method is to create higher resolution feature map from bilinearly upsampled image and passing it to the foundational model backbone. Cuttano et al. (2026) upsample the image to produce higher resolution feature map. Another class of feature upsamplers such as JAFAR (Couairon et al., 2026) and AnyUp (Wimmer et al., 2026) use additional input from the RGB input image as a guide to upsample the feature map. These guides are available at much higher spatial resolution compared to the ViT encoder. These methods however, use a linear combination of neighbourhood patch which we find is not the best approach to recover finer details in a given patch. When foreground and background objects are both present in a patch, their information is entangled in the resulting token and can't be independently recovered. The identity is present but it is not the same as the original foreground object and similarity lies only in a smaller subspace. We therefore propose a method that learns different class prototype in a labelled dataset that learns to query objects within a patch.

3 METHOD

3.1 REPRESENTATION ANALYSIS

We study how semantic information is organised within DINOv3 patch tokens and how it changes when multiple objects occupy the same patch. Given a training-set mean feature g, we first centre the tokens by subtracting g and analyse the similarity and angular structure of class representations. We separately test the contribution of token direction and magnitude by replacing one while preserving the other. We also examine which feature directions are used by trained segmentation heads by retaining or removing principal directions of the patch representations. Full definitions and additional covariance analyses are given in Appendix A.

To study sub-patch objects, we construct matched background-only, mixed, and foreground-only images. These allow us to test whether the mixed representation can be explained by direct background subtraction, whether removing background-associated directions exposes the foreground, and whether the change produced by inserting the foreground contains information about its identity. The full subtraction, subspace-removal, and angular analyses are described in Appendix B.

3.2 BACKGROUND-STABLE DISCRIMINATIVE DIRECTIONS

Foreground identity can be difficult to recover from the full mixed representation because many feature directions also vary with the surrounding background. We therefore look for directions that distinguish foreground identities while remaining stable when the background changes. Let Σ_b denote the variation between foreground-class means, and let Σ_w denote the variation of the same foreground identity across different backgrounds, positions, and occupancies.

We select directions using the Fisher criterion

$$J(w) = \frac{w^\top \Sigma_b w}{w^\top \Sigma_w w + \lambda \|w\|^2}. \tag{1}$$

Directions with a high value of $J(w)$ separate foreground classes while being relatively insensitive to changes in the background. Details of the covariance construction, regularisation, and search space are provided in Appendix C.1.

3.3 PROTOTYPE-CONDITIONED SUB-PATCH UPSAMPLER

The representation analysis suggests that sub-patch identity is accessible from coarse DINOv3 tokens when the model reads a small set of class-discriminative, background-stable directions. We use this observation to design a feature upsampler that predicts fine-resolution DINOv3 tokens directly from coarse features, without using the RGB image as an additional guide. Given an $h \times w$ feature map from a frozen DINOv3 ViT-L/16, the upsampler predicts a $4h \times 4w$ feature map in the same 1024-dimensional feature space. Each parent patch therefore produces $4 \times 4 = 16$ child tokens.

Context and child queries. We use features from blocks 6, 13, 18, and 24 of DINOv3. For each parent patch, we collect a 5×5 neighbourhood from all four layers. The features from each layer are projected to 384 dimensions and processed by a two-layer context transformer. A two-layer decoder then produces 16 child queries from the parent representation, with one query corresponding to each sub-patch location.

Evidence selection. For each child query, attention over the local block-24 neighbourhood produces an evidence vector v in the original DINOv3 feature space. We project this evidence onto a set of 128 discriminative directions,

$$a = W(v - g_{\text{in}}), \qquad W \in \mathbb{R}^{128 \times 1024}, \tag{2}$$

where W is initialised using the Fisher criterion described above and is subsequently optimised during training. A child-dependent gate selects the top $K = 16$ coordinates from this representation. The semantic prediction for each child is therefore made using only a small subset of the available feature directions. Implementation details of the gating mechanism are given in Appendix C.2.

Prototype routing. The model maintains multiple DINOv3 prototypes for each semantic class. These prototypes are computed from high-resolution ground-truth child tokens and are updated as exponential moving averages after each optimisation step. The prototype memory is fixed during evaluation. For each child, the router compares the selected semantic coordinates with the class prototypes. The most relevant prototypes are combined to obtain a prototype representation $\bar{P}$. The router also uses the child position and local context, while an image-level presence prediction provides a prior over which classes are likely to occur in the image. Further details of prototype construction and updating are provided in Appendix C.2.

Semantic and detail paths. The output separates semantic reconstruction from the remaining feature content. The prototype branch writes only into the K directions selected for a given child. The complementary subspace is initialised from bilinearly upsampled DINOv3 features and can be refined by a learned detail branch. This prevents the unrestricted reconstruction path from overwriting the directions used for the semantic prediction. Let m denote the binary mask selecting the semantic directions, and let Q denote the corresponding write basis.

The final child token is

$$\hat{z} = g_{\text{out}} + (I - \Pi_K)\,(z_{\text{bil}} - g_{\text{out}} + \beta d) + Q(m \odot c), \tag{3}$$

where

$$\Pi_K = Q\,\text{diag}(m)Q^\top. \tag{4}$$

Here, z_{bil} is the bilinearly upsampled feature, c contains the prototype-conditioned semantic coordinates, d is a learned detail correction, and β controls the strength of the detail branch. The projection $I - \Pi_K$ prevents the detail branch from writing into the directions used for the semantic decision. The construction of Q, the selected-direction mask, and the detail path are described in Appendix C.2.

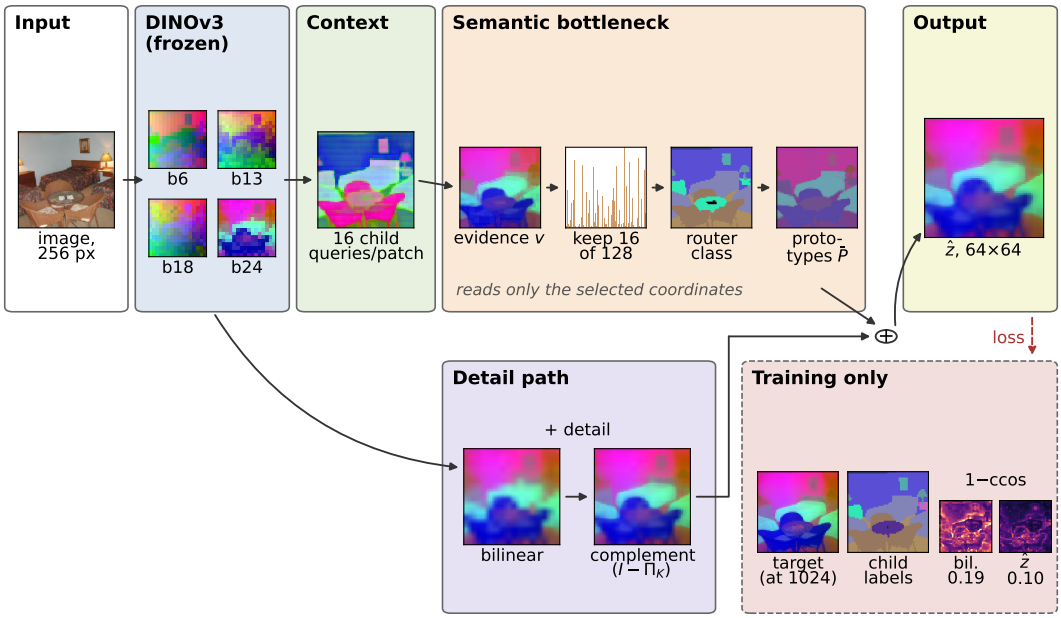

Figure 1: Prototype-conditioned feature upsampler. Features from four frozen DINOv3 blocks are aggregated over a local neighbourhood to form 16 child queries per parent patch. Each child selects a small subset of discriminative feature directions and routes them to class prototypes. The semantic branch writes only into the selected directions, while a separate detail branch reconstructs their complement. The upsampler receives DINOv3 tokens only; high-resolution features and semantic labels are used during training.

Training objective. The input to the upsampler is obtained by running DINOv3 on the image at 256×256 resolution. The target is the block-24 feature map obtained by running the same frozen DINOv3 encoder on the image at 1024×1024 resolution. Training combines feature reconstruction with prototype-routing and class-discrimination objectives. The individual losses and their coefficients are given in Appendix C.3.

4 Experiments

Backbone and datasets. All experiments use a frozen DINOv3 ViT-L/16 (Siméoni et al., 2025) pretrained on LVD-1689M, with 1024-dimensional patch tokens and 24 transformer blocks. Natural-image representation experiments and the main segmentation benchmark use ADE20K (Zhou et al., 2017). Segmentation is evaluated on the full ADE20K validation split against the original-resolution annotations using sliding-window inference.

Controlled sub-patch experiments. We use both natural-image compositions and synthetic data to study what information is retained when multiple objects occupy the same patch.

For the natural-image experiments, an object crop is placed in one of the four 8×8 quadrants of a 16×16 DINOv3 patch. Source objects and background images used for evaluation are held out from those used for fitting the probes. For the synthetic experiments, we render objects with controlled shape, colour, scale, and position. To study mixed representations, a small foreground object covering approximately 15-25% of the target patch is placed over a larger background object. The foreground-recovery experiments use 15 foreground identities and disjoint sets of background shape-colour combinations for training and evaluation. All class prototypes, subspaces, and Fisher directions are fitted using training mixtures only and are evaluated on previously unseen background combinations. Additional dataset construction and sampling details are given in Appendix B.

Feature upsampling. The prototype-conditioned upsampler described in Section 3.3 is trained on ADE20K.

Its input consists of DINOv3 features obtained from an image at 256×256 resolution, while the target consists of block-24 features obtained by evaluating the same frozen backbone at 1024×1024 resolution.

The resulting 16×16 input feature map is upsampled to 64×64.

We compare against bilinear interpolation, AnyUp (Wimmer et al., 2026), and JAFAR (Couairon et al., 2026).

JAFAR and AnyUp use the RGB image as an additional guidance signal, whereas our model receives only the coarse DINOv3 features.

Evaluation. We evaluate the upsampler at both the feature and downstream-task levels. For feature reconstruction, we compare each predicted child token with the corresponding DINOv3 token obtained from the 1024×1024 image. We report centred cosine similarity and relative ℓ_2 error. We additionally report centred cosine on *minority cells*, defined as high-resolution cells whose majority semantic class differs from that of their coarse parent patch. For downstream evaluation, we train the same linear segmentation head on four representations: the original DINOv3 tokens, JAFAR-upsampled features, our upsampled features, and DINOv3 features obtained by running the backbone at 1024×1024 resolution. This last setting provides a reference for the accuracy obtained by increasing the encoder input resolution directly. We further evaluate whether the predicted features improve a multi-layer DPT decoder (Ranftl et al., 2021) by replacing bilinear upsampling of its finest DINOv3 feature map with our upsampler. We also evaluate cross-domain transfer on OpenEarthMap (Xia et al., 2023). Complete training recipes and implementation details are given in Appendix C.4.

5 RESULTS

We first ask whether information below the native patch resolution is present in DINOv3 tokens and how this information is represented when multiple objects occupy the same patch. We then test whether the foreground can be isolated from the dominant patch content, whether a smaller set of feature directions makes its identity easier to recover, and finally whether these observations can be used for feature upsampling.

5.1 COARSE TOKENS CONTAIN SUB-PATCH LOCATION AND IDENTITY

DINOv3 patch representations contain a large shared component. Across 29 ADE20K classes, the mean cosine between raw class means is 0.943, while centring the features reduces it to -0.004. Class discrimination also depends much more strongly on token direction than magnitude: replacing each token norm by a common value changes linear-probe mIoU only from 41.50 to 41.31, whereas replacing the token direction reduces it to 0.14. The full cone geometry, direction–magnitude interventions, PCA analysis, and evolution across transformer depth are reported in Appendix A.

We next test whether a single patch token contains information about structures below its 16×16 spatial resolution. When the same natural object is placed in one of four 8×8 quadrants of a patch, a linear probe predicts its location with accuracy 0.631, against a chance level of 0.25. Subtracting the mean token over the four placements increases accuracy to 0.936, while texture-scrambled objects reach 0.931. The location signal therefore remains accessible even when object texture is removed. Location and object-to-location binding are strongest in intermediate layers, peaking around block 7 (Figure 2); the complete location experiment is reported in Appendix B.1.

The token also carries information about the identity of a small foreground object. For foregrounds covering approximately 15–25% of a patch, the block-13 mixed token retrieves the correct foreground among 15 identities with Recall@1 of 0.800. However, the margin over the strongest competing identity is only 0.009, indicating that foreground identity is present in the representation but weakly separated from competing identities.

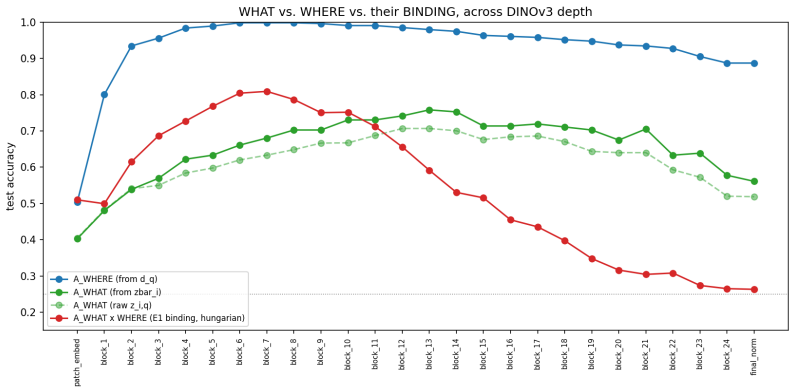

Figure 2: Within-patch location (WHERE), identity (WHAT), and their binding across depth for natural crops pasted into one quadrant of a patch. The y-axis reports held-out accuracy, with chance at 0.25. Location and the binding of a fixed inventory of objects to positions are strongest in intermediate layers and peak around block 7.

5.2 Foreground information is entangled with the background

We first test whether the foreground contributes an additive component to the mixed token. If this were the case, subtracting the corresponding background-only token should recover a representation closer to the isolated foreground. Instead, at block 13, exact subtraction reduces Recall@1 from 0.800 to 0.533.

The displacement caused by inserting the foreground is nevertheless identity-dependent. At block 7, displacements produced by the same foreground on different backgrounds have cosine 0.875, compared with 0.745 for different foreground identities. The same displacement has cosine only 0.648 with the isolated foreground prototype. The insertion therefore produces a repeatable identity-dependent change, but this change is not equivalent to adding the isolated foreground representation.

Removing background-associated directions gives a similar result. At block 13, removing the PCA subspace of the matched background gives Recall@1 of 0.673, compared with 0.667 when the subspace is obtained from an unrelated background, 0.787 for random removal, and 0.800 without removal. The matched background basis is therefore no more effective than an unrelated one at isolating the foreground.

The reason is that background variation and identity information occupy many of the same directions. At block 6, the 32 directions containing 98.6% of the within-identity variation also contain 97.8% of the between-identity variation. Removing these directions therefore removes semantic information together with background variation. Additional rank-one, rank-32, and layerwise removal results are provided in Appendix B.2.

We additionally examine insertion as an angular displacement of the background representation. The displacement direction carries foreground identity but is only partly aligned with the direction toward the isolated foreground. A Fisher readout on these displacements reaches high held-out retrieval, but applying the same type of readout directly to the mixed token performs at least as well and does not require a background-only image at inference. We therefore use the mixed token in the remaining experiments. The full angular and tangent analysis is given in Appendix B.3, with the occupancy analysis in Appendix B.4.

5.3 Foreground identity is recoverable in background-stable directions

The previous experiments show that simply removing background-associated directions is ineffective because those directions also contain foreground information. We therefore test the complementary strategy of retaining directions that distinguish foreground identities while remaining stable when the background changes.

Table 1: Held-out 15-way foreground Recall@1 on unseen background combinations. Every fitted readout keeps 14 dimensions. Fisher estimates within-class variation from mixed tokens across backgrounds.

Layer	Raw mixed	Between PCA	Fisher
Block 1	0.213	0.180	**0.693**
Block 6	0.200	0.213	**0.927**
Block 9	0.647	0.680	**1.000**
Block 13	0.787	0.827	**1.000**
Block 18	0.720	0.707	**0.860**
Final norm	0.560	0.540	**0.680**

Using 14 Fisher-selected directions, held-out Recall@1 at block 6 increases from 0.200 for the full mixed token to 0.927. The leading between-class PCA directions reach only 0.213 under the same 14-dimensional constraint. Fisher retrieval reaches 1.000 at blocks 9–14 and remains above the raw-token result at later layers (Table 1).

The improvement comes primarily from separating competing identities. At block 6, cosine similarity to the correct prototype changes only from 0.836 to 0.829, while the margin over the strongest competitor increases from -0.056 to $+0.242$. The complete layerwise Recall@1, cosine, and margin curves are provided in Appendix B.5.

Selecting informative directions is also more effective than deleting nuisance-like directions. At block 13, removing 24 directions with high within-class and low between-class variation removes 7.9% of the within-class scatter but leaves Recall@1 at 0.827, compared with 1.000 when 14 discriminative directions are retained.

We finally test whether the supervised fitting procedure can produce a similar result for arbitrary labels. Incorrect foreground labels produce above-chance structure on the fitting samples, but their held-out Recall@1 falls to 0.069–0.084. With the correct labels, held-out Recall@1 remains between 0.813 and 1.000. The recovery therefore transfers to unseen samples only when the assigned identity corresponds to information present in the patch. The full wrong-label capacity sweep and foreground-versus-background specificity controls are reported in Appendix B.5.

5.4 Prototype upsampling improves dense features and segmentation

The representation analysis gives two observations that guide the upsampler. First, sub-patch position and identity are already accessible from coarse DINOv3 tokens, with several of these signals strongest in intermediate layers. Second, foreground identity becomes substantially easier to recover when the readout is restricted to a small set of background-stable directions. Prototype upsampling incorporates these observations by using features from multiple DINOv3 layers and making each child's semantic prediction through Fisher-initialised directions.

We first compare the predicted features with DINOv3 features obtained by evaluating the same image at 1024×1024 resolution. Prototype upsampling reaches a centred cosine of 0.874, compared with 0.750 for bilinear interpolation, 0.750 for AnyUp, and 0.778 for JAFAR (Table 2). The difference is larger on minority cells, where the high-resolution cell contains a different majority class from its coarse parent patch: centred cosine increases from 0.718-0.759 for the baselines to 0.840 with prototype upsampling. Relative ℓ_2 error also decreases to 0.454, compared with 0.593 for JAFAR.

The reconstructed features also improve semantic segmentation. On ADE20K, a linear probe on raw DINOv3 tokens reaches 54.99 mIoU, JAFAR-upsampled features reach 55.51, and prototype-upsampled features reach 56.70. DINOv3 tokens obtained by evaluating the image directly at 1024×1024 reach 56.48. Inside a DPT decoder, replacing bilinear upsampling of the block-24 features with prototype upsampling increases mIoU from 56.51 to 57.12. Qualitative comparisons are provided in Appendix D.1.

The ADE20K-trained upsampler also transfers to OpenEarthMap. Prototype-upsampled features reach 62.00 mIoU, compared with 56.00 for the raw tokens and 59.00 for JAFAR, and match the

Table 2: Feature upsampling on 500 ADE20K validation images. 16×16 DINOv3 tokens from the image at 256 pixels are upsampled to 64×64 and compared with DINOv3 features obtained from the image at 1024 pixels. Minority cells are high-resolution cells whose majority class differs from that of their coarse parent patch.

Method	Guidance	Centred cos ↑	Minority cells ↑	Rel. L2 ↓
Bilinear	–	0.750	0.718	0.637
AnyUp	image	0.750	0.729	0.623
JAFAR	image	0.778	0.759	0.593
Prototype upsampling	none	**0.874**	**0.840**	**0.454**

Table 3: Semantic segmentation on frozen DINOv3 ViT-L/16, reported as mIoU percentages. ADE20K uses the full 2000-image validation set with no test-time augmentation. Within each ADE20K comparison group, the heads use the same training recipe.

Head	Features	ADE20K	OpenEarthMap
Linear	raw tokens	54.99	56.00
Linear	JAFAR	55.51	59.00
Linear	prototype upsampling	**56.70**	**62.00**
Linear	tokens of the image resized to 1024	56.48	**62.00**
DPT	3 block updates + block 24, bilinear	56.51	–
DPT	3 block updates + block 24, prototype upsampling	**57.12**	–

62.00 mIoU obtained by evaluating DINOv3 at the higher input resolution. This evaluates the upsampler outside the dataset on which its parameters and prototypes were learned. OpenEarthMap dataset and evaluation details are provided in Appendix D.2.

6 Conclusion

We studied how information below the native patch resolution is represented in frozen DINOv3 features.Our experiments show that a coarse patch token retains both the location and identity of objects that occupy only a fraction of the patch. These signals are embedded in a highly shared feature geometry, where class information is carried primarily by directional differences around a large common component. When a small foreground object is introduced into a patch dominated by another object, its information becomes entangled with the existing representation.Exact background subtraction does not recover the isolated foreground, and removing background-derived directions can also remove information that distinguishes foreground identities. The change caused by foreground insertion is identity-dependent, but it does not correspond to a clean foreground vector that can be separated from the background representation. Foreground identity becomes much easier to recover when the representation is restricted to directions that distinguish foreground classes while remaining stable across changes in background. These observations motivate prototype upsampling, which uses intermediate DINOv3 features and a small discriminative semantic readout to predict finer-resolution features without using the RGB image as an additional guide. On ADE20K, the predicted features are closer to those obtained by evaluating DINOv3 at higher input resolution, reaching a centred cosine of 0.874 and 0.840 on minority cells. They improve linear-probe segmentation from 54.99 to 56.70 mIoU and improve a DPT decoder from 56.51 to 57.12 mIoU when used in place of bilinear feature upsampling.

AI use statement

In this work, we used generative AI tools for Generate synthetic data set: we used simple synthetic shapes and colors such as rectangle, diamond, circles that were created with the help of AI; propose and refine hypothesis - to link our observations on Fisher's criterion and subspace removal with the proposed upsampler, we used AI tools to propose objectives that aligns with our findings; design or provide feedback on research methodology or experiments - experiments and observations were also verified using AI tools, AI tools were also used to suggest next set of experiments that can be

used to fill the gaps in the current experiments; implement methods: we used AI tools to implement repetitive model changes for which the structure of the code was already known; interpret results - AI tools were used to interpret and verify our interpretations of the results.

We have not used generative AI tools for help develop theoretical models or conceptual frameworks; formulate mathematical claims; provide critical ingredients for proving mathematical claims; assist in the writing of proofs - we are not proving theoretical or mathematical claims, assist with translation; clean and reformat dataset; and support qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for create or modify scientific figures or images; suggest experimental parameters; create or edit software code; draft parts of a research paper; summarize or analyse existing literature and discover research topics or identify gaps. We have reviewed all AI-assisted work. The actual versions of all the code were written manually by the authors. The modifications were done by AI models but all the changes were reviewed. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

The results of this work are reproducible completely through the supplementary materials. We use uv for handling dependencies and therefore there is no need to create Docker environment. The model checkpoints are uploaded to Hugging Face and the code provided automatically downloads them. The results in different sections of the results are divided into different scripts. The images present in the supplementary materials are a subset of the images for quick testing. The full ADE20K dataset can be downloaded from https://www.kaggle.com/datasets/awsaf49/ade20k-dataset and the OpenEarthMap dataset can be downloaded from https://www.kaggle.com/datasets/aletbm/global-land-cover-mapping-openearthmap .

References

Paul Couairon, Loick Chambon, Louis Serrano, Jean-Emmanuel Haugeard, Matthieu Cord, and Nicolas Thome. Jafar: Jack up any feature at any resolution. *Advances in Neural Information Processing Systems*, 38:99265–99292, 2026.

Claudia Cuttano, Gabriele Trivigno, Christoph Reich, Daniel Cremers, Carlo Masone, and Stefan Roth. Insid3: Training-free in-context segmentation with dinov3. In *Third Workshop on Visual Concepts*, 2026.

Tommie Kerssies, Niccolo Cavagnero, Alexander Hermans, Narges Norouzi, Giuseppe Averta, Bastian Leibe, Gijs Dubbelman, and Daan De Geus. Your vit is secretly an image segmentation model. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25303–25313. IEEE, 2025.

Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in neural information processing systems*, 35:17612–17625, 2022.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12159–12168. IEEE, 2021.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. Anyup: Universal feature upsampling. In *International Conference on Learning Representations*, volume 2026, pp. 140700–140720, 2026.

Junshi Xia, Naoto Yokoya, Bruno Adriano, and Clifford Broni-Bediako. Openearthmap: A benchmark dataset for global high-resolution land cover mapping. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6243–6253. IEEE, 2023.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa Polania Cabrera, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence. *Advances in Neural Information Processing Systems*, 36:45533–45547, 2023.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Eric Chen, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. Telling left from right: Identifying geometry-aware semantic correspondence. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3076–3085. IEEE, 2024.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 5122–5130. IEEE, 2017.

A Representation geometry

A.1 Cone geometry and anisotropy

We distinguish three coordinates for a patch token: the raw residual vector $r \in \mathbb{R}^{1024}$, its output f after the learned final LayerNorm of DINOv3, and the BatchNorm output z inside the linear head. LayerNorm standardises each token across channels and then applies a learned affine map, so it does not produce unit-length vectors. Given a training mean g, centring separates the component shared by all tokens from deviations around it. We summarise the geometry with the cosine between class means, the angle of each class mean to their resultant direction, the covariance eigenspectrum, and the participation ratio $\mathrm{PR}(C) = (\operatorname{tr} C)^2 / \operatorname{tr}(C^2)$, which counts how many directions carry appreciable variance. Finite-sample isotropic Gaussians provide a reference for each statistic.

Among 29 ADE20K classes with enough support, the mean pairwise cosine between raw class means is 0.943, and the median angle between a class mean and their resultant is $13.5°$. After subtracting the training mean, the average class-mean cosine drops to -0.004. The class means therefore sit in a narrow cone around a common component g, and the class-specific deviations δ_c in $\mu_c = g + \delta_c$ point in nearly unrelated directions. A single common direction accounts for 95.3% of the squared norm of the image-mean tokens, with little variation in its amplitude across images (coefficient of variation 0.042).

Outside this common component, the variance is spread unevenly. The final-layer token covariance has a participation ratio of 115.7 out of 1024 channels, against about 970 for an isotropic Gaussian with the same sample size, and 349 principal components are needed for 90% of the centred variance. The centred class means are more concentrated, with a participation ratio of 14.0 and 90% of their variance in 18 dimensions. Tests on independent images reject the isotropic reference for the spectrum, the norms and the angles.

A.2 Direction versus magnitude interventions

To test how much a trained head depends on token magnitude, we replace a feature x by $r_0 x/\|x\|$, where r_0 is the mean norm on a calibration set; to test its dependence on direction, we replace x by $\|x\|u$, where u is the calibration-mean direction. All later operations are left unchanged. Applying the interventions to raw, final-LayerNorm and post-BN coordinates separates information in the feature from effects of the normalisation layers.

Table 4 reports the results with the frozen linear head. Keeping each post-BN token's direction and setting its norm to the calibration mean changes mIoU from 41.50 to 41.31, whereas keeping the norm and replacing the direction with the mean direction reduces it to 0.14. Rescaling to unit length after the final LayerNorm (0.59) or after BatchNorm (8.09) is harmful, because it moves the features away from the scale on which the head was calibrated. Since BatchNorm acts per channel and can be folded into the 1×1 classifier, each class score is an affine function of the token, and the decision depends on which class direction the token aligns with best.

Table 4: Direction versus magnitude in the 96-image frozen-head study (mIoU). The common norm is the calibration-mean norm, not unit length. Raw-space interventions precede the final LayerNorm, which largely absorbs rescaling.

Intervention	Raw residual	Final LayerNorm	Post-BN
Unmodified	41.50	41.50	41.50
Preserve direction, common norm	41.50	40.63	41.31
Preserve norm, common direction	0.14	0.14	0.14
Unit norm	41.43	0.59	8.09

A.3 Image PCA and decoder readouts

To ask which directions a head reads, we keep or delete leading principal components within each image. For image i with token mean μ_i and an orthonormal basis V_k of k directions,

$$f_{ip}^{\text{keep}} = \mu_i + V_k V_k^\top (f_{ip} - \mu_i), \qquad f_{ip}^{\text{drop}} = \mu_i + (I - V_k V_k^\top)(f_{ip} - \mu_i), \tag{5}$$

for patch p. We compare image-specific PCA, PCA fitted on the training set, and random bases of the same rank. A DPT head receives the projection independently at each of its input layers.

Table 5 reports the interventions. Keeping three components gives about 25.8 mIoU for both the linear head and a four-layer DPT head, against 6.0 and 7.4 for random rank-three projections. Keeping ten gives 36.97 and 36.51, and deleting the same ten leaves 11.33 and 13.79.

Table 5: Per-image PCA interventions on 200 images at 512^2 (mIoU). The image mean is kept; the DPT input is projected independently at each layer. These diagnostic scores use an earlier linear checkpoint and are separate from the native benchmark.

Input intervention	Linear probe	Four-layer DPT
Full features	42.60	48.75
Keep 3 image PCs	25.82	25.84
Random rank 3	6.02	7.35
Keep 10 image PCs	36.97	36.51
Drop 10 image PCs	11.33	13.79
Keep 50 image PCs	42.94	46.97

These directions are specific to each image. In the 96-image linear-head study, ten image components keep 64.6% of the feature variance, 72.1% of the variance of class-contrast scores, and 32.4 of the original 41.5 mIoU, while ten components fitted on the whole training set keep only 8.9 mIoU (41.1 versus 15.8 at rank 50; bootstrap interval of the rank-10 difference [18.0, 24.7]). The leading dataset components are strongly predictable from grid position. For ten image components, the mean principal angle is 24.3° to the within-image class-mean contrasts and 72.6° to the contrasts between classifier weights. A direction can therefore matter for the predictions because the image varies along it, even when it carries little of the classifier's weight energy.

The DPT head adds multi-layer input and spatial decoding. On this diagnostic set it gains 6.15 points over the linear head (42.60 versus 48.75). On a separate 500-image diagnostic, boundary F1 rises from 40.49 to 55.79 while mIoU rises from 46.98 to 51.21, so a large part of the DPT gain is spatial precision.

A.4 Evolution across transformer depth

Across transformer blocks, hooks record $r_{\text{out}} = r_{\text{in}} + a + m$, where a and m are the attention and MLP updates; probes trained separately on each state measure how accessible class information is, and fixed-probe transfer measures how much a block changes compatibility with an existing readout.

A synthetic bank of large shape–colour objects shows how the similarity structure changes with depth (Figure 3; x-axis: block, y-axis: mean prototype cosine for identity pairs that share colour, share shape, or differ in both). Through block 12, prototypes of different shapes in the same colour

have cosine close to one. In later blocks, pairs with the same shape and different colours become the most similar, reaching 0.987 at the final norm against 0.820 for unrelated pairs. Applying the final LayerNorm at every block keeps this pattern. For small foregrounds the ordering changes earlier: in the Fisher readout of Section 5.3, the cosine of same-colour distractors is 0.755 at block 1 and -0.147 at the final norm, while that of same-shape distractors goes from -0.295 to 0.663, with the two crossing around blocks 5–6. Middle blocks balance the two attributes, which matters when identities differ in only one of them.

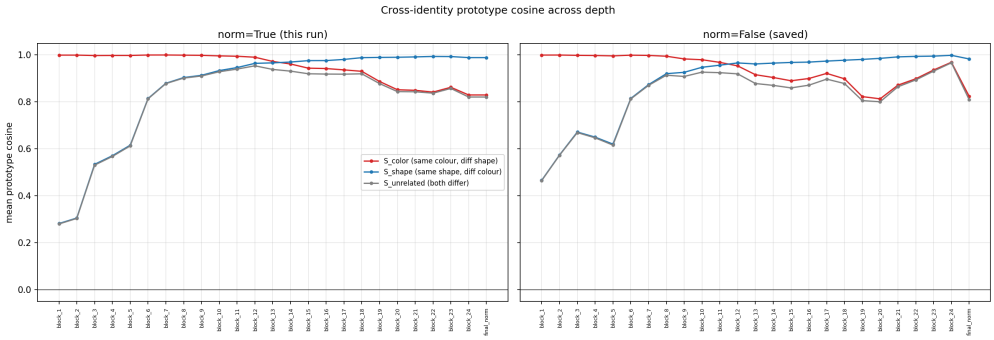

Figure 3: Cross-identity prototype cosine across depth, with the final LayerNorm applied at every block (left) and without it (right). Early blocks group identities by colour and later blocks by shape; the shared component remains in both.

Probes trained on the residual stream show that class information accumulates with depth: macro recall on 20 supported classes rises from 0.175 at block 1 to 0.560 at block 13 and 0.779 at block 24. The stream entering block 24 already reaches 0.758 with its own probe, yet a probe trained on the output of block 24 scores only 0.258 on that incoming stream and 0.738 after the final attention update. In a separate alignment test, an incoming-state probe scores 40.21% on its own state and 10.72% on the outgoing state; aligning the two coordinate systems by mean, orthogonal or affine maps recovers 24.17%, 37.59% and 38.22%, that is, 93% of the transfer loss with the affine map. Removing the final attention update reduces eight-image mIoU from 0.403 to 0.104. The last block therefore reorganises information that is largely present in its input.

B Sub-patch diagnostics

B.1 Location experiment

Object crops are pasted into one of the four 8×8 quadrants of a 16×16 patch, with source instances and backgrounds held out at evaluation. Four-view centring subtracts the mean token over the four placements of the same crop. Texture-scrambled crops, shuffled labels and a fixed-inventory what–where binding task separate location signals from recognition. A linear probe on the final token predicts the quadrant with accuracy 0.631 (chance 0.25), 0.936 after four-view centring, and 0.931 for texture-scrambled crops. A layer-wise version peaks near block 7 (0.996 for the raw token and 0.999 after four-view centring, against 0.560 and 0.900 at the final norm). In the binding task, a fixed set of four classes is placed in the four quadrants, so only the assignment of classes to quadrants carries information; it also peaks around block 7 (Figure 2).

B.2 Rank-one and rank-32 background removal

The synthetic studies vary shape, colour and scale. Large-object prototypes probe the shared geometry, and pixel-matched background, mixed and foreground-only images test additive recovery. Test backgrounds share the rendering family and the primitive vocabulary. Later synthetic studies apply the final LayerNorm at every block, and the earlier subtraction studies also report raw residuals. Subspace removal uses the matched background, a wrong background, a pooled set or random directions as the basis B (a single normalised prototype in the rank-one case); queries and prototypes receive the same projection.

Rank-one removal shows why removal fails: removing the prototype direction of a red triangle leaves only 5.3% of the norm of a red-circle prototype at block 7. In the large-object bank, the 32 directions with the most within-identity variation at block 6 contain 98.6% of the within-identity variance and also 97.8% of the between-identity variance. Subtraction is not uniformly harmful at every layer: at block 24, plain subtraction gives the best retrieval among the unlearned readouts of the angular study (0.676 against 0.471 for the mixed token).

B.3 Angular and tangent displacement

Because the final LayerNorm places tokens on a sphere before its affine map, we describe insertion as a rotation away from the background token and measure its angle and direction separately. We normalise each representation to a unit vector u, using learned LayerNorm outputs y, centred outputs $y-g$, and masked pre-affine outputs $M[(y-\beta)\oslash\gamma]$, where γ, β are the LayerNorm parameters, $\oslash$ is coordinatewise division, and the mask M removes channel 757, the only channel with $|\gamma_d| < 0.01$, before normalisation. The rotation angle and the tangent displacement are

$$\theta = \arccos(u_{\mathrm{bg}}^{\top} u_{\mathrm{mix}}), \qquad \delta = \frac{\theta}{\sin\theta}\big(u_{\mathrm{mix}} - \cos\theta\, u_{\mathrm{bg}}\big), \tag{6}$$

so δ is orthogonal to u_{bg}, its norm equals θ in radians, and $\hat{\delta} = \delta/\|\delta\|$ gives its direction. Replacing u_{mix} by a pure-foreground prototype u_c gives the target displacement t_c. Learned displacement prototypes average training δ vectors and are projected onto the tangent plane of each test background. The angular study uses 450 training and 225 held-out background/mixed pairs plus 450 pure-foreground tokens, with the same disjoint background pools as the Fisher study. Tangent readouts need the background-only image at test time.

The direction of the rotation depends on identity and is also shared across identities. At block 7, rotation directions for the same identity at different occupancies have mean cosine 0.819, against 0.700 for different identities at the same occupancy; at block 13 the values are 0.855 and 0.710. The high cosine between different identities indicates a substantial common response to insertion, and the gap between the two values indicates an identity-specific part. The rotation is also only partly directed toward the pure foreground: the share of the displacement not explained by the direction toward the correct prototype is 0.27–0.51 in pre-affine coordinates, against 0.46–0.67 for wrong prototypes.

The subtraction test compared a displacement with prototype points. Comparing it with the direction from the background toward each prototype removes the shared background component from the comparison. With these targets the chord $u_{\mathrm{mix}} - u_{\mathrm{bg}}$ and the tangent displacement δ give identical rankings, because their difference is a multiple of u_{bg}, which is orthogonal to every target direction. The improvement over the subtraction test comes from comparing a movement with a movement. Prototypes learned from training displacements also transfer to unseen backgrounds: in pre-affine coordinates they reach Recall@1 0.618 and 0.636 at blocks 6 and 7, against 0.400 and 0.418 for prototypes learned from the mixed tokens themselves, although this advantage is not consistent after centring (Figure 4).

Fisher selection (Appendix C.1) applied to tangent displacements with eight directions raises pre-affine Recall@1 on unseen backgrounds to 0.876 at block 6 and 0.898 at block 7, and to 0.849 (pre-affine) and 0.933 (centred) at block 13. The same procedure applied to the mixed tokens reaches 0.960 at block 6 and 1.000 at block 13 (Figure 4). Displacements help to describe how the token changes, but selecting directions in the mixed token recovers identity at least as well and does not need the background-only image at test time.

B.4 Occupancy study

A separate sweep crosses the 15 identities with six target occupancies (5–60%) on three fixed backgrounds. We report marginal η^2 of the rotation angle grouped by occupancy or by identity; these values do not partition background or interaction effects. Figure 5 shows, from left to right, the rotation angle against foreground occupancy for five blocks, the fraction of angle variance explained by occupancy or by identity at each block, and the mean cosine between rotation directions for the same identity at different occupancies versus different identities at the same occupancy. At block 6

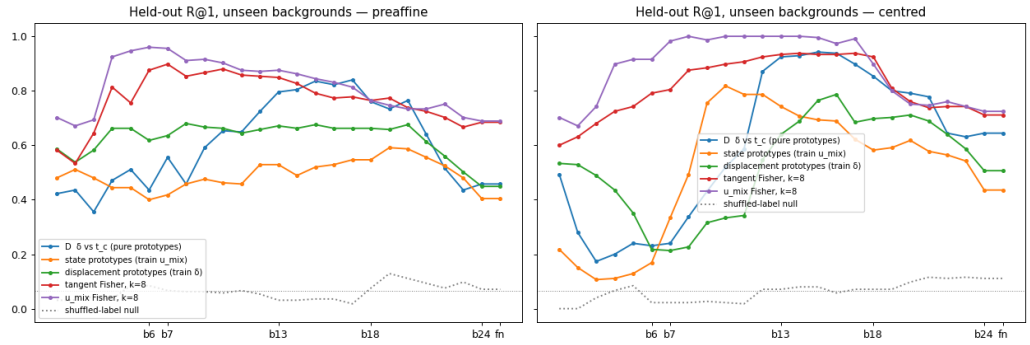

Figure 4: Held-out 15-way Recall@1 on unseen backgrounds for learned readouts of displacements and mixed tokens, in pre-affine (left) and centred (right) coordinates. Fits use 450 training pairs and 225 held-out pairs; the dotted line shuffles the displacement-prototype labels. Tangent readouts require background-only images.

the angle grows from $39.4°$ to $53.8°$ as occupancy rises from 5% to 60%. At block 13 it stays between $63.1°$ and $63.2°$ at the two ends, and all six levels lie within $1.6°$; occupancy explains 0.002 of the angle variance there and identity explains 0.314. In middle and late blocks, the size of the rotation therefore depends much more on which object was inserted than on how much of the patch it covers, within the tested range.

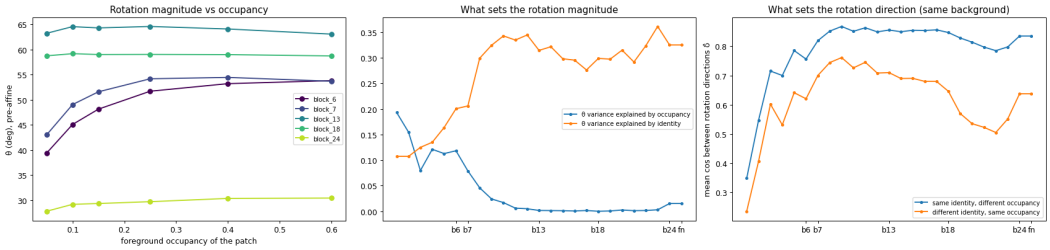

Figure 5: Rotation caused by inserting a foreground (masked pre-affine, unit-normalised features). Left: rotation angle versus the foreground's share of the patch. Centre: marginal η^2 of the angle for occupancy and for identity across blocks. Right: mean cosine between rotation directions for the same identity across occupancies and for different identities at equal occupancy, on a fixed background.

B.5 Fisher readout curves and wrong-label capacity sweep

Figure 6 shows Recall@1, the cosine to the correct prototype and the margin over the strongest competitor across all blocks for the 14-direction Fisher readout of Section 5.3.

Two controls test whether the recovered identity reflects information in the patch. Foreground- versus background-fitted bases and background-only images test specificity: at block 13, a basis fitted on foreground labels retrieves the foreground with 1.000 and the background with 0.093, while a basis fitted on background labels gives 0.360 and 0.933. Five wrong-label assignments at search ranks 16, 64 and 256, each keeping 14 directions, test how much structure the fitting procedure can create from labels alone. Leave-one-out scoring removes a token from its prototype but not from basis fitting; held-out tokens are excluded from all fitting. With a large enough search space, wrong labels produce structure on the training tokens (leave-one-out Recall@1 0.578 at the final norm with search rank 256), but on held-out tokens their retrieval stays near chance (0.069–0.084) with negative margins, while true labels transfer (0.813–1.000; Table 6). Background-only images also match the assigned foreground at about chance (median 0.073 across layers).

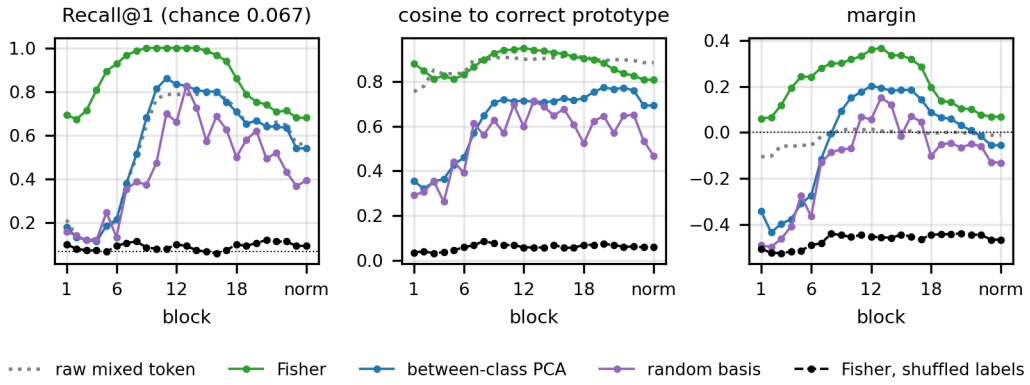

Figure 6: Recall@1, cosine to the correct prototype, and margin over the strongest competitor across blocks for held-out mixed tokens on unseen backgrounds. Recall and margin improve with the Fisher projection even where the correct-prototype cosine does not rise. Prototypes and directions are fitted on training backgrounds only; the dashed line fits the same Fisher readout with shuffled labels.

Table 6: True versus deliberately wrong labels. Leave-one-out (LOO) removes a token from its prototype but not from basis fitting. Wrong-label values average five assignments. All fits keep 14 directions.

	Search	True label		Wrong label	
Layer	rank	LOO	Held-out	LOO	Held-out
Block 6	64	0.964	0.927	0.180	0.076
Block 13	64	1.000	1.000	0.230	0.069
Final norm	256	0.989	0.813	0.578	0.084

C Method details

C.1 Fisher construction

For foreground identity c with mean μ_c, $g = C^{-1}\sum_c \mu_c$, and N training tokens,

$$\Sigma_b = \frac{1}{C}\sum_c (\mu_c - g)(\mu_c - g)^\top, \qquad \Sigma_w = \frac{1}{N}\sum_{c,i}(z_{ci} - \mu_c)(z_{ci} - \mu_c)^\top. \tag{7}$$

The Fisher criterion of Equation 1 is maximised over $w \in \mathrm{span}(U)$, where $U \in \mathbb{R}^{1024\times 64}$ holds the leading within-class principal components and $\lambda = 0.1\ \mathrm{tr}(U^\top \Sigma_w U)/64$. The top 14 generalised eigenvectors form $W \in \mathbb{R}^{14\times 1024}$, and we retrieve by cosine between $W(z-g)$ and $W(\mu_c - g)$. The angular study applies the same criterion to δ or u_{mix} and keeps eight directions.

For the upsampler, the readout $W \in \mathbb{R}^{128\times 1024}$ is initialised by the same solve on ADE20K training tokens (2000 images): the readout input is the 256-pixel parent token, the label is the class covering at least half of a child cell in the 1024-pixel image, and children whose class differs from the parent's majority class receive five times the weight.

C.2 Prototype updates

The upsampler keeps four prototype vectors per class. They are updated as an exponential moving average (decay 0.99) of ground-truth child tokens after each optimiser step, and never from the batch being predicted. The router mixes the four highest-scoring prototypes into $\bar{P}$.

C.3 Losses

Table 7 lists the training losses. The routing, prototype and margin terms weight minority children, whose class differs from the parent's majority class, five times.

Table 7: Training losses of prototype upsampling and their weights.

Loss	Weight
Centred cosine to the target	1.0
Relative L2 to the target	1.0
Routing cross-entropy against the class occupancy of each child cell	0.5
InfoNCE between semantic coordinates and class prototypes (temperature 0.15)	0.3
Margin on the hardest wrong class	0.1
Image-level class presence	0.1
Consistency: a learned downsampler of the children reproduces the parent token	0.1

C.4 Training hyperparameters and evaluation protocols

Evaluation sets. The linear head has 151 outputs, including the void class; each protocol applies its own ignore-label rule. The DPT head reads four intermediate layers. Several evaluation sets appear in the paper, and we keep them separate:

Protocol	Evaluation	Purpose
Native ADE20K	Full validation split (2000 images)	Segmentation benchmark
Image-PCA diagnostic	200 images, 512^2	Keep/drop PCs in both heads
Boundary diagnostic	500 images, 512^2	Spatial precision
Geometry/readout study	96 images, 512^2	Norm, dataset/image PCA, geometry
Feature-upsampling study	500 images, 1024^2 crops	Upsampled features vs. target

The native protocol resizes the shorter side to 512, predicts with 512×512 sliding windows at stride 341, averages logits in the overlaps, and scores against the original-resolution annotation without test-time augmentation. The geometry study calibrates on 128 training images. Historical decoder interventions use an earlier linear checkpoint, and the later geometry study uses the final one. Residual alignment uses 64 calibration and 47 evaluation images, and write attenuation uses eight.

Upsampler training. Each ADE20K training image's shorter side is resized to 1024 and a random 1024×1024 crop is taken, with random horizontal flips. The input is DINOv3 on the crop downsampled to 256 pixels (16×16 tokens), and the target is DINOv3 block 24 on the 1024-pixel crop (64×64 tokens). We train for 15 epochs with batch size 8, AdamW (learning rate 3×10^{-4}, weight decay 0.05), a cosine schedule with 1000 warm-up steps, and bf16 precision, and keep the epoch with the highest validation centred cosine (epoch 13).

Feature-upsampling comparison. We use the first 500 ADE20K validation images with the same 1024-pixel centre-crop geometry. The centred cosine is centred on the mean target token. A minority cell is a 16-pixel cell whose majority class covers at least half the cell and differs from the majority class of its 64-pixel parent patch. JAFAR gives the same scores with 256-pixel guidance as with 1024-pixel guidance.

Linear-probe benchmark. The linear head is Dropout2d, BatchNorm and a 1×1 convolution. The upsamplers are frozen and the logits are bilinearly resized to the input size. All heads use the same recipe: 40k steps, batch size 16, AdamW (learning rate 10^{-3}, weight decay 10^{-3}), a one-cycle schedule with 1500 warm-up steps, fp32, random-scale crops and photometric augmentation. The raw-token and JAFAR heads are evaluated at step 40k.

Decoder comparison. The base DPT decoder on frozen DINOv3 takes the updates written by blocks 8, 13, 16 and 23 (each with a trainable LayerNorm, restricted to channels 256–511), expanded by a two-layer transformer and pixel shuffling to 128^2, 64^2, 32^2 and 32^2 and fused by convolutional refinement blocks. In the two variants we compare, the deepest input is removed and the final block-24 tokens, upsampled to 128×128 either bilinearly or by prototype upsampling, are added at the finest level. Both variants use the linear-probe augmentation, a Dice plus cross-entropy loss, and up to 50 epochs (batch size 8, AdamW with learning rate 10^{-4} and weight decay 0.01, cosine schedule, mixed precision).

D ADDITIONAL EVALUATION

D.1 QUALITATIVE EXAMPLES

Figure 7 shows feature upsampling and linear-probe predictions on two randomly drawn ADE20K validation images.

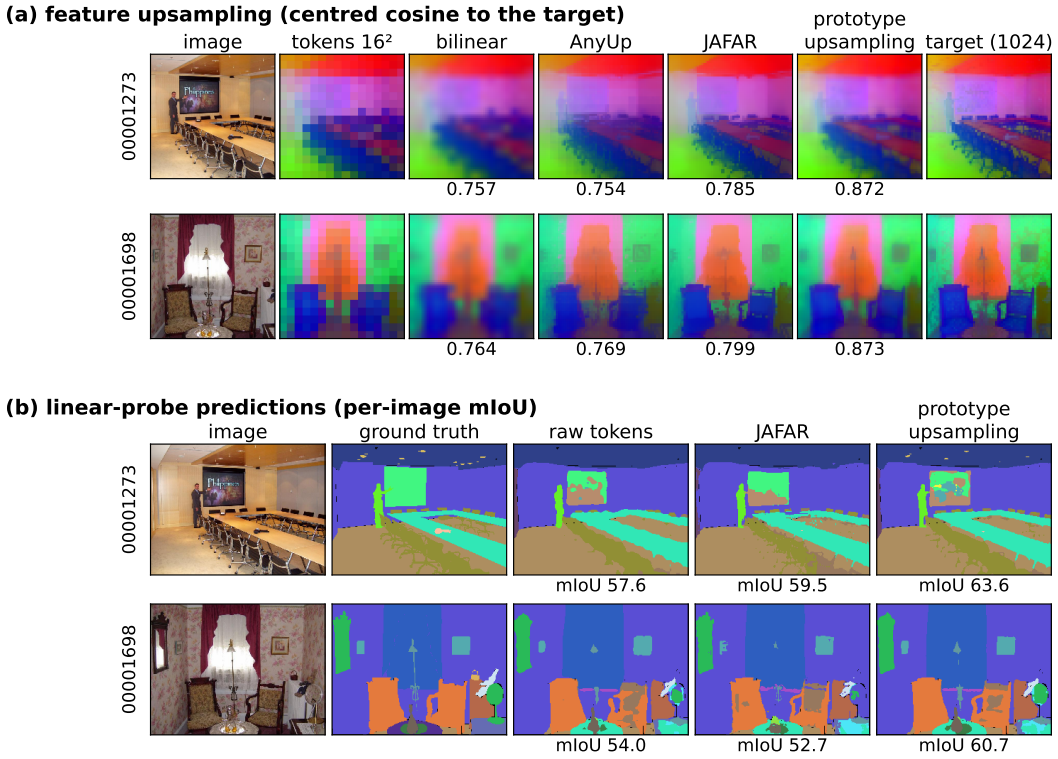

Figure 7: Two randomly drawn ADE20K validation images. (a) Upsampling of 16×16 DINOv3 tokens to 64×64; each row is projected onto one PCA basis fitted on the target (DINOv3 on the image at 1024 pixels), and numbers are centred cosines to the target. (b) Linear-probe predictions with per-image mIoU; unlabelled pixels are black.

D.2 OPENEARTHMAP

We repeat the linear-probe benchmark on OpenEarthMap with its official split (2303 training and 384 validation tiles with available images, eight land-cover classes plus an ignored unknown label), using the same head and recipe with 20k steps and validation every five epochs. Tiles are evaluated with the same sliding-window protocol against their original-resolution labels.