When Confidence Fails: Revisiting Pseudo-Label Selection in Semi-supervised Semantic Segmentation

Supplementary Material

A. Overview

In the supplementary material for CSL, we provide a proof of Eq. (8) (Sec.B), give a low-complexity implementation of CSL and pseudo-code (Sec.C), extend the implementation details and experimental comparison (Sec.D), supplements the analysis of residual dispersion(Sec.E), discuss the potential limitations (Sec.F), and gives more visual results (Sec.G).

B. Proof of Equation 8

For the pseudo-label selection problem in semi-supervised semantic segmentation, it can be formulated as:

$$\max_{S} Tr\left(S^T \Phi^T \Phi S\right), s.t. S \in \{0,1\}^{HW \times 2},$$

where S is the binary selection matrix subject to $\sum_c S_{n,c} = 1$ with $c \in \{1,2\}$ indicating class indices, and $Tr(\cdot)$ denotes the matrix trace. Each column of $\Phi = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{HW}]$ represents the feature $\mathbf{h}_n \in \mathbb{R}^2$ of pixel n.

Proof. Without loss of generality, the essential goal of pseudo-label selection is to assign pixel classes via S such that the selected pseudo-labels are optimal under a certain metric:

$$\mathcal{L}(S) = \sum_{n=1}^{HW} \varphi\left(\mathbf{z}_n, S_{n,:}\right),$$

where $\varphi(\cdot,\cdot)$ is a function that measures the potential risk of assigning pixel n to a specific class , with $\mathbf{z}_n$ representing the pixel's feature. Considering the potential natural separation exhibited in Fig. 1, we employ intra-class consistency as the metric, which measures the distance between the pixel feature vector $\mathbf{z}_n = \mathbf{h}_n$ and the mean feature vector μ_c of its assigned class c, which can be expressed as

$$\varphi\left(\mathbf{z}_n, S_{n,:}\right) = \sum_{c=1}^{2} S_{n,c} \|\mathbf{h}_n(c) - \mu_c\|^2,$$

thus $\mathcal{L}(S)$ can be reformulated as

$$\mathcal{L}(S) = \|\Phi - S(S^T S)^{-1} S^T \Phi\|_F^2,$$

let $\mathbf{P} = S(S^T S)^{-1} S^T$, the $\mathcal{L}(S)$ simplifies to

$$\mathcal{L}(S) = \|\Phi - \mathbf{P}\Phi\|_F^2.$$

Using the Frobenius norm identity

$$\mathcal{L}(S) = \|\Phi\|_F^2 - 2\,\mathrm{Tr}(\Phi^T \mathbf{P} \Phi) + \|\mathbf{P}\Phi\|_F^2,$$

since $\mathbf{P}^2 = \mathbf{P}$ (idempotence of projection matrices), thus

$$\mathcal{L}(S) = \|\Phi\|_F^2 - \mathrm{Tr}(\Phi^T \mathbf{P} \Phi).$$

Minimizing $\mathcal{L}(S)$ is equivalent to maximizing $\mathrm{Tr}(\Phi^T \mathbf{P} \Phi)$. Substituting $\mathbf{P}$:

$$\mathrm{Tr}(\Phi^T \mathbf{P} \Phi) = \mathrm{Tr}(\Phi^T S (S^T S)^{-1} S^T \Phi),$$

for binary classification, $S^T S = \mathrm{diag}(n_1, n_2)$, where n_1 and n_2 are the number of pixels in each class. Thus:

$$\mathrm{Tr}(\Phi^T \mathbf{P} \Phi) = \mathrm{Tr}(S^T \Phi \Phi^T S) \cdot (n_1^{-1} + n_2^{-1}),$$

ignoring the normalization constant, this reduces to:

$$\max_{S} \mathrm{Tr}(S^T \Phi^T \Phi S).$$

C. Algorithm

C.1. Low-complexity Implementation of Prediction Convex Optimization Separation

In Eq. (9), we used the eigenvectors u_i of $\Phi^T\Phi$. however, obtaining u_i through $\Phi^T\Phi$ is computationally expensive, especially when $\Phi \in \mathbb{R}^{2\times HW}$ has a large number of columns, where $HW \gg 2$. so we provide an efficient alternative with Singular Value Decomposition (SVD).

The SVD of Φ is given by:

$$\Phi = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

where $\mathbf{U} \in \mathbb{R}^{2\times 2}$ and $\mathbf{V} \in \mathbb{R}^{HW\times HW}$ are orthogonal matrix, $\mathbf{\Sigma} \in \mathbb{R}^{2\times HW}$ is a diagonal matrix containing the singular values σ_i.

Substituting the SVD of Φ,

$$\Phi^T\Phi = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T),$$

which simplifies to

$$\Phi^T\Phi = \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

Using the orthogonality of $\mathbf{U}$ ($\mathbf{U}^T\mathbf{U} = \mathbf{I}$),

$$\Phi^T\Phi = \mathbf{V}\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T.$$

Let $\mathbf{\Lambda} = \mathbf{\Sigma}^T\mathbf{\Sigma} = \mathrm{diag}(\sigma_1^2, \sigma_2^2)$, where σ_i^2 are the squared singular values. Then

$$\Phi^T\Phi = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T.$$

This shows that $\mathbf{V}$ contains the eigenvectors of $\Phi^T\Phi$, $\mathbf{\Lambda}$ contains the eigenvalues of $\Phi^T\Phi$.

Computing the eigenvectors of $\Phi^T\Phi$ via direct eigen-decomposition requires explicitly forming $\Phi^T\Phi$, which has a cost of $\mathcal{O}(HW^2)$ for matrix multiplication and $\mathcal{O}(HW^3)$ for eigen-decomposition. In contrast, computing the SVD of Φ has a cost of $\mathcal{O}(HW^2)$.

C.2. Pseudocode for Optimization Separation

Algorithm 1 provides the pseudocode for Prediction Convex Optimization Separation. Using a convex optimization strategy based on confidence distribution, CSL effectively excludes a large number of high-confidence false predictions in pseudo-label selection to improve the performance in semi-supervised semantic segmentation tasks.

Algorithm 1 Pseudocode of Prediction Convex Optimization Separation in a PyTorch-like style.

```
# pmax: Pixel-level maximum confidence
# vn: Pixel-level residual dispersion

def PCOS(pmax, vn):
  # combine pmax and vn into a feature matrix Φ
  Φ = stack([pmax, vn], axis=1).T
  # extract the top two eigenvectors from Φ
  U, Sigma, VT = svd(Φ)
  eig_vectors = VT[:, :2]
  # constructing the optimal selection matrix
  S = argmax(abs(eig_vectors), axis=1)
  # calculate stats for each class
  stats = [(Φ[S == c].mean(dim=1),
     Φ[S == c].std(dim=1)) for c in range(2)]
  # select the reliable class
  mu, sigma = max(stats, key=lambda x:x[0])
  # smooth loss weight
  weight = exp(-((Φ-mu)/(8*sigma))**2)
  weight = weight.prod(dim=0)
  # preserving reliable prediction weights
  weight[(Φ[0, :]>mu[0])‖(Φ[1, :]>mu[1])] = 1
  return weight
```

C.3. Pseudocode for Trusted Mask Perturbation

In Sec. 3.4 of our paper, we propose the Trusted Mask Perturbation Strategy. The core idea of this method is to enhance the mutual information between low-confidence regions and pseudo-labels to strengthen the fitting of these regions. Specifically, we use high-confidence predictions from weakly augmented outputs as pseudo-labels but randomly discard their image content, while the image content of low-confidence predictions is entirely preserved. This forces the network to infer the classes of high-confidence regions based on the image content of low-confidence regions, thereby compensating for the underfitting in low-confidence areas. To clarify things, we present the pseudocode of the threshold updating strategy in a PyTorch-like style.

Algorithm 2 Pseudocode of Trusted Mask Perturbation in a PyTorch-like style.

```
# x_w: Image with weak augmentation perturbation
# image_size: The length or width of the image
# block_size: The masking patch size
# masking_rate: The masking pixel ratio
# f: segmentation network

pred_w = f(x_w)
mask_w = pred_w.argmax(dim=1).detach()
# compute weights using PCOS on the projection
weight = PCOS(Projection(pred_w))
# create a reliability mask
reli_mask = (weight == 1)
# gain patch-based perturbation mask (Eq. (12))
mask_size = img_size // block_size
cover_mask = (rand(mask_size, mask_size) <
           masking_rate).float()
cover_mask = interpolate(cover_mask,
           size=img_size, mode='nearest')
# perturbation only for reliable predictions
cover_mask = cover_mask & reli_mask
# constructing perturbed images
x_m = x_s.clone()
x_m[cover_mask == 1] = 0
pred_m = f(x_m)
# calculated loss
criterion = CrossEntropyLoss()
loss_m = criterion(pred_m, mask_w)
```

D. Extensive Experiment Details and Results

D.1. More Implementation Details

Following prior works [34, 69] , we employ random scaling between [0.5, 2.0], cropping, and flipping as weak augmentations. We combine ColorJitter, random grayscale, Gaussian blur, and CutMix [71] for strong augmentations. Weak augmentations and a modified strong augmentation (without random grayscale and Gaussian blur) are applied before Trusted Mask Perturbation. Additionally, we incorporate 50% random channel dropout as feature perturbations to encourage robust feature representations as in previous works [50, 51, 69] . For computational efficiency, mixed-precision training based on BrainFloat16 is utilized.

D.2. More Experimental Results

Different loss weights. In Tab. 7, we investigate the impact of different loss weightings on segmentation accuracy. The imbalance between consistency loss and masking loss significantly affects model performance. Moreover, assigning excessively high or low loss weights to the unlabeled data also leads to a degradation in performance. The results indicate that the optimal performance is achieved when $[\lambda_1, \lambda_2]$ are set to [0.5, 0.5].

A:We conducted experiments comparing threshold-based methods and CSL on identical model predictions. As shown in Tab. 9, CSL achieves consistent accuracy improvements (+4.1%), while maintaining competitive recall (+1.0%).

λ_1	λ_2	1/16(92)	1/8(183)	1/2(732)
0.50	0.50	**76.8**	**79.6**	**80.9**
0.75	0.25	75.6	78.1	79.7
0.25	0.75	75.2	77.6	79.2
0.30	0.30	76.3	79.1	80.2
0.70	0,70	75.7	78.4	79.8
1.00	1.00	72.5	76.6	77.6

Table 7. Impact of different loss weights, evaluated on the original PASCAL VOC 2012 with a crop size of 321.

Method	1/16	1/8	1/4	1/2	Full
CSL	**76.8**	**79.6**	**80.3**	**80.9**	**82.3**
Class-specific CSL	74.6	78.3	79.1	79.7	80.6

Table 8. Comparison of CSL and class-specific strategy, evaluated on the original PASCAL VOC 2012 with a crop size of 321.

Metrics		plane	bicyc	bus	car	chair	perso	sofa	mean
Sampling	Base	**96.2**	**93.1**	**98.0**	**93.2**	**82.0**	**91.0**	**78.6**	**91.0**
Accuracy	Base	90.8	75.4	90.1	85.4	33.4	83.7	50.2	82.4
Recall	Base	96.6	97.6	93.2	89.5	**81.8**	87.5	86.7	93.6
Sampling	CSL	92.3	85.7	94.1	89.5	63.1	85.6	71.2	87.6
Accuracy	CSL	**95.1**	**82.4**	**96.4**	**91.7**	**42.5**	**89.1**	**60.4**	**86.5**
Recall	CSL	**97.1**	**98.2**	**95.8**	**92.3**	80.1	**87.7**	**94.5**	**94.6**

Table 9. Pseudo-Label sampling rate, accuracy and recall comparison under PASCAL original 1/4 Splits with class-wise metrics.

Class-specific prediction selection. In semi-supervised semantic segmentation, due to the long-tailed distribution of datasets, the confidence distributions of predictions vary significantly across classes. This suggests that employing a class-specific convex optimization strategy could potentially yield performance gains. To analyze this, we reconstruct the feature matrix Φ into class-specific feature spaces for Prediction Convex Optimization Separation:

$$\Phi_k = \{h_n \mid h_n \in \Phi, k_n^* = k\} \tag{16}$$

where Φ_c is Class-specific feature matrix for class k and k_n^* is the predicted class label for pixel n.

We conducted ablation experiments presented in Tab. 8, where it can be observed that using class-specific schemes results in significant performance degradation. This may be attributed to the fact that most classes have too few pixel samples within instances to maintain an effective convex optimization strategy.

Augmentations of Trusted Mask Perturbations. In Tab. 11, we evaluate the impact of various augmentations on Trusted Mask Perturbations. Adding CutMix leads to significant performance degradation, as it introduces misleading contextual information by directly stitching image patches while the trusted masking mechanism forces the model to learn contextual relationships, resulting in negative effects. Similarly, random grayscale enhances the sample by reducing color diversity, which conflicts with the masking mechanism and results in substantial information loss. Therefore, we adopt enhancements that exclude CutMix and random grayscale as additional Augmentations to the masking strategy.

dataset	92	183	366	732	1464
D_l	73.2	77.1	78.8	79.4	80.3
D_u	**76.8**	**79.6**	**80.3**	**80.9**	**82.3**
$D_l \cup D_u$	75.6	78.4	79.6	80.1	81.7

Table 10. Ablation study of masking datasets under 1/2 splits. For the labeled dataset, predictions are treated as reliable predictions.

Method	1/8	1/2
A^ω only	78.3	79.1
$A^\omega \& A^s$	78.7	79.5
$A^\omega \& A^s$ W/O Cutmix	79.1	79.8
$A^\omega \& A^s$ W/O grayscale	79.0	80.4
$A^\omega \& A^s$ W/O (Cutmix & grayscale)	**79.6**	**80.9**

Table 11. Comparison of augmentations strategies for TMP, evaluated on the original PASCAL VOC 2012 with a crop size of 321.

Method	Encoder	1/16	1/8	1/4	1/2	full
UniMatchV2	DINOv2	79.0	85.5	85.9	86.7	87.8
Ours	DINOv2	**80.2**	**85.8**	**86.3**	**87.4**	**88.1**
AllSpark	SegFormer	76.1	78.4	79.8	80.8	82.1
Ours	SegFormer	**77.4**	**80.2**	**81.5**	**83.5**	**85.3**

Table 12. Influence of different network architectures, evaluated on the original PASCAL VOC 2012 with a crop size of 513.

Selection of Masking Datasets. To evaluate the impact of applying the mask perturbation to different subsets: labeled images, unlabelled images, and the combination of both, experiments were performed under different splits, as detailed in Tab. 10. Results show applying masking to labeled data or the combination leads to progressively severe performance degradation as the number of labeled data samples decreases. This phenomenon can be attributed to the network learning contextual relationships present only in the labeled data instances and applying them to unlabeled data.

Different network Architectures. Considering that different encoders may exhibit varying degrees of overconfidence in their representations and utilize contextual relationships through distinct mechanisms, we supplement additional experiments with diverse network architectures in Tab. 12. Specifically, DINOv2-S adopts the hyperparameters from Unimatchv2 [70], while SegFormer-B5[54] shares

v_n	$H(p_n)$	H_{res}	m_n	1/8	1/4
✓				**79.6**	**80.3**
✓	✓			77.3	78.5
✓		✓		77.5	79.1
✓			✓	76.9	78.2
✓	✓	✓	✓	71.5	76.3

Table 13. Comparison of different combinations of metrics, evaluated on the original PASCAL VOC 2012 with a crop size of 321.

the training configuration described in Sec. 4.1. Experimental results reveal that our Confidence-aware Structure Learning (CSL) achieves consistent performance improvements across architectures, demonstrating its effectiveness as an architecture-independent framework.

E. More Analysis for residual dispersion

E.1. Why residual dispersion but not other common metrics

The choice of residual dispersion as a reliability metric stems from its theoretical foundation in entropy minimization principles. As derived in Eq. (4)-Eq. (7), the cross-entropy objective naturally decomposes into two complementary terms: the maximum confidence $p_n(k')$ and residual dispersion v_n. This decomposition reveals an intrinsic geometric relationship, that reliable predictions must simultaneously maximize confidence in the dominant class and dispersion among residual probabilities.

Traditional metrics like entropy $H(p_n)$ and prediction margin m_n fail to meet this criterion. Though entropy theoretically encourages unimodal distributions, it inadvertently tolerates pathological multi-peaked configurations. Consider predictions $p_A = [0.5, 0.5, 0, \ldots, 0]$ and $p_B = [0.5, 0.01, \ldots, 0.01]$. Paradoxically, p_A exhibits lower entropy despite being less reliable. This demonstrates entropy inability to distinguish valid unimodal predictions from problematic multi-modal ones.

Residual entropy $H_{\text{res}} = -\sum_{k \neq k'} p_n(k) \log p_n(k)$ avoid such problem and is ostensibly similar to the second term in Eq. (4), but its additional linear dependence on $pn_(k)$ significantly reduces its ability to judge prediction credibility under overconfidence.

Margin $m_n = p_n(k') - \max_{k \neq k'} p_n(k)$ focuses only on the top two categories. In the case of overconfidence, the margin is completely dominated by the maximum confidence and shows no discrimination.

E.2. Why not use multiple metrics

While combining multiple metrics theoretically enhances pseudo-label selection with negligible additional time overhead, considering that metrics like entropy or margin are repeated measurements of Eq. (4), adding such redundant features will introduce covariance conflicts that will degrade the model performance. As shown in Tab. 13, optimal performance is achieved when only maximum confidence and residual dispersion are considered.

F. Potential Limitations

In CSL, we employ a convex optimization strategy within the confidence distribution feature space to exclude potential high-confidence erroneous pseudo-labels caused by model overconfidence. However, we observe that when the proportion of labeled data is extremely small, the significant disparity in the marginal distributions between sample sets leads to unavoidable cognitive bias. This limitation is particularly pronounced in real-world scenarios, where such imbalanced data splits are common. Thus, an important avenue for future research could involve leveraging limited labeled data more effectively to calibrate cognitive biases and improve the quality of pseudo-labels.

Additionally, although the theoretical validity of residual dispersion is established under the general principles of semi-supervised semantic segmentation, given the broad applicability of entropy minimization, this proof can be readily extended to similar domains such as semi-supervised classification or unsupervised domain adaptation. Similarly, introducing direct confidence calibration methods commonly used in other fields into semi-supervised semantic segmentation represents another promising technical pathway. We leave these potential extensions for future exploration.

Last but not least, CSL utilizes the masking of reliable regions to leverage contextual relationships, thereby enhancing the model's learning in low-confidence areas. However, we find that as the interfering information surrounding low-confidence regions is masked, predictions in these areas tend to be more accurate compared to unmasked regions. Yet, due to the potential for errors, the information from these areas is discarded during training. Therefore, a potential approach could be to further screen this valid information. This could potentially complement the direct supervision signals for low-confidence regions, fostering a more effective learning process.

G. More Visualizations

In Fig. 8, we show that the method in this paper and other methods More segmentation results on the PASCAL VOC 2012 dataset.

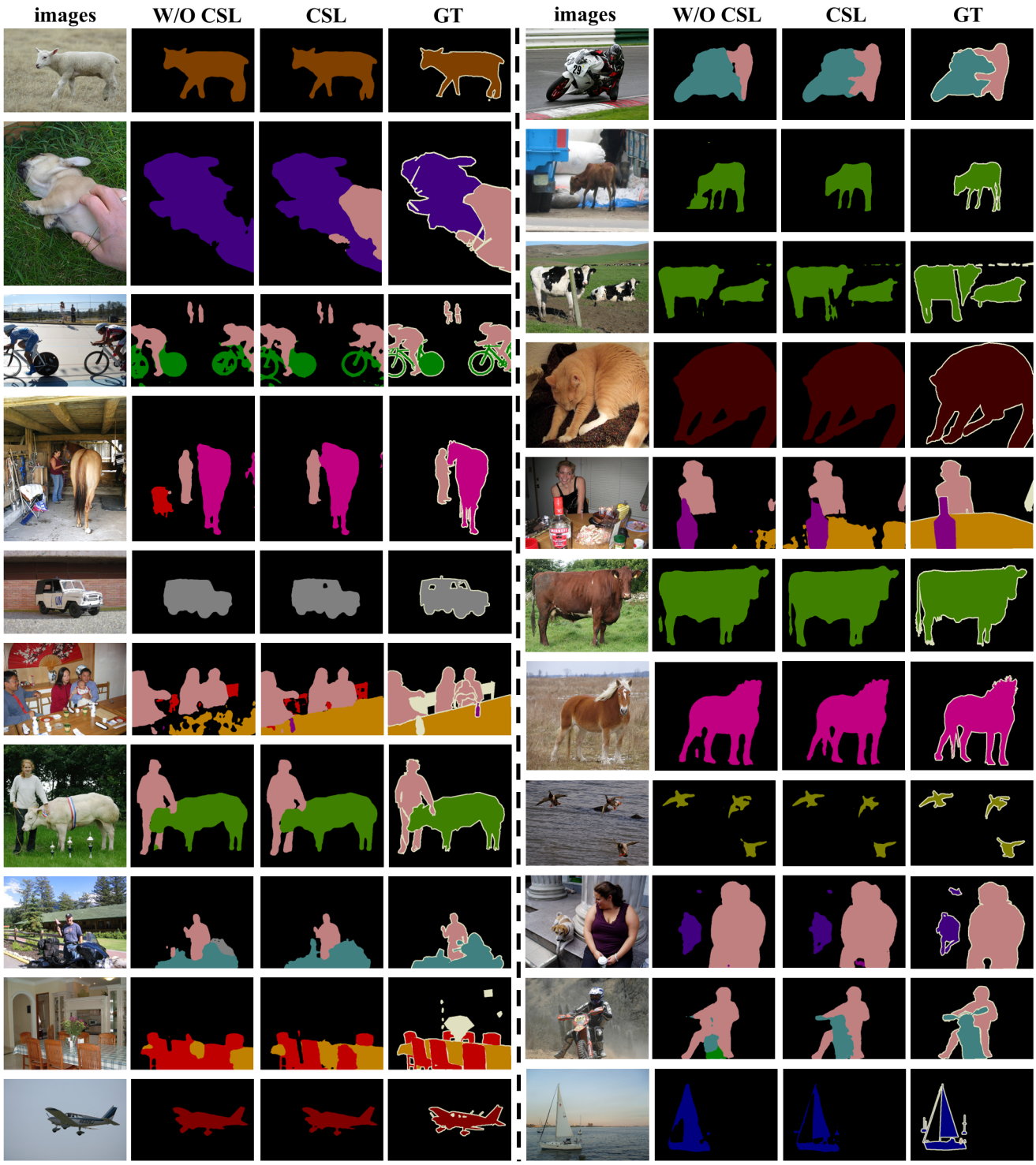

Figure 8. More visualization of the segmentation results on the PASCAL VOC 2012 dataset on 1/8 splits with a crop size of 513.