NAF: Zero-Shot Feature Upsampling via Neighborhood Attention Filtering

Supplementary Material

Table of contents

A. Mathematical Discussions

We analyze the mathematical foundations of NAF's upscaling mechanism, focusing on the interaction between RoPE [26] and neighborhood attention. Our key finding is that NAF does not merely reweight the inputs using a distance over the image encoder; instead, it learns the Inverse Discrete Fourier Transform (IDFT) of the upsampling aggregation kernel. In other words, NAF dynamically constructs an optimal upsampling filter by predicting spectral coefficients of the learned image encoder.

Preliminaries To recall, NAF shows analogies with Joint Bilateral Filtering due to the spatial-and-content aware kernel. It allows to obtain high-resolution features via the following attention-weighted interpolation:

$$\mathbf{F}_p^{\mathrm{HR}} = \frac{1}{Z(p)} \sum_{q \in \mathcal{N}(p)} \exp\Big(\frac{1}{\sqrt{d}} S(p,q)\Big) \mathbf{F}_q^{\mathrm{LR}}, \quad (1)$$

where $Z(p) = \sum_{q \in \mathcal{N}(p)} \exp\left(\frac{1}{\sqrt{d}} S(p,q)\right)$ is the normalization constant and $S(p,q) := \langle Q_p, K_q \rangle$ is an attention-score for a pair of points (p,q) with queries and keys defined as

$$Q_p := \mathrm{RoPE}(\mathbf{G})_p, \quad K_q := \underset{q' \in q}{\mathrm{AvgPool}} \left[\mathrm{RoPE}(\mathbf{G})_{q'} \right], \quad (2)$$

and $\mathbf{G} = \mathrm{Enc}_\theta(\mathbf{I}) \in \mathbb{R}^d$ denotes the image encoder output having d channels.

Substituting the pooled key yields the following attention-score:

$$S(p,q) = \frac{1}{|\{q' \in q\}|} \sum_{q' \in q} \langle \mathrm{RoPE}(\mathbf{G})_p, \mathrm{RoPE}(\mathbf{G})_{q'} \rangle. \quad (3)$$

A.1. RoPE Expansion

RoPE introduction. To develop the last equation we discuss about 2D-RoPE [26]. To do so, we consider channel pairs $(2c, 2c+1)$ where $c \in \{0, ..., d/2\}$ and define the 2D feature vector for the c-th pair as:

$$\vec{G}_c(p) := \begin{pmatrix} \mathbf{G}_p^{2c} \\ \mathbf{G}_p^{2c+1} \end{pmatrix}, \quad (4)$$

where $\mathbf{G}_p^{2c}$ is the value of the encoded image $\mathbf{G}$, at position p and in channel $2c$.

By definition of RoPE we have for each c:

$$\mathrm{RoPE}(\vec{G}_c(p)) = R_c(p)\, \vec{G}_c(p), \quad (5)$$

where the rotation matrix is

$$R_c(p) := \begin{pmatrix} \cos\Phi_c(p) & -\sin\Phi_c(p) \\ \sin\Phi_c(p) & \cos\Phi_c(p) \end{pmatrix}. \quad (6)$$

with the rotation angle $\Phi_c(p)$ encoding the axial positional information for channel pair c. It is defined by:

$$\Phi_c(p) = \begin{cases} 2\pi\, p_y/\lambda_c & \text{if } 0 \le c < d/4 \quad \text{(Height)} \\ 2\pi\, p_x/\lambda_c & \text{if } d/4 \le c < d/2 \quad \text{(Width)} \end{cases}, \quad (7)$$

with λ_c the wavelength of the c-th frequency band, and $p_x, p_y \in [-1, 1]$ are normalized coordinates along each axis.

Inner product after rotation. We can reinject the definition of RoPE [26] in the attention-score between two positions p and q'. It becomes:

$$\langle \mathrm{RoPE}(\vec{G}_c(p)), \mathrm{RoPE}(\vec{G}_c(q')) \rangle = \vec{G}_c(p)^\top R_c(p)^\top R_c(q')\, \vec{G}_c(q'). \quad (8)$$

Using properties of 2D rotation matrices, the product $R_c(p)^\top R_c(q')$ is itself a rotation by the relative angle

$$\Delta\Phi_c(p,q') := \Phi_c(q') - \Phi_c(p). \quad (9)$$

Hence, we can write

$$R_c(p)^\top R_c(q') = \begin{pmatrix} \cos\Delta\Phi_c(p,q') & -\sin\Delta\Phi_c(p,q') \\ \sin\Delta\Phi_c(p,q') & \cos\Delta\Phi_c(p,q') \end{pmatrix}. \quad (10)$$

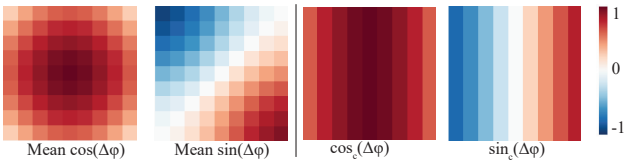


Figure 1. **Illustration of the mean and channel-specific cosine and sine of $\Delta\Phi_c$.** We compute the mean across all channels and select a single random channel to illustrate its individual behavior. For the cosine, we observe an overall decreasing pattern as the distance from the center increases.

To better visualize $\Delta\Phi_c$, we plot in Fig. 1 the mean of cosine and sine over all channels and their values at a specific channel given a neighborhood window of size 9×9. We see that in average the cosine decreases when the points become far, while the sine has a diagonal shape due to the axial-nature of $\Phi_c(p)$.

Dot and cross product decomposition. Expanding the inner product of 2D vectors under a rotation gives the per-channel contribution:

$$\begin{aligned} A_c(p,q') &= \vec{G}_c(p)^\top R_c(p)^\top R_c(q')\,\vec{G}_c(q') \\ &= (\vec{G}_c(p)\cdot\vec{G}_c(q'))\cos\Delta\Phi_c(p,q') \\ &\quad - (\vec{G}_c(p)\times\vec{G}_c(q'))\sin\Delta\Phi_c(p,q'), \end{aligned} \tag{11}$$

with the standard dot and cross products.

Interpretation. The dot product $\vec{G}_c(p)\cdot\vec{G}_c(q')$ measures *feature coherence* (alignment), while the cross product $\vec{G}_c(p)\times\vec{G}_c(q')$ captures *content orthogonality* (perpendicularity). RoPE modulates these content interactions based strictly on the relative axial distance: vertical distance for $d/2$ channels and horizontal distance for the remaining $d/2$ channels. As we can see in Fig. 2, the model learns to discriminate regions based on their encoding. While querying the dog, we recognize its shape and the model learns to aggregate inside values while discriminating outside ones.

A.2. Representation with magnitudes and angles.

Let $\Psi_c(p,q')$ be the angle from $\vec{G}_c(p)$ to $\vec{G}_c(q')$, and let

$$r_p^{(c)} := \|\vec{G}_c(p)\|, \qquad r_{q'}^{(c)} := \|\vec{G}_c(q')\| \tag{12}$$

denote the magnitudes of the feature vectors for channel pair c.

Then the dot and cross products can be expressed as:

$$\begin{aligned} \vec{G}_c(p)\cdot\vec{G}_c(q') &= r_p^{(c)} r_{q'}^{(c)}\cos\Psi_c(p,q'), \\ \vec{G}_c(p)\times\vec{G}_c(q') &= r_p^{(c)} r_{q'}^{(c)}\sin\Psi_c(p,q'), \end{aligned} \tag{13}$$

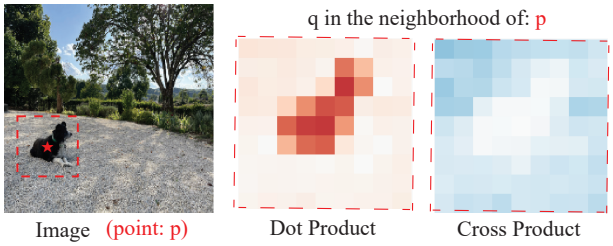


Figure 2. **Dot and cross products for a specific channel** given a query point p on an image. We highlight the neighborhood around p using a dashed red square. On the feature side, after VFM-downsampling, we observe that implicitly NAF discriminates boundaries.

so that the per-channel contribution becomes

$$\begin{aligned} A_c(p,q') &= (\vec{G}_c(p)\cdot\vec{G}_c(q'))\cos\Delta\Phi_c(p,q') \\ &\quad - (\vec{G}_c(p)\times\vec{G}_c(q'))\sin\Delta\Phi_c(p,q') \\ &= r_p^{(c)} r_{q'}^{(c)}\big[\cos\Psi_c(p,q')\cos\Delta\Phi_c(p,q') \\ &\quad - \sin\Psi_c(p,q')\sin\Delta\Phi_c(p,q')\big] \\ &= r_p^{(c)} r_{q'}^{(c)}\cos\big(\Psi_c(p,q')+\Delta\Phi_c(p,q')\big), \end{aligned} \tag{14}$$

where the last equality follows from the cosine angle addition formula. Finally, the pooled attention-score aggregates pairwise interactions over the pooling neighborhood as:

$$\begin{aligned} S(p,q) &= \frac{1}{|\{q'\in q\}|}\sum_{q'\in q}\langle \mathrm{RoPE}(\mathbf{G})_p, \mathrm{RoPE}(\mathbf{G})_{q'}\rangle \\ &= \frac{1}{|\{q'\in q\}|}\sum_{q'\in q}\sum_{c=0}^{d/2-1} r_p^{(c)} r_{q'}^{(c)}\cos\big(\Psi_c(p,q')+\Delta\Phi_c(p,q')\big). \end{aligned} \tag{15}$$

This decomposition clarifies the geometric mechanism of RoPE [26]. Rather than linearly adding a positional bias to a content score, Eq. 3 shows that position and content are **coupled** via phase addition. The magnitude term $r_p^{(c)} r_q^{(c)}$ represents the raw signal strength (feature confidence). The cosine term indicates that the spatial phase difference $\Delta\Phi_c$ acts as a rotation applied to the semantic phase alignment Ψ_c. Constructive interference (a high score) occurs only when the semantic relationship compensates for the spatial offset, effectively implementing a spatially-varying matched filter.

Fourier-inspired Interpretation The derivation of the pairwise attention-score in Eq. (3) reveals a structural equivalence to the Inverse Discrete Fourier Transform (IDFT). To see this, consider the standard reconstruction of a 1D spatial

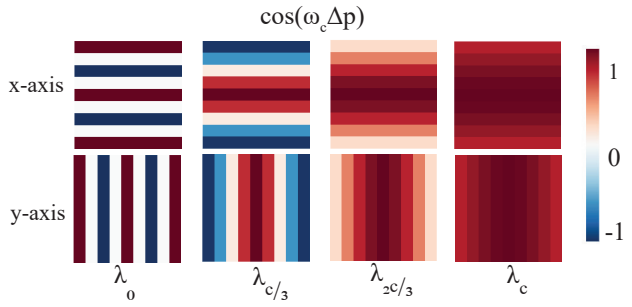


Figure 3. **Illustration of radial wavelets induced by NAF** for a 9×9 neighborhood. We plot $\cos(\omega_c \cdot \Delta x)$ for different λ where Δx is defined as the ℓ_1 distance over x-axis or y-axis between two coordinates over a grid map. In this plot we set the periods as: $\lambda_i = 100^{i/c}$.

signal $f(x)$ from its frequency components:

$$f(x) = \sum_{\omega} \underbrace{|F(\omega)|}_{\text{Amplitude}} \cdot \cos\Big(\underbrace{\psi}_{\text{Content Phase}} + \underbrace{\omega \Delta x}_{\text{Spatial phase}} \Big). \quad (16)$$

By viewing the channel dimension c through the lens of Rotary Embeddings, where each channel corresponds to a specific wavelength λ_c, we can identify c as a spectral frequency index $\omega_c \propto 1/\lambda_c$. We illustrate the resulting cosine and sine in Fig. 3. Consequently, our derived attention-score $S(p, q')$ represented as a 1D score: $S(x)$ acts for each axis as a kernel synthesized via IDFT:

$$S(x) \propto \sum_{c} \underbrace{r_p^{(c)} r_{q'}^{(c)}}_{\text{Amplitude}} \cos\Big(\underbrace{\Psi_c}_{\text{Content Phase}} + \underbrace{\omega_c \Delta x}_{\text{Spatial Phase}} \Big). \quad (17)$$

This mapping offers three fundamental insights into the mechanism of NAF:

1. Learning Fourier Coefficients. The network does not directly predict spatial filter weights. Instead, it predicts the **Fourier series coefficients** of the optimal upsampling kernel. The product of feature magnitudes $r_p^{(c)} r_{q'}^{(c)}$ acts as the *spectral power* for frequency band c. By modulating these magnitudes, the encoder determines how much contribution each frequency—low (global structure) or high (fine detail)—makes to the final interpolation kernel.

2. Spatially-Varying Filter Synthesis. This formulation allows NAF to function as a spatially-varying band-pass filter. In smooth image regions, the encoder can suppress high-frequency channels (reducing $\mathcal{A}_c$ for large c), effectively synthesizing a broad, low-pass smoothing kernel. Conversely, at sharp boundaries, the encoder can boost high-frequency amplitudes to synthesize a peaked, detail-preserving kernel (see Fig. 4).

3. Shift Demodulation. The RoPE term $\omega_c \Delta x$ explicitly encodes the spatial shift theorem. By decomposing the interaction into spectral bands, the attention mechanism can align features that are semantically coherent but spatially phase-shifted. The summation over channels then reconstructs the spatial impulse response required to interpolate the feature at the exact sub-pixel position required by the target resolution.

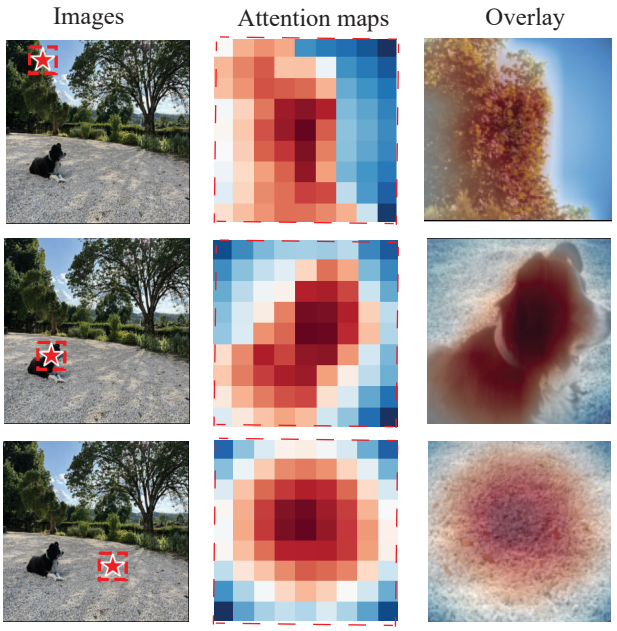


Figure 4. **Attention maps: $\langle Q_p, K_q \rangle$ between a query point p and its 9×9 patch neighborhood (q).** We take 896×896 input images to visualize finer details. In the first row, we see that NAF learns to discriminate between the sky and the tree, i.e., borders. On the second row, it learns to discriminate more complex shapes (dog). On the third row, in uniform region, it shows decreasing attention pattern, akin to the gaussian filter used in classical JBF.

B. Feature Upsampling Experiments

B.1. Training Details

1st step: core training stage. The core training stage of NAF exactly matches prior work. All upsamplers are trained on the same data using 32×32 high-resolution feature maps, with input image resolution chosen to match the VFM stride (448×448 for DINOv2 with stride 14, 512×512 for DINOv3 with stride 16) and low-resolution feature maps corresponding to a $\times2$ downsampling. This stage is identical to JAFAR [5] and AnyUP [30] except that we train for less steps (25k vs 100k).

2nd step: refinement stage. In addition we add an optional refinement stage, where NAF is fine-tuned for 2.500

extra steps with 1024×1024 targets, while input resolutions are randomly sampled between 256×256 and 896×896. Quantitatively, we observe an average of +0.4 mIoU gains on linear probing semantic segmentation evaluated on VOC [8]. Ablation studies in Sec. 5 are performed without this second training stage.

The full training procedure requires approximately 1 hour and 9 GB of memory on an A100 GPU, resulting in a $\times5$ speedup compared to the concurrent AnyUp method [30]. For the final model, the dual-branch encoder employs $L = 2$ blocks with $C = 256$ channels and a neighborhood kernel size of 9. This configuration ensures a fair comparison with other upsamplers, maintaining a parameter count similar to JAFAR [5] and an equivalent number of encoder blocks. For the neighborhood attention module, we adopt an efficient implementation [10–13].

B.2. Task setups

Our evaluations use standard protocols and we evaluate more than 280 combinations of model-dataset-task, covering 5 downstream tasks (semantic segmentation, depth estimation, video label propagation, open-vocabulary segmentation, image denoising), 8 datasets (COCO [19], ADE20K [34], VOC [8], Cityscapes [4], NYUv2 [24], DAVIS [21], ImageNet [23], [18]), and 15 VFMs spanning 9 families with multiple sizes (T/S/B/L). During inference, all methods use identical image resolutions in every table, always matching the evaluated VFM requirements. We provide additional details in the following paragraphs.

Semantic Segmentation. To evaluate the upsamplers, we freeze their parameters and train linear classifiers on the extracted features following Couairon et al. [5]. We train for 20 epochs on Cityscapes [4], Pascal VOC [8], and ADE20K [34], and for 5 epochs on COCO [19]. We employ the AdamW optimizer with a learning rate of 5×10^{-4} for most datasets; however unlike Couairon et al. [5], for Cityscapes, we reduce the learning rate to 1×10^{-4} to ensure stability. A one-cycle cosine annealing scheduler is applied, and all input and target images are resized to 448×448. The classifiers are optimized using the cross-entropy loss. Each dataset has a different number of classes: Cityscapes has 19 classes, Pascal VOC has 21 classes, ADE20K has 151 classes and COCO has 27 classes.

Depth Estimation. For monocular depth estimation, we train linear regressors on NYUv2 [24] for 20 epochs, with a learning rate of 5×10^{-4} and a one-cycle cosine annealing scheduler following depth estimation protocol of Couairon et al. [5]. Consistent with the segmentation task, input and target images are resized to 448×448. Ground-truth depth values are clipped to the range $[d_{\min}, d_{\max}]$, with $d_{\min} = 10^{-3}$ and $d_{\max} = 10.0$.

We optimize the model using a combination of scale-invariant and gradient-based losses. Let $\hat{D}$ denote the predicted depth map and D the target depth map, where values $D > d_{\max}$ are set to zero. The total depth loss is defined as:

$$\mathcal{L}_{\text{depth}} = \lambda_\sigma \, \mathcal{L}_{\text{SI}}(\hat{D}, D) + \lambda_\nabla \, \mathcal{L}_{\text{grad}}(\hat{D}, D), \tag{18}$$

where $\mathcal{L}_{\text{SI}}$ is the scale-invariant loss and $\mathcal{L}_{\text{grad}}$ is the gradient loss. We set the weighting terms to $\lambda_\sigma = 10.0$ and $\lambda_\nabla = 0.5$ to encourage both accurate depth prediction and spatial smoothness.

Video Propagation. To evaluate the upsamplers in the context of video propagation, we follow the protocol of Suri et al. [27]. We use 448×448 input images and extracted features are extracted and subsequently upscaled by a factor of 2 using the proposed upsamplers, followed by bilinear interpolation. Then we propagate labels using Suri et al. [27] protocol. Regarding the sparsity constraints, we set $k = 20$, retaining only the 20 strongest source pixels for each target pixel based on affinity scores; all lower affinity values are zeroed, and we apply a binary locality constraint with a neighborhood radius of $r = 24$. This restricts the potential source pixels to a spatial window of size $(2r+1)^2$ centered around the target pixel before the top-k selection is applied.

Open Vocabulary. We follow ProxyCLIP [17] protocol using a $\times4$ upsampling factor with 448×448 input images. Since it is direct inference, we did not change anything from the pipeline.

B.3. Visualizations

In Fig. 6, we present PCA visualizations of the upsampled feature maps produced by the evaluated methods. The feature maps generated by NAF exhibit smoother spatial variations compared to those of JAFAR [5] and AnyUp [30], which display sharper transitions, while also avoiding the halo artifacts observed in FeatUp. These smoother feature maps are reflected in the downstream linear probing results for both semantic segmentation (Fig. 7) and depth estimation (Fig. 8). For segmentation, NAF produces masks with substantially fewer sparse or fragmented artifacts compared to JAFAR and AnyUp. Likewise, the depth maps obtained with NAF exhibit markedly smoother predictions in flat regions while still preserving sharp object boundaries, outperforming Bilinear, JAFAR and AnyUp in this regard.

To further investigate how the design of NAF impacts downstream quality, we study the sensitivity of different upsamplers to color perturbations. Specifically, we add Gaussian noise ($\sigma = 0.1$) either to the image fed to the query-key branch or to the VFM branch, and inspect the resulting segmentation masks after upsampling. When noise is injected into the VFM branch, AnyUp produces highly degraded masks (JAFAR shows similar behavior, omitted for

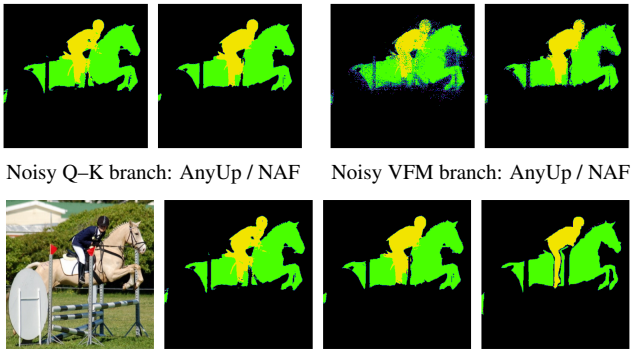


Figure 5. **Color sensitivity in semantic segmentation.** Injecting noise in different branches degrades predictions for prior methods, while NAF remains robust. *Zoom in for details.*

space), while NAF preserves much cleaner results. We attribute this robustness to NAF's design: the upsampling kernel has no pathway from low-resolution VFM features and is learned only from pixels, preventing shortcuts through VFM features and validating our design choice.

B.4. Generalization results

To better assess the quality of upsamplers across different state-of-the-art VFMs, we evaluate a wide range of models of various sizes (T–S–B–L) from different families (PE-Core [1], CAPI [7], DINO [2], PE-Spatial [1], SigLIP2 [28]), using both VFM-specific and VFM-agnostic upsamplers. on various datasets. To mitigate the computational cost of a full factorial evaluation, we adopt a randomized sampling strategy: two distinct VFMs are *randomly* assigned to each dataset. The resulting performance metrics are summarized in Tab. 1. From this analysis, we draw the following conclusions:

- Increasing the VFM input image size by a $\times 2$ factor leads to slightly higher average gains than using bilinear on standard images: +4.65 mIoU vs +4.13 mIoU.
- The larger the scaling factor, the lower the results. Using $\times 4$ factor instead of $\times 2$ leads to lower results from +4.65 mIoU gain to **-6.25 mIoU** drop. Only DINOv3-L [25] on COCO continue to have higher results when increasing image size highlighting that increasing image size can increase scores depending on models and datasets. NAF leads the average gains by **+7.46 mIoU** resulting in a +1.23 mIoU gain over next previous state of the art.

B.5. Learning representation

We investigate the following question for our framework: how does the choice of the VFM used during training affect the final upsampling performance? Although NAF is designed for zero-shot use with any VFM at inference, its Dual-Branch Encoder is optimized using features from one specific VFM during training. To analyze this, we trained NAF using features from DINOv3-B [25], DINOv2-R-B [6], and DINOv2-S [20]. The evaluation on other VFMs at inference time is detailed in Tab. 2. We find a counter-intuitive result: a VFM with strong general performance such as DINOv3-B does not necessarily yield the best results for training NAF, and we often achieve higher upsampling scores when training with smaller or less abstract representations such as DINOv2-R-S. Conversely, using raw RGB pixels (treating image upsampling as feature upsampling) caused a significant performance drop ('RGB'), confirming the need for encoded features. Furthermore, training with multiple VFMs simultaneously ('Mixture') did not improve scores (see. DINOv2-R-B), indicating that a single, appropriate representation is sufficient to efficiently guide the attention-based upsampling process. We also note that DINOv3-B performs slightly worse than DINOv2-S after the first training stage.

We retain DINOv3 due to its superior scale invariance, which proves particularly beneficial during the refinement stage (i.e., the final 2.5k higher-resolution iterations). In Tab. 3, we compare upsamplers trained (rows) and evaluated (columns) across different VFMs, with and without refinement, while keeping the input resolution fixed. We observe that NAF trained for only 25k iterations *without* any refinement stage already outperforms the official AnyUp model trained for 100k iterations *with* refinement, under identical conditions (same training data, DINOv2-S backbone, and resolution). Overall, Tab. 3 shows that the performance gains of NAF are not driven by the choice of training VFM, the use of a refinement stage, or resolution, but rather stem from the approach itself.

B.6. VFM upsamplers as filters

Instead of performing upsampling, we investigate how well learned upsamplers can function as feature filters. To do so, we apply the upsamplers using the target output resolution equal to that of the input features. In this setup, no upsampling is performed; the upsamplers effectively act as feature filters. The filtered features are subsequently upsampled using bilinear interpolation, and a linear classifier is trained on top of them. As shown in Tab. 4, NAF achieves the best average improvements compared to other filtering approaches with **+0.75 mIoU**. Although the gains are modest, they suggest a "free lunch": applying lightweight filters on top of VFMs yields additional mIoU without modifying the underlying model.

B.7. Scaling ratio

We evaluate the robustness of NAF to different upsampling ratio in Tab. 5. We take as input different image resolution (224×224, 448×448 and 896×896), feed them to the VFM and upscale the features to 448×448 resolution before

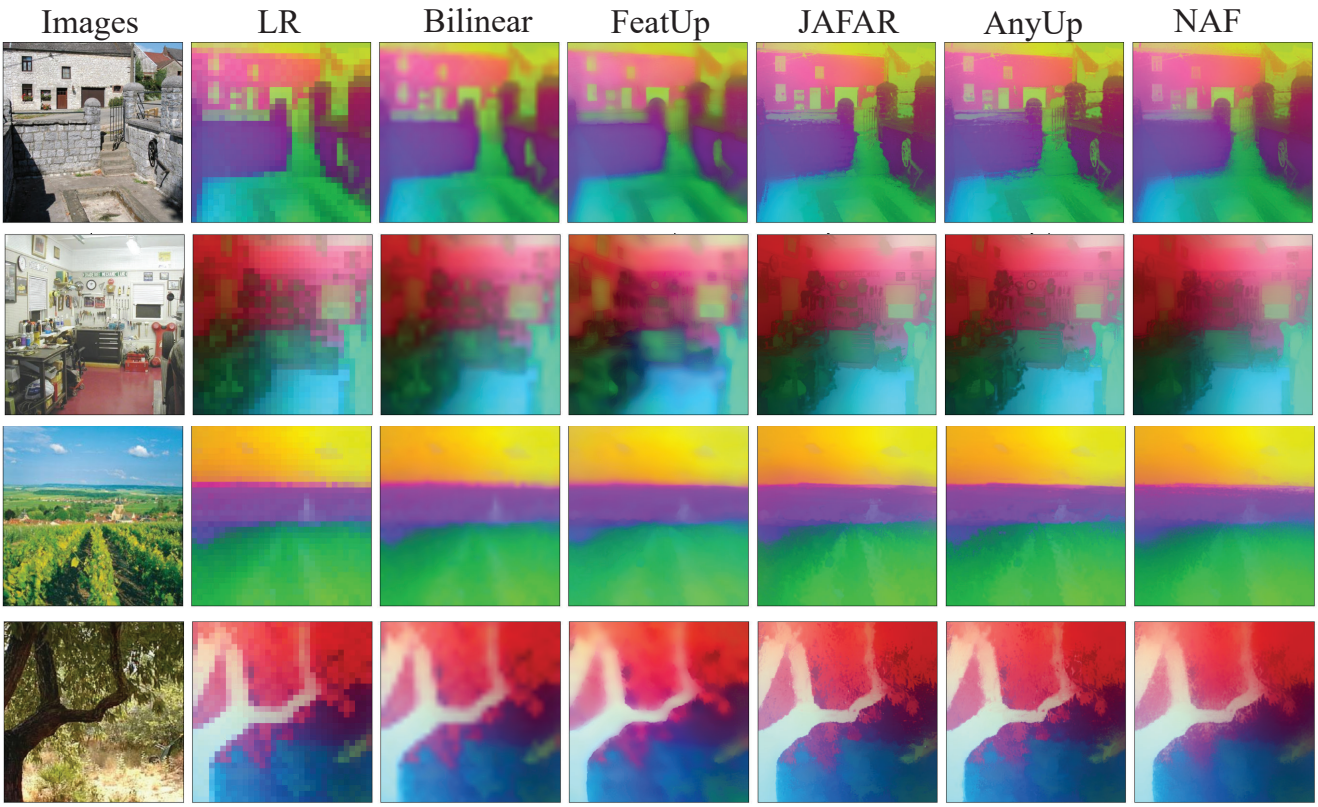


Figure 6. **PCA plots of different upsamplers for random images.** The first colum represents RGB images, the second one, the low resolution features, the others the PCA after upsampling. We use the same basis decomposition for plotting. Only JAFAR [5], AnyUp [30] and NAF produce sharp PCAs while preserving input feature representation.

		COCO [19]		VOC [8]		ADE20K [34]		Cityscapes [4]		Δ Mean
Method	**V.A**	**DINOv3-L**	**DINOv2-R-S**	**CAPI-L**	**PE-Core-S**	**DINOv2-B**	**PE-Spatial-T**	**SigLIP2-B**	**DINOv3-C-S**	
Large (×4)	✓	**67.59**	56.21	24.06	35.06	39.93	20.74	50.29	**69.21**	**-6.25**
LiFT [27]	✗	61.45	57.66	78.82	55.36	38.75	17.31	52.02	48.02	**-0.46**
Nearest	✓	64.47	57.34	76.99	59.25	39.82	22.24	45.05	47.90	0.00
Bicubic [16]	✓	66.55	58.98	79.75	62.76	41.93	23.35	49.61	54.70	+3.07
Bilinear	✓	66.61	59.77	81.37	66.20	42.85	23.89	51.29	54.14	+4.13
Large (×2)	✓	67.48	60.20	70.87	62.99	43.58	25.29	**56.98**	63.58	+4.65
FeatUp [9]	✗	66.93	60.91	82.20	68.10	44.28	24.55	51.73	51.57	+4.65
FeatSharp [22]	✗	66.85	59.80	84.10	67.35	42.90	23.61	52.75	56.84	+5.14
JAFAR [5]	✗	66.81	61.92	84.03	**74.85**	44.94	23.85	52.25	50.70	+5.79
AnyUp [30]	✓	66.54	61.58	84.00	73.66	44.44	24.68	53.58	54.40	+6.23
NAF	✓	67.35	**62.17**	**84.64**	74.47	**45.39**	**25.08**	**56.98**	56.69	**+7.46**

Table 1. **Semantic Segmentation (mIoU ↑) on Random Combinations of VFMs and datasets.** VFMs come from different sizes and families and are evaluated on many datasets: COCO [19], Pascal VOC [8], ADE20K [34]. '**Δ Mean**' is computed against Nearest. We highlight **best** and second best scores, and **best gain**. **V.A** indicates VFM-agnostic models.

learning the linear probing classifier.

NAF always leads to the best or second best score regardless of the scaling ratio, proving that it can be used across a wide range of resolutions. Interestingly, as already mentioned, for some VFM (e.g., PE-Core-B) feeding larger images does not lead to higher scores. Nonetheless, NAF still improves the mIoU of degraded representations.

C. Image Restoration

C.1. Evaluation Setup

We evaluate several image denoising models — DNCNN [32], IRCNN [33], RedNet [15], and Restormer [31] — on

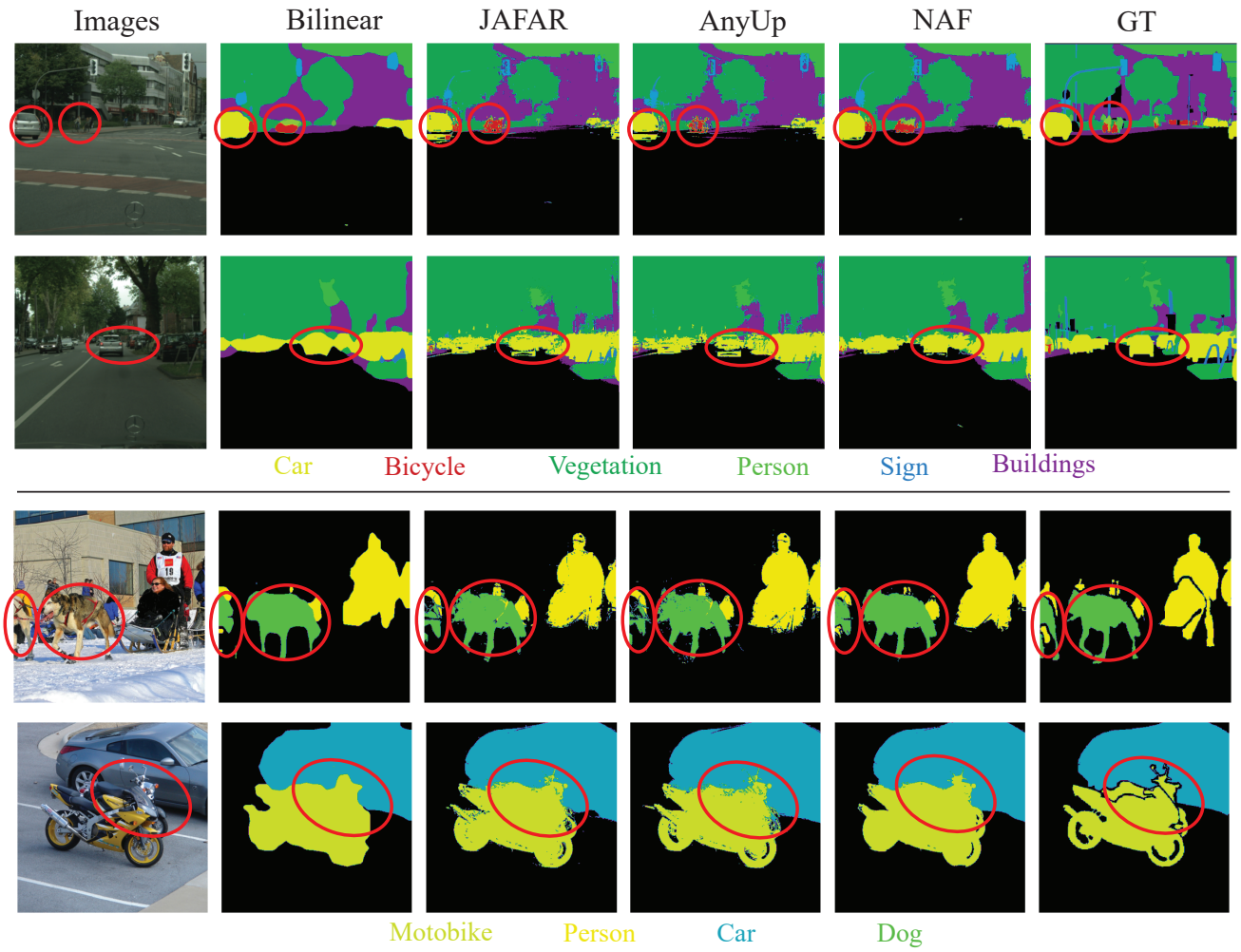


Figure 7. **Segmentation predictions using different upsamplers.** The first two rows are on Cityscapes [4], the last two rows on VOC [8]. We see that while being VFM-agnostic NAF better preserves structure compared to JAFAR [5] and AnyUp [30] that tend to focus too much on colors leading to noisy semantics.

Backbone	DINOv2-R-B	RADIOv2.5-B	Franca-B	DINOv3-B	Mean
RGB	58.36	54.79	54.52	60.40	57.02
No training	60.46	56.53	56.26	61.92	58.79
DINOv3-B	63.82	58.76	58.03	61.01	60.41
DINOv2-S	64.02	58.98	58.24	61.06	60.58
Mixture	64.16	59.18	58.29	61.73	60.84
DINOv2-R-B	65.25	59.86	59.24	62.54	61.72

Table 2. **Segmentation (mIoU ↑) on Cityscapes [4]**, training NAF with different backbones. The best average score is highlighted in orange, and the standard training configuration used for NAF is indicated in blue.

ImageNet for 25k steps using input images of size 448×448. We select the largest batch size that fits on a single A100 40GB GPU: $B = 32$ for the convolutional models and $B = 1$ for Restormer [31].

The denoisers are trained with a combination of three loss terms: L1, L2, and SSIM. Denoting the total loss as

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{L1}} + \lambda_2 \mathcal{L}_{\text{L2}} + \lambda_3 \mathcal{L}_{\text{SSIM}}, \tag{19}$$

we set the weights to $\lambda_1 = 1.0$, $\lambda_2 = 5.0$, and $\lambda_3 = 0.2$. To train NAF we keep the same architecture than for feature upsampling but we increase the neighborhood kernel size from 9 to 15 to take into account the receptive field difference.

C.2. Visualizations

To evaluate the denoiser's performance, we apply noise to a set of clean 448×448 images and feed them to NAF. In Fig. 9, we test models trained on dynamic ranges of Gaussian noise $\sigma \in [0.1, 0.5]$ and salt-and-pepper noise $p \in [0.1, 0.5]$. We observe that the model can effectively de-

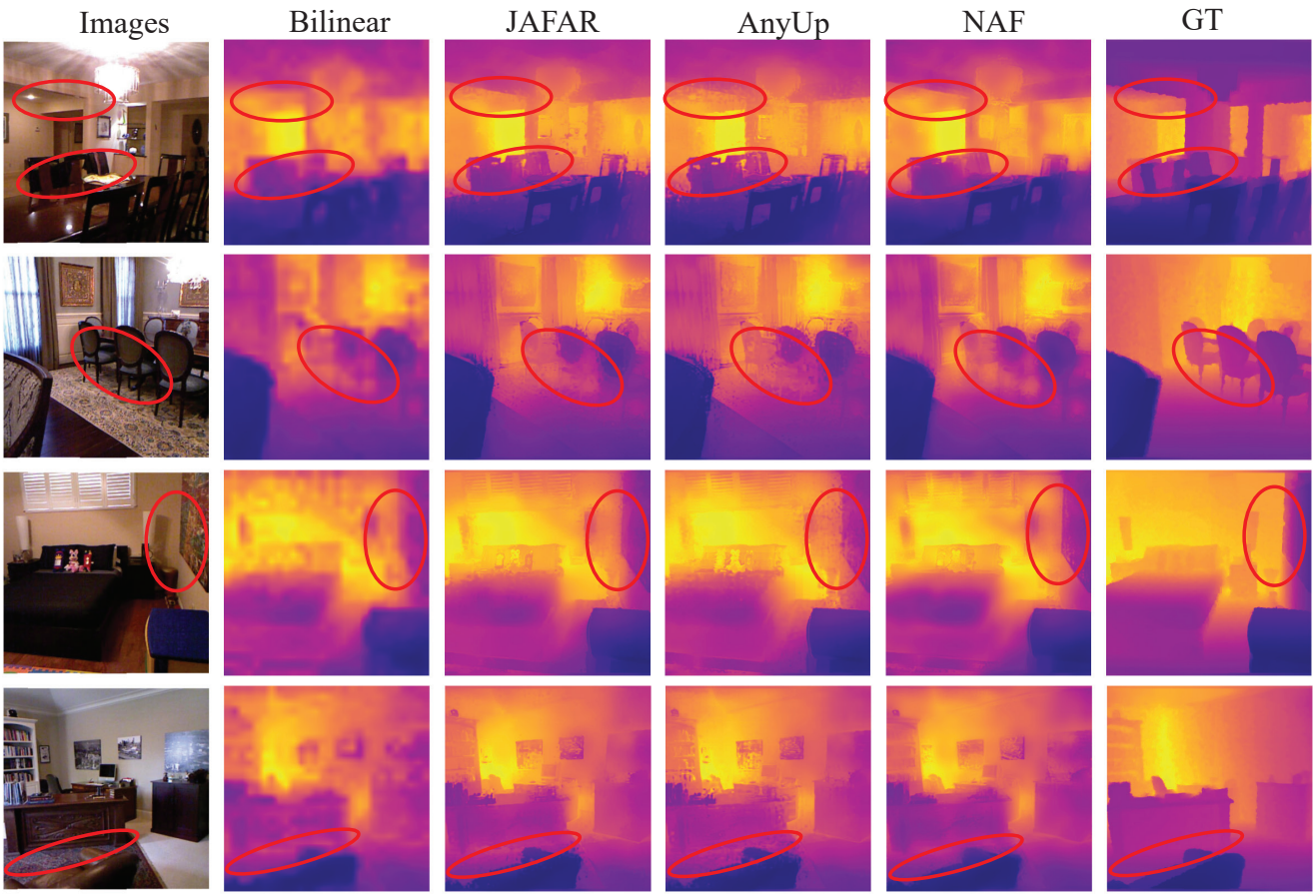


Figure 8. **Detph Estimation using different upsamplers** on NYUv2 [24]. Compared to AnyUp [30], NAF outputs smoother predictions without the noisy effect we observe on some regions using AnyUp.

noise channel-wise salt-and-pepper noise even beyond the maximal training range (rightmost image uses 0.6, while the model has been trained up to 0.5 noise intensity), while achieving high-quality reconstructions for other noise levels as well.

D. Computation footprint

We previously provided initial insights into the efficiency of different upsamplers in Tab. 1, where NAF is the only approach capable of achieving a $\times 72$ upsampling ratio, producing features at a resolution of 2048×2048. We now investigate in greater detail the behavior of these upsamplers when targeting different scaling factors or when processing higher-dimensional feature maps (Fig. 10, Fig. 11). To this end, we initialize a dummy feature tensor of size $(28, 28, 384)$, corresponding to the output of a standard model processing 448×448 input images. We then conduct two controlled studies: (i) varying the embedding dimension while keeping a fixed $\times 16$ upsampling ratio (Fig. 10), and (ii) varying the upsampling ratio while maintaining 384 channels (Fig. 11). For each configuration, we evaluate the total number of parameters (in millions, M), the computational cost (GFLOPs), the time required for the forward pass (relevant for inference) and backward pass (relevant for training), as well as the peak memory consumption during both forward and backward passes.

Although JAFAR [5] and AnyUp [30] are attention-based models like NAF, NAF achieves substantially higher memory and computational efficiency. In the embedding study, it provides an approximately 2–3$\times$ speedup and memory reduction compared to AnyUp. For low upsampling ratios, AnyUp is slightly more efficient; however, its efficiency degrades rapidly with larger upsampling factors, resulting in an approximately $3\times$ advantage for NAF at high upscaling levels. Furthermore, processing larger input images with a standard state-of-the-art VFM leads to a substantial increase in GFLOPs and runtime, whereas NAF maintains significantly lower computational cost, as shown in Tab. 6.

Across many configurations, the efficiency of NAF is comparable to that of FeatUp [9], which relies on convo-

Method	Backbone	Refinement	DINOv3-B (512×512)	DINOv2-S (448×448)
Pretrained (official checkpoints)				
JAFAR	Mixed VFMs	–	62.46 / 87.10	61.10 / 83.85
AnyUp	DINOv2-S	Image crops	60.35 / 86.63	60.85 / 84.22
Retrained on DINOv3-B				
JAFAR	DINOv3-B	–	60.84 / 86.63	–
AnyUp	DINOv3-B	–	59.35 / 86.42	56.36 / 82.68
NAF	DINOv3-B	–	61.01 / 86.60	61.34 / 84.25
NAF	DINOv3-B	Higher-res	**64.98 / 87.86**	**63.78 / 84.52**
Retrained on DINOv2-S				
JAFAR	DINOv2-S	–	–	61.72 / 84.01
AnyUp	DINOv2-S	–	58.93 / 86.09	58.67 / 83.60
NAF	DINOv2-S	–	61.06 / 86.85	61.78 / 84.42
NAF	DINOv2-S	Higher-res	64.38 / 87.67	63.14 / 84.42

Table 3. **Semantic segmentation (IoU ↑)** results on Cityscapes/VOC. The training configuration kept in the paper for training NAF is indicated in blue. **Best scores** are highlighted in **bold**. DINOv2-S does not benefit a lot from the refinement stage, whereas DINOv3-B, owing to its stronger scale invariance, consistently gains from it.

Method	V.A	DINOv2-R	RADIO	Franca	DINOv3	Δ Mean
JAFAR [5]	✗	84.47	84.76	81.63	86.26	+0.32
AnyUp [30]	✓	84.10	84.59	81.69	86.62	+0.29
Bilinear	✓	83.07	84.47	81.30	**86.99**	0.00
NAF (ours)	✓	**84.90**	**85.05**	**82.01**	86.88	**+0.75**

Table 4. **Semantic Segmentation (mIoU ↑) on VOC [8] using VFM-upsamplers as feature filters.** We use base VFMs: DINOv2-R-B [6], RADIOv2.5-B [14], Franca-B [29], and DINOv3-B [25]. **Δ Mean** is computed relative to Bilinear. **Bold** and underline indicate the best and second-best scores, respectively, while **highlighted** indicates the largest gain. **V.A** denotes VFM-agnostic models.

Method	Radiov2.5-B			PE-Core-B		
	14 → 448	28 → 448	56 → 448	14 → 448	28 → 448	56 → 448
AnyUp	53.85	62.04	OOM	**54.92**	58.64	OOM
Bilinear	55.85	66.63	68.74	53.54	56.83	43.81
Nearest	49.10	61.34	66.42	45.12	49.98	39.03
JAFAR	54.51	62.33	OOM	51.02	51.95	OOM
NAF	**56.31**	**67.02**	**69.04**	54.47	**61.06**	**48.44**

Table 5. **Semantic Segmentation (mIoU ↑) on Cityscapes [4] for different Upsampling-ratio**. We compare different upsamplers for generating 448×448 features from various feature input resolutions. The first three columns correspond to RADIOv2.5-B [14], and the last three columns to PE-Core-B [3]. **Bold** indicates the best score, and underline the second-best. **OOM** indicates linear probing-training 'Out-of-Memory'.

lutions but is intrinsically constrained to fixed output resolutions. In contrast, NAF combines the flexibility of an any-scale upsampler with the computational efficiency characteristic of convolution-based designs, while consistently outperforming attention-based alternatives.

Finally, we report efficiency results for the full pipeline,

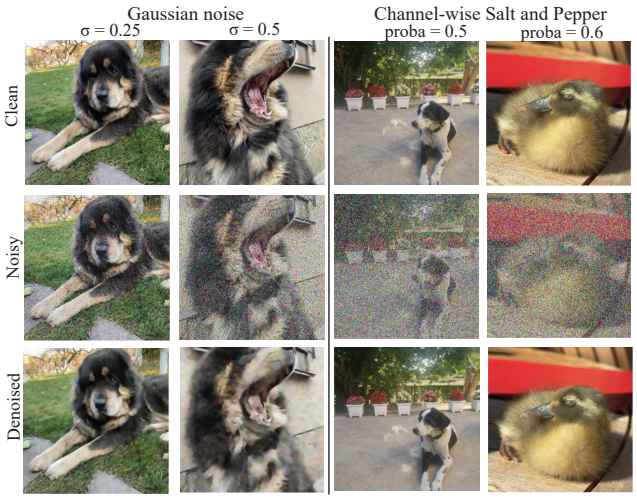


Figure 9. **Image restoration using NAF**. On the left two images we apply a gaussian noise. On the right we apply a channel-wise salt and pepper noise. NAF allows to restore very noisy images even on unseen noise range (rightmost image).

Method	GFLOPs			Time (ms)		
	×2	×4	×8	×2	×4	×8
Large	537	2148	8591	110	1035	6555
NAF	4	16	66	39	40	42

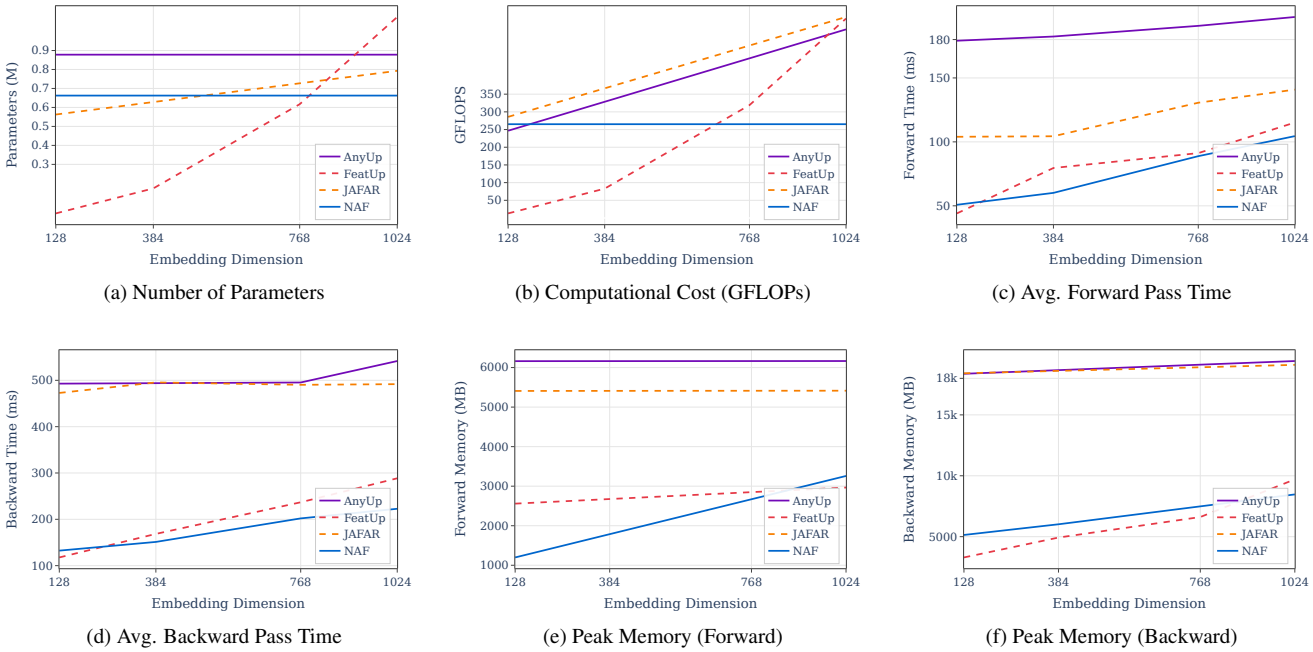
Table 6. **Comparison of NAF and the Large Image baseline.** We measure GFLOPs and inference time required to produce features at ×2, ×4, and ×8 the original resolution of DINOv3-B [25] outputs. For the Large Image baseline, this corresponds to feeding images scaled by the same factors, relative to the standard 448×448 resolution.

including feature extraction. Instead of relying on dummy inputs, Tab. 7 evaluates performance by processing a real image with a VFM and upsampling the resulting features by a factor of ×2. While bilinear upsampling remains the most efficient approach, it yields relatively modest IoU improvements. In contrast, NAF provides the best trade-off among learnable methods, combining strong efficiency with the highest average IoU gain. It outperforms both convolutional approaches [9] and attention-based methods [5, 30], which we attribute to its efficient localized attention mechanism.

E. Limitations and Perspectives

Compared to both VFM-specific and VFM-agnostic upsamplers, NAF achieves state-of-the-art performance across multiple datasets and tasks. Nevertheless, several avenues remain for improvement.

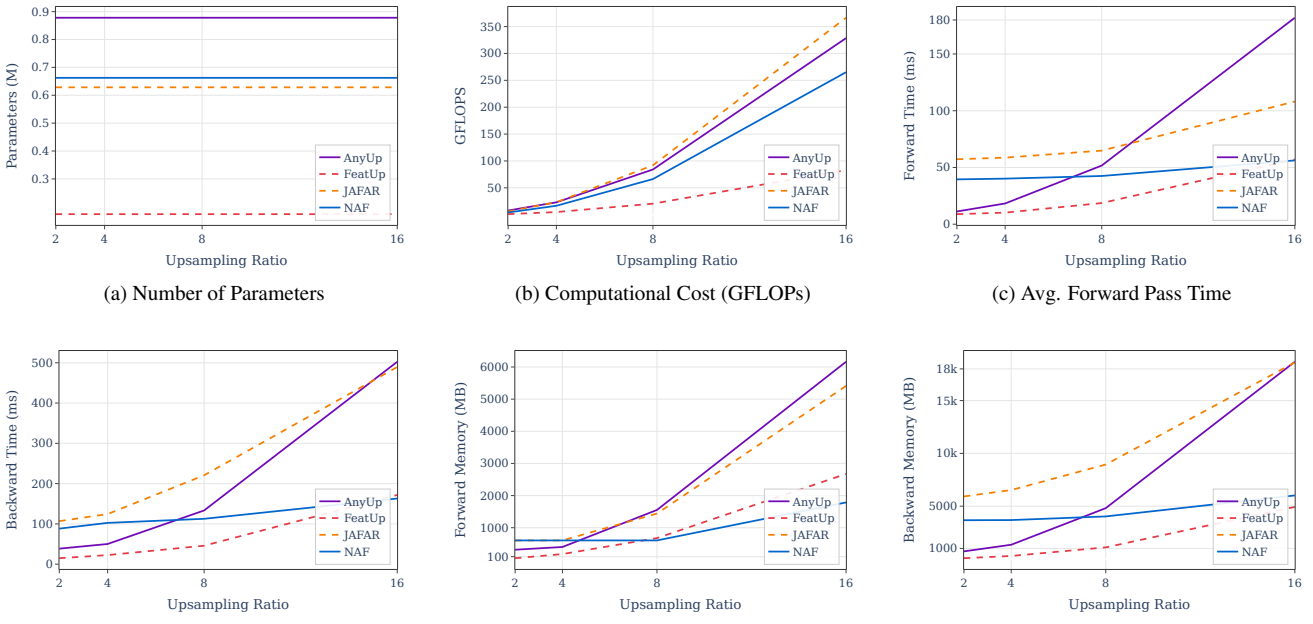
By design, NAF employs a neighborhood-attention mechanism with a fixed kernel size. We observe that the attention maps vary depending on the query point (see.



(a) Number of Parameters (b) Computational Cost (GFLOPs) (c) Avg. Forward Pass Time

(d) Avg. Backward Pass Time (e) Peak Memory (Forward) (f) Peak Memory (Backward)

Figure 10. **Benchmarking analysis across embedding dimensions.** We study 4 different standard embedding sizes: 128, 384, 768 and 1024. (a)-(b) show model complexity, (c)-(d) compare execution time, and (e)-(f) analyze memory consumption.



(a) Number of Parameters (b) Computational Cost (GFLOPs) (c) Avg. Forward Pass Time

(d) Avg. Backward Pass Time (e) Peak Memory (Forward) (f) Peak Memory (Backward)

Figure 11. **Benchmarking analysis across upscaling ratio.** We studied different upscaling ratio: $\times 2$, $\times 4$, $\times 8$, $\times 16$. (a)-(b) show model complexity, (c)-(d) compare execution time, and (e)-(f) analyze memory consumption.

Fig. 4). Introducing dynamic kernel adaptation—similar to approaches in Deformable Attention or Deformable Convolutions—could, in principle, reduce the computational cost per interpolation step while potentially enhancing reconstruction accuracy by introducing sampling flexibility.

Although our method is VFM-agnostic, we currently lack a principled framework for identifying which VFMs provide the most informative patch representations for

Method	Learnable	Memory (MiB) ↓	FPS ↑	Δ IoU ↑
FeatUp	✓	3765	10	+3.29
Bilinear	✗	**1517**	**50**	+3.34
AnyUp	✓	7080	4	+4.09
JAFAR	✓	6330	7	+5.12
NAF	✓	3600	15	**+5.58**

Table 7. **Efficiency comparison using DINOv3-B**. Δ IoU denotes the average improvement over the baseline (see Main Tab. 2). Results are sorted by average gain, then by FPS.

learning the upsampling. Closing this gap requires a deeper understanding of the representational properties that support zero-shot consistent upsampling. Empirically, we observe that neither combining multiple VFMs nor using larger or stronger VFMs leads to clear gains (see. Tab. 2).

In terms of applications, looking ahead, the ability to preserve high-resolution spatial representations is especially valuable in medical imaging and remote sensing, highlighting promising avenues for future research. In addition, we have demonstrated that NAF's architecture is versatile and can be adapted across domains, particularly within the denoising context, paving the way to other applications such as in image restoration.

References

[1] Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. *arXiv:2504.13181*, 2025. 5

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 5

[3] Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Hanoona Rasheed, Peize Sun, Po-Yao Huang, Daniel Bolya, Suyog Jain, Miguel Martin, Huiyu Wang, Nikhila Ravi, Shashank Jain, Temmy Stark, Shane Moon, Babak Damavandi, Vivian Lee, Andrew Westbury, Salman Khan, Philipp Krähenbühl, Piotr Dollár, Lorenzo Torresani, Kristen Grauman, and Christoph Feichtenhofer. Perceptionlm: Open-access data and models for detailed visual understanding. *arXiv:2504.13180*, 2025. 9

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 4, 6, 7, 9

[5] Paul Couairon, Loick Chambon, Louis Serrano, Jean-Emmanuel Haugeard, Matthieu Cord, and Nicolas Thome. Jafar: Jack up any feature at any resolution. In *NeurIPS*, 2025. 3, 4, 6, 7, 8, 9

[6] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024. 5, 9

[7] Timothée Darcet, Federico Baldassarre, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Cluster and predict latents patches for improved masked image modeling. *TMLR*, 2025. 5

[8] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 4, 6, 7, 9

[9] Stephanie Fu, Mark Hamilton, Laura E. Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. Featup: A model-agnostic framework for features at any resolution. In *ICLR*, 2024. 6, 8, 9

[10] Ali Hassani and Humphrey Shi. Dilated neighborhood attention transformer. *arXiv preprint arXiv:2209.15001*, 2022. 4

[11] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[12] Ali Hassani, Wen-Mei Hwu, and Humphrey Shi. Faster neighborhood attention: Reducing the o(n2) cost of self attention at the threadblock level. In *Advances in Neural Information Processing Systems*, 2024.

[13] Ali Hassani, Fengzhe Zhou, Aditya Kane, Jiannan Huang, Chieh-Yun Chen, Min Shi, Steven Walton, Markus Hoehnerbach, Vijay Thakkar, Michael Isaev, et al. Generalized neighborhood attention: Multi-dimensional sparse attention at the speed of light. *arXiv preprint arXiv:2504.16922*, 2025. 4

[14] Greg Heinrich, Mike Ranzinger, Hongxu Yin, Yao Lu, Jan Kautz, Andrew Tao, Bryan Catanzaro, and Pavlo Molchanov. Radiov2.5: Improved baselines for agglomerative vision foundation models. In *CVPR*, 2025. 9

[15] Jindong Jiang, Lunan Zheng, Fei Luo, and Zhijun Zhang. Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation. *arXiv preprint arXiv:1806.01054*, 2018. 6

[16] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 2003. 6

[17] Mengcheng Lan, Chaofeng Chen, Yiping Ke, Xinjiang Wang, Litong Feng, and Wayne Zhang. Proxyclip: Proxy attention improves clip for open-vocabulary segmentation. In *ECCV*, 2024. 4

[18] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *Pattern Analysis and Machine Intelligence (PAMI)*, 2022. 4

[19] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 4, 6

[20] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 5

[21] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *CoRR*, abs/1704.00675, 2017. 4

[22] Mike Ranzinger, Greg Heinrich, Pavlo Molchanov, Bryan Catanzaro, and Andrew Tao. Featsharp: Your vision model features, sharper. In *ICML*, 2025. 6

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 4

[24] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012. 4, 8

[25] 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. 5, 9

[26] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 1, 2

[27] Saksham Suri, Matthew Walmer, Kamal Gupta, and Abhinav Shrivastava. Lift: A surprisingly simple lightweight feature transform for dense vit descriptors. In *ECCV*, 2024. 4, 6

[28] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025. 5

[29] Shashanka Venkataramanan, Valentinos Pariza, Mohammadreza Salehi, Lukas Knobel, Spyros Gidaris, Elias Ramzi, Andrei Bursuc, and Yuki M Asano. Franca: Nested matryoshka clustering for scalable visual representation learning. *arXiv preprint arXiv:2507.14137*, 2025. 9

[30] Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. Anyup: Universal feature upsampling. *arXiv preprint arXiv:2510.12764*, 2025. 3, 4, 6, 7, 8, 9

[31] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022. 6, 7

[32] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *TIP*, 2017. 6

[33] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *CVPR*, 2017. 6

[34] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 4, 6