A Mixed Diet Makes DINO An Omnivorous Vision Encoder

Supplementary Material

This Appendix is organized into two broad sections: Section 6 describes our training and evaluation framework, while Section 7 extends the results of the main paper.

6. Training and Evaluation Details

We first present a summary of our training configuration in Table 8. Next, we describe our training data pipeline in Section 6.1. Finally, we elaborate on our evaluation protocols in Section 6.2.

6.1. Data Pipeline

Below we elaborate on all elements of the training-data processing steps previously introduced in Section 3.2.

6.1.1. Photometric Augmentation (RGB)

The photometric augmentation pipeline applies a sequence of standard distortions to the RGB image to encourage robustness against lighting variations and color shifts. The pipeline first adjusts brightness by adding a delta sampled from $[-0.1, 0.1]$. This is followed by a saturation adjustment, where the image is scaled by a factor drawn from $[0.8, 1.2]$. Next, the hue is shifted by a delta within the range $[-0.03, 0.03]$. Finally, the contrast is scaled by a factor sampled from $[0.8, 1.2]$. All random scalars are sampled independently for each distortion type per image instance.

6.1.2. Colorization (Depth & Segmentation)

Using standard colormaps (e.g., grayscale or jet) for the Depth and Segmentation images would allow the encoder to shortcut the alignment task by exploiting low-level channel statistics, thus learning modality-specific features. To counter this, we employ a *natural colorization* strategy.

We define a transformation $\Phi(x_m^{\text{raw}}, x_r^{\text{aug}})$ that re-renders the scalar structural map x_m^{raw} (e.g., depth) using the chromatic distribution of the corresponding RGB image x_r^{aug}. This process creates "hard positives" for the contrastive objective: by forcing the structural map to share the same color histogram as the RGB image, we deny the network the ability to distinguish or align modalities based on superficial color signals. Consequently, the encoder must attend to the shared geometric content to solve the alignment task.

See Algorithm 1 for a pseudocode description of our natural colorization Φ. Formally, we normalize x_m^{raw} to $[0, 1]$ and discretize it into $B = 64$ intensity bins (step 1). Let $b_{u,v} \in \{0, \dots, B-1\}$ denote the bin index of pixel (u, v) in x_m^{raw}. We construct a scene-specific natural color palette $\mathcal{P} \in \mathbb{R}^{B\times 3}$ by aggregating the RGB colors corresponding to each structural intensity bin (step 2). The accumulated color sum $\mathbf{S}_k$ and pixel count $\mathbf{N}_k$ for bin k are computed as:

$$\mathbf{S}_k = \sum_{u,v} \mathbf{1}[b_{u,v} = k] \cdot x_r^{\text{aug}}(u, v), \quad \mathbf{N}_k = \sum_{u,v} \mathbf{1}[b_{u,v} = k]$$

Algorithm 1 Natural Colorization

Require: Scalar map $x_m^{\text{raw}} \in \mathbb{R}^{H\times W}$ where $m \in \{\text{Depth}, \text{Segmentation}\}$
Require: Augmented RGB image $x_r^{\text{aug}} \in \mathbb{R}^{H\times W\times 3}$
Require: Number of bins $B = 64$, kernel size $K = 5$, constant $\epsilon = 10^{-6}$
Ensure: Colorized map $x_m \in \mathbb{R}^{H\times W\times 3}$

Step 1: Normalization and Discretization
$x_m^{\text{norm}} \leftarrow \frac{x_m^{\text{raw}} - \min(x_m^{\text{raw}})}{\max(x_m^{\text{raw}}) - \min(x_m^{\text{raw}}) + \epsilon}$ ▷ Normalize modality m to $[0, 1]$
for each pixel (u, v) **do**
  $b_{u,v} \leftarrow \text{clip}(\lfloor x_m^{\text{norm}}[u, v] \cdot B \rfloor, 0, B-1)$ ▷ Compute bin indices
end for

Step 2: Palette Accumulation ▷ Aggregates x_r^{aug} stats per bin
Initialize $S \in \mathbb{R}^{B\times 3}$ and $N \in \mathbb{R}^{B}$ with zeros
for each pixel (u, v) **do**
  $k \leftarrow b_{u,v}$
  $S[k] \leftarrow S[k] + x_r^{\text{aug}}[u, v]$
  $N[k] \leftarrow N[k] + 1$
end for

Step 3: Palette Smoothing ▷ Fills gaps via 1D convolution
Define uniform kernel $w \in \mathbb{R}^K$ where $w_i = 1$
$\tilde{S} \leftarrow \text{Convolve1D}(S, w)$
$\tilde{N} \leftarrow \text{Convolve1D}(N, w)$

Step 4: Palette Normalization
for $k \in \{0, \dots, B-1\}$ **do**
  $\mathcal{P}[k] \leftarrow \tilde{S}[k]/(\tilde{N}[k] + \epsilon)$ ▷ Compute avg color per bin
end for

Step 5: Image Re-rendering
for each pixel (u, v) **do**
  $x_m[u, v] \leftarrow \mathcal{P}[b_{u,v}]$ ▷ Map bins to palette colors
end for
return x_m

Table 8. Training Configuration for Omnivorous DINO

Category	Details
Architecture	DINOv2 ViT-B/14 (173M parameters)
	Layers 0–7 are frozen, 8–11 are fine-tuned.
Optimizer	AdamW with learning rate 1×10^{-4}
Compute	TPU v4 ($4 \times 4 \times 4$) for 20,000 steps, with a total runtime of 1 hour 14 minutes
Batch Size	512 (Global)
Datasets	ScanNet [7], TartanAir [46], Hypersim [39], MOVi [15], PointOdyssey [53], DynamicReplica [24]
Preprocessing	224×224 resolution (RGB–bilinear resize; Depth & Seg–nearest neighbor; center crop to square)
	Photometric Augmentation if training (RGB)
	Colorization (Depth and Seg)
	Normalization using ImageNet-1k mean and std (RGB, Depth, Seg)
	Modality Mixup ($\alpha_{max} = 0.5$ if training else 0.0)

To ensure continuity, we apply a 1D smoothing convolution to $\mathbf{S}$ and $\mathbf{N}$ using a kernel of size 5 (step 3). The final palette value for bin k is $\mathcal{P}_k = \tilde{\mathbf{S}}_k/(\tilde{\mathbf{N}}_k + \epsilon)$, where ϵ is set to $1e-6$ for numerical stability. The colorized map x_m is generated by mapping each pixel in the raw map to its corresponding palette entry: $x_m(u, v) = \mathcal{P}_{b_{u,v}}$.

6.1.3. Normalization (RGB, Depth, & Segmentation)

We use the ImageNet-1k mean pixel value $(0.485, 0.456, 0.406)$ and standard deviation $(0.229, 0.224, 0.225)$ to standardize all [0,1] images.

6.1.4. Modality Mixup

While natural colorization forces the encoder to focus on structure, it leaves the depth and segmentation maps stripped of textured. Due to this domain gap, the model may struggle to relate geometric shapes to rich photometric cues. To bridge the gap, we use modality mixup. By stochastically blending the colorized structural maps with the original RGB image, we span a continuous "modality spectrum" that interpolates between pure geometry (Depth/Segmentation) and pure texture (RGB). This exposes the encoder to a smooth space of inputs, encouraging it to learn representations that are invariant to the ratio of texture-to-structure, rather than overfitting to discrete modality tokens.

Let x_m be the naturally colorized map for modality $m \in \{\text{Depth}, \text{Segmentation}\}$ (from Algorithm 1) and x_r^{aug} be the photometrically augmented RGB image. We generate the final mixed input x_m^{mixup} via convex combination:

$$x_m^{\text{mixup}} = (1 - \alpha_m)x_m + \alpha_m x_r^{\text{aug}}$$

where the mixing coefficient α_m is sampled uniformly from the range $[0, \alpha_{\text{max}}]$ independently for each training example. We set $\alpha_{\text{max}} = 0.5$ to ensure the structural signal remains dominant while re-introducing sufficient texture to facilitate alignment. This strategy effectively constructs a "continuous bridge" between modalities, preventing the feature space from fragmenting into disjoint islands of geometry and texture.

6.2. Evaluation Protocols

We adopt the protocols established by DINOv2 [37] or Probe3D [10] wherever possible. We elaborate all details in the following subsections for completeness.

6.2.1. Cross-Modal Retrieval

For all datasets (ScanNet, MOVi, TartanAir), inputs are resized to 224×224 (using bilinear interpolation for RGB and nearest-neighbor for depth/segmentation) followed by a center crop. Single-channel structural inputs (depth and segmentation) are tiled to 3 channels and normalized using standard ImageNet statistics ($\mu = [0.485, 0.456, 0.406], \sigma = [0.229, 0.224, 0.225]$) after scaling pixel values to $[0, 1]$. Features are extracted using the frozen DINOv2 backbone and our adapter, applying L_2 normalization to the final embeddings.

We compute pairwise cosine similarity between the query and gallery sets. To handle large-scale evaluation efficiently, similarity matrices are computed in batches of 2048. The rank for a given query is determined by counting the number of gallery items with a similarity score strictly greater than or equal to the ground-truth pair's score (using a numerical stability threshold $\epsilon = 10^{-6}$). As our evaluation setup assumes a strict one-to-one mapping between modalities (i.e., exactly one positive match per query), the Mean Average Precision (mAP) reported is equivalent to the Mean Reciprocal Rank (MRR). We average results over all six directed modality pairs.

6.2.2. Monocular Depth Estimation

Data and Preprocessing. We evaluate on NYUv2 [34] and NAVI Probe3D [10]. Unlike the classification or retrieval tasks which often resize inputs to a standard 224×224, we perform evaluation on high-resolution images to preserve geometric details (i.e., 480×640 for NYUv2, 512×512 for

NAVI). To process these variable resolutions with a ViT backbone trained on fixed patch sizes, we employ a "pad-to-patch" strategy: images are first center-cropped to the target resolution and then padded to the nearest multiple of the patch size (p=14). This allows the frozen backbone to process the dense grid of patches without interpolation artifacts. Standard photometric distortions and random rotations are applied during training, while horizontal flipping is used for test-time augmentation.

Decoder Architectures. We investigate the expressivity of our learned features using two distinct decoder heads.

- **Linear Head:** A lightweight baseline that projects the final layer's patch tokens directly to depth bins using a single linear layer. The output is bilinearly upsampled to the input resolution. This setup tests the explicit geometric information present in the final semantic embedding.
- **DPT Head:** A Dense Prediction Transformer (DPT) decoder that aggregates intermediate features from the backbone. Specifically, we gather tokens from layers 3, 6, 9, 12 (for the ViT-B/14 variant), fuse them using valid convolutions and upsampling blocks to recover high-resolution details. This head evaluates the backbone's ability to provide multi-scale hierarchical features suitable for dense prediction.

Training Objective. Both heads are trained (while keeping the backbone frozen) to classify pixels into 256 depth bins. We minimize a combined objective consisting of a Scale-Invariant Gradient Loss (sigloss) to enforce global structural consistency and an edge-aware gradient loss to sharpen local discontinuities. We train for 50,000 steps using AdamW with a compound learning rate schedule (constant, piecewise constant, and linear warmup).

6.2.3. Semantic Segmentation

Data and Preprocessing. We evaluate semantic segmentation on ADE20k, Cityscapes, and Pascal VOC. During training, we employ standard data augmentation techniques: input images undergo random resizing (ratio range $[0.5, 2.0]$), random horizontal flipping, and photometric distortion. The images are then randomly cropped to a fixed resolution of 512×512.

Evaluation Protocol. Unlike the monocular depth evaluation which processes full images via padding, our segmentation evaluation employs a sliding window protocol to handle high-resolution inputs (e.g., Cityscapes) without downsampling artifacts. We perform inference on 512×512 crops with a stride of 341 pixels. Predictions from overlapping windows are averaged (mean logits) before the final argmax.

Decoder Architectures. We utilize the same two decoder configurations—Linear and DPT—as described in the Monocular Depth Estimation section (Appendix 6.2.2). The backbone remains frozen as before. The only modification is the final projection layer, which maps to K semantic classes (e.g., $K = 150$ for ADE20k) instead of depth bins.

Optimization. We train for 40,000 steps with a batch size of 16. We use the AdamW optimizer with a weight decay of 10^{-4}. The learning rate follows a polynomial decay schedule (power 1.0) combined with a linear warmup for the first 1,500 steps. Performance is measured using the Mean Intersection-over-Union (mIoU), computed by aggregating confusion matrices over the entire validation set.

6.2.4. Multiview Correspondence

Data and Preprocessing. We evaluate 3D feature correspondence using the NAVI dataset. Image pairs are resized to 224×224. We extract feature maps from the encoder, which correspond to a 16×16 grid of patches (given the patch size $p = 14$). We do not employ a trained prediction head for this task; instead, we evaluate the raw feature representations directly.

Matching Protocol. For a given image pair, we compute the pairwise cosine similarity matrix between the flattened spatial tokens ($N = 256$) of the source and target views. We determine the predicted correspondence for each token by selecting the nearest neighbor (argmax of cosine similarity) in the other view. We evaluate bidirectional matches.

Metric: PCK@0. We report the Percentage of Correct Keypoints (PCK) at a strict threshold of 0.0. Since our evaluation operates on the discrete 16×16 token grid, a threshold of 0.0 requires the predicted token index to exactly match the ground-truth token index (i.e., the predicted patch must be the exact same patch as the ground truth). We generally report performance using the final layer's features. That said, we also include an ablation (Table 11) measuring 3D correspondence across all fine-tuned Omnivorous ViT blocks (i.e., the last four).

6.2.5. Linear Probe Classification

Architecture and Training. To assess the linear separability of the learned representations, we train a linear classifier on top of the frozen backbone. We attach a single linear layer (projecting from the feature dimension D to the number of classes $K = 1000$) to the extracted features. The linear layer is trained to minimize the weighted softmax cross-entropy loss, while the backbone remains frozen.

Evaluation Protocol. We evaluate on ImageNet-1k [8], reporting top-1 accuracy on the validation split. We employ standard data augmentation during training (random resized crops and horizontal flips), while validation images are resized to 256 pixels and center-cropped to 224×224. We and train the prober for 10 epochs, sweeping over a range of learning rates (base values: [0.15, 0.2, 0.5, 1.0, 2.0]) along with the nesterov optimizer. We report the best accuracy achieved across the base learning rates. We report results using both the CLS token embedding and the concatenation

of the CLS token and the global average pooled (GAP) features.

6.2.6. k-NN Classification

ImageNet. We follow the standard DINO evaluation protocol for ImageNet-1k. We extract features for the training set (index) and validation set (query) using the frozen backbone. The images are preprocessed by resizing the shorter side to 256 pixels, taking a central 224×224 crop, and normalizing with ImageNet statistics. We employ weighted soft voting: for each query, we compute the cosine similarity with its k nearest neighbors in the training set. These similarities are converted to weights using a softmax with temperature $\tau = 0.07$. The class probabilities are summed across the neighbors, and the class with the highest aggregate probability is selected. We report the top-1 accuracy corresponding to the best k swept over $\{5, 10, 20, 50, 100\}$.

Transfer Datasets. For iNaturalist, SOP, Google Landmarks v2 (GLDv2), RP2K, and Food2k, we perform "hard" k-NN classification (which is equivalent to Recall@1). We use the same image preprocessing as ImageNet ($256 \rightarrow 224$ center crop).

- For **GLDv2**, we match queries from the test set against the distinct index set provided by the dataset ($N \approx 761k$).
- For **iNaturalist, SOP, RP2K, and Food2k**, we follow the standard metric learning protocol where the test set serves as both the query and the index. We compute the nearest neighbor for each query from the index, excluding the query itself (self-match), and check if the retrieved class label matches the query label.

6.2.7. Zero-Shot Modality Transfer

Protocol. To assess the universality of the learned feature space, we design a strict transfer protocol. We train a depth estimation head (either Linear or DPT as before) using *only* RGB images from the NYUv2 dataset. Once trained, we freeze the entire model (backbone + depth head) and evaluate it on the PACE dataset. This setup introduces a small domain shift and, crucially, a modality shift.

Modalities. We evaluate performance on three distinct input types:

- **RGB:** Serves as the baseline. The model encounters a domain shift (NYUv2 $\rightarrow$ PACE) but the modality remains consistent with training.
- **Segmentation:** A modality seen by the Omnivorous backbone during pre-training, but *never* seen by the depth head. To render these inputs compatible with the frozen backbone, segmentation maps are preprocessed using our Natural Colorization scheme (Algorithm 1) to match the spectral statistics of RGB images. Unlike the backbone pre-training stage, we do *not* apply modality mixup during this evaluation.
- **NOCS (Normalized Object Coordinate Space):** NOCS maps represent dense coordinate fields rather than photometric data. It is a modality that is completely out-of-distribution; neither the Omnivorous backbone nor the depth head observes NOCS maps during training. The 3-channel coordinate maps are normalized using standard ImageNet RGB statistics before being fed into the model.

Success on NOCS and Segmentation inputs indicates that the encoder maps these diverse signals to a shared feature space that is interpretable by the RGB-trained head.

7. Extended Results

7.1. Diagnostic Metrics

Expanding on Fig 1, we report detailed cross-modal alignment and cross-scene discernibility metrics before and after Omnivorous training. Table 9 shows that our default checkpoint of Omnivorous DINO greatly improves cross-modal alignment while sacrificing some cross-scene discernibility (e.g., from 0.198 to 0.259 $< R_1, R_2 >$ similarity on ScanNet). This echoes Fig 4a which showed the trade-off as a function of our λ_{anchor} loss weight.

7.2. 3D Tasks

We revisit all tasks from the Probe3D framework for the Omnivorous DINO ViT-B/14 checkpoint introduced in Sec 4. We present two evaluations that were omitted in the main paper (normals estimation and multiview 3D correspondence), and add qualitative results for those already presented in the main paper (e.g., depth estimation and segmentation):

7.2.1. Normals Estimation

See Table 10. Omnivorous is consistently at par with DINOv2 across all metrics.

7.2.2. Multiview Correspondence

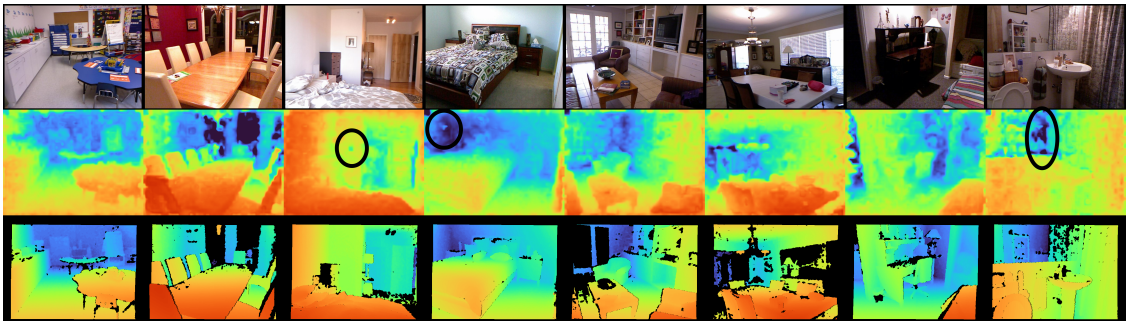
See Table 11. While our model is consistently more 3D-consistent than the original DINOv2, the performance gap is a bit inconsistent with respect to the block where the features are taken from, i.e., there is no clear pattern of increasing/decreasing 3D-correspondence as a function of network depth. This merits future investigation.

7.2.3. Semantic Segmentation

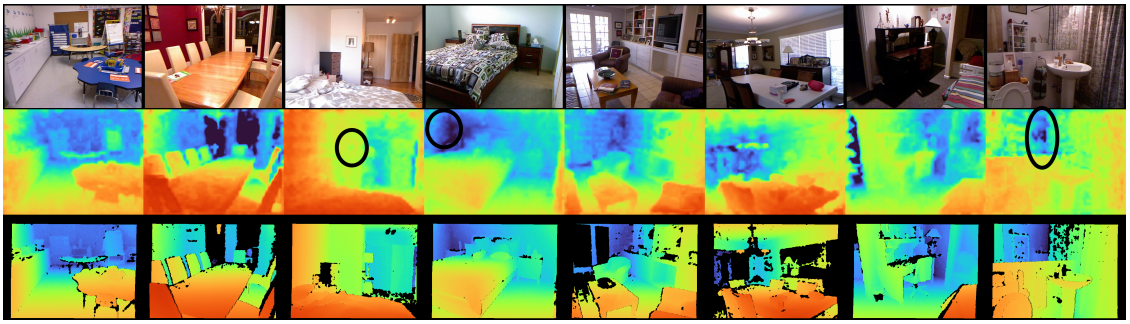
See Fig 8 & 9 for a qualitative comparison between Omnivorous and DINOv2. We find that our model helps reduce over-segmentation, and is consistently more resilient to textural details in the input images.

7.2.4. Monocular Depth

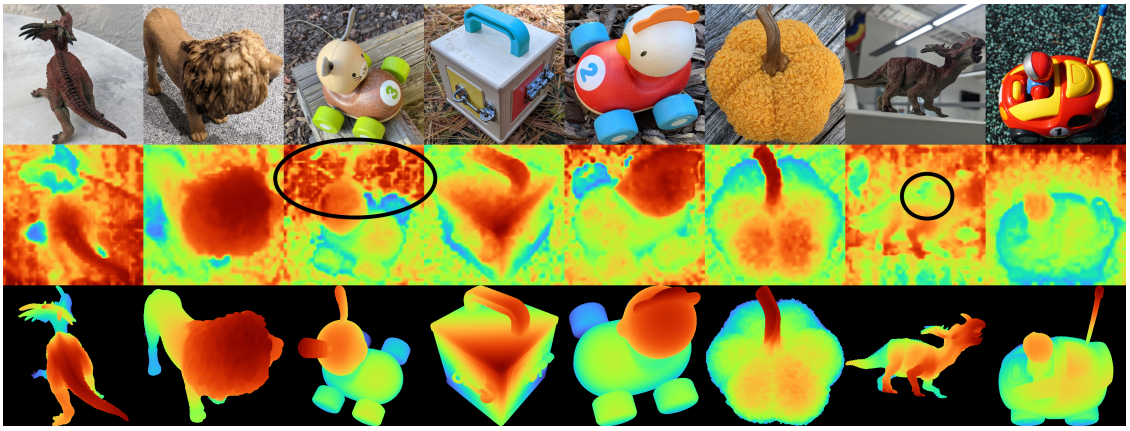
See Fig 7 for a qualitative comparison between Omnivorous and DINOv2. As with predicted segmentations, we find that our model helps reduce high-frequency noise in the linear head's depth predictions. Our model performs consistently better on flat surfaces, and cases where a flat object is placed on a flat surface (e.g., a painting on the wall).



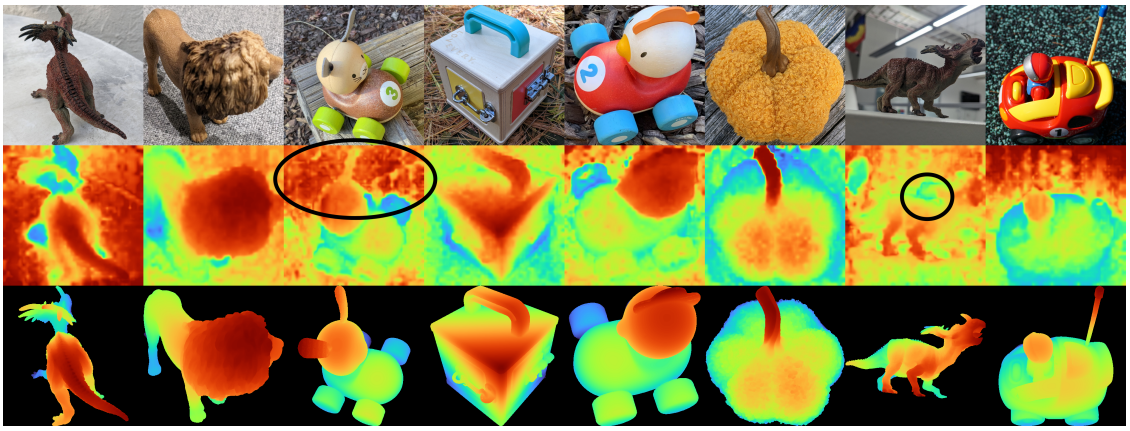
(a) DINO ViT-B/14 depth prediction on NYUv2. Top: input images, middle: predictions, bottom: ground-truth.



(b) Omnivorous DINO ViT-B/14 depth prediction on NYUv2. Top: input images, middle: predictions, bottom: ground-truth.

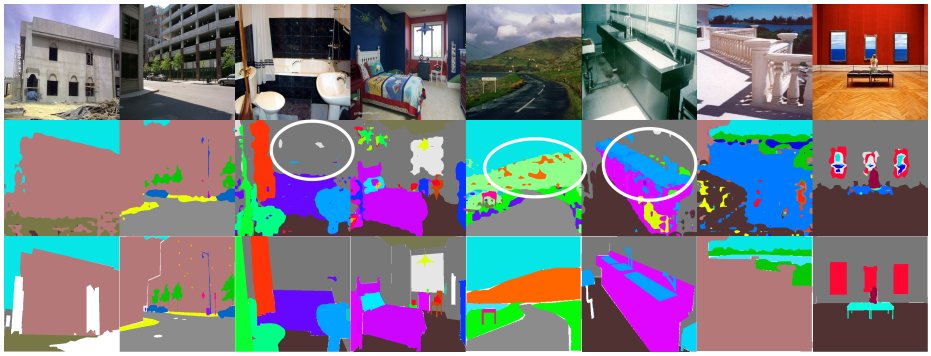


(c) DINO ViT-B/14 depth prediction on NAVI Probe3D. Top: input images, middle: predictions, bottom: ground-truth.

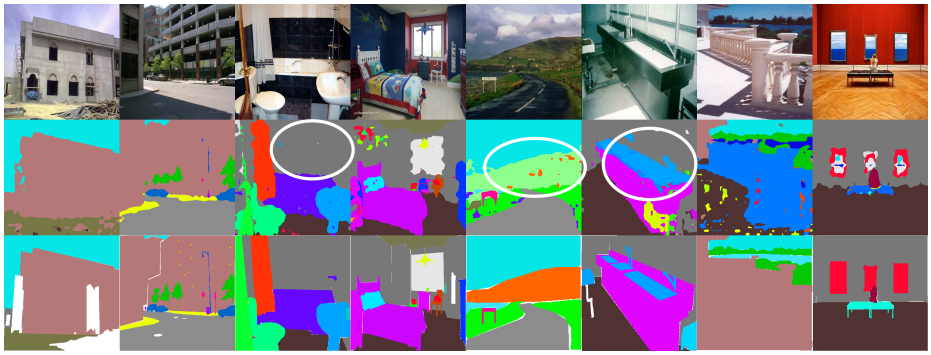


(d) Omnivorous DINO ViT-B/14 depth prediction on NAVI Probe3D. Top: input images, middle: predictions, bottom: ground-truth.

Figure 7. **Qualitative comparison (Omnivorous vs DINOv2) on depth prediction** using a linear head. Please compare a versus b, and c versus d. We highlight notable differences using a black oval.



(a) DINO ViT-B/14 segmentation prediction on ADE20k. Top: input images, middle: predictions, bottom: ground-truth.



(b) Omnivorous DINO ViT-B/14 segmentation prediction on ADE20k. Top: input images, middle: predictions, bottom: ground-truth.

(c) DINO ViT-B/14 segmentation prediction on Pascal VOC. Top: input images, middle: predictions, bottom: ground-truth.

(d) Omnivorous DINO ViT-B/14 segmentation prediction on Pascal VOC. Top: input images, middle: predictions, bottom: ground-truth.

Figure 8. **Qualitative comparison (Omnivorous vs DINOv2) on segmentation prediction** using a linear head. We highlight notable differences using a white oval.

Table 9. **Diagnostic metrics:** we expand Fig 1, showing cross-modal alignment and cross-scene discernibility metrics across three datasets for both pretrained DINOv2 and the adapted Omnivorous model (at our default $\lambda_{anchor} = 10$)). We denote the three modalities R, D, and S (RGB, Depth, and Segmentation, respectively). The metrics are computed without modality-mixup (i.e., $\alpha_{max} = 0$). For $< R_1, R_2 >$, lower similarity is considered better.

	DINOv2 ViT-B/14				Omnivorous ViT-B/14			
dataset	$< R, D >$	$< R, S >$	$< D, S >$	$< R_1, R_2 >$	$< R, D >$	$< R, S >$	$< D, S >$	$< R_1, R_2 >$
movi	0.263	0.284	0.481	0.237	0.567	0.579	0.721	0.279
scannet	0.285	0.216	0.413	0.198	0.600	0.550	0.663	0.259
tartanair	0.345	0.359	0.543	0.172	0.607	0.603	0.736	0.223

Table 10. **Downstream eval:** normals estimation using a DPT head.

dataset	model	absrel ↓	diff 11.25 ↑	diff 22.50 ↑	diff 30.00 ↑	mean diff angle ↓	rmse angle ↓
navi	DINOv2 ViT-B/14	197.9	43.5	72.2	82.1	18.6	24.6
	Omnivorous ViT-B/14	**197.8**	**43.6**	**72.3**	**82.2**	18.6	24.6
nyuv2	DINOv2 ViT-B/14	134.9	63.4	80.8	86.5	14.1	21.7
	Omnivorous ViT-B/14	**134.1**	**63.5**	80.8	86.5	14.1	**21.6**

Table 11. **Multiview correspondence:** we report the Percentage of Correct Keypoints (↑) at the 0.0 level (i.e., only exact matches are counted). We measure correspondence for all the four blocks that are fine-tuned in the Omnivorous case, comparing them with their frozen DINOv2 counterparts.

block number model	9	10	11	12
DINOv2 ViT-B/14	29.76	28.49	27.68	28.57
Omnivorous ViT-B/14	29.76	**28.93**	**28.63**	**29.00**

7.3. Ablations

7.3.1. Modality mixup

We ablate the mixup hyperparameter α_{max} which controls the degree to which the modalities are blended during training (see Table 12). While depth prediction is an outlier, the performance on all other tasks continues to increase up to $\alpha_{max} = 1$, which implements full-spectrum blending of the three modalities. Our default value $\alpha_{max} = 0.5$ was chosen to balance across the tasks.

7.3.2. TIPS instead of DINOv2

As TIPS [32] shares the same ViT architecture as DINOv2, we can "ablate" our pretrained teacher by running Omnivorous training on TIPS instead of DINOv2. Two important distinctions are the shape of the position encoding parameter (TIPS uses 16×16 vs DINOv2's 37×37) and the number of CLS tokens (TIPS uses two while DINOv2 uses one). We train Omnivorous TIPS ViT-B/14 using the default $\alpha_{max} = 0.5$, and freezing the first 8 blocks as we did for Omnivorous DINOv2.

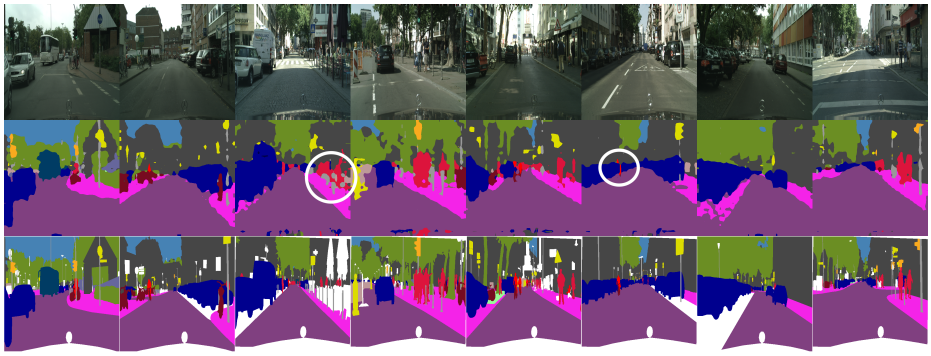
Table 12. **Ablating modality mixup.** We vary α_{max} which controls the degree of blending between modalities during training. We report linear-probe performance on (i) classification (ImageNet accuracy using the TOK feature, without intermediate layers), (ii) depth prediction (δ_1 on NYUv2), and (iii) segmentation (mean IoU on Cityscapes). We also report (iv) 3D correspondence (percentage of correct keypoints at threshold 0.0 on NAVI), which is assessed directly from features without a linear probe.

α_{max}	classif. ↑	depth ↑	segment. ↑	3D corresp. ↑
0	0.831	**0.899**	0.624	28.40
0.25	0.834	0.898	0.630	28.96
0.5	0.834	0.896	**0.632**	29.00
0.75	0.834	0.894	**0.632**	**29.04**
1.0	**0.835**	0.891	**0.632**	29.03

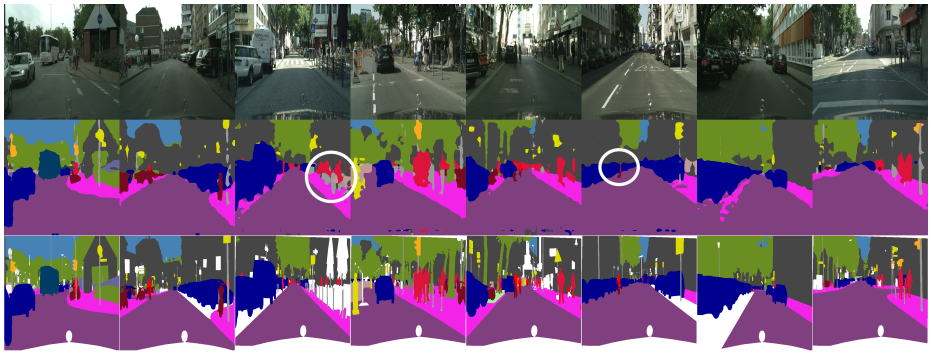
Fig 10 shows that although it is harder for Omnivorous distillation to improve on the performance of TIPS (than in the case of DINOv2), $\lambda_{anchor} = 100$ nevertheless does exceed the depth and segmentation performance of higher values of λ_{anchor}, which are anchored more strongly to the pretrained teacher. This attests to the generality of the Omnivorous framework regardless of the choice of pretrained teacher network.

7.3.3. Training an Adapter on Top vs. Fine-Tuning Final Blocks

We now ablate the parametrization of the student network. Rather than the default setting of fine-tuning the final blocks of a pretrained backbone, we train a zero-initialized adapter



(a) DINO ViT-B/14 segmentation prediction on Cityscapes. Top: input images, middle: predictions, bottom: ground-truth.



(b) Omnivorous DINO ViT-B/14 segmentation prediction on Cityscapes. Top: input images, middle: predictions, bottom: ground-truth.

Figure 9. **Qualitative comparison (Omnivorous vs DINOv2) on segmentation prediction (contd.)** using a linear head. We highlight notable differences using a white oval.

network on top of the frozen backbone. In this scenario, the student network is in fact larger than the teacher. All the teacher blocks are frozen and preserved in the student network; only the adapter blocks are trained. We use the same number of adapter blocks (four) as we fine-tuned for our default version of Omnivorous DINOv2.

We evaluate each scenario using a linear head on the final layer. We do not use a DPT head as it would require intermediate activations (typically from blocks [3, 6, 9, 12] in a 12-block ViT-B network), which cannot be consistently applied between the "adapter-on-top" and "finetune-final-blocks" settings, because the former in fact comprises 16 blocks rather than 12.

Table 13 shows comparable performance between the two settings, showing our distillation-based approach and training losses can easily be applied to alternative parametrizations of the student network.

7.3.4. Number of Blocks to Freeze

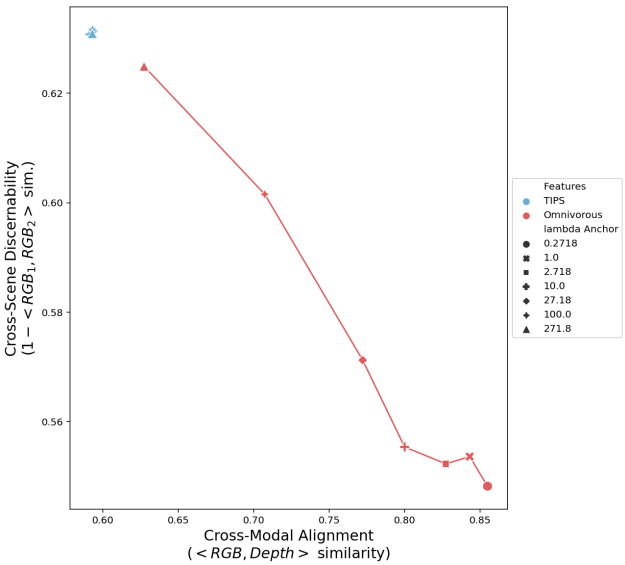
We assess how many ViT blocks can be inherited from the teacher network and kept frozen in Table 14. As before, we fine-tune only the final blocks of the network, keeping the preceding $L_{\text{stop-gradient}}$ blocks frozen. We evaluate depth and segmentation prediction using both a DPT and linear head. Our default setting for Omnivorous ViT-B/14, $L_{\text{stop-gradient}} = 8$ is chosen on this basis.

Table 13. **Ablating the parametrization of the student:** we either train a 4-block ViT on top of the DINOv2 ViT-B/14 backbone, or fine-tune the final 4 blocks of the backbone (ours). As before in Table 12, we report metrics (all ↑) on (i) classification, using either linear probes on TOK & GAP, or k-NN, (ii) depth prediction (linear head), (iii) segmentation (linear head), and (iv) multiview correspondence.

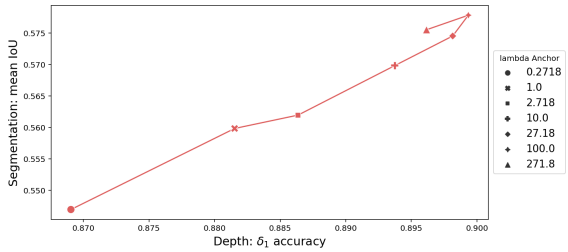
	Classification (acc.)		Depth (δ_1)		Segmentation (mean IoU)			Corresp. (PCK)
dataset	inet (linear)	inet (k-NN)	navi	nyuv2	ade20k	cityscapes	pascal voc	navi
parametrization								
Adapter on top	**0.840**	81.832	0.679	**0.905**	0.470	0.628	0.826	28.15
Fine-tune final blocks	0.838	**81.974**	**0.706**	0.896	**0.475**	**0.632**	0.826	**29.00**

Table 14. **Ablating the number of blocks kept frozen**, denoted by $L_{\text{stop-gradient}}$, when training Omnivorous DINOv2. There are 12 total blocks in the ViT-B/14 architecture.

		Depth (δ_1)		Segmentation (mean IoU)		
	dataset	navi	nyuv2	ade20k	cityscapes	pascal voc
readout	$L_{\text{stop-gradient}}$					
DPT	4	0.777	0.948	0.495	0.727	0.855
	6	0.778	0.947	0.494	**0.733**	0.853
	8	**0.781**	0.948	**0.505**	0.732	**0.857**
	10	0.780	**0.949**	0.504	0.731	0.852
Linear	4	0.698	0.894	0.475	0.622	0.829
	6	0.703	0.896	**0.476**	0.629	0.829
	8	**0.706**	0.896	0.475	**0.632**	0.826
	10	0.705	0.895	0.473	0.628	0.825



(a) Performance frontier for Omnivorous TIPS (Alignment vs. Discernibility) on TartanAir. We omit the datapoint for $\lambda_{anchor} = 0.0$, located at $(x = 0.783, y = 0.859)$, for clarity.



(b) Performance frontier for Omnivorous TIPS (Segmentation vs. Depth). As in Fig 4b, we use linear-head evaluation prediction performance for Depth (NYUv2) and Segmentation (Cityscapes). We omit the datapoint for $\lambda_{anchor} = 0.0$, located at $(x = 0.745, y = 0.435)$, for clarity.

Figure 10. **Behavior of Omnivorous TIPS.**