Self-supervised Difference Detection for Refinement CRF and Seed Interpolation

Wataru Shimoda  Keiji Yanai  
Department of Informatics, The University of Electro-Communications, Tokyo

Abstract

To minimize annotation costs associated with training of semantic segmentation models, weakly-supervised segmentation approaches have been extensively studied. In this paper, we propose a novel method: Self-Supervised Difference Detection (SSDD) module which evaluates confidence of each of the pixels of segmentation masks and integrate highly confident pixels of two candidate masks.

1. Introduction

In this paper, we focus on avoiding performance degradation by CRF. In the proposed method, we estimate confidence maps on segmentation masks before and after applying CRF, and integrate them into a final mask by selecting more confident pixels comparing both confident maps. To estimate confidence maps of segmentation masks, we propose a Self-Supervised Difference Detection (SSDD) module. SSDD contains a Difference Detection network (DD-net), which estimates differences of the regions. Since the difference of segmentation masks can be obtained automatically without any human supervision, we can train a DD-net without any additional information. Thus, training of DD-net can be considered as self-supervised learning.

It is hard to estimate complicated parts even if an input region mask contains the regions to be estimated, while it is easy to estimate simple parts. Then we regard an output of a DD-net as confidence maps of difference between the masks before or after a refinement process. By using the confidence maps on mask difference, we pick up the pixels having higher confidence from both the masks, and generate re-refined segmentation masks.

In this paper, we demonstrate that the propose SSDD module can be used in both the seed generation stage and the training stage of fully-supervised segmentation as a “re-refinement module”. In the seed generation stage, we refine the CRF results of PSA [1] by a SSDD module. In the training stage, we introduce two SSDD modules inside training loop of a fully-supervised segmentation network. In the experiments, we demonstrate the effectiveness of SSDD modules in both stages. Especially, SSDD modules boosted the performance of weakly-supervised semantic segmentation on the PASCAL VOC 2012 dataset greatly and, achieved new state-of-the-art.

2. Related Works

Region refinement for WSS results using CRF [2] can refine the ambiguous outlines using low-level features such as a color of pixels. Chen et al. [3] and Pathak et al. [4] adopted CRF as a post-processing method for region refinement and demonstrated the effectiveness of the CRF for WSS. Kolesnikov et al. [5] proposed to use CRF during training of a semantic segmentation model. Ahn et al. [1] proposed a method to learn pixel-level similarity from CRF results, and apply random walk based region refinement, which achieved the state-of-the-art on the Pascal VOC 2012 dataset. We focus on preventing a segmentation mask from being degraded by applying CRF.

Generating pixel-level labels during training of a fully-supervised semantic segmentation (FSS) model [6] on the PASCAL VOC 2012 dataset greatly improved the performance of weakly-supervised semantic segmentation (WSS) model. EM-adopt [7] and Prohibitive Segment Propagation (PSP) [8] proposed online Prohibitive Segmentation Propagation (PSL), that utilizes classification score and the output of the segmentation model for generating masks in training. Huang et al. [8] proposed Deep Seeded Region Growing (DSRG), which is a method to expand the seed region during training. The authors prepared pixel-level seed labels before training that have unlabeled regions for unconsidered pixels. In this work, we proposed new constraint for the generating pixel-level labels during training of a FSS model.

3. Method for “Re-refinement” of Refinement

In this paper, we focus on avoiding performance degradation by refinement methods of segmentation masks such as CRF. To do that, we propose a Self-Supervised Difference Detection (SSDD) module. Fig.1 shows the basic idea of the processing in SSDD.

Difference Detection Network We denote one pair of segmentation masks as \( (m_{\text{before}}, m_{\text{after}}) \). Basically, we consider \( m_{\text{before}} \) is a raw segmentation mask and \( m_{\text{after}} \) is a processed mask. We denote their difference regions by a binary mask \( M_{\text{before,after}} \). The difference regions defined by:

\[
M_{u}^{\text{before,after}} = \begin{cases} 
1 & \text{if } (m_{u}^{\text{before}} = m_{u}^{\text{after}}) \\
0 & \text{if } (m_{u}^{\text{before}} \neq m_{u}^{\text{after}})
\end{cases},
\]

where \( u \in \{1, 2, ..., n\} \) indicates a location of pixels, and
\( \theta \) is a binarized mask with the same number of channels to the \( \theta \) image. We suppose that an input mask which is difficult to apply CRF. As shown in Fig. 2, the confidence map of the input mask, \( d \), is generated by a Difference Detection network, \( DDnet(e(x; \theta_e), e'(x; \theta_d), \hat{m}; \theta_d) \), \( d \in \mathbb{R}^{H \times W} \), where \( \hat{m} \) is a binarized mask with the same number of channels to the target class number. The architecture of DD-Net is shown in Fig. 3.

Self-Supervised Difference Detection Module In this section, we describe the detail of a SSDD module shown in Fig. 2. We suppose that an input mask which is difficult to predict difference regions is relatively reliable. This insight is based on an assumption that the overall processed masks, \( m^\text{after} \), have enough correct labels against error labels in different regions in the training phase.

We define confidence score \( w \) based on the difference detection results between \( (d^\text{before}, d^\text{after}) \). We calculate the confidence score \( w \) for each pixel \( u \) as follows:

\[
w^\text{before,after}_u = (d^\text{after}_u + b^\text{before,after}_u) - (d^\text{before}_u + b^\text{before,after}_u) + \hat{b},
\]

where \( \hat{b} \) is a bias for a pair of masks, \( b^\text{before,after} \) is a bias for a class \( (c, c \in C) \), and \( C \) is a set of image-level label. We add the class bias \( b^\text{before,after} \) for missing classes between \( m^\text{before} \) and \( m^\text{after} \). In practice, we select missing classes after applying CRF based on change ratio of pixels. We denote a number of pixels belonging to class \( c \) on \( m^\text{before} \) as \( N^\text{before}_c \) and a number of pixels belonging to class \( c \) on \( m^\text{after} \) as \( N^\text{after}_c \). The change ratio of pixels belonging to class \( c \) is calculated by:

\[
\delta^\text{before,after}_c = \frac{N^\text{after}_c - N^\text{before}_c}{N^\text{before}_c}.
\]

The class bias \( b_c \) is represented using the change ratio of pixels \( \delta_c^\text{before,after} \) as follows: if \( \delta_c^\text{before,after} > th \) then \( b^\text{before,after}_c = b_\delta \), else \( b^\text{before,after}_c = 0 \), where \( b_\delta \) is a bias for missing classes and we simply set the threshold \( th \) to 0.5.

Finally, we obtain the refined mask \( m^\text{after} \) using the confident score \( w \) by:

\[
m^\text{after}_u = \begin{cases} 
m^\text{before}_u & \text{if } (w^\text{before,after}_u < 0) \\
m^\text{after}_u & \text{if } (w^\text{before,after}_u > 0) 
\end{cases}
\]

For simplicity, we denote this region refinement processing by \( m^\text{after} = \text{SSDD}(e(x; \theta_e), m^\text{before}, m^\text{after}; \theta_d) \).

4. Introducing SSDD modules into the processing flow of WSS

Seed mask generation stage with static region refinement Pixel-level Semantic Affinity (PSA) [1] is a method to propagate label responses to nearby areas which belong to the same semantic entity. In this section, we refine the outputs of CRF in PSA by the proposed Self-Supervised Difference Detection (SSDD) module.

We denote an input image as \( x \) and the probability maps obtained by PSA as \( p^\text{PSA}(x; \theta_P) \) and its CRF results as \( p^\text{crf} \). We convert the probability maps \( (p^\text{PSA}, p^\text{crf}) \) to segmentation masks by \( (m^\text{PSA} = \text{arg max } p^\text{PSA}_k, m^\text{crf} = \text{arg max } p^\text{crf}_k) \), where \( k \in C^\text{PSA}, C^\text{crf} \) by \( \arg \) indicates a background class. The difference detection result for target \( M^\text{PSA,crf} \) is computed by \( d^\text{PSA} = f(e(x; \theta_\delta), e'(x; \theta_\delta), \hat{m}; \theta_\delta) \), where \( \delta \in \mathbb{R}^H \times W \). From the difference detection results, we obtain a target refined mask \( m^\text{after} \) by the proposed SSDD module as follows: \( m^\text{after} = \text{SSDD}(e(x; \theta_e), m^\text{PSA}, m^\text{crf}; \theta_d) \).

Suppose that \( \sigma(\cdot) \) is sigmoid function and \( S \) is a set of locations. The loss function for a Difference Detection net-
wort (DD-Net) is given by:

\[
\mathcal{L}_{\text{change}}(d^{\text{psa}}, d^{\text{df}}, M^{\text{psa,crf0}}) = \frac{1}{|S|} \sum_{u \in S} (J(M^{\text{psa,crf}}, d^{\text{psa}}, u) + J(M^{\text{psa,crf}}, d^{\text{df}}, u)),
\]

(5)

\[
J(M^{\text{psa,crf}}, d^{\text{psa}}, u) = M^{\text{psa,crf}} \log \sigma(d^{\text{psa}}) + (1 - M^{\text{psa,crf}}) \log(1 - \sigma(d^{\text{psa}})),
\]

\[
J(M^{\text{psa,crf}}, d^{\text{df}}, u) = M^{\text{psa,crf}} \log \sigma(d^{\text{df}}) + (1 - M^{\text{psa,crf}}) \log(1 - \sigma(d^{\text{df}})).
\]

Note that \(M^{\text{psa,crf0}}\) corresponds to \(M^{\text{before,after}}\) in Eq.(1).

The proposed method is not effective in the cases that either or both of the segmentation masks has no correct labels. We exclude the bad training samples by simple processing, and define it by the change ratio of pixels \(\delta\). We calculate the change ratio \(\delta_{C}^{\text{psa,crf}}\) in the same manner to Eq.(3) and obtain the decisions \(\gamma_{C}^{\text{psa,crf}}\) for the bad training samples:

- if \(\delta_{C}^{\text{psa,crf}} > 0.5\) then \(\gamma_{C}^{\text{psa,crf}} = 1\) else \(\gamma_{C}^{\text{psa,crf}} = 0\). The interpolated loss function \(\mathcal{L}_{\text{change}}\) as follows: if \(\sum_{k \in C} \gamma_{C}^{\text{psa,crf}} > 0\) then \(\mathcal{L}_{\text{change}} = 0\) else \(\mathcal{L}_{\text{change}} = \mathcal{C}\). \(C\) is a set of category \(C\) for image-level label \(y\).

We also train embedding function \(\theta_{d}\) by training a segmentation network with \(m^{\text{psa}}\) as follows:

\[
\mathcal{L}_{\text{seg}} = -\frac{1}{|S|_{k}} \sum_{k \in C} \sum_{k \in C} \sum_{k \in C} \log(h_{u}^{k}(x; \theta_{d})),
\]

(6)

where \(S_{k}^{C}\) is a set of locations that belong to class \(k\) on the mask \(m^{\text{seg}}\). Final loss function for static region refinement using difference detection is given by:

\[
\mathcal{L}_{\text{static}} = \mathcal{L}_{\text{seg}} + \mathcal{L}_{\text{change}}
\]

(7)

Training stage of a fully-supervised segmentation model with dynamic region refinement In this work, we propose a novel approach to constrain interpolation of the seed labels during training of a segmentation model. The idea of the constraint is to limit interpolation of seed labels to only predictable regions of the difference between newly generated pixel-level labels and seed labels.

In the first step, for an input image \(x\), we obtain outputs of a segmentation model \(p^{\text{seg}} = g(x; \theta_{s})\) and its CRF outputs \(p^{\text{df}}\), where \(g\) is a function of the segmentation model. We convert them to segmentation masks \(m^{\text{seg}} = \arg \max_{k \in C} p^{\text{seg}}_{k}\).

\(m^{\text{df}} = \arg \max_{k \in C} p^{\text{df}}_{k}\). Then, we obtain refined pixel-level labels \(m^{\text{df}}\) by applying the proposed refinement method as follows:

\(m^{\text{df}} = \text{SSDD}(e(x; \theta_{s1}), m^{\text{seg}}; \theta_{s1})\). In the second step, we apply the proposed method to the obtained mask \(m^{\text{df}}\) and seed labels \(m^{\text{seg}}\). The further refined mask \(m^{\text{df}}\) is obtained by \(m^{\text{df}} = \text{SSDD}(e(x; \theta_{s2}), m^{\text{df}}, m^{\text{seg}}; \theta_{s2})\). We generate the mask \(m^{\text{df}}\) in each iteration and train the segmentation model using the generated mask \(m^{\text{df}}\).

We train the semantic segmentation model with the generated mask \(m^{\text{df}}\) by:

\[
\mathcal{L}_{\text{seg-main}} = -\frac{1}{|S|_{k}^{C}} \sum_{k \in C} \sum_{k \in C} \log(h_{u}^{k}(x; \theta_{s1})).
\]

(8)

The difference detection network (DD-net) between \(m^{\text{df}}\) and \(m^{\text{df}}\) is optimized by a below loss:

\[
\mathcal{L}_{\text{diff-ref}}(d^{\text{psa}}, d^{\text{df}}, M^{\text{seg,crf}}) = \frac{1}{|S|} \sum_{u \in S} (J(M^{\text{seg,crf}}, d^{\text{psa}}, u) + J(M^{\text{seg,crf}}, d^{\text{df}}, u)),
\]

(9)

\[
J(M^{\text{seg,crf}}, d^{\text{psa}}, u) = M^{\text{seg,crf}} \log \sigma(d^{\text{psa}}) + (1 - M^{\text{seg,crf}}) \log(1 - \sigma(d^{\text{psa}})),
\]

\[
J(M^{\text{seg,crf}}, d^{\text{df}}, u) = M^{\text{seg,crf}} \log \sigma(d^{\text{df}}) + (1 - M^{\text{seg,crf}}) \log(1 - \sigma(d^{\text{df}})).
\]

In the similar manner to Sec.4, in the second stage, we also exclude bad samples based on the change ratio of pixels. We define the interpolated loss function \(\mathcal{L}_{\text{df-ref}}\) for the pair of masks \((m^{\text{df}}, m^{\text{df}})\) by applying below processing. If \(\sum_{k \in C} \gamma_{C}^{\text{psa,crf}} > 0\) then \(\mathcal{L}_{\text{df-ref}} = 0\) else \(\mathcal{L}_{\text{df-ref}} = \mathcal{L}_{\text{df-ref}}\).

We explain about the training of the DD-net for \((m^{\text{df}}, m^{\text{df}})\). The difference detection between \((m^{\text{df}}, m^{\text{df}})\) is an easy task because the seed masks \(m^{\text{seg}}\) are always constant during training. To avoid the problem, we add a new segmentation branch to a segmentation network, which is also trained with the pixel-level seed labels \(m^{\text{df}}\). We obtain segmentation probability maps from this branch \(p^{\text{seg}}\) and convert the map to mask by \(m^{\text{seg}} = \arg \max_{k \in C} p^{\text{seg}}_{k}\). We train the difference detection network, DD-net, to predict the difference of the pair of masks \((m^{\text{df}}, m^{\text{df}})\) and \((m^{\text{df}}, m^{\text{df}})\).

This replacing of the mask makes it hard to predict the difference between \(m^{\text{df}}\) and \(m^{\text{df}}\) because the situation of training and inference become different. We denote the loss function for the difference detection in this case as follows:

\[
\mathcal{L}_{\text{df-ref}}(d^{\text{psa}}, d^{\text{df}}, M^{\text{df,ref}}) = \frac{1}{|S|} \sum_{u \in S} (J(M^{\text{df,ref}}, d^{\text{psa}}, u) + J(M^{\text{df,ref}}, d^{\text{df}}, u)),
\]

(10)

\[
J(M^{\text{df,ref}}, d^{\text{psa}}, u) = M^{\text{df,ref}} \log \sigma(d^{\text{psa}}) + (1 - M^{\text{df,ref}}) \log(1 - \sigma(d^{\text{psa}})),
\]

\[
J(M^{\text{df,ref}}, d^{\text{df}}, u) = M^{\text{df,ref}} \log \sigma(d^{\text{df}}) + (1 - M^{\text{df,ref}}) \log(1 - \sigma(d^{\text{df}}))).
\]

The parameter \(\theta_{s1}\) of the new segmentation branch is trained by:

\[
\mathcal{L}_{\text{seg-ref}} = \alpha \mathcal{L}_{\text{df-ref}} + (1 - \alpha) \mathcal{L}_{\text{df-ref}};
\]

\[
\mathcal{L}_{\text{seg-ref}} = -\frac{1}{|S|} \sum_{k \in C} \sum_{k \in C} \sum_{k \in C} \log(h_{u}^{k}(x; \theta_{s1})).
\]

\[
\mathcal{L}_{\text{se}} = -\sum_{k \in C} \sum_{k \in C} \sum_{k \in C} \log(h_{u}^{k}(x; \theta_{s1})).
\]

\(\alpha\) is a hyper parameter of the mixing ratio of \(\mathcal{L}_{\text{df-ref}}\) and \(\mathcal{L}_{\text{df-ref}}\).

The final loss function of the proposed dynamic region refinement method is calculated as below:

\[
\mathcal{L}_{\text{dynamic}} = \mathcal{L}_{\text{diff-ref}} + \mathcal{L}_{\text{df-ref}} + \mathcal{L}_{\text{seg-main}} + \mathcal{L}_{\text{seg-ref}},
\]

(12)

75
5. Experiments

We evaluated the proposed methods using the PASCAL VOC 2012 data. For calculating the mean IoU on val and test sets, We used the official evaluation server.

Implementation details Our experiments are heavily based on previous work [1]. For the generating results of PSA, we used implementations provided by the authors that publicly available. We followed the paper [1] and set hyperparameters that achieved the best performance in their paper. As segmentation model we used a ResNet-38 model, which is almost the same to the architecture used in [1] except for upsampling rate. We explore good hyperparameters in the proposed method by grid search.

Comparison Table 1 shows the comparison of dynamic region refinement method with PSA. We denote the dynamic region refinement as “SSDD” in all the tables. We observe that the proposed method outperforms PSA with over 3.2 point margin. In Table 1, we show the gains between the proposed method and PSA for detailed analysis as well.

Table 2 shows the results of the proposed method and recent weakly supervised segmentation methods that use no additional supervisions on PASCAL VOC 2012 validation data and PASCAL VOC 2012 test data. We observe that our method achieves the highest score compared with all the existing methods, which use the same types of supervision [14, 13, 22, 19, 10, 9, 18, 4, 1].

Table 3 shows the comparison of the proposed method with some weakly supervised segmentation methods, which employ relatively cheap additional information. Though completely fair comparisons for them are difficult because of the difference of network model, augmentation technique, the number of iteration epochs and so on, the proposed method demonstrates comparable or better performance without any additional information for training. Fig.4 shows the examples of difference detection results and Fig.5 shows the examples of semantic segmentation results.

6. Conclusion

In this paper, we proposed a novel method to refine a segmentation mask from a pair of segmentation masks before and after refinement process such as CRF by the proposed SSDD module. We demonstrated the proposed method can be used effectively in two stages: static region refinement in the seed generation stage, and dynamic region refinement in the training stage.

Acknowledgments This work was supported by JSPS KAKENHI Grant Number 17J10261, 15H05915, 17H01745, 19H04929 and 17H06100.
References


