A Algorithm of MPS

As discussed in Sec. 3.1, we adopt the log-scaling version of Sinkhorn approaches for our proposed Multiple Prompt Sinkhorn (MPS) algorithm as follow:

Algorithm 1: Multiple Prompt Sinkhorn Algorithm

```
  Given: The feature map size $M = HW$, the number of prompts $N$,
         $\mu = \mathbf{1}^M/M$ , $\nu = \mathbf{1}^N/N$, the score map $\mathbf{S}$ ;
  Input  : The cost matrix $\mathbf{C} = \mathbf{1} - \mathbf{S}$, hyper-paramter $\epsilon$, the max iteration $t_{\max}$;
  Initialization: $\mathbf{K} = \exp(-\mathbf{C}/\epsilon)$, $t \leftarrow 1, \mathbf{b}^0 = 0$;
1 while $t \le t_{max}$ and not converge do
2     $\mathbf{a}^t = \log \mu - \log\left[\sum \exp\left[-\frac{1}{\epsilon}\mathbf{C} + \mathbf{b}^{t-1})\right]\right]$;
3     $\mathbf{b}^t = \log \nu - \log\left[\sum \exp\left[-\frac{1}{\epsilon}\mathbf{C}^\top + \mathbf{a}^{t-1})\right]\right]$;
  Output: Optimal transport plan $\mathbf{T}^* = \mathrm{diag}(\exp(\mathbf{a}))^t\mathbf{K}\mathrm{diag}(\exp(\mathbf{b}))^t$ ;
```

B Details of Loss function

As discussed in Sec. 4.3, we combine three different losses, including the focal loss based on Cross Entropy (CE) loss, and the dice loss, which are given by:

$$\begin{aligned}\mathcal{L}_{\mathrm{ce}} = -\frac{1}{HW}\sum_{i=1}^{HW} \mathbf{Y}_i^{\mathrm{gt}} \log(\phi(\mathbf{Y}_i)) \\ +(1-\mathbf{Y}_i^{\mathrm{gt}})\log(1-\phi(\mathbf{Y}_i)),\end{aligned} \tag{19}$$

$$\begin{aligned}\mathcal{L}_{\mathrm{focal}} = -\frac{1}{HW}\sum_{i=1}^{HW} \mathbf{Y}_i^{\mathrm{gt}}(1-\sigma(\mathbf{Y}_i))^\gamma \log(\sigma(\mathbf{Y}_i)) \\ +(1-\mathbf{Y}_i^{\mathrm{gt}})\sigma(\mathbf{Y}_i)^\gamma\log(1-\sigma(\mathbf{Y}_i)),\end{aligned} \tag{20}$$

$$\mathcal{L}_{\mathrm{dice}} = 1 - \frac{2\sum_{i=1}^{HW}\mathbf{Y}_i^{\mathrm{gt}}\mathbf{Y}_i}{\sum_{i=1}^{HW}{\mathbf{Y}_i^{\mathrm{gt}}}^2 + \sum_{i=1}^{HW}{\mathbf{Y}_i}^2}, \tag{21}$$

where $\mathbf{Y}$ is the model decoder outputs, $\mathbf{Y}^{\mathrm{gt}}$ is the ground truth label, $\sigma(\cdot)$ is Sigmoid operations, γ is a hyper-parameter to balance hard and easy samples, which is set to 2. Throughout the entire experiments, λ_{ce}, λ_{focal} and λ_{dice} are set to 1, 20, and 1, respectively.

C Relation Descriptor

As discussed in Sec. 4.1, when an image is inputted into the CLIP image encoder, we obtain the ([CLS] token) $\bar{\mathbf{h}}_{\mathrm{img}} \in \mathbb{R}^{1\times D}$ and the pixel embedding $\mathbf{h}_{\mathrm{img}}$ in the last layer. To merge these CLIP's text and pixel embeddings, we utilize the relationship descriptor following an approach introduced in [38] to yield the refined text embedding $\mathbf{h}_{\text{txt}}$ as follows:

$$\tilde{\mathbf{h}}_{\text{txt}} = \mathcal{R}_{\psi}(\text{cat}\left[\mathbf{h}_{\text{img}} \odot \mathbf{h}_{\text{txt}}, \mathbf{h}_{\text{txt}}\right]) \in \mathbb{R}^{KN \times D}$$

where $\mathcal{R}_{\psi}$ denotes a linear layer for matching the concatenated embedding dimension to the original dimension D, and cat is the concatenation operator, $\odot$ is the Hadamard product.

D Definition of Harmonic mean IoU

Following the previous works [35,37,38], we define harmonic mean IoU (hIoU) among the seen (S) and unseen (U) classes as:

$$\text{hIoU} = \frac{2 * \text{mIoU (S)} * \text{mIoU (U)}}{\text{mIoU (S)} + \text{mIoU (U)}} \tag{22}$$

E Details of Dataset

We utilize a total of three datasets, *i.e.,* VOC 2012 [11], PASCAL Context [20], and COCO-Stuff164K [3]. We divide seen and unseen classes for each dataset, following the settings of previous methods [2,10,35,37,38]. VOC 2012 consists of 10,582 / 1,449 images with 20 categories, for training / validation. The dataset is divided into 15 seen and 5 unseen classes. PASCAL Context is an extensive dataset of PASCAL VOC 2010 that contains 4,996 / 5,104 images for training / test with 60 categories. The dataset is categorized into 50 seen and 10 unseen classes. COCO-Stuff 164K is a large-scale dataset that consists of 118,287 / 5,000 images for training / validation with 171 classes. The dataset is categorized into 156 seen and 15 unseen classes.

F Implementation Detail

We further declare the implementation detail for our work. Input image resolution is set as 480×480 for PASCAL Context, and 512×512 for the rest of the datasets. For training, we choose the lightest training schedule. For the inductive settings, the total training iterations were 20K for VOC 2012, 40K for PASCAL Context, and 80K for COCO-Stuff164K. For the transductive settings, the model was trained on seen classes in the first half of training iterations and then applied self-training by generating pseudo-labels in the remaining iterations, following previous approaches [37,38]. The balance factor λ for the score map in Eq. (17) is set to 0.5.

G Further Comparison Studies

To further validate the generalizability of our proposed method in the open-vocabulary segmentation (OVS) setting, we compared our OTSeg and OTSeg+ with FreeSeg [25] and SAN [34]. It's noteworthy that OTSeg is designed for ZS3, utilizing a partial dataset of 156 classes, whereas FreeSeg and SAN use the entire COCO-Stuff 171 classes dataset tailored for OVS. Despite differing goals and settings, our proposed OTSeg variants demonstrate comparable results in cross-dataset settings, as shown in Table S1, validating their effectiveness.

Table S1: Cross-data comparison with open-vocabulary segmentation methods.

Category	Method	Source	ADE20K	PC59	VOC
Zero-shot segmentation (ZS3)	OTSeg	COCO-156	21.9	52.9	94.2
	OTSeg+		21.1	53.4	94.4
Open-vocabulary segmentation (OVS)	Freeseg [25]	COCO-171	17.9	34.4	85.6
	SAN [34]		27.5	53.8	94.1

H Robustness of OTSeg with Multiple Seeds

To evaluate the robustness of our proposed Multiple Prompt Sinkhorn (MPS) algorithm, we repeatedly trained both OTSeg and OTSeg+ with multiple seeds and compared the results on the VOC dataset. Table S2 confirms the robustness and consistent performance of our proposed method across various seeds.

Table S2: Quantitative results with multiple seeds on VOC dataset and comparison in cross-data settings with further approaches.

Settings	Method	Seed				Mean ± STD
		#1	#2	#3	#4	
Inductive	OTSeg	84.5	84.9	84.3	84.0	84.4 ± 0.4
	OTSeg+	87.1	88.0	88.0	86.5	87.4 ± 0.7
Transductive	OTSeg	94.2	94.6	94.2	94.3	94.4 ± 0.2
	OTSeg+	94.3	94.5	94.2	94.3	94.3 ± 0.1

I Further Component Analysis

In this section, we conduct additional component analysis by varying the conditions comprising our proposed method.

Balance Factor In OTSeg+, we generate the final prediction by ensembling the decoder path and score map path as indicated in Eq. (17). Tab. S3 (a) represents the analysis of the balance factor λ. In inductive settings, the effect of the balance factor is significant compared to transductive settings. We empirically determine our default configuration of the balance factor as 0.5.
Learnable Module To validate the rationale behind our choice of the learnable module, we conduct experiments by altering the conditions outlined in Tab. S3 (b). We observe that training both the visual prompt in the image encoder and the text prompt in the text module results in a significant drop in performance due to overfitting. While tuning only the learnable text prompt provide relatively improved performance, we observe that our adoption of only tuning the image encoder via visual prompt tuning leads to the best performance.

Table S3: Further component analysis on the VOC 2012 dataset.

Component	Ours	Factor λ	Module	Inductive setting			Transductive setting		
				mIoU(U)	mIoU(S)	hIoU	mIoU(U)	mIoU(S)	hIoU
(a) Balance factor		0.1		79.6	92.9	85.7	94.2	94.2	94.2
		0.3		80.4	92.8	86.2	94.2	**94.3**	94.2
	✓	0.5		**81.6**	**93.3**	**87.1**	**94.4**	**94.3**	**94.3**
		0.7		80.6	93.1	86.4	94.3	94.2	94.2
		0.9		78.8	92.1	84.9	94.1	94.2	94.1
(b) Learnable Module	✓		Decoder, Visual Prompts	**81.6**	**93.3**	**87.1**	**94.4**	**94.3**	**94.3**
			Decoder, Visual Prompts, Text Prompts	35.9	85.6	50.7	48.6	94.1	64.1
			Decoder, Text Prompts	42.9	89.8	58.1	45.6	91.9	61.0

J Analysis on Sinkhorn Convergence Comparison Studies

We further analyzed the convergence of the Sinkhorn algorithm and its effect on performance. We established a defined threshold and applied the Sinkhorn algorithm iteratively until the total errors dropped below this threshold. In Figure S1, we report that the Sinkhorn algorithm converged in 6 iterations for the score map and 14 iterations for the OT-adapted decoder part on the VOC test dataset. Additionally, we observed that the convergence of the Sinkhorn algorithm around 15K training iterations led to the best performance.

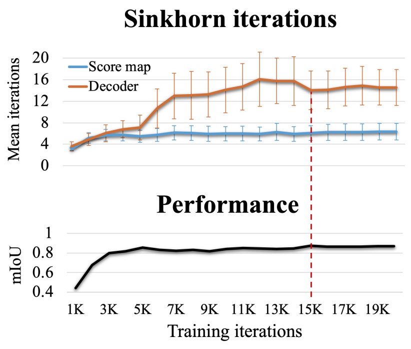

Fig. S1: Sinkhorn convergence.

K Additional Visual Results

We provide additional visual results. Fig. S2 show qualitative zero-shot segmentation performance for COCO-Stuff164K dataset. We reproduce the segmentation results using publically available weights [37, 38]. Our OTSeg demonstrates superior performance in precisely sectioning boundaries of both seen and unseen objects, unlike previous SOTA method which fail to classify the category or produce noisy results.

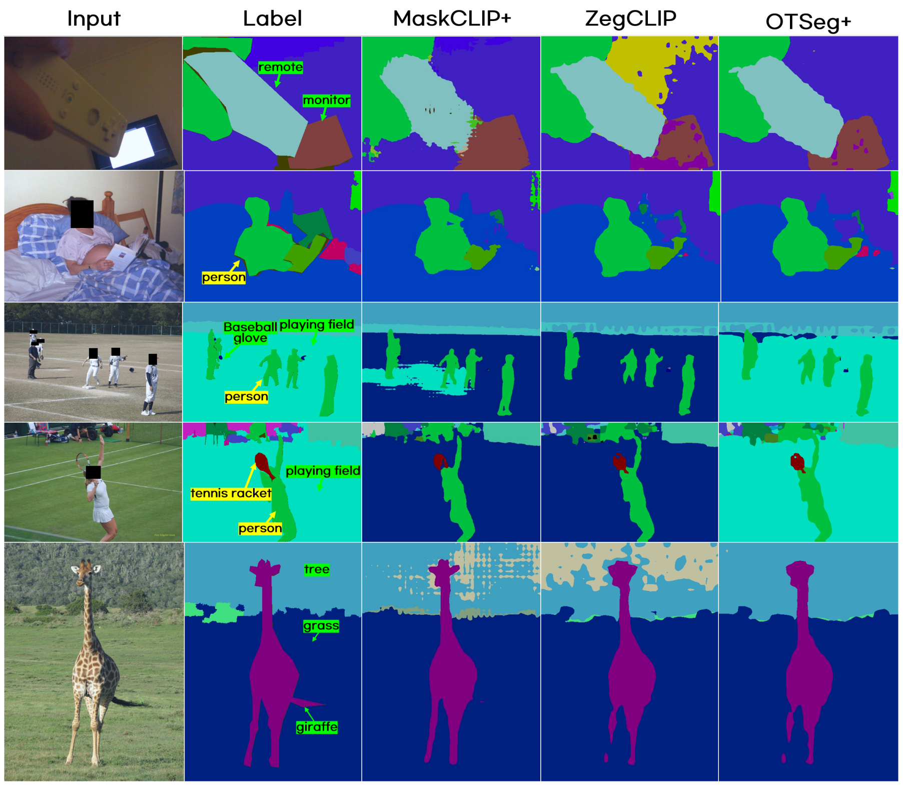

Fig. S2: Qualitative zero-shot segmentation results of COCO-Stuff164K dataset. The yellow tag indicates seen classes, while the green tag indicates unseen classes.

L Further Score Map Comparison with or without MPSA

We provide additional visual comparison of prompt-related score map with or without MPSA in Fig. S3. While all the text prompt-related score map S^i are cohered without MPSA (white arrows), with our MPOT, each S^i focuses on different semantic attributes (red arrows), which helps the model utilizes various score maps to differentiate each class name-driven object boundary.

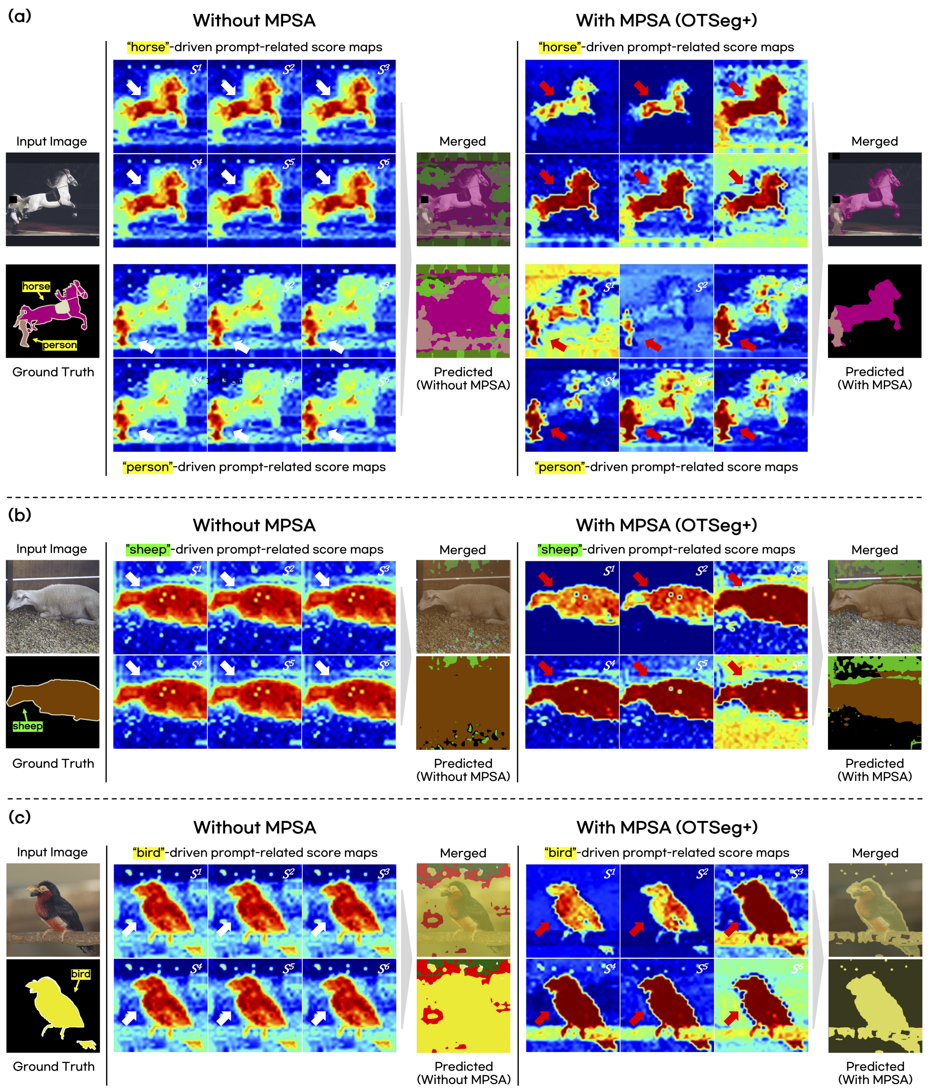

Fig. S3: Further visual comparison of class name-driven prompt-related score maps. While all the text prompt-related score maps S^i are cohered without MPSA (white arrows), with MPSA within our proposed OTSeg+, each S^i is diversely activated (red arrows) to help the model segment each object sharply. The yellow tag indicates seen classes, while the green tag indicates unseen classes.