Location-aware Upsampling for Semantic Segmentation

Xiangyu He   Zitao Mo   Qiang Chen   Anda Cheng   Peisong Wang
Jian Cheng

NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China

xiangyu.he, zitao.mo, qiang.chen, anda.cheng, peisong.wang, jcheng@nlpr.ia.ac.cn

Abstract

Many successful learning targets such as dice loss and cross-entropy loss have enabled unprecedented breakthroughs in segmentation tasks. Beyond semantic supervision, this paper aims to introduce location prediction into semantic segmentation from a new viewpoint: let pixels determine their own coordinates. Based on this idea, we present a Location-aware Upsampling (LaU) that adaptively refines the interpolating coordinates with trainable offsets. Then, location-aware losses are established by encouraging pixels to move towards well-classified locations. An LaU is offset prediction coupled with interpolation, which is trained end-to-end to generate confidence score at each position from coarse to fine. Guided by location-aware losses, the new module can replace its plain counterpart (e.g., bilinear upsampling) in a plug-and-play manner to further boost the leading encoder-decoder approaches. Extensive experiments validate the consistent improvement over the state-of-the-art methods on benchmark datasets. Our code is available at https://github.com/HolmesShuan/Location-aware-Upsampling-for-Semantic-Segmentation.

1. Introduction

Recent advances in deep learning have empowered a wide range of applications [23, 36, 19, 41, 12], including semantic segmentation [51, 47, 29, 6]. The pioneering Fully Convolutional Network (FCN) [29] with deconvolutional operations [37] dramatically surpasses classical methods relying on hand-crafted features. Following FCNs, a series of CNN-based methods further enhance the state-of-the-art by introducing dilated (atrous) convolutions [47, 7], shortcut connections [1] and CRFs [45, 2, 27]. Latest semantic segmentation methods [50, 8] exploit fully trainable encoder-decoder architectures with simple bilinear upsamplings to extract high-resolution predictions.

Although recent works have exhibited state-of-the-art performance, we suggest that the location information hidden in the label map has been overlooked. To introduce location prediction into segmentation models, we revisit the idea of semantic segmentation: *the task of assigning each pixel of a photograph to a semantic class label* [24], then decompose this target into two relatively simple sub-tasks: pixel-level localization and classification. Instead of following previous methods to simply predict class labels, we propose to predict locations simultaneously, more precisely, to learn offsets with additional supervision.

This fresh viewpoint leads to the Location-aware Upsampling (LaU) directly, which upsamples classification results while accounting for the effect of offsets. Besides that, LaU tends to meet the key challenges in semantic segmentation: First, CNN-based segmentation methods extract high-level semantic features at the cost of losing detailed pixel-level spatial information. Second, the feature distribution of objects is imbalanced, where the interior points enjoy the rich knowledge after high-ratio upsampling yet pixels around edges suffer from indiscriminative features due to bilinear interpolation. A natural solution is to borrow the information from interior points to guide the categorization near the boundary, which exactly corresponds to our two sub-tasks: learning offsets and utilizing the information at the new position to make a prediction.

Our observation is that feature maps used by pixel-level classification, *e.g.*, the second last layer in decoders, can also be used for predicting locations. On top of these features, we construct LaUs by adding an offset prediction branch to produce offsets $(\Delta x, \Delta y)$ at each position in the high-resolution feature map (details in section 3.3). To facilitate the offset prediction of interior points and improve the convergence, we first define the candidate locations $(\frac{x}{k}, \frac{y}{k})$ then use predicted offsets to regress fine-grained coordinates, *i.e.*, $(\frac{x}{k} + \Delta x, \frac{y}{k} + \Delta y)$.

The proposed offset prediction makes it possible to further introduce location supervision implicitly/explicitly into the training process. In this paper, we present two location-aware losses forcing the offsets to predict fine-grained coordinates with the correct class. Besides, our location-aware losses avoid human labeling and produce "label" online. In the experiments, we empirically show that location-aware
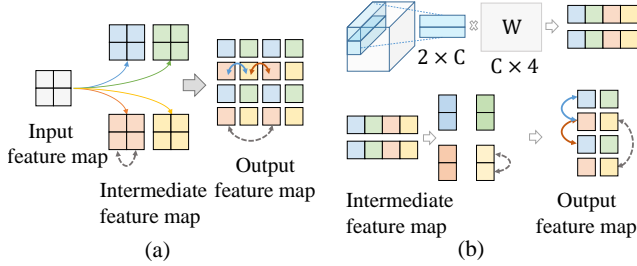
Figure 1: (a) PixelShuffle [38] is in the form of periodical shuffling of multiple intermediate feature maps generated by independent filters, which cuts off the strong spatial relationship among adjacent pixels on the low-resolution feature map (connected by gray dashed lines). (b) DUpsampling [40] replaces convolutions with a matrix product to produce a small patch, yet can be transformed into PixelShuffle. In both cases, networks are forced to learn the new spatial relationship generated by hand-crafted rearrangement.

Figure 2: The illustrations of the original convolution (a), deformable convolution [9] (b), bilinear upsampling (c) and LaU (d).

mechanism outperforms strong baseline methods, and it is feasible to apply the rich knowledge in the design of location regression to the location-aware upsampling. We believe dense predictions can widely benefit from the location-aware module, including but not limited to the semantic segmentation.

2. Related Work

Feature map upsampling has been widely used in deep models, such as encoder-decoder architectures [32, 37], generative models [35, 30], super-resolution tasks [13, 38, 20] and semantic segmentation [29, 6]. Deconvolution operations [48, 49], also known as transposed convolutions [44], first bridge the gap between low-resolution feature maps and desired full-resolution predictions. Despite its successes, recent works [33, 16] show that deconvolutions can easily suffer from the checkboard problem when the kernel size is not divisible by the stride.

Advances like PixelShuffle [38] and DUpsampling [40] turn to hand-crafted pixel rearrangement to reconstruct high-resolution features. However, the empirical design results in the inconsistent spatial relationship among adjacent pixels on the high-resolution feature map (As illustrated in Figure 1, pixels connected by blue lines are generated by different filters, yet the input of each filter corresponds to the same receptive field. Both filter and receptive filed are different for pixels connected by red lines.). Currently, PixelDCL [16] generates intermediate feature maps sequentially so that later stages depend on previously generated ones, which is similar to pixel-recurrent neural networks (Pixel-RNNs) [43] and PixelCNNs [42]. Yet pixel-wise prediction is time-consuming and there is no explicit time sequence
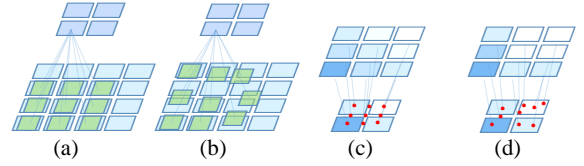
relation between adjacent pixels, *i.e.*, previously generated pixels can still rely on later produced ones which leads to a chicken-and-egg problem.

Conditional Random Field (CRF) is another group of methods, that is primarily used in [5] as a disjoint post-processing. Following works [27, 45, 28] further integrate CRF into networks to model the semantic relationships between objects. Due to the increase of computing complexity caused by integration, [3, 4] propose a Gaussian conditional random field to avoid the iterative optimization [53] while suffering from heavy gradient computation. Similar to CRFs, [2, 22] focus on refining the output label map by random walks. However, the dense matrix inversion term and the two branch model with cascaded random walks cannot be easily merged into leading encoder-decoder architectures.

Spatial Transformer Networks (STN) [21] is the first work to introduce spatial transformation capacity into neural networks, which enables global parametric transformation like 2D affine transformation. Since the pointwise location sampling in STN is weakly coupled with the input feature map at each position (only through global transformation parameters produced by the preceding feature maps), this greatly limits its capacity in pixel-wise predictions such as semantic segmentation. The module we present in this paper can be seen as a new transformation in a local and dense manner. The upsampled feature map can still enjoy the detailed information of low-resolution inputs by using pixel-wise offsets.

Deformable convolution networks (DCNN) [9, 55] achieve strong performance on object detection and semantic segmentation with trainable offsets. However, it is commonly used in feature extraction and we empirically prove that replacing the last bilinear upsampling with "DCNN+Bilinear" will cause damage to the performance (details in Table 9 and appendix section B.2 Figure 9). As illustrated in Figure 2, DCNN aims at augmenting the spatial sampling grid of filters, yet our method directly refines the sampling coordinates with location-guided losses which contribute to higher performance.
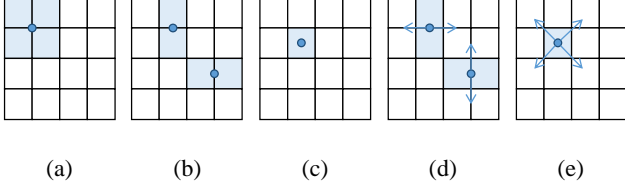
Figure 3: Different cases in 2D bilinear interpolation. (a) An ideal case that fully exploits four points to obtain the desired estimate. (b) If one of the variables (*i.e.*, x or y) happens to be an integer, bilinear interpolation degrades into a linear interpolation. (c) The special case : both x and y are integers. Bilinear interpolation is an injective non-surjective function. (d)/(e) A natural solution to the degradation in (b,c) is to slightly move the interpolation point, which leads to another problem: how to determine the offsets?

3. Methodology

Data-independent upsampling such as bilinear interpolation is one of the basic techniques in computer vision and image processing. Despite its high efficiency, this module can be oversimple (illustrated in Figure 3) and fail to capture the semantic information. In this section, we alleviate the existing problems in bilinear upsampling by introducing the location-aware mechanism.

Learning data-dependent offsets is the key step in LaU. This idea is based on the fact that adjacent boundary points on the high-resolution label map can degrade into the same pixel in the low-resolution feature map, since $8\times$ upsampling is commonly used in popular methods [7, 52, 50]. Instead of sampling from the local area, we propose to use the predicted offsets, generating the output map from the location-refined coordinates. We describe the formulation of a location-aware upsampling in the following sections.

3.1. Spatial Upsampling

To estimate the values in the output feature map $V \in \mathbb{R}^{C \times H \times W}$, a sampler applys the sampling kernel to a particular grid Ω in the input feature map $U \in \mathbb{R}^{C \times h \times w}$. Each coordinate (x_j^u, y_j^u) defines the spatial location in U and $\psi(\cdot)$ determines the kernel function at a particular pixel $V_i^{c_{\text{out}}}$ in channel c_{out}. Formally,

$$V_i^{c_{\text{out}}} = \sum_j^{hw} \psi(x_j^u; x_i^v; \Phi_x)\, \psi(y_j^u; y_i^v; \Phi_y)\, U_j^{c_{\text{in}}} \quad (1)$$

where $\forall i \in [1, ..., HW], \forall c_{\text{out}} \in [1, ..., C]$, (x_i^v, y_i^v) are the integer coordinates in the output feature map, Φ_x and Φ_y are the parameters of a generic sampling function $\psi(\cdot)$ which can be preset or data-dependent [1]. Note that the existing up-samplings such as bilinear and PixelShuffle apply the same

---

[1] Φ can be 0, 1 in $\max(0, 1 - |x_j^u - x_i^v|)$, and $\Delta x_i^v, \Delta y_i^v$ in LaU.

$\psi(\cdot)$ to every channel of the input (*i.e.*, data-independent), while LaU enables *free changes of Φ at each position*. That is, the sampling kernel in LaU changes with the input data and coordinates.

In theory, $\psi(\cdot)$ is a (sub-)differentiable function of the coordinates (x_i^v, y_i^v) and (x_j^u, y_j^u), along with the parameters Φ, which can be in any form. For example, $\psi(x_j^u; x_i^v; \Phi_x) = \max(0, 1 - |x_j^u - x_i^v|)$ corresponds to the bilinear interpolation. Here, we show that PixelShuffle is another data-independent case, which may lead to sub-optimal results.

3.1.1 PixelShuffle

PixelShuffle can also be reformulated in the form of Eq.(1). Specifically, $\psi(\cdot)$ in PixelShuffle $2\times$ upsampling is

$$\psi(x_j^u; x_i^v; \Phi_x)_{2\times} =$$
$$\begin{cases} \delta(x_i^v - 2x_j^u) & \forall x_i^v \in [0, 2, ..., W\text{-}2], c_{\text{in}} \in [1, 3, ..., C\text{-}1] \\ \delta(x_i^v - 2x_j^u - 1) & \forall x_i^v \in [1, 3, ..., W\text{-}1], c_{\text{in}} \in [2, 4, ..., C] \end{cases}$$

where $\delta(\cdot)$ is the Kronecker delta function. Note that Φ_x changes along with channel c_{in}, yet still fixed with respect to input data. From the view of Φ, this sampling kernel equates to the data-independent methods like bilinear interpolation. Furthermore, the output of Kronecker delta is either $+1$ or 0, which partly pinpoints the root of missing spatial relationship. That is, $\psi(\cdot)$ generated by PixelShuffle can be considered as a bijection. There is no direct relationship between adjacent elements in its codomain (output feature map), unless the source elements in its domain (intput feature map) are strongly correlated. For example, neighboring pixels connected by red lines in Figure 1 belong to different receptive fields, and are generated by different filters; They are likely to look different (*i.e.*, uncorrelated) unless the filter weights are similar/same. Therefore, the network has to consider the high-resolution spatial relationships when producing low-resolution intermediate feature maps.

3.1.2 Location-aware Upsampling

Mathematically, bilinear upsampling is a non-injective surjective function (not a bijection), denoted as $f : U \mapsto V$. Neighboring points in V sampled from the same grid in U are strongly correlated, since they are generated from the same elements. Compared with PixelShuffle, this scheme avoids checkerboard artifacts. As illustrated in Figure 2, location-aware upsampling is a simple yet effective improvement to the bilinear interpolation with the following sampling kernel

$$\psi(x_j^u; x_i^v; \Phi_x) = \max(0, 1 - |x_j^u - (\frac{x_i^v}{k} + \Delta x_i^v)|) \quad (2)$$
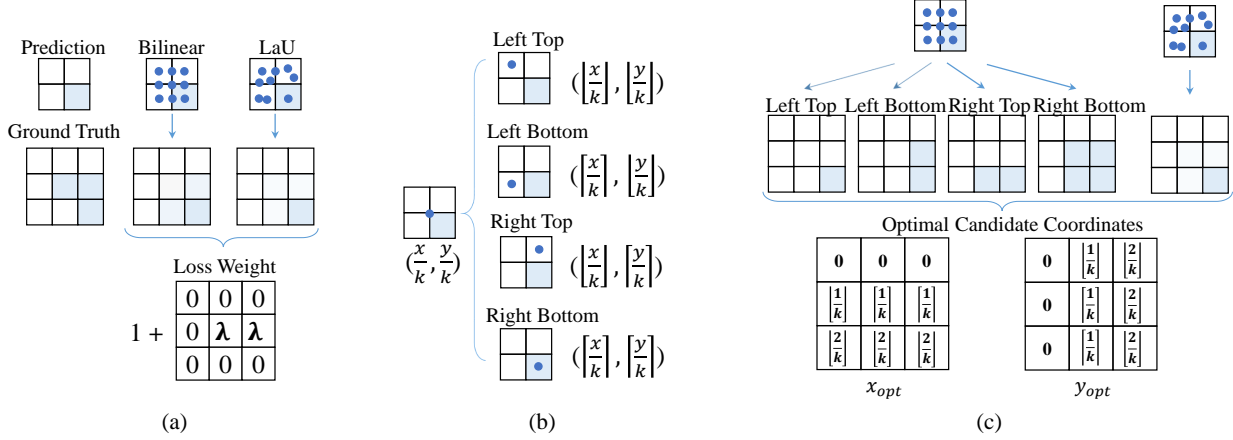
Figure 4: Illustrations of the location-aware loss. $\downarrow, \searrow, \swarrow$ refer to the interpolation and upsampling. $\bullet$ is the coordinate point for interpolating. $\lceil \cdot \rceil, \lfloor \cdot \rfloor$ rounds input upward and downward, respectively. (a) Offset-guided loss will punish the moved pixel with loss weight λ if it yields even larger loss than the original point. (b) The illustrations of different sampling kernels $\psi_{\text{left top}}, \psi_{\text{left bottom}}, \psi_{\text{right top}}$ and $\psi_{\text{right bottom}}$. (c) The optimal candidate coordinates correspond to pixels with the smallest loss among LaU, left top, left bottom, right top and right bottom upsamplings. Ground-truth is the same as subfigure (a).

where k is the upsampling ratio. Inspired by the anchor-based methods in object detection [17, 36], we introduce $\frac{x_i^v}{k}$ as a candidate location then regress the detailed location with predicted offsets. To encourage the network to jump out of the local grid, we add no constraint on $(\Delta x_i^v, \Delta y_i^v)$.

To learn the offsets in the upsampling module, we utlize the gradients with respect to U and $(\Delta x_i^v, \Delta y_i^v)$. This allows the backpropagation of the loss function through the (sub-)differentiable location-aware upsampling module. For upsampling kernel (2), the partial derivatives are

$$\frac{\partial V_i^{c_{\text{out}}}}{\partial U_j^{c_{\text{in}}}} = \sum_j^{hw} \max(0, 1 - |x_j^u - (\frac{x_i^v}{k} + \Delta x_i^v)|) \cdot$$

$$\max(0, 1 - |y_j^u - (\frac{y_i^v}{k} + \Delta y_i^v)|)$$

$$\frac{\partial V_i^{c_{\text{out}}}}{\partial \Delta x_i^v} = \sum_j^{hw} U_j^{c_{\text{in}}} \max(0, 1 - |y_j^u - (\frac{y_i^v}{k} + \Delta y_i^v)|) \cdot$$

$$\begin{cases} 0 & \text{if } |x_j^u - (\frac{x_i^v}{k} + \Delta x_i^v)| \geq 1 \\ -1 & \text{if } \Delta x_i^v < x_j^u - \frac{x_i^v}{k} \\ 1 & \text{if } \Delta x_i^v \geq x_j^u - \frac{x_i^v}{k} \end{cases}.$$

With the gradients to offsets, we can further define the offset prediction branch to allow the end-to-end training.

3.2. Location-aware Loss

To fully exploit the effect of predicted offsets $(\Delta x_i^v, \Delta y_i^v)$, we introduce (semi-)supervised location-aware losses into the training process. Since offsets could point to any input location, as long as it contains the correct category, we first consider the offset-guided loss without

hard constraints. Inspired by the bounding box regression, we further propose the offset regression loss to guide the offset prediction directly.

3.2.1 Offset-guided Loss

In common training process of semantic segmentation, the network is optimized by per-pixel cross-entropy loss $\ell(\cdot)$. Instead of learning from isolated per-pixel supervision, we consider the loss *pair* generated by bilinear upsmapling and LaU (shown in Figure 4a). Formally, we introduce the auxiliary loss L' (with no gradient) and the target loss L,

$$L = \ell(x, w, \psi_{\text{LaU}}(\Phi_x, \Phi_y)), \quad L' = \ell(x, w, \psi_{\text{bilinear}})$$

where x is the input image, w is the model parameters and Φ refers to the predicted offsets. The per-pixel offset-guided loss

$$L_i^{\text{off}} = L_i \Lambda_i = L_i \cdot \begin{cases} 1 & \text{if } L_i < L_i' \\ 1 + \lambda & \text{otherwise} \end{cases} \quad (3)$$

with loss weight Λ_i forces the network to "take a step", since the trivial solution $\Delta x, \Delta y = 0$ (i.e., $L_i = L_i'$) will be punished by the extra cost λ. Besides, $L_i < L_i'$ demonstrates that the moved pixel should have a smaller cross-entropy loss (i.e., a higher confidence score) than the original point. Therefore, (3) encourages the pixel to move towards a better position, which contains potentially correct class.

3.2.2 Regression Loss

Since there is no groud-truth label of the offsets, we could intuitively search for the closest well classified point among

the neighbouring pixels to generate the offsets. However, the brute search is time-consuming for online label generation. Hence, we emprically constrain the search space to the intergral coordinate points (shown in Figure 4b).

Following the preivous subsection, we further introduce the auxiliary losses L^{lt}, L^{lb}, L^{rt} and L^{rb}, *e.g.*, $L^{\text{rt}} = \ell(x, w, \psi_{\text{right top}})$ with the sampling kernels

$$\psi_{\text{right top}}^x = 1\{x_j^u = \left\lfloor \frac{x_i^v}{k} \right\rfloor\}, \quad \psi_{\text{right top}}^y = 1\{y_j^u = \left\lceil \frac{y_i^v}{k} \right\rceil\}$$

where $1\{\cdot\}$ is the "indicator function", that is $1\{\text{true condition}\} = 1$ and $1\{\text{false condition}\} = 0$. Then, we define the loss set

$$\Omega = \{L, L^{\text{lt}}, L^{\text{lb}}, L^{\text{rt}}, L^{\text{rb}}\}$$

and the coordinate set

$$\Theta = \left\{ (\frac{x}{k} + \Delta x, \frac{y}{k} + \Delta y), (\left\lfloor \frac{x}{k} \right\rfloor, \left\lfloor \frac{y}{k} \right\rfloor), \right.$$
$$\left. (\left\lceil \frac{x}{k} \right\rceil, \left\lfloor \frac{y}{k} \right\rfloor), (\left\lfloor \frac{x}{k} \right\rfloor, \left\lceil \frac{y}{k} \right\rceil), (\left\lceil \frac{x}{k} \right\rceil, \left\lceil \frac{y}{k} \right\rceil) \right\}.$$

Combining Ω and Θ, we obtain the optimal candidate cordinates

$$\Theta^{\text{opt}} = \sum_k 1\{\min(\Omega) = \Omega_k\} \cdot \Theta_k$$

where the summation stops at $1\{\min(\Omega) = \Omega_k\} = 1$. Given Θ^{opt} (illustrated in Figure 4c), we formulate the per-pixel offset regression loss as

$$L_i^{\text{reg}} = \gamma \text{SmoothL}_1\left(\Theta_i^{\text{opt}}, (\frac{x_i^v}{k} + \Delta x_i^v, \frac{y_i^v}{k} + \Delta y_i^v)\right) + L_i \Lambda_i$$
$$= \gamma \text{SmoothL}_1 + L_i \cdot \begin{cases} 1 & \text{if } L_i \leq \forall \Omega_i \\ 1 + \lambda & \text{otherwise} \end{cases}.$$
$$(4)$$

where γ controls the strength of the regression loss. In our experiments, we set γ to 0.1 without fine-tuning. Note that the auxiliary losses will be excluded from the backward propagation (with no gradient) and omitted at inference time.

3.3. Location-aware Network

The combination of predicted offsets and interpolation operation forms a location-aware upsampling (Figure 5). Note that the offset prediction branch can take any form in practice, such as a fully-connected network like RPN in Faster-RCNN [36]. In this paper, we use "Conv$_{1\times1}$+LReLU+Conv$_{3\times3}$+PixelShuffle" to illustrate the basic framework of location-aware mechanism in semantic segmentation.

The first 1×1 convolution layer reduces the channel number of the input feature map $U \in \mathbb{R}^{C \times h \times w}$ from C
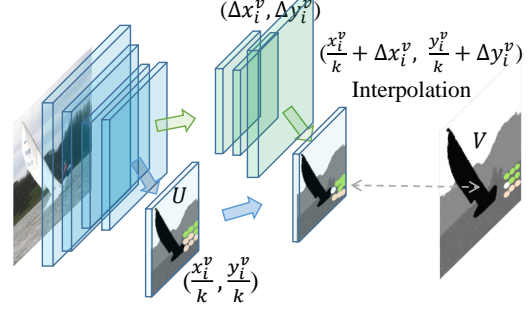

Figure 5: The framework overview of LaU. Our approach employs the same encoder-decoder architecture as the original method, which adds a 1×1 convolution layer over the decoder output to generate per-pixel classification scores (*i.e.*, $U \in \mathbb{R}^{C \times h \times w}$). The offset prediction branch (green blocks) shares the input feature map with the 1×1 convolution then produces $(\Delta x_i^v, \Delta y_i^v)$. A location-aware upsampling merges two branches and generates high-resolution outputs.

to C'. Then, the last convolution layer produces an offset tensor $O \in \mathbb{R}^{2M \times H \times W}$ where $H = kh$ and $W = kw$. M can be the number of class to allow the per-element shift. However, the extra computing cost will be unacceptable if M is too large. When we set M to 1, the output feature map shares offset across different channels. Compared with $M = 150$, this setting saves over 19G FLOPs on ADE20K dataset.

Encoder-decoder architecture with LaU tends to first predict the general location and rough boundary, then recontruct the detailed information using predicted offsets. As shown in Figure 7c and 7d, this two-step scheme decomposes the original problem into two sub-tasks: pixel-level localization and classification, which correspond to the two branches in location-aware network.

4. Experiments

We evaluate the proposed method on the PASCAL VOC 2012 semantic segmentation benchmark [14], PASCAL Context [31] and ADE20K [54] datasets. The standard evaluation metrics are pixel accuracy (pixAcc) and mean Intersection of Union (mIoU). In this section, we first introduce datasets used in this paper as well as the implementation details. We then apply LaU to the top of networks and the upsampler in decoders to show the effectiveness of the location-aware mechanism. Moreover, the visualization results demonstrate the superiority of optimizing the proposed location-aware segmentation (shown in Figure 6).
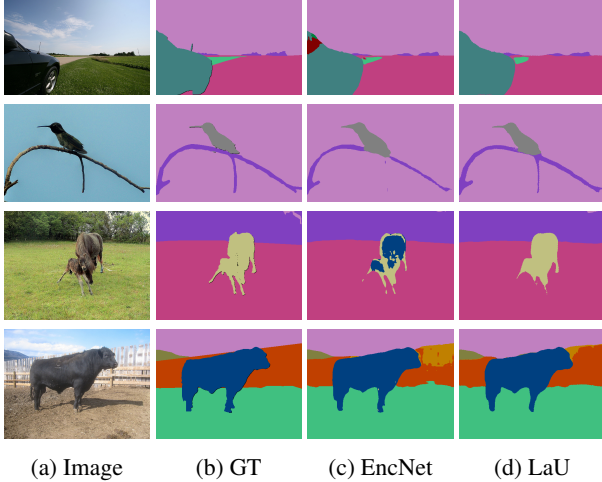
| (a) Image | (b) GT | (c) EncNet | (d) LaU |

Figure 6: Visualization results of different upsampling modules using EncNet-ResNet50 [50]. "EncNet" refers to the original bilinear upsampling. "LaU" is $4\times$ location-aware upsampling with offset-guided loss.

Table 1: Performance on Pascal Context *val* set with EncNet-ResNet50 (60 classes) using different upsampling ratios. "LaU$_{off}$" refers to LaU with offset-guided loss. We cover the reamining upsampling factor of LaU with bilinear upsampling.

Upsampling	Ratio	mIoU%	Params	FLOPs
Bilinear ([50])	$8\times$	49.08	41.82M	150.69G
LaU$_{off}$	$2\times$	49.61	+0.14M	+0.51G
	$3\times$	49.67	+0.15M	+0.55G
	$4\times$	**50.27**	+0.16M	+0.60G
	$5\times$	49.65	+0.18M	+0.68G

Table 2: Ablation study on loss weight λ using EncNet-ResNet50+LaU$_{off_{4\times}}$ (Pascal Context *val* 60 classes).

λ	0.0	0.1	0.2	0.3	0.4
mIoU%	49.77	49.84	49.94	**50.27**	49.98
pixAcc%	78.69	78.77	78.75	**79.01**	78.89

4.1. Experimental Settings

Dataset PASCAL Context dataset [31] provides pixel-wise semantic annotations for whole scenes, which contains 4998 images for training and 5105 images for testing. Following [26, 15, 46, 50], we use the most frequent 59 object categories plus background (60 classes in total) as the groundtruth labels. PASCAL VOC 2012 consists of 1464 (*train*), 1449 (*val*) and 1456 (*test*) pixel-level annotated images, which involves 20 foreground object classes and one background class. Following previous works [8, 18], we augment the dataset by the extra annotations, resulting in 10582 (*trainaug*) training images. The mIoU is averaged across 21 classes. ADE20K [54] scene parsing benchmark consists of 150 categories. There are 20K images for training, 2K for validation and 3K for testing. The results on test set are provided by the ADE20K online evaluation server.

Implementation Details For the experiments on PAS-CAL Context, we adopt the same settings as described in [50, 46]. The learning rate starts at 0.001, which gradually decrease to 0 using the "poly" strategy (power=0.9). The networks are trained for 80 epochs with SGD. Concretely, the momentum is set to 0.9 and weight decay is 0.0001. To encourage the LaU to "look wider", we set the weight decay of the convolutional layer in LaU to 0. For data augmentation, we randomly scale (from 0.5 to 2.0) and horizontally flip the input images, then apply a 480×480 random crop to the images. All the experiments are conducted on 4 G-PUs with batch size 16. Since ImageNet pre-trained backbones can significantly contribute to higher performances, we follow the practice of [50, 46, 15, 40] to use the stan-

dard ResNet-50 and ResNet-101 [19].

For training on PASCAL VOC, we follow the setting in [40, 8]. The initial learning rate is set to 0.007 with "poly" policy and weight decay is 0.0001. The total iteration is $30k$ for ResNet-50 experiments with batch size 48, yet another $30k$ iterations are required for training ResNet-101 with initial learning rate being 0.001 [8]. The scaling with flipping data augmentation and LaU settings remain the same as PASCAL Context.

For the experiments on ADE20K, as described in EncNet [50], we train our model on the training set for 180 epochs with learning rate 0.004 and evaluate it on the validation set using multi-scale testing. We then fine-tune our models on the *train-val* set for another 30 epochs with learning rate $4e^{-4}$ and submit results to the test server. We use a 520×520 base size with 480×480 crop for training. The backbone networks are the same as previous experiments.

Unless specified, we employ the same loss function as described in [7, 50] plus a location-aware loss and report the single-scale results, *without any bells and whistles*.

Ablation Study for LaU To evaluate LaU, the key parameter is the upsampling factor. As listed in Table 1, $4\times$ upsampling works better than other settings with negligible additional time and space complexity. With proposed LaU, the results consistently outperform standard bilinear upsampling and the best setting exceeds baseline by +1.2 mIoU with only 0.16M extra parameters. Since loss weight of the original auxiliary loss in EncNet is 0.2, in this section, we intuitively set λ to 0.3 to enlarge the gradient of location-aware loss.

Table 3: Semantic segmentation results on Pascal Context *val* set. mIoU on 60 classes w/ background.

(a) Applying to encoder-decoder methods with ResNet-50. "OS" refers to the output stride. We first apply an upsampling with LaU/PixelShuffle, then cover the remaining upsampling factor with bilinear upsampling. "$2\times +2\times$" means the cascade of two $2\times$, *i.e.*, $4\times$.

Head	Upsampling	OS	mIoU%
	PixelShuffle$_{2\times}$	8	43.4
	PixelShuffle$_{3\times}$	8	38.3
EncNet [50]	PixelShuffle$_{2\times+2\times}$	8	27.9
	Bilinear	8	49.1
	LaU$_{off_{4\times}}$	8	**50.3**
	PixelShuffle$_{2\times}$	32	37.8
EncNet [50]	Bilinear	32	45.7
	LaU$_{off_{4\times}}$	32	**46.7**
	PixelShuffle$_{2\times}$	8	43.3
ASPP [7]	Bilinear	8	48.4
	LaU$_{off_{4\times}}$	8	**49.8**

(b) Results with state-of-the-art methods. "off$_{4\times}$" and "reg$_{4\times}$" are offset-guided loss and offset regression loss, respectively. "MS" means multi-scale testing.

Method	Backbone	mIoU%	MS
CCL [11]	ResNet-101	50.7	
DUpsampling [40]	Xception-65	51.4	✓
DUpsampling [40]	Xception-71	52.5	✓
DANet [15]	ResNet-101	52.6	✓
HRNet [39]	HRNetV2-W48	53.1	✓
JPU [46]	ResNet-101	53.1	✓
BFP [10]	ResNet-101	52.7	✓
EncNet [50]	ResNet-101	51.7	✓
EncNet + LaU$_{off_{4\times}}$	ResNet-101	**53.9**	✓
EncNet + LaU$_{off_{4\times}}$	ResNet-101	52.7	
EncNet + LaU$_{reg_{4\times}}$	ResNet-101	**53.9**	✓
EncNet + LaU$_{reg_{4\times}}$	ResNet-101	52.8	

Ablation Study for Location-aware Loss We experiment with loss weight λ between 0 and 0.4 and show the results in Table 2. Note that LaU without additional supervision ($\lambda = 0$) still produces performance gain and $\lambda = 0.3$ yields the best performance. Both location-aware upsampling and location-aware loss contribute to higher performance than bilinear baseline (similar results are reported in Table 9). Due to the limited computing resources, we directly apply the best settings to regression loss experiments and emprically set γ to 0.1 without grid search.

4.2. Applying to Label Prediction

In this section, we apply LaU to the top of the decoders (as shown in Figure 5), which corresponds to the prediction of label maps.

4.2.1 Quantitative Analysis

PASCAL Context To estimate the generalization ability of LaU, we conduct experiments on two popular methods Encoding [50] and ASPP [7] with different output strides. Table 3a shows that our methods consistently outperform the original bilinear methods. Following the setting in super-resolution tasks [27, 20, 38], one may decompose a single $8\times$ upsampling into three cascaded $2\times$ upsamplings. To simplify the experimental settings, we leave this scheme as future work.

Compared with PixelShuffle [38] (Table 3a) and Deformable [9] bilinear upsampling (Table 9), LaU notably outperforms those methods with/without location-aware loss. Note that PixelShuffle and Deformable cause damages to the performance; We show the visualization results and give a brief discussion in the appendix section 2.

Table 4: Comparison with Deformable Convolution Netwrok [9] on Pascal Context *val* set. We add an "LaU/Deformable+Bilinear" module to the top of EncNet/ASPP. "LaU$_{4\times}$" refers to LaU without location-aware loss. "OS" is the output stride.

Method	Upsampling	OS	mIoU%
	Deformable+Bilinear	32	44.7
	LaU$_{4\times}$	32	46.2
EncNet [50]	LaU$_{off_{4\times}}$	32	**46.7**
	Deformable+Bilinear	8	48.0
	LaU$_{4\times}$	8	49.8
	LaU$_{off_{4\times}}$	8	**50.3**
	Deformable+Bilinear	8	47.9
ASPP [7]	LaU$_{4\times}$	8	49.4
	LaU$_{off_{4\times}}$	8	**49.8**

Table 3b further shows that single-scale LaUs surpass recent upsampling method DUpsampling [40] with strong Xception-71 backbone, *e.g.*, ImageNet Top-1 accuracy of Xception-65 is $+2.21\%$ higher than ResNet-101. Our multi-scale result achieves $+2.2$ mIoU over EncNet, while LaU is still compatible with attention mechanisms and stronger encoder-decoder architectures to further enhance the leading performances.

ADE20K As shown in Table 5, our methods achieve similar results on ADE20K dataset. With ResNet-50 backbone, LaUs outperform EncNet baseline by $+1.8$ and $+1.3$ mIoU. Regression loss seems to perform worse than location-guided loss. We attribute this to the hand-crafted design of optimal candidate coordinates and the selection of γ. The

Table 5: ADE20K *val* set results with top-performing models (using mutli-scale testing) .

Method	Backbone	pixAcc%	mIoU%
EncNet [50]	ResNet-50	79.73	41.11
EncNet+LaU$_{off_{4\times}}$	ResNet-50	**80.53**	**42.78**
EncNet+LaU$_{reg_{4\times}}$	ResNet-50	80.28	42.41
HRNet [39]	HRNetV2-W48	–	43.20
PSPNet [52]	ResNet-152	81.38	43.51
EncNet [50]	ResNet-101	81.69	44.65
PSPNet [52]	ResNet-269	**81.69**	44.94
EncNet+LaU$_{off_{4\times}}$	ResNet-101	81.21	44.55
EncNet+LaU$_{reg_{4\times}}$	ResNet-101	81.49	**45.02**

Table 6: Results on ADE20K *test* set. Best entries in COCO-Place challenge 2017 are listed.

Team	Model	Final Score ↑	mIoU%
PSPNet [52]	ResNet-269	0.5538	–
WinterISComing[2nd]	–	0.5544	–
CASIA_IVA_JD[1st]	–	0.5547	–
EncNet [50]	ResNet-101	0.5567	–
JPU [46]	ResNet-101	0.5584	38.39
EncNet+LaU$_{off_{4\times}}$	ResNet-101	**0.5641**	39.04
EncNet+LaU$_{reg_{4\times}}$	ResNet-101	0.5632	**39.14**

current search space contains only four points, which can be suboptimal. We will include a larger search space in our future work.

Although our methods achieve slightly worse performance than EncNet on *val* set, LaUs notably surpass EncNet and all entries in COCO-Place challenge 2017 on the final *test* score (see results in Table 6).

4.2.2 Qualitative Analysis

We show the PASCAL-Context visual results in Figure 6. LaU labels out the missing parts in EncNet and corrects the misclassifications. More examples in ADE20K and the visual comparison between offset-guided loss and regression loss are shown in the appendix (Figure 11). To further illustrate the effect of location-aware mechanism, we compare the segmentation results before and after LaU in Figure 7. LaU first produces coarse segmentation of objects with rough boundary (in Figure 7c), then refines the boundary using offsets (in Figure 7d). The whole segmentation process of LaU is from coarse to fine.

4.3. Applying to Decoder

Recent state-of-the-arts [7, 8] utilize upsampling modules in decoders to combine the high-resolution features with low-resolution ones. To show the general applicability
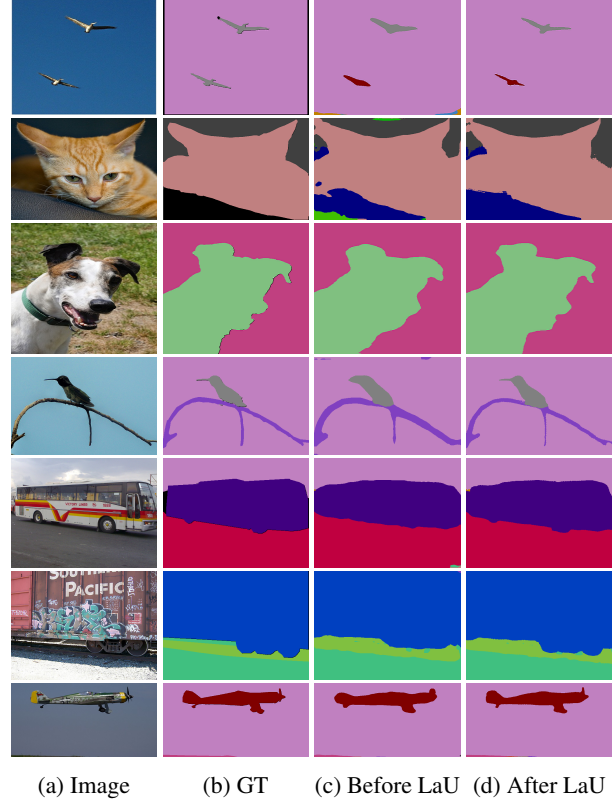

(a) Image   (b) GT   (c) Before LaU   (d) After LaU

Figure 7: Illustrations of the effect of predicted offsets generated by LaU$_{off_{4\times}}$. We visualize the segmentation results of EncNet+LaU before and after LaU.

Table 7: mIoU over Pascal VOC 2012 *val* set using DeepLabv3+ [8].

Method	Upsampling	Backbone	mIoU%
DeepLabv3+	Bilinear	ResNet-50	76.52
	LaU$_{4\times}$	ResNet-50	**78.55**
	Bilinear	ResNet-101	78.85
	LaU$_{4\times}$	ResNet-101	**80.16**

of proposed method, we replace the bilinear upsampling in DeepLabv3+ [8] with LaU. The performance of ResNet-50+LaU (+2.03% mIoU in Table 7) has approached the original ResNet-101 backbone results. Even with strong ResNet-101 baseline, LaU can still boost the performance by one point.

5. Conclusion

In this paper, we provide a new perspective for optimizing the semantic segmentation problem. We decompose the original pixel-level classification problem into offsets prediction and classification, which introduces the idea of location prediction into semantic segmentation. Based on this

fresh viewpoint, we propose the location-aware upsampling and location-aware losses. Our models achieve promising performance compared with various baseline methods. The effectiveness of learning offsets is also verified through qualitative and quantitative results.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495, 2017. 1

[2] Gedas Bertasius, Lorenzo Torresani, Stella X. Yu, and Jianbo Shi. Convolutional random walk networks for semantic image segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6137–6145, 2017. 1, 2

[3] Siddhartha Chandra and Iasonas Kokkinos. Fast, exact and multi-scale inference for semantic image segmentation with deep gaussian crfs. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VII*, pages 402–418, 2016. 2

[4] Siddhartha Chandra, Nicolas Usunier, and Iasonas Kokkinos. Dense and low-rank gaussian crfs using deep embeddings. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5113–5122, 2017. 2

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 2

[6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848, 2018. 1, 2

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017. 1, 3, 6, 7, 8, 11, 12

[8] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, pages 833–851, 2018. 1, 6, 8

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 764–773, 2017. 2, 7, 12

[10] Henghui Ding, Xudong Jiang, Ai Qun Liu, Nadia Magnenat Thalmann, and Gang Wang. Boundary-aware feature propagation for scene segmentation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 7

[11] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2393–2402, 2018. 7

[12] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(2):295–307, 2016. 1

[13] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, pages 391–407, 2016. 2

[14] Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 5

[15] Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *2019 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-21, 2019*, pages 1–8, 2019. 6, 7

[16] H. Gao, H. Yuan, Z. Wang, and S. Ji. Pixel transposed convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. 2

[17] Ross B. Girshick. Fast R-CNN. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1440–1448, 2015. 4

[18] Bharath Hariharan, Pablo Arbelaez, Lubomir D. Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 991–998, 2011. 6

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. 1, 6

[20] Xiangyu He, Zitao Mo, Peisong Wang, Yang Liu, Mingyuan Yang, and Jian Cheng. Ode-inspired network design for single image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1732–1741, 2019. 2, 7, 11

[21] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2017–2025, 2015. 2

[22] Peng Jiang, Fanglin Gu, Yunhai Wang, Changhe Tu, and Baoquan Chen. Difnet: Semantic segmentation by diffusion networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 1637–1646, 2018. 2

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012. 1

[24] Victor S. Lempitsky, Andrea Vedaldi, and Andrew Zisserman. Pylon model for semantic segmentation. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 1485–1493, 2011. 1

[25] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 11

[26] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5168–5177, 2017. 6

[27] Guosheng Lin, Chunhua Shen, Anton van den Hengel, and Ian D. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3194–3203, 2016. 1, 2, 7

[28] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1377–1385, 2015. 2

[29] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3431–3440, 2015. 1, 2

[30] Alireza Makhzani and Brendan J. Frey. Winner-take-all autoencoders. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2791–2799, 2015. 2

[31] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan L. Yuille. The role of context for object detection and semantic segmentation in the wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 891–898, 2014. 5, 6

[32] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1520–1528, 2015. 2

[33] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 2

[34] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. 11

[35] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 2

[36] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015. 1, 4, 5

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, pages 234–241, 2015. 1, 2

[38] Wenzhe Shi, Jose Caballero, Ferenc Huszar, 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 *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1874–1883, 2016. 2, 7, 11

[39] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *CoRR*, abs/1904.04514, 2019. 7, 8

[40] Zhi Tian, Tong He, Chunhua Shen, and Youliang Yan. Decoders matter for semantic segmentation: Data-dependent decoding enables flexible feature aggregation. In *2019 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-21, 2019*, pages 1–8, 2019. 2, 6, 7

[41] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 1653–1660, 2014. 1

[42] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4790–4798, 2016. 2

[43] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of the 33nd International Conference on Machine*

Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016, pages 1747–1756, 2016. 2

[44] Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for MATLAB. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM '15, Brisbane, Australia, October 26 - 30, 2015*, pages 689–692, 2015. 2

[45] Raviteja Vemulapalli, Oncel Tuzel, Ming-Yu Liu, and Rama Chellappa. Gaussian conditional random field network for semantic segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3224–3233, 2016. 1, 2

[46] Huikai Wu, Junge Zhang, Kaiqi Huang, Kongming Liang, and Yizhou Yu. Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation. In *2019 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-21, 2019*, pages 1–8, 2019. 6, 7, 8

[47] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 1

[48] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Robert Fergus. Deconvolutional networks. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 2528–2535, 2010. 2

[49] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 2018–2025, 2011. 2

[50] Hang Zhang, Kristin J. Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7151–7160, 2018. 1, 3, 6, 7, 8, 11, 12

[51] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6230–6239, 2017. 1

[52] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6230–6239, 2017. 3, 8

[53] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1529–1537, 2015. 2

[54] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5122–5130, 2017. 5, 6

[55] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets V2: more deformable, better results. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9308–9316, 2019. 2, 11

A. Run-time Speed

In this section, we employ the frame per second (FPS) as the evaluation metirc to show the time efficiency about the proposed method. Compared with the traditional bilinear upsampling, LaU consistently improves the performance at the cost of negligible extra time complexity. We report the FPS averaged over 100 iterations. As shown in Table 8, LaU is competitive with the bilinear upsamling network at actual run-time speed.

Table 8: Actual inference time on a GTX 1080Ti GPU with 512×512 input size. "OS" refers to the output stride.

Head	OS	Backbone	Upsampling	FPS
EncNet [50]	8	ResNet-50	Bilinear	18.60
	8	ResNet-50	LaU$_{4\times}$	18.39
	32	ResNet-50	Bilinear	70.80
	32	ResNet-50	LaU$_{4\times}$	69.84
	8	ResNet-101	Bilinear	11.66
	8	ResNet-101	LaU$_{4\times}$	11.49
	32	ResNet-101	Bilinear	47.37
	32	ResNet-101	LaU$_{4\times}$	46.46
ASPP [7]	8	ResNet-50	Bilinear	15.93
	8	ResNet-50	LaU$_{4\times}$	15.72
	8	ResNet-101	Bilinear	10.48
	8	ResNet-101	LaU$_{4\times}$	10.32

B. Implementation Details

Our experiments base on the PyTorch framework [34] and pre-trained models provided by EncNet [50]. We use the official open source code [55] to conduct the deformable convolution experiments. In this section, we employs ResNet-50 as the backbone without multi-scale testing. The results are reported on PASCAL Context *val* set. To make a fair comparison, we follow the same setting in our main paper.

B.1. PixelShuffle

We assume that the class number is 59. Following the popular setting in super-resolutions [25, 38, 20], the archi-
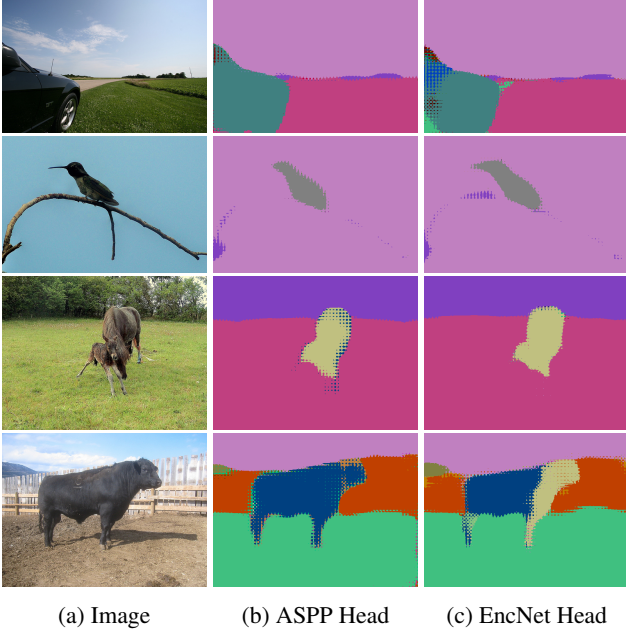
(a) Image   (b) ASPP Head   (c) EncNet Head

Figure 8: PixelShuffle$_{2\times}$ segmentation results. Figure (b) and (c) show the checkerboard artifacts. Although the last convolution layer (i.e., $59C_3$) may smooth the outputs, the results still consist of strange checkerboard pattern of artifacts.

tecture of PixelShuffle-based upsamplings are as follow :

$2_\times$: $236C_3 + \text{PixelShuffle}_{2\times} + 59C_3$

$3_\times$: $531C_3 + \text{PixelShuffle}_{3\times} + 59C_3$

$4_\times$: $236C_3 + \text{PixelShuffle}_{2\times} + 236C_3 + \text{PixelShuffle}_{2\times} + 59C_3$

where $236C_3$ refers to the 3×3 convolution layer with 236 filters.

The intuitive idea is that, in the worst case, pixelshuffle should degrade into nearest neighbor upsampling (same feature map and weight for each shuffle group). However, the experiment results in Table 3a demostrates that PixelShuffle damages the performance. In this subsection, we show the segmentation results to reveal checkboard problems in PixelShuffle. As illustrated in Figure 8, both EncNet [50] head and ASPP [7] head suffer from the checkboard artifacts.

B.2. Deformable Convolution

We replace the traditional bilinear upsamling in encoder-decoder networks with "Deformable 3×3 Convolution [9] + Blinear$_{8\times}$" module. Since "Blinear + Deformable Convolution" will hughly increase the extra computing cost, we only conduct experiments under the first setting.

Although deformable convolution has been proved to improve the segmentation task when properly used in feature extraction, we show that the "deformable upsampling"

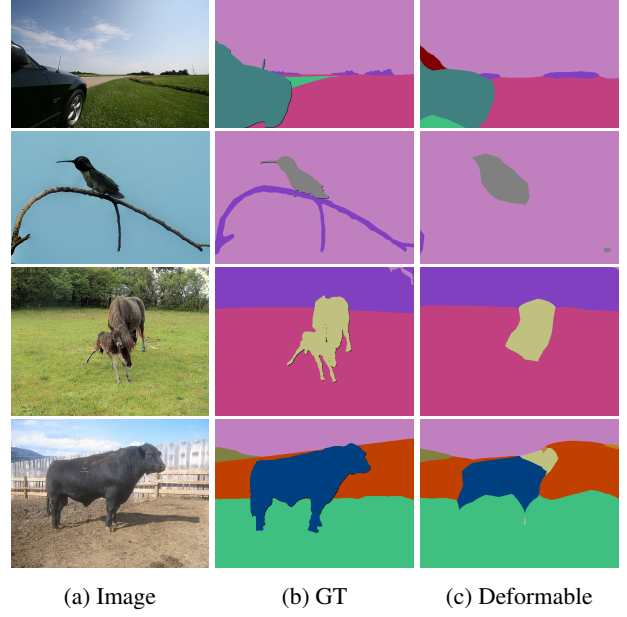(a) Image   (b) GT   (c) Deformable

Figure 9: Visualization results using deformable bilinear upsampling with EncNet-ResNet50 [50].

is probably not appropriate for the final prediction. As illustrated in Figure 9, deformable bilinear upsampling tends to focus on the main body of the objects yet miss the details, which consists with the design of deformable convolution networks.

C. Visualization Results

In this section, we show the visual results from ADE20K dataset. For the qualitative analysis, we compare the outputs generated by LaU and bilinear upsampling, shown in Figure 10. Since offset-guided loss and regression loss achieve similar mIoU and pixel accuracy, we visualize the segmentation results on *val* set in Figure 11 and make a qualitative evaluation. Regression loss performs slightly better than offset-guided loss, which is consistent with the mIoU results.
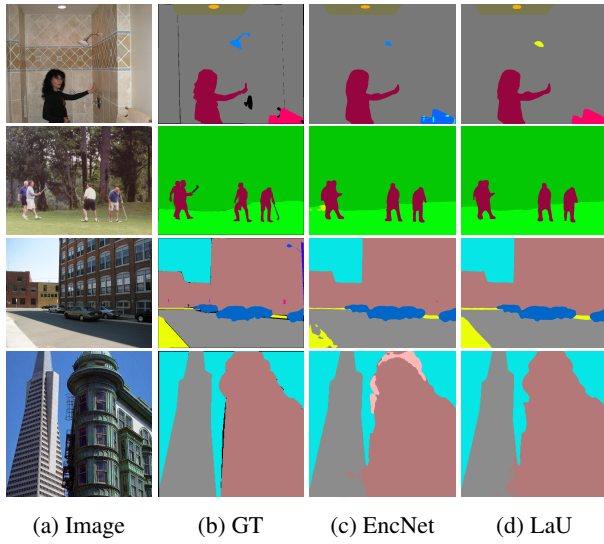
| (a) Image | (b) GT | (c) EncNet | (d) LaU |

Figure 10: Visual improvements on ADE20K with ResNet101 backbone. "EncNet" is the standard bilinear up-sampling. "LaU" refers to $\text{LaU}_{\text{off}_{4\times}}$.

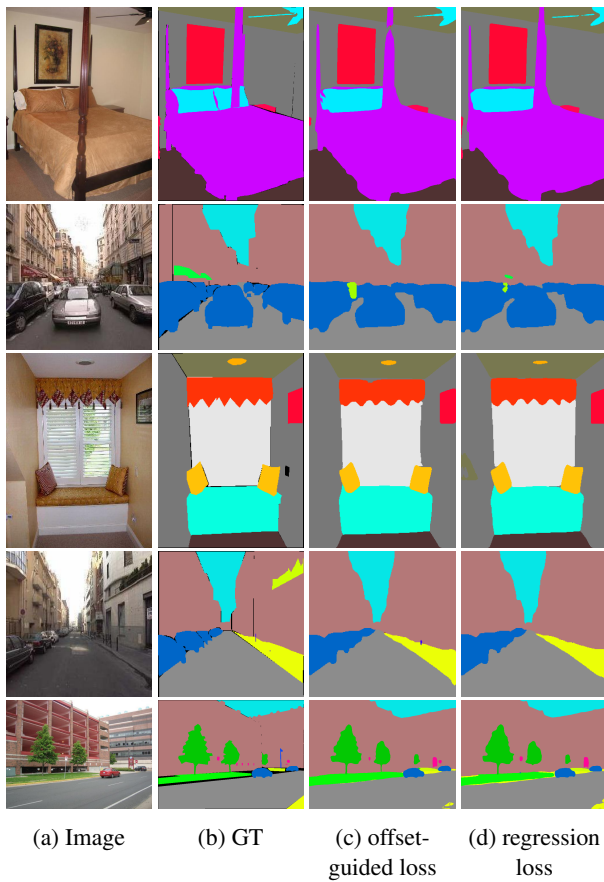| (a) Image | (b) GT | (c) offset-guided loss | (d) regression loss |

Figure 11: Visual comparison on ADE20K *val* set.