# Appendix for Hunting Attributes: Context Prototype-Aware Learning for Weakly Supervised Semantic Segmentation

Feilong Tang[1,2*] Zhongxing Xu[3*] Zhaojun Qu[4] Wei Feng[1,2]
Xingjian Jiang[5] Zongyuan Ge[1,2†]

[1]AIM Lab, Faculty of IT, Monash University [2]Faculty of IT, Monash University
[3]Weill Cornell Medicine, Cornell University [4]Xi'an Jiaotong-Liverpool University
[5]Ann Arbor, University of Michigan
{feilong.tang,zongyuan.ge}@monash.edu

## A Proof for Claim 1

*Proof.* We first review some relevant notations and definitions. In section 3.3 (main paper), $\tilde{\mathcal{P}}_n^c$ is introduced to represent the context prototype of a class, denoted as $\tilde{\mathcal{P}}_n^c = \{\mathbf{p}_i\}_{i=1}^K$. $\mathcal{P}_n^I$ is the instance prototype within mini-batch. Considering the importance of each context prototype. The $w_i$ represents the positiveness score used to adjust the prototype pair. Our optimization objective is defined as maximizing the intra-class similarity (*i.e.,* maximizing the similarity between the current instance prototype and the context prototype of the same class). This objective is mathematically formulated as follows:

$$\max_{\theta} \sum_{i=1}^{K} w_i \cdot \log\left(\frac{\exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_i^\top\right)}{\sum_{k=1}^{K} \exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_k^\top\right)}\right), \quad \mathbf{p}_i \in \tilde{\mathcal{P}}_n^c, \tag{15}$$

where $K$ represents the number of soft positive neighbors, then the similarity $s_i$ is defined as:

$$s_i = \frac{\exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_i^\top\right)}{\sum_{k=1}^{K} \exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_k^\top\right)}. \tag{16}$$

The objective function is then construed to minimize the sum of negative weighted logarithmic similarities, tantamount to maximizing weighted logarithmic similarities. The objective function is defined as:

$$\min_{\theta} \sum_{i=1}^{K} \left[ -w_i \cdot \log\left(\frac{\exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_i^\top\right)}{\sum_{k=1}^{K} \exp\left(\mathcal{P}_n^I \cdot \mathbf{p}_k^\top\right)}\right) \right]. \tag{17}$$

Then, we solve the optimization problem using the Lagrangian multiplier. The problem is defined as follows:

$$\begin{cases} \text{Minimize:} & f\left(s_1, s_2, \ldots, s_K\right) = -\sum_{i=1}^{K} w_i \cdot \log(s_i) \\ \text{Subject to:} & g\left(s_1, s_2, \ldots, s_K\right) = \sum_{i=1}^{K} s_i - 1 = 0, \end{cases} \tag{18}$$

where $f$ is the objective function and $g$ represents the constraint function. The corresponding Lagrangian function and its partial derivatives are defined as follows:

$$\mathcal{L}(s_1, ..., s_K, \lambda) = -\sum_{i=1}^{K} (w_i \cdot \log(s_i)) + \lambda \left( \sum_{i=1}^{K} s_i - 1 \right), \tag{19}$$

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial s_i} = -\frac{w_i}{s_i} + \lambda = 0, \\ \frac{\partial \mathcal{L}}{\partial \lambda} = \sum_{i=1}^{K} s_i - 1 = 0, \end{cases} \tag{20}$$

From Eq. 20, the optimal value of $s_i$ is $s_i^* = \frac{w_i}{\sum_{k=1}^{K} w_k}$ to maximize the Lagrange function $\mathcal{L}$ while satisfying the constraint conditions. which concludes the proof for Claim 1 in section 3.3 (main paper).

Through the method of Lagrange multipliers, we identify the optimal value $s_i$ values that maximize the objective function detailed in Eq. 15 while adhering to the constraint function. This optimization method encourages the model to proportionally relate the similarity between different context prototypes within the same category and the current instance with the corresponding *positiveness* score. Effectively incorporating knowledge from the self-supervised branch into the model, this strategy enhances its capability to distinguish between different prototypes, consequently leading to an improvement in the ability of the model to generalize across various scenarios.

*The first two authors contribute equally to this work.
†Corresponding author: Zongyuan Ge

## B More Details of Support Bank

On the PASCAL VOC 2012 *train* set, the feature set is denoted as $N \times L \times D$, where $N$ represents the number of categories, $L$ represents the number of features per category, which we set to 1000, and $D$ is the dimensional vector for each feature. In the initialization of the support bank, all storage positions are set to zero. When a new prototype needs to be added, the system follows the First-In, First-Out (FIFO) principle to update the position indicated by the index. Subsequently, the index is advanced to point to the next position to be updated. In this way, the oldest data is naturally replaced by the incoming new data, while the index consistently points to the next position that will be updated. At each training step, we establish a confidence threshold $\alpha$ for controlling the pace of bank updates. The support bank $\mathcal{C}$ is updated when the $n$-th class is detected in the $I$ (*i.e.,* $y_i = 1$) and the classification score is higher than $\alpha$, *i.e.,* $\hat{y}_i > \alpha$. Otherwise, we keep the support Bank $\mathcal{C}$.

We are also interested in how our method would perform when various confidence threshold values are selected. The results shown in Table S1 indicate that the optimal value for $\alpha$ is 0.8 (our default).

Table S1. Ablation study of sensitivity of our approach to the selection of confidence threshold $\alpha$ on PASCAL VOC 2012 *train* set.

| threshold $\alpha$ | 0.7 | 0.75 | 0.8 | 0.85 | 0.9 | 0.95 |
|---|---|---|---|---|---|---|
| mIoU(%) | 62.0 | 62.2 | 62.5 | 62.5 | 62.3 | 62.2 |

## C More Experimental Details

**Dataset.** We evaluate our proposed method on PASCAL VOC 2012 [4] segmentation benchmark with 20 foreground classes and one background class. The official dataset split contains 1,464 images for training, 1,449 for validation, and 1,456 for testing. The used MS COCO 2014 [8] dataset has 81 classes and contains 80k train and 40k validation images, which is more challenging for weakly supervised semantic segmentation.

**Implementation Details.** For pseudo-label generation, 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, 3]. In our experiments, the CAM-generation model is trained with a batch size of 16 on one Nvidia 3090 GPU for Pascal VOC 2012 and on 4 Nvidia 3090 GPUs for MS COCO 2014. The scale ratios of multi-scaled CAM during inference are set to $\{0.5, 1.0, 1.5, 2.0\}$ following [3].

With regard to the training details of semantic segmentation DeepLabV2 model: (1) For PASCAL VOC 2012 dataset, images are randomly scaled to $\{0.5, 0.75, 1.0, 1.25, 1.5\}$ and cropped to 321×321. The batch size is set to 10, and the number of total training iterations is set to 20k. (2) For MS COCO 2014 dataset, we apply data augmentation following the approach proposed in [6]. Images are randomly scaled to $\{0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$, and then cropped to 481×481. The training batch size is set to 5 and the number of training iterations is set to 100k. For both (1) and (2), the initial learning rate is $2 \times 10^{-4}$ for the ImageNet pre-trained model. We use a poly learning rate scheduler decayed with a power of 0.9. Balanced cross-entropy loss is adopted for MS COCO train set as in [5, 6]. For testing, we employ a multi-scaled strategy and utilize denseCRF with default hyper-parameters from [2] for post-processing.

Table S2. Evaluation (mIoU (%)) of different pseudo labels on MS COCO 2014 training set.

| Method | MS COCO | |
|---|---|---|
| | CAM | Mask |
| CONTA [11] | 28.7 | 35.2 |
| ESOL [7] | 35.7 | 44.6 |
| IRN [1] | 33.5 | 42.9 |
| **+CPAL (Ours)** | **37.6** ↑4.1 | **46.2** ↑3.3 |
| AMN [6] | 40.3 | 46.7 |
| **+CPAL (Ours)** | **42.9** ↑2.6 | **48.9** ↑2.2 |
| CLIP-ES [9] | 40.5 | 47.3 |
| **+CPAL (Ours)** | **43.4** ↑2.9 | **49.1** ↑1.8 |

## D Ablation Study on MS COCO

For MS COCO, the loss coefficients $\lambda_{BCE}$ and $\lambda_{Self}$ are both set as 1 in Eq. 1 (main paper). The threshold $\tau$ in Eq. 4 (main paper) is set to 0.2. Support bank size for each class to store region embeddings, with the size set to 2000. The $k$-means prototype clustering in Section 3.2 (main paper) is performed only once at the beginning of each epoch, and the per-class prototype number $N_p$ is set to 60, and the top-$K$ candidate neighbors is set to 25 in Eq. 6 (main paper).

**Pseudo-Label Quality on COCO Dataset.** Experimental results reported in Table S2 show that the PACAM from CPAL is significantly better than the previous works on MS COCO 2014 training set [8], which is a larger-scale dataset compared with PASCAL VOC 2012 [4]. We use CRF-improved CAM as the final pseudo-label following previous works [6, 10]. In the table, we can observe that the use of the CPAL plugin, for instance, in the IRN [1] method, results in an improvement of 4.1% in CAM pseudo-label and 3.3% in Mask pseudo-label relative to the baseline. Similarly, improvements are observed in the AMN [6] and CLIP-ES [9] methods. These results are a supplement to the MS COCO 2014 validation set performance reported in the main paper.

**Effectiveness of candidate neighbors and positiveness.**

Table S3. Analysis of the positiveness and number of candidate neighbors $K$. The mIoU values are evaluated on the MS COCO.

| Neighbor | Positiveness | $K$ | mIou(%) |
|---|---|---|---|
| ✔ | ✔ | 25 | 37.6 |
| ✗ | ✗ | - | 35.2 |
| ✔ | ✗ | 25 | 36.3 |
| ✔ | ✔ | 10 | 36.2 |
| ✔ | ✔ | 25 | 37.6 |
| ✔ | ✔ | 60 | 35.8 |

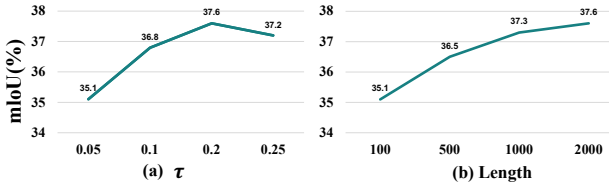


Figure S1. Sensitivity analysis on MS COCO dataset, in terms of (a) the threshold $\tau$ used to generate 0-1 seed masks from heatmaps. (b) the length of the support set. The results show that CPAL is not sensitive to them.

We assess the significance of candidate neighbors and positiveness on MS COCO dataset, as illustrated in Table S3. Removing positiveness and utilizing all neighbors for prediction, Miou accuracy decrease in CAM from 36.3% to 35.2%. It allows the model to adaptively and selectively focus on neighbors that contribute significantly to the task during the learning process, while disregarding uninformative neighbors.

In the third section of Table S3, we conducted experiments to analyze the impact of the number of neighbors on this large-scale dataset. On the one hand, having a sufficient number of neighbors can enhance the diversity of features, allowing the model to comprehensively learn different aspects of features and improve its robustness. On the other hand, including less-correlated prototypes may introduce too much noise during the training process and diminish the ability of the model to perceive discriminative features.

**Analysis of Hyper-parameters.** We conduct a hyperparameter sensitivity analysis, varying values such as (a) the threshold $\tau$ for generating the 0-1 seed mask. Fig. S1 (a) indicates that the optimal $\tau$ value is 0.2. Additionally, we examine (b) the length of the support set, finding that a larger set enhances model performance. Fig. S1 (b) shows that the encoder trained with the largest set achieves the highest accuracy of 37.6%, suggesting that increasing capacity enables the model to find more correlated neighbors for support.

## E Pseudo Code

To make our training procedure clear, we first present the overall training process of our proposed CPAL, which includes the detailed training steps and configurations of neighbor exploring and positiveness calculation. We also illustrate how to align the mini-batch feature to the globe feature and the self-supervised loss.

## F Limitation

The Context Prototype-Aware Learning (CPAL) strategy proposed in this work, while demonstrating its superiority, requires maintaining an external support bank during training, which may increase the complexity of memory usage. However, experiments have shown that CPAL does not demand a high amount of memory, and we have implemented the strategy using smaller memory. Furthermore, the aspect of consistency regularization within CPAL is yet to be thoroughly explored. Future research should focus on integrating CPAL more deeply into frameworks based on consistency regularization, such as by applying CPAL to perturbed samples to explore potential improvement.

## G More Qualitative Results

In Fig. S2 and Fig. S3, we show segmentation results of CPAL on VOC 2012 test and COCO 2014 val, respectively. Consistent with our analysis in the main paper, we observe that CPAL is able to produce accurate segmentation results with crisp boundaries in diverse scenarios ($i.e.$, co-occurrence and similar categories).

## References


[1] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *CVPR*, 2019. 2

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017. 2

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

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 2

[5] Jungbeom Lee, Eunji Kim, and Sungroh Yoon. Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In *CVPR*, 2021. 2

[6] 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. 2

[7] Jinlong Li, Zequn Jie, Xu Wang, Xiaolin Wei, and Lin Ma. Expansion and shrinkage of localization for weakly-supervised semantic segmentation. In *NeurIPS*, 2022. 2

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

**Algorithm 1:** Pseudocode of CPAL

**Input:**

$\eta_{\text{base}}$: base learning rate
$\mathcal{C}$: the support bank
GAP: global average pooling function
$\theta, \mathbf{w}_n, v,$: the classification model, the classifier weight, the projection head
$y$: image-level label vector
$n$: the n-th class
$\text{IP}(\cdot)$: the operator of calculating instance prototype in Eq. 4
$\text{CP}(\cdot)$: the operator of calculating context prototypes in Eq. 5
$\text{NN}(\cdot)$: the operator of calculating soft positive neighbors in Eq. 6
$W$: the operator of calculating the positiveness score in Eq. 7
$\delta_n$: the shift terms in Eq. 12
$\mathcal{L}^{BCE}(\cdot)$: multi binary cross-entropy loss in Eq. 2
$\mathcal{L}^{Self}(\cdot)$: self-supervised loss in Eq. 14
$T, B$: total optimization steps, batch size

**Output:** Updated classification model

**for** $t = 1$ **to** $T$ **do**
  $\chi \leftarrow \left\{x^{(i)} \sim \mathcal{D}\right\}_{i=1}^{N}$ // Sample a batch of B images
  **for** $x^{(i)} \in \chi$ **do**
    $f \leftarrow \theta\left(x^{(i)}\right)$ ; // Extract features using the network $\theta$
    $\hat{y} \leftarrow \text{GAP}(\mathbf{w}_n(f))$ ; // Compute logits prediction with classifier weights $\mathbf{w}_n$
    $\text{CAM} \leftarrow \text{ReLU}(\mathbf{w}_n^{\top} f)$ // Generate CAM using feature maps and classifier weights
    $z \leftarrow v(f)$ ; // Transform features using projection head $v$
    $\hat{z} \leftarrow z + \delta_n$ ; // Features distribution alignment by adding the shift terms when the $n$-class appears in $x$.
    $\mathcal{P}_n^I \leftarrow \text{IP}(\text{CAM}, \hat{z}), \mathcal{P}_n^c \leftarrow \text{CP}(\mathcal{C}, \mathcal{P}_n^I, \text{k-mean})$ ; // Modeling instance prototype and context prototypes
    $w \leftarrow W(\mathcal{P}_n^I, \mathcal{P}_n^c)$ ; // Calculate positiveness score between instance and context prototypes
    $\hat{\mathcal{P}}_n^c \leftarrow \text{NN}(\mathcal{P}_n^I, w)$ ; // Calculate the soft positive neighbors
    $\text{PACAM} \leftarrow \cos(\hat{\mathcal{P}}_n^c, f)$ ; // Generate PACAM using cosine similarity with $\hat{\mathcal{P}}_n^c$
    $\mathcal{L}_{x_1} \leftarrow \mathcal{L}^{BCE}(y, \hat{y})$ ; // multi binary cross-entropy loss
    $\mathcal{L}_{x_2} \leftarrow \mathcal{L}^{Self}(\text{PACAM}, \text{CAM})$ ; // self-supervised loss with consistency regularization
    $\mathcal{L}_{\text{total}} := \mathcal{L}_{x_1} + \mathcal{L}_{x_2}$ ; // sum up the total loss for the instance $x$
  **end**
  $\nabla_{\theta, v, \mathbf{w}_n} \leftarrow \frac{1}{B} \sum_{i=1}^{B} \partial_{\theta, v, \mathbf{w}_n} \mathcal{L}_{\text{total}}$ ; // compute the total loss gradient w.r.t. $\theta, v, \mathbf{w}_n$
  $\eta \leftarrow \eta_{\text{base}} \cdot \left(\cos\left(\pi\left(t - t_0\right) / \left(T - t_0\right)\right) + 1\right) / 2$ ; // cosine decay learning rate
  $\theta, v, \mathbf{w}_n \leftarrow \text{optimizer}\left(\theta, v, \mathbf{w}_n \nabla_{\theta, v, \mathbf{w}_n}, \eta\right)$ ; // update parameters
**end**


Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2

[9] 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. 2

[10] Jinheng Xie, Xianxu Hou, Kai Ye, and Linlin Shen. Clims: cross language image matching for weakly supervised semantic segmentation. In *CVPR*, 2022. 2

[11] Dong Zhang, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, and Qianru Sun. Causal intervention for weakly-supervised semantic segmentation. In *NeurIPS*, 2020. 2

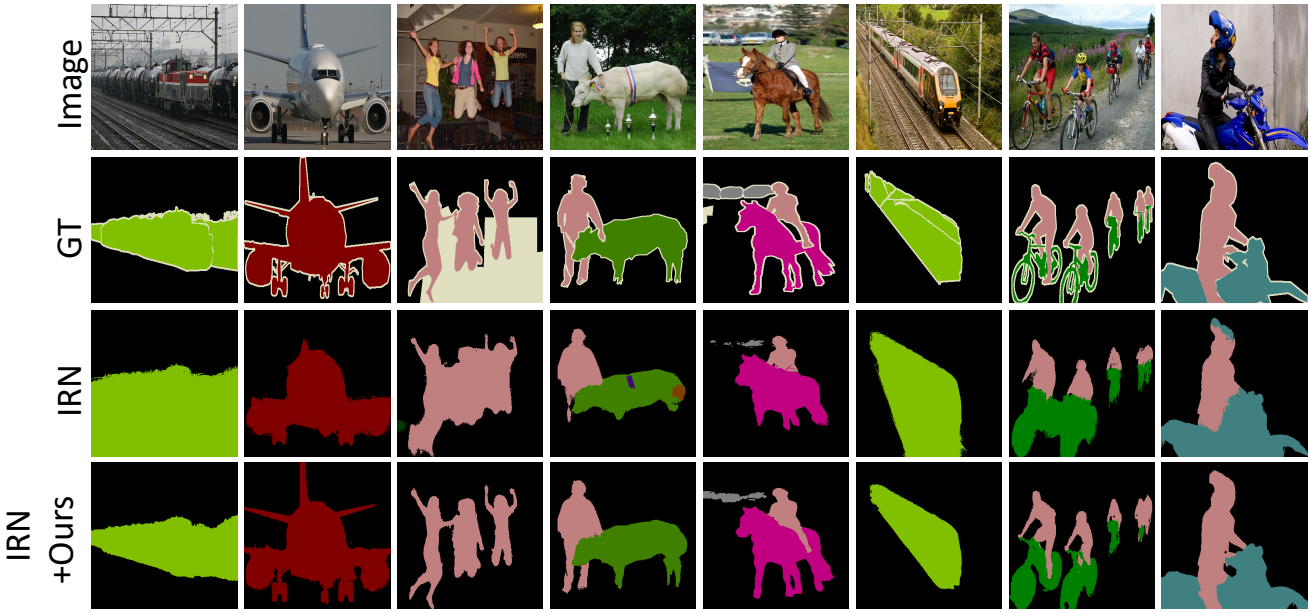


Figure S2. Qualitative segmentation results on VOC 2012 test.

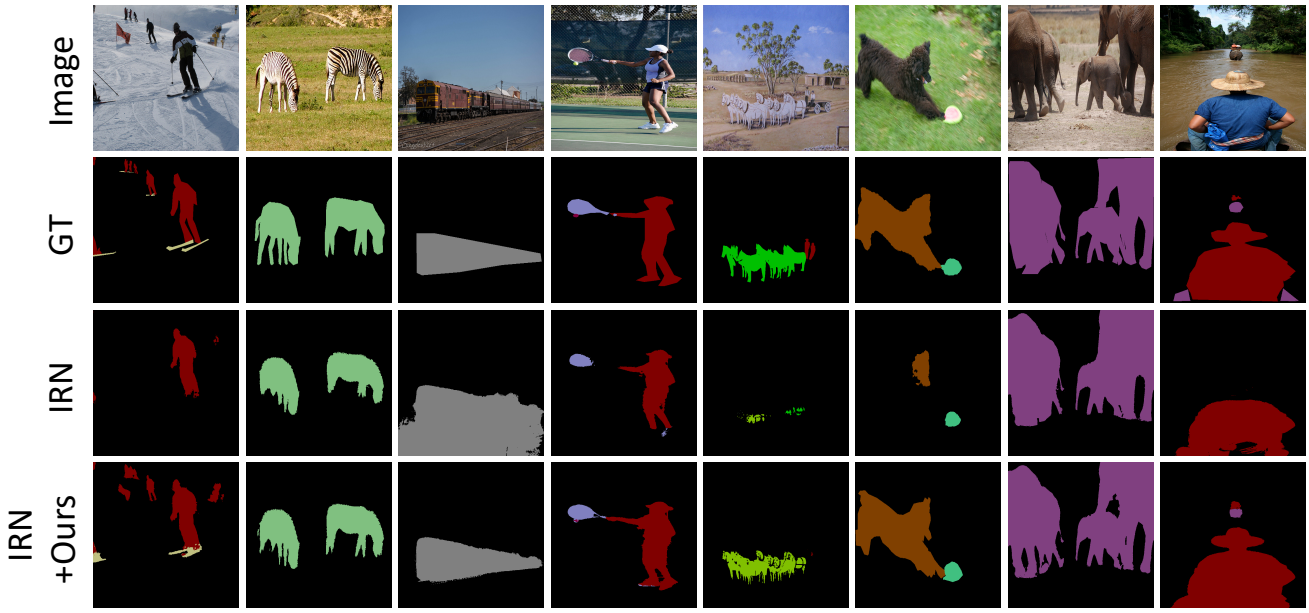


Figure S3. Qualitative segmentation results on MS COCO val.