PSDPM: Prototype-based Secondary Discriminative Pixels Mining for Weakly Supervised Semantic Segmentation

Xinqiao Zhao[1,2,*] Ziqian Yang[1,2,*] Tianhong Dai[3] Bingfeng Zhang[4] Jimin Xiao[1,†]

[1]XJTLU [2]University of Liverpool [3]University of Aberdeen [4]China University of Petroleum

Abstract

*Image-level **W**eakly **S**upervised **S**emantic **S**egmentation (WSSS) has received increasing attention due to its low annotation cost. **C**lass **A**ctivation **M**apping (CAM) generated through classifier weights in WSSS inevitably ignores certain useful cues, while the CAM generated through class prototypes can alleviate that. However, because of the different goals of image classification and semantic segmentation, the class prototypes still focus on activating primary discriminative pixels learned from classification loss, leading to incomplete CAM. In this paper, we propose a plug-and-play **P**rototype-based **S**econdary **D**iscriminative **P**ixels **M**ining (PSDPM) framework for enabling class prototypes to activate more secondary discriminative pixels, thus generating a more complete CAM. Specifically, we introduce a **F**oreground **P**ixel **E**stimation **M**odule (FPEM) for estimating potential foreground pixels based on the correlations between primary and secondary discriminative pixels and the semantic segmentation results of baseline methods. Then, we enable WSSS model to learn discriminative features from secondary discriminative pixels through a consistency loss calculated between FPEM result and class-prototype CAM. Experimental results show that our PSDPM improves various baseline methods significantly and achieves new state-of-the-art performances on WSSS benchmarks. Codes are available at https://github.com/xinqiaozhao/PSDPM.*

1. Introduction

Deep learning technology shows impressive performances on image [3, 18, 57] and video segmentation [56, 60] tasks. Among them, image-level weakly supervised semantic segmentation has received increasing attention due to its low annotation cost. Existing methods mainly rely on **C**lass **A**ctivation **M**apping (CAM) and knowledge from CLIP [16] to obtain pseudo labels for training semantic segmentation models [1]. Considering the fact that CAM based on classifier weights only focuses on limited regions as the classifier weights deviate from the center of foreground features and can only activate foreground pixels around it [5], many works intend to acquire a more complete CAM by leveraging class prototypes. Chen *et al.* [7] use the cluster centers of each class pixel-wise features as the class prototypes to replace the classifier weights to generate CAMs. Similarly, Chen *et al.* [5] propose an image-specific prototype CAM generation method, which fuses class discriminative features learned by classifier weight with other image-specific information, mitigating the activation center shifting issues and completing the CAM.

However, the aforementioned prototype-based methods still remain shortcomings. Since the pixel-wise feature distribution of each class in one image is still very sparse (*e.g.*, t-SNE visualization results in Fig. 1(b2)), only foreground features close to the class prototype can be activated with high confidence, while those far away from the class prototype cannot be activated. This observation explains the reason that the class-prototype CAMs are still incomplete for certain images (*e.g.*, SIPE CAMs in Fig. 1(b1)).

One reason behind the sparse distribution of foreground pixel-wise features is that the discriminative features in certain foreground pixels are not learned by WSSS model, and the extracted features from these pixels fail to exhibit close proximity to the discriminative prototype which originates from the classifier weights of WSSS model. Specifically, current prototype-based works [5, 7] concentrate on fusing features acquired from classifier weights with image-specific features, and the learning of their WSSS models are still driven by classification loss. However, WSSS model trained through classification loss primarily captures and learns from parts of foreground pixels which are most discriminative and enough for reducing the classification loss to the converge point, while ignoring the other foreground pixels. In this paper, we refer to the discriminative pixels captured and learned by WSSS model through classification loss as *primary* discriminative pixels. In contrast, the remaining discriminative pixels which are not captured by WSSS model are referred to as *secondary* discriminative pixels. Since current prototype-based approaches do not en-

∗:Equal contributions †:Corresponding author

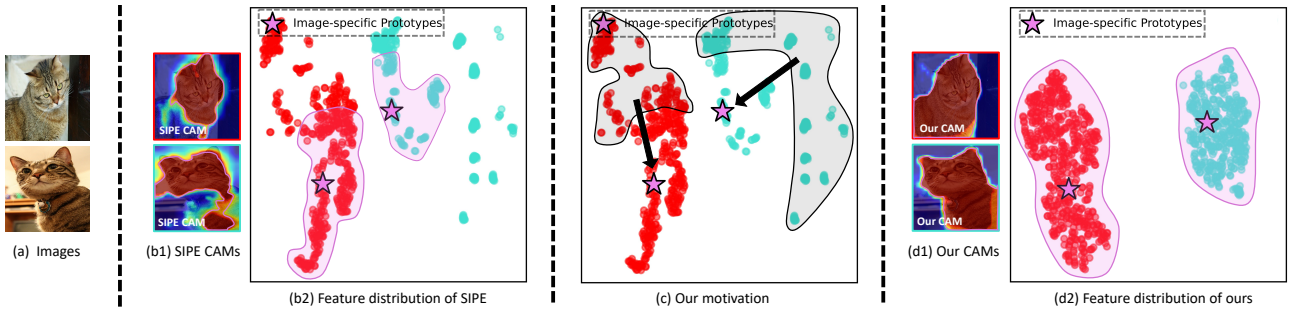

Figure 1. Image-specific prototype CAMs and pixel-wise foreground feature distribution comparisons. Prototype-based method SIPE [5] is chosen as an example and the feature distribution is visualized through t-SNE [34]. (c) depicts our motivation that compelling features extracted from estimated secondary discriminative pixels to be closely align with image-specific class prototype. Compared with the SIPE result, our method achieves a more compact foreground feature distribution, with the image-specific prototype being the distribution centroid. This makes the prototype from our method activate a more complete CAM.

able WSSS model to capture *secondary* discriminative pixels and do not make its classifier learn the discriminative features in *secondary* discriminative pixels, the features extracted from *secondary* discriminative pixels are far away from the class prototypes which originates from the classifier of WSSS model. Consequently, the foreground pixel-wise feature distribution becomes sparse, leading to incomplete prototype CAM. If WSSS model can learn discriminative features from both *primary* and *secondary* discriminative pixels, the pixel-wise feature distribution of foreground pixels will become more compact towards class prototype, leading to a more complete CAM prediction.

Considering the challenges described above, constraining the features extracted from all foreground pixels to closely align with discriminative class prototype could be a solution, as shown in Fig. 1(c). This is because, to satisfy the constraint, WSSS model has to learn discriminative features not only from *primary* discriminative foreground pixels but also from *secondary* discriminative foreground pixels. However, it is infeasible to acquire all foreground pixels for each image in advance during WSSS model training. To address this, we introduce a plug-and-play **P**rototype-based **S**econdary **D**iscriminative **P**ixels **M**ining (PSDPM) framework, which can estimate the potential foreground pixels and optimize WSSS model to learn compact-distributed discrimiantive features from both *primary* and *secondary* discriminative foreground pixels, improving various existing WSSS methods significantly. Specifically, a **F**oreground **P**ixel **E**stimation **M**odule (FPEM) is first proposed to estimate potential foreground pixels by leveraging the correlations between *primary* and *secondary* discriminative features, based on the semantic segmentation results of baseline WSSS methods. Then, we optimize the features within potential foreground pixels to closely align with class prototype, driving WSSS model to learn discriminative features from *secondary* discriminative pixels. As can be found in Fig. 1(d1) and Fig. 1(d2), through our PSDPM, a compact foreground pixel-wise feature distribution can be achieved, and the prototype can thus activate a complete CAM. The contributions of this work include:

- We first argue that WSSS model only captures and learns from primary discriminative pixels under current prototype-based WSSS methods because of the different goals of image classification and semantic segmentation. This causes the features extracted from secondary discriminative pixels far away from the discriminative prototype in feature space, leading to incomplete CAMs.
- We propose a plug-and-play **P**rototype-based **S**econdary **D**iscriminative **P**ixels **M**ining (PSDPM) framework for image-specific prototype CAM generation, which can effectively encourage WSSS model to learn from secondary discriminative pixels and significantly improve the performance of various existing WSSS methods.
- Our method achieves new state-of-the-art WSSS performances with only image-level labels on two benchmarks, reaching 74.1% and 74.9% mIoU on the validation and test sets of Pascal VOC 2012, respectively, and 47.2% mIoU on MS COCO 2014 validation set.

2. Related Work

2.1. Image-level Weakly Supervised Semantic Segmentation

Image-level **W**eakly **S**upervised **S**emantic **S**egmentation (WSSS) has received increasing attention due to its low annotation cost [33, 36, 37, 55]. The key step of WSSS is to generate high-quality CAMs for pseudo-labels [32, 36, 40, 48]. Attention mapping [30, 42, 51], heuristic approaches (*e.g.,* erasing and accumulation) [17, 48, 54] have been proposed to encourage the network to explore novel regions rather than solely focusing on limited discriminative regions. Moreover, other strategies include self-supervised learning [5, 42], contrastive learning [11, 59], and cross-image information [27, 38, 47] have been used to generate

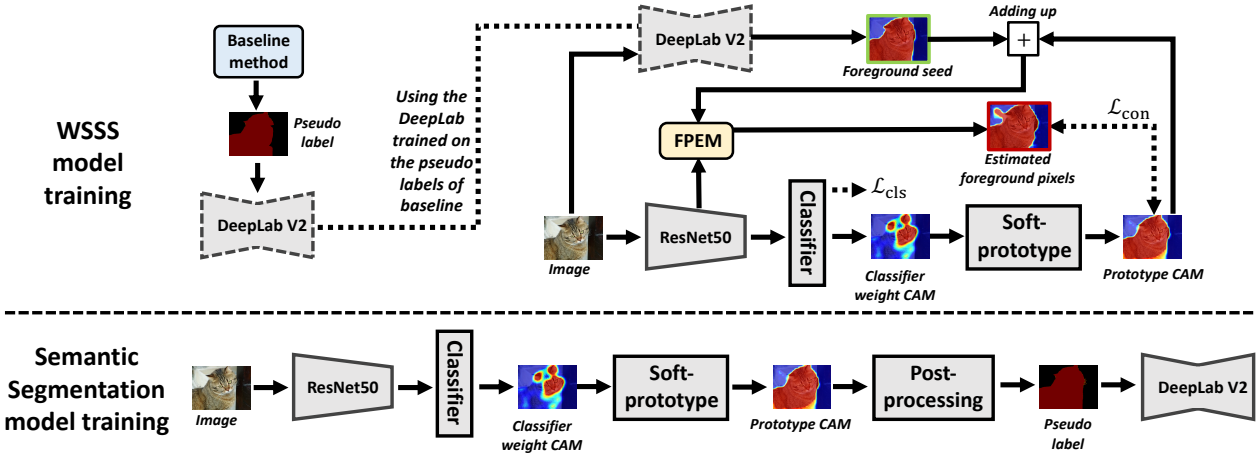

Figure 2. The training pipeline of our proposed PSDPM. For WSSS model training, we first train a DeepLab semantic segmentation model using the pseudo labels provided by baseline WSSS method. Then, the DeepLab model is used for calculating a consistency loss $\mathcal{L}_{\text{con}}$ between our FPEM refined CAM and prototype CAM calculated from classifier weight CAM. IRN [1] or DenseCRF [20] is adopted as post-processing methods during semantic segmentation model training.

accurate CAMs. To further improve the quality of the generated CAMs, CRF [20] and IRN [1] are employed to refine the rough boundary of the CAMs to produce final enhanced pseudo labels, which are then used for semantic segmentation model training.

2.2. Class Prototype in WSSS

A series of class prototype methods have been proposed [7, 41]. From the view of optimization, Du *et al.* [11] propose a pixel-to-prototype contrast that estimates the prototypes from pixel-wise feature embeddings across the training batch, obtaining a more suitable feature representation for semantic segmentation. From the view of CAM calculation, Chen *et al.* [7] perform clustering on all pixel-wise features of an object class and selects the cluster centers as class prototypes. Then, similarity matrices are derived by comparing all pixel-wise features to every prototype for CAM generation. Besides, Chen *et al.* [5] propose a prototype-based image-specific CAM. The prototype is first calculated through masked average pooling of hierarchical extracted features and then used to activate class-specific regions. However, these prototype-based methods still generate incomplete CAMs, as illustrated in Sec. 1. This paper first analyzes the reason behind this issue and then proposes a prototype-based secondary discriminative pixels mining method that significantly improves the performances of current state-of-the-art WSSS method.

2.3. Self-attention in WSSS

Self-attention is an efficient module to capture context information and refine pixel-wise prediction results [2, 35]. Its effectiveness in WSSS has been verified in [13]. The general idea behind the self-attention mechanism is recalculating each feature representation based on all other features through the dot-product pixel affinity in an embedding space [39]. It revises feature maps by capturing context feature dependency, which also meets the ideas of most WSSS methods using the similarity of pixels to refine the original activation map. Inspired by the self-attention mechanism, Wang *et al.* [42] first introduce a Pixel Correlation Module (PCM) at the end of the network to integrate the low-level feature of each pixel for WSSS. This method calculates the CAM affinity only using the high-level feature and image RGB information. However, other different level information (*e.g.*, feature position, various levels of features) are not fully considered in PCM, restricting its performance improvement in WSSS task. Different from the previous work PCM [42] which is a pure context refinement module and aims at refining final CAMs, in this work, we adopt self-attention mechanism to continuously estimate potential foreground pixels for driving WSSS model to learn from secondary discriminative pixels. Additionally, we also introduce a cascade fashion in our FPEM to leverage all kinds of low-level information and improve potential foreground pixels estimation accuracy.

3. Methodology

For enabling WSSS model to learn discriminative features from both primary and secondary discriminative pixels, and simultaneously rectifying the CAM activation center through class prototypes, our main idea is to first estimate potential foreground regions which contain primary and secondary discriminative pixels, and then ensure all features from estimated foreground pixels are aligned as closely as possible with the discriminative prototype. This

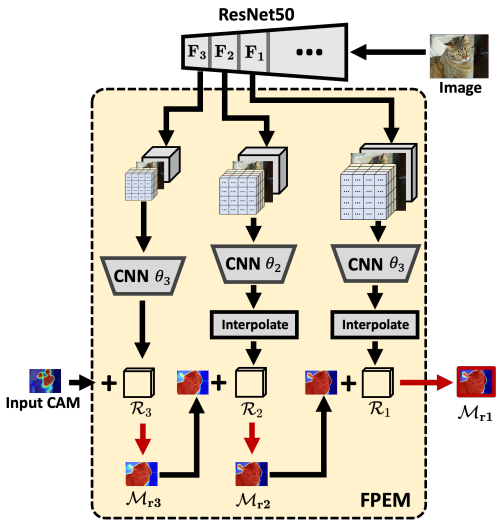

Figure 3. The calculation flow of our **F**oreground **P**ixel **E**stimation **M**odule (FPEM). The position matrix, RGB image, and extracted features are first concatenated, and then embedded through three 1×1 Convolutional Neural Networks (CNNs), respectively, following Eq. (1). Finally, three embedded results are processed in turn through a cascade manner. The red arrow denotes the calculation formulated in Eq. (2).

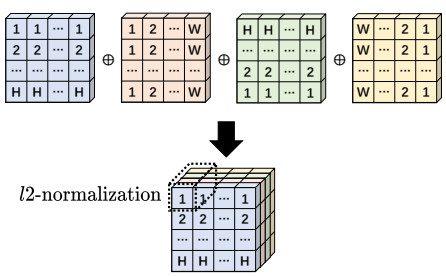

Figure 4. The illustration of position matrix calculation. First, four height- or width-based incremented matrixes are created. Then, four matrixes are concatenated and $\ell 2$-normalization is applied on the concatenated dimension for each position vector. In this way, closer pixels have higher cosine similarity on our position matrix and vice versa.

makes WSSS model have to learn discriminative features from both primary and secondary discriminative pixels, resulting in a more compact feature distribution of foreground pixels and a more complete CAM. The pipeline of our **P**rototype-based **S**econdary **D**iscriminative **P**ixels **M**ining (PSDPM) is depicted in Fig. 2. It involves a **F**oreground **P**ixel **E**stimation **M**odule (FPEM) with a consistency loss $\mathcal{L}_{\text{con}}$ calculated between the prototype CAM and the FPEM output. The following sub-sections will describe each component in details.

3.1. Foreground Pixel Estimation Module (FPEM)

Chen *et al.* [5] show that CAM refinement methods (*e.g.*, DenseCRF [20], IRN [1]) substantially improve the quality of CAM which considers discriminative features only from primary discriminative pixels. These methods activate partial secondary discriminative pixels and make CAMs more complete by leveraging the correlations in low-level information among different pixels. This indicates that primary and secondary discriminative pixels exhibit strong correlations in low-level information, such as RGB values and spatial positions. However, primary discrminative pixels do not only correlate with secondary discrminative pixels, they could also have low-level correlations with the non-discriminative ones. On the other hand, the low-level correlations become fewer when the encoder network goes deeper, as the high-level information dominates the deeper feature map [49]. In this case, the pixel correlations are mainly retained through discriminative features related features, making the remained correlations less related to non-discriminative pixels, compared with low-level correlations. Inspired from these findings, in order to estimate foreground pixels (including primary and secondary discriminative pixels) more accurately only based on the learned primary discriminative pixels, we leverage the correlations between primary and secondary discriminative pixels with varying confidence levels. Specifically, consider an input CAM that has already activated primary discriminative pixels, we adopt our **F**oreground **P**ixel **E**stimation **M**odule (FPEM) to make CAM activate more potential secondary discriminative pixels and regard the expanded CAM as potential foreground pixels. This module introduces the most confident correlations (*i.e.*, the correlations are not related to non-discriminative pixels) in the deepest feature map first to refine the input CAM in general and then gradually introduces less confident correlations to refine the CAM details, minimizing the total refinement error.

The detailed procedure of our FPEM is shown in Fig. 3. Specifically, we select the feature maps of the last three layers in the feature encoder (*i.e.*, ResNet50 in this paper) and then concatenate each feature map with a position matrix and a scaled input RGB image, providing three combined representations, each with RGB, position, and feature information. For example, concerning the last layer feature map $\mathbf{F}_3$ (supposing its size is $32\times32\times1024$ and 1024 is its feature map channel size), we interpolate the original RGB image to the size of $32\times32\times3$ through bilinear interpolation and create a position matrix whose size is $32\times32\times2$. The position matrix is created following the illustration shown in Fig. 4. Each position is represented by one four-dimensional vector and closer pixels will have high cosine similarity, fully preserving relative position information when doing self-attention calculation. Then, the feature map, interpolated image, and position matrix are

concatenated on the third channel dimension, obtaining a combined representation. After obtaining three combined representations for $\mathbf{F}_3$, $\mathbf{F}_2$, and $\mathbf{F}_1$, we embed them through three **C**onvolutional **N**eural **N**etworks (CNNs) (*i.e.*, θ_3, θ_2, and θ_1) and interpolate the embedded results to the size of input CAM through bilinear interpolation, obtaining three correlation maps $\mathcal{R}_3$, $\mathcal{R}_2$ and $\mathcal{R}_1$. Their formulations are presented through Eq. (1):

$$\mathcal{R}_i = \theta_i(\mathbf{POS}_i \oplus \mathbf{F}_i \oplus \mathbf{I}_i), \tag{1}$$

where $\mathbf{POS}_i$ denotes the position matrix with the same size as feature map $\mathbf{F}_i$; $\mathbf{I}_i$ denotes the bilinear interpolated image also with the same size as feature map $\mathbf{F}_i$; θ_i denotes individual 1×1 convolutional layer as shown in Fig. 3.

After having three correlation maps, we first calculate a revised CAM $\mathcal{M}_{\mathbf{r3}}$ based on the input CAM and the most confident correlation map $\mathcal{R}_3$ ($\mathcal{R}_3$ originates from the deepest feature map $\mathbf{F}_3$ and is thus most confident), as follows:

$$\mathcal{M}_i = \Big(\big(\frac{1}{\mathcal{C}_i} \sum\nolimits_{\forall j} \mathrm{ReLU}\big(\frac{(\mathcal{R}_n)_i^{\mathrm{T}} (\mathcal{R}_n)_j}{\|(\mathcal{R}_n)_i\| \cdot \|(\mathcal{R}_n)_j\|} \big) \hat{\mathcal{M}}_i \big) + \hat{\mathcal{M}}_i \Big) / 2, \tag{2}$$

where $(\mathcal{R}_n)_i$ denotes the correlation map feature value of $\mathcal{R}_n$ at position i, which is a vector; j denotes any position in the CAM; $\mathcal{C}_i$ denotes the coefficient sum $\sum_{\forall j} \mathrm{ReLU}(\frac{(\mathcal{R}_n)_i^{\mathrm{T}} (\mathcal{R}_n)_j}{\|(\mathcal{R}_n)_i\| \cdot \|(\mathcal{R}_n)_j\|})$; $\hat{\mathcal{M}}_i$ denotes the input CAM value at position i; $\mathcal{M}_i$ denotes the revised CAM value at position i, which is the average of two terms: weighted sum of input CAM $\hat{\mathcal{M}}_i$ with normalized similarities calculated from $\mathcal{R}_n$ and $\hat{\mathcal{M}}_i$. We adopt a residual way when calculating $\mathcal{M}_i$ from $\hat{\mathcal{M}}_i$ to prevent the final result of our cascade-style FPEM from being too wrong.

After that, the second revised CAM $\mathcal{M}_{\mathbf{r2}}$ is calculated through Eq. (2) based on the correlation map $\mathcal{R}_2$ and the first revised CAM $\mathcal{M}_{\mathbf{r3}}$ as input CAM. Similarly, the third revised CAM $\mathcal{M}_{\mathbf{r1}}$ is calculated based on the correlation map $\mathcal{R}_1$ and the second revised CAM $\mathcal{M}_{\mathbf{r2}}$ as input CAM. In this way, the most confident correlation map $\mathcal{R}_3$ which originates from the deepest feature map is first considered to give a coarse CAM refinement result. Then, based on this coarse CAM refinement result, lower-confident correlation maps with higher resolutions (*i.e.*, $\mathcal{R}_2$ and $\mathcal{R}_1$) which originate from shallower feature maps are gradually included for further adjusting the details of the coarse CAM refinement result, and the third revised CAM $\mathcal{M}_{\mathbf{r1}}$ is regarded as final estimated potential foreground pixels.

3.2. Plug-and-play Prototype-based Training

On the one hand, for each training image, we calculate image-specific prototype CAM based on classifier weight CAM. Specifically, the soft prototype $\mathbf{P}_c$ of class c is calculated through Eq. (3):

$$\mathbf{P}_c = \mathrm{MAP}\big((\mathcal{M}_{\mathbf{W}})_c \odot L(\mathbf{F_1}, \mathbf{F_2}, \mathbf{F_3}) \big), \tag{3}$$

where $\mathcal{M}_{\mathbf{W}}$ denotes the classifier weight CAM which is calculated through 1×1 convolutional kernels; $\mathrm{MAP}(\cdot)$ denotes masked average pooling; $L(\cdot)$ denotes the linear projection; $\mathbf{F_1}, \mathbf{F_2}, \mathbf{F_3}$ are the last three layer feature maps of feature encoder as shown in Fig. 3. Then, we calculate prototype CAM based on soft prototype $\mathbf{P}_c$ through Eq. (4):

$$(\mathcal{M}_{\mathbf{P}})_c = \mathrm{ReLU}\big(\cos \langle \mathbf{P}_c, L(\mathbf{F_1}, \mathbf{F_2}, \mathbf{F_3}) \rangle \big), \tag{4}$$

where $\mathcal{M}_{\mathbf{P}}$ denotes the prototype CAM; $\cos\langle \cdot, \cdot \rangle$ denotes the cosine similarity between the two terms within it. With our prototype CAM, as the discriminative classifier weights are fused with image-specific information for CAM activation, the CAM activation center can be rectified based on the characteristics of input images.

On the other hand, each training image is also input to a DeepLab V2 model for calculating semantic segmentation probability maps. This DeepLab model is trained on the pseudo-labels provided by the baseline WSSS method, and the output semantic segmentation probability map is denoted as $\mathcal{M}_D$ to represent the potential foreground seed pixels. The quality of seed pixels depends on the performance of baseline WSSS method. The seed pixels are mainly composed of primary discriminative pixels as the baseline method is unable to learn discrminative features from secondary discriminative pixels. Then, the semantic segmentation probability map $\mathcal{M}_D$ and our prototype CAM $\mathcal{M}_{\mathbf{P}}$ are added as the input to our FPEM module for further expanding CAM with more secondary discriminative pixels.

Finally, a consistency loss $\mathcal{L}_{\text{con}}$ is used to minimize the difference between the FPEM output and prototype CAM $\mathcal{M}_{\mathbf{P}}$ through Eq. (5).

$$\mathcal{L}_{\text{con}} = \sum_c \Big\| \phi\Big(\big((\mathcal{M}_D)_c + (\mathcal{M}_{\mathbf{P}})_c \big) / 2 \Big) - (\mathcal{M}_{\mathbf{P}})_c \Big\|_1, \tag{5}$$

where $(\mathcal{M}_D)_c$ denotes the DeepLab V2 output of class c, which is the potential foreground seed pixels; $\phi(\cdot)$ denotes our FPEM module.

Here, FPEM aims to expand the seed pixels $\mathcal{M}_D$ which mainly contain primary discriminative pixels, to include more additional secondary discriminative pixels, providing potential foreground pixels (*i.e.,* primary and secondary discriminative pixels) to guide the learning of prototype CAM. As the prototype CAM originates from discriminative classifier weights, when narrowing the gap between prototype CAM and FPEM estimated foreground pixels through minimizing $\mathcal{L}_{\text{con}}$, WSSS model is compelled to learn all discriminative features from FPEM estimated foreground pixels as much as possible to make class prototype activate the same pixels as FPEM estimated. In this way, WSSS model can learn discriminative features from both primary and secondary discriminative pixels.

As WSSS model continues to learn increasing numbers

of discriminative features from the secondary discriminative pixels during training, the prototype CAM becomes more and more complete. Considering this, the prototype CAM is combined with the output of DeepLab V2 as the input of FPEM, which encourages FPEM to continuously explore more secondary discriminative foreground pixels beyond the output of DeepLab V2.

In summary, the total WSSS model training loss $\mathcal{L}_{\text{total}}$ of our PSDPM is formulated as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{con}}, \tag{6}$$

where $\mathcal{L}_{\text{cls}}$ is the multi-label soft margin classification loss [5] calculated from classifier output as shown in Fig. 2.

3.3. Semantic Segmentation Model Training

As WSSS model has learned sufficient discriminative features from both primary and secondary discriminative pixels, our prototype CAM can cover complete foreground pixels, with CAM activation center being rectified. However, the resolution of our prototype CAM is relative low compared with the resolution of the original image. Therefore, we adopt post-processing (*e.g.*, IRN [1] or DenseCRF [20]) on our prototype CAMs to refine the activation details, and regard the post-processed prototype CAMs as pseudo labels for semantic segmentation model training.

4. Experiments

4.1. Dataset and Implementation Details

Datasets and Evaluation Metrics. Experiments are conducted on two benchmarks: PASCAL VOC 2012 [12] with 21 classes and MS COCO 2014 [25] with 81 classes. For PASCAL VOC 2012, following [5, 24, 42, 50, 52], we use the augmented SBD set [14] with 10,582 annotated images. **M**ean **I**ntersection **o**ver **U**nion (mIoU) [28] is used to evaluate segmentation results. To evaluate the quality of CAM and pseudo label, CAM and pseudo label are generated for every image in the train set and the mIoUs are calculated based on the ground truth masks. To evaluate the semantic segmentation model performance, we train DeepLab V2 and use it to predict masks for the images in val and test sets, and calculate mIoU based on ground truth masks.

Implementation Details. For WSSS model training (as shown in Fig. 2), we adopt the ImageNet [10] pretrained ResNet50 [15]. SGD optimizer with a momentum of 0.9 and a weight decay of 10^{-4} are adopted. The initial learning rate is set to 0.1 for backbone and 1 for other parameters. The learning rate is then adjusted by poly decay with power of 0.9, following [1, 5]. The WSSS model is trained with a batch size of 16 for Pascal VOC 2012 and MS COCO 2014. The scale ratios of multi-scaled CAM during inference are set to $\{0.5, 1.0, 1.5, 2.0\}$ following [5]. Random cropping size 512×512 is adopted for training data augmentation. Our prototype CAM $\mathcal{M}_{\mathbf{P}}$ is further post-processed by DenseCRF [20] or IRN [1] to generate the final pseudo labels, which are used to train the segmentation model. With regard to the semantic segmentation model, we adopt ResNet101-based DeepLab V2 with weights pretrained on ImageNet dataset. For PASCAL VOC 2012, we follow the default training settings of HSC [43]. Input images are randomly scaled to $\{0.5, 1.5\}$ and cropped to 448×448 for training. The batch size is set to 10, and iteration is 20k. Besides, for MS COCO 2014, we follow the default training settings of CLIP-ES [26]. Input images are randomly scaled to $\{0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$ and 481×481 are cropped. The batch size and the number of iterations are set to 5 and 100k, respectively.

Table 1. Evaluation results of pseudo-label performance (mIoU (%)) on various WSSS methods on PASCAL VOC 2012.

Method	Pseudo label
IRN [1] *CVPR'19*	66.3
SEAM [42] *CVPR'20*	63.6
IRN+CONTA [53] *NeurIPS'20*	67.9
AdvCAM [22] *CVPR'21*	69.9
RIB [21] *NeurIPS'21*	70.6
AMN [23] *CVPR'22*	72.2
SIPE [5] *CVPR'22*	68.0
ESOL [24] *NeurIPS'22*	68.7
CLIMS [44] *CVPR'22*	70.5
CLIMS + `Ours`	75.5 (+5.0)
IRN+LPCAM [7] *CVPR'23*	71.2
IRN+LPCAM + `Ours`	75.1 (+3.9)
CLIP-ES [26] *CVPR'23*	75.0
CLIP-ES + `Ours`	**77.3** (+2.3)

4.2. Comparison of Pseudo-label Quality

To validate the effectiveness of our PSDPM, we evaluate the quality of pseudo label quantitatively in Tab. 1. The pseudo labels on Pascal VOC 2012 are generated through IRN [1] based on the prototype CAMs of our PSDPM. Experimental results show that our PSDPM can consistently improve various existing works on PASCAL VOC. Particularly, through applying our PSDPM on CLIP-ES [26], we improve the pseudo-label performance of CLIP-ES by 2.3 %, achieving new state-of-the-art pseudo-label performance 77.3% on PASCAL VOC 2012.

In addition, we study the qualitative effects of our PSDPM on both CAMs and pseudo labels in Fig. 5. We also compare the differences between our method and another prototype-based method SIPE [5]. Firstly, it can be seen that the prototype-based method SIPE can improve the classifier weight CAM quality by activating more complete foreground pixels. However, for certain images (*e.g.*, 'bird' in the last column and 'person' in the third and fourth

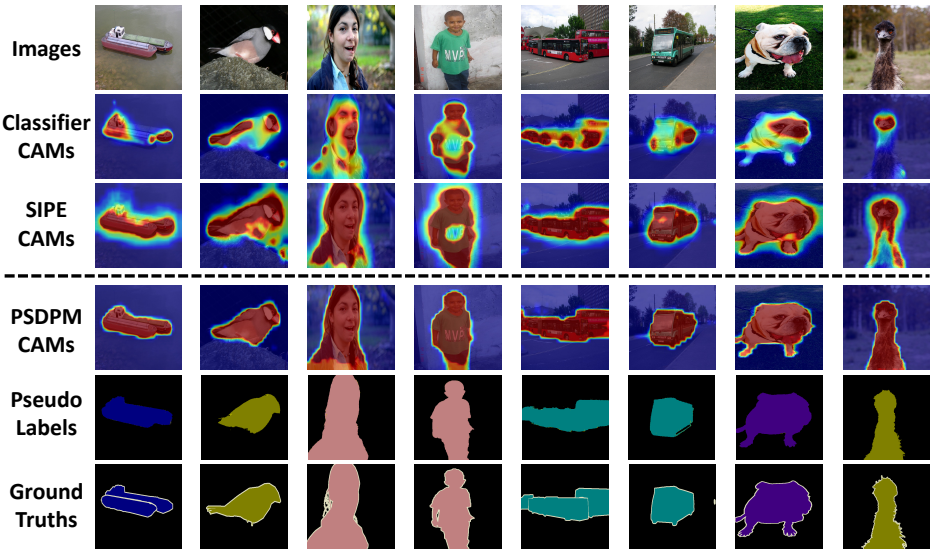

Figure 5. CAM and pseudo-label visualization results on PASCAL VOC 2012.

Table 2. Evaluation results of DeepLab V2 on PASCAL VOC 2012. The best results are in bold.

Methods	PASCAL VOC	
	Val	Test
SEAM [42] *CVPR'20*	64.5	65.7
AdvCAM [22] *CVPR'21*	68.1	68.0
SEAM+PPC [11] *CVPR'22*	67.7	67.4
ReCAM [8] *CVPR'22*	68.5	68.4
SIPE [5] *CVPR'22*	68.8	69.7
AMN [23] *CVPR'22*	70.7	70.6
MCTformer [45] *CVPR'22*	71.9	71.6
ESOL [24] *NeurIPS'22*	69.9	69.3
SAS [19] *AAAI'23*	69.5	70.1
SEAM+OCR [9] *CVPR'23*	67.8	68.4
AMN+LPCAM [7] *CVPR'23*	70.1	70.4
MMCST [46] *CVPR'23*	72.2	72.2
ToCo [31] *CVPR'23*	69.8	70.5
BECO [29] *CVPR'23*	72.1	71.8
AdvCAM+FPR [4] *ICCV'23*	70.3	70.1
SFC [58] *AAAI'24*	71.2	72.5
CLIMS [44] *CVPR'22*	69.3	68.7
CLIMS + `Ours`	73.1 (+3.8)	73.9 (+5.2)
IRN+LPCAM [7] *CVPR'23*	68.6	68.7
IRN+LPCAM + `Ours`	72.3 (+3.7)	73.2 (+4.5)
CLIP-ES [26] *CVPR'23*	71.1	71.4
CLIP-ES + `Ours`	**74.1** (+3.0)	**74.9** (+3.5)

Table 3. Evaluation results of DeepLab V2 on MS COCO 2014. The best results are in bold.

Methods	MS COCO
	Val
SIPE [5] *CVPR'22*	40.6
AMN [23] *CVPR'22*	44.7
MDBA [6] *TIP'23*	37.8
SAS [19] *AAAI'23*	44.8
MMCST [46] *CVPR'23*	45.9
ToCo [31] *CVPR'23*	41.3
SFC [58] *AAAI'24*	46.8
CLIP-ES [26]	45.4
CLIP-ES + `Ours`	**47.2** (+1.8)

columns), the CAM results of SIPE are still incomplete as WSSS model does not capture and learn from secondary discriminative pixels. Besides, for certain image (*e.g.*, bird in the second column), SIPE has an over-activating issue because it adopts a structure-aware seeds locating algorithm for expanding the incomplete prototype CAM, which may expand the CAM in an inaccurate way. Through our PSDPM, as the WSSS model can learn discriminative features from both primary and secondary discriminative pixels, all CAM results are consistently improved in terms of object completeness and boundary fineness.

4.3. Comparison of Final Semantic Segmentation Performance

In this study, IRN [1] (on Pascal VOC 2012) and DenseCRF [20] (on MS COCO 2014) are utilized for post-processing our PSDPM prototype CAMs. Their results are then treated as pseudo labels to train semantic segmentation model DeepLab V2 in a fully supervised manner. Tab. 2 reports the mIoU scores of our method and recent WSSS methods on the validation and test sets of PASCAL VOC 2012. On

Table 4. Ablation study of our PSDPM in WSSS model training on PASCAL VOC 2012 training set and mIoU (%) is used for evaluation. CLIMS [44] is adopted as baseline method for PSDPM. 'D', 'A' and 'F' denote different parts in our WSSS model training (Fig. 2). 'D' denotes using the output of DeepLab V2 which is trained on the pseudo labels of baseline method; 'A' denotes adding up DeepLab V2 output and prototype CAM; 'FPEM' denotes using our FPEM.

#	D	A	FPEM	Prototype CAM
1	✓			59.7
2	✓	✓		61.3
3	✓		✓	67.9
4	✓	✓	✓	**68.5**

this dataset, we achieve 74.1% and 74.9% mIoUs using an ImageNet pre-trained backbone based on CLIP-ES, outperforming all other image-level WSSS methods. Tab. 3 reports the performance comparison on MS COCO 2014. Our method achieves 47.2% mIoU on the validation set, outperforming all other WSSS methods. Additionally, the consistent performance improvements of our PSDPM on various baseline methods and datasets demonstrate that these baseline methods are originally limited to primary discriminative pixels and can benefit from our PSDPM.

4.4. Ablation Studies

In Tab. 4, we first verify the effectiveness of different components of our PSDPM (Fig. 2). By regarding the output of DeepLab trained on the pseudo labels of baseline method as estimated foreground pixels, and calculating $\mathcal{L}_{\text{con}}$ between the estimated foreground pixels and prototype CAM (*i.e.*, setting #1), we achieve 59.7% mIoU. Base on this, adding up Deeplab output and prototype CAM, and regard the sum as estimated foreground pixels for $\mathcal{L}_{\text{con}}$ calculation (*i.e.*, setting #2) can improve the mIoU to 61.3%. This is because prototype CAM can be complementary to the DeepLab output and enable WSSS model to learn new features from $\mathcal{L}_{\text{con}}$. However, WSSS model learning is still limited to the primary discriminative pixels. Further adopting FPEM to the result of 'A' (*i.e.*, 'setting #4') can introduce more secondary discriminative pixels into consideration for our PSDPM, achieving the best performance 68.5%. Using FPEM without adding prototype CAM to DeepLab output (*i.e.*, 'setting #3') makes WSSS model being restrained to the baseline method and cannot continuously mine more secondary discriminative pixels, only achieving 67.9% mIoU. Tab. 5 studies the effectiveness of different steps in FPEM. When using $\mathcal{M}_{\mathbf{r3}}$ for foreground pixel estimation (*i.e.*, 'w/ $\mathcal{M}_{\mathbf{r3}}$'), we can achieve 64.8%, which is better than 'w/o FPEM' which regards the sum of DeepLab output and prototype CAM as estimated foreground pixels. Using $\mathcal{M}_{\mathbf{r2}}$ (*i.e.*, 'w/ $\mathcal{M}_{\mathbf{r2}}$') can continue the performance improvement significantly by 3.2%. Using $\mathcal{M}_{\mathbf{r1}}$ (*i.e.*, 'w/ $\mathcal{M}_{\mathbf{r1}}$') further improves the FPEM effectiveness to 68.5%. Settings #5~#7 show that $\mathbf{POS}$, $\mathbf{F}$, $\mathbf{I}$ in Eq. (1) are all helpful. Tab. 6 shows that simply adding post-processing can not improve the baseline performance much, indicating the improvements brought by our PSDPM on various baselines are not because of post-processing.

Table 5. Ablation study of FPEM on PASCAL VOC 2012 training set and mIoU (%) is used for evaluation. CLIMS [44] is adopted as baseline method for PSDPM. 'w/o FPEM' indicates our PSDPM without FPEM. 'w/ $\mathcal{M}_{\mathbf{r3}}$', 'w/ $\mathcal{M}_{\mathbf{r2}}$' and 'w/ $\mathcal{M}_{\mathbf{r1}}$' indicate regarding the results of FPEM different steps as estimated foreground pixels. 'w/o $\mathbf{POS}$', 'w/o $\mathbf{F}$' and 'w/o $\mathbf{I}$' denote without the corresponding component in Eq. (1).

#	FPEM	Prototype CAM
1	w/o FPEM	61.3
2	w/ $\mathcal{M}_{\mathbf{r3}}$	64.8
3	w/ $\mathcal{M}_{\mathbf{r2}}$	68.0
4	w/ $\mathcal{M}_{\mathbf{r1}}$	**68.5**
5	w/o $\mathbf{POS}$	68.1
6	w/o $\mathbf{F}$	64.2
7	w/o $\mathbf{I}$	68.2

Table 6. The influence of post-processing method on baseline. The result shows that the improvement brought by our PSDPM is not due to post-processing (*e.g.*, IRN).

Methods	Pseudo label
CLIP-ES [26]	75.0
CLIP-ES [26] + IRN [1]	75.1
CLIP-ES + `Ours` (w/ IRN)	**77.3**

5. Conclusion

In this paper, we demonstrate that WSSS model does not learn discriminative features from secondary discriminative pixels in current classification-loss-driven WSSS methods, and this causes the incomplete CAM activation issue. For solving this, we propose a plug-and-play **P**rototype-based **S**econdary **D**iscriminative **P**ixels **M**ining (PSDPM) framework for enabling WSSS model to learn discrminative features from secondary discriminative pixels, improving various existing WSSS methods to new state-of-the-art performances. Other possible solutions for secondary discriminative pixel issue will be investigated in our future works.

Acknowledgement This work was supported by the National Key R&D Program of China (No.2022YFE0200300), the National Natural Science Foundation of China (No. 61972323, 62331003), Suzhou Basic Research Program (SYG202316) and XJTLU REF-22-01-010, XJTLU AI University Research Centre, Jiangsu Province Engineering Research Centre of Data Science and Cognitive Computation at XJTLU and SIP AI innovation platform (YZCXPT2022103).

References

[1] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *CVPR*, 2019. 1, 3, 4, 6, 7, 8

[2] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *CVPR*, 2005. 3

[3] Jinpeng Chen, Runmin Cong, Yuxuan Luo, Horace Ip, and Sam Kwong. Saving 100x storage: Prototype replay for reconstructing training sample distribution in class-incremental semantic segmentation. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[4] Liyi Chen, Chenyang Lei, Ruihuang Li, Shuai Li, Zhaoxiang Zhang, and Lei Zhang. Fpr: False positive rectification for weakly supervised semantic segmentation. In *ICCV*, 2023. 7

[5] Qi Chen, Lingxiao Yang, Jian-Huang Lai, and Xiaohua Xie. Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation. In *CVPR*, 2022. 1, 2, 3, 4, 6, 7

[6] Tao Chen, Yazhou Yao, and Jinhui Tang. Multi-granularity denoising and bidirectional alignment for weakly supervised semantic segmentation. *IEEE TIP*, 2023. 7

[7] Zhaozheng Chen and Qianru Sun. Extracting class activation maps from non-discriminative features as well. In *CVPR*, 2023. 1, 3, 6, 7

[8] Zhaozheng Chen, Tan Wang, Xiongwei Wu, Xian-Sheng Hua, Hanwang Zhang, and Qianru Sun. Class re-activation maps for weakly-supervised semantic segmentation. In *CVPR*, 2022. 7

[9] Zesen Cheng, Pengchong Qiao, Kehan Li, Siheng Li, Pengxu Wei, Xiangyang Ji, Li Yuan, Chang Liu, and Jie Chen. Out-of-candidate rectification for weakly supervised semantic segmentation. In *CVPR*, 2023. 7

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[11] Ye Du, Zehua Fu, Qingjie Liu, and Yunhong Wang. Weakly supervised semantic segmentation by pixel-to-prototype contrast. In *CVPR*, 2022. 2, 3, 7

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 6

[13] Junsong Fan, Zhaoxiang Zhang, Tieniu Tan, Chunfeng Song, and Jun Xiao. Cian: Cross-image affinity net for weakly supervised semantic segmentation. In *AAAI*, 2020. 3

[14] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *ICCV*, 2011. 6

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[16] Tianyu Huang, Bowen Dong, Yunhan Yang, Xiaoshui Huang, Rynson WH Lau, Wanli Ouyang, and Wangmeng Zuo. Clip2point: Transfer clip to point cloud classification with image-depth pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22157–22167, 2023. 1

[17] Peng-Tao Jiang, Ling-Hao Han, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Online attention accumulation for weakly supervised semantic segmentation. *IEEE TPAMI*, 2021. 2

[18] Siyu Jiao, Yunchao Wei, Yaowei Wang, Yao Zhao, and Humphrey Shi. Learning mask-aware clip representations for zero-shot segmentation. *Advances in Neural Information Processing Systems*, 36:35631–35653, 2023. 1

[19] Sangtae Kim, Daeyoung Park, and Byonghyo Shim. Semantic-aware superpixel for weakly supervised semantic segmentation. In *AAAI*, 2023. 7

[20] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NeurIPS*, 2011. 3, 4, 6, 7

[21] Jungbeom Lee, Jooyoung Choi, Jisoo Mok, and Sungroh Yoon. Reducing information bottleneck for weakly supervised semantic segmentation. In *NeurIPS*, 2021. 6

[22] Jungbeom Lee, Eunji Kim, and Sungroh Yoon. Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In *CVPR*, 2021. 6, 7

[23] Minhyun Lee, Dongseob Kim, and Hyunjung Shim. Threshold matters in wsss: manipulating the activation for the robust and accurate segmentation model against thresholds. In *CVPR*, 2022. 6, 7

[24] Jinlong Li, Zequn Jie, Xu Wang, Xiaolin Wei, and Lin Ma. Expansion and shrinkage of localization for weakly-supervised semantic segmentation. In *NeurIPS*, 2022. 6, 7

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[26] Yuqi Lin, Minghao Chen, Wenxiao Wang, Boxi Wu, Ke Li, Binbin Lin, Haifeng Liu, and Xiaofei He. Clip is also an efficient segmenter: A text-driven approach for weakly supervised semantic segmentation. In *CVPR*, 2023. 6, 7, 8

[27] Man Liu, Chunjie Zhang, Huihui Bai, Riquan Zhang, and Yao Zhao. Cross-part learning for fine-grained image classification. *IEEE TIP*, 2021. 2

[28] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 6

[29] Shenghai Rong, Bohai Tu, Zilei Wang, and Junjie Li. Boundary-enhanced co-training for weakly supervised semantic segmentation. In *CVPR*, 2023. 7

[30] Lixiang Ru, Yibing Zhan, Baosheng Yu, and Bo Du. Learning affinity from attention: End-to-end weakly-supervised semantic segmentation with transformers. In *CVPR*, 2022. 2

[31] Lixiang Ru, Heliang Zheng, Yibing Zhan, and Bo Du. Token contrast for weakly-supervised semantic segmentation. In *CVPR*, 2023. 7

[32] Guolei Sun, Wenguan Wang, Jifeng Dai, and Luc Van Gool. Mining cross-image semantics for weakly supervised semantic segmentation. In *ECCV*, 2020. 2

[33] Feilong Tang, Zhongxing Xu, Zhaojun Qu, Wei Feng, Xingjian Jiang, and Zongyuan Ge. Hunting attributes: Context prototype-aware learning for weakly supervised semantic segmentation, 2024. 2

[34] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008. 2

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3

[36] Changwei Wang, Rongtao Xu, Shibiao Xu, Weiliang Meng, and Xiaopeng Zhang. Treating pseudo-labels generation as image matting for weakly supervised semantic segmentation. In *ICCV*, 2023. 2

[37] Jian Wang, Siyue Yu, Bingfeng Zhang, Xinqiao Zhao, Ángel F. García-Fernández, Eng Gee Lim, and Jimin Xiao. Cross-frame feature-saliency mutual reinforcing for weakly supervised video salient object detection. *PR*, 2024. 2

[38] Wenguan Wang, Guolei Sun, and Luc Van Gool. Looking beyond single images for weakly supervised semantic segmentation learning. *IEEE TPAMI*, 2022. 2

[39] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[40] Xiaoyang Wang, Jimin Xiao, Bingfeng Zhang, and Limin Yu. Card: Semi-supervised semantic segmentation via class-agnostic relation based denoising. In *IJCAI*, 2022. 2

[41] Xiaoyang Wang, Bingfeng Zhang, Limin Yu, and Jimin Xiao. Hunting sparsity: Density-guided contrastive learning for semi-supervised semantic segmentation. In *CVPR*, 2023. 3

[42] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *CVPR*, 2020. 2, 3, 6, 7

[43] Yuanchen Wu, Xiaoqiang Li, Songmin Dai, Jide Li, Tong Liu, and Shaorong Xie. Hierarchical semantic contrast for weakly supervised semantic segmentation. In *IJCAI*, 2023. 6

[44] Jinheng Xie, Xianxu Hou, Kai Ye, and Linlin Shen. Clims: cross language image matching for weakly supervised semantic segmentation. In *CVPR*, 2022. 6, 7, 8

[45] Lian Xu, Wanli Ouyang, Mohammed Bennamoun, Farid Boussaid, and Dan Xu. Multi-class token transformer for weakly supervised semantic segmentation. In *CVPR*, 2022. 7

[46] Lian Xu, Wanli Ouyang, Mohammed Bennamoun, Farid Boussaid, and Dan Xu. Learning multi-modal class-specific tokens for weakly supervised dense object localization. In *CVPR*, 2023. 7

[47] Rongtao Xu, Changwei Wang, Jiaxi Sun, Shibiao Xu, Weiliang Meng, and Xiaopeng Zhang. Self correspondence distillation for end-to-end weakly-supervised semantic segmentation. In *AAAI*, 2023. 2

[48] Sung-Hoon Yoon, Hyeokjun Kweon, Jegyeong Cho, Shinjeong Kim, and Kuk-Jin Yoon. Adversarial erasing framework via triplet with gated pyramid pooling layer for weakly supervised semantic segmentation. In *ECCV*, 2022. 2

[49] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014. 4

[50] Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Mingjie Sun, and Kaizhu Huang. Reliability does matter: An end-to-end weakly supervised semantic segmentation approach. In *AAAI*, 2020. 6

[51] Bingfeng Zhang, Jimin Xiao, Jianbo Jiao, Yunchao Wei, and Yao Zhao. Affinity attention graph neural network for weakly supervised semantic segmentation. *IEEE TPAMI*, 2021. 2

[52] Bingfeng Zhang, Jimin Xiao, Yunchao Wei, and Yao Zhao. Credible dual-expert learning for weakly supervised semantic segmentation. *IJCV*, 2023. 6

[53] Dong Zhang, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, and Qianru Sun. Causal intervention for weakly-supervised semantic segmentation. In *NeurIPS*, 2020. 6

[54] Fei Zhang, Chaochen Gu, Chenyue Zhang, and Yuchao Dai. Complementary patch for weakly supervised semantic segmentation. In *ICCV*, 2021. 2

[55] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19148–19158, 2023. 2

[56] Yabo Zhang, Yuxiang Wei, Dongsheng Jiang, Xiaopeng Zhang, Wangmeng Zuo, and Qi Tian. Controlvideo: Training-free controllable text-to-video generation. *arXiv preprint arXiv:2305.13077*, 2023. 1

[57] Zekang Zhang, Guangyu Gao, Jianbo Jiao, Chi Harold Liu, and Yunchao Wei. Coinseg: Contrast inter-and intra-class representations for incremental segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 843–853, 2023. 1

[58] Xinqiao Zhao, Feilong Tang, Xiaoyang Wang, and Jimin Xiao. Sfc: Shared feature calibration in weakly supervised semantic segmentation. *arXiv preprint arXiv:2401.11719*, 2024. 7

[59] Tianfei Zhou, Meijie Zhang, Fang Zhao, and Jianwu Li. Regional semantic contrast and aggregation for weakly supervised semantic segmentation. In *CVPR*, 2022. 2

[60] Hongguang Zhu, Yunchao Wei, Xiaodan Liang, Chunjie Zhang, and Yao Zhao. Ctp: Towards vision-language continual pretraining via compatible momentum contrast and topology preservation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22257–22267, 2023. 1