# DIFFERENTIAL FEATURE UPSAMPLING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We present DiffUp, a novel feature upsampler that exploits differential kernels to enhance details. Since feature upsamplers evolve from hand-crafted to content-aware designs, they excel at detail preservation yet still produce ambiguous boundaries. In this work, we uncover a boundary suppression phenomenon behind the de facto use of softmax kernel normalization: positive-only weights reduce boundary sharpness. Inspired by zero-mean filters, we introduce signed weights and differential kernels to amplify the differences. However, introducing signed weights alone cannot guarantee the amplification behavior; their spatial distribution also matters. We show that the same differential formulation can enhance details while also executing local averaging. We therefore establish conditional guarantees for when and where the behaviors can take place. By embedding our insights into DiffUp, we show through controlled experiments and four dense prediction tasks that DiffUp outperforms or matches state-of-the-art dynamic upsamplers. Notably, it improves the boundary IoU on COCONut by up to 4.37% over bilinear interpolation. Our approach suggests yet another design principle for modern feature upsamplers, i.e., amplification over mere preservation.

## 1 INTRODUCTION

Feature upsamplers are fundamental components in modern dense prediction models (Badrinarayanan et al., 2017; Chen et al., 2022; Xie et al., 2021; Xu et al., 2017; Li et al., 2023) requiring high-res predictions. In pursuit of generic, task-agnostic feature upsampling, existing upsamplers have increasingly moved from fixed-rule interpolation to content-aware paradigms, a.k.a. dynamic upsampling (Wang et al., 2019; Lu et al., 2022a;b; Liu et al., 2023). At the core of dynamic upsampling is to learn an input-dependent mapping from low-res features to high-res ones (Wang et al., 2019; Lu et al., 2022a;b). However, we empirically observe a consistent pattern among prior arts where upsampled feature maps still suffer from boundary ambiguity, even with detail preservation and high-res guidance (Lu et al., 2019; 2020; Dai et al., 2021; Lu et al., 2022a; 2024; 2022b). This raises a fundamental question: Why does boundary ambiguity persist despite increasingly content-aware and detail-sensitive upsampling designs?

To answer this question, we delve into their implementations and make an apples-to-apples comparison between them. Our observation is that, although they vary in how upsampling kernels are formalized and what input features are used for kernel generation (Wang et al., 2019; Lu et al., 2019; 2020; Dai et al., 2021; Lu et al., 2022a; 2024; 2022b), they ultimately follow the same procedure of softmax normalization. Inspecting this normalization reveals a commonsensical but counterfactual finding: softmax produces only positive weights; but we show that *positive-only feature assembly by theory will weaken boundary sharpness*. We refer to this phenomenon as *local boundary suppression*. This phenomenon inspires us to rethink the pursuit of detail preservation in dynamic upsampling. Indeed once fine-grained details have been attenuated or discarded by downsampling, simple preservation seems not enough. What is missing is the ability to actively rebuild informative local differences. Inspired by zero-mean filters, we introduce Differential Feature Upsampling (DiffUp), which uses signed weights and differential kernels to amplify local differences.

Yet introducing signed weights alone does not automatically yield boundary sharpness; their spatial distribution also matters. Semantic segmentation provides an intuitive example. Near an object boundary, features from different semantic regions should be distinguishable; inside a smooth region, neighboring features should instead remain consistent. This raises a natural concern for differential upsampling: if differentiation sharpens boundaries, will it also amplify unimportant tex-



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.

tures or details within regions? To address this concern, DiffUp constructs a differential kernel with explicit zero-mean and magnitude constraints through mean centering and $\ell_1$ normalization, and combines it with an identity base kernel to maintain the magnitude consistency. We theoretically show that our differential formulation can support two complementary behaviors conditionally: i) near semantic boundaries, signed weights can enlarge feature differences across the boundary; ii) within smooth regions, the differential kernel can instead exhibit a center-negative pattern that yields a local averaging effect. Our controlled experiments and kernel visualizations further confirm that learned DiffUp kernels conform our expectations.

Beyond controlled experiments, the next step is to examine whether enhanced boundary sharpness can translate into improved task performance. We therefore evaluate DiffUp across four dense prediction tasks. On COCONut (Deng et al., 2024), DiffUp consistently improves boundary quality on SegFormer-B1 (Xie et al., 2021), increasing boundary Intersection over Union (bIoU) (Cheng et al., 2021) by 4.37% over bilinear upsampling. On P3M-500-NP (Li et al., 2021) for image matting, DiffUp reduces SAD by 6.3% relative to DySample. DiffUp also delivers improvements on NYUv2 (Silberman et al., 2012) for depth estimation and VisDrone 2019-DET (Zhu et al., 2022) for object detection. In a nut shell, our findings suggest that amplification over mere preservation seems yet another design principle for modern feature upsamplers. More broadly, the community may rethink whether positive-only aggregation should remain the default choice in other fundamental operators such as self- and cross-attention.

## 2 Related Work

We review existing feature upsamplers and summarize the underlying design principles behind them. We also discuss a recent complementary direction, *i.e.*, model-agnostic feature upsampling.

**Empirical Design Principles of Upsamplers.** Existing upsamplers have identified a number of useful design principles to improve upsampling. Early interpolation-based upsamplers (*e.g.*, nearest neighbor and bilinear interpolation) establish i) *aggregation of neighborhood information* for upsampled point generation. Deconvolution (Noh et al., 2015) and PixelShuffle (Shi et al., 2016) incorporate ii) *learnable parameters* and *cross-channel feature aggregation*. CARAFE (Wang et al., 2019) emphasizes iii) *content-aware kernel generation*. IndexNet (Lu et al., 2019; 2020) and A2U (Dai et al., 2021) additionally advocate for iv) *high-res guidance* such as encoder features, particularly for detail recovery. FADE (Lu et al., 2022a; 2024) integrates the idea of CARAFE and IndexNet, highlighting that v) *using both encoder and decoder features* for kernel generation is the key to task-agnostic upsampling. SAPA (Lu et al., 2022b) and LDA (Du et al., 2024) leverage vi) *local cross-attention* for similarity-aware feature aggregation, showing the benefit of simultaneous detail preserving and region smoothing. DySample (Liu et al., 2023) adopts vii) *deformable point sampling* and achieves ultra-lightweight upsampling. FreqFusion (Chen et al., 2024) formulates viii) *frequency-aware feature fusion* that jointly models upsampling and feature fusion. Despite these advances, they still produce ambiguous boundaries, even with detail preservation and high-res guidance. This suggests existing design principles remain deficient. In this work, we introduce a novel design principle to sharpen boundaries.

**Model-Agnostic Feature Upsampling.** Aside from the upsamplers above, model-agnostic upsamplers, pioneered by FeatUp (Fu et al., 2024), are mainly developed in the era of Vision Foundation Models (VFMs), responding to the need of upsampling frozen pre-trained features. Due to the unavailability of high-res encoder features from VFMs, raw images are often encoded to guide upsampling, and self-supervised signal is typically employed. Unlike prior upsamplers that require joint training with backbone networks, model-agnostic upsamplers can be standalone as independent networks. Such designs enable them to work with any off-the-shelf VFMs, yet at the cost of relatively high computational overhead. In this line of work, subsequent work (Ranzinger et al., 2025; Huang et al., 2025; Couairon et al., 2025; Wimmer et al., 2026; Seo et al., 2025; Chambon et al., 2025; Walmer et al., 2026) has introduced various refinements, several of which reflect the design principles discussed above. For example, LoftUp (Huang et al., 2025) and JAFAR (Couairon et al., 2025) use high-res feature supervision, echoing the principle of *high-res guidance*, while NAF (Chambon et al., 2025) and UPLiFT (Walmer et al., 2026) also highlight the role of *neighborhood information*. However, these upsamplers also suffer from the boundary ambiguity discussed above. DiffUp as yet another design principle may offer new insights for future model-agnostic upsampling.

# 3 DIFFERENTIAL UPSAMPLING

Here we revisit fundamental concepts in dynamic upsampling, analyze the limitations of softmax normalization, and introduce our DiffUp, followed by probing its working mechanism.

## 3.1 PRELIMINARY

Given a low-res feature map, often from decoding layers, say $\mathcal{X}^d \in \mathbb{R}^{C\times H\times W}$, feature upsampling upsamples $\mathcal{X}^d$ to $\mathcal{X}' \in \mathbb{R}^{C\times \sigma H\times \sigma W}$, where $\sigma$ is the upsampling ratio. When high-frequency details are expected, upsampling can incorporate an additional high-res cue, usually from shallow encoder layers, say $\mathcal{X}^e \in \mathbb{R}^{C\times \sigma H\times \sigma W}$. For each upsampled target location $l' = (i', j')$, let $l = (i, j) = (\lfloor i'/\sigma \rfloor, \lfloor j'/\sigma \rfloor)$ denote its corresponding location in the low-res map. We use $\mathcal{N}_l^d(k)$ and $\mathcal{N}_{l'}^e(k)$ to denote the $k \times k$ neighborhoods centered at $l$ in $\mathcal{X}^d$ and at $l'$ in $\mathcal{X}^e$, respectively.

**Kernel Generation.** The content-aware kernel generation mechanism dynamically produces kernel weights through two configurations:

$$\mathcal{W}_{l'} = \begin{cases} \psi(\mathcal{N}_l^d(k_g)), & \text{(for decoder-only)} \\ \psi(\mathcal{N}_l^d(k_g), \mathcal{N}_{l'}^e(k_g)), & \text{(for encoder-guided)} \end{cases} \tag{1}$$

where $\psi(\cdot)$ denotes the kernel generation function, $k_g$ is the window size of the neighborhood used by $\psi(\cdot)$, and $\mathcal{W}_{l'}$ is the unnormalized kernel.

**Feature Reassembly.** Let $k_r$ denote the odd-sized reassembly window width and $r = (k_r - 1)/2$ its radius. For each location $l'$, dynamic upsampling reassembles features from the $k_r \times k_r$ decoder neighborhood centered at its corresponding low-res location $l = (i, j)$:

$$\mathcal{X}'_{l'} = \sum_{n=-r}^{r} \sum_{m=-r}^{r} \mathcal{K}_{l'}(n, m) \cdot \mathcal{X}^d(i + n, j + m), \tag{2}$$

where $\mathcal{K}_{l'} = f_n(\mathcal{W}_{l'}) \in \mathbb{R}^{k_r\times k_r}$ is the normalized kernel, and $f_n(\cdot)$ a normalization function.

## 3.2 LIMITATION OF SOFTMAX NORMALIZATION

Here we show that the conventional softmax normalization entails an inherent drawback: it reduces boundary sharpness after upsampling. We dissect this effect and offer a mathematical analysis.

**Definition 1 (Boundary Point).** Given an intensity transformation $I : \mathcal{X}^d \cup \mathcal{X}' \to \mathbb{R}$ that maps spatial positions to feature values, we define two adjacent pixels $A$ with $I(A) = a$ and $B$ with $I(B) = b$ that form an ideal boundary, and $A' \in \mathcal{X}'$ and $B' \in \mathcal{X}'$ denote their corresponding upsampled positions (Fig. 1). Without loss of generality, we assume $a < b$.

**Assumption 1 (Binary Partition and Softmax Normalization).** i) Per Definition 1, the boundary neighborhood constitutes a binary partition of the feature space within the reassembly regions of $A$ and $B$, satisfying $I(o) \in \{a, b\}$ for every $o \in \mathcal{N}_A^d(k_r) \cup \mathcal{N}_B^d(k_r)$. ii) The reassembly kernels are softmax-normalized. Thus, their weights are strictly positive and sum to one at every target location $l'$:

$$\mathcal{K}_{l'}(p) > 0, \qquad \sum_{p=1}^{k_r^2} \mathcal{K}_{l'}(p) = 1. \tag{3}$$

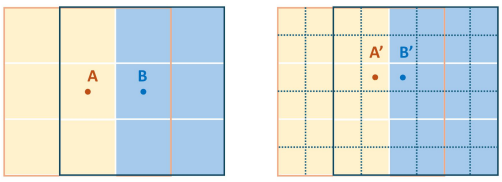



Figure 1: **The case of idealized boundary points before and after upsampling**. Adjacent points $A$ and $B$ lie on opposite sides of a boundary in the low-res decoder feature map $\mathcal{X}^d$ (left), and $A'$ and $B'$ are their corresponding upsampled counterparts in $\mathcal{X}'$ (right). Under binary partition, the two colored regions have constant feature values.

**Theorem 1 (Local Boundary Suppression).** Suppose that Assumption 1 holds, and the upsampled feature map $\mathcal{X}'$ is produced following the feature reassembly equation. For point pairs $(A, B) \in \mathcal{X}^d$ forming a semantic boundary with $I(A) = a < I(B) = b$, their upsampled counterparts $(A', B') \in \mathcal{X}'$, which also form a semantic boundary in $\mathcal{X}'$, satisfy

$$|I(B') - I(A')| < |I(B) - I(A)|. \tag{4}$$

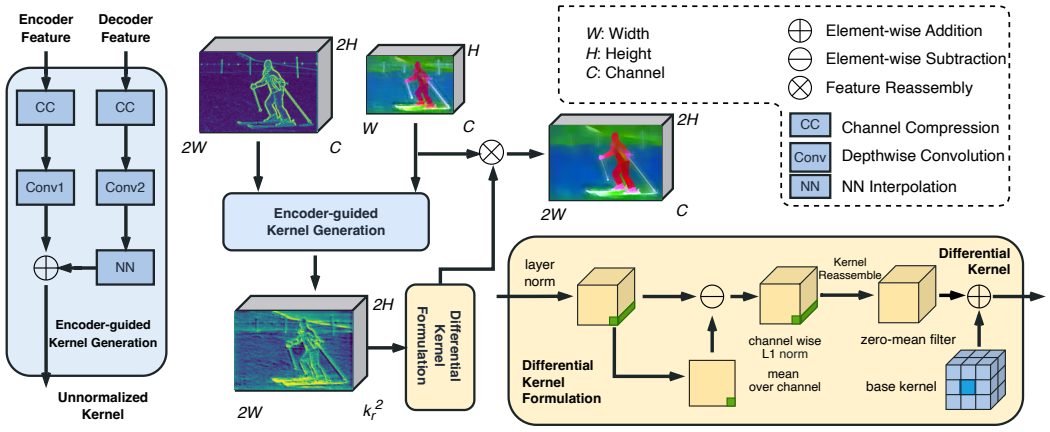



Figure 2: **Overview of DiffUp**. The high-res encoder feature and low-res decoder feature jointly generate an unnormalized kernel, which is converted into a differential kernel and combined with an identity base kernel for feature reassembly.

See Appendix D.1 for the proof and discussion of the boundary conditions.

Theorem 1 suggests an easily overlooked limitation in dynamic upsampling: softmax promotes intra-semantic consistency yet suppresses local semantic boundaries due to probability competition. This inherently conflicts with the objective of dynamic upsamplers—simultaneous detail delineation and semantic preservation—leading to progressive boundary degradation in decoder layers.

### 3.3 Design of Differential Upsampling

**Design Goals.** Given the local boundary suppression of positive-only aggregation, we design DiffUp with three complementary goals in mind: i) high-res guidance, using high-res encoder features to guide where and how low-res decoder features are reassembled; ii) dynamic, conditional sharpening and smoothing, amplifying differences across semantic boundaries while retaining local averaging within regions; and iii) computational efficiency, introducing the capabilities above efficiently.

As illustrated in Fig. 2, DiffUp features two key designs: a differential kernel formulation and an encoder-guided kernel generation mechanism.

**Differential Kernel Formulation.** The analyses above suggest that escaping the positive-only assembly induced by softmax requires an upsampling kernel with signed weights. Meanwhile, its differential component should have zero sum such that it can modify local feature contrasts without changing the response to constant features. We adopt a formulation with explicit zero-mean and magnitude constraints. Specifically, for each target location $l'$, we subtract the mean of the predicted kernel weights $\mathcal{W}_{l'}$ from each entry:

$$\overline{\mathcal{W}}_{l'} = \mathcal{W}_{l'} - \frac{1}{k_r^2}\sum_{q=1}^{k_r^2} \mathcal{W}_{l'}(q)\,. \tag{5}$$

We then obtain the differential kernel through $\ell_1$ normalization:

$$\mathcal{K}^{\text{diff}}_{l'} = \frac{2\overline{\mathcal{W}}_{l'}}{\sum_{q=1}^{k_r^2} \left|\overline{\mathcal{W}}_{l'}(q)\right| + \epsilon}\,, \tag{6}$$

where $\epsilon$ is a small number to prevent zero division. Mean centering guarantees $\sum_p \mathcal{K}^{\text{diff}}_{l'}(p) = 0$, while $\ell_1$ normalization bounds its magnitude. Ignoring the negligible $\epsilon$, the factor of 2 normalizes the positive sum and negative sum to 1 and $-1$, respectively, balancing the scale of the response. We further add an identity base kernel $\mathcal{K}^{\text{base}}$, whose center weight is 1 and all other weights are 0, to preserve the magnitude consistency of feature responses. Note that, reassembly with this base kernel alone is equivalent to nearest neighbor (NN) upsampling. The final reassembly kernel takes the form

$$\mathcal{K} = \mathcal{K}^{\text{base}} + \mathcal{K}^{\text{diff}}\,. \tag{7}$$

Combined with the identity base kernel, the differential kernel enables the upsampler to adaptively sharpen or smooth features. We provide a detailed explanation of these behaviors in Section 4.

**Encoder-Guided Kernel Generation.** The differential formulation determines how features are reassembled, but its effectiveness also depends on where the signed weights are assigned. Generating kernels solely from the low-res decoder feature may provide sufficient semantic context but lacks the fine-grained spatial cues to inform boundaries. Following FADE (Lu et al., 2024), we jointly exploit the low-res decoder feature $\mathcal{X}^d$ and the high-res encoder feature $\mathcal{X}^e$ for kernel generation. For efficiency, we follow the NN trick in FADE (Lu et al., 2024) to implement the cross-resolution feature interaction after channel compression. Formally, the unnormalized kernel $\mathcal{W} \in \mathbb{R}^{\sigma H \times \sigma W \times k_r^2}$ takes the form $\mathcal{W} = \mathcal{F}_e(\phi_e(\mathcal{X}^e)) + \mathrm{NN}\big(\mathcal{F}_d\big(\phi_d(\mathcal{X}^d)\big)\big)$ where $\phi_e(\cdot)$ and $\phi_d(\cdot)$ execute channel compression via $1 \times 1$ convolution to reduce parameter count, and $\mathcal{F}_e(\cdot)$ and $\mathcal{F}_d(\cdot)$ indicate $k_g \times k_g$ convolutions for generating $k_r^2$ upsampling-kernel channels.

**Gated Variant.** We further extend DiffUp to DiffUp+, a variant that introduces spatial modulation while retaining the same differential kernel formulation. Specifically, a lightweight gate predicts a spatial modulation coefficient $\mathcal{G}_{l'} \in (0, 1)$ from the aligned encoder and decoder features to control the strength of the differential correction. Its reassembly kernel is given by

$$\mathcal{K}^+ = \mathcal{K}^{\mathrm{base}} + \mathcal{G} \odot \mathcal{K}^{\mathrm{diff}}\,. \tag{8}$$

where $\odot$ denotes element-wise multiplication with broadcasting. Note that, this variant increases the channel-compression width from 32 to 64 and replaces the depthwise separable convolutions in kernel generation with standard $3 \times 3$ convolutions. Implementation details of the modulation branch are provided in Appendix A.

## 4 Understanding Differential Upsampling

Allowing signed weights makes boundary enhancement possible, but does not ensure that a learned upsampler will exploit this capacity. We therefore examine whether such enhancement occurs in learned features, whether it is accompanied by measurable prediction gains, and how the proposed differential formulation supports both boundary enhancement and regional smoothing.

### 4.1 Boundary Sharpness

**Conditional Boundary Sharpness Guarantee.** Cross-boundary differential penalizes contributions from the opposite semantic region, increasing the difference between the two sides. Under the binary-partition condition in Assumption 1(i), we assume that for a target location on either side of the boundary, the sum of differential weights assigned to the opposite semantic region is negative.

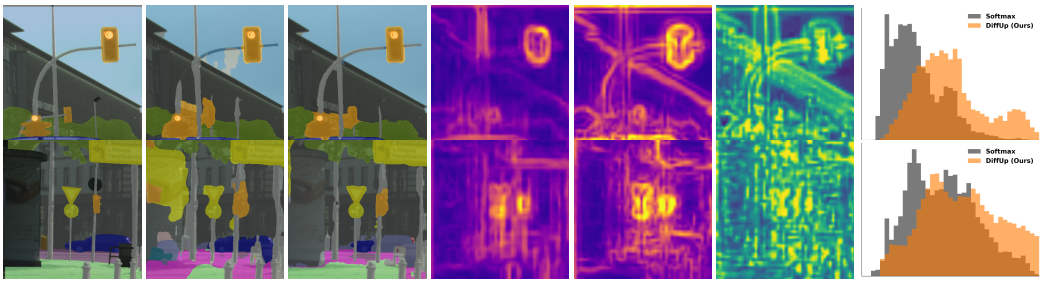
**Theorem 2 (Conditional Boundary Sharpness).** Under these conditions, a boundary pair $(A, B)$ with $I(A) = a < b = I(B)$ and its upsampled counterparts $(A', B')$ satisfy

$$|b - a| < |I(B') - I(A')| < (1 + 2\lambda)|b - a|. \tag{9}$$

where $\lambda = \max(\gamma_{A'}, \gamma_{B'})$, with $\gamma_{l'} = 1$ for DiffUp and $\gamma_{l'} = \mathcal{G}_{l'} \in (0, 1)$ for DiffUp+. The formal condition and complete proof are given in Appendix D.2.

**Controlled Experimental Setup.** To examine whether the conditional enhancement mechanism appears in learned features and is accompanied by prediction gains, we compare DiffUp with the softmax normalization (with the same kernel generation) on a standard SegNet (Badrinarayanan et al., 2017) baseline trained on Cityscapes (Cordts et al., 2016) for 80k iterations. We choose SegNet due to its clean architecture such that the improvement can be attributed to our design choices. We report mIoU and bIoU (Cheng et al., 2021) at boundary widths of 1% and 2% of the image diagonal.

**Evidence of Boundary Enhancement.** Fig. 3(d,e) shows stronger contour gradients under DiffUp than under the Softmax normalization, with gradient distributions on ground-truth boundaries shifted toward larger magnitudes (Fig. 3(g)). As shown in Fig. 3(f), the off-center negative responses broadly associate with many of these contours. While this kernel distribution reflects complex local adaptations rather than strict boundary confinement, the resulting feature gradients confirm that the learned signed weights effectively strengthen local boundary differences.




(a) Ground Truth (b) Softmax (c) DiffUp (d) Softmax Grad (e) DiffUp Grad (f) Neg. Weights (g) Grad Dist.

Figure 3: **Comparison of softmax-normalized and differential kernels.** DiffUp better preserves structural continuity like thin poles. (d)-(e) Gradient maps showing channel-averaged spatial-gradient magnitudes at the second-deepest upsampling stage. (f) Negative response representing the absolute sum of negative off-center kernel weights. Brighter colors indicate stronger local feature gradients and negative response. (g) Local gradient distributions computed over ground-truth boundaries. DiffUp produces larger gradient magnitudes. This quantitative evidence shown in the gradient maps verifies its improved boundary contrast.

**From Feature Responses to Performance Gains.** Table 1 reports the quantitative results of the controlled comparison. DiffUp improves mIoU by $2.11\%$, and remarkably, boosts bIoU by $3.08\%$ and $2.75\%$ at the $1\%$ and $2\%$ boundary-width condition, respectively, over the softmax counterpart. Fig. 3 also shows more complete thin structures and cleaner contours. These quantitative and qualitative improvements support the practical value of the observed boundary enhancement.

Table 1: Controlled comparison of upsampling formulations using SegNet trained on Cityscapes for 80k iterations. bIoU is evaluated at boundary widths of 1% and 2% of the image diagonal.

| Upsampling formulation | mIoU | bIoU (1%) | bIoU (2%) |
|---|---|---|---|
| Baseline (SegNet) | 58.97 | 41.71 | 50.45 |
| Softmax normalization | 60.62 | 43.24 | 52.11 |
| DiffUp (Ours) | **62.73** | **46.32** | **54.86** |

### 4.2 Regional Smoothing

Having examined the boundary enhancement capability of DiffUp, a natural concern is whether boundary enhancement comes at the cost of amplified fluctuations within regions. To justify this, we cluster the learned differential kernels and analyze their spatial patterns. As shown in Fig. 4, while some kernels still exhibit differential patterns on internal local textures, most kernels within homogeneous areas have a center-negative pattern (after adding the identity base kernel, the center response would cancel the negative effect and turn to an averaging kernel). The intermediate example shows two typical kernel patterns on smoothing and boundary regions. Further clustered kernel patterns are visualized in Fig 6 (Appendix C). Next we formally analyze this smoothing behavior.

**Assumption 2 (Center-Negative Kernel Pattern).** Within a homogeneous region, the differential kernel has a non-positive center weight and non-negative weights at all other positions.

**Theorem 3 (Conditional Regional Smoothness).** Under Assumption 2, the final reassembly kernel is non-negative and has unit sum. Consequently,

$$\min_{p\in\mathcal{N}_l^d(k_r)} I(p) \le I(l') \le \max_{p\in\mathcal{N}_l^d(k_r)} I(p)\,, \tag{10}$$

so the reassembly introduces no new local extrema. The proof is provided in Appendix D.3.

In short, the controlled comparison and response visualizations support boundary enhancement as an empirically useful behavior. While Assumption 2 defines an idealized condition and real-world predictions may deviate due to local noise or textures, the learned kernel patterns demonstrate how a unified differential formulation provides the capacity to gracefully shift between contrast enhancement and regional smoothing.

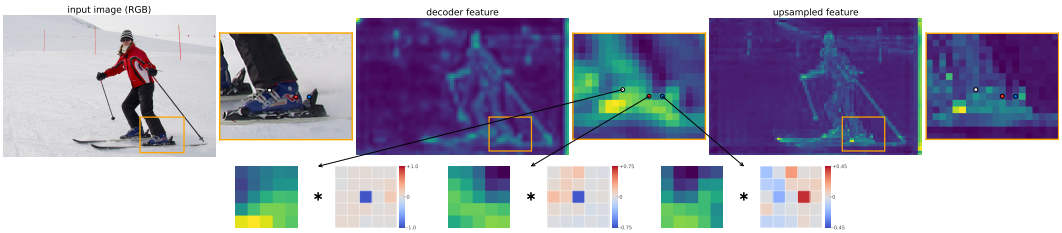



Figure 4: **Visualization of spatially adaptive reassembly kernels around a skier's ski-boot boundary.** The three sampled locations illustrate distinct behaviors: the interior point (white) produces a center-negative smoothing kernel, the transition point (red) exhibits an intermediate pattern, and the boundary point (blue) produces a contrastive differential kernel that sharpens the edge.

Table 2: Semantic segmentation results on COCONut with SegFormer-B1 and SegFormer-B5. The bilinear baseline reports total model parameters and FLOPs, while "+" in the cost columns denotes the additional costs introduced by each upsampler. Each Δ column reports the absolute improvement over Bilinear for the preceding metric, in percentage points. Dashes indicate entries that are not reported or not applicable. Best results are in **boldface**, and second best are underlined.

| | SegFormer-B1 | | | | | | SegFormer-B5 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Upsampler | Params (M) | GFLOPs | mIoU ↑ | Δ | bIoU ↑ | Δ | Params (M) | mIoU ↑ | Δ | bIoU ↑ | Δ |
| Bilinear | 13.71 | 15.96 | 58.42 | – | 39.35 | – | 82.00 | 65.84 | – | 48.85 | – |
| FeatUp (Fu et al., 2024) | +0.02 | +0.37 | 59.24 | +0.82 | 41.17 | +1.82 | – | – | – | – | – |
| LoftUp (Huang et al., 2025) | +13.18 | +185.25 | 59.71 | +1.29 | 42.02 | +2.67 | – | – | – | – | – |
| NAF (Chambon et al., 2025) | +1.99 | +25.69 | 59.70 | +1.28 | 42.34 | +2.99 | – | – | – | – | – |
| DySample | +0.05 | +0.11 | 59.55 | +1.13 | 41.04 | +1.69 | +0.05 | 66.02 | +0.18 | 49.18 | +0.33 |
| FADE | +0.28 | +2.18 | 60.09 | +1.67 | 43.07 | +3.72 | +0.28 | 66.50 | +0.66 | 50.89 | +2.04 |
| SAPA | +0.11 | +0.62 | 59.58 | +1.16 | 42.45 | +3.10 | +0.11 | 66.26 | +0.42 | 50.23 | +1.38 |
| DiffUp (Ours) | +0.11 | +0.62 | 60.32 | +1.90 | 43.72 | +4.37 | +0.11 | **66.91** | +1.07 | **51.36** | +2.51 |
| DiffUp+ (Ours) | +0.42 | +2.19 | **60.52** | +2.10 | **43.87** | +4.52 | +0.42 | 66.65 | +0.81 | 51.07 | +2.22 |

## 5 Results and Discussion

We evaluate DiffUp with various baseline models across four dense prediction tasks: semantic segmentation (Noh et al., 2015; Xie et al., 2021), image matting (Xu et al., 2017; Rhemann et al., 2009), depth estimation (Li et al., 2023), and object detection (Jocher et al., 2023; Zhu et al., 2022). Implementation details and training configurations are provided in Appendix A and B.

### 5.1 Semantic Segmentation

**Experimental Setup.** We evaluate the upsamplers on the 133-class COCONut (Deng et al., 2024) dataset, whose fine-grained annotations enable detailed evaluation of object boundaries. We adopt ImageNet-1K pre-trained SegFormer-B1 and SegFormer-B5 (Xie et al., 2021) backbones to test performance across different model capacities. All compared upsamplers replace only the upsampling modules and the models are trained for 160k iterations under a unified MMSegmentation protocol. We report mIoU and bIoU (Cheng et al., 2021). Complete training and inference settings are provided in Appendix B.

**Results.** As shown in Table 2, DiffUp achieves the best mIoU and bIoU on both backbones. Compared with bilinear upsampling, it improves mIoU and bIoU, respectively, by $1.90\%$ and $4.37\%$ on SegFormer-B1 and by $1.07\%$ and $2.51\%$ on SegFormer-B5. Notably, the gain on bIoU exceeds that on mIoU, showing that DiffUp excels at sharpening boundaries.

We further conduct cross-paradigm comparisons against model-agnostic upsamplers. Constrained by GPU memory footprint, we perform this cross-paradigm evaluation only with SegFormer-B1. All methods follow the same SegFormer-B1 training recipe; released pre-trained weights are used and jointly fine-tuned when applicable. Under this common downstream setting, DiffUp exceeds the best model-agnostic mIoU (LoftUp) by $0.61\%$ and the best model-agnostic bIoU (NAF) by $1.38\%$,

Table 3: Portrait matting results on P3M-500-P and P3M-500-NP using the official Hybrid inference strategy. MSE and MAD are reported in units of $10^{-3}$. Best results are in **boldface**, and second best are underlined.

| | P3M-500-P | | | | | P3M-500-NP | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Upsampler | SAD ↓ | MSE ↓ | MAD ↓ | GRAD ↓ | CONN ↓ | SAD ↓ | MSE ↓ | MAD ↓ | GRAD ↓ | CONN ↓ |
| Bilinear | 9.152 | 2.866 | 5.327 | 13.749 | 8.931 | 12.600 | 4.394 | 7.359 | 12.763 | 12.083 |
| SAPA | 9.449 | 3.130 | 5.493 | 14.174 | 9.185 | 12.102 | 4.171 | 7.113 | 13.285 | 11.523 |
| FADE | 8.853 | 2.682 | 5.158 | 13.378 | 8.613 | 10.558 | **3.373** | 6.127 | 11.609 | 10.067 |
| DySample | 8.768 | 2.814 | 5.109 | 13.879 | 8.497 | 11.257 | 3.551 | 6.507 | 12.759 | 10.677 |
| DiffUp (Ours) | **7.910** | **2.381** | **4.609** | **12.637** | **7.610** | **10.544** | 3.437 | **6.033** | **11.417** | **9.978** |
| DiffUp+ (Ours) | 8.261 | 2.524 | 4.814 | 12.813 | 7.991 | 11.473 | 3.905 | 6.708 | 12.414 | 10.880 |

Table 4: Depth estimation results on NYUv2. Best results are **in boldface**, and second best are underlined.

| Upsampler | Abs Rel ↓ | RMSE ↓ | $\delta_1$ ↑ | $\delta_2$ ↑ | $\delta_3$ ↑ | $\log_{10}$ ↓ | RMSE log ↓ | SILog ↓ | Sq Rel ↓ |
|---|---|---|---|---|---|---|---|---|---|
| Bilinear | 0.1193 | 0.4031 | 0.8717 | 0.9787 | **0.9950** | 0.0499 | 0.1484 | 12.0196 | 0.0694 |
| SAPA | 0.1189 | 0.4098 | 0.8669 | 0.9783 | 0.9945 | 0.0505 | 0.1504 | 12.2446 | 0.0721 |
| DySample | 0.1171 | 0.4005 | 0.8731 | 0.9786 | **0.9950** | 0.0494 | 0.1471 | 11.9256 | **0.0690** |
| FADE | 0.1169 | 0.4027 | 0.8725 | 0.9790 | 0.9944 | 0.0493 | 0.1470 | 11.9489 | 0.0695 |
| DiffUp (Ours) | 0.1179 | **0.3976** | **0.8756** | 0.9782 | 0.9947 | **0.0491** | **0.1468** | 11.9593 | 0.0696 |
| DiffUp+ (Ours) | **0.1161** | 0.4018 | 0.8741 | **0.9791** | 0.9944 | 0.0493 | **0.1468** | **11.9170** | 0.0694 |

while adding only 0.11M parameters and 0.62 GFLOPs. These results demonstrate the effectiveness of DiffUp as a lightweight upsampling module in an end-to-end network.

### 5.2 Image Matting

**Experimental Setup.** For all evaluated upsamplers, we replace the corresponding operations in both the semantic and matting decoders of P3M-Net (Li et al., 2021). Models are trained on face-blurred P3M-10k and tested on P3M-500-P (in-domain) and P3M-500-NP (cross-domain) using the official Hybrid inference strategy (full settings are provided in Appendix B).

**Results.** As shown in Table 3, DiffUp yields the best performance across all metrics on P3M-500-P, reducing SAD from $9.152$ to $7.910$ relative to bilinear upsampling. On the cross-domain P3M-500-NP set, it maintains robust generalization by ranking first in four metrics. These results extend the benefit of differential upsampling from discrete segmentation boundaries to continuous alpha estimation. DiffUp consistently outperforms DiffUp+ on both test sets, with the advantage remaining evident under the cross-domain setting. This may be partly attributed to the reduced capacity of DiffUp, which provides regularization and reduces sensitivity to domain-specific features.

### 5.3 Depth Estimation

**Experimental Setup.** We evaluate monocular depth estimation on NYUv2 (Silberman et al., 2012) using an ImageNet-1K pre-trained Swin-T (Liu et al., 2021) backbone with DepthFormer (Li et al., 2023) and a DenseDepthHead. All upsamplers replace the same operations and are trained under an identical protocol. We report metrics including Abs Rel, RMSE, and $\delta_1$. Full training configurations are provided in Appendix B.

**Results.** As shown in Table 4, DiffUp achieves the lowest RMSE, highest $\delta_1$, and smallest $\log_{10}$. RMSE penalizes blurring at depth discontinuities, and $\delta_1$ quantifies prediction fidelity for pixels around object contours. Together with the $\log_{10}$ result, this evidence further verifies the capability of DiffUp to sharpen boundaries, aligning with our findings on segmentation and matting.

### 5.4 Object Detection

**Experimental Setup.** We evaluate the upsamplers on the VisDrone 2019-DET (Zhu et al., 2022) validation set using YOLOv8m-p2 (Jocher et al., 2023). All methods use identical training hyperparameters, with 100 epochs and an input image size of 640. We report mAP at an IoU threshold of

Table 5: Object detection results on VisDrone 2019-DET with YOLOv8m-p2. The baseline reports total costs, while "+" denotes additional parameters and FLOPs. Best results are in **boldface**, and second best are underlined.

| Upsampler | $mAP_{50}$ ↑ | $mAP_{50:95}$ ↑ | Params (M) | GFLOPs |
|---|---|---|---|---|
| Baseline | 44.8 | 27.2 | 23.80 | 86.1 |
| FADE | 45.4 | <u>27.7</u> | +0.16 | +2.1 |
| SAPA | <u>45.6</u> | 27.5 | +0.06 | +0.5 |
| DySample | 45.5 | <u>27.7</u> | +0.04 | +0.2 |
| DiffUp (Ours) | 45.5 | 27.6 | +0.07 | +0.5 |
| DiffUp+ (Ours) | **45.9** | **27.8** | +0.23 | +2.6 |

Table 6: Ablation study of implementation details of DiffUp on COCONut. All variants use a $5 \times 5$ reassembly kernel. "DW" denotes depthwise separable convolution.

| Width | Predictor | Gate | Params (M) | mIoU | bIoU |
|---|---|---|---|---|---|
| 32 | DW $3 \times 3$ | | 0.11 | 60.32 | 43.72 |
| 64 | DW $3 \times 3$ | | 0.22 | 60.56 | 43.93 |
| 64 | Standard Conv $3 \times 3$ | | 0.37 | 60.51 | 43.82 |
| 64 | Standard Conv $3 \times 3$ | ✓ | 0.42 | 60.52 | 43.87 |

0.50 and averaged over IoU thresholds from 0.50 to 0.95, together with parameter and computational overhead.

**Results.** As shown in Table 5, DiffUp improves the baseline with only $+0.07$M parameters and $+0.5$ GFLOPs, achieving accuracy comparable to FADE and DySample. Its $mAP_{50}$ and $mAP_{50:95}$ remain broadly on par with the strongest existing results. Notably, DySample is particularly well suited to small-object detection, making the comparable performance of DiffUp encouraging. DiffUp+ further improves both metrics, reaching $45.9$ $mAP_{50}$ and $27.8$ $mAP_{50:95}$. These results suggest that the enhanced refinement in DiffUp+ provides additional benefits for preserving spatial details. Overall, DiffUp offers a lightweight and competitive solution, while DiffUp+ provides a favorable option when higher detection accuracy is desired.

### 5.5 Ablation Study

The effectiveness of the differential kernel formulation has been validated in Section 4. Here, we further analyze the architectural choices of DiffUp. Following previous dynamic upsamplers such as CARAFE and FADE, all variants adopt a $5 \times 5$ reassembly kernel. We use the same experimental setup as Table 2. Table 6 shows that increasing predictor width and replacing depthwise separable convolutions bring only marginal improvements with additional parameters. Therefore, we adopt the 32-channel depthwise separable predictor as the default DiffUp configuration, which provides a favorable accuracy–efficiency trade-off. The gated variant DiffUp+ introduces spatial modulation with a higher-capacity predictor, providing stronger adaptive capability for tasks requiring more complex feature refinement.

## 6 Conclusion

In this work, we reveal the intrinsic limitation of softmax normalization used in dynamic upsamplers in suppressing boundaries. To address this, we propose DiffUp, a novel, plug-and-play, effective, and task-agnostic upsampler that actively enhances details through a differential kernel formulation. Our findings motivate a reconsideration of the role of softmax in learned feature aggregation: while its non-negative, sum-to-one weights provide stable averaging, this constraint can limit local contrast enhancement. DiffUp demonstrates that bounded signed weights offer an alternative for feature upsampling. More broadly, this perspective suggests investigating signed aggregation in other visual operators where boundary fidelity is important, rather than treating positive-only normalization as a universal design choice.

### AI USE STATEMENT

In this work, we have used generative AI tools to aid or polish writing.

### ETHICS STATEMENT

This work does not raise any questions regarding the Code of Ethics.

### REPRODUCIBILITY STATEMENT

To ensure reproducibility, detailed experimental setups and hyperparameters are provided in the respective sections.

## REFERENCES

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017. doi: 10.1109/TPAMI.2016.2644615.

Loick Chambon, Paul Couairon, Eloi Zablocki, Alexandre Boulch, Nicolas Thome, and Matthieu Cord. Naf: Zero-shot feature upsampling via neighborhood attention filtering, 2025. URL https://arxiv.org/abs/2511.18452.

Linwei Chen, Ying Fu, Lin Gu, Chenggang Yan, Tatsuya Harada, and Gao Huang. Frequency-aware feature fusion for dense image prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10763–10780, 2024. doi: 10.1109/TPAMI.2024.3449959.

Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

Bowen Cheng, Ross Girshick, Piotr Dollár, Alexander C Berg, and Alexander Kirillov. Boundary iou: Improving object-centric image segmentation evaluation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15329–15337. IEEE, 2021.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Paul Couairon, Loick Chambon, Louis Serrano, Jean-Emmanuel Haugeard, Matthieu Cord, and Nicolas Thome. Jafar: Jack up any feature at any resolution, 2025. URL https://arxiv.org/abs/2506.11136.

Yutong Dai, Hao Lu, and Chunhua Shen. Learning affinity-aware upsampling for deep image matting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6841–6850, 2021.

Xueqing Deng, Qihang Yu, Peng Wang, Xiaohui Shen, and Liang-Chieh Chen. Coconut: Modernizing coco segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Zewen Du, Zhenjiang Hu, Guiyu Zhao, Ying Jin, and Hongbin Ma. Lda-aqu: Adaptive query-guided upsampling via local deformable attention. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 4919–4927, 2024.

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=GkJiNn2QDF.

Haiwen Huang, Anpei Chen, Volodymyr Havrylov, Andreas Geiger, and Dan Zhang. Loftup: Learning a coordinate-based feature upsampler for vision foundation models, 2025. URL https://arxiv.org/abs/2504.14032.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolo. *Zenodo*, 2023.

Jizhizi Li, Sihan Ma, Jing Zhang, and Dacheng Tao. Privacy-preserving portrait matting. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 3501–3509, 2021. doi: 10.1145/3474085.3475512.

Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. *Machine Intelligence Research*, 20(6):835–851, 2023.

Wenze Liu, Hao Lu, Hongtao Fu, and Zhiguo Cao. Learning to upsample by learning to sample. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6027–6037, 2023.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Hao Lu, Yutong Dai, Chunhua Shen, and Songcen Xu. Indices matter: Learning to index for deep image matting. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Hao Lu, Yutong Dai, Chunhua Shen, and Songcen Xu. Index networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Hao Lu, Wenze Liu, Hongtao Fu, and Zhiguo Cao. Fade: Fusing the assets of decoder and encoder for task-agnostic upsampling. In *Proc. European Conference on Computer Vision (ECCV)*, 2022a.

Hao Lu, Wenze Liu, Zixuan Ye, Hongtao Fu, Yuliang Liu, and Zhiguo Cao. Sapa: Similarity-aware point affiliation for feature upsampling. In *Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022b.

Hao Lu, Wenze Liu, Hongtao Fu, and Zhiguo Cao. Fade: A task-agnostic upsampling operator for encoder–decoder architectures. *International Journal of Computer Vision*, 2024.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1520–1528, 2015. doi: 10.1109/ICCV.2015.178.

Mike Ranzinger, Greg Heinrich, Pavlo Molchanov, Bryan Catanzaro, and Andrew Tao. Featsharp: Your vision model features, sharper. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=lioemOcq3H.

Christoph Rhemann, Carsten Rother, Jue Wang, Margrit Gelautz, Pushmeet Kohli, and Pamela Rott. A perceptually motivated online benchmark for image matting. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1826–1833. IEEE, 2009. doi: 10.1109/CVPR.2009.5206503.

Minseok Seo, Mark Hamilton, and Changick Kim. Upsample anything: A simple and hard to beat baseline for feature upsampling. *arXiv preprint arXiv:2511.16301*, 2025.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, 2016.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pp. 746–760. Springer, 2012.

Matthew Walmer, Saksham Suri, Anirud Aggarwal, and Abhinav Shrivastava. Uplift: Efficient pixel-dense feature upsampling with local attenders. *arXiv preprint arXiv:2601.17950*, 2026.

Jiaqi Wang, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. Carafe: Content-aware reassembly of features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3007–3016, 2019.

Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. Anyup: Universal feature upsampling. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Ning Xu, Brian Price, Scott Cohen, and Thomas Huang. Deep image matting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2970–2979, 2017.

Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7380–7399, 2022.

# APPENDIX

## A ADDITIONAL IMPLEMENTATION DETAILS

### A.1 MODULE CONFIGURATION

Both DiffUp variants use a $5 \times 5$ reassembly kernel and one kernel group by default. DiffUp compresses the encoder and decoder features to 32 channels and predicts the unnormalized kernel with depthwise separable $3 \times 3$ convolutions. To enrich feature representation, DiffUp+ instead compresses the encoder and decoder features to 64 channels, uses standard convolutions for kernel prediction and adds a modulation branch. The branch concatenates the aligned compressed features and applies two $1 \times 1$ convolutions with an intermediate ReLU and a final sigmoid. These configurations are shown in Table 7.

Using the aligned compressed features, the modulation branch computes

$$\mathcal{G} = \mathrm{Sigmoid}\big(\mathcal{C}_2\big(\mathrm{ReLU}\big(\mathcal{C}_1\big[\mathrm{NN}(\phi_d(\mathcal{X}^d)); \phi_e(\mathcal{X}^e)\big]\big)\big)\big)\,, \tag{11}$$

where $\phi_d$ and $\phi_e$ are the channel-compression projections, $\mathcal{C}_1$ and $\mathcal{C}_2$ are $1 \times 1$ convolutions, and $[\cdot;\cdot]$ denotes channel-wise concatenation.

After combining the encoder- and decoder-conditioned predictions, we apply layer normalization over the kernel channels at each spatial location. The kernel is then mean-centered and L1-normalized with $\epsilon = 10^{-8}$ before adding the identity base kernel. Nearest-neighbor interpolation is used for cross-resolution alignment. When rounding produces a one-pixel spatial mismatch, replication padding is applied to the encoder-conditioned kernel prediction. All convolutional weights are initialized with Xavier normal initialization, and their biases are initialized to zero.

Table 7: Default configurations of DiffUp and DiffUp+.

| Configuration | DiffUp | DiffUp+ |
|---|---|---|
| Compression width | 32 | 64 |
| Kernel predictor | Depthwise separable $3 \times 3$ | Standard $3 \times 3$ |
| Gate branch | No | Yes |
| Reassembly kernel | $5 \times 5$ | $5 \times 5$ |
| Kernel groups | 1 | 1 |

## B COMPLETE EXPERIMENTAL CONFIGURATIONS

Unless stated otherwise, each comparison changes only the upsampling operator; the model architecture, initialization, optimization, data processing, and evaluation protocol remain identical across methods.

### B.1 SEMANTIC SEGMENTATION ON COCONUT

We conduct semantic segmentation experiments using MMSegmentation, following the standard SegFormer training and inference recipe. All compared upsamplers are evaluated under identical experimental settings (see Table 8).

### B.2 IMAGE MATTING

Following the privacy-preserving training protocol, we train P3M-Net on the 9,421 face-blurred portraits in the P3M-10k training set. We evaluate on P3M-500-P and P3M-500-NP, which contain 500 face-blurred and 500 unblurred portraits, respectively, using the official Hybrid inference strategy. For all compared methods, the corresponding upsampling operations in both the semantic and matting decoders are replaced. Relative to the original P3M recipe, we change only the optimization configuration (see Table 9).

Table 8: Training and evaluation configuration for semantic segmentation.

| Item | Configuration |
|---|---|
| Dataset | COCONut (133 semantic categories) |
| Architectures | SegFormer-B1 and SegFormer-B5 |
| Initialization | Official ImageNet-1K pre-trained backbones |
| Input/crop size | $512 \times 512$ |
| Batch size | 16 in total |
| Optimizer | AdamW, $(\beta_1, \beta_2) = (0.9, 0.999)$ |
| Learning rate and schedule | Base learning rate $6 \times 10^{-5}$ for the backbone and a $10\times$ multiplier for the decode head; linear warm-up for 1,500 iterations from $10^{-6}$, followed by PolyLR with power 1.0 and minimum learning rate 0 |
| Weight decay | 0.01; no weight decay for positional embeddings and normalization layers |
| Training length | 160k iterations |
| Data augmentation | Random scaling in $[0.5, 2.0]$, random cropping, horizontal flipping, and photometric distortion |
| Inference protocol | Sliding-window inference with a $512 \times 512$ crop and a $341 \times 341$ stride |
| Boundary width for bIoU | 2% of the image diagonal ($d = \mathrm{round}(0.02 \times \sqrt{H^2 + W^2})$) |

Table 9: Training and evaluation configuration for portrait matting.

| Item | Configuration |
|---|---|
| Dataset | P3M-10k |
| Training data | 9,421 face-blurred portraits |
| Evaluation sets | P3M-500-P (500 face-blurred portraits) and P3M-500-NP (500 unblurred portraits) |
| Inference strategy | Official Hybrid inference |
| Architecture | P3M-Net |
| Upsampler placement | Corresponding operations in both the semantic and matting decoders |
| Optimizer | AdamW |
| Initial learning rate | $2 \times 10^{-5}$ |
| Weight decay | $1 \times 10^{-4}$ |
| Warm-up | Linear warm-up for 3 epochs |
| Learning-rate schedule | Cosine decay to $1 \times 10^{-7}$ |
| Training length | 120 epochs |

### B.3 Depth Estimation on NYUv2

All methods use the same training configuration, as shown in Table 10.

Table 10: Training configuration for monocular depth estimation.

| Item | Configuration |
|---|---|
| Dataset | NYUv2 |
| Backbone | Swin-T; window size 7; ImageNet-1K pretrained |
| Neck/head | DepthFormer / DenseDepthHead; FPN convolution dimension 256 |
| Training crop | $480 \times 640$ |
| Optimizer | AdamW |
| Initial learning rate | $10^{-4}$ |
| Weight decay | 0.01 |
| Learning-rate schedule | Iteration-based cosine annealing to 0 |
| Training length | 38,400 iterations |
| Batch size | 16 |
| Evaluation depth range and mask | 0.001–10 m, Eigen crop |

### B.4 Object Detection on VisDrone 2019-DET

All upsamplers are evaluated with YOLOv8m-p2 under identical training hyperparameters. Table 11 lists the available settings.

Table 11: Training and evaluation configurations for object detection on VisDrone.

| **Item** | **Configuration** |
|---|---|
| Dataset | VisDrone 2019-DET |
| Evaluation split | Validation set (548 images) |
| Architecture | YOLOv8m-p2 (4 scales: $P_2 \sim P_5$) |
| Pre-training | COCO pre-trained weights (`yolov8m.pt`) |
| Initialization | Compatible layers transferred; $P_2$ head & upsamplers random init |
| Input size | $640 \times 640$ |
| Training epochs | 100 epochs |
| Batch size | 16 |
| Optimizer | SGD (momentum = 0.937, weight decay = $5 \times 10^{-4}$) |
| Learning rate schedule | Linear decay ($lr_0 = 0.01 \rightarrow lr_f = 10^{-4}$) with 3-epoch warmup |
| Data augmentation | Mosaic ($p = 1.0$, closed at epoch 90), Horiz. flip ($p = 0.5$), HSV jitter ($0.015, 0.7, 0.4$), Translation, Scaling, Erasing (0.4) |
| Inference / NMS | Conf threshold = 0.001, IoU threshold = 0.7, Max det = 500 |

### B.5 Controlled Ablation on Cityscapes

We conduct ablation experiment (Section 4.1) on Cityscapes, the configurations are shown in Table 12.

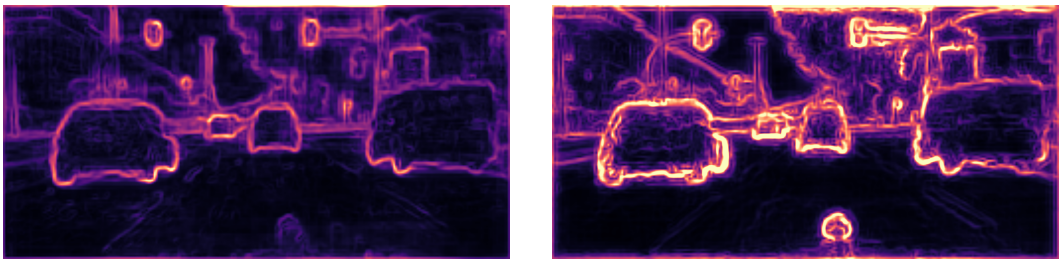
Table 12: Training and evaluation configuration for the controlled ablation.

| Item | Configuration |
|---|---|
| Dataset | Cityscapes |
| Architecture | SegNet (VGG-16 backbone with Batch Normalization) |
| Initialization | ImageNet-1K pre-trained weights for the VGG-16 encoder |
| Input/crop size | $512 \times 1024$ random crop during training; $1024 \times 2048$ full resolution during evaluation |
| Batch size | 8 |
| Optimizer | Adam ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$) |
| Learning rate and schedule | Initial lr $5 \times 10^{-4}$, 500 warmup iterations, followed by polynomial decay (power $= 0.9$, min_lr $= 10^{-6}$) |
| Weight decay | $1 \times 10^{-4}$ |
| Training length | 80k iterations |
| Data augmentation | Random scaling ($\times 0.5 \sim 2.0$), random crop ($512 \times 1024$), random horizontal flip ($p = 0.5$), and color jittering |
| Inference protocol | Single-scale, whole-image evaluation at $1024 \times 2048$ without test-time augmentation |

## C Additional Qualitative Results

### C.1 Additional Evidence

In Section 4.1, we demonstrated the capability of DiffUp in boundary enhancement using cropped patches. To provide a comprehensive view of the feature responses, we present the full-resolution feature gradient maps in Figure 5. As shown, compared to the conventional Softmax normalization, DiffUp consistently produces stronger and sharper gradient responses along semantic contours across the entire image. Furthermore, Figure 6 visualizes the clustered spatial patterns of the learned kernels, which corroborates our analysis of regional smoothing and boundary sharpness.



(a) Softmax Normalization (b) DiffUp (Ours)

Figure 5: Full-resolution comparison of feature gradient magnitudes. These uncropped maps expand upon the qualitative results in Figure 3. Compared to the standard Softmax normalization (a), our proposed DiffUp (b) yields significantly stronger and sharper gradient responses along semantic boundaries. This provides global visual evidence supporting our analysis in Section 4.1 that DiffUp effectively overcomes the local boundary suppression issue.

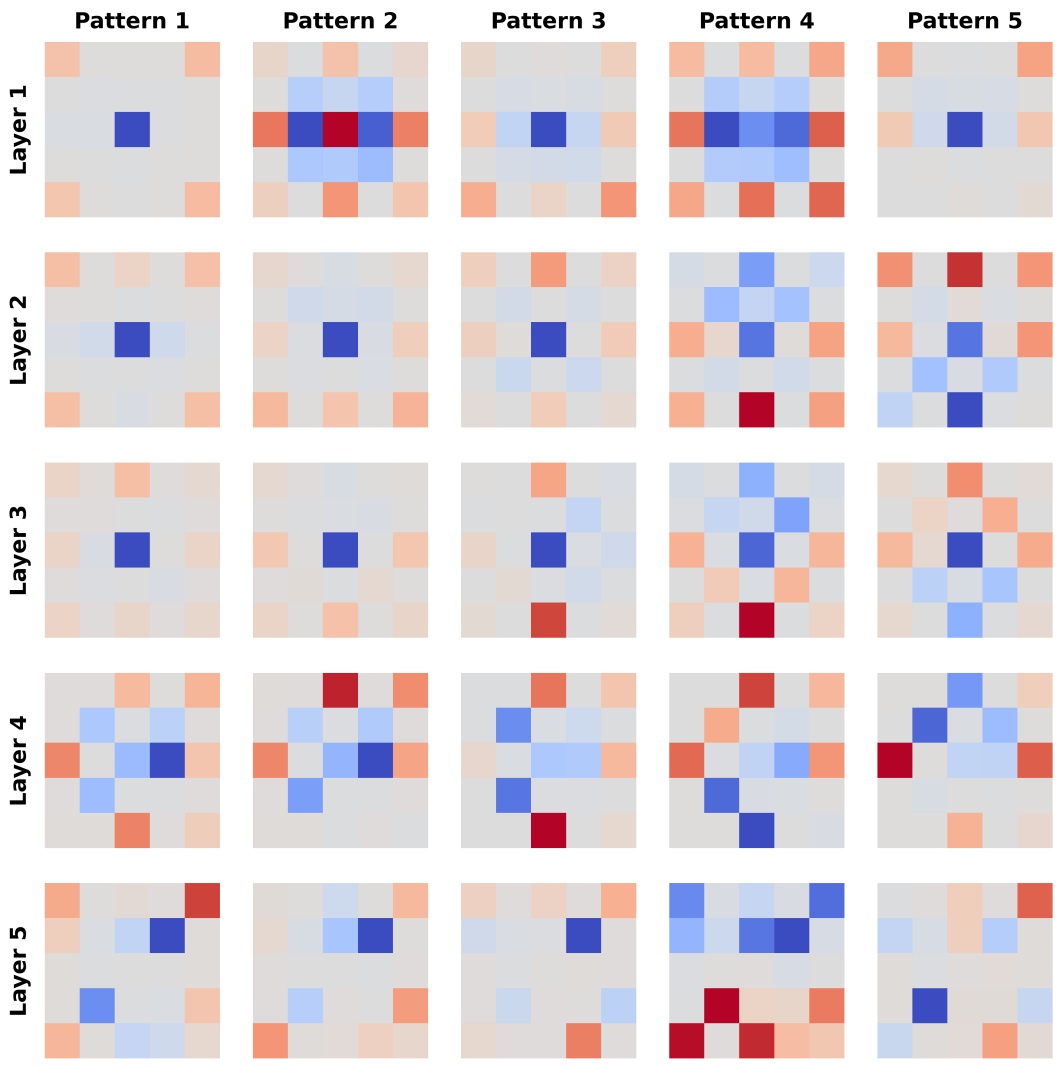



Figure 6: **K-means clustering of the learned differential kernels across different layers.** The kernels are extracted on a single Cityscapes image, corresponding to the experiment detailed in Section 4 and Appendix B. Rows denote different decoding layers, while columns represent the representative cluster centers (Pattern 1 to Pattern 5). These clustered spatial patterns provide additional visualization for the regional smoothing and sharpness behavior discussed in Section 4.1 and 4.2.

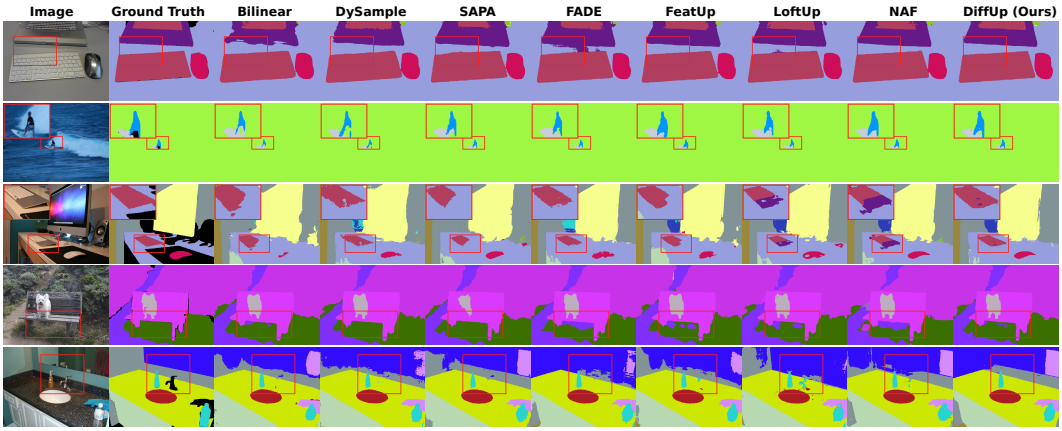



Figure 7: **Semantic segmentation on COCONut using SegFormer-B1 as the fixed backbone.** From left to right: input image, ground truth, and prediction results produced by different upsampler variants. Zoomed insets highlight local fine-grained segmentation details.

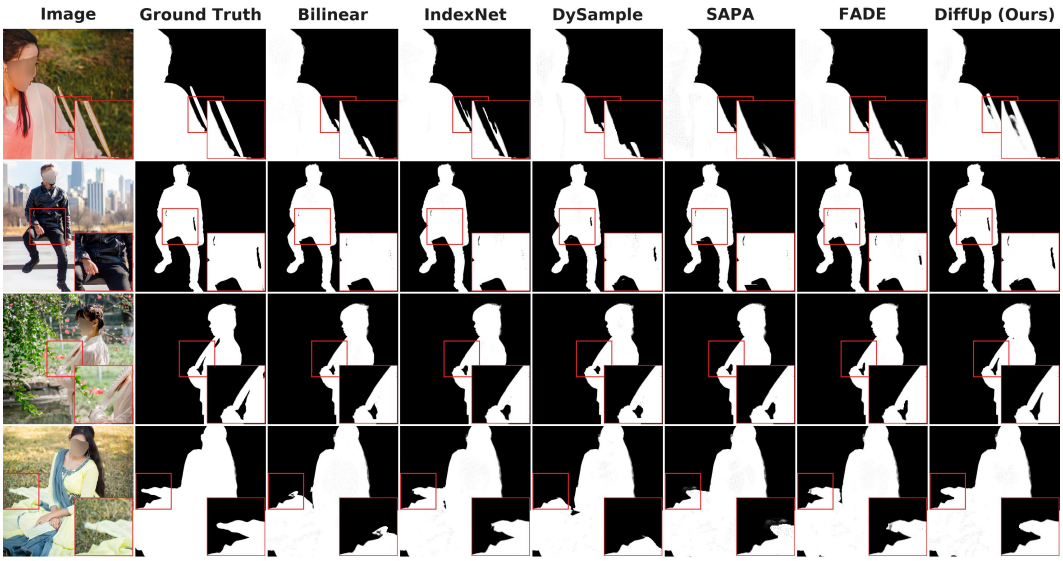



Figure 8: **Matting on the P3M dataset using P3MNet as the fixed backbone.** From left to right: input image, ground truth, and prediction results produced by different upsampler variants. Zoomed insets highlight sharper boundaries.

### C.2 Additional Visualizations

We provide visualization comparisons across semantic segmentation, image matting, depth estimation and object detection, with only the upsampler replaced and all other components kept unchanged. Representative results are shown in Fig. 7, Fig. 8, Fig. 9, and Fig. 10 respectively.

## D Complete Proofs

### D.1 Proof of Local Boundary Suppression

*Proof.* For adjacent boundary pixels $A$ (value $a$) and $B$ (value $b$) with $a < b$, we partition their neighborhoods as:

- $\Omega_A^a \subset \mathcal{N}_A^d(k_r)$: point set with value $a$ in $\mathcal{N}_A^d(k_r)$.

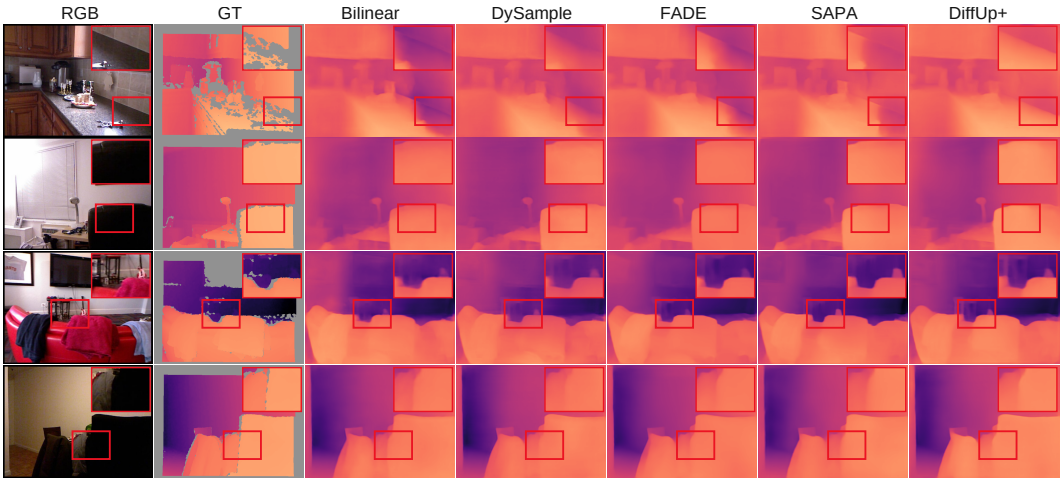



Figure 9: **Depth estimation on the NYUv2 dataset using DepthFormer as the fixed backbone.** From left to right: input image, ground truth, and prediction results produced by different upsampler variants.

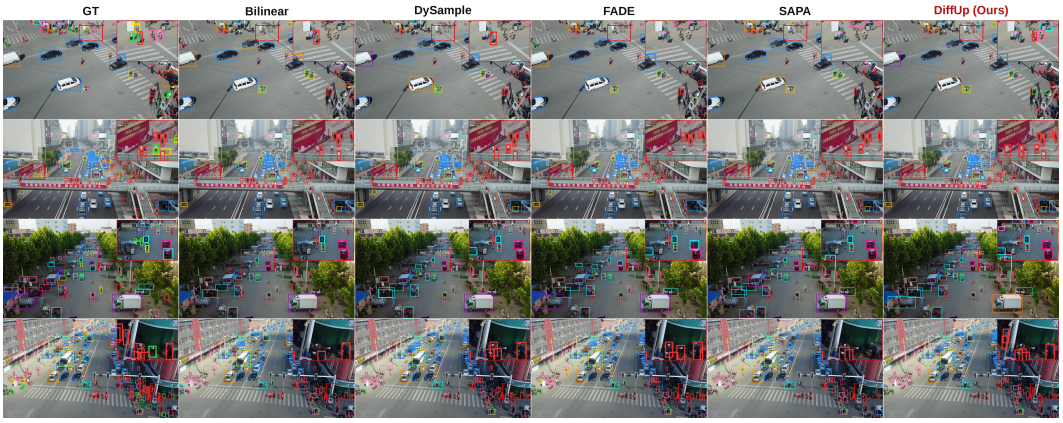



Figure 10: **Object detection on VisDrone 2019-DET with YOLOv8m-p2.** From left to right: ground truth, and prediction results produced by different upsampler variants.

- $\Omega_A^b \subset \mathcal{N}_A^d(k_r)$: point set with value $b$ in $\mathcal{N}_A^d(k_r)$.
- $\Omega_B^a \subset \mathcal{N}_B^d(k_r)$: point set with value $a$ in $\mathcal{N}_B^d(k_r)$.
- $\Omega_B^b \subset \mathcal{N}_B^d(k_r)$: point set with value $b$ in $\mathcal{N}_B^d(k_r)$.

where $|\Omega_A^a| + |\Omega_A^b| = |\Omega_B^a| + |\Omega_B^b| = k_r^2$.

Let $\{\mathcal{K}_{A'}(i)\}$ and $\{\mathcal{K}_{B'}(j)\}$ denote normalized weight vectors satisfying $\sum_i \mathcal{K}_{A'}(i) = \sum_j \mathcal{K}_{B'}(j) = 1$ and $\mathcal{K}_{A'}(i) > 0, \mathcal{K}_{B'}(j) > 0$. The reconstructed values are:

$$I(A') = \sum_{i \in \Omega_A^a} \mathcal{K}_{A'}(i) \times a + \sum_{i \in \Omega_A^b} \mathcal{K}_{A'}(i) \times b \tag{12}$$

$$I(B') = \sum_{j \in \Omega_B^a} \mathcal{K}_{B'}(j) \times a + \sum_{j \in \Omega_B^b} \mathcal{K}_{B'}(j) \times b \tag{13}$$

Define the value difference between reconstructed points:

$$\Delta_{A'B'} := I(B') - I(A') = a\left(\sum_{\Omega_B^a}\mathcal{K}_{B'}(j) - \sum_{\Omega_A^a}\mathcal{K}_{A'}(i)\right) + b\left(\sum_{\Omega_B^b}\mathcal{K}_{B'}(j) - \sum_{\Omega_A^b}\mathcal{K}_{A'}(i)\right) \tag{14}$$

Introducing the boundary weight difference parameter:

$$\gamma := \sum_{\Omega_B^b}\mathcal{K}_{B'}(j) - \sum_{\Omega_A^b}\mathcal{K}_{A'}(i) \tag{15}$$

which satisfies $\gamma \in (-1, 1)$ by normalization constraints. Substituting $\sum_{\Omega_B^a}\mathcal{K}_{B'}(j) = 1 - \sum_{\Omega_B^b}\mathcal{K}_{B'}(j)$ and $\sum_{\Omega_A^a}\mathcal{K}_{A'}(i) = 1 - \sum_{\Omega_A^b}\mathcal{K}_{A'}(i)$ into Eq. (14) yields:

$$\Delta_{A'B'} = (b-a)\gamma \tag{16}$$

Taking the absolute value with $|\gamma| < 1$, we directly have $|\Delta_{A'B'}| = (b-a)|\gamma| < b - a$.

Thus $|I(B') - I(A')| < |I(B) - I(A)|$ holds universally. □

Note that, this pairwise difference reduction propagates through all sampling pairs, establishing the global difference reduction.

### D.2 Proof of Conditional Boundary Sharpness

We use the binary-partition condition in Assumption 1(i) of the main paper: the reassembly neighborhoods of a boundary pair $(A, B)$ contain only feature values $a = I(A) < b = I(B)$. The softmax condition in Assumption 1(ii) is not used.

**Assumption 3 (Cross-Boundary Negative Mass).** Let $\Omega_A^b$ denote the $b$-valued points in the reassembly neighborhood of $A$, and define $\Omega_B^a$ analogously. We assume

$$s_A := \sum_{p\in\Omega_A^b}\mathcal{K}_{A'}^{\text{diff}}(p) < 0, \qquad s_B := \sum_{p\in\Omega_B^a}\mathcal{K}_{B'}^{\text{diff}}(p) < 0. \tag{17}$$

*Proof.* By construction, the differential kernel satisfies

$$\sum_p \mathcal{K}_{l'}^{\text{diff}}(p) = 0, \qquad \left\|\mathcal{K}_{l'}^{\text{diff}}\right\|_1 < 2. \tag{18}$$

The strict magnitude bound follows from the positive numerical constant $\epsilon$ in the denominator. Because the kernel has zero sum, its positive and negative masses have equal absolute magnitude, each strictly less than 1. Any subset of kernel weights therefore has a sum strictly greater than $-1$. Together with Assumption 3, this gives $-1 < s_A, s_B < 0$.

Let $\gamma_{l'} = \mathcal{G}_{l'} \in (0, 1)$ for DiffUp+ and $\gamma_{l'} = 1$ for DiffUp, and define $\lambda = \max(\gamma_{A'}, \gamma_{B'})$. The identity base kernel preserves the values $a$ and $b$. Under the binary-partition condition, zero-sum differential reassembly gives

$$I(A') = a + \gamma_{A'}s_A(b-a), \qquad I(B') = b - \gamma_{B'}s_B(b-a). \tag{19}$$

Consequently,

$$I(B') - I(A') = (1 - \gamma_{A'}s_A - \gamma_{B'}s_B)\,(b-a). \tag{20}$$

Since $-1 < s_A, s_B < 0$ and $\gamma_{A'}, \gamma_{B'} > 0$,

$$1 < 1 - \gamma_{A'}s_A - \gamma_{B'}s_B < 1 + \gamma_{A'} + \gamma_{B'} \le 1 + 2\lambda. \tag{21}$$

Multiplying by $b - a > 0$ yields

$$|b-a| < |I(B') - I(A')| < (1+2\lambda)|b-a|\,. \tag{22}$$

Proof completes. □

### D.3 Proof of Conditional Regional Smoothness

**Formal Formulation of Assumption 2 (Center-Negative Kernel Pattern).** Let $c$ denote the center of the reassembly kernel. Within a homogeneous region, the differential kernel satisfies

$$\mathcal{K}_{l'}^{\text{diff}}(c) \leq 0, \qquad \mathcal{K}_{l'}^{\text{diff}}(p) \geq 0 \quad \text{for all } p \neq c. \tag{23}$$

*Proof.* Combining Assumption 2 with the zero-sum and magnitude constraints in Eq. (18), there exists $\beta_{l'} \in [0, 1)$ such that

$$\mathcal{K}_{l'}^{\text{diff}}(c) = -\beta_{l'}, \qquad \sum_{p \neq c} \mathcal{K}_{l'}^{\text{diff}}(p) = \beta_{l'}. \tag{24}$$

Indeed, zero sum makes the off-center mass equal to the absolute center weight, and the L1 norm is $2\beta_{l'} < 2$. With $\gamma_{l'} = \mathcal{G}_{l'} \in (0, 1)$ for DiffUp+ and $\gamma_{l'} = 1$ for DiffUp, adding the identity base kernel yields

$$\mathcal{K}_{l'}(c) = 1 - \gamma_{l'}\beta_{l'} > 0, \qquad \mathcal{K}_{l'}(p) = \gamma_{l'}\mathcal{K}_{l'}^{\text{diff}}(p) \geq 0 \quad \text{for } p \neq c. \tag{25}$$

Moreover, the differential component has zero sum and the identity base kernel has unit sum, so $\sum_p \mathcal{K}_{l'}(p) = 1$. Feature reassembly is therefore a convex combination of the local input values, implying

$$\min_{p \in \mathcal{N}_l^d(k_r)} I(p) \leq I(l') \leq \max_{p \in \mathcal{N}_l^d(k_r)} I(p). \tag{26}$$

This proves that reassembly introduces no new local extrema under Assumption 2.

Proof completes. □

The center and neighborhood masses are $1 - \gamma_{l'}\beta_{l'}$ and $\gamma_{l'}\beta_{l'}$, respectively; hence, the gate controls the interpolation between identity preservation and neighborhood averaging. When $\beta_{l'} = 0$, the operation reduces to identity reassembly.