FickleNet: Weakly and Semi-supervised Semantic Image Segmentation using Stochastic Inference (2019)

Jungbeom Lee, Eunji Kim, Sungmin Lee, Jangho Lee, Sungroh Yoon

Notes

1 Introduction

In weakly supervised semantic segmentation, the difficulty is to create coarse image-level annotation, the majority of methods use localization maps trained from the classifier, but they only focus the small and most discriminative parts of the objects and do not capture precise boundaries, with FickleNet the authors propose a simple layers to be added to a standard convolutionnal neural network to selects hidden units randomly and then uses them to obtain activation scores for image classification, this will give the model the possibility to implicitly learn the coference of each location in the feature maps and resulting in a localization that identifies both discriminative and other parts of the objects.

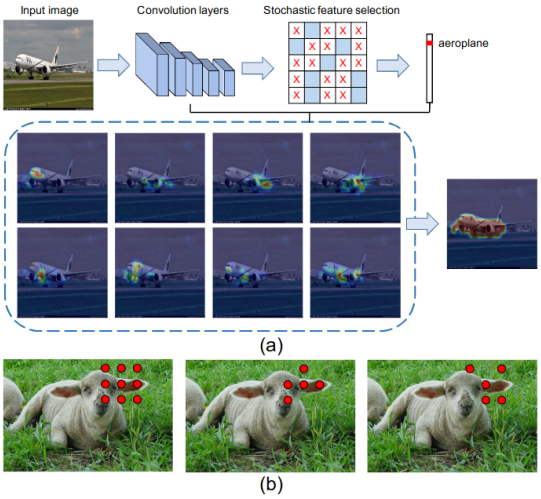

Figure 1. (a) FickleNet allows a single network to generate multiple localization maps from a single image. (b) Conceptual description of hidden unit selection. Selecting all hidden units (deterministic, *left*) produces smoothing effects as background and foreground are activated together. Randomly selected hidden units (stochastic, *center* and *right*) can provide more flexible combinations which can correspond more clearly to parts of objects or the background.

FickleNet chooses hidden units at random for each sliding window position, which corresponds to each stride in the convolution operation. This process is simply realized by the dropout method. Selecting all the available hidden units in a sliding window position (the deterministic approach) tends to produce a smoothing effect that confuses foreground and background, which can result in both areas being activated or deactivated together. However, random selection of hidden units (the stochastic approach) produces regions of different shapes which can delineate objects more sharply. Since the patternsof hidden units randomly selected by FickleNet include the shapes of the kernel of

the dilated convolution with different dilation rates, FickleNet can be regarded as a generalization of dilated convolution, but FickleNet can potentially match objects of different scales and shapes using only a single network because it is not limited to a square array of hidden units, whereas dilated convolution requires networks with different dilation rates just to scale its kernel.

2 FickleNet

FickleNet is trained on multi-class classification, it then generates localization maps which are considered as pseudo labels for the segmentation network, this is done as follows:

Algorithm 1: Training and Inference Procedure

Input: Image I, ground-truth label c, dropout rate p
Output: Classification score S and localization maps M

x = Forward(I) until conv5 layer;
Stochastic hidden unit selection: Sec. 3.1
    x^{expand} = Expand(x); Sec. 3.1.1
    x_p^{expand} = Center-fixed spatial dropout(x^{expand}, p); Sec. 3.1.2
    S = Classifier(x_p^{expand}); Sec. 3.1.3
Training Classifier:
    Update network by L=SigmoidCrossEntropy(S, c)
Inference CAMs: Sec. 3.2
    For different random selections i $(1 \leq i \leq N)$:
        $M^c[i]$ = Grad-CAM(x, S^c); Sec. 3.2.1
    M^c = Aggregate($M^c[i]$); Sec. 3.2.2

Stochastic Hidden Unit Selection

To be able to generate pseudo labels from the classification maps that contrain other parts than the discriminative ones, at each convolution at a given conv layer, instead of having a determenistic number of elements that contribute to each output activation, with FickleNet, at each iteration we sample a random number of pixels for the convolutions, say for a FxF filter, instead of having FxF elements in the input feature contribute to the output, we going to radomly remove some elements, this method of selecting hidden units can generate re-ceptive fields of many different shapes and sizes:

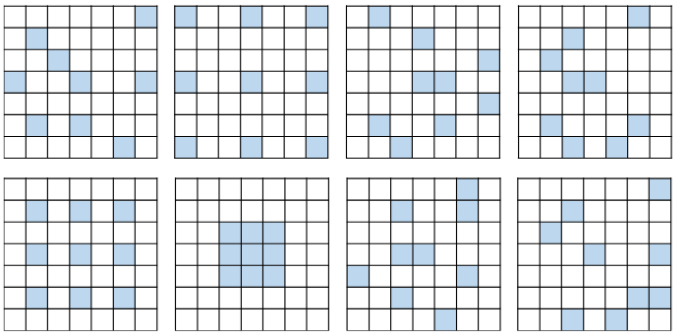

Figure 3. Examples of the selection of 9 hidden units (marked as blue) from a 7×7 kernel. Channels are not shown for simplicity. The shapes of those selected hidden units sometimes contain the shape of kernel of convolution with different dilation rates.

Now to implement this type of stochastic convolutions that samples new combinations in each sliding positions, we can either implement a new version of the convolution that randomly masks some elements of the inputs at each iteration (figure **a** below) by calling the spatial dropout function at each convolution ($h \times w$ times), or as an alternative, we can expand the input feature to a larger feature map, say for a conv 3x3 with a stride of 1, we'll expand our feature map by a factor of 3 in both the width and height, and then apply dropout and the convoltuion with a stride = 3,

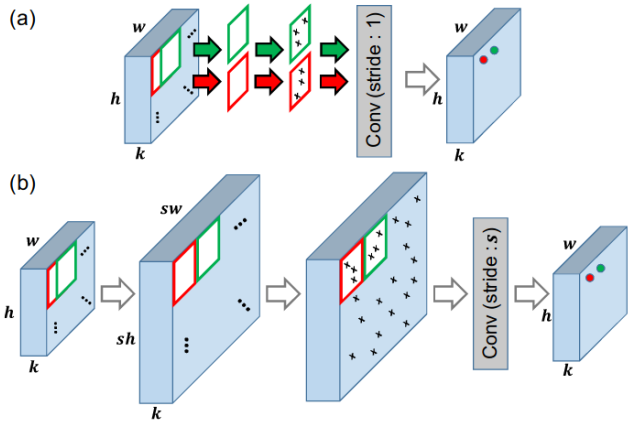

Figure 2. (a) Naive implementation of FickleNet, which requires a dropout and convolution function call at each sliding window position (the red and green boxes). (b) Implementation using map expansion: convolution is now performed once with a stride of s. The input feature map is expanded so that successive sliding kernels (the red and green boxes) do not overlap.

More precisely, we first add some zero padding to avoid shrinking the spatial dimensions, so the feature map of size $k \times h \times w$ becomes $k \times (h + s - 1) \times (w + s - 1)$ after padding, with s the size of the kernel, and then we exapand them to $k \times sh \times sw$ giving us x^{expand}, we then apply dropout to x^{expand} giving us x_p^{expand}, that we then convolve with a stride of s and size of s, to obtain the feature maps of size $c \times w \times h$, we then apply global average pooling and a sigmoid function for multiclass classification.

Localization maps

With the classification scores, we can use gradCam to backpropage the errors to the activation maps before expansion, and then sum them over the channel axis weighted by the gradients, this is expressed as follows:

$$\text{Grad} - \text{CAM}^{\text{c}} = \text{ReLU}\left(\sum_k x_k \times \frac{\partial S^c}{\partial x_k}\right)$$

where $x_k \in \mathbb{R}^{w \times h}$ is the k^{th} channel of the feature map x, and S_c is the classification score of class c, but this is only one localization map, to take into advantage the stochastic nature of fickle net, we can apply N forward passes and then construct N different localization maps from a single image and combine them, for N localization map, we assign a class c at a location w, h in the final map, if the values at a given map $M[i]$ is larger than a threshold $> \theta$, if we have many pixels $> \theta$ we average them.

3 Training and Experiments

In a semi supervised setting, we have the following loss function to optimize:

$$L = L_{\text{seed}} + L_{\text{boundary}} + \alpha L_{\text{full}}$$

L_{seed} and L_{boundary} are for the weakly supervised setting, and L_{full} is the cross entropy loss for the pixel wise labeled examples.

FickleNet is based on VGG16 pretrained on ImageNet, the last pooling layer was replaced by a block of dilated convolutions with a rate of 2, the dropout applied in the expansion is 0.9 and the kernel size is 9, input ilage is of size 321 x 321, learning rate of 0.001 that is halved each 10 epochs and the optimize used is ADAM, $N = 200$ and $\theta = 0.35$.

4 Resutls

Table 1. Comparison of weakly supervised semantic segmentation methods on VOC 2012 validation and test image sets. The methods listed here use DeepLab-VGG16 for segmentation.

Methods	Training	*val*	*test*
Supervision: Image-level and additional annotations			
MIL-seg $_{\text{CVPR '15}}$ [23]	700K	42.0	40.6
STC $_{\text{TPAMI '17}}$ [32]	50K	49.8	51.2
TransferNet $_{\text{CVPR '16}}$ [9]	70K	52.1	51.2
CrawlSeg $_{\text{CVPR '17}}$ [10]	970K	58.1	58.7
AISI $_{\text{ECCV '18}}$ [11]	11K	61.3	62.1
Supervision: Image-level annotations only			
SEC $_{\text{ECCV '16}}$ [16]	10K	50.7	51.1
CBTS-cues $_{\text{CVPR '17}}$ [24]	10K	52.8	53.7
TPL $_{\text{ICCV '17}}$ [14]	10K	53.1	53.8
AE_PSL $_{\text{CVPR '17}}$ [31]	10K	55.0	55.7
DCSP $_{\text{BMVC '17}}$ [2]	10K	58.6	59.2
MEFF $_{\text{CVPR '18}}$ [8]	10K	-	55.6
GAIN $_{\text{CVPR '18}}$ [19]	10K	55.3	56.8
MCOF $_{\text{CVPR '18}}$ [30]	10K	56.2	57.6
AffinityNet $_{\text{CVPR '18}}$ [1]	10K	58.4	60.5
DSRG $_{\text{CVPR '18}}$ [12]	10K	59.0	60.4
MDC $_{\text{CVPR '18}}$ [33]	10K	60.4	60.8
FickleNet (Ours)	10K	**61.2**	**61.9**

Table 2. Comparison of weakly supervised semantic segmentation methods on VOC 2012 validation and test image sets. The methods listed here use ResNet-based DeepLab for segmentation.

Methods	Backbone	*val*	*test*
MCOF [30]	ResNet 101	60.3	61.2
DCSP [2]	ResNet 101	60.8	61.9
DSRG [12]	ResNet 101	61.4	63.2
AffinityNet [1]	ResNet 38	61.7	63.7
FickleNet (ours)	ResNet 101	**64.9**	**65.3**

Table 3. Comparison of semi-supervised semantic segmentation methods on VOC 2012 validation sets. We also give the performances of DeepLab using 1.4K and 10.6K strongly annotated data.

Methods	Training Set	mIoU
DeepLab [3]	1.4K strong	62.5
WSSL [21]	1.4K strong + 9K weak	64.6
GAIN [19]	1.4K strong + 9K weak	60.5
MDC [33]	1.4K strong + 9K weak	65.7
DSRG [12] (baseline)	1.4K strong + 9K weak	64.3
FickleNet (ours)	1.4K strong + 9K weak	**65.8**
DeepLab [3]	10.6K strong	67.6

Table 4. Run time and GPU memory usage for training and CAM extraction without and with map expansion.

Methods	Training	CAM Extract	GPU Usage
Naive	20 sec/iter	2.98 sec/img	8.4 GB
Expansion	1.3 sec/iter	0.21 sec/img	10.1 GB

Table 5. Comparison of mIoU scores using different dropout rates (p) on PASCAL VOC 2012 validation images.

Methods	Dropout Rate (p)	mIoU
Deterministic	0.0	56.3
General Dropout	0.5	45.6
	0.9	49.1
FickleNet	0.3	58.8
	0.5	59.4
	0.7	60.0
	0.9	**61.2**

was provided with a very large number of unlabeled YouTube videos. AISI [11] utilized salient instance detector [7] which is trained using well-annotated instance-level annotations. Note that instance-level annotation is one of the most difficult annotations to obtain.