Learning Pixel-level Semantic Affinity with Image-level Supervision for Weakly Supervised Semantic Segmentation

(2018)

Jiwoon Ahn, Suha Kwak

Summary

Contributions

Given that pixel-level labels are hard to acquire and time consuming. Weakly-supervised segmentation lends itself as a possible alternative where only weak and easily obtainable labels are used, such as image-level labels to generate pixel-level labels and train the segmentation network. The previous approaches use the class to generate the class activation maps, however, the localization maps often only focus on the most discriminative parts of the objects. To overcome this, the authors propose to learn affinity maps so that similar pixel have similar affinities, the affinities can then be used to construct a transition matrix, and starting from CAMs, they can propagate the classes using the matrix and create more precise masks.

Method

The proposed method can be divided into three parts. The first step is to generate CAMs and use them to extract training labels for training the affinity net. A trained affinity net can then be used to generate transition matrices and propagate the CAMs using random walks from precise pseudo pixel-level labels. The final step is to train a segmentation network on the pseudo labels.

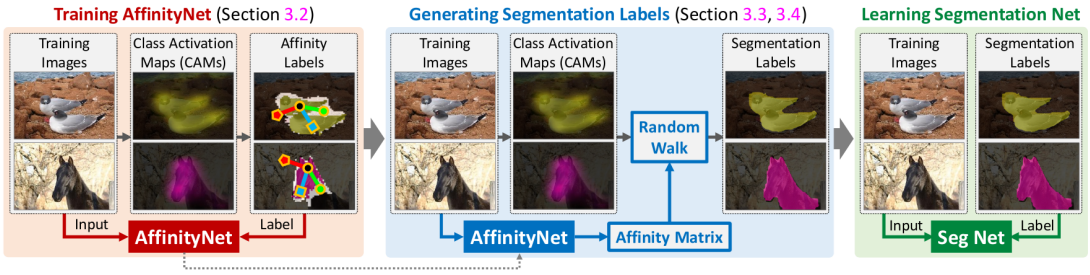


Figure 1. Illustration of our approach. Salient areas for object classes and background are first localized in training images by CAMs [40] (Section 3.1). From the salient regions, we sample pairs of adjacent coordinates and assign binary labels to them according to their class consistency. The labeled pairs are then used to train AffinityNet (Section 3.2). The trained AffinityNet in turn predicts semantic affinities within local image areas, which are incorporated with random walk to revise the CAMs (Section 3.3) and generate their segmentation labels (Section 3.4). Finally, the generated annotations are employed as supervision to train a semantic segmentation model.

CAMs and affinity labels The first step is to use a trained classification network (train using the available image-level examples) to extract the CAMs. The CAMs are in the form of $C \times H \times W$, with C as the number of classes without the bg. The objective is to create a pixel level mask of size $1 \times H \times W$, where element is the class assigned to the given pixel, this time with two additional classes: bg and ignore (for the pixel where we can say for certain which class they belong only based on CAMs). Using CAMs M_c, we can add the attention scores for the bg as follows:

$$M_{\text{bg}}(x,y) = \left\{1 - \max_{c \in C} M_c(x,y)\right\}^{\alpha}$$

By adjusting alpha (4/32), we can create two masks, with small alpha, we assign the back ground class to majority of the pixels, and only the confident fg pixels remain. Inversely, to only have the confident bg pixel, we increase alpha. The two masks can then be merged to only have the confident fg and bg pixels. The remaining pixel are assigned an ignore class label.

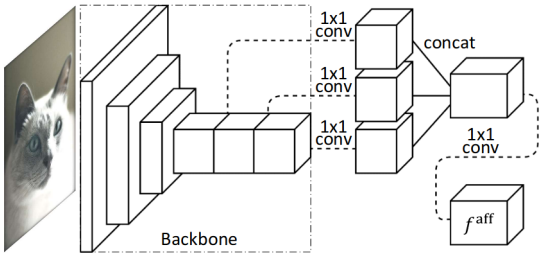


Figure 3. Overall architecture of AffinityNet. The output feature map f^{aff} is obtained by aggregating feature maps from multiple levels of a backbone network so that f^{aff} can take semantic information at various field-of-views. Specifically, we first apply 1×1 convolutions to the multi-level feature maps for dimensionality reduction, concatenate the results as a single feature map, and employ one more 1×1 convolution for adaptation to the target task. More details of the architecture is described in Section 4.

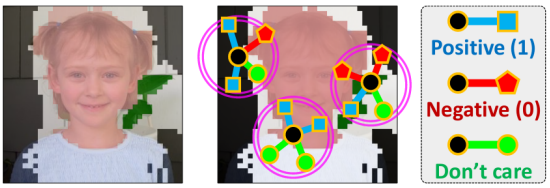


Figure 4. Conceptual illustration of generating semantic affinity labels. (a) Confident areas of object classes and background: peach for *person*, green for *plant*, and black for *background*. The *neutral* area is color-coded in white. (b) Coordinate pairs sampled within a small radius for training AffinityNet. Each pair is assigned label 1 if its two coordinates come from the same class, and label 0 otherwise. When at least one of the two coordinates belongs to the *neutral* area, the pair is ignored during training.

Affinity Net The second step is training affinity net, the training objective is to produce embeddings or affinities for each pixel, so that the L_1 distance between two pixels of the same class is 0 (in this case $W_{ij} = 1$) and inf otherwise (in this case $W_{ij} = 0$).

$$W_{ij} = \exp\left\{-\left\|f^{\text{aff}}(x_i, y_i) - f^{\text{aff}}(x_j, y_j)\right\|_1\right\}$$

However, it is computationally heavy to compute W_{ij} over the whole pixels. For each pixel, the authors only compute it over a small radius of size r (eg, 5). And using the CAMs, the loss can be computed using only the pairs of pixels with a given class.

$$\mathcal{L}^{+}_{\text{fg}} = -\frac{1}{\left|\mathcal{P}^{+}_{\text{fg}}\right|} \sum_{(i,j)\in\mathcal{P}^{+}_{\text{fg}}} \log W_{ij}$$

$$\mathcal{L}^{+}_{\text{bg}} = -\frac{1}{\left|\mathcal{P}^{+}_{\text{bg}}\right|} \sum_{(i,j)\in\mathcal{P}^{+}_{\text{bg}}} \log W_{ij}$$

$$\mathcal{L}^{-} = -\frac{1}{|\mathcal{P}-|} \sum_{(i,j)\in\mathcal{P}^{-}} \log\left(1 - W_{ij}\right)$$

Label Propagation Now with a trained affinity net, we can create a transition matrix of size $n \times n$, where each element is computed using the formula for W_{ij}, so the larger W_{ij}, the similar the pixel j is the pixel i, and in this case, if pixel j is not labeled (class ignored) we can assign to it the class of pixel i. Like training, each row i (corresponding to the similarity of pixel i to all the other pixel in the image) of the transition matrix is mostly zero only for the pixel within radius r of pixel i. The transition probability is derived from the affinity matrix as follows:

$$T = D^{-1}W^{\circ\beta}, \text{ where } D_{ii} = \sum_{j} W^{\beta}_{ij}$$

The transition matrix is then applied to the original CAMs (in the form of a vector of size n) with an added bg class (using alpha = 16) for a given number of iterations.

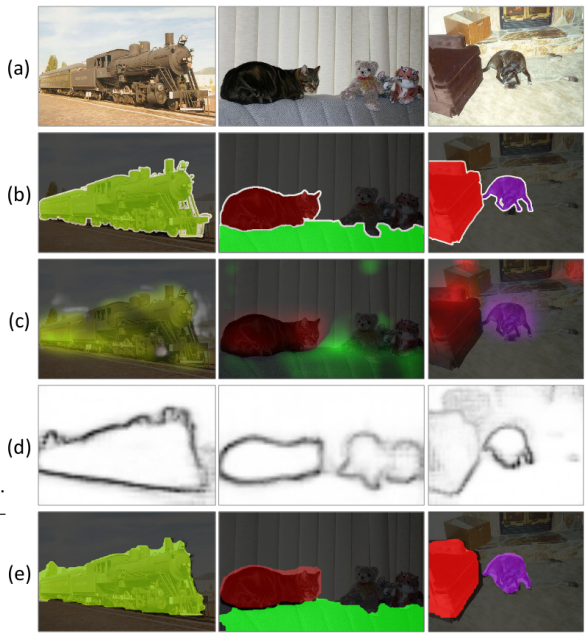
$$\text{vec}\left(M^{*}_{c}\right) = T^{t} \cdot \text{vec}\left(M_c\right) \quad \forall c \in C \cup \{\text{bg}\}$$

The last step, is to use the refined labels to train a segmentation network.

Results

Method	Sup.	Extra Data	*val*	*test*
TransferNet [10]	$\mathcal{I}$	MS-COCO [20]	52.1	51.2
Saliency [26]	$\mathcal{I}$	MSRA [21], BSDS [24]	55.7	56.7
MCNN [35]	$\mathcal{I}$	YouTube-Object [31]	38.1	39.8
CrawlSeg [11]	$\mathcal{I}$	YouTube Videos	58.1	58.7
What'sPoint [1]	$\mathcal{P}$	-	46.0	43.6
RAWK [36]	$\mathcal{S}$	-	61.4	-
ScribbleSup [18]	$\mathcal{S}$	-	63.1	-
WSSL [28]	$\mathcal{B}$	-	60.6	62.2
BoxSup [6]	$\mathcal{B}$	-	62.0	64.6
SDI [12]	$\mathcal{B}$	BSDS [24]	65.7	67.5
FCN [22]	$\mathcal{F}$	-	-	62.2
DeepLab [3]	$\mathcal{F}$	-	67.6	70.3
ResNet38 [38]	$\mathcal{F}$	-	80.8	82.5
Ours-DeepLab	$\mathcal{I}$	-	58.4	60.5
Ours-ResNet38	$\mathcal{I}$	-	61.7	63.7

Table 4. Performance on the PASCAL VOC 2012 *val* and *test* sets. The supervision types (Sup.) indicate: $\mathcal{P}$–point, $\mathcal{S}$–scribble, $\mathcal{B}$–bounding box, $\mathcal{I}$–image-level label, and $\mathcal{F}$–segmentation label.



Method	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbk	person	plant	sheep	sofa	train	tv	mean
EM-Adapt [28]	67.2	29.2	17.6	28.6	22.2	29.6	47.0	44.0	44.2	14.6	35.1	24.9	41.0	34.8	41.6	32.1	24.8	37.4	24.0	38.1	31.6	33.8
CCNN [29]	68.5	25.5	18.0	25.4	20.2	36.3	46.8	47.1	48.0	15.8	37.9	21.0	44.5	34.5	46.2	40.7	30.4	36.3	22.2	38.8	36.9	35.3
MIL+seg [30]	79.6	50.2	21.6	40.9	34.9	40.5	45.9	51.5	60.6	12.6	51.2	11.6	56.8	52.9	44.8	42.7	31.2	55.4	21.5	38.8	36.9	42.0
SEC [14]	82.4	62.9	26.4	61.6	27.6	38.1	66.6	62.7	**75.2**	22.1	53.5	28.3	65.8	57.8	62.3	52.5	32.5	62.6	32.1	45.4	45.3	50.7
AdvErasing [37]	83.4	**71.1**	30.5	72.9	41.6	55.9	63.1	60.2	74.0	18.0	**66.5**	32.4	71.7	56.3	64.8	52.4	37.4	69.1	31.4	58.9	43.9	55.0
Ours-DeepLab	87.2	57.4	25.6	69.8	45.7	53.3	76.6	**70.4**	74.1	28.3	63.2	**44.8**	75.6	**66.1**	65.1	71.1	40.5	66.7	37.2	58.4	49.1	58.4
Ours-ResNet38	**88.2**	68.2	**30.6**	**81.1**	**49.6**	**61.0**	**77.8**	66.1	75.1	**29.0**	66.0	40.2	**80.4**	62.0	**70.4**	**73.7**	**42.5**	**70.7**	**42.6**	**68.1**	**51.6**	**61.7**

Table 2. Performance on the PASCAL VOC 2012 *val* set, compared to weakly supervised approaches based only on image-level labels.

Method	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbk	person	plant	sheep	sofa	train	tv	mean
EM-Adapt [28]	76.3	37.1	21.9	41.6	26.1	38.5	50.8	44.9	48.9	16.7	40.8	29.4	47.1	45.8	54.8	28.2	30.0	44.0	29.2	34.3	46.0	39.6
CCNN [29]	70.1	24.2	19.9	26.3	18.6	38.1	51.7	42.9	48.2	15.6	37.2	18.3	43.0	38.2	52.2	40.0	33.8	36.0	21.6	33.4	38.3	35.6
MIL+seg [30]	78.7	48.0	21.2	31.1	28.4	35.1	51.4	55.5	52.8	7.8	56.2	19.9	53.8	50.3	40.0	38.6	27.8	51.8	24.7	33.3	46.3	40.6
SEC [14]	83.5	56.4	28.5	64.1	23.6	46.5	70.6	58.5	71.3	23.2	54.0	28.0	68.1	62.1	70.0	55.0	38.4	58.0	39.9	38.4	48.3	51.7
AdvErasing [37]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	55.7
Ours-DeepLab	88.0	61.1	29.2	73.0	40.5	54.1	75.2	**70.4**	**75.1**	27.8	62.5	51.4	78.4	**68.3**	76.2	71.8	40.7	**74.9**	**49.2**	55.0	48.3	60.5
Ours-ResNet38	**89.1**	**70.6**	**31.6**	**77.2**	**42.2**	**68.9**	**79.1**	66.5	74.9	**29.6**	**68.7**	**56.1**	**82.1**	64.8	**78.6**	**73.5**	**50.8**	70.7	47.7	**63.9**	**51.1**	**63.7**

Table 3. Performance on the PASCAL VOC 2012 *test* set, compared to weakly supervised approaches based only on image-level labels.