Cascade-CLIP: Cascaded Vision-Language Embeddings Alignment for Zero-Shot Semantic Segmentation

Yunheng Li [1] **Zhong-Yu Li** [1] **Quansheng Zeng** [1] **Qibin Hou** [1 2] **Ming-Ming Cheng** [1 2]

Abstract

Pre-trained vision-language models, *e.g.,* CLIP, have been successfully applied to zero-shot semantic segmentation. Existing CLIP-based approaches primarily utilize visual features from the last layer to align with text embeddings, while they neglect the crucial information in intermediate layers that contain rich object details. However, we find that directly aggregating the multi-level visual features weakens the zero-shot ability for novel classes. The large differences between the visual features from different layers make these features hard to align well with the text embeddings. We resolve this problem by introducing a series of independent decoders to align the multi-level visual features with the text embeddings in a cascaded way, forming a novel but simple framework named Cascade-CLIP. Our Cascade-CLIP is flexible and can be easily applied to existing zero-shot semantic segmentation methods. Experimental results show that our simple Cascade-CLIP achieves superior zero-shot performance on segmentation benchmarks, like COCO-Stuff, Pascal-VOC, and Pascal-Context. Our code is available at https://github.com/HVision-NKU/Cascade-CLIP.

1. Introduction

Semantic segmentation, as one of the fundamental topics in computer vision, has achieved remarkable success in predicting the category of each pixel of an image (Chen et al., 2021; Huang et al., 2021; Xie et al., 2021; Cheng et al., 2022a). However, semantic segmentation models (Zhao et al., 2017; Zeng et al., 2022) trained on closed-set annotated images are only capable of segmenting the predefined categories.

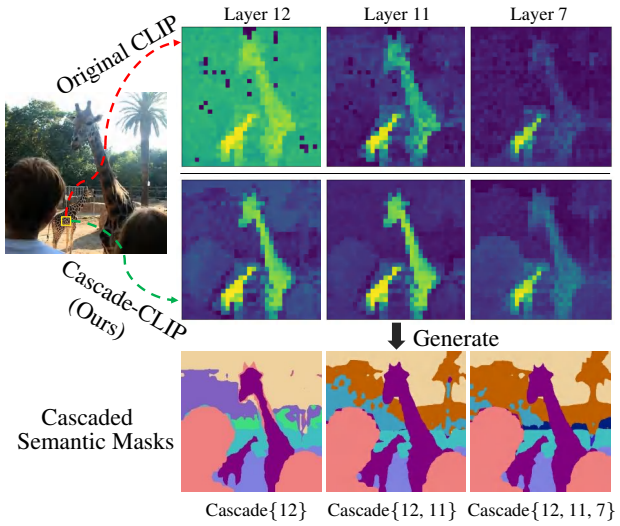


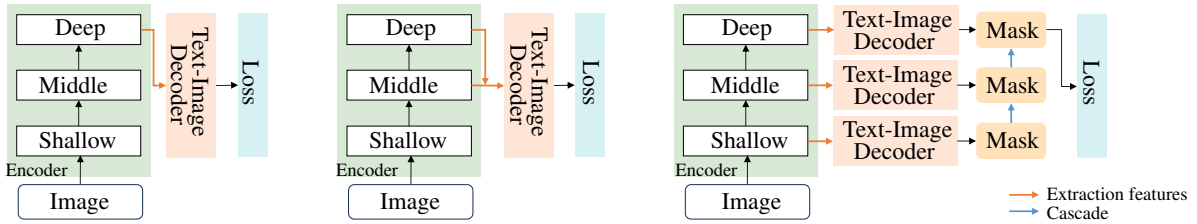
Figure 1. Motivation illustration of Cascade-CLIP. The cosine similarity map (above) indicates the visual features from the intermediate layers of CLIP (Radford et al., 2021) layers can capture richer local object details compared to the last one (Layer 12).

This motivates some researchers to study the zero-shot semantic segmentation models (Bucher et al., 2019; Gu et al., 2020; Han et al., 2023b), which can segment categories that are not even present in the training images and is attracting more and more attention.

Recently, benefiting from the impressive zero-shot capability at the image level, large-scale visual-language pre-training models, typified by CLIP (Radford et al., 2021), have been considered in zero-shot semantic segmentation. Nevertheless, directly applying CLIP to the zero-shot semantic segmentation task is ineffective as it requires dense pixel/region-wise predictions. Two-stage methods (Xu et al., 2022b; Ding et al., 2022) solve the aforementioned issue via generating region proposals by trained proposal generator and feeding the cropped masked regions to CLIP for zero-shot classification. Although this paradigm retains CLIP's image-level zero-shot ability well, it introduces high computational cost. The one-stage line (Zhou et al., 2023; Xu et al., 2024) produces pixel-wise segmentation by matching text embeddings and pixel-level features extracted from the last layer of CLIP's visual encoder, achieving a good balance between efficiency and effectiveness. However, these

[1]VCIP, College of Computer Science, Nankai University [2]Nankai International Advanced Research Institute (Shenzhen Futian). Correspondence to: Qibin Hou <houqb@nankai.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.



(a) ZegCLIP (Zhou et al., 2023) (b) Multi-layer Features Fusion (c) Cascade-CLIP (Ours)

Figure 2. Three zero-shot segmentation approaches based on CLIP. (a) ZegCLIP relies on the **last-layer** visual features without considering information from intermediate layers. (b) Inspired by SegFormer (Xie et al., 2021), we fuse both **intermediate-** and **last-layers** features to enhance feature representation, yet this integration disrupts the correlation between text and visual features. (c) To alleviate this issue, our Cascade-CLIP **separats** the image encoder and **aligns** independent text-image decoders for deep features and middle features respectively, and finally **cascades** the segmentation results.

Table 1. Comparisons with different methods on COCO-Stuff 164K, and PASCAL VOC 2012 datasets[1]. Straightforward fusion of multi-layer features leads to performance degradation.

Methods	COCO-Stuff 164K			PASCAL VOC 2012		
	mIoU^S	mIoU^U	hIoU	mIoU^S	mIoU^U	hIoU
ZegCLIP (Baseline)	40.1	39.5	39.8	90.5	78.3	84.0
Multi-layer Fusion	37.7	39.0	38.4	91.2	75.0	82.3
Cascade-CLIP	**41.1**	**43.4**	**42.2**	**92.7**	**83.1**	**87.7**

methods exhibit weaknesses in segmenting object details, especially the boundaries of the semantic objects.

Building on insights from the closed-set segmentation methods (Zheng et al., 2021; Hou et al., 2020; Xie et al., 2021; Guo et al., 2022), a viable solution for capturing rich local details is to aggregate multi-level features from the encoder to improve the coarse segmentation results. For the CLIP model, we observe that the visual features extracted from the intermediate layers contain rich object details as illustrated in Fig. 1. Nevertheless, directly fusing the multi-level features produces unsatisfactory results. As shown in Tab. 1, a straightforward fusion of the middle-layers and the last-layer features (Fig. 2(b)) degrades the performance compared to the baseline model (Fig. 2(a)). The baseline's success lies in effectively leveraging the pre-trained correlations in CLIP between the last-layer visual features and the text embeddings. However, the fusion of multi-level features disrupts these original visual-language correlations due to the significant disparity between the middle-layer and last-layer features, weakening CLIP's zero-shot capability on unseen classes. Moreover, after feature fusion, the differences between features also disrupt the pre-trained visual representations, further increasing the difficulty of aligning visual features with the text embeddings during fine-tuning.

In this paper, we renovate the way of aligning visual and text embeddings and propose Cascade-CLIP, a multi-level framework that can better leverage the diverse visual features from CLIP and enhance the transferability to novel classes. Specifically, Cascade-CLIP splits the visual encoder into multiple stages, ensuring a little variation in the features within each stage. Each stage is equipped with an independent text-image decoder, employing distinct text embeddings to align the multi-level visual features better and build better vision-language correlation. In this way, we can integrate complementary multi-level semantic masks from the visual encoder to enhance the segmentation results as demonstrated in Fig. 1 (bottom row).

By exploiting multi-level features, for the first time, we demonstrate that our Cascade-CLIP can largely improve the image-to-pixel adaptability of CLIP in zero-shot semantic segmentation. In addition, Cascade-CLIP is also flexible and can be seamlessly utilized in existing state-of-the-art methods, such as ZegCLIP (Zhou et al., 2023) and SPTSEG (Xu et al., 2024), to lift their performance on three commonly used zero-shot segmentation benchmarks. In particular, thanks to the cascaded vision-language alignment, our method performs especially well for unseen classes, reflecting strong adaptability. The contributions can be summarized as follows:

- We unveil that visual features from the intermediate layers of CLIP contain rich local information about objects. However, simply fusing the multi-level visual features weakens CLIP's zero-shot capability.
- We propose Cascade-CLIP, a flexible cascaded vision-language embedding alignment framework that can effectively leverage the multi-level visual features from CLIP to improve the transferability for novel classes.
- Extensive experiments demonstrate the effectiveness of our Cascade-CLIP in zero-shot semantic segmentation on three widely-used benchmarks.

2. Related work

2.1. Pre-trained Vision Language Models

Large-scale vision-language models (Jia et al., 2021; Kim et al., 2021; Radford et al., 2021) pre-trained with web-scale

[1] mIoU^S and mIoU^U represent the mean Intersection over Union (%) of seen and unseen classes, respectively. hIoU denotes the harmonic mean IoU score among seen and unseen classes.

image-text pairs have made great progress in aligning image and text embeddings and achieved strong zero/few-shot generalization capabilities. For example, one of the most popular vision-language models, CLIP (Radford et al., 2021), is trained contrastively using 400 million image-text pairs. Due to its zero-shot recognition capabilities and its simplicity, CLIP has been widely adapted to various down-stream tasks, such as zero-shot visual recognition (Khattak et al., 2023), dense prediction (Rao et al., 2022), object detection (Gu et al., 2021), and visual referring expression (Wang et al., 2022). This work explores how to efficiently transfer CLIP's powerful generalization ability from images to pixel-level classification.

2.2. Zero-shot Semantic Segmentation

Zero-shot semantic segmentation performs a pixel-level classification that includes unseen categories during training. Previous works, such as SPNet (Xian et al., 2019), ZS3 (Bucher et al., 2019), CaGNet (Gu et al., 2020), SIGN (Cheng et al., 2021b), JoEm (Baek et al., 2021), and STRICT (Pastore et al., 2021), focus on learning a mapping between visual and semantic spaces to improve the generalization ability of the semantic mapping from the seen classes to the unseen ones. Recent approaches mostly employ large-scale visual-language models (*e.g.,* CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021)) with powerful zero-shot classification ability for zero-shot semantic segmentation. Some training-free methods, such as ReCo (Shin et al., 2022) and CaR (Sun et al., 2023), directly use CLIP to perform zero-shot semantic segmentation. Other methods, like MaskCLIP+ (Zhou et al., 2022a), apply CLIP to generate pseudo-annotations for unseen classes to train existing segmentation models, but the requirement for unseen class names constrains it. To relieve this limitation, some works, like ZegFormer (Ding et al., 2022), Zsseg (Xu et al., 2022b), FreeSeg (Qin et al., 2023), and DeOP (Han et al., 2023a), decouple the zero-shot semantic segmentation into the class-agnostic mask generation process and the mask category classification process using CLIP. While they retain the zero-shot capability of CLIP at the image level, the computational cost inevitably increases due to the introduction of proposal generators.

Instead of using heavy proposal generators, ZegCLIP (Zhou et al., 2023) introduces a lightweight decoder to match the text embeddings against the visual embeddings extracted from CLIP. Similarly, SPT-SEG (Xu et al., 2024) enhances the CLIP's semantic understanding ability by integrating spectral information. Although the above methods have successfully translated CLIP's image classification into pixel segmentation, there is still a large room for improvement. Unlike previous works, we take a new look at zero-shot semantic segmentation by investigating the role of the features from the intermediate layers in the visual encoder.

3. Method

The zero-shot semantic segmentation task (Bucher et al., 2019; Zhou et al., 2023) aims to segment both the seen classes $\mathcal{C}$ and unseen classes $\hat{\mathcal{C}}$ after training on a dataset with the seen part of available pixel annotations. Normally, $\mathcal{C} \cap \hat{\mathcal{C}} = \varnothing$ and the labels of $\hat{\mathcal{C}}$ are unavailable while training. The key problem is to retain the ability to identify unseen classes when training on the seen classes.

3.1. Revisiting ZegCLIP

Recent zero-shot semantic segmentation approaches (Zhou et al., 2023; Xu et al., 2024) are mostly based on the one-stage scheme due to its high efficiency and good performance. Here, we revisit the ZegCLIP work (Zhou et al., 2023) as our baseline.

As illustrated in Fig. 2(a), ZegCLIP (Zhou et al., 2023) first extracts CLIP's text embeddings of C classes as $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_C] \in \mathbb{R}^{C \times d}$ and CLIP's visual features of an image as [CLS] token $\mathbf{g} \in \mathbb{R}^{1 \times d}$ and patch tokens $\mathbf{H} \in \mathbb{R}^{N \times d}$, where d is the feature dimension of the CLIP model, and N is the number of patch tokens. C is the number of classes with $C = |\mathcal{C}|$ during training and $C = |\mathcal{C} \cup \hat{\mathcal{C}}|$ during inference. To avoid overfitting, a relationship descriptor, denoted as $\hat{\mathbf{T}} = \mathrm{concat}(\mathbf{T} \odot \mathbf{g}, \mathbf{T}) \in \mathbb{R}^{C \times 2d}$, where $\odot$ and concat are the Hadamard product and concatenation, is employed by (Zhou et al., 2023) instead of $\mathbf{T}$. Then, the semantic masks $\mathbf{M} \in \mathbb{R}^{C \times N}$ can be generated in the text-image decoder by measuring the similarity between the text embeddings $\hat{\mathbf{T}}$ and the visual features $\mathbf{H}$. The whole process can be represented as follows:

$$\mathbf{M} = \mathrm{Softmax}(\mathcal{D}(\phi_q(\hat{\mathbf{T}}), \phi_k(\mathbf{H}))), \tag{1}$$

where $\mathcal{D}(\cdot)$ denotes the text-image decoder, as illustrated in the right part of Fig. 3. ϕ_q and ϕ_k are two linear projections that align the feature dimension of $\hat{\mathbf{T}}$ and $\mathbf{H}$.

Since the visual features are only extracted from the last layer of the visual encoder, previous methods often cannot identify the boundaries of the semantic objects well. This is because the deep features carry high-level semantic global features as shown in Fig. 1 but less low-level local details compared to the intermediate layers, which we will pay close attention to in this paper.

3.2. Motivation

Multi-level features are commonly used in closed-set segmentation models (Zheng et al., 2021; Xie et al., 2021) to sharpen the object segmentation details. Our analysis in Sec. 1 also reveals that features from the middle layers of CLIP (Radford et al., 2021) can capture rich local object details. This motivates us to investigate how to effectively take advantage of these distinct features to enhance

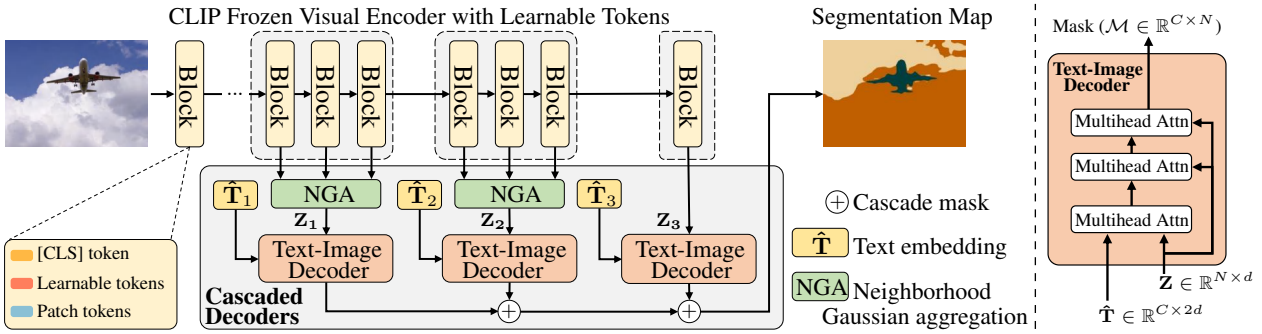
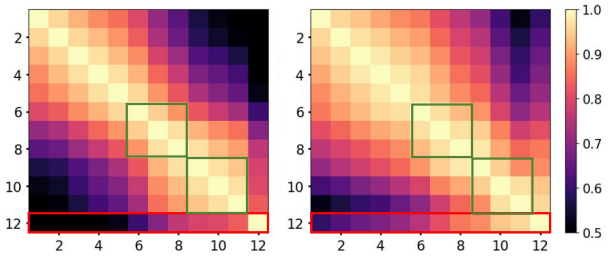


Figure 3. Architecture of our Cascade-CLIP. The CLIP visual encoder is divided into multiple stages. Then, we employ the NGA module to aggregate features of blocks within each stage and assign an independent text-image decoder for aggregated visual features and non-sharing text embeddings. In the text-image decoder (right part of the figure), the segmentation mask could be calculated by the scaled dot product attention via the Multihead Attention (Attn) layers, inspired by (Zhang et al., 2022). Finally, we combine the multi-level semantic masks produced by different cascaded decoders to enhance segmentation predictions. (Please refer to Sec. 3.3 for details.)



(a) Original CLIP (b) Cascade-CLIP (Ours)

Figure 4. Centered kernel alignment heatmap (Kornblith et al., 2019) between layers of (a) Original CLIP and (b) Cascade-CLIP (Ours). The last row (red box) shows the similarity between features from the last layer and other layers. The green box illustrates the similarity between adjacent layers.

CLIP's transferability to novel classes, which has been omitted by previous works. However, simply aggregating the multi-level visual features as done in Fig. 2(b) degrades the segmentation performance. To analyze the reasons for the performance degradation, we attempt to visualize the centered kernel alignment map (Kornblith et al., 2019) of the visual features of CLIP as shown in Fig. 4(a), which measures the similarity between different layers. We observe a substantial dissimilarity between the shallow and deep features, and the difference increases with network depth. This indicates that directly integrating the multi-level intermediate features into the last ones may break the alignment of vision-language embedding in pre-trained CLIP due to the substantial differences, thereby weakening the zero-shot capability of CLIP.

Given the above analysis, we intend to investigate how to effectively leverage the intermediate features with rich local details to improve zero-shot segmentation. To address this challenge, we propose two strategies, namely Cascaded Vision-Language Embedding Alignment and Neighborhood Gaussian Aggregation to better align the multi-level visual features with the text embeddings. These strategies aim to reduce the feature differences between different layers so that the mid-level visual features can be well aligned with the text embeddings and complement deep-level features, improving the ability of zero-shot segmentation.

3.3. Cascaded Alignment Framework

The overview of the proposed Cascade-CLIP framework is illustrated in Fig. 3. Basically, the visual encoder of CLIP is split into multiple stages to extract multi-level visual features, with a little variation for the features in each stage. Then, to better build vision-language correlations during fine-tuning, we assign each stage of the visual encoder an independent text-image decoder considering the feature differences between various stages. The decoder is similar to the one mentioned in Sec. 3.1. Finally, the refined results are generated by cascading complementary segmentation masks from various stages.

To be specific, let $\mathbf{H}_l$ denote the patch tokens of the l-th Transformer block. For ViT-B, the number of blocks should be 12. First, we separate the CLIP's visual encoder into S stages, each containing a group of Transformer blocks. In each stage, *e.g.,* s-th stage, to better leverage the multi-level features from different Transformer blocks, we introduce a Neighborhood Gaussian Aggregation (NGA) module to aggregate these features, resulting in aggregated features $\mathbf{Z}_s$. We will describe the NGA module in detail later. Then, for the output from the s-th stage $\mathbf{Z}_s$, we associate a corresponding text embedding $\hat{\mathbf{T}}_s$, acquired through linear projection from $\hat{\mathbf{T}}$. Subsequently, $\mathbf{Z}_s$ and $\hat{\mathbf{T}}_s$ are fed into an independent text-image decoder to generate semantic masks. Finally, we replace the single semantic masks in Eq. 1 by combining all semantic masks generated by multiple stages as follows:

$$\mathbf{M}_S = \text{Softmax}\left(\sum\nolimits_{s=1}^{S} \mathcal{D}_s(\hat{\mathbf{T}}_s, \mathbf{Z}_s)\right), \qquad (2)$$

where $\mathcal{D}_s(\cdot)$ denotes the s-th text-image decoder. Here, we use an element-wise summation operation, which can

be regarded as an ensemble of the outputs from multiple cascaded decoders.

As shown in Fig. 3, the vision-language alignment process can be applied multiple times to different blocks in a cascaded way. In practice, we do not append the text-image encoder to the shallow Transformer blocks because the shallow features contain little semantic information. Our experiment in Sec. 4.4 will show how to split the visual encoders to take advantage of the multi-level visual features.

Loss function with cascaded masks. Given the text-image decoder $\mathcal{D}_s(\cdot)$ from the s-th stage, let $\mathcal{M}_s = \mathcal{D}_s(\hat{\mathbf{T}}_s, \mathbf{Z}_s)$ be the predicted segmentation mask. $\mathcal{M} = \sum_{s=1}^{S} \mathcal{M}_s$ is the multi-level cascaded mask. The objective loss function $\mathcal{L}^{\text{pixel}}$ is defined as:

$$\mathcal{L}^{\text{pixel}} = \alpha \mathcal{L}^{\text{dice}}(\mathbf{Y}, \mathcal{M}) + \beta \mathcal{L}^{\text{focal}}(\mathbf{Y}, \mathcal{M}), \tag{3}$$

where $\mathcal{L}^{\text{dice}}$ and $\mathcal{L}^{\text{focal}}$ are the dice loss (Milletari et al., 2016) and focal loss (Lin et al., 2017) with $\mathrm{Sigmoid}$ as activation function, respectively. $\mathbf{Y}$ is the ground truth. $\{\alpha, \beta\}$ are two weights with the default values of $\{1, 100\}$, respectively.

To better align the intermediate visual features with the text embeddings, we employ visual prompt tuning (Zhou et al., 2022b; Ding et al., 2022) by introducing learnable tokens onto the visual features of each block in the frozen encoder. During visual prompt tuning, the cascaded alignment manner enables the gradients to be directly back-propagated to the middle layers of the visual encoder. This can promote the alignment of mid-layer features and text embeddings, substantially enhancing the similarity across different layers. We illustrate this in Fig. 4(b), which reflects a clear difference from Fig. 4(a).

Neighborhood Gaussian aggregation. To better utilize the potential of the features from each Transformer block, we propose the Neighborhood Gaussian Aggregation (NGA) module to fuse the multi-level features within each stage. Based on the analysis in Sec. 3.2 and the illustration in Fig. 4(b), we observe a gradual decline in feature similarity across layers with increasing distance. Thus, we propose to assign blocks with distinct Gaussian weights during feature fusion based on their relative neighborhood distances. Furthermore, these weights are trainable concerning the training data, facilitating the acquisition of adaptive weight information from different blocks within each encoder stage. Considering the s-th stage of encoder that consists of d Transformer blocks, the Gaussian weights $W_{s,l}$ and aggregated features $\mathbf{Z}_s$ can be computed as:

$$\begin{aligned} W_{s,l} &= \exp\left(-\frac{1}{2}\frac{(d-l+1)^2}{\sigma^2}\right), \quad l \in [1, d], \\ \mathbf{Z}_s &= \sum_{l=1}^{d} \mathbf{H}_l \cdot W_{s,l}, \end{aligned} \tag{4}$$

where the variance parameter σ of the Gaussian function is set to 1 by default. l corresponds to the index of the Transformer block. Increasing σ results in uniform weighting across Transformer blocks while decreasing σ leads to the dependence on single block features (as demonstrated in our ablation experiments in Sec. C of the Appendix). By setting variance parameter σ, the NGA module can assign a higher weight to nearby blocks and a lower weight to distant ones, facilitating more effective and flexible integration of features across different depth levels.

4. Experiments

4.1. Datasets and Evaluation Metrics

To evaluate the effectiveness of our proposed method, we perform extensive experiments on three widely used benchmark datasets, including COCO-Stuff (Caesar et al., 2018), Pascal-VOC (Everingham et al., 2015), and Pascal-Context (Mottaghi et al., 2014). The split of seen and unseen categories follows the common settings of the previous works (Zhou et al., 2023), and the mean IoU and harmonic mean IoU of both seen and unseen categories are reported. Further details on dataset statistics, data splitting, and evaluation metrics can be found in Sec. B of the Appendix.

4.2. Implementation Details

We implement the proposed method on the open-source toolbox MMSegmentation (Contributors, 2020) and conduct all experiments using a machine with 4 NVIDIA RTX 3090 GPUs. VIT-B/16 (Dosovitskiy et al., 2020), which contains 12 Transformer blocks, is adopted as the image encoder of CLIP (Radford et al., 2021). The batch size on each GPU is set to 4, and the input image resolution is 512×512. The optimizer is AdamW (Loshchilov & Hutter, 2019) with the default training schedule in the MMSeg toolbox. For a fair comparison, we use the same number of training iterations on each dataset as ZegCLIP (Zhou et al., 2023).

4.3. Comparisons with the State-of-the-art Methods

To demonstrate the effectiveness of our Cascade-CLIP, the evaluation results are compared with previous state-of-the-art methods, including two-encoder approaches (*e.g.,* ZegFormer (Ding et al., 2022), Zsseg (Xu et al., 2022b) and DeOP (Han et al., 2023a)) and one-encoder approaches (*e.g.,* ZegCLIP (Zhou et al., 2023)).

Comparisons in the inductive setting. As shown in Tab. 2, Cascade-CLIP remarkably improves the performance in the inductive setting, where features and annotations for unseen classes are not provided. It is worth noting that while boosting the results of the seen classes, our method improves the performances of the unseen classes. For example,

Table 2. Comparison with the ***inductive*** and ***transductive*** state-of-the-art zero-shot segmentation methods on COCO-Stuff 164K, and PASCAL VOC 2012 datasets. R denotes ResNet (He et al., 2016). ST represents re-train the model with the generating pseudo-labels on unseen classes. Our Cascade-CLIP integrates features extracted from layers 6 to 12 of the image encoder using a three-stage cascade decoder: {6-8}, {9-11}, {12}.

Methods	Backbone	Segmentor[2]	COCO-Stuff 164K (171)			PASCAL VOC 2012 (20)		
			mIoU^S↑	mIoU^U↑	hIoU↑	mIoU^S↑	mIoU^U↑	hIoU↑
Inductive: training images do not contain any unseen objects.								
ZegFormer (Ding et al., 2022)	R101&CLIP-B	MaskFormer	36.6	33.2	34.8	86.4	63.6	73.3
Zsseg (Xu et al., 2022b)	R101&CLIP-B	MaskFormer	39.3	36.3	37.8	83.5	72.5	77.5
DeOP (Han et al., 2023a)	R101&CLIP-B	MaskFormer	38.0	38.4	38.2	88.2	74.6	80.8
Zsseg+MAFT (Jiao et al., 2023)	R101&CLIP-B	MaskFormer	40.6	40.1	40.3	88.4	66.2	75.7
SPNet-C (Xian et al., 2019)	R101	W2V&FT	35.2	8.7	14.0	78.0	15.6	26.1
ZS3Net (Bucher et al., 2019)	R101	W2V	34.7	9.5	15.0	77.3	17.7	28.7
CaGNet (Gu et al., 2020)	R101	W2V&FT	33.5	12.2	18.2	78.4	26.6	39.7
SIGN (Cheng et al., 2021b)	R101	W2V&FT	32.3	15.5	20.9	75.4	28.9	41.7
JoEm (Baek et al., 2021)	R101	W2V	-	-	-	77.7	32.5	45.9
ZegCLIP (Zhou et al., 2023)	CLIP-B	SegViT	40.2	41.4	40.8	91.9	77.8	84.3
Cascade-CLIP (Ours)	CLIP-B	SegViT	**41.1**	**43.4**	**42.2**	**92.7**	**83.1**	**87.7**
Transductive: training images employ the names of unseen classes.								
Zsseg+ST (Xu et al., 2022b)	R101&CLIP-B	MaskFormer	39.6	43.6	41.5	79.2	78.1	79.3
FreeSeg (Qin et al., 2023)	R101&CLIP-B	Mask2Former	42.4	42.2	42.3	91.9	78.6	84.7
FreeSeg+MAFT (Jiao et al., 2023)	R101&CLIP-B	Mask2Former	**44.1**	55.2	49.0	90.0	86.3	88.1
SPNet-C+ST (Xian et al., 2019)	R101	W2V&FT	34.6	26.9	30.3	77.8	25.8	38.8
ZS5Net (Bucher et al., 2019)	R101	W2V	34.9	10.6	16.2	78.0	21.2	33.3
CaGNet+ST (Gu et al., 2020)	R101	W2V&FT	35.6	13.4	19.5	78.6	30.3	43.7
MaskCLIP+ (Zhou et al., 2022a)	R101	DeepLabv2	38.1	54.7	45.0	88.8	86.1	87.4
MVP-SEG+ (Guo et al., 2023)	R101	DeepLabv2	38.3	55.8	39.9	44.9	67.5	54.0
ZegCLIP+ST (Zhou et al., 2023)	CLIP-B	SegViT	40.7	59.9	48.5	92.3	89.9	91.1
TagCLIP+ST (Li et al., 2023)	CLIP-B	SegViT	40.4	60.0	48.3	**94.3**	92.7	**93.5**
Cascade-CLIP+ST (Ours)	CLIP-B	SegViT	41.7	**62.5**	**50.0**	93.3	**93.4**	93.4

Table 3. Comparison with the state-of-the-art zero-shot segmentation methods on PASCAL Context dataset.

Methods	PASCAL Context (59)		
	mIoU^S↑	mIoU^U↑	hIoU↑
Inductive			
SPNet-C (Xian et al., 2019)	27.1	9.8	14.4
ZS3Net (Bucher et al., 2019)	20.8	12.7	15.8
CSRL (Li et al., 2020)	29.4	14.6	19.5
JoEm (Baek et al., 2021)	33.0	14.9	20.5
ZegCLIP (Zhou et al., 2023)	53.8	45.5	49.3
Cascade-CLIP (Ours)	**55.9**	**47.2**	**51.2**
Transductive			
ZS5Net (Bucher et al., 2019)	27.0	20.7	23.4
ZegCLIP+ST (Zhou et al., 2023)	54.5	41.4	47.1
Cascade-CLIP+ST (Ours)	**56.4**	**55.0**	**55.7**

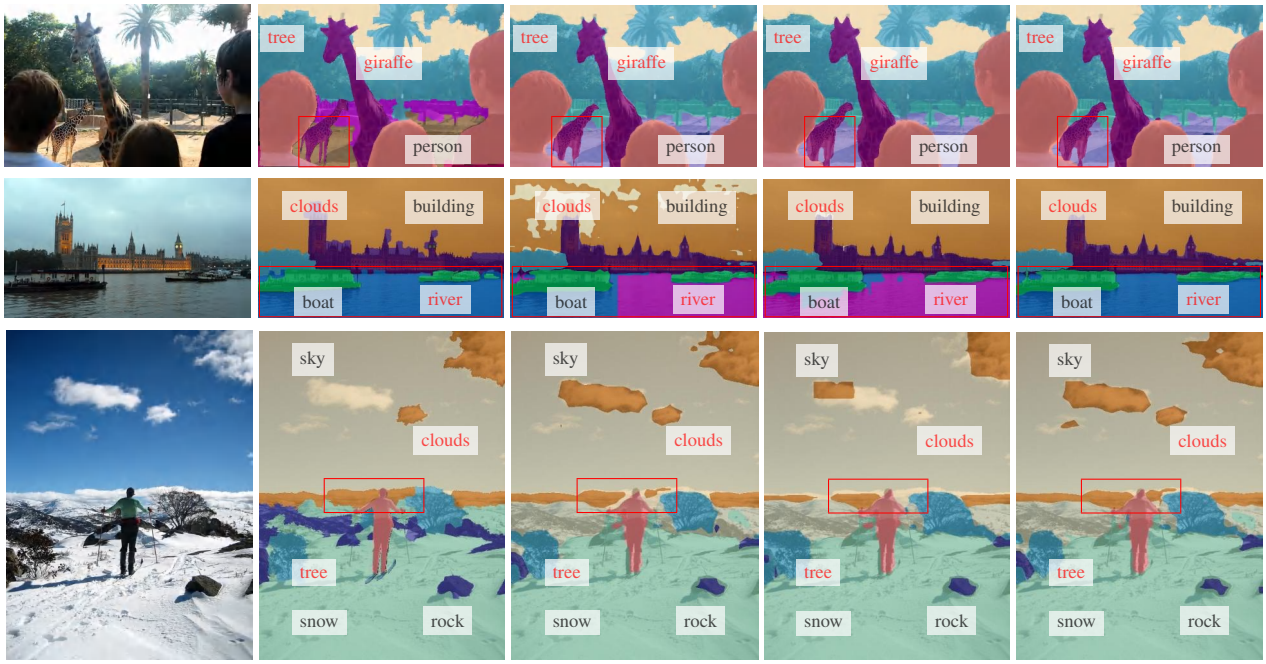
Cascade-CLIP promotes the state-of-the-art performance by 2.0% on COCO and 5.3% on Pascal VOC in terms of mIoU for unseen classes, demonstrating its robust generalization capabilities of zero-shot segmentation.

[2]The description of segmentors is acquired from (Zhu & Chen, 2023). MaskFormer and MasksFormer are proposed by (Cheng et al., 2021a) and (Cheng et al., 2022b); DeepLabv2 is proposed by (Chen et al., 2018); W2V is proposed by (Mikolov et al., 2013); SegViT is proposed by (Zhang et al., 2022).

Comparisons in the transductive setting. We further evaluate the transferability of Cascade-CLIP in the transductive setting, where models are retrained by generating pseudo labels for unseen pixels and utilizing ground truth labels for seen pixels. Tab. 2 demonstrates that our model significantly improves the performance for unseen classes while consistently maintaining excellent performance across seen classes after transductive self-training.

To further validate the effectiveness of our Cascade-CLIP, we conduct comparisons with other methods on the PASCAL Context dataset. As shown in Tab. 3, our Cascade-CLIP consistently outperforms other methods, particularly regarding mIoU for unseen classes. The results above clearly demonstrate the effectiveness of our proposed methods. For additional experimental results on the effectiveness and generality of our method, please refer to Sec. 4.5.

Qualitative results. Fig. 5 shows the segmentation results of the baseline and our proposed Cascade-CLIP on seen and unseen classes. Cascade-CLIP shows impressive segmentation ability for both seen and unseen classes and can clearly distinguish similar unseen classes. For example, our method can better distinguish the 'giraffe' regions from the 'tree' regions (Fig. 5(1)), the 'boat' regions from the 'river'



(a) Origin Image (b) Ground truth (c) ZegCLIP (d) SPT-SEG (e) Cascade-CLIP

Figure 5. Qualitative transductive results on COCO-Stuff 164K. The black and red tags represent seen and unseen classes, respectively.

Table 4. Ablation on components of Cascade-CLIP.

Cascaded decoders	NGA	mIoU^S↑	mIoU^U↑	hIoU↑
✗	✗	40.1	39.5	39.8
✓	✗	40.8	42.6	41.7
✓	✓	**41.1**	**43.4**	**42.2**

regions (Fig. 5(2)), and the 'clouds' regions from the 'sky' regions (Fig. 5(3)). More qualitative results are available in Sec. D of the Appendix.

4.4. Ablation Study

Component-wise ablations. To understand the effect of each component in our Cascade-CLIP, including the cascaded decoders and the NGA modules, we initiate with the baseline ZegCLIP, which employs the visual features of CLIP's last layer, and then gradually incorporate each proposed module. As shown in Tab. 4, employing cascaded decoders can capture distinct and complementary information from different blocks in the encoder, improving 3.1% mIoU scores on unseen classes (the 2nd result). On this basis, the NGA module is introduced to aggregate rich local information of objects within each split encoder, further enhancing the mIoU scores on unseen classes (the 3rd result).

Effect of the proposed block splitting manner. The cascaded decoder architecture is pivotal in our Cascade-CLIP due to its ability to preserve visual-language associations. Our analysis in Tab. 5 indicates that separating the last block into an independent stage (the 3rd result) is more effective than other combinations of partitioning strategies (the 1st result and the 2nd result). This is because the last layer feature of CLIP's image encoder possesses the strongest association with text embedding, and matching it to a separate decoder reduces the destruction of this correlation.

Table 5. Effect of different block splitting manners. The elements in {} means the block numbers that are used to fuse features in the encoder. When the number of blocks exceeds 1, an NGA is used.

Block splitting manner	#Decoders	mIoU^S↑	mIoU^U↑	hIoU↑
{4-6}, {7-9}, {10-12}	3	40.4	42.7	41.5
{6-8}, {9-10}, {11-12}	3	40.7	42.4	41.5
{6-8}, {9-11}, {12}	3	**41.1**	**43.4**	**42.2**

Table 6. Numbers of cascaded decoders and corresponding blocks in the visual encoder of Cascade-CLIP.

Block splitting manner	Number	mIoU^S↑	mIoU^U↑	hIoU↑
Number of cascaded decoders				
{12} (Baseline)	1	40.1	39.5	39.8
{9-11}, {12}	2	40.2	40.2	40.2
{6-8}, {9-11}, {12}	3	**41.1**	**43.4**	**42.2**
{3-5}, {6-8}, {9-11}, {12}	4	40.8	42.5	41.7
Number of blocks				
{8-9}, {10-11}, {12}	2	40.7	41.3	41.0
{6-8}, {9-11}, {12}	3	**41.1**	**43.4**	**42.2**
{4-7}, {8-11}, {12}	4	41.1	41.5	41.3

Number of cascaded decoders and number of blocks in each stage. To show the importance of information fusion across different layers, we present the performance of

Table 7. Cascade-CLIP vs. other methods with multi-decoders. Utilizing the last-layer features or directly fusing multi-layer features with the same blocks as our Cascade-CLIP and aligning them with multiple decoders results in a decline in performance.

Description	Param. (M)	mIoU^S↑	mIoU^U↑	hIoU↑
Last-layer	40.5	40.3	40.6	40.5
Multi-layer fusion	40.5	39.9	35.7	37.7
Cascade-CLIP	40.5	**41.1**	**43.4**	**42.2**

Table 8. Comparison of different aggregation methods. † denotes that the weights are trainable.

Description	mIoU^S↑	mIoU^U↑	hIoU↑
Concat	39.9	41.0	40.5
Self-attention	37.7	39.0	38.4
Sum	39.3	42.0	40.6
NGA	40.9	43.0	41.9
Sum†	39.9	41.2	40.5
NGA†	**41.1**	**43.4**	**42.2**

Table 9. Effect of independent/shared text embedding.

Description	mIoU^S↑	mIoU^U↑	hIoU↑
Shared	40.5	42.4	41.4
Independent	**41.1**	**43.4**	**42.2**

Cascade-CLIP with different numbers of cascaded decoders and different blocks in each corresponding encoder stage in Tab. 6. We can see that increasing the number of cascaded decoders from 1 to 3 gradually improves the segmentation performance. This indicates the complementarity of features from various layers compared to previous works only using the features from the last layer. Our default value of Transformer blocks per stage is 3. Reducing the number of blocks to 2 causes a degradation in performance due to the neglect of mid-layer features. The best performance is achieved by cascading three decoders (including an extra decoder for the last block). Note that we do not use the beginning blocks as they encode features with little semantics.

To demonstrate the effectiveness of our designs in leveraging CLIP's multi-level features, we also present cosine similarity maps of features and qualitative segmentation results. At the top of Fig. 6, we show the patch similarity of unseen classes not included in the training process. We observe that the intermediate layers of our method contain detailed information on local objects, including boundaries. Moreover, as illustrated at the bottom of Fig. 6, by leveraging these distinctive features, our Cascade-CLIP improves segmentation performance for both seen and unseen classes compared to using only last-block features.

NGA v.s. other aggregation methods. In each split encoder stage, the differences between features across various layers will disrupt the feature space after aggregating various layers. To overcome this issue, we propose Neighborhood Gaussian Aggregation (NGA) to reduce disruptions in the original feature space by considering the distance between blocks. As shown in Tab. 8, our NGA outperforms common aggregation strategies (*e.g.,* Sum, Connect and Self-attention). With learnable weights, our NGA further boosts performance. This indicates that our NGA, which assigns smaller weights to distant features when fusing multi-level features, is advantageous in improving zero-shot segmentation compared to other feature aggregation methods.

Table 10. Extending Cascade-CLIP to existing methods to improve zero-shot segmentation results.

Methods	COCO-Stuff 164K		VOC 2012	
	mIoU^S↑	mIoU^U↑	mIoU^S↑	mIoU^U↑
Inductive				
Frozen CLIP	32.3	32.5	85.9	59.5
Frozen CLIP+Ours	**36.3**	**35.3**	**89.0**	**69.7**
ZegCLIP	40.2	41.4	91.9	77.8
ZegCLIP+Ours	**41.1**	**43.4**	**92.7**	**83.1**
SPT-SEG	38.0	40.7	92.0	85.0
SPT-SEG+Ours	**40.2**	**43.6**	**92.1**	**86.1**
Transductive				
ZegCLIP+ST	40.7	59.9	92.3	89.9
ZegCLIP+Ours+ST	**41.7**	**62.5**	**93.3**	**93.4**
SPT-SEG	40.4	57.5	93.6	92.2
SPT-SEG+Ours+ST	**41.7**	**62.1**	**93.8**	**94.4**
Fully Supervised				
ZegCLIP	40.7	63.2	92.4	90.9
ZegCLIP+Ours	**41.5**	**64.0**	**93.7**	**94.6**

Cascade-CLIP vs. other methods with multi-decoder. To demonstrate the efficacy of integrating diverse semantic masks generated from distinct cascaded decoders, rather than introducing extra parameters that could impact performance, we construct a multi-decoder model based on the last-layer features or the fusion of multi-layer features. As shown in Tab. 7, Cascade-CLIP outperforms the last-layer and multi-layer approaches with equivalent parameter amounts. This indicates that relying only on the last-layer features fails to yield complementary and enhanced segmentation results. Moreover, the direct fusion of features leads to a decrease in zero-shot capability, which is not improved even with the use of multiple decoders.

Effect of independent/shared text embedding. Since the features from different split encoder stages exhibit significant discrepancies, it is essential to align distinct text embeddings with the features of each stage. This is validated by the results presented in Tab. 9, where our Cascade-CLIP with independent text embedding achieves higher mIoU scores than those with shared text embedding.

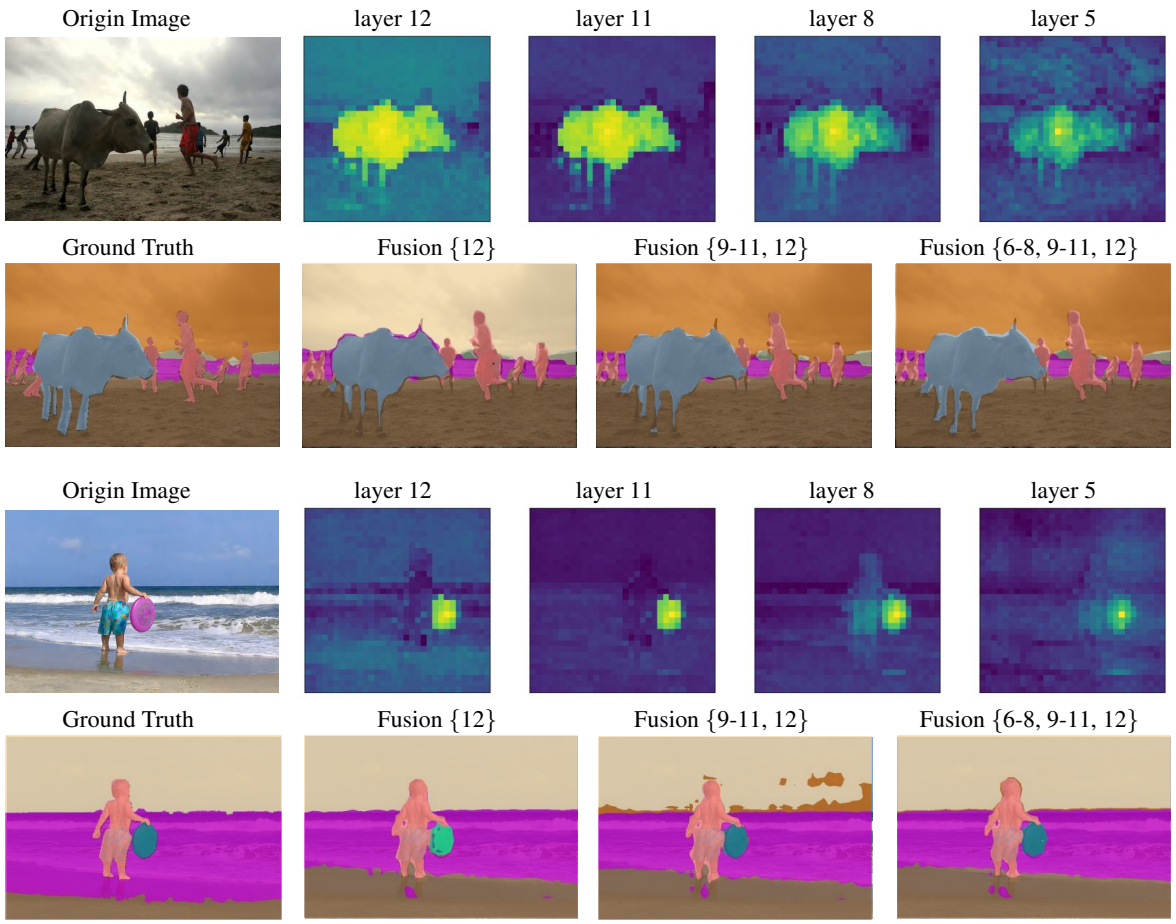


Figure 6. Visualizations of cosine similarity maps about features and qualitative segmentation results. We visualize feature correspondence through cosine similarity calculations on visual patches of **unseen** classes from both deep and shallow layers in our Cascade-CLIP. The inductive segmentation results are generated by cascading different decoders.

4.5. Extending Cascade-CLIP to Other Methods

Our approach is a generalized framework for improving the zero-shot segmentation capabilities. Specifically, we can seamlessly integrate Cascade-CLIP into existing popular zero-shot semantic segmentation methods, *e.g.,* Frozen CLIP (Radford et al., 2021), ZegCLIP (Zhou et al., 2023) and SPT-SEG (Xu et al., 2024). As shown in Tab. 10, our method can significantly enhance these methods' results, proving the proposed approach's generalization ability.

5. Conclusions

This paper focuses on leveraging the intermediate features from CLIP with rich local details but significant differences from deep features to enhance zero-shot semantic segmentation. By introducing the cascaded mask mechanism, we present the Cascade-CLIP framework, which aims to effectively align multi-level visual features with the text embeddings in a cascaded way, thereby enhancing CLIP's adaptability from image to pixel level. Experiments demonstrate the effectiveness of the proposed method.

Impact Statement

Our work explores how to leverage multi-level features from the pre-trained CLIP to enhance the model's zero-shot capability, which has been proven effective in downstream tasks. Since our method is not designed for a specific application, it does not directly involve societal issues.

Acknowledgments

This research was supported by NSFC (NO. 62225604, No. 62276145), the Fundamental Research Funds for the Central Universities (Nankai University, 070-63223049), CAST through Young Elite Scientist Sponsorship Program (No. YESS20210377). Computations were supported by the Supercomputing Center of Nankai University (NKSC).

References

Baek, D., Oh, Y., and Ham, B. Exploiting a joint embedding space for generalized zero-shot semantic segmentation. In *ICCV*, pp. 9536–9545, 2021.

Bucher, M., Vu, T., Cord, M., and Pérez, P. Zero-shot semantic segmentation. In *NeurIPS*, pp. 466–477, 2019.

Caesar, H., Uijlings, J., and Ferrari, V. Coco-stuff: Thing and stuff classes in context. In *CVPR*, pp. 1209–1218, 2018.

Cha, J., Mun, J., and Roh, B. Learning to generate text-grounded mask for open-world semantic segmentation from only image-text pairs. In *CVPR*, pp. 11165–11174, 2023.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pp. 801–818, 2018.

Chen, X., Yuan, Y., Zeng, G., and Wang, J. Semi-supervised semantic segmentation with cross pseudo supervision. In *CVPR*, pp. 2613–2622, 2021.

Cheng, B., Schwing, A., and Kirillov, A. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, pp. 17864–17875, 2021a.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation. In *CVPR*, pp. 1290–1299, 2022a.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation. In *CVPR*, pp. 1290–1299, 2022b.

Cheng, J., Nandi, S., Natarajan, P., and Abd-Almageed, W. Sign: Spatial-information incorporated generative network for generalized zero-shot semantic segmentation. In *ICCV*, pp. 9556–9566, 2021b.

Contributors, M. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.

Ding, J., Xue, N., Xia, G.-S., and Dai, D. Decoupling zero-shot semantic segmentation. In *CVPR*, pp. 11583–11592, 2022.

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. *arXiv preprint arXiv:2010.11929*, 2020.

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. The pascal visual object classes challenge: A retrospective. *IJCV*, 111: 98–136, 2015.

Ghiasi, G., Gu, X., Cui, Y., and Lin, T.-Y. Scaling open-vocabulary image segmentation with image-level labels. In *ECCV*, pp. 540–557, 2022.

Gu, X., Lin, T.-Y., Kuo, W., and Cui, Y. Open-vocabulary object detection via vision and language knowledge distillation. In *ICLR*, 2021.

Gu, Z., Zhou, S., Niu, L., Zhao, Z., and Zhang, L. Context-aware feature generation for zero-shot semantic segmentation. In *ACM Multimedia*, pp. 1921–1929, 2020.

Guo, J., Wang, Q., Gao, Y., Jiang, X., Tang, X., Hu, Y., and Zhang, B. Mvp-seg: Multi-view prompt learning for open-vocabulary semantic segmentation. *arXiv preprint arXiv:2304.06957*, 2023.

Guo, M.-H., Lu, C.-Z., Hou, Q., Liu, Z., Cheng, M.-M., and Hu, S.-M. Segnext: Rethinking convolutional attention design for semantic segmentation. In *NeurIPS*, pp. 1140–1156, 2022.

Han, C., Zhong, Y., Li, D., Han, K., and Ma, L. Open-vocabulary semantic segmentation with decoupled one-pass network. In *ICCV*, pp. 1086–1096, 2023a.

Han, K., Liu, Y., Liew, J. H., Ding, H., Liu, J., Wang, Y., Tang, Y., Yang, Y., Feng, J., Zhao, Y., and Wei, Y. Global knowledge calibration for fast open-vocabulary segmentation. In *ICCV*, pp. 797–807, 2023b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Hou, Q., Zhang, L., Cheng, M.-M., and Feng, J. Strip pooling: Rethinking spatial pooling for scene parsing. In *CVPR*, pp. 4003–4012, 2020.

Huang, Z., Wei, Y., Wang, X., Liu, W., Huang, T. S., and Shi, H. Alignseg: Feature-aligned segmentation networks. *IEEE TPAMI*, 44(1):550–557, 2021.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, pp. 4904–4916, 2021.

Jiao, S., Wei, Y., Wang, Y., Zhao, Y., and Shi, H. Learning mask-aware clip representations for zero-shot segmentation. In *NeurIPS*, pp. 35631–35653, 2023.

Khattak, M. U., Wasim, S. T., Naseer, M., Khan, S., Yang, M.-H., and Khan, F. S. Self-regulating prompts: Foundational model adaptation without forgetting. In *ICCV*, pp. 15190–15200, 2023.

Kim, W., Son, B., and Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*, pp. 5583–5594, 2021.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. In *ICML*, pp. 3519–3529, 2019.

Li, J., Chen, P., Qian, S., and Jia, J. Tagclip: Improving discrimination ability of open-vocabulary semantic segmentation. *arXiv preprint arXiv:2304.07547*, 2023.

Li, P., Wei, Y., and Yang, Y. Consistent structural relation learning for zero-shot segmentation. In *NeurIPS*, pp. 10317–10327, 2020.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *ICCV*, pp. 2980–2988, 2017.

Liu, Q., Wen, Y., Han, J., Xu, C., Xu, H., and Liang, X. Open-world semantic segmentation via contrasting and clustering vision-language embedding. In *ECCV*, pp. 275–292, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *ICLR*, 2019.

Luo, H., Bao, J., Wu, Y., He, X., and Li, T. Segclip: Patch aggregation with learnable centers for open-vocabulary semantic segmentation. In *ICML*, pp. 23033–23044, 2023.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013.

Milletari, F., Navab, N., and Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, pp. 565–571, 2016.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, pp. 891–898, 2014.

Mukhoti, J., Lin, T.-Y., Poursaeed, O., Wang, R., Shah, A., Torr, P. H., and Lim, S.-N. Open vocabulary semantic segmentation with patch aligned contrastive learning. In *CVPR*, pp. 19413–19423, 2023.

Pastore, G., Cermelli, F., Xian, Y., Mancini, M., Akata, Z., and Caputo, B. A closer look at self-training for zero-label semantic segmentation. In *CVPR*, pp. 2693–2702, 2021.

Qin, J., Wu, J., Yan, P., Li, M., Yuxi, R., Xiao, X., Wang, Y., Wang, R., Wen, S., Pan, X., and Wang, X. Freeseg: Unified, universal and open-vocabulary image segmentation. In *CVPR*, pp. 19446–19455, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *ICML*, pp. 8748–8763, 2021.

Rao, Y., Zhao, W., Chen, G., Tang, Y., Zhu, Z., Huang, G., Zhou, J., and Lu, J. Denseclip: Language-guided dense prediction with context-aware prompting. In *CVPR*, pp. 18082–18091, 2022.

Shin, G., Xie, W., and Albanie, S. Reco: Retrieve and co-segment for zero-shot transfer. In *NeurIPS*, pp. 33754–33767, 2022.

Sun, S., Li, R., Torr, P., Gu, X., and Li, S. Clip as rnn: Segment countless visual concepts without training endeavor. *arXiv preprint arXiv:2312.07661*, 2023.

Wang, Z., Lu, Y., Li, Q., Tao, X., Guo, Y., Gong, M., and Liu, T. Cris: Clip-driven referring image segmentation. In *CVPR*, pp. 11686–11695, 2022.

Xian, Y., Choudhury, S., He, Y., Schiele, B., and Akata, Z. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, pp. 8256–8265, 2019.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, pp. 12077–12090, 2021.

Xu, J., De Mello, S., Liu, S., Byeon, W., Breuel, T., Kautz, J., and Wang, X. Groupvit: Semantic segmentation emerges from text supervision. In *CVPR*, pp. 18134–18144, 2022a.

Xu, J., Hou, J., Zhang, Y., Feng, R., Wang, Y., Qiao, Y., and Xie, W. Learning open-vocabulary semantic segmentation models from natural language supervision. In *CVPR*, pp. 2935–2944, 2023.

Xu, M., Zhang, Z., Wei, F., Lin, Y., Cao, Y., Hu, H., and Bai, X. A simple baseline for zero-shot semantic segmentation with pre-trained vision-language model. In *ECCV*, pp. 736–753, 2022b.

Xu, W., Xu, R., Wang, C., Xu, S., Guo, L., Zhang, M., and Zhang, X. Spectral prompt tuning: Unveiling unseen classes for zero-shot semantic segmentation. In *AAAI*, pp. 6369–6377, 2024.

Zeng, Y., Zhang, X., and Li, H. Multi-grained vision language pre-training: Aligning texts with visual concepts. In *ICML*, pp. 25994–26009, 2022.

Zhang, B., Tian, Z., Tang, Q., Chu, X., Wei, X., Shen, C., et al. Segvit: Semantic segmentation with plain vision transformers. In *NeurIPS*, pp. 4971–4982, 2022.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. Pyramid scene parsing network. In *CVPR*, 2017.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, pp. 6881–6890, 2021.

Zhou, C., Loy, C. C., and Dai, B. Extract free dense labels from clip. In *ECCV*, pp. 696–712, 2022a.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Learning to prompt for vision-language models. *IJCV*, 130(9):2337–2348, 2022b.

Zhou, Z., Lei, Y., Zhang, B., Liu, L., and Liu, Y. Zegclip: Towards adapting clip for zero-shot semantic segmentation. In *CVPR*, pp. 11175–11185, 2023.

Zhu, C. and Chen, L. A survey on open-vocabulary detection and segmentation: Past, present, and future. *arXiv preprint arXiv:2307.09220*, 2023.

A. Implementation Details

Text templates for prompts. Following the training details of CLIP, we utilize text templates to generate embeddings for class descriptions using the CLIP text encoder. Specifically, we employ a single template, "A photo of a { }," for PASCAL VOC 2012 (VOC). For large-scale datasets such as COCO-Stuff 164K (COCO) and PASCAL Context (Context), we employ 15 augmented templates for improved representation. The details of the 15 augmented templates are: 'A photo of a { }.', 'A photo of a small { }.', 'A photo of a medium { }.', 'A photo of a large { }.', 'This is a photo of a { }.', 'This is a photo of a small { }.', 'This is a photo of a medium { }.', 'This is a photo of a large { }.', 'A { } in the scene.', 'A photo of a { } in the scene.', 'There is a { } in the scene.', 'There is the { } in the scene.', 'This is a { } in the scene.', 'This is the { } in the scene.' and 'This is one { } in the scene.'.

B. Datasets and Evaluation Metrics

Datasets. COCO-Stuff is an extensive semantic segmentation dataset comprising 171 categories, encompassing 80 things classes and 91 stuff classes. It contains 117k training images and 5k validation images and it is divided into 156 seen classes and 15 unseen classes. In comparison, **PASCAL VOC** consists of 11,185 training images and 1,449 validation images across 20 classes. We exclude the 'background' category, utilizing 15 classes as the seen part and 5 classes as the unseen part. Additionally, **PASCAL Context** provides supplementary annotations for PASCAL VOC 2010, consisting of 4,998 training images and 5,005 validation images. For testing, we classify the dataset into 49 seen classes (excluding 'background') and use the remaining 10 classes as unseen.

Data split. We adopt the identical unseen class setup introduced in the previous method (Zhou et al., 2023) to ensure a fair comparison. The specific name of unseen classes for COCO-Stuff (COCO), PASCAL VOC 2012 (VOC), and PASCAL Context (Context) datasets in Tab. 11.

Table 11. Details of unseen class names.

Dataset	Name
COCO	$cow, giraffe, suitcase, frisbee, skateboard, carrot, scissors, cardboard, clouds, grass, playingfield, river, road, tree, wallconcrete$
VOC	$pottedplant, sheep, sofa, train, tvmonitor$
Context	$cow, motorbike, sofa, cat, boat, fence\ bird, tvmonitor, keyboard, aeroplane$

Evaluation metrics. Following the previous work (Zhou et al., 2023; Xu et al., 2024), we report the mean IoU (mIoU) for seen and unseen classes denoted as mIoU^S and mIoU^U respectively. In addition, we compute the harmonic mean IoU (hIoU) among seen and unseen classes, which is calculated as

$$\text{hIoU} = \frac{2(\text{mIoU}^S + \text{mIoU}^U)}{\text{mIoU}^S + \text{mIoU}^U}. \tag{5}$$

The mIoU metric is also adopted for the evaluation of cross-dataset.

Table 12. Efficiency comparison with different methods. The input image is set to 512×512. All models are evaluated on a single 3090 GPU. Params. represents the number of parameters of the model. Note that our Cascade-CLIP requires less learnable tokens.

Methods	Param. (M)		GFLOPs↓	FPS↑
	Total↓	Trainable↓		
ZegFormer (Ding et al., 2022)	210.0	60.3	1875.1	7.4
DeOP (Han et al., 2023a)	218.4	-	670.0	5.8
ZegCLIP (Zhou et al., 2023)	164.9	14.6	123.9	20.9
Cascade-CLIP (Ours)	193.1	40.5	123.8	18.4

C. Experiment

Efficiency analysis. Besides performance comparison, we also compare our approach with the typical two-encoder methods (*e.g.,* ZegFormer (Ding et al., 2022) and DeOP (Han et al., 2023a)) and one-encoder methods (*e.g.,* ZegCLIP (Zhou et al., 2023)). We test these methods under the same environment to ensure a fair comparison, including hardware and image resolution. Tab. 12 shows that our approach introduces the trainable parameters compared to ZegCLIP, notably less than ZegFormer's. Furthermore, the total parameter of the model increase is marginal. Moreover, our Cascade-CLIP does not increase the computational effort since Cascade-CLIP requires fewer learnable tokens than ZegCLIP (refer to subsequent experiments for details). Because of this, our Cascade-CLIP can still maintain a high FPS speed during inference.

Generalization ability to other datasets. To further explore the generalization ability of our proposed Cascade-CLIP, we perform additional experiments detailed in Tab. 13. In this setting, we utilize the pre-trained model from the source dataset (*i.e.,* COCO) through supervised learning on seen classes, assessing segmentation performance on both seen and unseen classes in the target datasets (*i.e.,* VOC and Context). We also compare the previous state-of-the-art zero-shot segmentation methods including the two-encoder approaches (*e.g.,* ZegFormer (Ding et al., 2022), Zsseg (Xu et al., 2022b) and DeOP (Han et al., 2023a)) and one-encoder approaches (*e.g.,* GKC (Han et al., 2023b) and TCL (Cha et al., 2023)). Our approach exhibits superior cross-domain generalization compared to the previous methods, particularly outperforming the recent ZegCLIP, which is also built on the CLIP model.

Table 13. Generalization ability to other datasets. The models are trained on the COCO-Stuff dataset.

Methods	Backbone	VOC↑	Context↑
Fully Supervised			
ZegFormer (Ding et al., 2022)	R101&CLIP-B	89.5	45.5
Zsseg (Xu et al., 2022b)	R101&CLIP-B	-	47.7
DeOP (Han et al., 2023a)	R101&CLIP-B	91.7	48.8
GroupViT (Xu et al., 2022a)	ViT-S	52.3	22.4
LSeg+ (Ghiasi et al., 2022)	R101	59.0	36.0
ViL-Seg (Liu et al., 2022)	ViT-B	33.6	15.9
SegCLIP (Luo et al., 2023)	CLIP-B	52.6	24.7
OpenSeg (Ghiasi et al., 2022)	R101	60.0	36.9
OVSegmentor (Xu et al., 2023)	ViT-B	53.8	20.4
PACL (Mukhoti et al., 2023)	ViT-B	72.3	50.1
TCL (Cha et al., 2023)	CLIP-B	83.2	33.9
GKC (Han et al., 2023b)	R101	83.2	45.2
Inductive			
ZegCLIP (Zhou et al., 2023)	CLIP-B	93.6	47.5
Cascade-CLIP (Ours)	CLIP-B	93.8	48.4
Transductive			
ZegCLIP+ST (Zhou et al., 2023)	CLIP-B	**94.1**	52.7
Cascade-CLIP+ST (Ours)	CLIP-B	94.0	**52.8**

Table 14. Effect of applying the different numbers of learnable tokens. We report the results of ZegCLIP and our Cascade-CLIP.

	ZegCLIP			Cascade-CLIP (Ours)		
Number	mIoU^S↑	mIoU^U↑	hIoU↑	mIoU^S↑	mIoU^U↑	hIoU↑
35	39.4	40.9	40.1	40.6	42.3	41.4
50	39.2	39.7	39.5	**41.1**	**43.4**	**42.2**
100	**40.2**	**41.4**	**40.8**	40.8	42.0	41.4

Effect of the number of learnable tokens. The learnable tokens are integrated into the CLIP frozen visual encoder to facilitate deep prompt tuning. We present results for different numbers of learnable tokens and compare them against the baseline ZegCLIP. As shown in Tab. 14, our proposed Cascade-CLIP with 35 learnable tokens outperforms ZegCLIP with 100 tokens, achieving the best performance with 50 learnable tokens. This demonstrates the superior effectiveness and efficiency of our approach in zero-shot segmentation by leveraging multi-level visual features in a cascade manner.

Ablation on the cascaded encoders. Our Cascade-CLIP can flexibly integrate the features from different encoder blocks. As shown in Tab. 15, both deep (the 1st result) and intermediate features (the 2nd result) contribute to improving the performance compared to using only the last features. This again demonstrates that intermediate layer features can complement last layer features, thus improving segmentation results. Notably, competitive performance is achieved even without employing last-layer features (the 3rd result). Moreover, simultaneously fusing these features further enhances the overall model performance, indicating that our approach has the ability to effectively exploit distinctive features without compromising transferability.

Table 15. Ablation of cascaded decoder. ✓indicates the fusion of { }-layer features in the Transformer block, aligning them with an individual decoder.

{6-8}	{9-11}	{12}	mIoU^S↑	mIoU^U↑	hIoU↑
	✓	✓	40.2	40.2	40.2
✓		✓	40.5	42.2	41.3
✓	✓		40.3	41.8	41.1
✓	✓	✓	**41.1**	**43.4**	**42.2**

Table 16. Ablation of variance σ of the Gaussian function in NGA module. We present the initial weights of the corresponding features at different variance values.

σ	values of weights	mIoU^S↑	mIoU^U↑	hIoU↑
0.8	{0.04, 0.46, 1.00}	92.3	82.6	87.3
1.0	{0.14, 0.61, 1.00}	**92.7**	**83.1**	**87.7**
1.2	{0.25, 0.71, 1.00}	92.4	79.1	85.1

Table 17. Variance weights of the Gaussian function in NGA module before and after training.

Block splitting manner	Initial weights	Weight after training
{6-8}	{0.14, 0.61, 1.00}	{0.13, 0.59, 1.02}
{9-11}	{0.14, 0.61, 1.00}	{0.13, 0.59, 1.01}

Table 18. Ablation of weights of dice loss (α) and focal loss (β).

α	β	mIoU^S↑	mIoU^U↑	hIoU↑
1	10	87.7	72.4	79.3
1	50	91.3	81.1	85.9
1	100	**92.7**	**83.1**	**87.7**
1	150	91.3	82.5	86.7

Ablation on the variance parameter in NGA. Our NGA module assigns distinct Gaussian weights to blocks for feature fusion based on their neighborhood relative distances. The variance σ of Gaussian weights is a key parameter: higher values lead to consistent weighting across feature blocks, while lower values degenerate into reliance on a single layer of features. Tab. 16 shows that increasing the value of σ causes the mIoU in the unseen class to drop significantly despite similar performance in seen classes. We also present the variance weights of the NGA module in different split encoder stages before and after training, as shown in Tab. 17.

Ablation on the loss weights. We conduct ablation experiments to examine the impact of dice loss weights (α) and focal loss weights (β) on performance using the Pascal-VOC dataset. The results in Tab. 18 indicate that emphasizing focal loss weights 100 times more than dice loss weights is more effective.

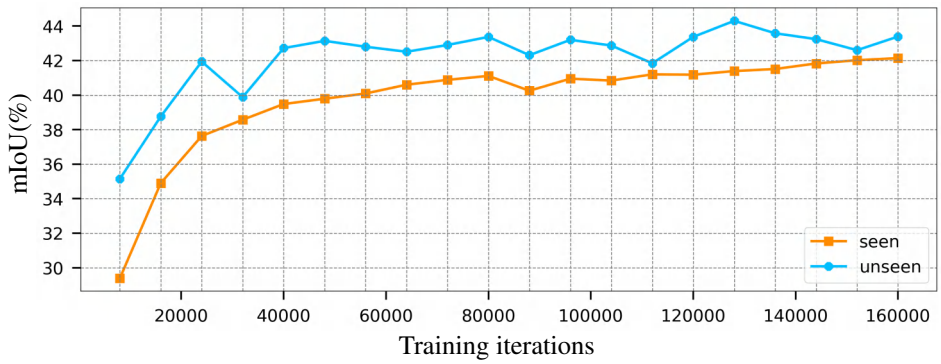


Figure 7. Intersection over mIoU (%) on seen and unseen classes rises with the number of training iterations.

Training iterations. We employ 80000 default training iterations on COCO for Cascade-CLIP to ensure a fair comparison with previous methods. Additionally, We investigate the impact of training iterations for our Cascade-CLIP. Fig. 7 shows the increment in iterations correlates with a steady enhancement of mIoU for seen classes, accompanied by a notable improvement in performance for unseen classes. This indicates that our approach maintains strong zero-shot capability on unseen classes, transitioning CLIP from image-level recognition to pixel-level segmentation.

D. Visualizations

We further evaluate the segmentation performance of our method in transductive settings, comparing it with existing segmentation methods. The qualitative zero-shot segmentation results shown in Fig. 8 demonstrate that our Cascade-CLIP model yields very clear segmentation masks in most cases. For example, our model more accurately segments the shape of the clouds, although their shape is irregular in both the first and second examples. In the third example, our segmentation mask clearly displays the shape of the skateboard, although it occupies only a very small area in the image. These observations confirm the remarkable effectiveness of our Cascade-CLIP approach.

Figure 8. Comparing qualitative zero-shot segmentation results among various semantic segmentation methods with the transductive setting on the COCO-Stuff dataset.