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

Jungbeom Lee Eunji Kim Sungmin Lee Jangho Lee Sungroh Yoon†
Electrical and Computer Engineering, ASRI, INMC, and Institute of Engineering Research
Seoul National University, Seoul 08826, South Korea
sryoon@snu.ac.kr

Abstract

The main obstacle to weakly supervised semantic image segmentation is the difficulty of obtaining pixel-level information from coarse image-level annotations. Most methods based on image-level annotations use localization maps obtained from the classifier, but these only focus on the small discriminative parts of objects and do not capture precise boundaries. FickleNet explores diverse combinations of locations on feature maps created by generic deep neural networks. It selects hidden units randomly and then uses them to obtain activation scores for image classification. FickleNet implicitly learns the coherence of each location in the feature maps, resulting in a localization map which identifies both discriminative and other parts of objects. The ensemble effects are obtained from a single network by selecting random hidden unit pairs, which means that a variety of localization maps are generated from a single image. Our approach does not require any additional training steps and only adds a simple layer to a standard convolutional neural network; nevertheless it outperforms recent comparable techniques on the Pascal VOC 2012 benchmark in both weakly and semi-supervised settings.

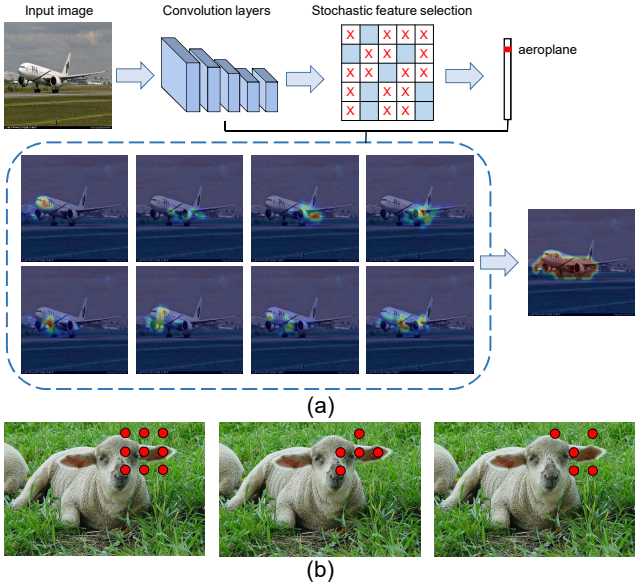


Figure 1. (a) FickleNet allows a single network to generate multiple localization maps from a single image. (b) Conceptual description of hidden unit selection. Selecting all hidden units (deterministic, *left*) produces smoothing effects as background and foreground are activated together. Randomly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

1. Introduction

Semantic segmentation is one of the most important and interesting tasks in computer vision, and the development of deep learning has produced tremendous progress in a fully supervised setting [3, 36]. However, to use semantic image segmentation in real life requires a large variety of object classes and a great deal of labeled data for each class. Labeling pixel-level annotations of each object class is laborious, and hampers the expansion of object classes. This problem can be addressed by weakly supervised methods that use annotations, which are less definite than those at the pixel level and much easier to obtain. However, current weakly supervised segmentation methods produce inferior results to fully supervised segmentation.

Pixel-level annotations allow fully supervised semantic segmentation to achieve reliability in learning the boundaries of objects and the relationship between their components. But, it is difficult to use image-level annotations to train segmentation networks because weakly labeled data only indicates the existence of objects of a certain class, and does not provide any information about their locations or boundaries. Most weakly supervised methods using image-level annotations depend on localization maps obtained by a classification network [37] to bridge the gap between

†Correspondence to: Sungroh Yoon <sryoon@snu.ac.kr>.

image-level and pixel-level annotations. However, these localization maps focus only on the small discriminative parts of objects, without precise representations of their boundaries. To bring the performance of these methods closer to that of fully supervised image segmentation means diverting the classifier from its primary task of discrimination between objects to discovering the relations between pixels.

We address this problem with FickleNet, which can generate a variety of localization maps from a single image using random combinations of hidden units in a convolutional neural network, as shown in Figure 1(a). Starting with a feature map created by a generic classification network such as VGG-16 [26], FickleNet chooses hidden units at random for each sliding window position, which corresponds to each stride in the convolution operation, as shown in Figure 1(b). This process is simply realized by the dropout method [28]. 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 patterns of 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.

The selection of random hidden units at each sliding window position is not an operation that is optimized at the CUDA level in common deep-learning frameworks such as PyTorch [22]. Thus, a naive implementation of FickleNet, in which random hidden units are selected at each sliding window position and then convolved, would require a large number of iterative operations. However, we can use the optimized convolution functions provided by deep-learning frameworks, if we expand the feature maps before making the random selection of hidden units. The maps need to be expanded sufficiently to prevent successive sliding window positions from overlapping. We can then apply dropout in the spatial axis of the expanded feature maps, and perform a convolution operation with a stride equal to the kernel size. This saves a significant amount of time without much increase in GPU memory usage, because the number of parameters to be back-propagated remains constant.

While many existing networks use stochastic regularization in their training process (e.g. Dropout [28]), stochastic effects are usually excluded from the inference process. However, our inference process contains random processes and thus produces a variety of localization maps. The pixels that were allocated to a specific class with high scores in each localization map are discovered, and those pixels are aggregated into a single localization map. The localization map obtained from FickleNet is utilized as pseudo-labels for the training of a segmentation network.

The main contributions of this paper can be summarized as follows:

- We propose FickleNet, which is simply realized using the dropout method, that discovers the relationship between locations in an image and enlarges the regions activated by the classifier.
- We introduce a method of expanding feature maps which makes our algorithm much faster, with only a small cost in GPU memory.
- Our work achieves state-of-the-art performance on the Pascal VOC 2012 benchmark in both weakly supervised and semi-supervised settings.

2. Related Work

Weakly supervised semantic image segmentation methods substitute inexact annotations such scribbles, bounding boxes, or image-level annotations, for strong pixel-level annotations. The methods of recent introduction have achieved successful results using annotations that provide location information such as scribbles or bounding boxes [4, 29]. We now review some recently introduced weakly supervised approaches which use image-level annotations.

A class activation map (CAM) [37] is a good starting-point for the classification of pixels from image-level annotations. A CAM discovers the contribution of each hidden unit in a neural net to the classification score, allowing the hidden units which make large contributions to be identified. However, a CAM tends to focus on the small discriminative region of a target object, which makes it unsuitable for training a semantic segmentation network. Weakly supervised methods of recent introduction expand the regions activated by a CAM, operating on the image (Section 2.1), on features (Section 2.2), or by growing the regions found by a CAM (Section 2.3).

2.1. Image-level Processing

Image-level hiding and erasure have been proposed [19, 27, 31] as ways of preventing a classifier from focusing exclusively on the discriminative parts of objects. HaS [27] hides random regions of a training image, forcing the classification network to seek other parts of the object. However, the process of hiding random regions does not consider the semantics and sizes of objects. AE-PSL [31] starts with a single small region in the object, and then drives the classification network to discover a sequence of new and complement any object regions by erasing the regions that have

already been found. Although it can progressively expand regions belonging to an object, it requires multiple classification networks to perform the repetitive classification and erasure steps. GAIN [19] has a CAM which is trained to erase regions in a way that deliberately confuses the classifier. This CAM has to be large enough to cover an entire object. However, the classifier mainly reacts to high activation, and so it can become confused if an object's only discriminative parts are erased.

2.2. Feature-level Processing

Feature-level processing can be used to expand the regions activated by a CAM. ACoL [35] and TPL [14] use a classifier to identify the discriminative parts of an object and erase them based on features. A second classifier then is trained to find the complementary parts of the object from those erased features. This is an efficient technique which operates at a relatively high level. However, it has a similar drawback to image-level erasure, in that a second classifier and training step are essential for those methods, which may cause a suboptimal performance. In addition, features whose discriminative parts are erased can confuse the second classifier, which may not be correctly trained. PG-CAM [18] collects features from each of several densely connected layers and merges the resulting localization maps.

MDC [33] uses several convolutional blocks, dilated at different rates, within a generic classification network, and aggregates CAMs obtained from each block in a process that resembles ensemble learning. The different-sized receptive fields produced by different dilation rates can be shown to capture different patterns, but MDC requires a separate training procedure for each dilation rate, and its limitation to integer dilation rates (e.g. 1, 3, 6, 9) means that only a limited number of ensembles is possible. In addition, the receptive field produced by a standard dilated convolution is square with a fixed size, so that MDC tends to identify false positive regions.

2.3. Region Growing

Region growing can be used to expand the localization map produced by a CAM, which initially identifies just the small discriminative part of an object. AffinityNet [1] learns pixel-level semantic affinities, which identify pixels belonging to the same object, under the supervision of an initial CAM, and then expands the initial CAM by a random walk with the transition matrix computed from semantic affinities. However, the learning of semantic affinities requires an additional network, and the outcome depends heavily on the quality of the CAM. SEC [16] uses a new type of loss function to expand the localization map and constrain it to object boundaries using a conditional random field (CRF) [17]. DSRG [12] refines initial localization maps during the training of its segmentation network, so that DSRG does not require additional networks to grow regions. The seeds for region growing are obtained from a CAM, and if these seeds only come from the discriminative parts of objects, it is difficult to grow regions into non-discriminative parts. We therefore utilize as a segmentation network with the localization maps produced by FickleNet.

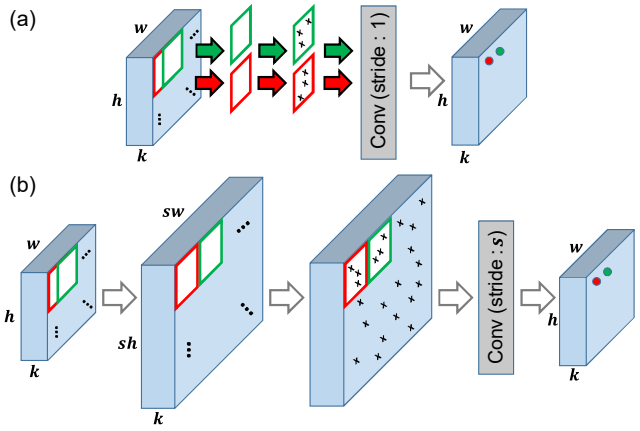


Figure 2. (a) Naive implementation of FickleNet, which requires a dropout and convolution function call at each sliding window position (the red and green boxes). (b) Implementation using map expansion: convolution is now performed once with a stride of s. The input feature map is expanded so that successive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

3. Proposed Method

Our procedure has the following steps: FickleNet, which uses stochastic selection of hidden units, is trained for multi-class classification. It then generates localization maps of training images. Finally, the localization maps are used as pseudo-labels to train a segmentation network. We denote the sort of feature map typically obtained from a standard deep neural network as $x \in \mathbb{R}^{k \times h \times w}$, where w and h are the width and the height of each of k channels, respectively. The procedures for training FickleNet and generating localization maps are shown as Algorithm 1.

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

3.1. Stochastic Hidden Unit Selection

Stochastic hidden unit selection is used in FickleNet to discover relations between parts of objects by exploring the classification score computed from the randomly selected pairs of hidden units, with the aim of associating a non-discriminative part of an object with a discriminative part of the same object. This process is realized by applying spatial dropout [28] to the feature x at each sliding window position, as shown in Figure 2(a). This differs from the standard dropout technique, which only samples hidden units in the feature maps once in each forward pass, and thus hidden units which are not sampled cannot contribute to the class scores. Our method samples hidden units at each sliding window position, which means that a hidden unit may be activated at some window positions and dropped at others.

This method of selecting hidden units can generate receptive fields of many different shapes and sizes, as shown in Figure 3. Some of these fields are likely to be similar to those produced by a standard dilated convolution; thus the results produced by this technique can be expected to contain those produced by standard dilated convolution at various rates. This selection process can be simply and efficiently realized by the expansion technique described in Section 3.1.1 with a method which we call center-preserving dropout, which is described in Section 3.1.2.

3.1.1 Feature Map Expansion

As our method needs to sample new combinations in each sliding window position, we cannot directly utilize the CUDA-level optimized convolution functions provided by popular deep learning frameworks such as PyTorch [22]. If we were to implement our method naively, as shown in Figure 2(a), we would have to call the convolution function and the dropout function in $w \times h$ times in each forward pass. By expanding the feature map, we reduce this to a single call to each function during each forward pass.

Figure 2(b) shows how we expand the input feature maps so that no sliding window positions overlap. Before expanding the feature map, we apply zero padding on x so that the size of the final output is equal to that of the input. The size of the feature map after zero padding becomes $k \times (h + s - 1) \times (w + s - 1)$, where s is the size of the convolution kernel. We expand the zero-padded feature map so that successive sliding window positions do not overlap, and the size of the expanded feature map x^{expand} is $k \times (sh) \times (sw)$. We then select hidden units on x^{expand} using the center-preserving dropout technique explained in Section 3.1.2. Examples of this expansion process are presented in the appendix (Section **??**). Although the expanded feature map requires more GPU memory, the number of parameters to be trained remains constant, and so the load on the GPU does not increase significantly.

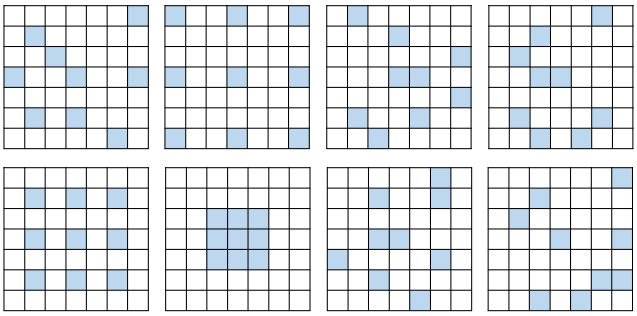


Figure 3. Examples of the selection of 9 hidden units (marked as blue) from a 7×7 kernel. Channels are not shown for simplicity. The shapes of those selected hidden units sometimes contain the shape of kernel of convolution with different dilation rates.

3.1.2 Center-preserving Spatial Dropout

We realize stochastic hidden unit selection by applying the dropout method [28] to spatial locations. We can achieve the same results as the naive implementation by applying dropout only once to the expanded feature map x^{expand}. Note that dropout is applied uniformly across all channels. We do not drop the center of the kernel of each sliding window position, so that relationships between kernel center and other locations in each stride can be found. After spatial dropout with a rate of p, we denote the modified feature map as x_p^{expand}. While dropout is usually only employed during training, we apply it to both training and inference.

3.1.3 Classification

In order to obtain classification scores, convolution with kernel of size s and stride s are applied to the dropped feature map x_p^{expand}. We then obtain an output feature map of size $c \times w \times h$, where c is the number of object classes. By applying global average pooling and a sigmoid function to this map, we obtain a classification score S. We then update FickleNet using the sigmoid cross-entropy loss function, which is widely used for multi-label classification.

3.2. Inference Localization Map

We can now obtain various classification scores from a single image, which correspond to randomly selected combinations of hidden units, and each random selection generates a various localization map. Section 3.2.1 describes how to obtain a localization map from each random selection, and Section 3.2.2 describes how the maps from the random selections are aggregated into a single localization map.

3.2.1 Grad-CAM

We use gradient based CAM (Grad-CAM) [25], which is a generalization of class activation map (CAM) [37], to ob-

tain localization maps. Grad-CAM discovers the class specific contribution of each hidden unit to the classification score from gradient flows. We compute the gradients of the target class score with respect to x, which is the feature map before expansion, and then sum the feature maps along the channel axis, weighted by these gradients. We can express Grad-CAM for each target class c as follows:

$$\text{Grad-CAM}^{\text{c}} = \text{ReLU}(\sum_k x_k \times \frac{\partial S^c}{\partial x_k}), \quad (1)$$

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.

3.2.2 Aggregate Localization Map

FickleNet allows many localization maps to be constructed from a single image, because different combinations of hidden units are used to compute classification scores at each random selection. We construct N different localization maps from a single image and aggregate them into a single localization map. Let $M[i]$ $(1 \leq i \leq N)$ denote the localization map constructed from the i^{th} random selection. We aggregate the N localization maps so that a pixel located at u in the aggregated map is allocated to class c if the activation score for class c in any $M[i]$ at u is higher than a threshold θ. Pixels which are not allocated to any class are ignored during training. If there is a pixel assigned to multiple classes, we examine its class score in a map averaged over the N maps and assign the pixel to the class with the highest score in the average map.

3.3. Training the Segmentation Network

The localization map, whose construction was described in Sections 3.1 and 3.2, provides pseudo-labels to train a semantic image segmentation network. We use the same background cues as DSRG [12]. We feed the generated localization maps from FickleNet to DSRG as the seed cues for weakly supervised segmentation.

For semi-supervised learning we introduce an additional loss derived from data fully annotated by a person. Let $\mathcal{C}$ be the set of classes that are present in the image. We train a segmentation network with the following loss function:

$$L = L_{\text{seed}} + L_{\text{boundary}} + \alpha L_{\text{full}}, \quad (2)$$

where L_{seed} and L_{boundary} respectively are the balanced seeding loss and boundary loss used in DSRG [12], and

$$L_{\text{full}} = -\frac{1}{\sum_{c \in \mathcal{C}} |F_c|} \sum_{c \in \mathcal{C}} \sum_{u \in F_c} \log H_{u,c}, \quad (3)$$

where $H_{u,c}$ is the probability of an entry of class c at location u in the segmentation map H, and F_c is the ground-truth mask.

4. Experiments

4.1. Experimental Setup

Dataset: We conducted experiments on the PASCAL VOC 2012 image segmentation benchmark [6], which contains 21 object classes, including one background class. Using the same protocol as other work on weakly supervised semantic segmentation, we trained our network using augmented 10,582 training images with image-level annotations. We report mean intersection-over-union (mIoU) for 1,449 validation images and 1,456 test images. The results for the test images were obtained on the official PASCAL VOC evaluation server.

Network details: FickleNet is based on the VGG-16 network [26], pre-trained using the Imagenet [5] dataset. The VGG-16 network was modified by removing all fully-connected layers and the last pooling layer, and we replaced the convolution layers of the last block with dilated convolutions with a rate of 2. We set the kernel size s and the dropout rate p to 9 and 0.9 respectively. Segmentation is performed by DSRG [12].

Experimental details: We trained FickleNet using a mini-batch size to 10. We cropped the training images to 321×321 pixels at random locations, so that the size of feature map x becomes $512 \times 41 \times 41$. The initial learning rate was set to 0.001 and halved every 10 epochs. We used the Adam optimizer [15] with its default settings. During segmentation training, we use the same settings as DSRG [12]. We set the number of different localization maps N for each image to 200 and the threshold θ to 0.35. We set α to 2 for semi-supervised learning.

Reproducibility: PyTorch [22] was used for training FickleNet and conducting localization maps, and we used the Caffe framework [13] in the segmentation step. All the experiments were performed on an NVIDIA TITAN Xp GPU.

4.2. Comparison to the State of the Art

Weakly supervised segmentation: We compared our method with other recently introduced weakly supervised semantic segmentation methods with various levels of supervision. Table 1 shows results on PASCAL VOC 2012 images. Our method outperformed others which provide the same level of supervision through image-level annotations, achieving mIoU values of 61.2 and 61.9 for validation and test images respectively. This represents a 2.2% and 1.5% improvement respectively on validation and test images, when compared to DSRG, which is our backbone network. Note that we do not need additional training steps or additional networks, in contrast to many other recent techniques, such as AffinityNet [1], which requires an additional network for learning semantic affinities, or AE-PSL [31], which requires several training steps.

Table 2 shows result on PASCAL VOC 2012 images

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 segmentation models.

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**

with a ResNet-based segmentation network. We achieved mIoU values of 64.9 and 65.3 for validation and test images respectively using DeepLab-v2-ResNet101. This represents 3.5% and 2.1% improvement, respectively, on validation and test images, when compared to DSRG. AffinityNet [1] uses ResNet-38 based network [34], which has more powerful representation ability than ResNet-101.

Our method also significantly outperforms methods based on additional supervision except AISI [11]. These methods include TransferNet [9], which was trained on pixel-level annotations of 60 classes (not Pascal VOC classes) of COCO [20] images, and CrawlSeg [10], which 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.

Figure 4 shows qualitative results of predicted segmentation masks, in FickleNet and DSRG. The supervision provided by FickleNet produces larger and more accurate regions of a target object than that used in DSRG, allowing the segmentation network to consider a wider range of target objects. Thus, the segmentation network trained with localization maps generated by FickleNet produces more accurate results than DSRG in that FickleNet can make fewer false positives and cover larger regions of a target object.

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**

Semi-supervised segmentation: Table 3 shows that the mIoU of 65.8 produced by our method, trained on only 13.8% of images with pixel-level annotations in the PASCAL dataset, was 97.3% of that of Deeplab, which is trained with fully annotated data. The performance of FickleNet on validation images was 1.5% better than that of DSRG which is our baseline network. Note that GAIN shows lower performance than Deeplab, trained on only 1.4K fully annotated data. GAIN uses pixel-level annotations for the training of a classifier, rather than a segmentation network so that pixel-level ground-truth indirectly affects the training of the segmentation network. Figure 4 shows examples of segmentation maps from DSRG and FickleNet, which demonstrate that our system is able to operate satisfactorily in a semi-supervised manner.

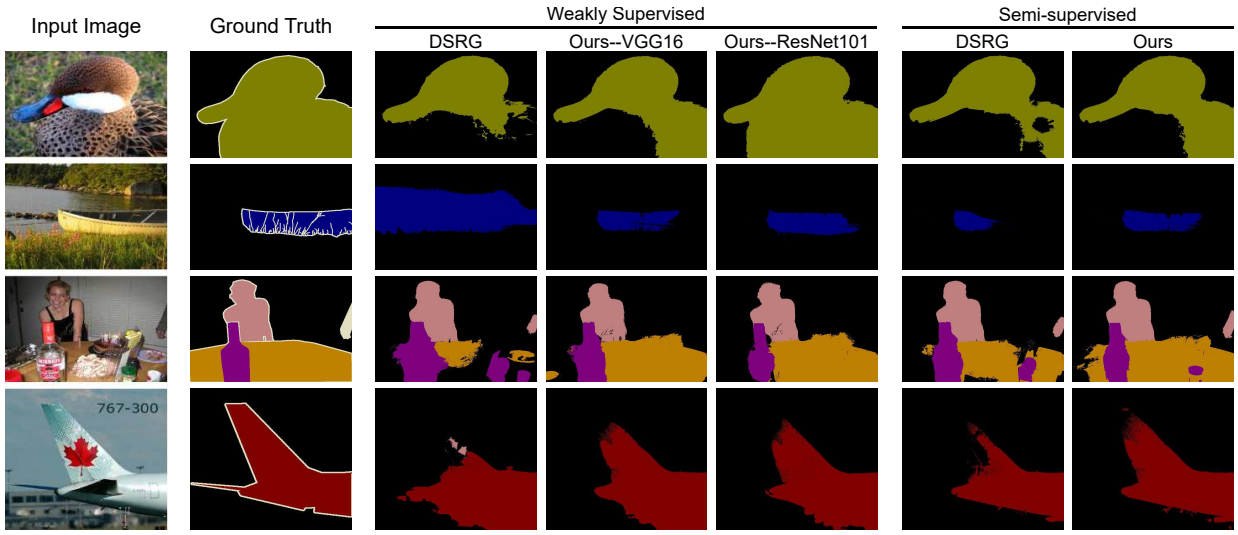
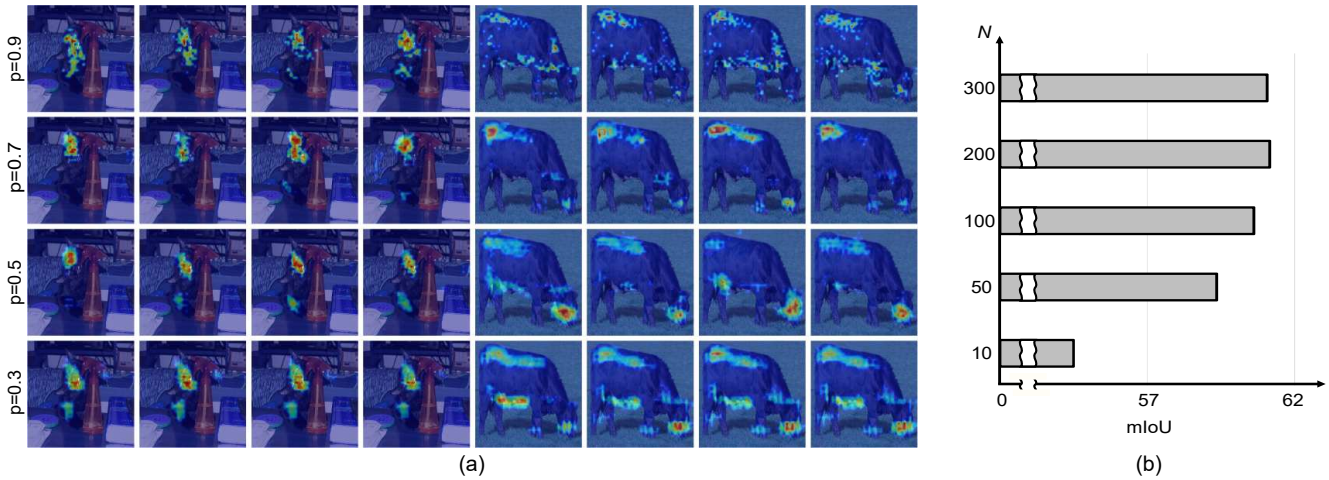


Figure 4. Examples of predicted segmentation masks for Pascal VOC 2012 validation images in weakly and semi-supervised manner.



(a) (b)

Figure 5. (a) Localization maps from each random selection of hidden unit with different dropout rates p. (b) Performance on Pascal VOC 2012 validation images for different N.

4.3. Ablation studies

Effects of the Map Expansion Technique: In order to show the effect of the map expansion technique presented in Section 3.1.1, we compare runtime and GPU usage of a naive implementation of FickleNet (Fig. 2(a)) with that of an implementation of FickleNet with map expansion (Fig. 2(b)). Table 4 shows that training and CAM extraction times are reduced factors of 15.4 and 14.2 respectively, at a cost of 12% in GPU memory usage.

Analysis of Iterative Inference: We compare mIoU scores with different numbers of localization maps N from a single image. Figure 5(b) shows that the mIoU increases with the number of maps N. We interpret this as meaning that additional random selection identifies more regions of a target object, so that larger regions of that object are represented by the aggregated localization map. If N is greater than 200, the mIoU converges to 61.2. Examples of different CAMs obtained from a single image are shown in Figure 5(a).

Table 6. Effectiveness of each step. $\mathcal{G}-$ general dropout, $\mathcal{S}-$ stochastic selection, $\mathcal{D}-$ deterministic approach.

Training	$\mathcal{G}$	$\mathcal{G}$	$\mathcal{G}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{D}$
Inference	$\mathcal{G}$	$\mathcal{S}$	$\mathcal{D}$	$\mathcal{S}$	$\mathcal{D}$	$\mathcal{D}$
mIoU	49.1	55.5	57.1	**61.2**	59.6	59.0

Effects of dropout rate: We analyzed the effects of the dropout rate used by FickleNet. Figure 6 shows that a dropout rate p of 0.9 allows FickleNet to cover larger regions of the target object than DSRG, which uses the localization maps from deterministic classifiers. Higher dropout rates also lead to more widely activated localization maps, because it becomes more likely that the discriminative part of an object will be dropped, leaving the non-discriminative parts of the object to be considered for classification. Conversely, if the dropout rate is low, the discriminative parts of objects are unlikely to be dropped, and they usually suffice for classification; so the classifier is unlikely to activate non-

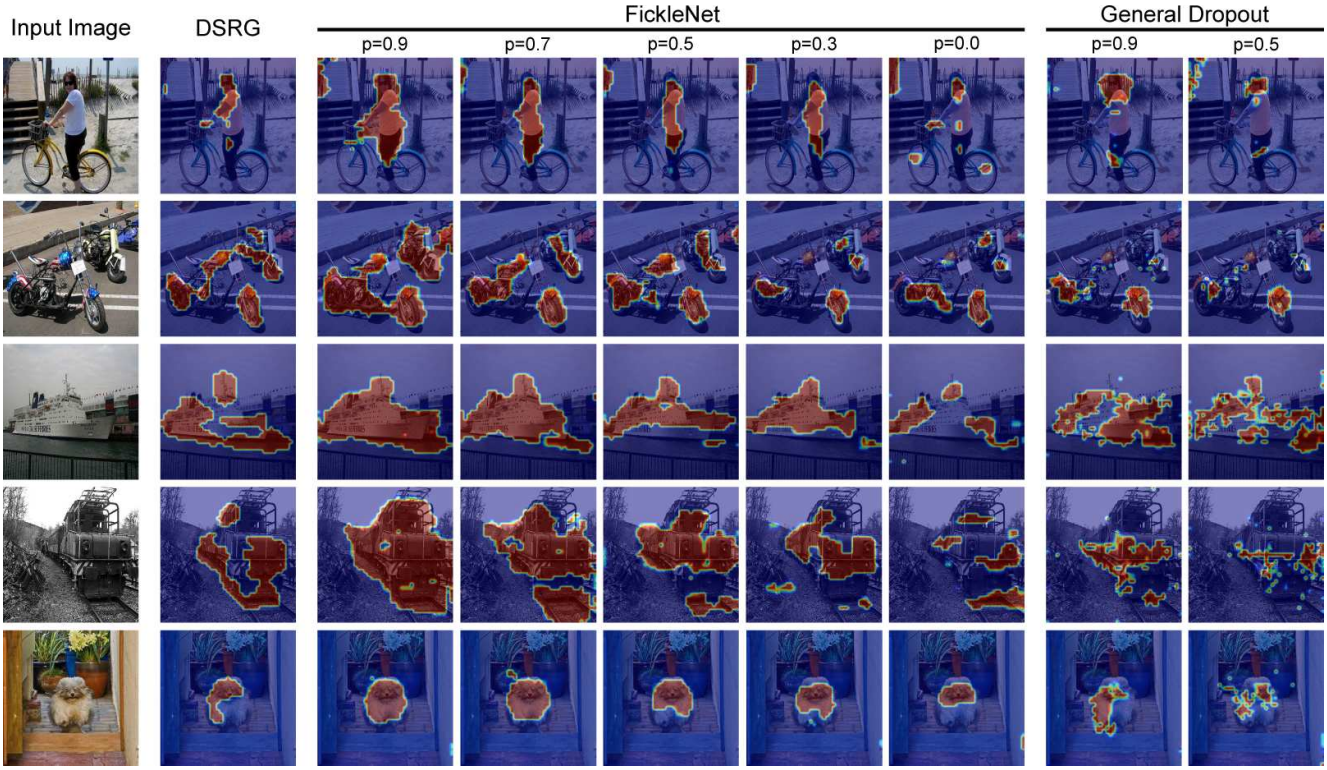


Figure 6. Localization maps from DSRG and FickleNet, with various dropout rates (p = 0 denotes a deterministic network), and from the general dropout method. Localization maps of DSRG (the 2nd column) were visualized using the publicly available DSRG localization cue.

discriminative parts. As shown in Figure 5(a), FickleNet with a low dropout rate tends to activate only the discriminative part of objects, even though random sampling produces many patterns of hidden units. Higher dropout rates result in more randomness in the activated patterns so that different non-discriminative parts of an object are more likely to be considered for each random selection. This effect is also reflected in the quantitative results shown in Table 5.

Comparison to general dropout: We compared FickleNet with a network created using a general dropout method rather than the hidden unit selection. Figure 6 shows that localization maps from the network created with general dropout tend to show noisy activation: hidden units which are not sampled cannot contribute to the class score during a forward pass, which means these dropped units do not contribute to the localization map. Note that a hidden unit in FickleNet may be activated at some window positions and dropped at others so that every hidden unit is able to affect the classification score. In Table 5, a segmentation network trained with localization maps from the network with general dropout shows inferior results to FickleNet.

Effectiveness of each step: Table 6 shows results obtained using several combinations of general dropout ($\mathcal{G}$), stochastic selection ($\mathcal{S}$), and the deterministic approach ($\mathcal{D}$) for training and inference. As expected, "train $\mathcal{S}$ + infer $\mathcal{D}$" is better than "train $\mathcal{D}$ + infer $\mathcal{D}$", because stochastic selection lets the network consider the non-discriminative part, but the best mIoU is obtained by "train $\mathcal{S}$ + infer $\mathcal{S}$".

5. Conclusions

We have addressed the problem of semantic image segmentation using only image-level annotations. By choosing features at random during both training and inference, we obtain many different localization maps from a single image, and then aggregate those maps into a single localization map. This map contains regions corresponding to parts of objects which are both larger and more consistent than those on a map produced by an equivalent deterministic technique. Our method can be implemented efficiently using operations readily available on a GPU by expanding the feature maps to avoid overlaps between the sliding kernels used during convolution. We show that the results produced by FickleNet on both weakly supervised and semi-supervised segmentation are better than those produced by other state-of-the-art approaches.

Acknowledgements: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) [2018R1A2B3001628], AIR Lab (AI Research Lab) in Hyundai Motor Company through HMC-SNU AI Consortium Fund, and the Brain Korea 21 Plus Project in 2019.

References

[1] J. Ahn and S. Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. *arXiv preprint arXiv:1803.10464*, 2018. 3, 5, 6

[2] A. Chaudhry, P. K. Dokania, and P. H. Torr. Discovering class-specific pixels for weakly-supervised semantic segmentation. *arXiv preprint arXiv:1707.05821*, 2017. 6

[3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014. 1, 6

[4] J. Dai, K. He, and J. Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1635–1643, 2015. 2

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009. 5

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[7] R. Fan, Q. Hou, M.-M. Cheng, T.-J. Mu, and S.-M. Hu. s^4 net: Single stage salient-instance segmentation. *arXiv preprint arXiv:1711.07618*, 2017. 6

[8] W. Ge, S. Yang, and Y. Yu. Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1277–1286, 2018. 6

[9] S. Hong, J. Oh, H. Lee, and B. Han. Learning transferrable knowledge for semantic segmentation with deep convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3204–3212, 2016. 6

[10] S. Hong, D. Yeo, S. Kwak, H. Lee, and B. Han. Weakly supervised semantic segmentation using web-crawled videos. *arXiv preprint arXiv:1701.00352*, 2017. 6

[11] S.-M. Hu. Associating inter-image salient instances for weakly supervised semantic segmentation. 2018. 6

[12] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang. Weakly-supervised semantic segmentation network with deep seeded region growing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7014–7023, 2018. 3, 5, 6

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014. 5

[14] D. Kim, D. Cho, D. Yoo, and I. S. Kweon. Two-phase learning for weakly supervised object localization. *arXiv preprint arXiv:1708.02108*, 2017. 3, 6

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[16] A. Kolesnikov and C. H. Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *European Conference on Computer Vision*, pages 695–711. Springer, 2016. 3, 6

[17] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011. 3

[18] S. Lee, J. Lee, J. Lee, C.-K. Park, and S. Yoon. Robust tumor localization with pyramid grad-cam. *arXiv preprint arXiv:1805.11393*, 2018. 3

[19] K. Li, Z. Wu, K.-C. Peng, J. Ernst, and Y. Fu. Tell me where to look: Guided attention inference network. *arXiv preprint arXiv:1802.10171*, 2018. 2, 3, 6

[20] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[21] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. *arXiv preprint arXiv:1502.02734*, 2015. 6

[22] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 2, 4, 5

[23] P. O. Pinheiro and R. Collobert. From image-level to pixel-level labeling with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1713–1721, 2015. 6

[24] A. Roy and S. Todorovic. Combining bottom-up, top-down, and smoothness cues for weakly supervised image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3529–3538, 2017. 6

[25] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 4

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 5

[27] K. K. Singh and Y. J. Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 2, 4

[29] M. Tang, F. Perazzi, A. Djelouah, I. B. Ayed, C. Schroers, and Y. Boykov. On regularized losses for weakly-supervised cnn segmentation. *arXiv preprint arXiv:1803.09569*, 2018. 2

[30] X. Wang, S. You, X. Li, and H. Ma. Weakly-supervised semantic segmentation by iteratively mining common object features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1354–1362, 2018. 6

[31] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *IEEE CVPR*, volume 1, page 3, 2017. 2, 5, 6

[32] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, J. Feng, Y. Zhao, and S. Yan. Stc: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2314–2320, 2017. 6

[33] Y. Wei, H. Xiao, H. Shi, Z. Jie, J. Feng, and T. S. Huang. Revisiting dilated convolution: A simple approach for weakly- and semi-supervised semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7268–7277, 2018. 3, 6

[34] Z. Wu, C. Shen, and A. v. d. Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv preprint arXiv:1611.10080*, 2016. 6

[35] X. Zhang, Y. Wei, J. Feng, Y. Yang, and T. Huang. Adversarial complementary learning for weakly supervised object localization. In *IEEE CVPR*, 2018. 3

[36] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017. 1

[37] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016. 1, 2, 4