POT: Prototypical Optimal Transport for Weakly Supervised Semantic Segmentation

Supplementary Material

A1. Additional Training Details

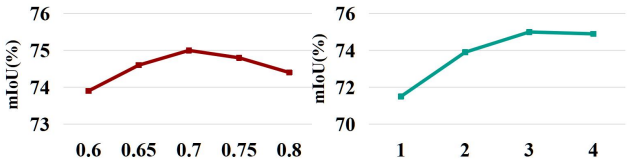
In our experiment for obtaining prototype CAMs, we train our model on a single NVIDIA RTX 3090 GPU with 24GB memory, using a batch size of 16. To ensure robust and consistent results, we adopt the same data augmentation strategies as previous works, including random flipping, random scaling, and cropping, as described in [3, 6]. When employing CLIP-ES [8] as baseline methods, we first save the results from the CLIP-ES model before training and then load the results during training to reduce computational costs, and all images are resized to 512×512. For pseudo label generation in the PASCAL VOC 2012 dataset [4], we utilize the IRN [1] post-processing method to refine the CAMs. However, due to the computational cost, we directly use DenseCRF [5] as the post-processing method in the MS COCO 2014 dataset [7] following SIPE [3] and SFC [14]. For further segmentation model training, we employ ResNet101-based DeepLabV2 [2] and follow the settings established by previous methods [11, 12].

A2. Additional Experimental Results

A2.1. Hyper-parameter Analysis

Threshold τ in Sec. 3.1. In the POT framework for WSSS, confident feature vectors are partitioned into multiple clusters, which are further employed to construct cluster prototypes as anchors for the subsequent feature allocation stage. Therefore, it is essential to set an appropriate threshold τ for selecting these confident features. As shown in Fig. A1 (a), we demonstrate the impact of different τ values on the performance of generated CAMs on the PASCAL VOC 2012 train set. Specifically, we vary the threshold from 0.6 to 0.8 with an interval of 0.05. The results indicate that the highest mIoU is obtained when τ is set to 0.7. Increasing τ beyond this value results in the exclusion of some foreground pixels, thereby resulting in the cluster prototypes failing to adequately capture the essential characteristics of these foreground pixels. When generating prototype CAMs using cosine similarity, some foreground regions cannot be fully activated. Conversely, decreasing τ below 0.7 leads to the inclusion of background pixels, thereby resulting in the cluster prototypes capturing the characteristics of a mixture of both foreground and some background pixels. When generating prototype CAMs using cosine similarity, such prototypes activate a higher proportion of background pixels. Both of these two scenarios degrade the overall mIoU performance of the model.

K for k-means clustering in Sec. 3.1. In our proposed framework, confident features are partitioned into K clusters using the k-means clustering technique. To evaluate the impact of different K values on the performance of the generated CAMs, we conduct a series of experiments on the PASCAL VOC 2012 train set as illustrated in Fig. A1 (b). The results demonstrate that the performance of the generated CAMs improves as K increases from 1 to 3, with the highest mIoU achieved when K is set to 3. Further increasing K beyond 3 does not yield any significant performance gains. Moreover, a larger K leads to a substantial increase in computational cost. Therefore, K is set to 3 in this paper. These results demonstrate the limitation of single-prototype methods and the effectiveness of multi-cluster activation proposed in our framework.



(a) Threshold τ (b) K for k-means clustering

Figure A1. Effect of τ and K on the quality of generated CAMs on the PASCAL VOC 2012 train set. τ is leveraged to select confident features with CAM values above this threshold. K is the number of clusters per class in the k-means clustering algorithm.

Method	Sup.	Time (hour)	mIoU (%)
SIPE$_{CVPR'22}$ [3]	I	10.1	68.8
PSDPM$_{CVPR'24}$ [15]	I + L	18.7	74.1
POT (Ours)	I + L	11.3	76.1

Table A1. Training cost comparisons on PASCAL VOC 2012 validation dataset. Methods are run on one NVIDIA RTX 3090 GPU. Sup.: Supervision; I: Image-level labels; L: Large language model.

A2.2. Training Cost Comparisons

When integrating with CLIP-ES [8], both the image and text encoders are frozen. To further optimize computational resources, we first save the results generated by CLIP-ES before training. Subsequently, these saved results are loaded during training. Detailed integration procedures are provided in Sect. A3. Tab. A1 offers a comparative analysis of the training time and performances on the PASCAL VOC 2012 validation set among three different methods. SIPE [3] is the first method to utilize a single prototype for each class per image to acquire prototype CAMs. Relying solely on image-level labels, SIPE achieves the shortest training time among the three methods. However, this approach also demonstrates a notable performance gap, exhibiting a 7.3% lower mIoU compared to our proposed method. In contrast, PSDPM [15] integrates its method with CLIP-ES and adopts a two-round training strategy. The first round is dedicated to training a segmentation model, which is subsequently leveraged to assist in the prototype generation process in the second round. This multi-round approach leads to the longest training time among the compared methods. Our method, however, not only reduces the training time by 39.6% relative to PSDPM [15] but also achieves a 2.0% improvement in performance, highlighting its efficiency and effec-

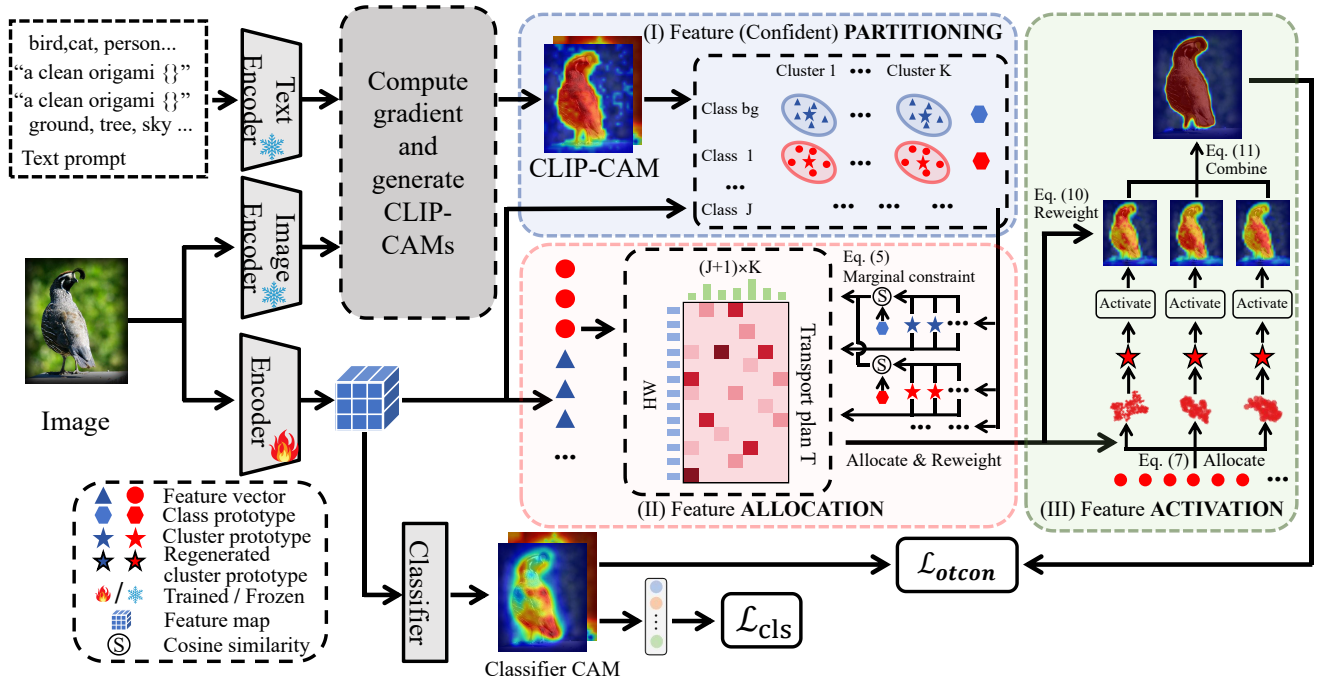


Figure A2. Illustration of the POT framework integrating to the CLIP-ES [8] model. Image and text prompts are separately fed into the frozen CLIP encoders to extract their respective features. The features are then leveraged to generate initial GradCAMs [9], using the gradients of calculating cosine similarity between them. These initial GradCAMs are subsequently refined using Sinkhorn normalization following previous works [8, 13], resulting in CLIP-CAMs. Features from the training encoder, corresponding to the positions of confident CLIP-CAM values, are partitioned into multiple clusters. Then, an optimal transport is constructed to allocate all features to these clusters. This allocation process is guided by a marginal constraint, which uses cosine similarities between cluster prototypes and class prototypes to ensure accurate allocation. Once the features are allocated, new prototypes are generated using all the features within each cluster in the feature activation stage. These newly generated prototypes are used to activate their corresponding feature clusters. The activated results are reweighted by the transport plan and combined to form the final CAM predictions. Additionally, an OT-based consistency loss is applied between the CAM predictions and the classifier CAMs to optimize feature representations within the framework. For simplicity, the process in the feature activation stage is demonstrated with a single class.

tiveness for generating robust CAM results in WSSS.

A3. Integration Details on CLIP-ES

CLIP-ES [8] explores leveraging Contrastive Language-Image Pre-training models (CLIP) to localize different categories in WSSS with image-level labels, demonstrating superior performances. Further methods, such as [10, 15], integrate their methods into the CLIP-ES model and achieve significant performances. However, CLIP-ES [8] is a training-free model, and we illustrate the procedure of integrating our framework into the CLIP-ES [8] in this section.

During training, the input image and text prompt are separately fed into frozen CLIP encoders to generate corresponding features. Then, cosine similarity is used to capture the correlation between these features, with gradients in this process utilized to generate the initial GradCAMs [9]. These CAMs are subsequently refined using Sinkhorn normalization following previous works [8, 13], resulting in CLIP-CAMs. The position indexes of confident CLIP-CAMs (above the threshold τ) are leveraged to select corresponding features from a trainable encoder (i.e., ResNet50). These feature vectors are grouped into a fixed number of clusters for each class in the first stage. The clusters capture the most critical characteristics of their respective classes, and a cluster prototype is generated for each of them to serve as an anchor for the subsequent feature allocation. The second stage employs Optimal Transport (OT) to allocate features to clusters. The resulting transport plan provides the probability distribution for assigning each feature to a specific cluster prototype established in the first stage, thereby guiding the systematic allocation of feature vectors. In the final stage, the cluster prototypes are updated based on the newly allocated feature vectors, and features are activated individually by calculating cosine similarities with their corresponding cluster prototypes. In addition, an OT-based consistency loss is introduced to maintain the consistency of the final activation results and the classifier CAMs, which optimizes the feature representation and provides an effective and bounded exploration of prototypes.

A4. Qualitative Results

We present a detailed visualization comparison of the pseudo-label results obtained in the PASCAL VOC 2012 and COCO 2014

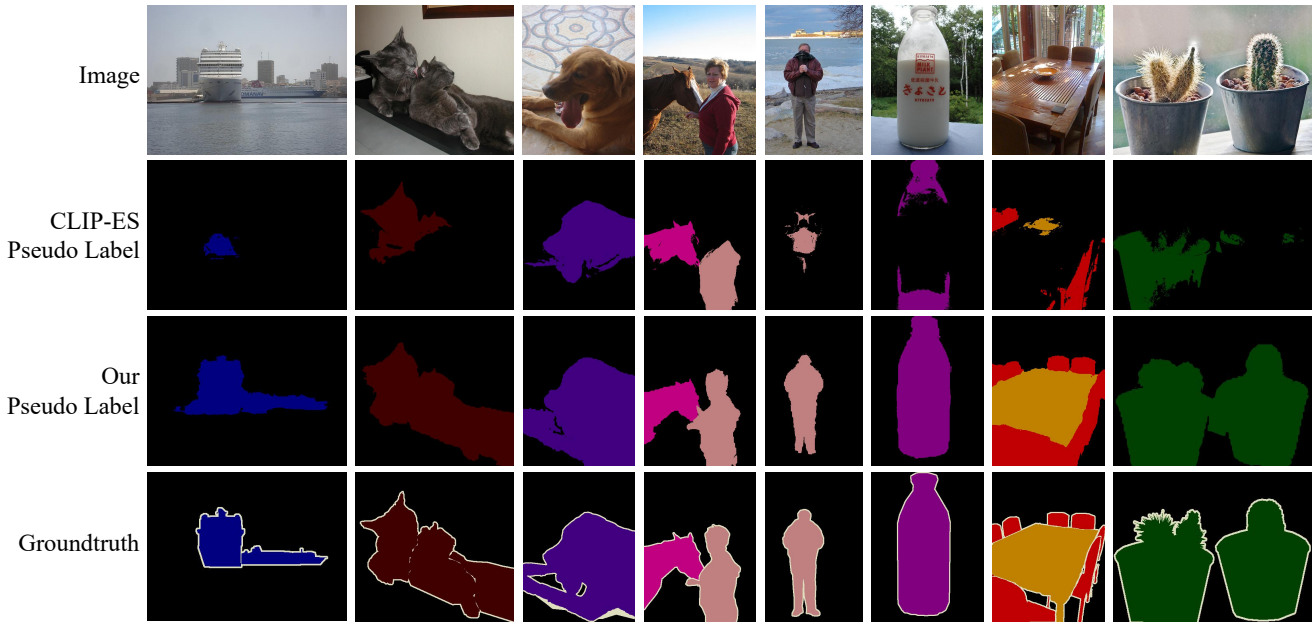


Figure A3. Visualization of pseudo-label results in PASCAL VOC 2012 dataset. Our pseudo-labels are more complete than CLIP-ES [8].

Figure A4. Visualization of pseudo-label results in COCO 2014 dataset. Our pseudo-labels are more complete than CLIP-ES [8].

datasets. To ensure a comprehensive evaluation, we selected a diverse set of scenes encompassing various categories, including persons, animals, plants, and transportation. As shown in Fig. A3 and Fig. A4, our results exhibit greater completeness and accuracy compared to those generated by the CLIP-ES [8] model. These visual examples highlight the robust performance of our framework in accurately segmenting objects across different categories and environments. The improved completeness and precision can be attributed to the refined feature clustering and optimal transport mechanisms employed in our framework. These techniques enhance the capture of discriminative features, leading to more reliable and consistent segmentations. Moreover, the visual comparisons demonstrate the advanced capabilities of our framework in recognizing and delineating diverse object categories under varying conditions, thereby validating its effectiveness in WSSS.

References

[1] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *CVPR*, pages 2209–2218, 2019. 7, 1

[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. *IEEE TPAMI*, 40(4):834–848, 2017. 1

[3] Qi Chen, Lingxiao Yang, Jian-Huang Lai, and Xiaohua Xie. Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation. In *CVPR*, pages 4288–4298, 2022. 1, 2, 4, 6, 7

[4] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 84(4):98–136, 2015. 6, 1

[5] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NeurIPS*, pages 834–848, 2011. 7, 1

[6] Jungbeom Lee, Eunji Kim, and Sungroh Yoon. Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In *CVPR*, pages 4071–4080, 2021. 6, 7, 1

[7] 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*, page 740–755, 2014. 6, 1

[8] 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*, pages 15305–15314, 2023. 6, 7, 8, 1, 2, 3

[9] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 2

[10] Feilong Tang, Zhongxing Xu, Zhaojun Qu, Wei Feng, Xingjian Jiang, and Zongyuan Ge. Hunting attributes: Context prototype-aware learning for weakly supervised semantic segmentation. In *CVPR*, pages 3324–3334, 2024. 1, 2, 4, 6, 7

[11] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *CVPR*, pages 12275–12284, 2020. 2, 4, 1

[12] Yuanchen Wu, Xiaoqiang Li, Songmin Dai, Jide Li, Tong Liu, and Shaorong Xie. Hierarchical semantic contrast for weakly supervised semantic segmentation. In *IJCAI*, pages 1542–1550, 2023. 1

[13] Bingfeng Zhang, Siyue Yu, Yunchao Wei, Yao Zhao, and Jimin Xiao. Frozen clip: A strong backbone for weakly supervised semantic segmentation. In *CVPR*, pages 3796–3806, 2024. 2

[14] Xinqiao Zhao, Feilong Tang, Xiaoyang Wang, and Jimin Xiao. Sfc: Shared feature calibration in weakly supervised semantic segmentation. In *AAAI*, pages 7525–7533, 2024. 1, 4, 6, 7

[15] Xinqiao Zhao, Ziqian Yang, Tianhong Dai, Bingfeng Zhang, and Jimin Xiao. Psdpm: Prototype-based secondary discriminative pixels mining for weakly supervised semantic segmentation. In *CVPR*, pages 3437–3446, 2024. 1, 4, 6, 7, 2