

# Weakly Supervised Co-training with Swapping Assignments for Semantic Segmentation

Xinyu Yang<sup>1</sup>, Hossein Rahmani<sup>1</sup>, Sue Black<sup>2</sup>, and Bryan M. Williams<sup>1</sup>

<sup>1</sup> Lancaster University, Bailrigg Lancaster, LA1 4YW, UK

<sup>2</sup> University of Oxford, Wellington Square Oxford, OX1 2JD, UK  
 {xyang28,h.rahmani,b.williams6}@lancaster.ac.uk, sue.black@sjc.ox.ac.uk

**Abstract.** Class activation maps (CAMs) are commonly employed in weakly supervised semantic segmentation (WSSS) to produce pseudo-labels. Due to incomplete or excessive class activation, existing studies often resort to offline CAM refinement, introducing additional stages or proposing offline modules. This can cause optimization difficulties for single-stage methods and limit generalizability. In this study, we aim to reduce the observed CAM inconsistency and error to mitigate reliance on refinement processes. We propose an end-to-end WSSS model incorporating guided CAMs, wherein our segmentation model is trained while concurrently optimizing CAMs online. Our method, Co-training with Swapping Assignments (CoSA), leverages a dual-stream framework, where one sub-network learns from the swapped assignments generated by the other. We introduce three techniques in this framework: i) soft perplexity-based regularization to penalize uncertain regions; ii) a threshold-searching approach to dynamically revise the confidence threshold; and iii) contrastive separation to address the coexistence problem. CoSA demonstrates exceptional performance, achieving mIoU of 76.2% and 51.0% on VOC and COCO validation datasets, respectively, surpassing existing baselines by a substantial margin. Notably, CoSA is the first single-stage approach to outperform all existing multi-stage methods including those with additional supervision. Source code is publicly available at [here](#).

**Keywords:** Weakly-supervised Learning · Semantic Segmentation · CAM

## 1 Introduction

The objective of weakly supervised semantic segmentation (WSSS) is to train a segmentation model without relying on pixel-level labels but on weak and cost-effective annotations, such as image-level classification labels [3, 26, 50], object points [4, 45], and bounding boxes [16, 28, 36, 57]. In particular, image-level classification labels have commonly been employed as weak labels due to the minimal or negligible annotation effort required [2, 60]. With the absence of precise localization information, image-level WSSS often makes use of the coarse localization offered by class activation maps (CAMs) [75]. CAMs pertain to the intermediate outputs derived from a classification network. They visually illustrate the activation regions corresponding to each individual class. Thus, they are often used to

generate pseudo masks for training. However, CAMs suffer from i) Inconsistent Activation: CAMs demonstrate variability and lack robustness in accommodating geometric transformations of input images [60], resulting in inconsistent activation regions for the same input. ii) Inaccurate Activation: activation region accuracy is often compromised, resulting in incomplete or excessive class activation, only covering discriminative object regions [1]. Despite enhanced localization mechanisms in the variants GradCAM [55] and GradCAM<sup>++</sup> [7], they still struggle to generate satisfactory pseudo-labels for WSSS [60]. Thus, many WSSS works are dedicated to studying CAM refinement or post-processing [1, 15, 31].

In general, they [2, 18, 50, 65] comprise three stages: CAM generation, refinement, and segmentation training with pseudo-labels. Multi-stage frameworks are known to be time-consuming and complex as several models must be trained at different stages. In contrast, single-stage models [3, 52, 73], which include a unified network of all stages, are more efficient. They are trained to co-optimize the segmentation and classification tasks, but the generated CAMs are not explicitly trained. As a result, they need refinement to produce high-quality pseudo-labels, often leveraging hand-crafted modules, such as CRF in [73], PAMR in [3], PAR in [52, 53]. As the refinement modules are predefined and offline, they decouple the CAMs from the primary optimization. When the refined CAMs are employed as segmentation learning objectives, the optimization of the segmentation branch may deviate from that of the classification branch. Hence, it is difficult for single-stage models to optimize the segmentation task while yielding satisfactory CAM pseudo-labels. This optimization difficulty underlies the inferior performance in single-stage approaches compared to multi-stage [50, 65]. Further, hand-crafted refinement modules require heuristic tuning and empirical changes, thereby limiting their adaptability to novel datasets [3, 52]. Despite the potential benefits of post-refinement in addressing the aforementioned issues associated with CAMs, which have been extensively discussed in WSSS studies, there has been limited exploration of explicit online optimization for CAMs.

The absence of fully optimized CAMs is an important factor in the indispensability of this refinement. In this paper, we take a different approach by optimizing CAMs in an end-to-end fashion. We ask a core question: Can we train a model that delivers reliable, consistent and accurate CAMs, which can be applied directly for WSSS without the necessity for subsequent refinements? We show that the answer is *yes*, in two respects: 1) we note that even though CAM is differentiable, it is not robust to variation. As the intermediate output of classification, CAMs are not fully optimized for segmentation purposes since the primary objective is to minimize classification error. This implies that within an optimized network, numerous weight combinations exist that can yield accurate classification outcomes, while generating CAMs of varying qualities. To investigate this, we conduct *oracle experiments*, training a classification model while simultaneously guiding the CAMs with the segmentation ground truth. A noticeable enhancement in quality is observed in guided compared to vanilla CAMs, without compromising classification accuracy. 2) we demonstrate the feasibility of substituting the oracle with segmentation pseudo-labels (SPL) in the



context of weak supervision. Consequently, we harness the potential of SPL for WSSS by co-training both CAMs and segmentation through mutual learning.

We explore an effective way to substitute the CAM refinement process, *i.e.* guiding CAMs in an end-to-end fashion. Our method optimizes the CAMs and segmentation prediction simultaneously thanks to the differentiability of CAMs. To achieve this, we adopt a dual-stream framework that includes an online network (ON) and an assignment network (AN), inspired by self-training frameworks [5, 22, 68]. The AN is responsible for producing CAM pseudo-labels (CPL) and segmentation pseudo-labels (SPL) to train the ON. Since CPL and SPL are swapped for supervising segmentation and CAMs, respectively, our method is named **Co**-training with **S**wapping **A**ssignments (CoSA).

The benefit of this end-to-end framework is that it enables us to quantify pseudo-label reliability online, as opposed to the offline hard pseudo-labels used in existing methodologies [2, 15, 50, 65]. We can then incorporate soft regularization to compensate for CPL uncertainty, where the segmentation loss for different regions is adaptively weighted according to our estimated perplexity map. In comparison to existing literature, this dynamic learning scheme can exploit the potential of CPL and enhance the final performance, as opposed to performance being constrained by predetermined CPL. The threshold is a key hyper-parameter for generating the CPL [50, 53, 60]. It not only requires tuning but necessitates dynamic adjustment to align with the model’s learning state at various time-steps. CoSA integrates threshold searching to dynamically adapt its learning state, as opposed to the fixed thresholding [12, 18, 52]. This can enhance performance and help to eliminate the laborious manual parameter-tuning task. We further address a common issue with CAMs, known as the coexistence problem, whereby certain class activations display extensive false positive regions that inaccurately merge the objects with their surroundings (Fig. 4). In response, we introduce a technique to leverage low-level CAMs enriched with object-specific details to contrastively separate those coexistent classes.

The proposed CoSA greatly surpasses existing WSSS methods. Our approach achieves the leading results on VOC and COCO benchmarks, highlighting the contribution of this work: **i)** We are the first to propose SPL as a substitute for guiding CAMs. We present compelling evidence of its potential to produce more reliable, consistent and accurate CAMs. **ii)** We introduce a dual-stream framework with swapped assignments, which co-optimizes the CAMs and segmentation predictions in an end-to-end fashion. **iii)** We address the learning dynamics, proposing two components within our framework: reliability-based adaptive weighting and dynamic thresholding. **iv)** We address the CAM coexistence issue, proposing a contrastive separation approach to regularize CAMs, significantly enhancing the results of affected classes.

## 2 Related Work

**Multi-Stage WSSS.** Most image-level WSSS work is multi-stage, typically comprising three stages: CAM generation, CAM refinement, and segmentation train-

ing. Some approaches employ heuristic strategies to address incomplete activation regions, such as adversarial erasing [30, 58, 70, 74], feature map optimization [12–14, 32], self-supervised learning [11, 56, 60], and contrastive learning [15, 27, 64, 76]. Some methods focus on post-refining the CAMs by propagating object regions from the seeds to their semantically similar pixels. AffinityNet [2], for instance, learns pixel-level affinity to enhance CPL. This has motivated other work [1, 10, 20, 38] that utilize additional networks to generate more accurate CPL. Other work studies optimization given coarse pseudo-labels: [40] explores uncertainty of noisy labels, [43] adaptively corrects CPL during early learning, and [50] enhances boundary prediction through co-training. Since image-level labels alone do not yield satisfactory results, several methods incorporate additional modalities, such as saliency maps [18, 37, 38, 76] or CLIP models [42, 63, 67]. Recently, vision transformers [17] have emerged as prominent models for various vision tasks. Several WSSS studies benefit from vision transformers: [21] enhances CAMs by incorporating the attention map from ViT; [65] introduces class-specific attention for discriminative object localization; [42] and [67] leverage multi-modal transformers to enhance performance.

**Single-Stage WSSS.** In contrast, single-stage methods are much more efficient. They contain a shared backbone with heads for classification and segmentation [3, 52, 53, 73]. The pipeline involves generating and refining the CAMs, leveraging an offline module, such as PAMR [3], PAR [52], or CRF [73]. Subsequently, the refined CPL are used for segmentation. Single-stage methods exhibit faster speed and a lower memory footprint but are challenging to optimize due to the obfuscation in offline refinement. As a result, they often yield inferior performance compared to multi-stage methods. More recently, with the success of ViT, single-stage WSSS has been greatly advanced. AFA [52] proposes learning reliable affinity from attention to refine the CAMs. Similarly, ToCo [53] mitigates the problem of over-smoothing in vision transformers by contrastively learning from patch tokens and class tokens. The existing works depend heavily on offline refinement of CAMs. In this study, we further explore the potential of single-stage approaches and showcase the redundancy of offline refinement. We propose an effective alternative for generating consistent, and accurate CAMs in WSSS.

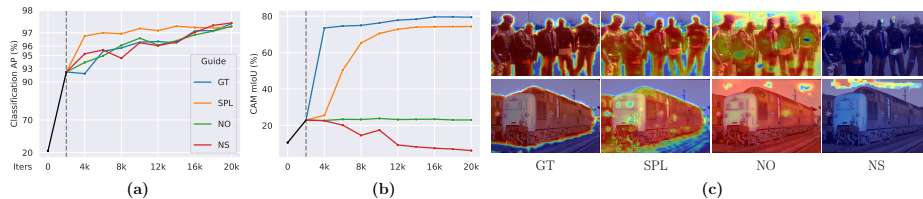
### 3 Method

#### 3.1 Guiding Class Activation Maps

Class activation maps are determined by the feature map  $F$  and the weights  $W_{fc}$  for the last FC layer [75]. Let us consider a  $C$  classes classification problem:

$$\mathcal{L}_{cls}(Z, Y) = \frac{-1}{C} \sum_{c=1}^C \left[ Y^c \log \sigma_Z^c + (1 - Y^c) \log (1 - \sigma_Z^c) \right], \quad (1)$$

where  $\sigma_Z^c \triangleq \sigma(Z^c)$  represents **Sigmoid** activation,  $Y \triangleq Y_{gt}$  denotes the one-hot multi-class label, and  $Z \triangleq GW_{fc}^T \in \mathbb{R}^C$  represents the prediction logits, derived



**Fig. 1: Oracle Experiments** on VOC. CAMs are guided by the ground truth (GT), proposed segmentation pseudo-labels (SPL), no guidance (NO) and random noise (NS). **(a)**: classification performance; **(b)**: CAM quality; **(c)** CAM visualization. All experiments employ 2k-itsers warm-up before guidance is introduced.

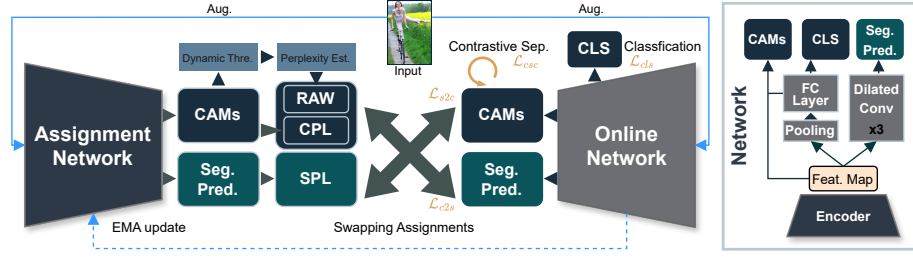
from the final FC layer, where  $G = \text{Pooling}(F) \in \mathbb{R}^D$  is a spatial pooled feature from  $F \in \mathbb{R}^{HW \times D}$ . During training, Eq. (1) is optimized with respect to the learnable parameters  $\theta$  in the backbone. When gradients flow backwards from  $G$  to  $F$ , only a fraction of elements in  $F$  get optimized, implying that a perturbation in  $F$  does not guarantee corresponding response in  $G$  due to the spatial pooling, resulting in non-determinism in the feature map  $F$ . This indeterminate nature can lead to stochasticity of the generated CAMs.

To demonstrate, we conduct oracle experiments wherein we supervise the output CAMs from a classifier with the ground truth segmentation (GT), enabling optimization of all elements in  $F$ . For comparison, we conduct experiments where the CAMs are not guided (NO), and guided with random noise masks (NS). Results, shown in Fig. 1, demonstrate that different guidance for  $M$  does not affect classification even for the NS group, as all experiment groups achieved over 97% classification precision. However, drastic differences can be observed *w.r.t.* the quality of the CAMs. The GT group results in a notable quality improvement over the NO group, as shown in Fig. 1(b)(c). In contrast, the NS group sabotages the CAMs. This suggests the stochasticity of CAMs and explains their variability and lack robustness, something also observed in [2, 12, 60].

Since relying on GT segmentation is not feasible in WSSS, we propose an alternative for guiding CAMs, employing predicted masks as segmentation pseudo-labels (SPL). As shown in Fig. 1, a SPL-guided classifier yields CAMs that significantly outperform vanilla CAMs (NO), performing close to the oracle (GT). Motivated by this, we introduce a co-training mechanism in which CAMs and predicted masks are optimized mutually without additional CAM refinement.

### 3.2 Co-training with Swapping Assignments

**Overall Framework.** As shown in Fig. 2, CoSA contains two networks: an *online* network (ON) and an *assignment* network (AN). ON, parameterized by  $\Theta$ , comprises three parts: a backbone encoder, FC layers, and a segmentation head. AN has the same architecture as ON but uses different weights, denoted  $\Theta'$ . ON is trained with the pseudo assignments generated by AN, while AN is updated by the exponential moving average of ON:  $\Theta' \leftarrow m\Theta' + (1 - m)\Theta$ , where  $m \in [0, 1]$  denotes a momentum coefficient. Consequently, the weights of



**Fig. 2: Co-training with Swapping Assignments (CoSA).** We propose an end-to-end dual-stream weakly-supervised segmentation framework, capable of co-optimizing the segmentation prediction and CAMs by leveraging the swapped assignments, namely CAM pseudo-labels (CPL) and segmentation pseudo-labels (SPL). Our framework comprises two networks: an assignment network (AN) and an online network (ON), where the AN is responsible for generating pseudo-labels for training the ON. While the AN has identical architecture to the ON, it is updated through exponential moving average (EMA) of the ON. The diagram on the right provides an illustration of the architecture. Given weak-augmented images as input, the AN produces CPL to supervise segmentation in the ON ( $\mathcal{L}_{c2s}$ ). During training, the CPL is softened by reliability-based adaptive weighting (RAW), formed based on CAM perplexity estimation and dynamic thresholding. The AN also generates SPL which is utilized to supervise the CAMs ( $\mathcal{L}_{s2c}$ ). Further, the CAMs are regularized to contrastively separate the foreground from the background regions ( $\mathcal{L}_{csc}$ ). Note that the ON is also trained for classification using the image-level class labels ( $\mathcal{L}_{cls}$ ).

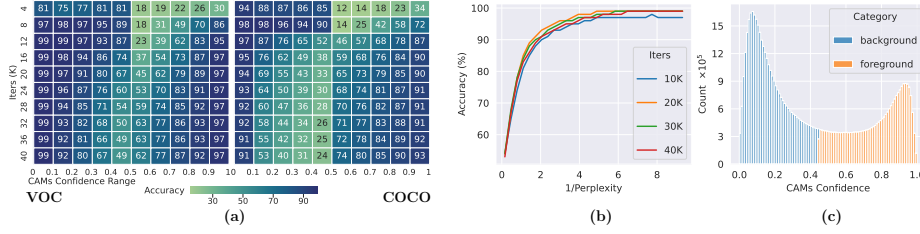
AN represent a delayed and more stable version of the weights of ON, which helps to yield a consistent and stabilized learning target [22].

An image and class label pair  $(x, Y_{gt})$  is randomly sampled from a WSSS dataset  $\mathcal{D}$ . CoSA utilizes two augmented views  $\mathcal{T}_s(x)$  and  $\mathcal{T}_w(x)$  as input for ON and AN, respectively, representing strong and weak image transformations. During training, AN produces CAMs  $\mathcal{M}'$  and segmentation predictions  $\mathcal{S}'$ . The CAM pseudo-labels (CPL) and segmentation pseudo-labels (SPL) are generated by  $\mathcal{M}'$  and  $\mathcal{S}'$  after filtering with respect to  $Y_{gt}$ . CPL and SPL are subsequently used as learning targets for supervising the segmentation predictions  $\mathcal{S}$  and CAMs  $\mathcal{M}$  from ON, respectively.

**Swapping Assignments.** Our objective is to co-optimize  $\mathcal{S}$  and  $\mathcal{M}$ . A naive approach could enforce the learning objectives  $\mathcal{S} \triangleq \mathcal{S}'$  and  $\mathcal{M} \triangleq \mathcal{M}'$  as a knowledge distillation process [25], where AN and ON play the roles of teacher and student. However, this assumes availability of a pretrained teacher which is not possible in WSSS settings. Instead, we setup a swapped self-distillation objective:

$$\mathcal{L}_{\text{swap}} = \mathcal{L}_{c2s}(\mathcal{S}, \mathcal{M}') + \mathcal{L}_{s2c}(\mathcal{M}, \mathcal{S}') , \quad (2)$$

where  $\mathcal{L}_{c2s}$  optimizes the segmentation performance given the CPL, and  $\mathcal{L}_{s2c}$  assesses the CAM quality with respect to SPL. Building on self-distillation [6, 48], we present this swapped self-distillation framework tailored specifically to facilitate information exchange between the CAMs and segmentation.



**Fig. 3: CPL Analysis (a):** heatmap of CPL accuracy *vs.* confident ranges (x-axis) for different time-steps (y-axis) for VOC and COCO. **(b):** correlation between perplexity and accuracy of CPL for different time-steps. **(c):** distribution of CAMs' confidence categorized by the proposed dynamic threshold on VOC. See *Supp.* for COCO analysis.

### 3.3 Segmentation Optimization

**CAM2Seg Learning.** Previous studies [2, 3, 29, 52] refine the CAMs to obtain pseudo-label, then perform pseudo-label to segmentation learning (PL2Seg). As our guided-CAMs do not require extra refinement process, they can be directly employed as learning targets (CAM2Seg). Nonetheless, CAMs primarily concentrate on the activated regions of the foreground while disregarding the background. As per the established convention [15, 53, 60], a threshold value  $\xi$  is employed for splitting the foreground and the background. Formally, our CAM pseudo-label (CPL) is given by:

$$\hat{\mathcal{Y}}_{x,y}^{\text{CPL}} = \begin{cases} \text{argmax}(\mathcal{M}'_{x,y}) + 1, & \text{if } \nu \geq \xi, \\ 0, & \text{if } \nu < \xi, \end{cases} \quad (3)$$

where  $\nu \triangleq \max(\mathcal{M}'_{x,y})$  denotes the the maximum activation, 0 denotes the background index. Then, the CAM2seg learning objective  $\mathcal{L}_{c2s}$  is cross entropy between  $\hat{\mathcal{Y}}^{\text{CPL}}$  and  $\mathcal{S}$ , as with the general supervised segmentation loss [8].

**Reliability based Adaptive Weighting (RAW).** Segmentation performance depends heavily on the reliability of the pseudo-labels. Thus, it is important to assess their reliability. Existing methods use post-refinement to enhance pseudo-label credibility [3, 73]. As CoSA can generate online CPL, we propose to leverage confidence information to compensate the CAM2Seg loss during training. Specifically, we propose to assess the perplexity scores for each pixel in  $\hat{\mathcal{Y}}^{\text{CPL}}$  and leverage these scores to re-weight  $\mathcal{L}_{c2s}$  for penalizing unreliable regions. However, estimating per-pixel perplexity is non-trivial. Through empirical analysis, we observe a noteworthy association between the confidence values of CAMs and their accuracy at each time-step. This correlation suggests that regions with extremely low or high confidence exhibit higher accuracy throughout training, as shown in Fig. 3(a). To quantitatively model perplexity, we make two assumptions: i) the reliability of pseudo-labels is positively correlated with their accuracy, and ii) the perplexity score is negatively correlated with the reliability. Then, per-pixel perplexity of  $\hat{\mathcal{Y}}_{x,y}^{\text{CPL}}$  is defined as:

$$\mathcal{P}_{x,y} = \begin{cases} [-\log(\lambda_\alpha(\nu - \xi)/(1 - \xi))]^{\lambda_\beta} & \text{if } \nu \geq \xi, \\ [-\log(\lambda_\alpha(\xi - \nu)/\xi)]^{\lambda_\beta} & \text{if } \nu < \xi, \end{cases} \quad (4)$$

where the term within the logarithm denotes the normalized distance to  $\xi$  in  $[0, 1]$ . The logarithm ensures  $\mathcal{P}_{x,y} \rightarrow +\infty$  as distance  $\rightarrow 0$ , and  $\mathcal{P}_{x,y} \rightarrow 0$  as distance  $\rightarrow 1$ .  $\lambda_\alpha \in \mathbb{R}^+$  controls the perplexity score's minimum value and  $\lambda_\beta \in \mathbb{R}^+$  determines the sharpness or smoothness of the distribution. Higher  $\mathcal{P}_{x,y}$  indicates confidence of  $\hat{\mathcal{Y}}_{x,y}^{\text{CPL}}$  closer to threshold  $\xi$ . This observation is substantiated by Fig. 3(a), where confidence values near  $\xi = 0.5$  exhibit lower reliability. Furthermore, the correlation between perplexity and accuracy remains significant across various training time-steps and datasets, as depicted in Fig. 3(b).

Since we hypothesize negative reliability-perplexity correlation, the reliability score can be defined as the reciprocal of perplexity. To accommodate reliability variation for different input, we use the normalized reliability as the per-pixel weights for  $\mathcal{L}_{c2s}$ . This arrives our RAW-based CAM2Seg objective:

$$\mathcal{L}_{c2s}(x, y) = - \frac{|\mathcal{R}|}{\sum_{i,j \in \mathcal{R}} (\mathcal{P}_{i,j} \mathcal{P}_{x,y})^{-1}} \sum_{c=0}^C \left[ \mathbb{1}[\hat{\mathcal{Y}}_{x,y}^{\text{CPL}} = c] \log \left( \frac{\exp \mathcal{S}_{x,y}^c}{\sum_{k=0}^C \exp \mathcal{S}_{x,y}^k} \right) \right], \quad (5)$$

where  $|\mathcal{R}|$  represents total number of pixels in a batch.

**Dynamic Threshold.** Existing WSSS work [52, 53] prescribes a fixed threshold to separate foreground and background, which neglects inherent variability due to prediction confidence fluctuation during training. Obviously, applying a fixed threshold in Fig. 3(a) is sub-optimal.

To alleviate this, we introduce dynamic thresholding. As shown in Fig. 3(c), the confidence distribution reveals discernible clusters. We assume the foreground and background pixels satisfy a bimodal Gaussian Mixture distribution. Then, the optimal dynamic threshold  $\xi^*$  is determined by maximizing the Gaussian Mixture likelihood:

$$\xi^* = \underset{\xi}{\operatorname{argmax}} \prod_{x \in \{\mathcal{M}' \geq \xi\}} \tilde{\pi}_{fg} \mathcal{N}(x | \tilde{\mu}_{fg}, \tilde{\Sigma}_{fg}) + \prod_{x \in \{\mathcal{M}' < \xi\}} \tilde{\pi}_{bg} \mathcal{N}(x | \tilde{\mu}_{bg}, \tilde{\Sigma}_{bg}), \quad (6)$$

where  $\mathcal{N}(x | \mu, \Sigma)$  denotes the Gaussian function and  $\pi, \mu, \Sigma$  are the weight, mean and covariance. To avoid mini-batch bias, we maintain a queue to fit GMM, with the current  $\mathcal{M}'$  batch enqueued and the oldest dequeued. This facilitates establishment of a gradually evolving threshold, contributing to learning stabilization.

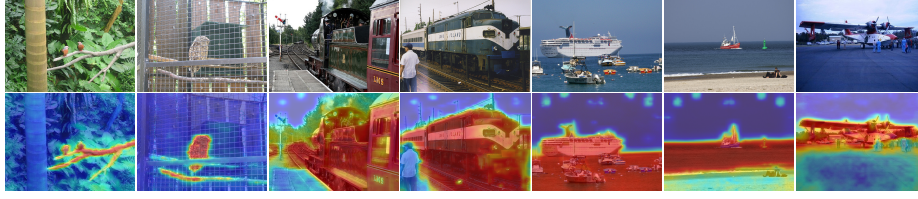
### 3.4 CAM Optimization

**Seg2CAM Learning.** To generate SPL, segmentation predictions  $\mathcal{S}'$  are filtered by the weak labels  $Y_{\text{gt}}$  and transformed into probabilities:

$$\mathcal{S}'_{x,y} = \begin{cases} -\infty, & \text{if } Y_{\text{gt}}^c = 0, \\ \mathcal{S}'_{x,y}, & \text{if } Y_{\text{gt}}^c \neq 0, \end{cases} \quad \hat{\mathcal{Y}}_{x,y}^{\text{SPL}} = \operatorname{Softmax} \left( \frac{\mathcal{S}'_{x,y}}{\tau} \right), \quad (7)$$

where  $\tau$  denotes the temperature to sharpen  $\hat{\mathcal{Y}}_{x,y}^{\text{SPL}}$ . Let  $\mathcal{R}$  be all the positions in SPL, then the Seg2CAM learning objective is defined as:

$$\mathcal{L}_{s2c} = - \frac{1}{C|\mathcal{R}|} \sum_{c=1}^C \sum_{x,y \in \mathcal{R}} \left[ \hat{\mathcal{Y}}_{x,y}^{\text{SPL}}[c] \log(\sigma(\mathcal{M}_{x,y}^c)) + (1 - \hat{\mathcal{Y}}_{x,y}^{\text{SPL}}[c]) \log(1 - \sigma(\mathcal{M}_{x,y}^c)) \right]. \quad (8)$$



**Fig. 4: Illustration of Coexistence Problem in CAMs.** The first row shows the input images. The second row shows the coexistence problem e.g. ‘bird’ with ‘branches’, ‘train’ with ‘railways’ and ‘boat’ with ‘the sea’.

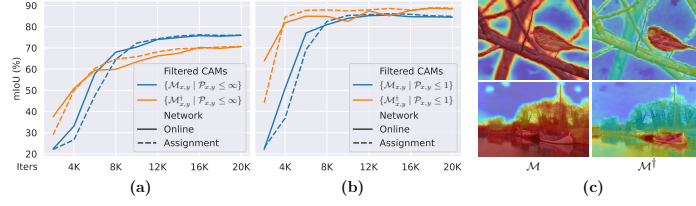
**Coexistence Problem in CAMs.** Certain class activations often exhibit large false positive regions, where objects are incorrectly merged with surroundings, as shown in Fig. 4. For instance, the classes ‘bird’ and ‘tree branches’, ‘train’ and ‘railways’, *etc.* frequently appear together in VOC dataset. We refer to this issue as the coexistence problem. We hypothesize that the coexistence problem is attributed to three factors: i) Objects that coexist in images, such as ‘tree branches’, are not annotated *w.r.t.* weak labels, which makes it challenging for a model to semantically distinguish coexistence. ii) Training datasets lack sufficient samples for such classes. iii) High-level feature maps, though rich in abstract representations and semantic information, lack essential low-level features such as edges, textures, and colors [24]. Thus, CAMs generated from the last layer are poor in low-level information for segmenting objects. Conversely, segmenting objects with high-level semantics is hindered due to factors i) and ii).

**Contrastive Separation in CAMs.** We posit that the effective usage of low-level information can alleviate the coexistence problem. Since shallower-layer feature is rich in low-level info [72], we propose to extract CAMs  $\mathcal{M}^\dagger$  from an earlier layer, and present its comparison with  $\mathcal{M}$  in Fig. 5, showing that directly substituting  $\mathcal{M}$  with  $\mathcal{M}^\dagger$  is not feasible due to the lower mIoU upperbound of  $\mathcal{M}^\dagger$ . However, if we consider the confident regions in  $\mathcal{M}$  and  $\mathcal{M}^\dagger$ , *i.e.* filter by a low-pass perplexity, then  $\{\mathcal{M}_{x,y}^\dagger \mid \mathcal{P}_{x,y} \leq \epsilon\}$  result in higher mIoU than  $\{\mathcal{M}_{x,y} \mid \mathcal{P}_{x,y} \leq \epsilon\}$ , as shown in Fig. 5(b), where  $\epsilon$  denotes a low-pass coefficient. Further, we observe in some examples the presence of coexistence issues in  $\mathcal{M}$  but absence in  $\mathcal{M}^\dagger$  as shown in Fig. 5(c). This suggests that  $\mathcal{M}^\dagger$  performs worse than  $\mathcal{M}$  in general, but better for those regions with low perplexity. Driven by these findings, we propose to regularize  $\mathcal{M}$  by  $\mathcal{M}^{\dagger'}$  (from AN). Specifically,  $\mathcal{M}^{\dagger'}$  after a low-pass filter are used to determine the positive  $\mathcal{R}_{i,j}^+$  and negative  $\mathcal{R}_{i,j}^-$  regions:

$$\begin{aligned} \mathcal{R}_{i,j}^+ &= \left\{ (x,y) \mid \mathcal{P}_{x,y} \leq \epsilon, \hat{y}_{x,y}^{\text{CPL}} = \hat{y}_{i,j}^{\text{CPL}}, (x,y) \neq (i,j) \right\} \\ \mathcal{R}_{i,j}^- &= \left\{ (x,y) \mid \mathcal{P}_{x,y} \leq \epsilon, \hat{y}_{x,y}^{\text{CPL}} \neq \hat{y}_{i,j}^{\text{CPL}} \right\}, \end{aligned} \quad (9)$$

where  $(i,j) \in \Omega$ ,  $\Omega = \{(x,y) \mid \mathcal{P}_{x,y} \leq \epsilon\}$  is low-perplexity region in  $\mathcal{M}^{\dagger'}$ , and  $\hat{y}^{\text{CPL}}$  represents the CPL of  $\mathcal{M}^{\dagger'}$ . Then, we have contrastive separation loss for  $\mathcal{M}$ :

$$\mathcal{L}_{\text{csc}} = -\frac{1}{|\Omega|} \sum_{i,j \in \Omega} \frac{1}{|\mathcal{R}_{i,j}^+|} \sum_{x,y \in \mathcal{R}_{i,j}^+} \log \frac{L_{x,y}^{i,j}}{L_{x,y}^{i,j} + K_{n,m}^{i,j}}, \quad (10)$$



**Fig. 5:  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  Comparisons.** (a): mIoU vs. time-steps for  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  on VOC val. (b): same as (a) but filtered by perplexity. (c): cases of coexistence in  $\mathcal{M}$  but not in  $\mathcal{M}^\dagger$ .

where  $L_{x,y}^{i,j} = \exp(l_d(\mathcal{M}_{i,j}, \mathcal{M}_{x,y})/\tau)$ ,  $l_d(a,b)$  measures the similarity between (a,b),  $\tau$  denotes the InfoNCE loss [47] temperature, and  $K_{n,m}^{i,j} = \sum_{n,m \in \mathcal{R}_{i,j}^-} L_{n,m}^{i,j}$ .

**Overall Objectives.** The objectives encompass the aforementioned losses and a further  $\mathcal{L}_{c2s}^{\mathcal{M}^\dagger}$  to stabilize training and accelerate convergence, resulting in the CoSA objective:

$$\mathcal{L}_{\text{CoSA}} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{cls}}^{\mathcal{M}^\dagger} + \lambda_{c2s}(\mathcal{L}_{c2s} + \mathcal{L}_{c2s}^{\mathcal{M}^\dagger}) + \lambda_{s2c}\mathcal{L}_{s2c} + \lambda_{csc}\mathcal{L}_{csc}. \quad (11)$$

## 4 Experiments

### 4.1 Experiment Details and Results

**Datasets.** We evaluate on two benchmarks: VOC [19] and COCO [41]. VOC encompasses 20 categories with **train**, **val**, and **test** splits of 1464, 1449, and 1456 images. Following WSSS practice [2,3,65], SBD [23] is used to augment the **train** split to 10,582. COCO contains 80 categories with **train** and **val** splits of approx. 82K and 40K images. Our model is trained and evaluated using *only* the image-level classification labels<sup>3</sup>, and employing mIoU as evaluation metrics.

**Implementation Details.** Following [53], we use ImageNet pretrained ViT-base (ViT-B) [17] as the encoder. For classification, we use global max pooling (GMP) [51] and the CAM approach [75]. For the segmentation decoder, we use LargeFOV [8], as with [53]. ON is trained with AdamW [46]. The learning rate is set to 6E-5 in tandem with polynomial decay. AN is updated with a momentum of 0.9994. For preprocessing, the images are cropped to 448<sup>2</sup>, then weak/strong augmentations are applied (see *Supp.*). The perplexity constants ( $\lambda_\alpha, \lambda_\beta$ ) are set to (0.8, 1), GMM-fitting queue length is 100, and softmax temperature  $\tau$  is 0.01. The low perplexity threshold  $\epsilon$  is set to 1 and the loss weight factors ( $\lambda_{c2s}, \lambda_{s2c}, \lambda_{csc}$ ) to (0.1, 0.05, 0.1).

**Semantic Segmentation Comparison.** We compare our method with existing SOTA WSSS methods on VOC and COCO for semantic segmentation in Tab. 1. CoSA achieves 76.2% and 75.1% on VOC12 **val** and **test**, respectively, surpassing the highest-performing single-stage model (ToCo) by 5.1% and 2.9%, as well as all multi-stage methods, including those with additional supervision. In the COCO evaluation, CoSA consistently outperforms other approaches,

<sup>3</sup> Not available for VOC **test** split and so not used in evaluation.



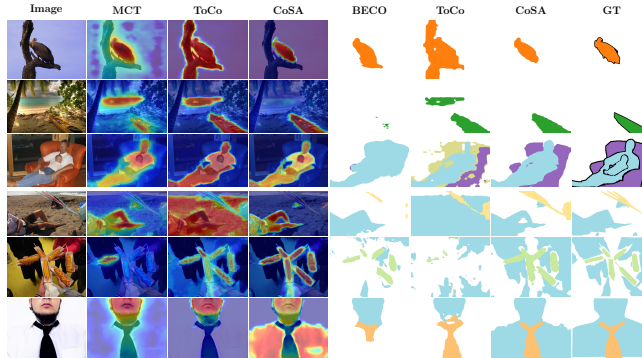
demonstrating a significant increase of 8.7% over the top-performing single-stage methods. Further, there is a also 2.7% improvement over the leading multi-stage method [13]. While our primary goal is to provide an end-to-end WSSS solution, we also offer a multi-stage version of CoSA, denoted as **CoSA-MS** in Tab. 1, where various standalone segmentation networks are trained using our CPL. Our CoSA-MS models can also attains SOTA performance in multi-stage scenarios.

Methods	Sup.	Net.	VOC		COCO
			val	test	val
<b>Supervised Upperbounds.</b>					
Deeplab [8] TPAMI'2017	$\mathcal{F}$	R101	77.6	79.7	–
WideRes38 [61] PR'2019	$\mathcal{F}$	WR38	80.8	82.5	–
ViT-Base [17] ICLR'2021	$\mathcal{F}$	ViT-B	80.5	81.0	–
UperNet-Swin [44] ICCV'2021	$\mathcal{F}$	SWIN	83.4	83.7	–
<b>Multi-stage Methods.</b>					
L2G [26] CVPR'2022	$\mathcal{I} + \mathcal{S}$	R101	72.1	71.7	44.2
Du <i>et al.</i> [18] CVPR'2022	$\mathcal{I} + \mathcal{S}$	R101	72.6	73.6	–
CLIP-ES [42] CVPR'2023	$\mathcal{I} + \mathcal{L}$	R101	73.8	73.9	45.4
ESOL [39] NeurIPS'2022	$\mathcal{I}$	R101	69.9	69.3	42.6
BECO [50] CVPR'2023	$\mathcal{I}$	R101	72.1	71.8	45.1
Mat-Label [59] ICCV'2023	$\mathcal{I}$	R101	73.0	72.7	45.6
<b>CoSA-MS</b>	$\mathcal{I}$	R101	<b>76.5</b>	<b>75.3</b> <sup>[1]</sup>	<b>50.9</b>
Xu <i>et al.</i> [66] CVPR'2023	$\mathcal{I} + \mathcal{L}$	WR38	72.2	72.2	45.9
W-OoD [35] CVPR'2022	$\mathcal{I}$	WR38	70.7	70.1	–
MCT [65] CVPR'2022	$\mathcal{I}$	WR38	71.9	71.6	42.0
ex-ViT [71] PR'2023	$\mathcal{I}$	WR38	71.2	71.1	42.9
ACR-ViT [31] CVPR'2023	$\mathcal{I}$	WR38	72.4	72.4	–
MCT+OCR [15] CVPR'2023	$\mathcal{I}$	WR38	72.7	72.0	42.0
<b>CoSA-MS</b>	$\mathcal{I}$	WR38	<b>76.6</b>	<b>74.9</b> <sup>[2]</sup>	<b>50.1</b>
ReCAM [14] CVPR'2022	$\mathcal{I}$	SWIN	70.4	71.7	47.9
LPCAM [13] CVPR'2023	$\mathcal{I}$	SWIN	73.1	73.4	48.3
<b>CoSA-MS</b>	$\mathcal{I}$	SWIN	<b>81.4</b>	<b>78.4</b> <sup>[3]</sup>	<b>53.7</b>
<b>Single-stage (End-to-end) Methods.</b>					
RRM [73] AAAI'2020	$\mathcal{I}$	WR38	62.6	62.9	–
AFA [52] CVPR'2022	$\mathcal{I}$	MiT-B1	66.0	66.3	38.9
RRM [73] <sup>†</sup> AAAI'2020	$\mathcal{I}$	ViT-B	63.1	62.4	–
ViT-PCM [51] ECCV'2022	$\mathcal{I}$	ViT-B	69.3	–	45.0
ToCo [53] CVPR'2023	$\mathcal{I}$	ViT-B	71.1	72.2	42.3
SeCo [69] CVPR'2024	$\mathcal{I}$	ViT-B	74.0	73.8	46.7
<b>CoSA</b>	$\mathcal{I}$	ViT-B	76.2	75.1 <sup>[4]</sup>	51.0
<b>CoSA*</b>	$\mathcal{I}$	ViT-B	<b>76.4</b>	<b>75.2</b> <sup>[5]</sup>	<b>51.1</b>

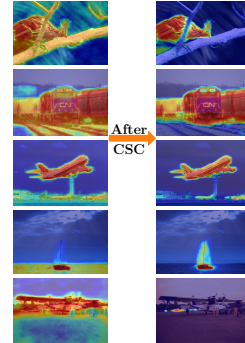
**Table 1: Weakly Supervised Semantic Segmentation Results.** *Sup.*: supervision type. *Net.*: segmentation backbone.  $\mathcal{F}$ : Fully supervised,  $\mathcal{I}$ : Image-level labels,  $\mathcal{S}$ : Saliency maps,  $\mathcal{L}$ : language models. \* represents CRF [8] postprocessing results.

**CAM Quality Comparison.** Tab. 2 shows CoSA’s CPL results compared with existing WSSS methods. Our method yields 78.5% and 76.4% mIoU on **train** and **val**. Notably, an ensemble of  $\mathcal{M}'$  and  $\mathcal{M}^{\dagger'}$  improves performance to 78.9% and 77.2%, suggesting the activation of  $\mathcal{M}'$  is orthogonal to that of  $\mathcal{M}^{\dagger'}$ .

**Qualitative Comparison.** Fig. 6 presents CAMs and segmentation visualizations, comparing with recent methods: MCT, BECO, and ToCo. As shown, our method can generate improved CAMs and produce well-aligned segmentation, exhibiting superior results in challenging segmentation problems with intra-class



**Fig. 6: Qualitative Comparison.** The results are reported on the val splits of VOC (in  $R1 - R3$ ) and COCO (in  $R4 - R6$ ). The official codebases and provided weights for MCT [65], BECO [50], and ToCo [53] are used for this comparison. (best viewed under zoom; see *Supp.* for more).

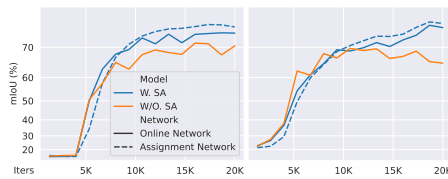


**Fig. 7: Effect of CSC.** The class activation for bird, train, plane, boat and car are presented from top to bottom.

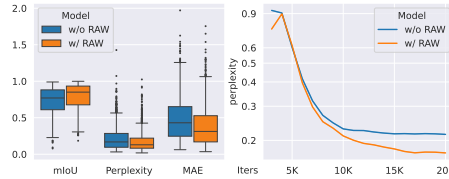
variation and occlusions. In addition, CoSA performs well *w.r.t.* the coexistence cases (Fig. 6  $R1$ ,  $R2$ ), while existing methods struggle. Moreover, CoSA reveals limitations in the GT segmentation (Fig. 6  $R4$ ).

## 4.2 Ablation Studies

**CoSA Module Analysis.** We begin by employing CAMs directly as the supervision signal for segmentation, akin to [73], albeit without refinement, and gradually apply CoSA modules to this baseline. As shown in Tab. 3(a), the mIoUs progressively improve with addition of our components. Further, we examine the efficacy of each CoSA component. As shown in Tab. 3(b), the elimination of each component results in deteriorated performance, most notably for CSC.



**Fig. 8: Ablative Study of SA.** The performance of SPL (*left*) and CPL (*right*) *w.r.t.* iterations on VOC val set are shown for CoSA with or without SA.



**Fig. 9: Ablation Study of RAW.** (*left*) boxplot of mIoU, perplexity and MAE to (1,0) for individual CPLs on VOC val. (*right*) perplexity reduction over times.

Method	ViT-PCM [51]	ACR-ViT [31]	CLIP-ES [42]	SeCo [69]	ToCo [53]	CoSA	CoSA <sup>•</sup>
train	71.4	70.9	75.0	76.5	73.6	78.5	<b>78.9</b>
val	69.3	—	—	—	72.3	76.4	<b>77.2</b>

**Table 2: Comparisons of CPL.** All methods use ViT as the backbone for generating the CAMs on VOC dataset. <sup>•</sup> represents the ensemble of  $\mathcal{M}'$  and  $\mathcal{M}^{t'}$  in CoSA.

(a)							(b)								
Base.	GC	SA	RAW	CSC	DT	mIoU (inc.)		CoSA	GC	SA	RAW	CSC	DT	mIoU (dec.)	
						VOC	COCO							VOC	COCO
✓						55.96	37.32	✓						<b>76.19</b>	<b>51.00</b>
✓	✓					63.09 (+7.13)	42.55 (+5.23)	✓					✗	75.54 (-0.65)	49.67 (-1.33)
✓	✓	✓				64.41 (+8.45)	43.92 (+6.60)	✓						69.89 (-6.30)	45.95 (-5.05)
✓	✓	✓	✓			68.22 (+12.26)	45.39 (+8.07)	✓					✗	72.45 (-3.74)	47.83 (-3.17)
✓	✓	✓	✓	✓		71.66 (+15.70)	47.10 (+9.78)	✓						72.10 (-4.09)	49.04 (-1.96)
✓	✓	✓	✓	✓	✓	75.54 (+19.58)	49.67 (+12.35)	✓	✗	✗				74.12 (-2.07)	49.67 (-1.33)
✓	✓	✓	✓	✓	✓	<b>76.19 (+20.23)</b>	<b>51.00 (+13.68)</b>								

**Table 3: Ablation Study on Contribution of Each Component.** (a): gradually add proposed components to baseline. (b): systematically exclude components from CoSA. **GC**: Guided CAMs, **SA**: Swapping Assignments, **RAW**: Reliability based Adaptive Weighting, **CSC**: Contrastive Separation in CAMs, and **DT**: Dynamic Threshold. **mIoU** is reported on PASCAL VOC12 and COCO val splits.

(a)				(b)			(c)			
Source	Detach	train	val	Method	C-mIoU	mIoU	Use CRF?	mIoU(%)		Speed
GT	None	83.99	80.16	FPR [9]	53.09	53.34	✓	✓	×	✓
NO	–	72.28	71.38	ToCo [53]	63.62	72.33		72.1	70.9(-1.2)	1.95 4.94
SPL	$F$	74.19	73.36	SeCo [69]	73.18	73.63		76.5	76.4(-0.1)	2.36 <b>9.60</b>
SPL	$W_{fc}$	78.05	76.15	w/o CSC	62.61	67.82		71.1	69.2(-1.9)	1.82 3.99
SPL	None	<b>78.54</b>	<b>76.37</b>	w/ CSC	<b>82.34</b>	<b>76.37</b>		76.4	76.2(-0.2)	1.83 <b>4.11</b>

**Table 4: Ablation Study of GC, CSC and CRF.** (a): CPL performance comparison on VOC. **Source**: guidance sources. **Detach**: stop gradient in GC for feature map  $F$  or  $W_{fc}$ . (b): CPL performance comparison. FPR, ToCo and SeCo results are based on their code and weights. **C-mIoU**: mIoU for classes with coexistence. (c): CRF Impact. Best speed-accuracy tradeoff is achieved without using CRF. Inference speeds (FPS) are tested on RTX 3090.

**Impact of Guided CAMs.** Our model is compared with a baseline [73] that directly uses CAMs as CPL. As shown in Tab. 4(a), our guided CAMs notably enhance CPL quality by 6.26% and 4.99% for **train** and **val** splits. Further, we conduct experiments to ascertain the extent to which the two CAM components, feature  $F$  and weights  $W_{fc}$ , exert greater impact on guiding CAMs. As shown, the detachement of  $F$  results in 74.19% and 73.36%, but  $W_{fc}$  can decrease the results slightly to 78.05% and 76.37%. This suggests that guiding CAMs primarily optimizes feature maps, verifying our hypothesis of the non-deterministic feature map contributing to the stochasticity of CAMs in Sec. 3.1.

**Impact of Swapping Assignments (SA).** Tab. 3(b) suggests that eliminating SA results in significant mIoU decreases, highlighting the importance of this training strategy. Further examination of the ON and AN *w.r.t.* SPL and CPL indicates that, in later training stages, AN consistently outperforms ON for both SPL and CPL, as shown in Fig. 8, due to AN performing a form of model ensembling similar to Polyak-Ruppert averaging [49, 54]. We observe a noticeable disparity of mIoUs between two ONs (solid orange line *vs.* solid blue line in Fig. 8), which may be attributed to the superior quality of CPL and SPL from the AN facilitating a more robust ON for CoSA. The momentum framework, originally introduced to mitigate noise and fluctuations of the online learning target [6, 22], is used for info exchange across CAMs and segmentation in CoSA.

**Impact of RAW.** Tab. 3(b) shows notable mIoU reduction without RAW. We conduct further studies to investigate its effect on perplexity reduction. The boxplot in Fig. 9 suggests that RAW leads to higher mIoU but lower perplexity. Fig. 9(right) illustrates a faster decrease in perplexity when RAW is used, affirming its impact on perplexity reduction.

**Impact of CSC.** Our CSC is introduced to address the coexistence issues. We establish C-mIoU to measure the CAM quality for those coexistence-affected classes. As shown in Tab. 4(b), applying CSC sees a boost in C-mIoU and mIoU, which surpass the existing methods. Some visual comparisons are given in Fig. 7.

**Impact of Dynamic Threshold.** We evaluate CoSA using some predetermined thresholds, comparing them with one employing dynamic threshold on VOC val split (see *Supp.* for results). The performance is sensitive to the threshold, but dynamic thresholding achieves 0.65% increased performance over the best manual finetuning while saving 80% of hyper-parameter searching time.

### 4.3 Further Analysis

**Training and Inference Efficiency Analysis.** Unlike multi-stage approaches, CoSA can be trained end-to-end efficiently. Compared to BECO [50], our method is 240% faster in training, uses 50% fewer parameters, and yields a 4.3% higher mIoU on VOC test. Please refer to *Supp.* for more discussion. At inference time, we find that CRF post-processing, which is commonly adopted for refining masks [15, 50] or the CAMs [51, 65, 73], can greatly slow down the inference speed. As our aim is to develop a fully end-to-end WSSS solution, incorporating CRF post-processing contradicts this principal. Through our experiments, we show that CoSA does not heavily depend on CRF: incorporating CRF results in marginal improvement of 0.2%, 0.1%, and 0.1% for VOC val, VOC test, and COCO val, respectively (Tab. 1). Conversely, eliminating CRF in CoSA can greatly speed up inference (a noteworthy 307% and 165%  $\uparrow$ ) and achieve better speed-accuracy tradeoff as suggested in Tab. 4(c).

**Hyper-parameter Sensitivity.** We apply grid search strategy to explore the hyper-parameters in CoSA. The analysis of parameters, such as *perplexity filter*, *loss weights*, *momentum* are discussed in *Supp.* CoSA maintains consistent with variations of those parameters, which demonstrate its robustness.

## 5 Conclusion

This paper presents an end-to-end WSSS method: Co-training with Swapping Assignments (CoSA), which eliminates the need for CAM refinement and enables concurrent CAM and segmentation optimization. Our empirical study reveals the non-deterministic behaviors of CAMs and that proper guidance can mitigate such stochasticity, leading to substantial quality enhancement. We propose explicit CAM optimization leveraging segmentation pseudo-labels in our

approach, where a dual-stream model comprising an online network for predicting CAMs and segmentation masks, and an ancillary assignment network providing swapped assignments (SPL and CPL) for training, is introduced. We further propose three techniques within this framework: RAW, designed to mitigate the issue of unreliable pseudo-labels; contrastive separation, aimed at resolving coexistence problems; and a dynamic threshold search algorithm. Incorporating these techniques, CoSA outperforms all SOTA methods on both VOC and COCO WSSS benchmarks while achieving exceptional speed-accuracy trade-off.

## Acknowledgments

The work is supported by European Research Council under the European Union’s Horizon 2020 research and innovation programme (GA No 787768).

## References

1. Ahn, J., Cho, S., Kwak, S.: Weakly supervised learning of instance segmentation with inter-pixel relations. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 2209–2218 (2019)
2. Ahn, J., Kwak, S.: Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4981–4990 (2018)
3. Araslanov, N., Roth, S.: Single-stage semantic segmentation from image labels. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4253–4262 (2020)
4. Bearman, A., Russakovsky, O., Ferrari, V., Fei-Fei, L.: What’s the point: Semantic segmentation with point supervision. In: European Conference on Computer Vision (ECCV). pp. 549–565. Springer (2016)
5. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. *Neural Information Processing Systems (NeurIPS)* **33**, 9912–9924 (2020)
6. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 9650–9660 (2021)
7. Chattopadhyay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 839–847. IEEE (2018)
8. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **40**(4), 834–848 (2017)
9. Chen, L., Lei, C., Li, R., Li, S., Zhang, Z., Zhang, L.: Fpr: False positive rectification for weakly supervised semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). pp. 1108–1118 (2023)
10. Chen, L., Wu, W., Fu, C., Han, X., Zhang, Y.: Weakly supervised semantic segmentation with boundary exploration. In: European Conference on Computer Vision (ECCV). pp. 347–362. Springer (2020)

11. Chen, Q., Yang, L., Lai, J.H., Xie, X.: Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 4288–4298 (2022)
12. Chen, Z., Tian, Z., Zhu, J., Li, C., Du, S.: C-cam: Causal cam for weakly supervised semantic segmentation on medical image. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 11676–11685 (2022)
13. Chen, Z., Sun, Q.: Extracting class activation maps from non-discriminative features as well. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 3135–3144 (2023)
14. Chen, Z., Wang, T., Wu, X., Hua, X.S., Zhang, H., Sun, Q.: Class re-activation maps for weakly-supervised semantic segmentation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 969–978 (2022)
15. Cheng, Z., Qiao, P., Li, K., Li, S., Wei, P., Ji, X., Yuan, L., Liu, C., Chen, J.: Out-of-candidate rectification for weakly supervised semantic segmentation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 23673–23684 (2023)
16. Dai, J., He, K., Sun, J.: Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: *IEEE International Conference on Computer Vision (ICCV)*. pp. 1635–1643 (2015)
17. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: *International Conference on Learning Representations (ICLR)*(2021)
18. Du, Y., Fu, Z., Liu, Q., Wang, Y.: Weakly supervised semantic segmentation by pixel-to-prototype contrast. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 4320–4329 (2022)
19. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. *International journal of computer vision (IJCV)* **88**, 303–338 (2010)
20. Fan, J., Zhang, Z., Tan, T., Song, C., Xiao, J.: Cian: Cross-image affinity net for weakly supervised semantic segmentation. In: *AAAI Conference on Artificial Intelligence (AAAI)*. vol. 34, pp. 10762–10769 (2020)
21. Gao, W., Wan, F., Pan, X., Peng, Z., Tian, Q., Han, Z., Zhou, B., Ye, Q.: Ts-cam: Token semantic coupled attention map for weakly supervised object localization. In: *IEEE International Conference on Computer Vision (ICCV)*. pp. 2886–2895 (2021)
22. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. *Neural Information Processing Systems (NeurIPS)***33**, 21271–21284 (2020)
23. Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: *IEEE International Conference on Computer Vision (ICCV)*. pp. 991–998. IEEE (2011)
24. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 447–456 (2015)
25. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015)
26. Jiang, P.T., Yang, Y., Hou, Q., Wei, Y.: L2g: A simple local-to-global knowledge transfer framework for weakly supervised semantic segmentation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. pp. 16886–16896 (2022)

27. Ke, T.W., Hwang, J.J., Yu, S.: Universal weakly supervised segmentation by pixel-to-segment contrastive learning. In: International Conference on Learning Representations (ICLR)(2020)
28. Khoreva, A., Benenson, R., Hosang, J., Hein, M., Schiele, B.: Simple does it: Weakly supervised instance and semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 876–885 (2017)
29. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. *Neural Information Processing Systems (NeurIPS)* **24** (2011)
30. Kweon, H., Yoon, S.H., Kim, H., Park, D., Yoon, K.J.: Unlocking the potential of ordinary classifier: Class-specific adversarial erasing framework for weakly supervised semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). pp. 6994–7003 (2021)
31. Kweon, H., Yoon, S.H., Yoon, K.J.: Weakly supervised semantic segmentation via adversarial learning of classifier and reconstructor. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 11329–11339 (2023)
32. Lee, J., Kim, E., Lee, S., Lee, J., Yoon, S.: Ficklenet: Weakly and semi-supervised semantic image segmentation using stochastic inference. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 5267–5276 (2019)
33. Lee, J., Kim, E., Mok, J., Yoon, S.: Anti-adversarially manipulated attributions for weakly supervised semantic segmentation and object localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*(2022)
34. Lee, J., Kim, E., Yoon, S.: Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4071–4080 (2021)
35. Lee, J., Oh, S.J., Yun, S., Choe, J., Kim, E., Yoon, S.: Weakly supervised semantic segmentation using out-of-distribution data. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 16897–16906 (2022)
36. Lee, J., Yi, J., Shin, C., Yoon, S.: Bbam: Bounding box attribution map for weakly supervised semantic and instance segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 2643–2652 (2021)
37. Lee, S., Lee, M., Lee, J., Shim, H.: Railroad is not a train: Saliency as pseudo-pixel supervision for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 5495–5505 (2021)
38. Li, J., Fan, J., Zhang, Z.: Towards noiseless object contours for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 16856–16865 (2022)
39. Li, J., Jie, Z., Wang, X., Ma, L., et al.: Expansion and shrinkage of localization for weakly-supervised semantic segmentation. In: *Neural Information Processing Systems (NeurIPS)*(2022)
40. Li, Y., Duan, Y., Kuang, Z., Chen, Y., Zhang, W., Li, X.: Uncertainty estimation via response scaling for pseudo-mask noise mitigation in weakly-supervised semantic segmentation. In: *AAAI Conference on Artificial Intelligence (AAAI)*. vol. 36, pp. 1447–1455 (2022)
41. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: *European Conference on Computer Vision (ECCV)*. pp. 740–755. Springer (2014)
42. Lin, Y., Chen, M., Wang, W., Wu, B., Li, K., Lin, B., Liu, H., He, X.: Clip is also an efficient segmenter: A text-driven approach for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 15305–15314 (2023)

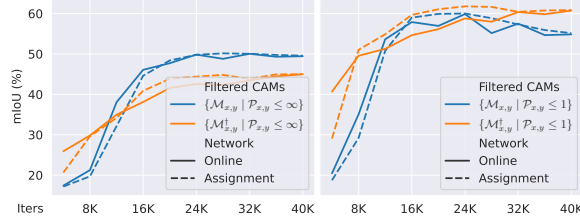
43. Liu, S., Liu, K., Zhu, W., Shen, Y., Fernandez-Granda, C.: Adaptive early-learning correction for segmentation from noisy annotations. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 2606–2616 (2022)
44. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: IEEE International Conference on Computer Vision (ICCV). pp. 10012–10022 (2021)
45. Liu, Z., Qi, X., Fu, C.W.: One thing one click: A self-training approach for weakly supervised 3d semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 1726–1736 (2021)
46. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
47. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
48. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)
49. Polyak, B.T., Juditsky, A.B.: Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization* **30**(4), 838–855 (1992)
50. Rong, S., Tu, B., Wang, Z., Li, J.: Boundary-enhanced co-training for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 19574–19584 (2023)
51. Rossetti, S., Zappia, D., Sanzari, M., Schaerf, M., Pirri, F.: Max pooling with vision transformers reconciles class and shape in weakly supervised semantic segmentation. In: European Conference on Computer Vision (ECCV). pp. 446–463. Springer (2022)
52. Ru, L., Zhan, Y., Yu, B., Du, B.: Learning affinity from attention: end-to-end weakly-supervised semantic segmentation with transformers. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 16846–16855 (2022)
53. Ru, L., Zheng, H., Zhan, Y., Du, B.: Token contrast for weakly-supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 3093–3102 (2023)
54. Ruppert, D.: Efficient estimations from a slowly convergent robbins-monro process. Tech. rep., Cornell University Operations Research and Industrial Engineering (1988)
55. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: IEEE International Conference on Computer Vision (ICCV). pp. 618–626 (2017)
56. Shimoda, W., Yanai, K.: Self-supervised difference detection for weakly-supervised semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). pp. 5208–5217 (2019)
57. Song, C., Huang, Y., Ouyang, W., Wang, L.: Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 3136–3145 (2019)
58. Sun, K., Shi, H., Zhang, Z., Huang, Y.: Ecs-net: Improving weakly supervised semantic segmentation by using connections between class activation maps. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 7283–7292 (2021)
59. Wang, C., Xu, R., Xu, S., Meng, W., Zhang, X.: Treating pseudo-labels generation as image matting for weakly supervised semantic segmentation. In: IEEE International Conference on Computer Vision (ICCV). pp. 755–765 (2023)



60. Wang, Y., Zhang, J., Kan, M., Shan, S., Chen, X.: Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 12275–12284 (2020)
61. Wu, Z., Shen, C., Van Den Hengel, A.: Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition* **90**, 119–133 (2019)
62. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: European Conference on Computer Vision (ECCV). pp. 418–434 (2018)
63. Xie, J., Hou, X., Ye, K., Shen, L.: Clims: Cross language image matching for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4483–4492 (2022)
64. Xie, J., Xiang, J., Chen, J., Hou, X., Zhao, X., Shen, L.: C2am: Contrastive learning of class-agnostic activation map for weakly supervised object localization and semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 989–998 (2022)
65. Xu, L., Ouyang, W., Bennamoun, M., Boussaid, F., Xu, D.: Multi-class token transformer for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4310–4319 (2022)
66. Xu, L., Ouyang, W., Bennamoun, M., Boussaid, F., Xu, D.: Learning multi-modal class-specific tokens for weakly supervised dense object localization. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 19596–19605 (2023)
67. Xu, R., Wang, C., Sun, J., Xu, S., Meng, W., Zhang, X.: Self correspondence distillation for end-to-end weakly-supervised semantic segmentation. In: AAAI Conference on Artificial Intelligence (AAAI). AAAI Press (2023)
68. Yang, X., Burghardt, T., Mirmehdi, M.: Dynamic curriculum learning for great ape detection in the wild. *International Journal of Computer Vision (IJCV)* **131**(5), 1163–1181 (2023)
69. Yang, Z., Fu, K., Duan, M., Qu, L., Wang, S., Song, Z.: Separate and conquer: Decoupling co-occurrence via decomposition and representation for weakly supervised semantic segmentation. *arXiv preprint arXiv:2402.18467* (2024)
70. Yoon, S.H., Kweon, H., Cho, J., Kim, S., Yoon, K.J.: Adversarial erasing framework via triplet with gated pyramid pooling layer for weakly supervised semantic segmentation. In: European Conference on Computer Vision (ECCV). pp. 326–344. Springer (2022)
71. Yu, L., Xiang, W., Fang, J., Chen, Y.P.P., Chi, L.: ex-vit: A novel explainable vision transformer for weakly supervised semantic segmentation. *Pattern Recognition* p. 109666 (2023)
72. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European Conference on Computer Vision (ECCV). pp. 818–833. Springer (2014)
73. Zhang, B., Xiao, J., Wei, Y., Sun, M., Huang, K.: Reliability does matter: An end-to-end weakly supervised semantic segmentation approach. In: AAAI Conference on Artificial Intelligence (AAAI). vol. 34, pp. 12765–12772 (2020)
74. Zhang, X., Wei, Y., Feng, J., Yang, Y., Huang, T.S.: Adversarial complementary learning for weakly supervised object localization. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 1325–1334 (2018)
75. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 2921–2929 (2016)
76. Zhou, T., Zhang, M., Zhao, F., Li, J.: Regional semantic contrast and aggregation for weakly supervised semantic segmentation. In: IEEE Computer Vision and Pattern Recognition (CVPR). pp. 4299–4309 (2022)

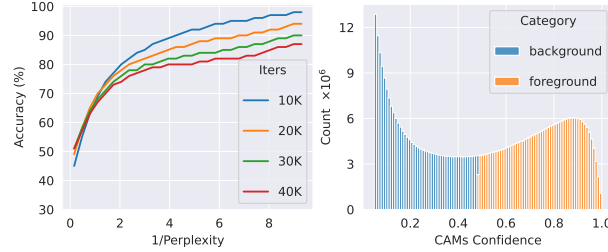
## A Further Analysis

**Contrastive Separation Analysis.** Fig. 10 shows the analysis of contrastive separation on COCO. A similar trend between  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  is observed as on VOC. This suggests that the distinct relationship between  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  extends beyond the VOC dataset to encompass a broader dataset as well.



**Fig. 10:  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  Comparisons on COCO.** (left): mIoU vs. time-steps for  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  on COCO val. (right): same as (left) but filtered by perplexity.

**CPL Analysis.** While the proposed dynamic learning techniques in Sec. 3.3, namely reliability based adaptive weighting and dynamic threshold, were originally inspired by the VOC dataset, we demonstrate their applicability on COCO here. As shown in Fig. 11(left), the negative correlation between perplexity and accuracy remains significant across various training time-steps, highlighting our purposed perplexity estimation method can be used to penalize inaccurate regions in our CAM2Seg loss. Fig. 11(right) also suggests that the confidence distribution on the COCO dataset shows discernible clusters.



**Fig. 11: CPL Analysis on COCO** (left) correlation between perplexity and accuracy of CPL for different time-steps. (right) distribution of CAMs' confidence categorized by the proposed dynamic threshold

**Impact of CRF.** The conditional random field (CRF) proposes to optimize the segmentation by utilizing the low-level information obtained from the local interactions of pixels and edges [8]. Traditionally, a manually designed CRF

postprocessing step has been widely adopted for refining segmentation [15, 50] or CAMs [51, 65, 73]. As our aim is to develop a fully end-to-end WSSS solution, incorporating CRF postprocessing contradicts this principal. Through our experiments, we demonstrate that CoSA, unlike other single-stage methods, does not heavily depend on CRF. Our results indicate that incorporating CRF results in marginal improvement of 0.2%, 0.1%, and 0.1% for VOC **val**, VOC **test**, and COCO **val**, respectively, as presented in Tab. 2 of the main paper. Tab. 5(a) suggests that in comparison to other SOTA models, our CoSA exhibits a lesser dependency on the CRF postprocessing. On the contrary, eliminating the CRF step leads to a noteworthy enhancement of 165% in terms of inference speed, as demonstrated in Tab. 5(b).

(a)			(b)	
Method	w/o CRF	w/ CRF	CoSA	Speed
AFA [52]	63.8	66.0 (+2.2)	w/o CRF	4.11 imgs/s
VIT-PCM [51]	67.7	71.4 (+3.7)	w/ CRF	1.83 imgs/s
ToCo [53]	69.2	71.1 (+0.9)		
SeCo [69]	72.2	73.7 (+1.5)		
CoSA	76.2	76.4 (+0.2)		

**Table 5: CRF Impact.** (a): Comparisons of CRF impact on SOTA single-stage WSSS methods on VOC **val**. (b): Inference speed with and without CRF. Speed tested using a RTX 3090.

**Training Efficiency Study.** Unlike multi-stage approaches, CoSA is extremely efficient in training. It can be trained end-to-end efficiently. When training a semantic segmentation model with weak labels on the VOC dataset, our method requires a mere 8.7 hours of training time and a total of 92M parameters. In contrast, MCT [65] would necessitate approximately 231% more time (20.1hrs  $\uparrow$ ) and 173% more parameters (159M  $\uparrow$ ) for the same task, and BECO [50] would require around 240% more time (20.9hrs  $\uparrow$ ) and 50% more parameters (46M  $\uparrow$ ). When compared to the single-stage method, CoSA also demonstrate its advantage in speed-accuracy trade-off. Further details regarding the efficiency study can be found in Tab. 6.

## B Further Implementation Details

**CoSA Implementation Details.** For image preprocessing, weak transformation  $\mathcal{T}_w$  and strong transformation  $\mathcal{T}_s$  are employed in CoSA for the input of assignment network and online network, respectively.  $\mathcal{T}_w$  and  $\mathcal{T}_s$  details are given in Tab. 12 and Tab. 13. Following [53], we use the multi-scale inference in assignment network to produce CPL and SPL. For VOC training, CoSA is warmed

	CAMs Generation	CAMs Refinement	Seg. Training	Total	mIoU
MCT [65]	2.2hrs (21M)	11.1hrs (106M)	15.5hrs (124M)	28.8hrs (251M)	71.6
BECO [50]	0.9hrs (23M)	6.5hrs (24M)	22.2hrs (91M)	29.6hrs (138M)	71.8
ToCo [53]		9.9hrs (98M)		9.9hrs (98M)	72.2
SeCo [69]		8.8hrs (98M)		8.8hrs (98M)	73.8
CAMs and Seg. Co-optimization					
CoSA		8.7hrs (92M)		<b>8.7hrs (92M)</b>	<b>75.1</b>

**Table 6: Training Speed and Parameters Comparisons.** We report the detailed training time, parameters and final mIoU on VOC **test** split for MCT, BECO, ToCo and our CoSA. All methods are tested using the same machine with a single 3090 GPU. The official MCT, BECO and ToCo code repositories are utilized in this study.

up with 6K iterations, where  $\lambda_{c2s}$ ,  $\lambda_{c2s}$ , and  $\lambda_{csc}$  are set to 0. In practice, we train CoSA for 20K iterations on 2 GPUs, with 2 images per GPU, or for 40K iterations on 1 GPU for some ablation experiments. For COCO training, CoSA is warmed up with 10K iterations and is trained on 2 GPUs, handling 4 images per GPU across 40K iterations.

**CoSA-MS Implementation Details.** Tab. 2 in the main paper presents the segmentation results of the multi-stage version of our approach, known as CoSA-MS. In those experiments, we leverage the CAM pseudo-labels generated by our CoSA to *directly* train standalone segmentation networks. It is important to note that we do not use PSA [2], which is widely used in [15, 65], nor IRN [1], extensively used in [13, 31, 50, 59], for CPL post-refinement. For our R101 segmentation network, we use a ResNet101 version of DeepLabV3+ model, same as BECO [50]. As for the CoSA-MS with WR38 network, we utilize a encoder-decoder framework, where encoder is WideResNet38 [61] and decoder is LargeFoV [8], following the final step described in MCT [65]. Regarding the SWIN implementation, we use the SWIN-Base encoder [44] in conjunction with UperNet decoder [62], following the description in [13, 14].

**Training Pseudo Code.** we present the pseudo code for training CoSA in Algorithm 1.

## C Additional Results

### C.1 Contribution of each Component

In addition to the module analysis in Sec. 4.2, we further present additional ablation studies regarding the impact of each component on the baseline. A one-stage WSSS framework, proposed in RRM [73], is utilized as the baseline model, albeit without the inclusion of the offline refinement module. As illustrated in Tab. 7, it is evident that all components proposed in CoSA exhibit positive effects on the baseline model.

**Algorithm 1** CoSA Training Pseudo Code

---

```

1: Require:  $\mathcal{D}$  ▷ image-level classification dataset
2: Require:  $\mathcal{F}_\Theta, \mathcal{F}_{\Theta'}$  ▷ online network parameterized by  $\Theta$  and assignment network by  $\Theta'$ 
3:  $\mathcal{F}_\Theta \leftarrow \text{Init}, \mathcal{F}_{\Theta'} \leftarrow \text{Init}$  ▷ initialize networks with pretrained backbone
4: do
5:    $x, Y_{\text{gt}} \leftarrow \text{Sample}(\mathcal{D})$  ▷ sample a mini-batch of image and weak-label pairs
6:    $x_s, x_w \leftarrow \mathcal{T}_s(x), \mathcal{T}_w(x)$  ▷ apply strong and weak augmentations
7:    $\{x_w^s\} \leftarrow \text{multiscale}(x_w)$  ▷ generate a set of  $x_w$  with different scales
8:    $\{\mathcal{M}', \mathcal{M}^{\dagger'}, S'\} \leftarrow \mathcal{F}_{\Theta'}(\{x_w^s\})$  ▷ forward a set of  $x_w$  in assignment network
9:    $\mathcal{M}', \mathcal{M}^{\dagger'}, S' \leftarrow \text{Maxpool}(\{\mathcal{M}'\}), \text{Maxpool}(\{\mathcal{M}^{\dagger'}\}), \text{Avgpool}(\{S'\})$  ▷ ensemble multiscale
   outputs
10:   $\mathcal{M}', \mathcal{M}^{\dagger'}, S' \leftarrow \text{Filter}(\mathcal{M}', \mathcal{M}^{\dagger'}, S')$  ▷ filter CAMs and segmentation prediction with
    $Y_{\text{gt}}$ 
11:   $Z, Z^\dagger, \mathcal{M}, \mathcal{M}^\dagger, S \leftarrow \mathcal{F}_\Theta(x_s)$  ▷ forward  $x_s$  in online network
12:   $\mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{cls}}^{\mathcal{M}^\dagger} \leftarrow \mathcal{L}_{\text{cls}}(Z, Y_{\text{gt}}) + \mathcal{L}_{\text{cls}}(Z^\dagger, Y_{\text{gt}})$  ▷ get classification losses for  $\mathcal{M}$  and  $\mathcal{M}^\dagger$  by eq.
   (1)
13:   $\xi^* \leftarrow \text{solve eq. (6) with } \mathcal{M}'$  ▷ get dynamic threshold
14:   $\hat{Y}^{\text{CPL}} \leftarrow \text{eq. (2) with } \mathcal{M}', \xi^*$  ▷ obtain CPL
15:   $\mathcal{P} \leftarrow \text{eq. (4) with } \mathcal{M}', \xi^*$  ▷ estimate perplexity score
16:   $\mathcal{L}_{c2s} \leftarrow \text{eq. (5) with } \hat{Y}^{\text{CPL}}, S, \mathcal{P}$  ▷ get CAM2seg loss
17:   $\mathcal{L}_{c2s}^{\mathcal{M}^\dagger} \leftarrow \text{follow 14 - 17 but with } \mathcal{M}^{\dagger'}$  ▷ get another CAM2seg loss
18:   $\hat{Y}^{\text{SPL}} \leftarrow \text{eq. (7) with } S'$  ▷ obtain SPL
19:   $\mathcal{L}_{s2c} \leftarrow \text{eq. (8) with } \hat{Y}^{\text{SPL}}, \mathcal{M}$  ▷ get Seg2CAM loss
20:   $\mathcal{R}^+, \mathcal{R}^- \leftarrow \text{eq. (9) with } \mathcal{P}, \hat{Y}^{\text{CPL}}$  ▷ define positive and negative correlation matrix
21:   $\mathcal{L}_{\text{csc}} \leftarrow \text{eq. (10) with } \mathcal{M}, \mathcal{R}^+, \mathcal{R}^-$  ▷ get contrastive seperation loss
22:   $\mathcal{L}_{\text{CoSA}} \leftarrow \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{cls}}^{\mathcal{M}^\dagger} + \lambda_{c2s}(\mathcal{L}_{c2s} + \mathcal{L}_{c2s}^{\mathcal{M}^\dagger}) + \lambda_{s2c}\mathcal{L}_{s2c} + \lambda_{\text{csc}}\mathcal{L}_{\text{csc}}$  ▷ weighted sum as the
   overall training objective
23:   $\Delta\Theta \leftarrow -\nabla_{\mathcal{L}_{\text{CoSA}}} \Theta$  ▷ backpropagate the overall loss
24:   $\Theta \leftarrow \Theta + \Delta\Theta$  ▷ undate online network with gradient
25:   $\Theta' \leftarrow m\Theta' + (1 - m)\Theta$  ▷ undate assignment network via EMA
26: until  $\mathcal{L}_{\text{CoSA}}$  converge
27: end

```

---

**C.2 Hyper-parameter Finetuning**

we examine the impact of hyper-parameter variation with CoSA resulting from our finetuning. The fine-tuning of each hyper-parameter is demonstrate with the remaining parameters fixed at their determined optimal values. Those hyper-parameters were tuned on VOC val set, and the SOTA results on COCO were achieved with the same hyper-parameters, except batch size and number of iterations were tweaked to accommodate the dataset size difference.

**Loss Weights.** We demonstrate the finetuning of the Seg2CAM and CAM2Seg loss weights in Tab. 8(a)(b). A significant mIoU decrease is observed as  $\lambda_{c2s}$  reduces the influence of the segmentation branch, as expected. The mIoU reaches its peak when  $\lambda_{s2c}=0.05$  and  $\lambda_{c2s}=0.1$ .

**Low-perplexity Filter.** We finetune the coefficient for the low-pass perplexity filter  $\epsilon$ , described in eq. (9) of the main paper. The corresponding findings are illustrated in Tab. 8(c). Optimum performance is obtained when  $\epsilon$  is set to 1, either decreasing or increasing this value can impair the performance of our model.

**EMA Momentum.** Here, the momentum used for updating the assignment network is finetuned. Results presented in Tab. 8(d) indicate that the optimal

Base.	GC	SA	RAW	CSC	DT	mIoU @ val (increase)	
						VOC	COCO
✓						55.96	37.32
✓	✓					63.09 (+7.13)	42.55 (+5.23)
✓	✓	✓				64.41 (+8.45)	43.92 (+6.60)
✓			✓			67.04 (+11.08)	42.65 (+5.33)
✓				✓		69.24 (+13.27)	44.11 (+6.79)
✓					✓	61.80 (+5.84)	43.28 (+5.96)

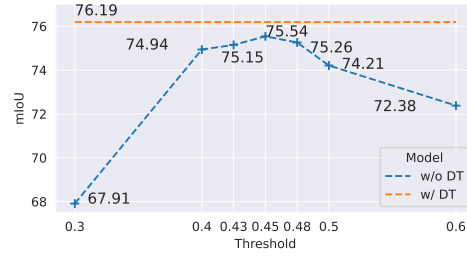
**Table 7: Contribution of Each Component.** We systematically include the proposed components on the baseline (**Base.**). **GC**: Guided CAMs, **SA**: Swapping Assignments, **RAW**: Reliability based Adaptive Weighting, **CSC**: Contrastive Separation in CAMs, and **DT**: Dynamic Threshold.

performance is achieved when  $m = 0.9994$ . Additionally, we find that setting  $m = 1$  freezes the assignment network, breaking the training of online network and leading to framework collapse.

**Fixed Threshold vs. Dynamic Threshold.** We evaluate CoSA with predetermined thresholds. The results are presented in Fig. 12. As shown, the performance peaks when this threshold is set to 0.45, with an mIoU of 75.54%. However, our dynamic threshold can outperform the best manual finetuning by 0.65%. Despite the incurred additional 10% computation overhead, our threshold searching algorithm obviates time-consuming finetuning efforts, resulting in nearly 80% reduction in hyper-parameter searching time in this case and  $(1 - 1/n^{-1})\%$  in general where  $n$  thresholds are considered. In addition, the adoption of dynamic thresholding can enhance the generalizability to novel datasets.

(a) Seg2CAM	(b) CAM2Seg	(c) Perplexity	(d) Momentum $m$
weight $\lambda_{s2c}$	weight $\lambda_{c2s}$	filter $\epsilon$	$m$   mIoU
$\lambda_{s2c}$   mIoU	$\lambda_{c2s}$   mIoU	$\epsilon$   mIoU	
0.2   73.79	0.4   74.67	$\infty$   73.66	0.9990   73.79
0.1   74.67	0.2   75.56	2   75.52	0.9992   75.40
0.05   <b>76.19</b>	0.1   <b>76.19</b>	1   <b>76.19</b>	0.9994   <b>76.19</b>
0.025   75.25	0.05   73.95	0.5   74.30	0.9996   75.58
0.0125   74.33	0.025   61.55	0.1   70.63	0.9999   71.42
			1.0000   15.99

**Table 8: Hyper-parameter Finetuning Results.** Parameter searching for (a) Loss weight for CAM2Seg  $\lambda_{c2s}$ ; (b) Loss weight for Seg2CAM  $\lambda_{s2c}$ ; (c) Low-pass perplexity filter coefficient  $\epsilon$ ; (e) EMA Momentum  $m$  for updating assignment network. **mIoU** represents semantic segmentation result on PASCAL VOC val split.



**Fig. 12: Threshold finetuning.** (left) determined dynamic threshold during training. (right) mIoU comparison of fixed threshold *vs.* the purposed dynamic threshold on VOC val.

### C.3 Per-class Segmentation Comparisons

We show the per-class semantic segmentation results on VOC val and test splits as well as COCO val split.

**Comparisons on VOC.** Tab. 9 illustrates the CoSA per-class mIoU results compared with recent works: AdvCAM [33], MCT [65], ToCo [53], Xu *et al.* [66], BECO [50]. To be fair in comparison, we include CoSA with CRF [8] postprocessing results, denoted as CoSA\*, same as other SOTA models. Notably, CoSA dominates in 10 out of 21 classes. In particular, categories like boat (5.9% ↑), chair (8.2% ↑), and sofa (17.2% ↑), demonstrate substantial lead over the SOTA models. In the VOC test split (depicted in Tab. 10), we still observe its superiority over other SOTA methods, where CoSA dominates in 15 out of 21 classes.

**Comparisons on COCO.** We compare CoSA with recent WSSS works for individual class performance on the COCO val set. As illustrated in Tab. 11, CoSA outperforms its counterparts in 56 out of 81 classes. Particularly, classes such as truck (10.6% ↑), tie (14.3% ↑), kite (12.4% ↑), baseball glove (20.3% ↑), knife (14.5% ↑), (10.6% ↑), carrot (13.0% ↑), donuts (10.0% ↑), couch (13.9% ↑), oven (13.0% ↑), and toothbrush (10.0% ↑) exhibit remarkable leading performance.

Method	bkg	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow
AdvCAM [34] CVPR21	90.0	79.8	34.1	82.6	63.3	70.5	89.4	76.0	87.3	31.4	81.3
MCT [65] CVPR22	91.9	78.3	39.5	<b>89.9</b>	55.9	76.7	81.8	79.0	90.7	32.6	87.1
ToCo [53] CVPR23	91.1	80.6	<b>48.7</b>	68.6	45.4	<b>79.6</b>	87.4	83.3	89.9	35.8	84.7
Xu <i>et al.</i> [66] CVPR23	92.4	84.7	42.2	85.5	64.1	77.4	86.6	82.2	88.7	32.7	83.8
BECO [50] CVPR23	91.1	81.8	33.6	87.0	63.2	76.1	<b>92.3</b>	<b>87.9</b>	<b>90.9</b>	39.0	<b>90.2</b>
CoSA* (Ours)	<b>93.1</b>	<b>85.5</b>	48.5	88.7	<b>70.0</b>	77.6	90.4	86.4	90.3	<b>47.2</b>	88.7
Method	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mIoU
AdvCAM [34] CVPR21	33.1	82.5	80.8	74.0	72.9	50.3	82.3	42.2	<b>74.1</b>	52.9	68.1
MCT [65] CVPR22	57.2	87.0	84.6	77.4	79.2	55.1	89.2	47.2	70.4	58.8	71.9
ToCo [53] CVPR23	<b>60.5</b>	83.7	83.7	76.8	83.0	56.6	87.9	43.5	60.5	63.1	71.1
Xu <i>et al.</i> [66] CVPR23	59.0	82.4	80.9	76.1	81.4	48.0	88.2	46.4	70.2	62.5	72.2
BECO [50] CVPR23	41.6	85.9	86.3	<b>81.8</b>	76.7	<b>56.7</b>	89.5	54.7	64.3	60.6	72.9
CoSA* (Ours)	54.1	<b>87.3</b>	<b>87.1</b>	79.6	<b>85.6</b>	53.2	<b>89.9</b>	<b>71.9</b>	65.1	<b>63.4</b>	<b>76.4</b>

**Table 9: Per-class Segmentation on VOC val Split.** Comparison of per-class segmentation results on VOC val. CoSA is compared with AdvCAM, MCTformer, ToCo, Xu *et al.* and BECO. Best results are in **bold**.

Method	bkg	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow
AdvCAM [34] CVPR21	90.1	81.2	33.6	80.4	52.4	66.6	87.1	80.5	87.2	28.9	80.1
MCT [65] CVPR22	90.9	76.0	37.2	79.1	54.1	69.0	78.1	78.0	86.1	30.3	79.5
ToCo [53] CVPR23	91.5	<b>88.4</b>	<b>49.5</b>	69.0	41.6	72.5	87.0	80.7	88.6	32.2	85.0
CoSA* (Ours)	<b>93.3</b>	88.1	47.0	<b>84.2</b>	<b>60.2</b>	<b>75.0</b>	<b>87.7</b>	<b>81.7</b>	<b>92.0</b>	<b>34.5</b>	<b>87.8</b>
Method	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mIoU
AdvCAM [34] CVPR21	38.5	84.0	83.0	79.5	71.9	47.5	80.8	59.1	65.4	49.7	68.0
MCT [65] CVPR22	58.3	81.7	81.1	77.0	76.4	49.2	80.0	55.1	65.4	54.5	68.4
ToCo [53] CVPR23	<b>68.4</b>	81.4	85.6	83.2	<b>83.4</b>	<b>68.2</b>	<b>88.9</b>	55.0	49.3	<b>65.0</b>	72.2
CoSA* (Ours)	59.6	<b>86.2</b>	<b>86.3</b>	<b>84.9</b>	82.8	<b>68.2</b>	87.4	<b>63.9</b>	<b>67.7</b>	61.6	<b>75.2</b>

**Table 10: Per-class Segmentation on VOC test Split.** Comparison of per-class segmentation results on VOC test. Results from AdvCAM, MCT, and ToCo are used for this comparison. Best results are in **bold**.

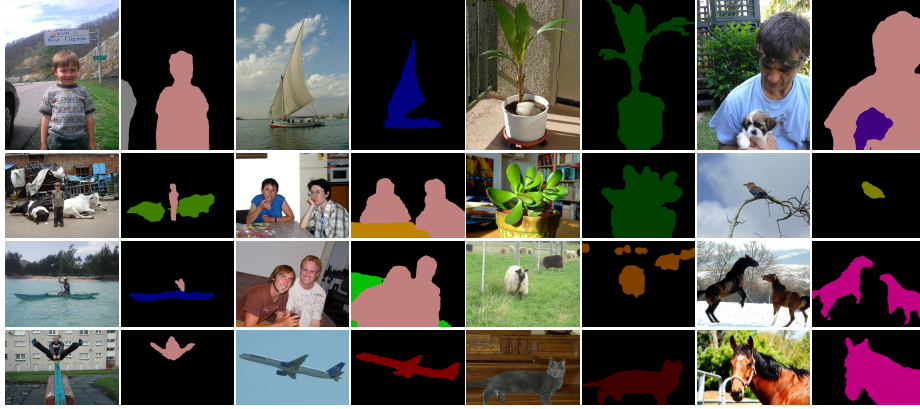


Class	MCT [65] (CVPR22)	Xu <i>et al.</i> [66] (CVPR23)	ToCo [53] (CVPR23)	CoSA* (Ours)	Class	MCT [65] (CVPR22)	Xu <i>et al.</i> [66] (CVPR23)	ToCo [53] (CVPR23)	CoSA* (Ours)
background	82.4	<b>85.3</b>	68.5	84.0	wine glass	27.0	33.8	20.6	<b>42.1</b>
person	62.6	<b>72.9</b>	28.1	70.3	cup	29.0	<b>35.8</b>	26.0	33.1
bicycle	47.4	49.8	39.7	<b>52.4</b>	fork	23.4	20.0	7.6	<b>24.2</b>
car	47.2	43.8	38.9	<b>54.3</b>	knife	12.0	12.6	18.4	<b>32.9</b>
motorcycle	63.7	66.2	55.1	<b>71.9</b>	spoon	6.6	6.7	3.0	<b>9.0</b>
airplane	64.7	69.2	62.1	<b>74.0</b>	bowl	22.4	<b>23.7</b>	19.8	22.8
bus	64.5	69.1	39.0	<b>77.2</b>	banana	63.2	64.4	<b>71.5</b>	69.3
train	<b>64.5</b>	63.7	48.7	60.0	apple	44.4	50.8	55.5	<b>61.3</b>
truck	44.8	43.4	37.3	<b>55.4</b>	sandwich	39.7	47.0	41.2	<b>48.3</b>
boat	42.3	42.3	49.1	<b>52.1</b>	orange	63.0	64.6	<b>70.6</b>	69.2
traffic light	49.9	49.3	47.3	<b>55.1</b>	broccoli	51.2	50.6	<b>56.7</b>	52.8
fire hydrant	73.2	74.9	69.6	<b>78.8</b>	carrot	40.0	38.6	46.4	<b>59.4</b>
stop sign	76.6	77.3	70.1	<b>82.2</b>	hot dog	53.0	54.0	<b>60.1</b>	59.9
park meter	64.4	67.0	67.9	<b>71.5</b>	pizza	62.2	<b>64.1</b>	54.9	56.5
bench	32.8	34.1	43.9	<b>50.2</b>	donut	55.7	59.7	61.1	<b>71.1</b>
bird	62.6	63.1	58.6	<b>65.4</b>	cake	47.9	50.6	42.5	<b>57.0</b>
cat	78.2	76.2	74.0	<b>79.8</b>	chair	22.8	24.5	24.1	<b>33.8</b>
dog	68.2	70.6	64.0	<b>72.8</b>	couch	35.0	40.0	44.2	<b>58.1</b>
horse	65.8	67.1	66.1	<b>71.4</b>	plant	13.5	13.0	<b>27.4</b>	23.5
sheep	70.1	70.8	67.9	<b>74.3</b>	bed	48.6	53.7	54.0	<b>61.5</b>
cow	68.3	71.2	69.0	<b>74.0</b>	table	12.9	19.2	25.6	<b>29.2</b>
elephant	81.6	<b>82.2</b>	79.7	81.9	toilet	63.1	66.6	62.0	<b>69.7</b>
bear	80.1	79.6	76.8	<b>85.3</b>	tv	47.9	50.8	49.1	<b>53.2</b>
zebra	<b>83.0</b>	82.8	77.5	76.3	laptop	49.5	55.4	55.7	<b>63.9</b>
giraffe	<b>76.9</b>	76.7	66.1	68.5	mouse	13.4	14.4	8.6	<b>16.4</b>
backpack	14.6	17.5	20.3	<b>28.6</b>	remote	41.9	47.1	<b>56.6</b>	49.1
umbrella	61.7	66.9	70.9	<b>73.4</b>	keyboard	49.8	<b>57.2</b>	41.8	49.6
handbag	4.5	5.8	8.1	<b>11.9</b>	cellphone	54.1	54.9	58.5	<b>66.2</b>
tie	25.2	31.4	33.4	<b>47.7</b>	microwave	38.0	46.1	<b>55.5</b>	53.2
suitcase	46.8	51.4	55.3	<b>63.8</b>	oven	29.9	35.3	36.2	<b>49.2</b>
frisbee	43.8	54.1	39.6	<b>63.1</b>	toaster	0.0	<b>2.0</b>	0.0	0.0
skis	12.8	13.0	4.0	<b>22.5</b>	sink	28.0	36.1	19.0	<b>41.9</b>
snowboard	31.4	30.3	15.5	<b>40.5</b>	refrigerator	40.1	52.7	51.9	<b>62.0</b>
sports ball	9.2	<b>36.1</b>	11.0	33.1	book	32.2	34.8	31.5	<b>37.8</b>
kite	26.3	47.5	40.7	<b>59.9</b>	clock	43.2	51.5	32.9	<b>55.2</b>
baseball bat	0.9	<b>7.0</b>	1.8	3.8	vase	22.6	25.8	33.3	<b>33.8</b>
glove	0.7	10.4	17.6	<b>37.9</b>	scissors	32.9	30.7	49.8	<b>54.7</b>
skateboard	7.8	<b>15.2</b>	13.3	12.5	teddy bear	61.9	61.4	67.5	<b>69.3</b>
surfboard	46.5	<b>51.5</b>	21.5	16.5	hair drier	0.0	1.3	<b>10.0</b>	0.3
racket	1.4	<b>26.4</b>	6.8	7.2	toothbrush	12.2	19.0	29.3	<b>39.3</b>
bottle	31.1	<b>37.1</b>	25.7	35.1	<b>mIoU</b>	42.0	45.9	42.4	<b>51.1</b>

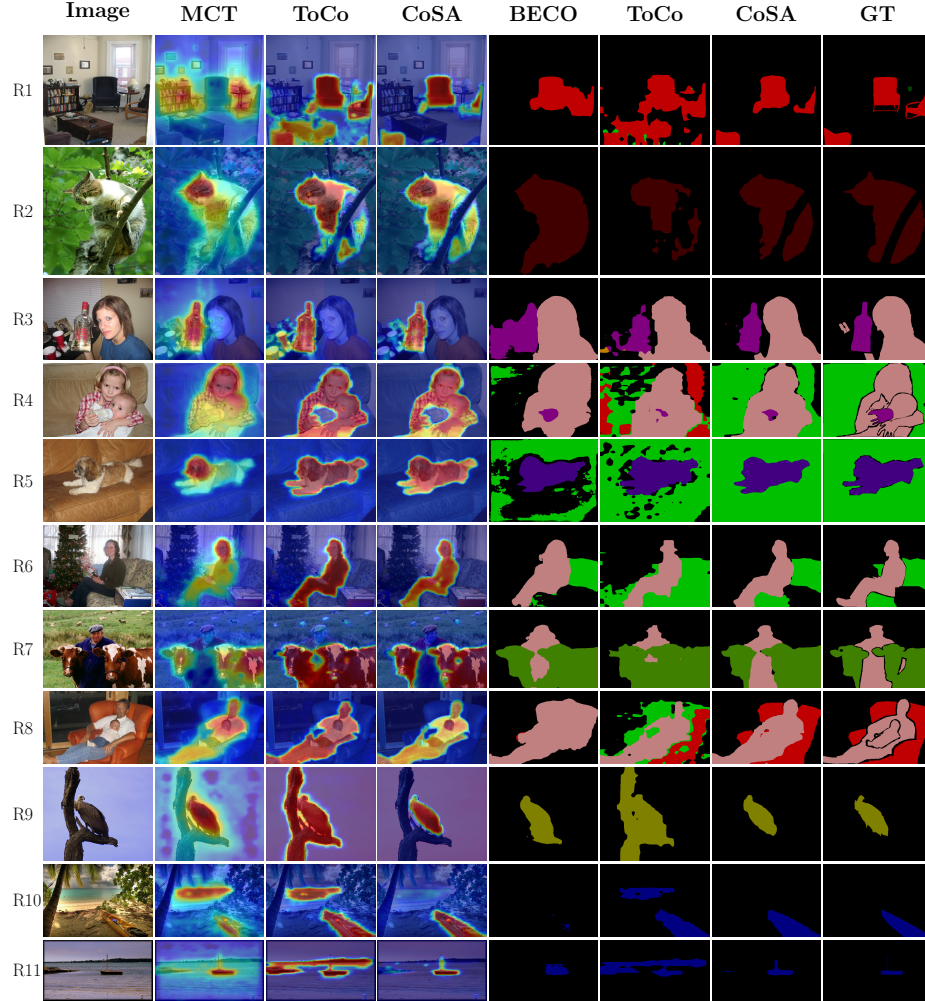
**Table 11: Per-class Segmentation Results on COCO.** Comparison of per-class segmentation results on COCO val. CoSA is compared with MCT, Xu *et al.* and ToCo. Best results are in **bold**.

#### C.4 Further Qualitative Comparisons

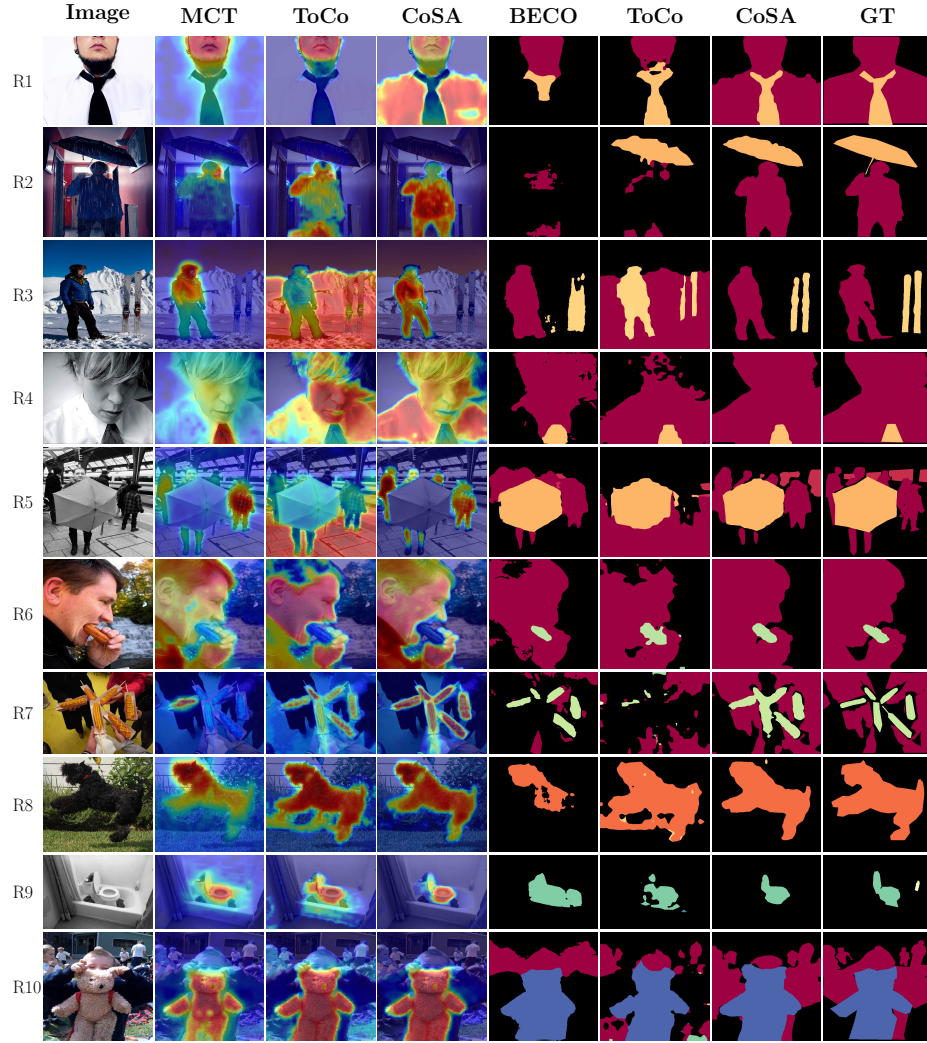
More visualizations of our CoSA results are given in Fig. 14 for VOC and Fig. 15, Fig. 16 for COCO. When compared to other SOTA models, CoSA exhibits i) better foreground-background separation (evidenced in  $R2-R3$  in Fig. 14 and  $R1-R10$  in Fig. 15); ii) more robust to inter-class variation and occlusion (affirmed in  $R4-R7$  in Fig. 14 and  $R1-R4$  in Fig. 16). iii) less coexistence problem (demonstrated in  $R9-R11$  in Fig. 14 and  $R8-R10$  in Fig. 16); Last but not least, our CoSA can reveal certain limitations in manual GT segmentation, as depicted in  $R8$  in Fig. 14 and  $R5-R7$  in Fig. 16. We also show our CoSA results on VOC **test** set in Fig. 13 and some failure cases in Fig. 17.



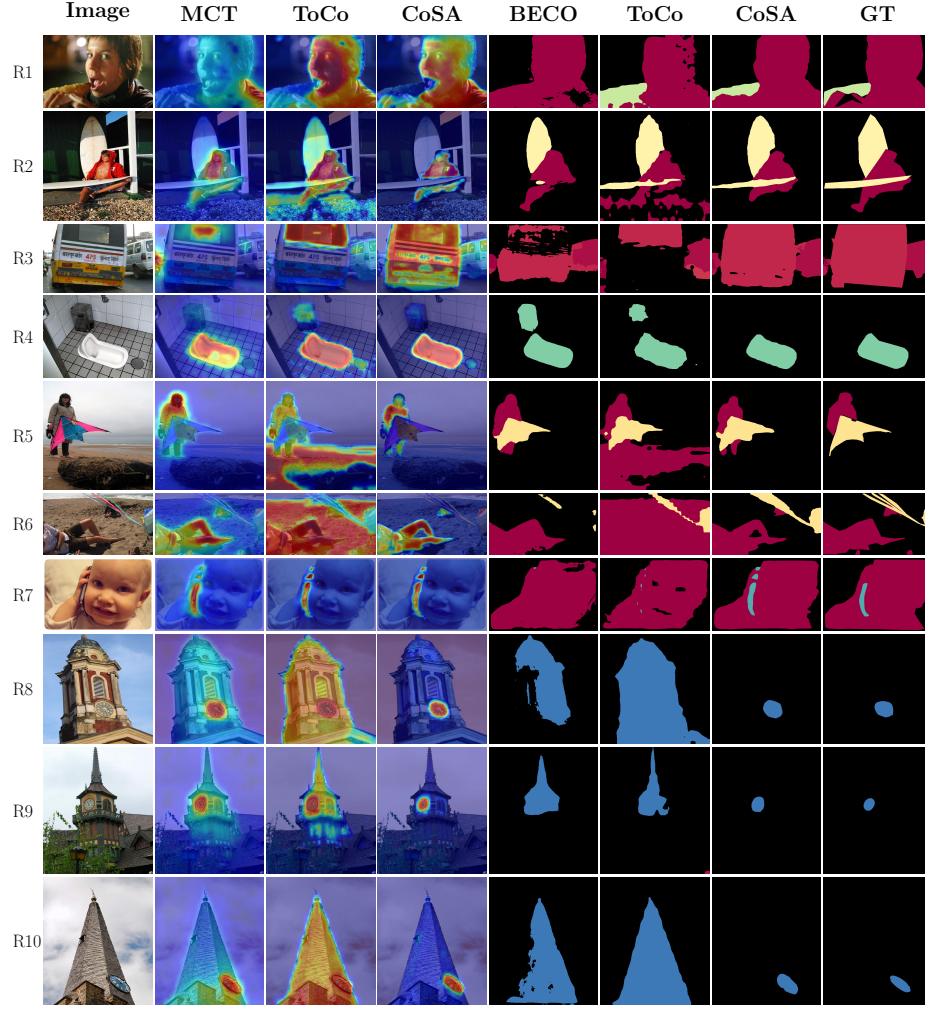
**Fig. 13: Visualization on VOC test.** Different colors represent different categories: black: background; grey: car; pink: person; blue: boat; green: plant; purple: dog; light green: cow. yellow: dining-table. olive: bird; light blue: sofa; brown: sheep; magenta: house; red: airplane; dark red: cat.



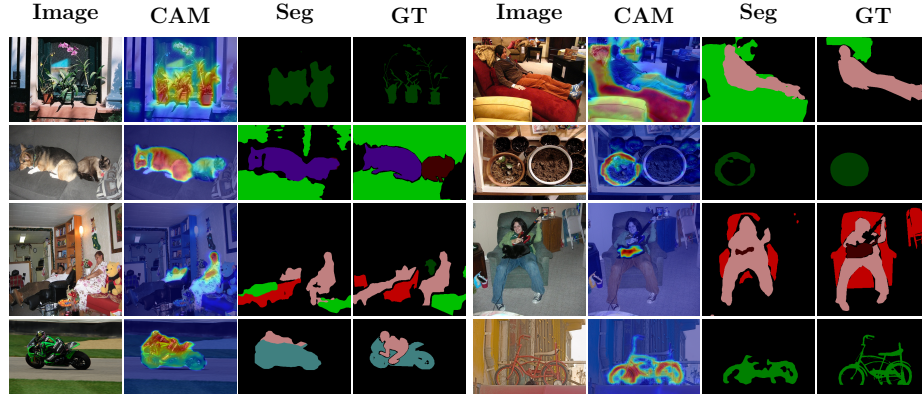
**Fig. 14: Qualitative Comparisons on VOC Dataset.** CoSA exhibits 1) better foreground-background separation (*R1–R3*); 2) more robust to inter-class variation and occlusion (*R4–R7*); 3) limitations in the ground truth annotations (*R8*); 4) less coexistence problem (*R9–R11*). Different colors represent different categories: black: background; white: ignore areas; red: chair; green: plant; brown: cat; pink: person; purple: bottle; blue: sofa; orange: dog; yellow: cow; light green: bird; dark blue: boat; The activated classes in the demonstration from top to bottom are: chair, cat, bottle, person, dog, person, cow, person, bird, boat, boat.



**Fig. 15: Qualitative Comparisons on COCO Dataset.** CoSA demonstrates superior quality in terms of foreground-background separation (*R1–R10*). Categories involved – *R1*: **person**, tie; *R2*: **person**, umbrella; *R3*: **person**, skis; *R4*: **person**, tie; *R5*: **person**, train, umbrella; *R6*: **person**, hot dog; *R7*: **person**, **hot dog**; *R8*: **dog**, frisbee; *R9*: bottle, **toilet**; *R10*: **person**, **teddy bear**; Categories in **Bold** denotes the activated classes in CAMs.



**Fig. 16: More Qualitative Comparisons on COCO Dataset.** CoSA shows 1) more robust to inter-class variation and occlusion ( $R1-R4$ ); 2) limitations in the ground truth annotations ( $R5-R7$ ); 3) less coexistence problem ( $R8-R10$ ). Categories involved –  $R1$ : **person**, donuts;  $R2$ : **person**, surfboard.  $R3$ : person, car, motorcycle, **bus**;  $R4$ : **toilet**;  $R5$ : **person**, kite;  $R6$ : **person**, kite;  $R7$ : person, **cell phone**;  $R8$ : **clock**;  $R9$ : **clock**;  $R10$ : **clock**. Categories in **Bold** denotes the activated classes in CAMs.



**Fig. 17: Illustrations of CoSA failure Cases.** Different colors represent different categories: black: background; white: ignore areas; ●: plant; ●: person; ●: sofa; ●: dog; ●: cat; ●: chair; ●: motorbike; ●: bicycle. The activated classes in the demonstration from left to right and from top to bottom are: plant, sofa, dog, plant, person, cat, person, bicycle.

Transformation	Description	Parameter Setting
RandomRescale	Rescale the image by $r$ times, $r$ randomly sampled from $r \sim U(r_{min}, r_{max})$ .	$r_{min} = 0.5, r_{max} = 2$
RandomFlip	Randomly horizontally flip a image with probability of $p$ .	$p = 0.5$
RandomCrop	Randomly crop a image by a hight $h$ and a width $w$ .	$w = 448, h = 448$
GaussianBlur	Randomly blur a image with probability of $p$ .	$p = 0.5$

**Table 12:** Weak data augmentation  $\mathcal{T}_w$  for the input of assignment network.

Transformation	Description	Parameter Setting
RandomRescale	Rescale the image by $r$ times, $r$ randomly sampled from $r \sim U(r_{min}, r_{max})$ .	$r_{min} = 0.5, r_{max} = 2$
RandomFlip	Randomly horizontally flip a image with probability of $p$ .	$p = 0.5$
RandomCrop	Randomly crop a image by a hight $h$ and a width $w$ .	$w = 448, h = 448$
GaussianBlur	Randomly blur a image with probability of $p$ .	$p = 0.5$
OneOf	Select one of the transformation in a transformation set $T$ .	$T = \text{TransAppearance}$

**Table 13:** Strong data augmentation  $\mathcal{T}_s$  for the input of online network image.

Transformation	Description	Parameter Setting
Identity	Returns the original image.	
Autocontrast	Maximizes the image contrast by setting the darkest (lightest) pixel to black (white).	
Equalize	Equalizes the image histogram.	
RandSolarize	Invert all pixels above a threshold value $T$ .	$T \in U(0, 1)$
RandColor	Adjust the color balance. $C = 0$ returns a black&white image, $C = 1$ returns the original image.	$C \in U(0.05, 0.95)$
RandContrast	Adjust the contrast. $C = 0$ returns a solid grey image, $C = 1$ returns the original image.	$C \in U(0.05, 0.95)$
RandBrightness	Adjust the brightness. $C = 0$ returns a black image, $C = 1$ returns the original image.	$C \in U(0.05, 0.95)$
RandSharpness	Adjust the sharpness. $C = 0$ returns a blurred image, $C = 1$ returns the original image.	$C \in U(0.05, 0.95)$
RandPolarize	Reduce each pixel to $C$ bits.	$C \in U(4, 8)$

**Table 14:** Appearance transformations, called **TransAppearance**, used in strong data augmentation.