Direct Segmentation without Logits Optimization for Training-Free Open-Vocabulary Semantic Segmentation

Supplementary Material

6. Proof

Given the cost matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$ and the regularization scalar ϵ, the objective is to solve the following equation:

$$\boldsymbol{\pi}^* = \min_{\boldsymbol{\pi}} \sum_{i,j} \boldsymbol{C}_{i,j}\boldsymbol{\pi}_{i,j} - \epsilon \sum_{i,j} \boldsymbol{\pi}_{i,j}(\ln \boldsymbol{\pi}_{i,j} - 1), \quad (13)$$

subject to marginal constraints:

$$\sum_j \boldsymbol{\pi}_{i,j} = \boldsymbol{f}_i^c, \sum_i \boldsymbol{\pi}_{i,j} = \boldsymbol{f}_j^t, \quad \forall i, j, \quad \boldsymbol{\pi}_{i,j} \geq 0, \quad (14)$$

where $\sum_i \boldsymbol{f}_i^c = 1, \sum_j \boldsymbol{f}_j^t = 1$. By introducing Lagrange multipliers $\alpha \in \mathbb{R}^N$ and $\beta \in \mathbb{R}^N$, the Lagrangian is defined as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\pi}, \alpha, \beta) = &\min_{\boldsymbol{\pi}} \sum_{i,j} \boldsymbol{C}_{i,j}\boldsymbol{\pi}_{i,j} - \epsilon \sum_{i,j} \boldsymbol{\pi}_{i,j}(\ln \boldsymbol{\pi}_{i,j} - 1) \\ &+ \sum_i \alpha_i(\boldsymbol{f}_i^c - \sum_j \boldsymbol{\pi}_{i,j}) + \sum_j \beta_j(\boldsymbol{f}_j^t - \sum_i \boldsymbol{\pi}_{i,j}).\end{aligned} \quad (15)$$

Next, taking the partial derivative of $\mathcal{L}$ with respect to $\boldsymbol{\pi}_{i,j}$ yields:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\pi}_{i,j}} = \boldsymbol{C}_{i,j} - \epsilon \ln \boldsymbol{\pi}_{i,j} - \alpha_i - \beta_j = 0. \quad (16)$$

Solving this equation for $\boldsymbol{\pi}_{i,j}$:

$$\begin{aligned}\boldsymbol{\pi}_{i,j} &= \exp(-\frac{\boldsymbol{C}_{i,j} - \alpha_i - \beta_j}{\epsilon} - 1) \\ &= \exp(-\frac{\boldsymbol{C}_{i,j}}{\epsilon}) \exp(\frac{\alpha_i}{\epsilon}) \exp(\frac{\beta_j}{\epsilon}) \cdot e^{-1}.\end{aligned} \quad (17)$$

We define:

$$\boldsymbol{\mu}_i = \exp(\frac{\alpha_i}{\epsilon} - \frac{1}{2}), \quad \boldsymbol{\nu}_j = \exp(\frac{\beta_j}{\epsilon} - \frac{1}{2}). \quad (18)$$

Then, the solution is expressed as:

$$\boldsymbol{\pi}_{i,j} = \boldsymbol{\mu}_i \boldsymbol{K}_{i,j} \boldsymbol{\nu}_j. \quad (19)$$

Substitute the above solution into the marginal constraints yields:

$$\begin{aligned}\sum_j \boldsymbol{\pi}_{i,j} = \boldsymbol{\mu}_i \sum_j \boldsymbol{K}_{i,j}\boldsymbol{\nu}_j = \boldsymbol{f}_i^c, \\ \sum_i \boldsymbol{\pi}_{i,j} = \boldsymbol{\nu}_j \sum_i \boldsymbol{\mu}_i \boldsymbol{K}_{i,j} = \boldsymbol{f}_j^t.\end{aligned} \quad (20)$$

Algorithm 1 Our pipeline

Input: Image $\boldsymbol{I}$, class-specific textual descriptions $\{\boldsymbol{L}_i\}_{i=1}^{N_c}$
Parameter: degenerate map f^t, optimal path (or maximum velocity) operation $\mathbf{D}(\cdot)$
Output: the segmentation map M

$f = \text{Cos.}(f^I, f^L)$ ▷ Construct the logits.
$N_c = \text{unique}(\arg\max(f))$ ▷ Execute category early rejection to obtain the most probable category N_c.
$f = \text{Norm.}(\text{NMS}(f))$ ▷ Apply NMS and normalization to the logits.
for c to N_c **do**
  $m^c = \mathbf{D}(f^c, f^t)$ ▷ Execute the proposed method.
end for
$M = \arg\max_c(\{m^c\}_{c=1}^{N_c})$ ▷ Construct the segmentation map
return M

Thus,

$$\boldsymbol{\mu}_i = \frac{\boldsymbol{f}_i^c}{\sum_j \boldsymbol{K}_{i,j}\boldsymbol{\nu}_j}, \quad \boldsymbol{\nu}_j = \frac{\boldsymbol{f}_j^t}{\sum_i \boldsymbol{\mu}_i \boldsymbol{K}_{i,j}}. \quad (21)$$

In summary, this yields the Sinkhorn iteration format:

$$\boldsymbol{\mu}^{(l+1)} = \frac{\boldsymbol{f}^c}{\boldsymbol{K}\boldsymbol{\nu}^{(l)}}, \quad \boldsymbol{\nu}^{(l+1)} = \frac{\boldsymbol{f}^t}{\boldsymbol{K}^\top \boldsymbol{\mu}^{(l+1)}}. \quad (22)$$

7. More Details

Non-maximum suppression. Given that existing visual language models are constrained by coarse-grained multi-modal training paradigms, the resulting logits often contain numerous misaligned patches, which serve as noise and interfere with downstream fine-grained tasks. In this work, this noise disrupts the distribution transmission process, particularly for similar patches, resulting in consistent differences in distribution between noisy and clean patches. Consequently, noise removal is crucial. Although numerous methods have made significant contributions, most concentrate on noise removal while preserving clean logit regions, necessitating precise localization of noise regions. We propose to treat low-confidence patches as noise and set their values to $-\infty$, thereby ensuring a probability distribution of zero in the softmax-normalized output. Specifically, we define patches with confidence less than 0.9 as noise. This means that it is possible to establish the distribution discrepancy representing semantic information by relying only on a small subset of reliable logits distribution.

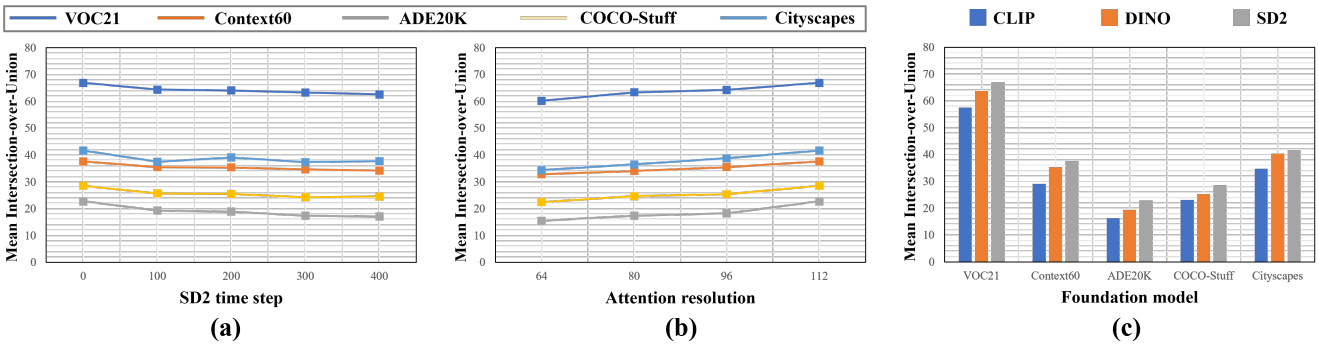


Figure 6. Quantitative evaluation of SD2 time step, attention resolution, and foundation model on standard benchmarks is presented, with each metric represented by a distinct color (unit: %). Here, SD2 time step denotes the denoising step in SD2, and attention resolution refers to the input size of SD2. All comparisons were performed using the optimal path mode.

Normalization. We adopt $\mathbf{softmax}$ operation to obtain the normalized logits.

Joint bilateral upsampling. Joint Bilateral Upsampling (JBU) is an edge-preserving image upsampling technique that integrates spatial and range information. It is widely employed in computer vision to align low-resolution processing results, such as segmentation maps and depth maps, with high-resolution reference images, including the original input. The core principle of JBU involves leveraging the texture structure of the high-resolution reference image to guide the upsampling of low-resolution features, thereby preventing edge blurring. The JBU process is defined as:

$$D^H(p) = \frac{1}{k_p} \sum_{q \in \Omega} D^L(q) \cdot f(\|p-q\|) \cdot g(\|I^H(p) - I^H(q)\|), \tag{23}$$

where D^L and I^H denote low-resolution segmentation maps and high-resolution RGB image. D^H denote the upsampled results. p denotes pixel location in a high-resolution image. q denotes pixel positions in the neighborhood Ω centered at p. $k_p = \sum_{q \in \Omega} f(\|p-q\|) \cdot g(\|I^H(p) - I^H(q)\|)$. $f(\cdot)$ and $g(\cdot)$ denote space gaussian kernel and range gaussian kernel. Usually, $f(x) = e^{-\frac{x^2}{\sigma_s^2}}$, and $g(x) = e^{-\frac{x^2}{\sigma_r^2}}$. We set $\sigma_s^2 = 1, \sigma_r^2 = 0.1$.

Category early rejection. When the number of categories is large, inference time increases substantially. To mitigate this issue while leveraging the observation that most images contain only a few semantic categories, we apply the $\arg\max$ operation to the normalized logits to identify the most probable category before executing the proposed method.

Table 4. Effect of different strategy for our components (unit: %). **q-q mean**: the average attention tensor between queries across all layers. **k-k mean**: the average attention tensor between keys across all layers. All comparisons were performed using the optimal path mode.

	Variants	VOC21	COCO-Stuff	Cityscapes	ADE20K	Avg.
self-attention weight combination ($down_0, down_1, up_0, up_1, up_2$)						
(I)	$(1,0,0,0,0)$	61.1	23.5	36.4	16.5	34.4
(II)	$(0,1,0,0,0)$	62.2	25.1	38.2	17.6	35.8
(III)	$(0,0,1,0,0)$	64.0	25.6	40.6	18.3	37.1
(IV)	$(0,0,0,1,0)$	65.1	26.4	41.5	20.8	38.5
(V)	$(0,0,0,0,1)$	64.1	25.4	41.3	19.2	37.5
(VI)	$(0,0,0.5,0.5,0)$	66.9	28.6	41.7	22.8	40.0
training-free logits-optimization methods						
(I)	origin	62.6	25.2	36.3	16.3	35.1
(II)	q-q mean	64.3	26.2	41.2	20.5	38.1
(III)	k-k mean	66.9	28.6	41.7	22.8	40.0

Pseudo algorithm. The algorithm is illustrated in Algorithm 1.

8. More results

Ablation about optimal path. We conduct component ablation experiments under the optimal path mode. As illustrated in Figure 6, our analysis reveals that the effect of denoising step length confirms that single-step denoising generates deterministic self-attention tensors with optimal performance. Moreover, higher resolution of the attention tensor generally correlates with improved performance. Consequently, SD2 was selected as the self-attention tensor extraction model to achieve the best results. Table 4 demonstrates the self-attention weight combination and training-free logits-optimization strategy under this mode, with results consistent with the maximum velocity mode.

Computational complexity analysis. To verify the efficiency of our method, we conduct efficiency analysis on

VOC21 benchmark using an NVIDIA RTX 3090 GPU. The experiments were performed under default settings: input resolution of 512×512 pixels, SD2 time step set to 0, and the logits model scale set to base/16. As shown in the table (please zoom in for details), our method achieves an optimal balance between performance and computational efficiency. For example, compared to CASS, our approach delivers superior performance while maintaining faster inference speed. In addition, we conduct a detailed breakdown of the inference time for each component. The "Logits" and "Attention" times represent the computational costs for generating logits via CLIP and attention maps via SD2, respectively. "Distribution" refers to the processing time of our proposed method (including Optimal Path (O.P.) & Maximum Velocity (M.V.)), while "JBU" denotes the time required for upsampling operation. Our analysis reveals that the primary bottleneck in inference time stems from generating attention maps using SD2. Even when setting the time step to 0 without introducing noise and performing only the intermediate attention map calculation, the overall inference speed remains constrained by this component. In contrast, O.P.& M.V. introduce minimal computational overhead, requiring only approximately 0.1 seconds of processing time (50 iterations for O.P. and $\tau = 0.3$ for M.V. can converge quickly).

Model	FLOPs(G)↓	Params(M)↓	Inference time(sec.)↓	mIoU(%)↑
CLIP	52.2	149.6	0.08	18.6
ProxyCLIP	103.2	235.4	0.16	59.1
CASS	1675.5	265.7	3.11	65.8
Ours (O.P.)	351.3	1006.8	0.59(-2.52)	66.9(+1.1)
Ours (M.V.)	339.2	1006.8	0.57(-2.54)	67.8(+2.0)

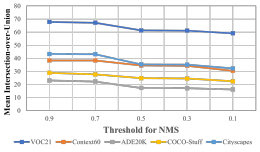
the inference time for each component

Model	logits time(sec.)↓	attention time(sec.)↓	distribution time(sec.)↓	JBU(sec.)↓
Ours (O.P.)	0.08	0.25	0.10	0.10
Ours (M.V.)	0.08	0.25	0.12	0.10

VFM integration for structural/spatial priors. Since VFMs provide good self-attention (spatial prior), its integration into OVSS has become common practice. For existing methods (including ours), integrating high/low-quality self-attention of VFMs inevitably increases/degrades performance. However, our approach demonstrates two advantages based on high- and low-quality self-attention. ① Integrating low-quality self-attention, our method demonstrates excellent robustness—even integrating the original CLIP's self-attention still achieves high performance. While the original CLIP suffers severe performance degradation due to its low-quality self-attention, our method achieves an average improvement of 24.4 points, demonstrating its excellent robustness to lower-quality self-attention. ② Several SOTA methods generate high-quality self-attention and achieve excellent performance. By integrating these approaches, our method achieves further improvements, demonstrating its ability to fully leverage high-quality self-attention. In addition, we observe that integrating DINOv2 with Registers yields stronger performance improvements due to its higher-quality self-attention. Therefore, while a more powerful VFM would undoubtedly improve performance (due to its higher-quality spatial priors), we emphasize that our method does not rely on model-specific attention improvement—enabling flexible integration of diverse VFMs.

Model	VOC21	Context60	ADE20K	COCO-Stuff	Cityscapes
original CLIP-B/16	18.6	7.8	3.2	7.2	6.7
integrating self-attention via maximum velocity					
- CLIP-B/16	58.5(+39.9)	30.1(+22.3)	17.3(+14.1)	24.1(+16.9)	35.7(+29.0)
- DINO-B/8	64.8	36.3	20.4	26.3	41.3
- DINOv2-B/14	66.9	36.9	21.3	27.9	42.7
- DINOv2-B/14 w/ Registers	**69.8**	*38.0*	*22.3*	**29.3**	**44.6**
- SD2	*67.8*	**38.3**	**23.0**	*28.9*	*43.3*

NMS analysis. With a higher NMS threshold, we obtain more reliable logits patches despite fewer filtered patches, without damaging performance. Conversely, lowering the threshold to get more patches (not necessarily more reliable) impairs performance, as evidenced by the significant drop below 0.5 threshold in the figure below (please zoom in for details).



Versatility analysis. Our method can indeed be regarded as a flexible and general post-processing step that operates independently of any specific setup. As demonstrated in the table (please zoom in for details), we apply our approach to several SOTA methods, including RF-CLIP, CASS, and SC-CLIP. The experimental results show that integrating our method leads to significant performance improvements across existing SOTA approaches. We observe an approximate 5-point improvement on the Context59 benchmark, along with an average gain of 3 points across other benchmark datasets. These consistent improvements demonstrate that our method functions as a universal enhancement method for open-vocabulary tasks.

Model	Scale	VOC21	Context60	COCO-Obj	VOC20	Context59	COCO-Stuff	Cityscapes	ADE20K
CASS	B/16	65.8	36.7	27.8	97.8	40.3	28.7	39.4	20.4
w/ O.P.		68.2(+2.4)	38.7(+2.0)	40.1(+12.3)	89.6(-8.2)	45.5(+5.2)	29.4(+0.7)	42.8(+3.4)	22.9(+2.5)
w/ M.V.		69.5(+3.7)	39.1(+2.4)	40.8(+13.0)	90.0(-7.8)	45.9(+5.7)	29.8(+1.1)	44.4(+5.0)	23.1(+2.7)
SC-CLIP	B/16	64.6	36.8	37.7	84.3	40.1	26.6	41.0	20.1
w/ O.P.		68.1(+3.5)	38.7(+1.9)	40.8(+3.1)	89.2(+4.9)	45.3(+5.2)	29.3(+2.7)	42.9(+1.9)	22.8(+2.7)
w/ M.V.		69.3(+4.7)	39.1(+2.3)	40.8(+3.1)	89.6(+5.3)	45.7(+5.6)	29.8(+3.2)	44.6(+3.6)	23.0(+2.9)
RF-CLIP	B/16	64.8	36.4	37.9	87.0	39.8	26.3	41.3	20.4
w/ O.P.		67.8(+3.0)	38.3(+1.9)	40.3(+2.4)	89.5(+2.5)	44.7(+4.9)	29.3(+3.0)	42.9(+1.6)	22.9(+2.5)
w/ M.V.		69.1(+4.3)	39.0(+2.6)	40.8(+2.9)	90.0(+3.0)	45.6(+5.8)	29.7(+3.4)	44.4(+3.1)	23.1(+2.7)
SC-CLIP	L/14	65.0	36.9	40.5	88.3	40.6	26.9	41.3	21.7
w/ O.P.		68.3(+3.3)	39.0(+2.1)	42.7(+2.2)	91.5(+3.2)	45.8(+5.2)	29.7(+2.8)	43.4(+2.1)	23.5(+1.8)
w/ M.V.		70.0(+5.0)	39.8(+2.9)	43.2(+2.7)	91.9(+3.6)	46.2(+5.6)	30.0(+3.1)	45.0(+3.7)	23.6(+1.9)
RF-CLIP	L/14	65.8	36.7	41.8	89.1	40.2	26.7	41.4	22.4
w/ O.P.		69.5(+3.7)	38.7(+2.0)	43.8(+2.0)	91.6(+2.5)	45.5(+5.3)	29.5(+2.8)	43.4(+2.0)	24.3(+1.9)
w/ M.V.		70.1(+4.3)	39.5(+2.8)	44.5(+2.7)	92.0(+2.9)	46.0(+5.8)	30.0(+3.3)	45.0(+3.6)	24.6(+2.2)

Discussion about Label Propagation Over Patches and Pixels for Open vocabulary Semantic Segmentation[41]. Paper[41] essentially proposes a method for refining labels, which still follows the idea of logit-optimization, continuously pushing the logit towards the true label distribution. However, we don't care about the true label distribution; we only care about the difference between the degenerate distribution and the logit distribution, thus eliminating the need for logit-optimization. Therefore, our proposed method is fundamentally different from paper[41].

More visualization. See the following figures.

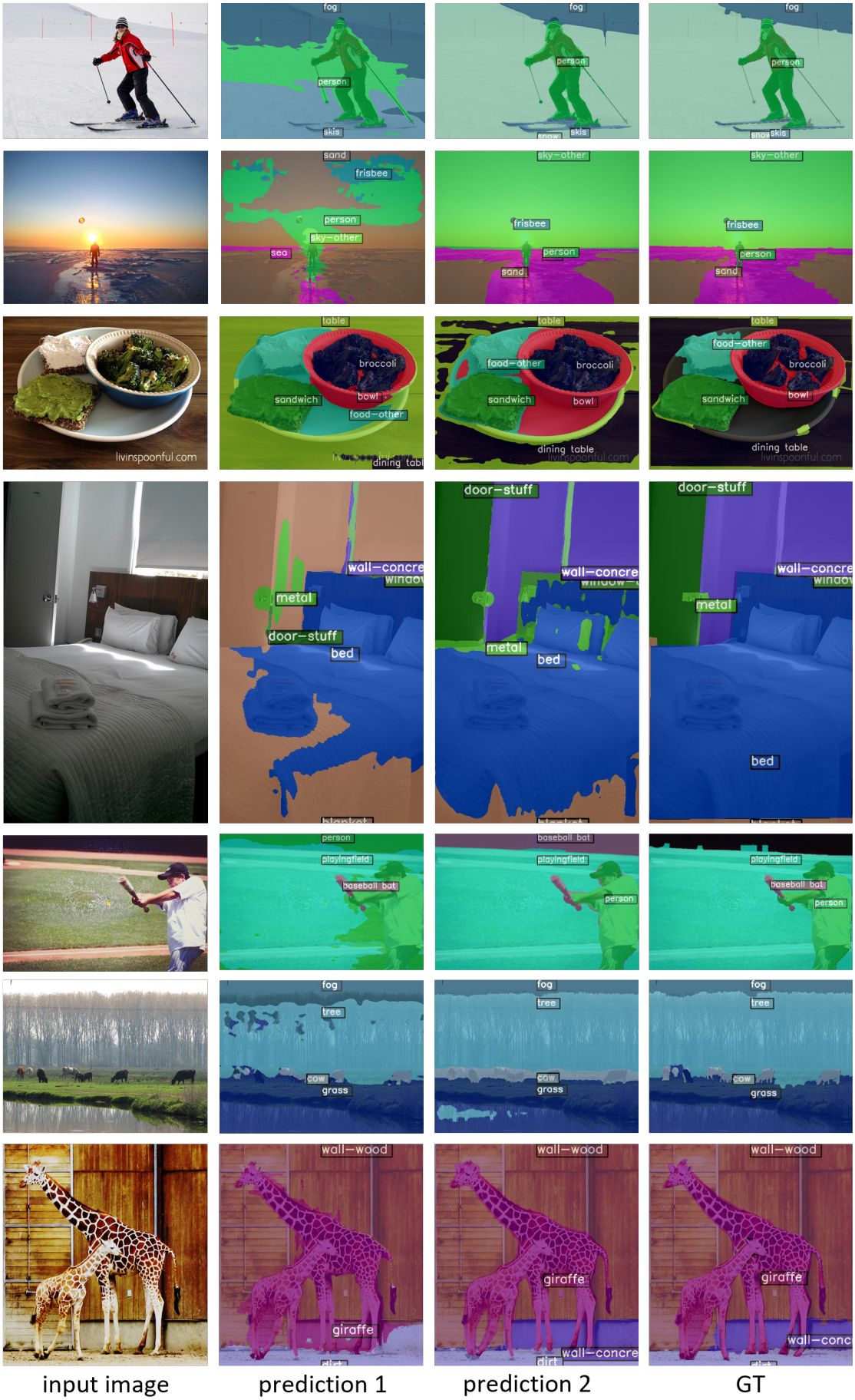


Figure 7. Visualization of segmentation maps on COCO-Stuff benchmark dataset.

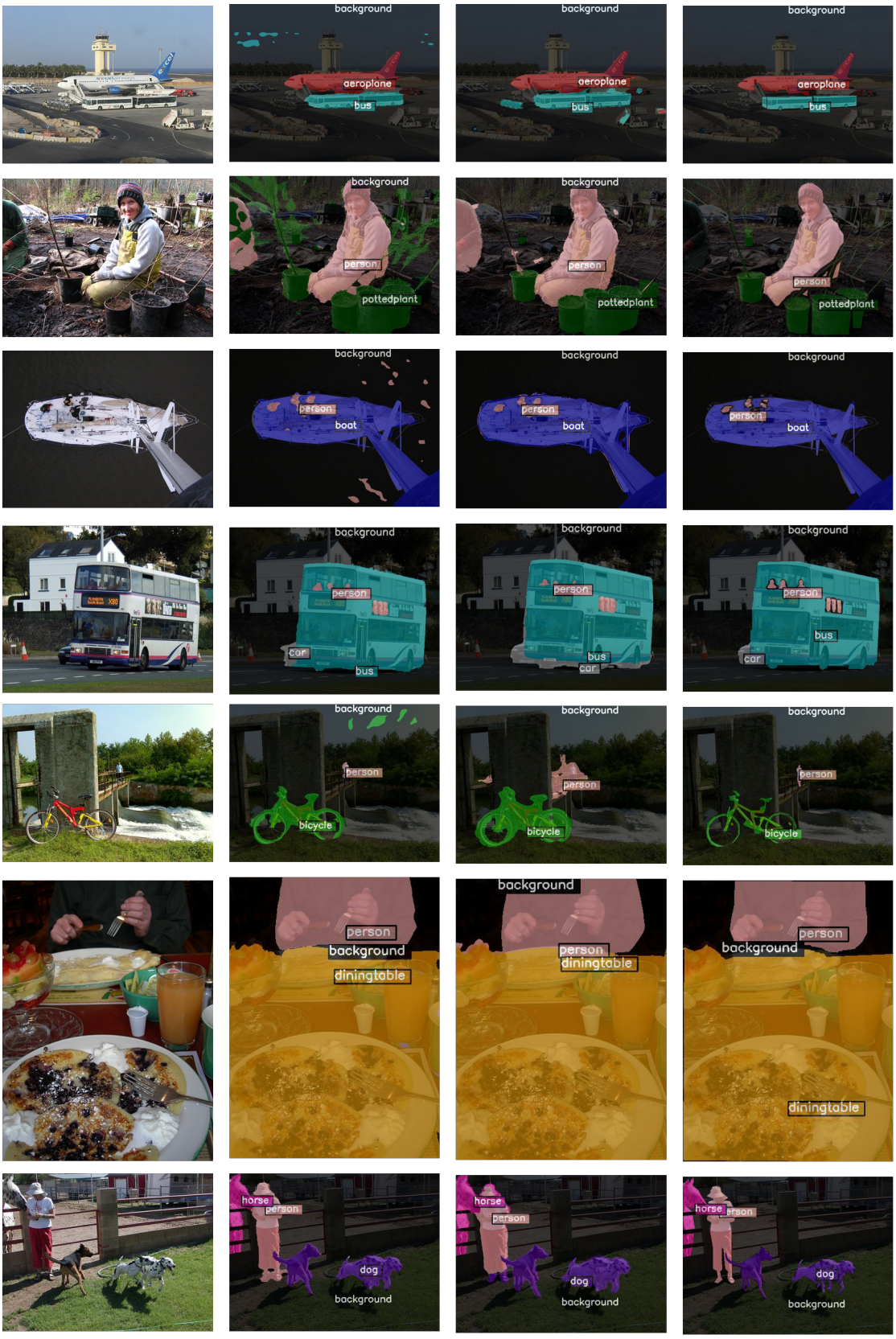


Figure 8. Visualization of segmentation maps on Pascal VOC benchmark dataset.

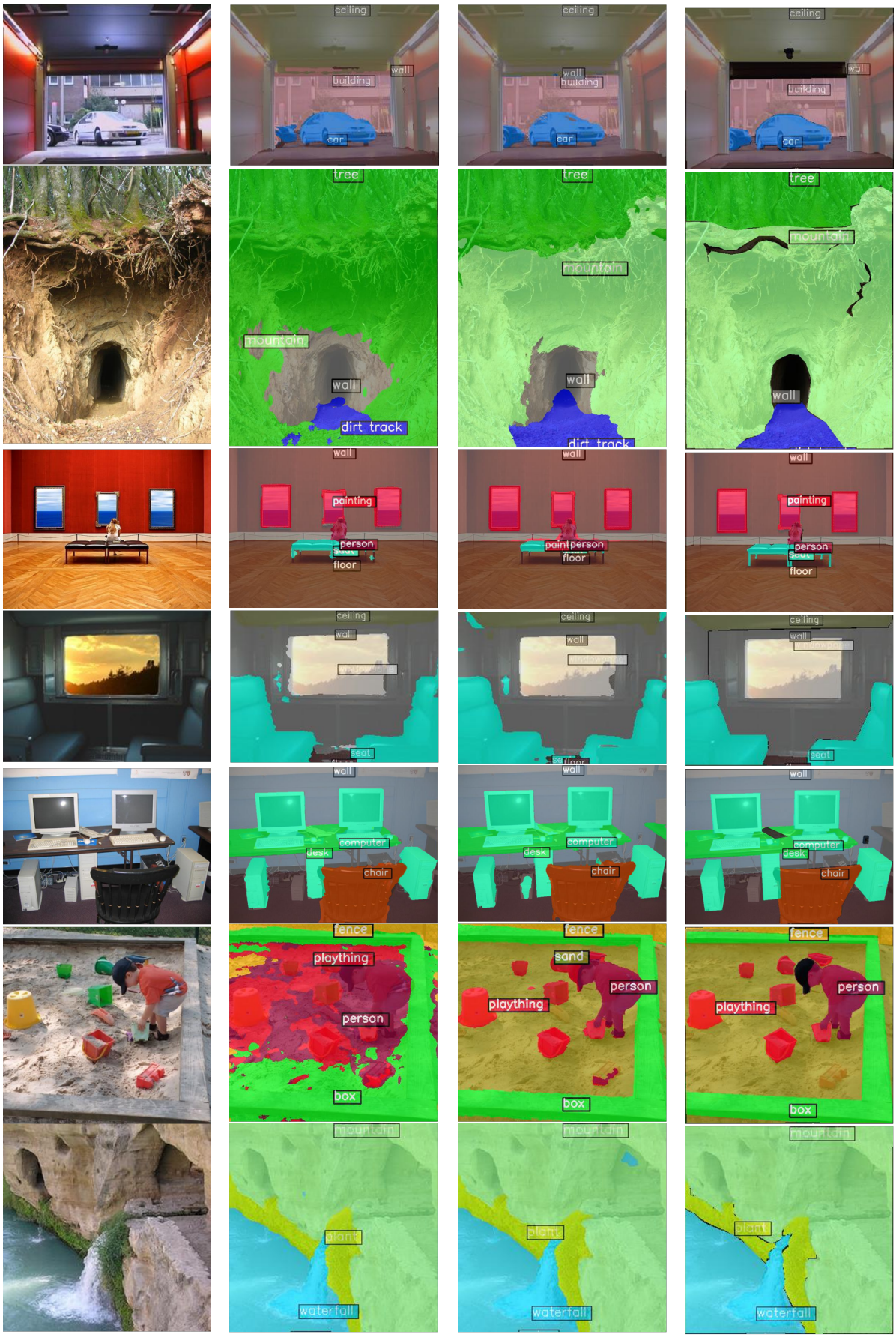


Figure 9. Visualization of segmentation maps on ADE150k benchmark dataset.

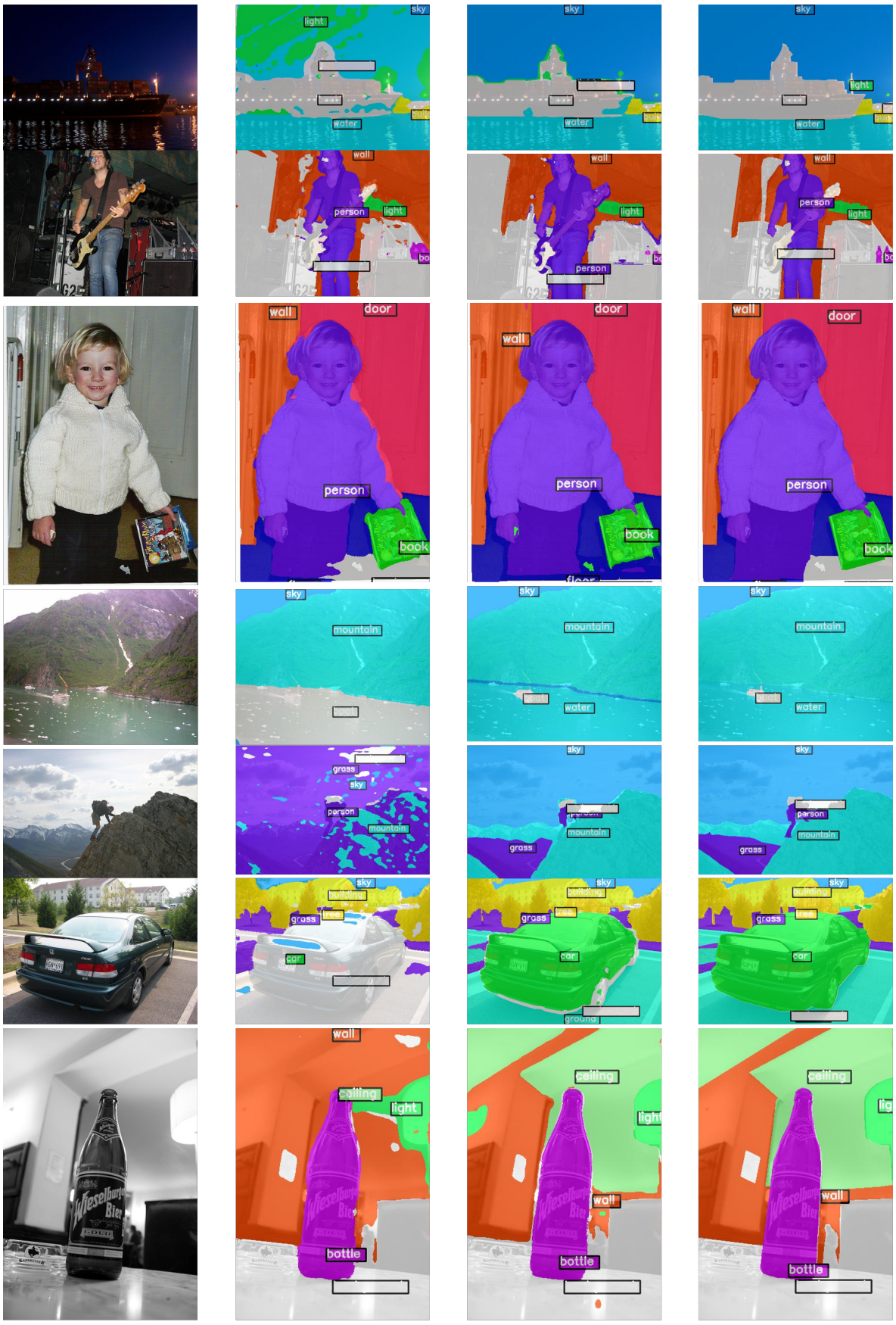


Figure 10. Visualization of segmentation maps on Pascal Context benchmark dataset.

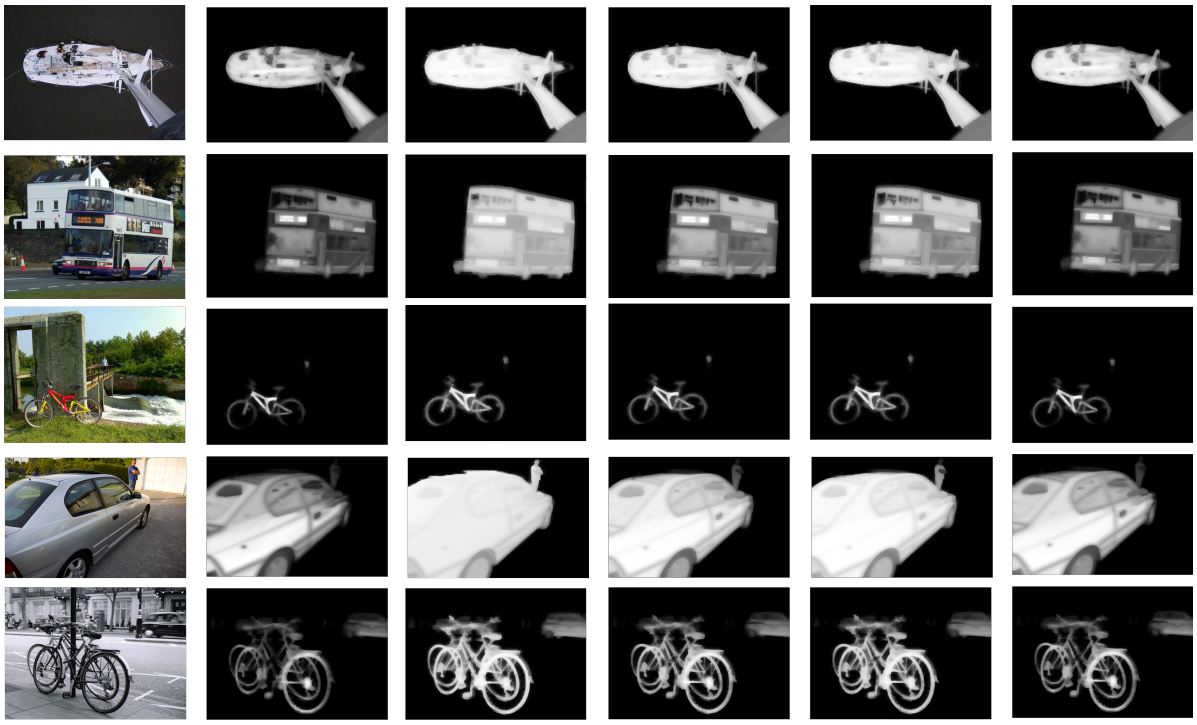


Figure 11. Visualization of maximum velocity maps across different threshold.

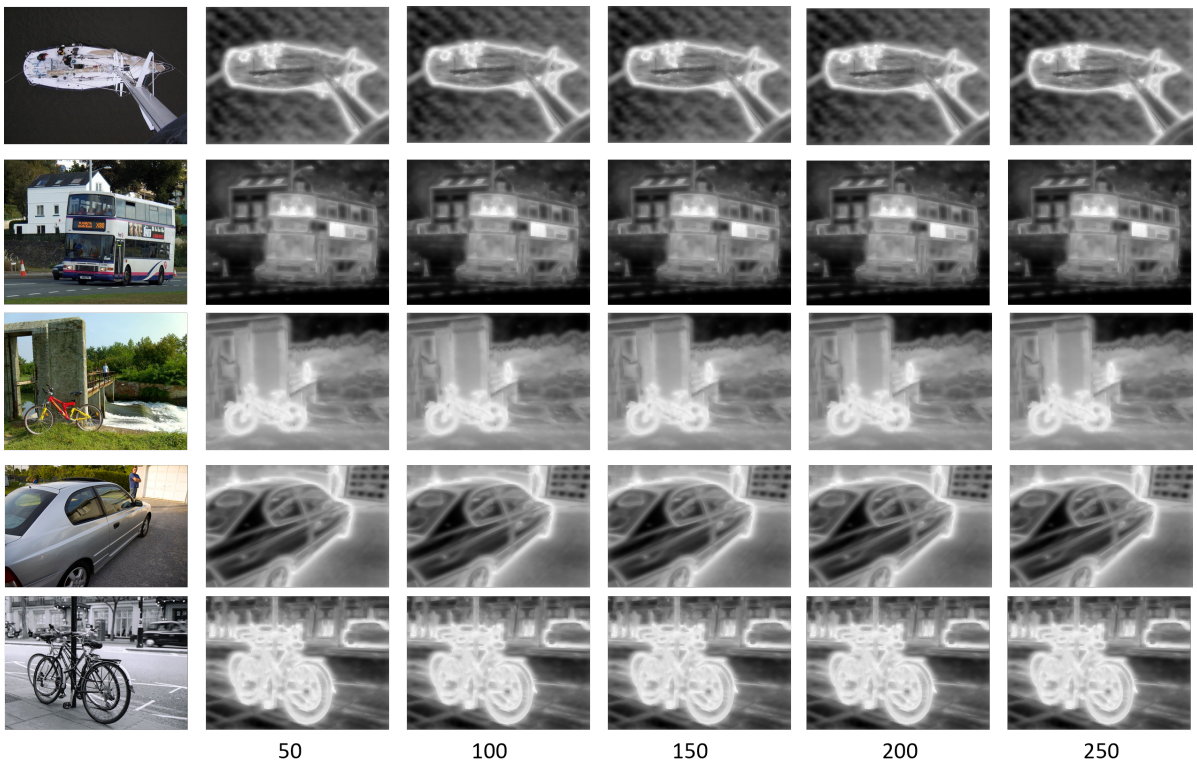


Figure 12. Visualization of optimal path maps across different iterations.