FADE: Fusing the Assets of Decoder and Encoder for Task-Agnostic Upsampling

Hao Lu, Wenze Liu, Hongtao Fu, and Zhiguo Cao[✉]

School of Artificial Intelligence and Automation, Huazhong University
of Science and Technology, Wuhan 430074, China
{hlu,wzliu,htfu,zgcao}@hust.edu.cn

Abstract. We consider the problem of task-agnostic feature upsampling in dense prediction where an upsampling operator is required to facilitate both region-sensitive tasks like semantic segmentation and detail-sensitive tasks such as image matting. Existing upsampling operators often can work well in either type of the tasks, but not both. In this work, we present FADE, a novel, plug-and-play, and task-agnostic upsampling operator. FADE benefits from three design choices: i) considering encoder and decoder features jointly in upsampling kernel generation; ii) an efficient semi-shift convolutional operator that enables granular control over how each feature point contributes to upsampling kernels; iii) a decoder-dependent gating mechanism for enhanced detail delineation. We first study the upsampling properties of FADE on toy data and then evaluate it on large-scale semantic segmentation and image matting. In particular, FADE reveals its effectiveness and task-agnostic characteristic by consistently outperforming recent dynamic upsampling operators in different tasks. It also generalizes well across convolutional and transformer architectures with little computational overhead. Our work additionally provides thoughtful insights on what makes for task-agnostic upsampling. Code is available at: http://lnkiy.in/fade_in.

Keywords: Feature upsampling · Dense prediction · Dynamic networks · Semantic segmentation · Image matting

1 Introduction

Feature upsampling, which aims to recover the spatial resolution of features, is an indispensable stage in many dense prediction models [1,25,34,36,37,41]. Conventional upsampling operators, such as nearest neighbor (NN) or bilinear interpolation [15], deconvolution [40], and pixel shuffle [26], often have a preference of a specific task. For instance, bilinear interpolation is favored in semantic segmentation [4,37], and pixel shuffle is preferred in image super-resolution [14].
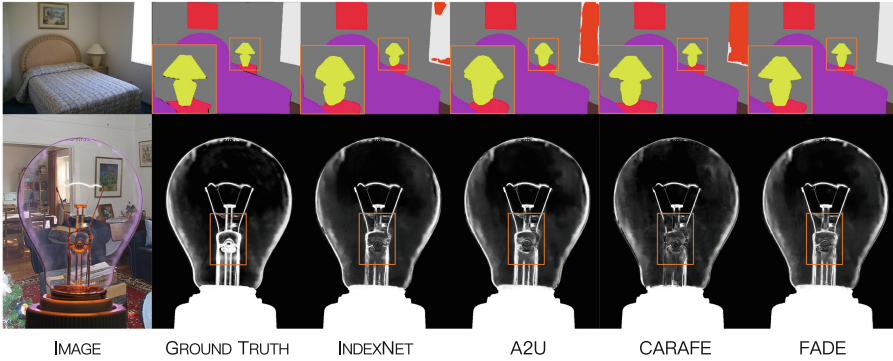
Fig. 1. Inferred segmentation masks and alpha mattes with different upsampling operators. The compared operators include IndexNet [19], A2U [7], CARAFE [32], and our proposed FADE. Among all competitors, only FADE generates both the high-quality mask and the alpha matte.

A main reason is that each dense prediction task has its own focus: some tasks like semantic segmentation [18] and instance segmentation [11] are region-sensitive, while some tasks such as image super-resolution [8] and image matting [19,39] are detail-sensitive. If one expects an upsampling operator to generate semantically consistent features such that a region can share the same class label, it is often difficult for the same operator to recover boundary details simultaneously, and vice versa. Indeed empirical evidence shows that bilinear interpolation and max unpooling [1] have inverse behaviors in segmentation and matting [19,20], respectively.

In an effort to evade 'trials-and-errors' from choosing an upsampling operator for a certain task at hand, there has been a growing interest in developing a generic upsampling operator for dense prediction recently [7,19,20,22,30,32,33]. For example, CARAFE [32] demonstrates its benefits on four dense prediction tasks, including object detection, instance segmentation, semantic segmentation, and image inpainting. IndexNet [19] also boosts performance on several tasks such as image matting, image denoising, depth prediction, and image reconstruction. However, a comparison between CARAFE and IndexNet [20] indicates that neither CARAFE nor IndexNet can defeat its opponent on both region- and detail-sensitive tasks (CARAFE outperforms IndexNet on segmentation, while IndexNet is superior than CARAFE on matting), which can also be observed from the inferred segmentation masks and alpha mattes in Fig. 1. This raises an interesting question: *Does there exist a unified form of upsampling operator that is truly task-agnostic?*

To answer the question above, we present FADE, a novel, plug-and-play, and task-agnostic upsampling operator which Fuses the Assets of Decoder and Encoder (FADE). The name also implies its working mechanism: upsampling features in a 'fade-in' manner, from recovering spatial structure to delineating subtle details. In particular, we argue that an ideal upsampling operator should

be able to preserve the semantic information and compensate the detailed information lost due to downsampling. The former is embedded in decoder features; the latter is abundant in encoder features. Therefore, we hypothesize that it is the insufficient use of encoder and decoder features bringing the task dependency of upsampling, and our idea is to design FADE to make the best use of encoder and decoder features, inspiring the following insights and contributions:

i) By exploring why CARAFE works well on region-sensitive tasks but poorly on detail-sensitive tasks, and why IndexNet and A2U [7] behave conversely, we observe that what features (encoder or decoder) to use to generate the upsampling kernels matters. Using decoder features can strengthen the regional continuity, while using encoder features helps recover details. It is thus natural to seek whether combining encoder and decoder features enjoys both merits, which underpins the core idea of FADE.

ii) To integrate encoder and decoder features, a subsequent problem is how to deal with the resolution mismatch between them. A standard way is to implement UNet-style fusion [25], including feature interpolation, feature concatenation, and convolution. However, we show that this naive implementation can have a negative effect on upsampling kernels. To solve this, we introduce a semi-shift convolutional operator that unifies channel compression, concatenation, and kernel generation. Particularly, it allows granular control over how each feature point participates in the computation of upsampling kernels. The operator is also fast and memory-efficient due to direct execution of cross-resolution convolution, without explicit feature interpolation for resolution matching.

iii) To enhance detail delineation, we further devise a gating mechanism, conditioned on decoder features. The gate allows selective pass of fine details in the encoder features as a refinement of upsampled features.

We conduct experiments on five data sets covering three dense prediction tasks. We first validate our motivation and the rationale of our design through several toy-level and small-scale experiments, such as binary image segmentation on Weizmann Horse [2], image reconstruction on Fashion-MNIST [35], and semantic segmentation on SUN RGBD [27]. We then present a thorough evaluation of FADE on large-scale semantic segmentation on ADE20K [42] and image matting on Adobe Composition-1K [39]. FADE reveals its task-agnostic characteristic by consistently outperforming state-of-the-art upsampling operators on both region- and detail-sensitive tasks, while also retaining the lightweight property by appending relatively few parameters and FLOPs. It has also good generalization across convolutional and transformer architectures [13,37].

To our knowledge, FADE is the first task-agnostic upsampling operator that performs favorably on both region- and detail-sensitive tasks.

2 Related Work

Feature Upsampling. Unlike joint image upsampling [12,31], feature upsampling operators are mostly developed in the deep learning era, to respond to

the need for recovering spatial resolution of encoder features (decoding). Conventional upsampling operators typically use fixed/hand-crafted kernels. For instance, the kernels in the widely used NN and bilinear interpolation are defined by the relative distance between pixels. Deconvolution [40], *a.k.a.* transposed convolution, also applies a fixed kernel during inference, despite the kernel parameters are learned. Pixel shuffle [26] instead only includes memory operations but still follows a specific rule in upsampling by reshaping the depth channel into the spatial channels. Among hand-crafted operators, unpooling [1] perhaps is the only operator that has a dynamic upsampling behavior, *i.e.*, each upsampled position is data-dependent conditioned on the max operator. Recently the importance of the dynamic property has been proved by some dynamic upsampling operators [7,19,32]. CARAFE [32] implements context-aware reassembly of features, IndexNet [19] provides an indexing perspective of upsampling, and A2U [7] introduces affinity-aware upsampling. At the core of these operators is the data-dependent upsampling kernels whose kernel parameters are predicted by a sub-network. This points out a promising direction from considering generic feature upsampling. FADE follows the vein of dynamic feature upsampling.

Dense Prediction. Dense prediction covers a broad class of per-pixel labeling tasks, ranging from mainstream object detection [23], semantic segmentation [18], instance segmentation [11], and depth estimation [9] to low-level image restoration [21], image matting [39], edge detection [38], and optical flow estimation [29], to name a few. An interesting property about dense prediction is that a task can be region-sensitive or detail-sensitive. The sensitivity is closely related to what metric is used to assess the task. In this sense, semantic/instance segmentation is region-sensitive, because the standard Mask Intersection-over-Union (IoU) metric [10] is mostly affected by regional mask prediction quality, instead of boundary quality. On the contrary, image matting can be considered detail-sensitive, because the error metrics [24] are mainly computed from trimap regions that are full of subtle details or transparency. Note that, when we emphasize region sensitivity, we do not mean that details are not important, and vice versa. In fact, the emergence of Boundary IoU [5] implies that the limitation of a certain evaluation metric has been noticed by our community. The goal of developing a task-agnostic upsampling operator capable of both regional preservation and detail delineation can have a board impact on a number of dense prediction tasks. In this work, we mainly evaluate upsampling operators on semantic segmentation and image matting, which may be the most representative region- and detail-sensitive task, respectively.

3   Task-Agnostic Upsampling: A Trade-off Between Semantic Preservation and Detail Delineation

Before we present FADE, we share some of our view points towards task-agnostic upsampling, which may be helpful to understand our designs in FADE.

How Encoder and Decoder Features Affect Upsampling. In dense prediction models, downsampling stages are involved to acquire a large receptive field,
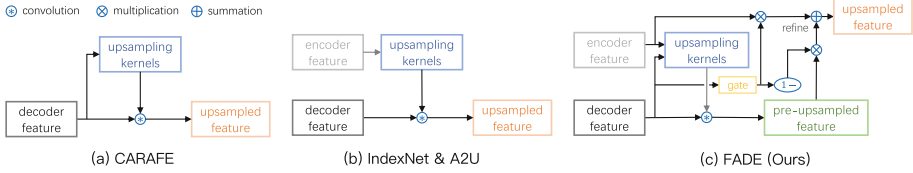
Fig. 2. Use of encoder and/or decoder features in different upsampling operators. (a) CARAFE generates upsampling kernels conditioned on decoder features, while (b) IndexNet and A2U generate kernels using encoder features only. By contrast, (c) FADE considers both encoder and decoder features not only in upsampling kernel generation but also in gated feature refinement.

bringing the need of peer-to-peer upsampling stages to recover the spatial resolution, which together constitutes the basic encoder-decoder architecture. During downsampling, details of high-resolution features are impaired or even lost, but the resulting low-resolution encoder features often have good semantic meanings that can pass to decoder features. Hence, we believe an ideal upsampling operator should appropriately resolve two issues: 1) preserve the semantic information already extracted; 2) compensate as many lost details as possible without deteriorating the semantic information. NN or bilinear interpolation only meets the former. This conforms to our intuition that interpolation often smooths features. A reason is that low-resolution decoder features have no prior knowledge about missing details. Other operators that directly upsample decoder features, such as deconvolution and pixel shuffle, can have the same problem with poor detail compensation. Compensating details requires high-resolution encoder features. This is why unpooling that stores indices before downsampling has good boundary delineation [19], but it hurts the semantic information due to zero-filling.

Dynamic upsampling operators, including CARAFE [32], IndexNet [19], and A2U [7], alleviate the problems above with data-dependent upsampling kernels. Their upsampling modes are illustrated in Fig. 2(a)-(b). From Fig. 2, it can be observed that, CARAFE generates upsampling kernels conditioned on decoder features, while IndexNet [19] and A2U [7] generate kernels via encoder features. This may explain the inverse behavior between CARAFE and IndexNet/A2U on region- or detail-sensitive tasks [20]. In this work, we find that generating upsampling kernels using either encoder or decoder features can lead to suboptimal results, and it is critical *to leverage both encoder and decoder features for task-agnostic upsampling*, as implemented in FADE (Fig. 2(c)).

How Each Feature Point Contributes to Upsampling Matters. After deciding what the features to use, the follow-up question is how to use the features effectively and efficiently. The main obstacle is the mismatched resolution between encoder and decoder feature maps. One may consider simple interpolation for resolution matching, but we find that this leads to sub-optimal upsampling. Considering the case of applying $\times 2$ NN interpolation to decoder features, if we apply 3×3 convolution to generate the upsampling kernel, the effective receptive field of the kernel can be reduced to be $< 50\%$: before interpolation there are 9 valid

points in a 3×3 window, but only 4 valid points are left after interpolation, as shown in Fig. 5(a). Besides this, there is another more important issue. Still in $\times 2$ upsampling, as shown in Fig. 5(a), the four windows which control the variance of upsampling kernels w.r.t. the 2×2 neighbors of high resolution are influenced by the hand-crafted interpolation. Controlling a high-resolution upsampling kernel map, however, is blind with the low-resolution decoder feature. It contributes little to an informative upsampling kernel, especially to the variance of the four neighbors in the upsampling kernel map. Interpolation as a bias of that variance can even worsen the kernel generation. A more reasonable choice may be to *let encoder and decoder features cooperate to control the overall upsampling kernel, but let the encoder feature alone control the variance of the four neighbors.* This insight exactly motivates the design of semi-shift convolution (Sect. 4).

Exploiting Encoder Features for Further Detail Refinement. Besides helping structural recovery via upsampling kernels, there remains much useful information in the encoder features. Since encoder features only go through a few layers of a network, they preserve 'fine details' of high resolution. In fact, nearly all dense prediction tasks require fine details, *e.g.*, despite regional prediction dominates in instance segmentation, accurate boundary prediction can also significantly boost performance [28], not to mention the stronger request of fine details in detail-sensitive tasks. *The demands of fine details in dense prediction need further exploitation of encoder features.* Instead of simply skipping the encoder features, we introduce a gating mechanism that leverages decoder features to guide where the encoder features can pass through.

4   Fusing the Assets of Decoder and Encoder

Dynamic Upsampling Revisited. Here we review some basic operations in recent dynamic upsampling operators such as CARAFE [32], IndexNet [19], and A2U [7]. Figure 2 briefly summarizes their upsampling modes. They share an identical pipeline, *i.e.*, first generating data-dependent upsampling kernels, and then reassembling the decoder features using the kernels. Typical dynamic upsampling kernels are content-aware, but channel-shared, which means each position has a unique upsampling kernel in the spatial dimension, but the same ones are shared in the channel dimension.

CARAFE learns upsampling kernels directly from decoder features and then reassembles them to high resolution. In particular, the decoder features pass through two consecutive convolutional layers to generate the upsampling kernels, of which the former is a channel compressor implemented by 1×1 convolution to reduce the computational complexity and the latter is a content encoder with 3×3 convolution, and finally the softmax function is used to normalize the kernel weights. IndexNet and A2U, however, adopt more sophisticated modules to leverage the merit of encoder features. Further details can be referred to [7,19,32].

FADE is designed to maintain the simplicity of dynamic upsampling. Hence, it generally follows the pipeline of CARAFE, but further optimizes the process
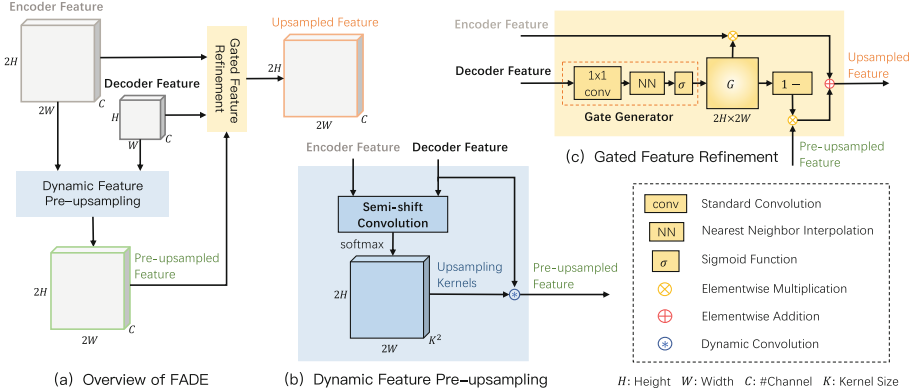
Fig. 3. Technical pipeline of FADE. From (a) the overview of FADE, feature upsampling is executed by jointly exploiting the encoder and decoder feature with two key modules. In (b) dynamic feature pre-upsampling, they are used to generate upsampling kernels using a semi-shift convolutional operator (Fig. 5). The kernels are then used to reassemble the decoder feature into pre-upsampled feature. In (c) gated feature refinement, the encoder and pre-upsampled features are modulated by a decoder-dependent gating mechanism to enhance detail delineation before generating the final upsampled feature.

of kernel generation with semi-shift convolution, and the channel compressor will also function as a way of pre-fusing encoder and decoder features. In addition, FADE also includes a gating mechanism for detail refinement. The overall pipeline of FADE is summarized in Fig. 3.

Generating Upsampling Kernels from Encoder and Decoder Features. We first showcase a few visualizations on some small-scale or toy-level data sets to highlight the importance of both encoder and decoder features for task-agnostic upsampling. We choose semantic segmentation on SUN RGBD [27] as the region-sensitive task and image reconstruction on Fashion MNIST [35] as the detail-sensitive one. We follow the network architectures and the experimental settings in [20]. Since we focus on upsampling, all downsampling stages use max pooling. Specifically, to show the impact of encoder and decoder features, in the segmentation experiments, we all use CARAFE but only modify the source of features used for generating upsampling kernels. We build three baselines: 1) *decoder-only*, the implementation of CARAFE; 2) *encoder-only*, where the upsampling kernels are generated from encoder features; 3) *encoder-decoder*, where the upsampling kernels are generated from the concatenation of encoder and NN-interpolated decoder features. We report Mask IoU (mIoU) [10] and Boundary IoU (bIoU) [5] for segmentation, and report Peak Signal-to-Noise Ratio (PSNR), Structural SIMilarity index (SSIM), Mean Absolute Error (MAE), and root Mean Square Error (MSE) for reconstruction. From Table 1, one can observe that the encoder-only baseline outperforms the decoder-only one in image reconstruction, but in semantic segmentation the trend is on the contrary. To understand why, we visualize the

Table 1. Results of semantic segmentation on SUN RGBD and image reconstruction on Fashion MNIST. Best performance is in **boldface**.

	Segmentation		Reconstruction			
	accuracy metric↑		accuracy metric↑		error metric↓	
	mIoU	bIoU	PSNR	SSIM	MAE	MSE
Decoder-only	37.00	25.61	24.35	87.19	0.0357	0.0643
Encoder-only	36.71	27.89	32.25	97.73	0.0157	0.0257
Encoder-decoder	**37.59**	**28.80**	**33.83**	**98.47**	**0.0122**	**0.0218**

DECODER-ONLY   ENCODER-ONLY   ENCODER-DECODER
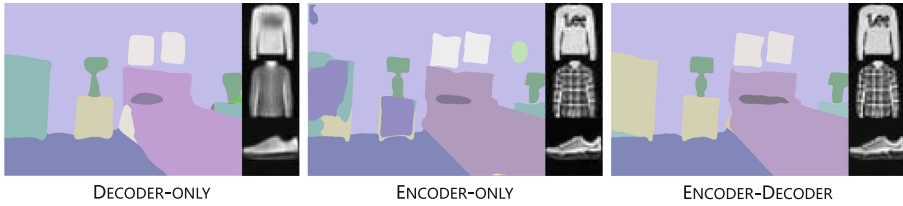
Fig. 4. Visualizations of inferred mask and reconstructed results on SUN RGBD and Fashion-MNIST. The decoder-only model generates good regional prediction but poor boundaries/textures, while the encoder-only one is on the contrary. When fusing encoder and decoder features, both region and detail predictions are improved, *e.g.*, the table lamp and stripes on clothes.

segmentation masks and reconstructed results in Fig. 4. We find that in segmentation the decoder-only model tends to produce region-continuous output, while the encoder-only one generates clear mask boundaries but blocky regions; in reconstruction, by contrast, the decoder-only model almost fails and can only generate low-fidelity reconstructions. It thus can be inferred that, encoder features help to predict details, while decoder features contribute to semantic preservation of regions. Indeed, by considering both encoder and decoder features, the resulting mask seems to integrate the merits of the former two, and the reconstructions are also full of details. Therefore, albeit a simple tweak, FADE significantly benefits from generating upsampling kernels with both encoder and decoder features, as illustrated in Fig. 2(c).

Semi-shift Convolution. Given encoder and decoder features, we next address how to use them to generate upsampling kernels. We investigate two implementations: a naive implementation and a customized implementation. The key difference between them is how each decoder feature point spatially corresponds to each encoder feature point. The naive implementation shown in Fig. 5(a) includes four operations: i) feature interpolation, ii) concatenation, iii) channel compression, iv) standard convolution for kernel generation, and v) softmax normalization. As aforementioned in Sect. 3, naive interpolation can have a few problems. To address them, we present semi-shift convolution that simplifies the first four operations above into a unified operator, which is schematically illustrated in Fig. 5(b). Note that the 4 convolution windows in encoder features all correspond to the same
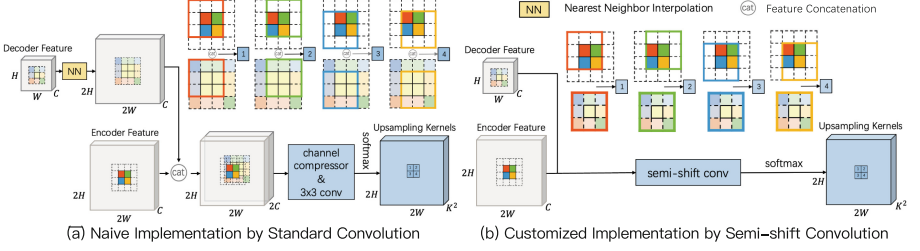
Fig. 5. Two forms of implementations for generating upsampling kernels. Naive implementation requires matching resolution with explicit feature interpolation and concatenation, followed by channel compression and standard convolution for kernel prediction. Our customized implementation simplifies the whole process with only semi-shift convolution.

window in decoder features. This design has the following advantages: 1) the role of control in the kernel generation is made clear where the control of the variance of 2×2 neighbors is moved to encoder features completely; 2) the receptive field of decoder features is kept consistent with that of encoder features; 3) memory cost is reduced, because semi-shift convolution directly operates on low-resolution decoder features, without feature interpolation; 4) channel compression and 3×3 convolution can be merged in semi-shift convolution. Mathematically, the single window processing with naive implementation or semi-shift convolution has an identical form if ignoring the content of feature maps. For example, considering the top-left window ('1' in Fig. 5), the (unnormalized) upsampling kernel weight has the form

$$w_m = \sum_{l=1}^{d} \sum_{i=1}^{h} \sum_{j=1}^{h} \beta_{ijlm} \left(\sum_{k=1}^{2C} \alpha_{kl} x_{ijk} + a_l \right) + b_m \tag{1}$$

$$= \sum_{l=1}^{d} \sum_{i=1}^{h} \sum_{j=1}^{h} \beta_{ijlm} \left(\sum_{k=1}^{C} \alpha_{kl}^{\mathsf{en}} x_{ijk}^{\mathsf{en}} + \sum_{k=1}^{C} \alpha_{kl}^{\mathsf{de}} x_{ijk}^{\mathsf{de}} + a_l \right) + b_m \tag{2}$$

$$= \sum_{l=1}^{d} \sum_{i=1}^{h} \sum_{j=1}^{h} \beta_{ijlm} \sum_{k=1}^{C} \alpha_{kl}^{\mathsf{en}} x_{ijk}^{\mathsf{en}} + \sum_{l=1}^{d} \sum_{i=1}^{h} \sum_{j=1}^{h} \beta_{ijlm} \left(\sum_{k=1}^{C} \alpha_{kl}^{\mathsf{de}} x_{ijk}^{\mathsf{de}} + a_l \right) + b_m \tag{3}$$

where $w_m, m = 1, ..., K^2$, is the weight of the upsampling kernel, K the upsampling kernel size, h the convolution window size, C the number of input channel dimension of encoder and decoder features, and d the number of compressed channel dimension. $\alpha_{kl}^{\mathsf{en}}$ and $\{\alpha_{kl}^{+}\mathsf{de}, a_l\}$ are the parameters of 1×1 convolution specific to encoder and decoder features, respectively, and $\{\beta_{ijlm}, b_m\}$ the parameters of 3×3 convolution. Following CARAFE, we fix $h = 3$, $K = 5$ and $d = 64$.

According to Eq. (3), by the linearity of convolution, Eq. (1) and Eq. (2) are equivalent to applying two distinct 1×1 convolutions to C-channel encoder and C-channel decoder features, respectively, followed by a shared 3×3 convolution

Table 2. The results on the Weizmann Horse dataset.

SegNet – baseline	mIoU
Unpooling	93.42
IndexNet [19]	93.00
NN	89.15
CARAFE [32]	89.29
NN + Gate	95.26
CARAFE + Gate	95.25

GRADIENT MAP   GATE MAP
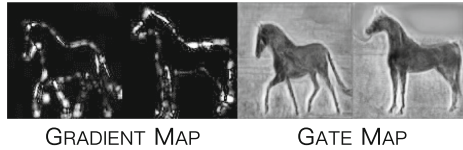
Fig. 6. Gradient maps and gate maps of horses.

and summation. Equation (3) allows us to process encoder and decoder features without matching their resolution. To process the whole feature map, the window can move s steps on encoder features but only $\lfloor s/2 \rfloor$ steps on decoder features. This is why the operator is given the name 'semi-shift convolution'. To implement this efficiently, we split the process to 4 sub-processes; each sub-process focuses on the top-left, top-right, bottom-left, and bottom-right windows, respectively. Different sub-processes have also different prepossessing strategies. For example, for the top-left sub-process, we add full padding to the decoder feature, but only add padding on top and left to the encoder feature. Then all the top-left window correspondences can be satisfied by setting stride of 1 for the decoder feature and 2 for the encoder feature. Finally, after a few memory operations, the four sub-outputs can be reassembled to the expected upsampling kernel, and the kernel is used to reassemble decoder features to generate pre-upsampled features, as shown in Fig. 3(b).

Extracting Fine Details from Encoder Features. Here we further introduce a gating mechanism to complement fine details from encoder features to pre-upsampled features. We again use some experimental observations to showcase our motivation. We use a binary image segmentation dataset, Weizmann Horse [2]. The reasons for choosing this dataset are two-fold: (1) visualization is made simple; (2) the task is simple such that the impact of feature representation can be neglected. When all baselines have nearly perfect region predictions, the difference in detail prediction can be amplified. We use SegNet pretrained on ImageNet as the baseline and alter only the upsampling operators. Results are listed in Table 2. An interesting phenomenon is that CARAFE works almost the same as NN interpolation and even falls behind the default unpooling and IndexNet. An explanation is that the dataset is too simple such that the region smoothing property of CARAFE is wasted, but recovering details matters.

A common sense in segmentation is that, the interior of a certain class would be learned fast, while mask boundaries are difficult to predict. This can be observed from the gradient maps w.r.t. an intermediate decoder layer, as shown in Fig. 6. During the middle stage of training, most responses are near boundaries. Now that gradients reveal the demand of detail information, feature maps

would also manifest this requisite with some distributions, e.g., in multi-class semantic segmentation a confident class prediction in a region would be a uni-modal distribution along the channel dimension, and an uncertain prediction around boundaries would likely be a bimodal distribution. Hence, we assume that all decoder layers have gradient-imposed distribution priors and can be encoded to inform the requisite of detail or semantic information. In this way fine details can be chosen from encoder features without hurting the semantic property of decoder features. Hence, instead of directly skipping encoder features as in feature pyramid networks [16], we introduce a gating mechanism [6] to selectively refine pre-upsampled features using encoder features, conditioned on decoder features. The gate is generated through a 1×1 convolution layer, a NN interpolation layer, and a `sigmoid` function. As shown in Fig. 3(c), the decoder feature first goes through the gate generator, and the generator then outputs a gate map instantiated in Fig. 6. Finally, the gate map G modulates the encoder feature $\mathcal{F}_{\texttt{encoder}}$ and the pre-upsampled feature $\mathcal{F}_{\texttt{pre-upsampled}}$ to generate the final upsampled feature $\mathcal{F}_{\texttt{upsampled}}$ as

$$\mathcal{F}_{\texttt{upsampled}} = \mathcal{F}_{\texttt{encoder}} \cdot G + \mathcal{F}_{\texttt{pre-upsampled}} \cdot (1 - G) \,. \tag{4}$$

From Table 2, the gating mechanism works on both NN and CARAFE.

5   Results and Discussions

Here we formally validate FADE on large-scale dense prediction tasks, including image matting and semantic segmentation. We also conduct ablation studies to justify each design choice of FADE. In addition, we analyze computational complexity in terms of parameter counts and GFLOPs.

5.1   Image Matting

Image matting [39] is chosen as the representative of the detail-sensitive task. It requires a model to estimate the accurate alpha matte that smoothly splits foreground from background. Since ground-truth alpha mattes can exhibit significant differences among local regions, estimations are sensitive to a specific upsampling operator used [7,19].

Data Set, Metrics, Baseline, and Protocols. We conduct experiments on the Adobe Image Matting dataset [39], whose training set has 431 unique foreground objects and ground-truth alpha mattes. Following [7], instead of compositing each foreground with fixed 100 background images chosen from MS COCO [17], we randomly choose background images in each iteration and generate composited images on-the-fly. The Composition-1K testing set has 50 unique foreground objects, and each is composited with 20 background images from PASCAL VOC [10]. We report the widely used Sum of Absolute Differences (SAD), Mean Squared Error (MSE), Gradient (Grad), and Connectivity (Conn) and evaluate them using the code provided by [39].

Table 3. Image matting and semantic segmentation results on the Adobe Composition-1k and ADE20K data sets. ΔParam. indicates the additional number of parameters compared with the bilinear baseline. Best performance is in **boldface**.

A2U matting/	Matting − error↓					Segm − accuracy↑		
SegFormer	SAD	MSE	Grad	Conn	ΔParam	mIoU	bIoU	ΔParam
Bilinear	37.31	0.0103	21.38	35.39	8.05M	41.68	27.80	13.7M
CARAFE [32]	41.01	0.0118	21.39	39.01	+0.26M	42.82	29.84	+0.44M
IndexNet [19]	34.28	0.0081	15.94	31.91	+12.26M	41.50	28.27	+12.60M
A2U [7]	32.15	0.0082	16.39	29.25	+38K	41.45	27.31	+0.12M
FADE (Ours)	**31.10**	**0.0073**	**14.52**	**28.11**	+0.12M	**44.41**	**32.65**	+0.29M

A2U Matting [7] is adopted as the baseline. Following [7], the baseline network adopts a backbone of the first 11 layers of ResNet-34 with in-place activated batchnorm [3] and a decoder consisting of a few upsampling stages with shortcut connections. Readers can refer to [7] for the detailed architecture. To control variables, we use max pooling in downsampling stages consistently and only alter upsampling operators to train the model. We strictly follow the training configurations and data augmentation strategies used in [7].

Matting Results. We compare FADE with other state-of-the-art upsampling operators. Quantitative results are shown in Table 3. Results show that FADE consistently outperforms other competitors in all metrics, with also few additional parameters. It is worth noting that IndexNet and A2U are strong baselines that are delicately designed upsampling operators for image matting. Also the worst performance of CARAFE indicates that upsampling with only decoder features cannot meet a detail-sensitive task. Compared with standard bilinear upsampling, FADE invites 16%∼32% relative improvement, which suggests upsampling can indeed make a difference, and our community should shift more attention to upsampling. Qualitative results are shown in Fig. 1. FADE generates a high-fidelity alpha matte.

5.2   Semantic Segmentation

Semantic segmentation is chosen as the representative region-sensitive task. To prove that FADE is architecture-independent, SegFormer [37], a recent transformer based segmentation model, is used as the baseline.

Data Set, Metrics, Baseline, and Protocols. We use the ADE20K dataset [42], which is a standard benchmark used to evaluate segmentation models. ADE20K covers 150 fine-grained semantic concepts, including 20210 images in the training set and 2000 images in the validation set. In addition to reporting the standard Mask IoU (mIoU) metric [10], we also include the Boundary IoU (bIoU) metric [5] to assess boundary quality.

Table 4. Ablation study on the source of features, the way for upsampling kernel generation, and the effect of the gating mechanism. Best performance is in **boldface**. en: encoder; de: decoder.

| No.	A2U MattingSegFormer			Matting - error ↓				Segm - accuracy ↑	
	source of feat	Kernel gen	Fusion	SAD	MSE	Grad	Conn	mIoU	bIoU
B1	en			34.22	0.0087	15.90	32.03	42.75	31.00
B2	de			41.01	0.0118	21.39	39.01	42.82	29.84
B3	en & de	naive		32.41	0.0083	16.56	29.82	43.27	31.55
B4	en & de	semi-shift		31.78	0.0075	15.12	28.95	43.33	32.06
B5	en & de	semi-shift	skipping	32.64	0.0076	15.90	29.92	43.22	31.85
B6	en & de	semi-shift	gating	**31.10**	**0.0073**	**14.52**	**28.11**	**44.41**	**32.65**

SegFormer-B1 [37] is chosen by considering both the effectiveness and computational sources at hand. We keep the default model architecture in SegFomer except for modifying the upsampling stage in the MLP head. All training settings and implementation details are kept the same as in [37].

Segmentation Results. Quantitative results of different upsampling operators are also listed in Table 3. Similar to matting, FADE is the best performing upsampling operator in both mIoU and bIoU metrics. Note that, among compared upsampling operators, FADE is the only operator that exhibits the task-agnostic property. A2U is the second best operator in matting, but turns out to be the worst one in segmentation. CARAFE is the second best operator in segmentation, but is the worst one in matting. This implies that current dynamic operators still have certain weaknesses to achieve task-agnostic upsampling. Qualitative results are shown in Fig. 1. FADE generates high-quality prediction both within mask regions and near mask boundaries.

5.3 Ablation Study

Here we justify how performance is affected by the source of features, the way for upsampling kernel generation, and the use of the gating mechanism. We build six baselines based on FADE:

1) B1: *encoder-only*. Only encoder features go through 1×1 convolution for channel compression (64 channels), followed by 3×3 convolution layer for kernel generation;
2) B2: *decoder-only*. This is the CARAFE baseline [32]. Only decoder features go through the same 1×1 and 3×3 convolution for kernel generation, followed by Pixel Shuffle as in CARAFE due to different spatial resolution;
3) B3: *encoder-decoder-naive*. NN-interpolated decoder features are first concatenated with encoder features, and then the same two convolutional layers are applied;
4) B4: *encoder-decoder-semi-shift*. Instead of using NN interpolation and standard convolutional layers, we use semi-shift convolution to generate kernels directly as in FADE;

(a) resolution=112 × 112   **(b)** channel=64   **(c)** channel=256
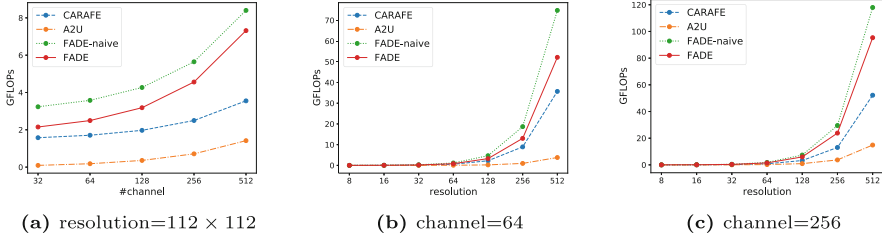
Fig. 7. GFLOPs comparison between FADE and other upsampling operators.

5) B5: B4 with *skipping*. We directly skip the encoder features as in feature pyramid networks [16];
6) B6: B4 with *gating*. The full implementation of FADE.

Results are shown in Table 4. By comparing B1, B2, and B3, the experimental results give a further verification on the importance of both encoder and decoder features for upsampling kernel generation. By comparing B3 and B4, the results indicate a clear advantage of semi-shift convolution over naive implementation in the way of generating upsampling kernels. As aforementioned, the rationale that explains such a superiority can boil down to the granular control of the contribution of each feature point in kernels (Sect. 4). We also note that, even without gating, the performance of FADE already surpasses other upsampling operators (B4 vs. Table 3), which means the task-agnostic property is mainly due to the joint use of encoder and decoder features and the semi-shift convolution. In addition, skipping is clearly not the optimal way to move encoder details to decoder features, at least worse than the gating mechanism (B5 vs. B6).

5.4   Comparison of Computational Overhead

A favorable upsampling operator, being part of overall network architecture, should not significantly increase the computation cost. This issue is not well addressed in IndexNet as it significantly increases the number of parameters and computational overhead [19]. Here we measure GFLOPs of some upsampling operators by i) changing number of channels given fixed spatial resolution and by ii) varying spatial resolution given fixed number of channels. Figure 7 suggests FADE is also competitive in GFLOPs, especially when upsampling with relatively low spatial resolution and low channel numbers. In addition, semi-shift convolution can be considered a perfect replacement of the standard 'interpolation+convolution' paradigm for upsampling, not only superior in effectiveness but also in efficiency.

6   Conclusions

In this paper, we propose FADE, a novel, plug-and-play, and task-agnostic upsampling operator. For the first time, FADE demonstrates the feasibility of

task-agnostic feature upsampling in both region- and detail-sensitive dense prediction tasks, outperforming the best upsampling operator A2U on image matting and the best operator CARAFE on semantic segmentation. With step-to-step analyses, we also share our view points from considering what makes for generic feature upsampling.

For future work, we plan to validate FADE on additional dense prediction tasks and also explore the peer-to-peer downsampling stage. So far, FADE is designed to maintain the simplicity by only implementing linear upsampling, which leaves much room for further improvement, *e.g.*, with additional nonlinearity. In addition, we believe how to strengthen the coupling between encoder and decoder features to enable better cooperation can make a difference for feature upsampling.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: SegNet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(12), 2481–2495 (2017)
2. Borenstein, E., Ullman, S.: Class-specific, top-down segmentation. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2351, pp. 109–122. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47967-8_8
3. Bulo, S.R., Porzi, L., Kontschieder, P.: In-place activated batchnorm for memory-optimized training of DNNs. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 5639–5647 (2018)
4. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Encoder-decoder with atrous separable convolution for semantic image segmentation. LNCS, vol. 11211, pp. 833–851. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_49
5. Cheng, B., Girshick, R., Dollár, P., Berg, A.C., Kirillov, A.: Boundary IOU: improving object-centric image segmentation evaluation. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 15334–15342 (2021)
6. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. arXiv Computer Research Repository (2014)
7. Dai, Y., Lu, H., Shen, C.: Learning affinity-aware upsampling for deep image matting. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 6841–6850 (2021)
8. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2015)
9. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS), pp. 2366–2374 (2014)
10. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)

11. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 2961–2969 (2017)
12. He, K., Sun, J., Tang, X.: Guided image filtering. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6311, pp. 1–14. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15549-9_1
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 770–778 (2016)
14. Ignatov, A., Timofte, R., Denna, M., Younes, A.: Real-time quantized image super-resolution on mobile NPUS, mobile AI 2021 challenge: report. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 2525–2534, June 2021
15. Lin, G., Milan, A., Shen, C., Reid, I.: RefineNet: multi-path refinement networks for high-resolution semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 1925–1934 (2017)
16. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 2117–2125 (2017)
17. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
18. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 3431–3440 (2015)
19. Lu, H., Dai, Y., Shen, C., Xu, S.: Indices matter: learning to index for deep image matting. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 3266–3275 (2019)
20. Lu, H., Dai, Y., Shen, C., Xu, S.: Index networks. IEEE Trans. Pattern Anal. Mach. Intell. **44**(1), 242–255 (2022)
21. Mao, X., Shen, C., Yang, Y.B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In: Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS), pp. 2802–2810 (2016)
22. Mazzini, D.: Guided upsampling network for real-time semantic segmentation. In: Proceedings of British Machine Vision Conference (BMVC) (2018)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS), 28 (2015)
24. Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., Rott, P.: A perceptually motivated online benchmark for image matting. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 1826–1833 (2009)
25. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
26. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 1874–1883 (2016)
27. Song, S., Lichtenberg, S.P., Xiao, J.: SUN RGB-D: A RGB-D scene understanding benchmark suite. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 567–576 (2015)

28. Tang, C., Chen, H., Li, X., Li, J., Zhang, Z., Hu, X.: Look closer to segment better: Boundary patch refinement for instance segmentation. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 13926–13935 (2021)

29. Teed, Z., Deng, J.: RAFT: recurrent all-pairs field transforms for optical flow. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12347, pp. 402–419. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_24

30. Tian, Z., He, T., Shen, C., Yan, Y.: Decoders matter for semantic segmentation: data-dependent decoding enables flexible feature aggregation. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 3126–3135 (2019)

31. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 839–846. IEEE (1998)

32. Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C.C., Lin, D.: CARAFE: context-aware reassembly of features. In: Proc. IEEE/CVF International Conference on Computer Vision (ICCV) (2019)

33. Wang, J., et al.: CARAFE++: unified content-aware ReAssembly of FEatures. IEEE Trans. Pattern Anal. Mach. Intell. (2021)

34. Wang, J., et al.: Deep high-resolution representation learning for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3349–3364 (2020)

35. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv Computer Research Repository (2017)

36. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11209, pp. 432–448. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01228-1_26

37. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: simple and efficient design for semantic segmentation with transformers. In: Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS) (2021)

38. Xie, S., Tu, Z.: Holistically-nested edge detection. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 1395–1403 (2015)

39. Xu, N., Price, B., Cohen, S., Huang, T.: Deep image matting. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 2970–2979 (2017)

40. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53

41. Zheng, S., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR), pp. 6881–6890 (2021)

42. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of IEEE Conference on Computer Vision Pattern Recognition (CVPR) (2017)