Side Adapter Network for Open-Vocabulary Semantic Segmentation

Mengde Xu[1,2]*, Zheng Zhang[1,2]*, Fangyun Wei[2], Han Hu[2], Xiang Bai[1]†

[1]Huazhong University of Science and Technology [2]Microsoft Research Asia

https://mendelxu.github.io/SAN

Figure 1. Segmentation results on ImageNet. For each image, we combine its category with the coco categories as the vocabulary during inference and only visualize mask of the annotated category.

Abstract

This paper presents a new framework for open-vocabulary semantic segmentation with the pre-trained vision-language model, named Side Adapter Network (SAN). Our approach models the semantic segmentation task as a region recognition problem. A side network is attached to a frozen CLIP model with two branches: one for predicting mask proposals, and the other for predicting attention bias which is applied in the CLIP model to recognize the class of masks. This decoupled design has the benefit CLIP in recognizing the class of mask proposals. Since the attached side network can reuse CLIP features, it can be very light. In addition, the entire network can be trained end-to-end, allowing the side network to be adapted to the frozen CLIP model, which makes the predicted mask proposals CLIP-aware. Our approach is fast, accurate, and only adds a few additional trainable parameters. We evaluate our approach on multiple semantic segmentation benchmarks. Our method significantly outperforms other counterparts, with up to 18 times fewer trainable parameters and 19 times faster inference speed. Fig. 1 shows some visual-

ization results on ImageNet. We hope our approach will serve as a solid baseline and help ease future research in open-vocabulary semantic segmentation.

1. Introduction

Recognizing and segmenting the visual elements of any category is the pursuit of semantic segmentation. Modern semantic segmentation methods [5, 7, 23] rely on large amounts of labeled data, but typically datasets often only consist of tens to hundreds of categories, and expensive data collection and annotation limit our possibilities to further expand the categories. Recently, large-scale vision-language models [17, 27, 36, 37], represented by CLIP [27], have enabled arbitrary category recognition at the image level, *i.e., open-vocabulary image classification*, and this great success encourages us to explore its adaptation in semantic segmentation.

Applying the CLIP model in open-vocabulary semantic segmentation is challenging because the CLIP model is trained by image-level contrastive learning. Its learned representation lacks the pixel-level recognition capability that is required for semantic segmentation. One solution [12, 19] to remedy the granularity gap of representation is fine-tuning the model on the segmentation dataset. However, the data sizes of segmentation datasets are much less than the vision-language pre-training dataset, so the capability of fine-tuned models on open-vocabulary recognition is often compromised.

Modeling semantic segmentation as a region recognition problem bypasses the above difficulties. Early attempts [9, 33] adopt a two-stage training framework. In the first stage, a stand-alone model is trained to generate a set of masked image crops as mask proposals. In the second stage, the vision-language pre-training model (*e.g.* CLIP) is used to recognize the class of masked image crops. However, since the mask prediction model is completely independent of the vision-language pre-training model, it misses the opportunity to leverage the strong features of the vision-language pre-training model and the predicted masked image crops may be unsuitable for recognition, which leads to a heavy, slow, and low-performing model.

This work seeks to fully unleash the capabilities of the vision-language pre-training model in open vocabulary semantic segmentation. To reach this goal, we present a new framework (Fig. 2), called *side adapter network* (SAN). Its mask prediction and recognition are *CLIP-aware* because of end-to-end training, and it can be lightweight due to leveraging the features of CLIP.

The side adapter network has two branches: one predicting mask proposals, and one predicting attention biases that are applied to the self-attention blocks of CLIP for mask class recognition. We show this *decoupled* design improves

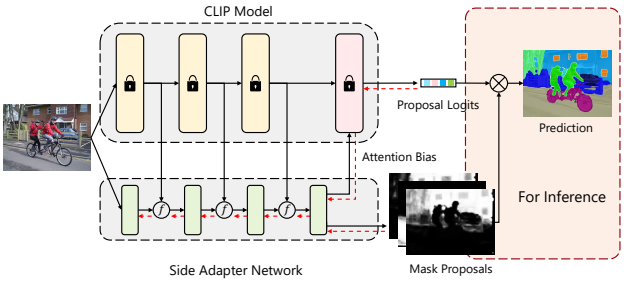

Figure 2. Overview of our *SAN*. The red dotted lines indicate the gradient flow during training. In our framework, the frozen CLIP model still serves as a classifier, and the side adapter network generates mask proposals and attention bias to guide the deeper layers of the CLIP model to predict proposal-wise classification logits. During inference, the mask proposals and the proposal logits are combined to get final predictions through *Matmul*.

the segmentation performance because the region used for CLIP to recognize the mask may be different from the mask region itself. To minimize the cost of CLIP, we further present a *single-forward* design: the features of shallow CLIP blocks are fused to SAN, and other deeper blocks are combined with attention biases for mask recognition. Since the training is end-to-end, the side adapter network can be maximally adapted to the frozen CLIP model.

With the aim of fairness and reproducibility, our study is based on officially released CLIP models. We focus on the released ViT CLIP models because the vision transformer has *de facto* substituted ConvNet as the dominant backbone in the computer vision community, and for conceptual consistency and simplicity, the side adapter network is also implemented by the vision transformer.

Accurate semantic segmentation needs high-resolution images, but the released ViT CLIP models are designed for low-resolution images (*e.g.* 224×224) and directly apply to high-resolution images giving a poor performance. To alleviate the conflicts in input resolutions, we use low-resolution images in the CLIP model and high-resolution images in the side adapter network. We show this *asymmetric input resolution* is very effective. In addition, we also explore only fine-tuning the positional embedding of the ViT model and note improvements.

We evaluate our method on various benchmarks. Following the setting of previous works [22, 33], the COCO Stuff [4] dataset is used for training, and Pascal VOC [11], Pascal Context-59 [25], Pascal Context-459 [25], ADE20K-150 [41], and ADE20K-847 [41] are used for testing. Without bells and whistles, we report state-of-the-art performance on all benchmarks: with the CLIP ViT-L/14 model, our method achieves 12.4 mIoU on ADE-847, 15.7 mIoU on PC-459, 32.1 mIoU on ADE-150, 57.7 mIoU on PC-59, and 94.6 mIoU on VOC. Compared to the previous best

method, our method has an average of +1.8 mIoU improvements on 5 datasets for ViT-B/16, and +2.3 mIoU improvements for ViT-L/14, respectively. By further applying ensemble trick, the average performance gap increases to +2.9 mIoU and +3.7 mIoU for ViT-B/16 and ViT-L/14.

Along with the excellent performance, our approach requires only 8.4M trainable parameters with 64.3 GFLOPs, which is only 13% and 20% of [10], 6% and less than 1% of [22], respectively.

2. Related Works

Large-scale vision-language pre-training model The goal of visual-language pre-training is to learn generic representations of vision and language. Early works [6, 20, 24, 28] in this area mainly followed the paradigm of first pre-training models on visual and language data of moderate size, and then fine-tuning them on downstream visual-language tasks, such as VQA [2] and image captioning, to validate the benefits of pre-training. Recently, however, CLIP [27] and ALIGN [17] demonstrates that visual-language models pre-trained on large-scale noisy text-image pairs also have the capabilities on open-vocabulary recognition, in addition to serving as a good starting point for downstream tasks. Many recent works have also confirmed the observation and achieved impressive performance on open-vocabulary image recognition [1,36,37] and other downstream tasks [13, 15, 26, 30].

Our work further explores leveraging the vision-language pre-training models' ability to open-vocabulary recognition on semantic segmentation, which is more challenging with the misalignment between the pre-training and the pixel-level recognition. Specially, we focus on the CLIP model and extend its power in open-vocabulary semantic segmentation.

Models Tuning of downstream tasks. Fine-tuning all model parameters is the most common approach to leverage the pre-training models on downstream tasks. However, as pre-training models become larger and stronger, fine-tuning gradually become an inefficient approach and can compromise the model capability learned in the pre-training stage. Therefore, the new approaches to model tuning are starting to attract attention. Earlier explorations [16, 18, 21] appeared first in NLP community. Recently, with the emergence of large-scale vision models, the exploration in computer vision has also become intensive. CoOp [43] fine-tunes the CLIP model for image classification tasks by training only the input prompt of the CLIP's text encoder. Tip-Adapter [39] and VL-Adapter [35] insert trainable adapter modules into a fixed CLIP model and finetune only the adapters with few-shot supervision. These methods mainly focus on image-level recognition tasks or vision-language tasks.

The most related works to us are Side-Tuning [38] and its variants [34], a side network is attached to the pre-training model and the final representation is a combination of the side network and the pre-training model. However, these efforts are mostly conceptual works and cannot be directly used for open-vocabulary semantic segmentation.

Open-vocabulary Semantic Segmentation Earlier work [3, 31, 40] for open-vocabulary semantic segmentation focus on learning a joint embedding space between image pixels and class name/description. Most recently, driven by the effectiveness of large-scale vision-language pre-training models for open-vocabulary recognition, many approaches explore their application on open-vocabulary semantic segmentation. Some of them [12, 19, 22, 42] fine-tune the vision-language pre-training models, which requires a large amount of additional data or compromises the open-vocabulary capability of the vision-language pre-training model.

SimSeg [33] presents a two-stage framework: first generating masked image crops and then recognizing the crops by a frozen CLIP. However, it requires a heavy mask generator, and CLIP must be forwarded multiple times, making it inefficient in terms of both model size and inference speed. Besides, the mask generator is *CLIP-unaware*, further limiting its performance. MaskCLIP [10] improves the two-stage framework by progressively refining the predicted masks by the CLIP encoder, and applying masks in attention layers to avoid forwarding multiple times, which was first introduced by [7]. However, MaskCLIP still needs a heavy mask generator, the initial mask prediction is also *CLIP-unaware*, and the mask prediction and recognition are *coupled*.

Our approach is an end-to-end framework, the mask prediction is lightweight and *CLIP-aware*, and the mask recognition is *decoupled* from mask prediction. These differences allow our approach can better leverage the capability of CLIP than two-stage approaches [10, 33].

3. Side Adapter Network

To fully unleash the capability of CLIP in open vocabulary semantic segmentation, we present *Side Adapter Network* (SAN), which is an end-to-end framework where mask prediction and recognition are intertwined with the CLIP model. The SAN is implemented by a lightweight vision transformer that can leverage the feature of CLIP, and it has two types of outputs: mask proposals and attention biases. The attention biases are applied to the self-attention of CLIP for recognizing the class of mask proposals. In practice, we fuse the feature of shallow CLIP layers into SAN, and apply the attention biases to rest deeper CLIP layers for recognition. With this *single-forward* design, the cost of CLIP model can be minimized.

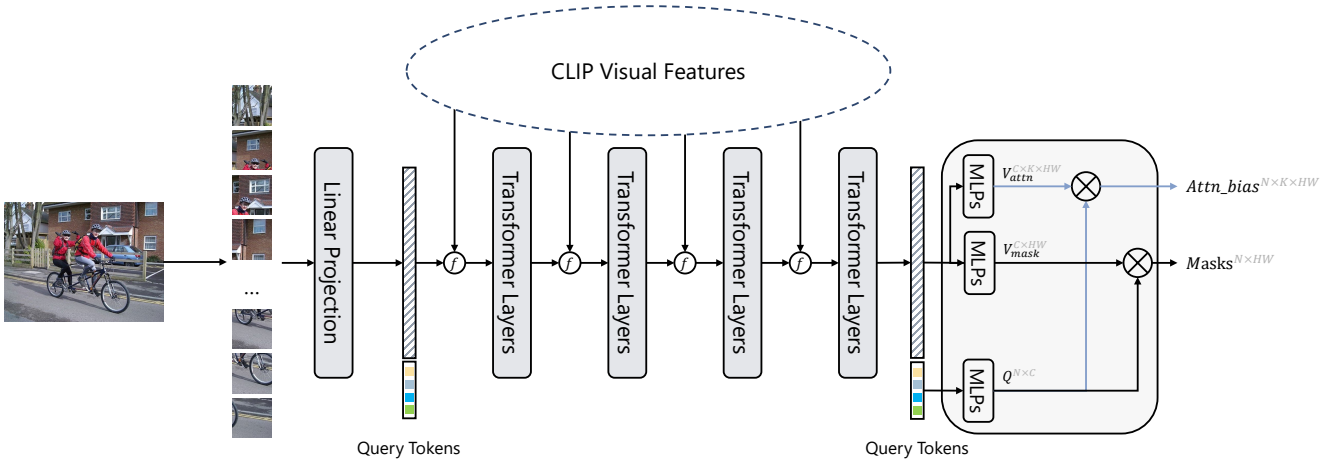

Figure 3. The architecture of the side adapter network. The side adapter network projects the input image to visual tokens and appends query tokens to them at the beginning. Further, it fuses the immediate features of the CLIP model in the middle of transformer layers. The query and visual features are encoded with MLP layers to generate the attention biases and the mask proposals.

The detailed architecture of SAN is shown in Fig. 3. The input image is split into 16×16 patches. A linear embedding layer is applied to project patches as visual tokens. These visual tokens are then concatenated with N learnable query, and fed into subsequent transformer layers. Following the common practices [7, 8], we add the absolute position embedding in each transformer block for both visual tokens and query tokens. The position embedding is shared across layers.

There are two outputs of SAN: the mask proposals and the corresponding attention biases used for mask recognition. In mask prediction, the query tokens and visual tokens are first projected as 256-dimension by two individual 3-layer MLPs, we denoted the projected query tokens as $\mathbf{Q}_{\texttt{mask}} \in \mathbb{R}^{N\times 256}$, where N[1] is the number of query tokens, and the projected visual tokens as $\mathbf{V}_{\texttt{mask}} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times 256}$, where H and W are the height and width of the input image. Then, the masks are generated by the inner product of $\mathbf{Q}_{\texttt{mask}}$ and $\mathbf{V}_{\texttt{mask}}$:

$$\mathbf{M} = \mathbf{V}_{\texttt{mask}}\mathbf{Q}_{\texttt{mask}}^{\mathrm{T}} \tag{1}$$

, where $\mathbf{M} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times N}$. Generating attention bias is similar to mask prediction. The query tokens and visual tokens are also projected by a 3-layer MLPs, denoted as $\mathbf{Q}_{\texttt{attn}} \in \mathbb{R}^{N\times 256}$ and $\mathbf{V}_{\texttt{attn}} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times K\times 256}$, where K is the attention head number of ViT CLIP model. By inner producing $\mathbf{Q}_{\texttt{attn}}$ and $\mathbf{V}_{\texttt{attn}}$, we have the the attention biases:

$$\mathbf{B} = \mathbf{V}_{\texttt{attn}}\mathbf{Q}_{\texttt{attn}}^{\mathrm{T}} \tag{2}$$

,where $\mathbf{B} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times K\times N}$. In addition, if needed, the attention biases will be further resized to $\mathbf{B} \in \mathbb{R}^{h\times w\times K\times N}$, where h and w is the height and width of the attention map in CLIP. In practice, the $\mathbf{Q}_{\texttt{mask}}$ and $\mathbf{Q}_{\texttt{attn}}$ can be shared, and the attention biases will be applied in several self-attention layers of CLIP, *i.e*. the biases are used in different self-attention layers.

The motivation behind the *decoupled* design of mask prediction and recognition is intuitive: the region of interest used to recognize the mask in CLIP may differ from the mask region itself. We show the effectiveness of this design in Tab. 7.

Feature fusion on visual tokens The ViT model consists of visual tokens and a `[CLS]` token, but we only fuse the visual tokens to the SAN. Since the number and feature dimension of the visual tokens may be different between CLIP and SAN, we first re-arrange visual tokens to feature maps that undergo a 1×1 convolution and the resize operation to adjust channel dimension and feature map size, and then merged them with the corresponding feature map of SAN by element-wise addition. The feature fusion will be performed several times, taking the 12-layer ViT-B/16 CLIP model and an 8-layers SAN model as an example. We fuse the feature of $\{\texttt{stem}, 3, 6, 9\}$ layer of CLIP with the feature of $\{\texttt{stem}, 1, 2, 3\}$ layer of SAN.

Our feature fusion has an intuitive design and a more sophisticated structure may improve the performance, but it is not the focus of this work.

Mask recognition with attention bias The original CLIP model can only perform image-level recognition through the `[CLS]` token. Our work, without changing the parameters of the CLIP model, attempts to allow accurate mask recognition by guiding the attention map of `[CLS]` token

[1]By default, N=100.

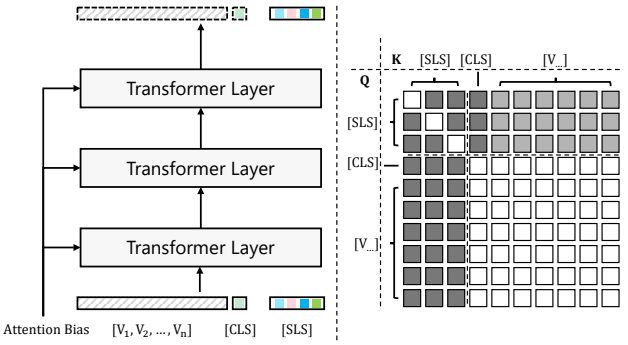

Figure 4. Illustration of using attention bias in CLIP to predict masks. **(Left)** A set of `[SLS]` tokens (*i.e*. The *shadow* `[CLS]` token copies) are created and applied to CLIP. These `[SLS]` tokens are updated under the effect of attention bias. **(Right)** The diagram shows how different types of tokens interact with each other. The color of the squares indicates the relationship between query token and key token: black means query is not updated by key, white means query can normally be updated by key, and gray means the the query can be updated by key under the effect of attention bias.

on the region of interest. To achieve this goal, we create a set of *shadow* `[CLS]` token copies, dubbed `[SLS]` tokens (MaskCLIP [10] adopted a conceptually similar design, See Sec. 1.2 of the supplementary material for detailed discussion). These `[SLS]` tokens are unidirectionally updated by visual tokens, but neither visual tokens nor `[CLS]` tokens are affected by them (Fig. 4). When updating `[SLS]` tokens, the predicted attention biases $\mathbf{B}_k \in \mathbb{R}^{h\times w\times N}$ are added to the attention matrix:

$$\mathbf{X}^{l+1}_{\texttt{[SLS]}} = \texttt{softmax}(\mathbf{Q}^l_{\texttt{[SLS]}}\mathbf{K}^l_{\texttt{visual}} + \mathbf{B}_k)\mathbf{V}^l_{\texttt{[SLS]}} \quad (3)$$

, where l indicates layer number, k indicates the k-th attention head, $\mathbf{Q}_{\texttt{[SLS]}} = \mathbf{W_q}\mathbf{X}_{\texttt{[SLS]}}$ and $\mathbf{V}_{\texttt{[SLS]}} = \mathbf{W_v}\mathbf{X}_{\texttt{[SLS]}}$ are query and value embedding of `[SLS]` tokens, and $\mathbf{K}_{\texttt{visual}} = \mathbf{W_k}\mathbf{X}_{\texttt{visual}}$ is the key embedding of visual tokens. $\mathbf{W_q}$, $\mathbf{W_k}$, $\mathbf{W_v}$ are weights of query, key, and value embedding layer, respectively.

We note that the computation complexity here is $\mathcal{O}((T_{\texttt{visual}}+T_{\texttt{[CLS]]}}+T_{\texttt{[SLS]}})^2)$, where $T_{\texttt{visual}}$, $T_{\texttt{[CLS]]}}$ and $T_{\texttt{[SLS]]}}$ are the number of different types of tokens, if implemented by concatenating all types of tokens together and using a masked self-attention layer. However, we can update the `[SLS]` token via the cross-attention, which shares the embedding weights with self-attention. Thus the computation complexity becomes $\mathcal{O}((T_{\texttt{visual}} + T_{\texttt{[CLS]]}})^2 + T_{\texttt{[SLS]}}(T_{\texttt{visual}} + T_{\texttt{[CLS]]}}))$.

With attention biases, the feature of `[SLS]` tokens gradually evolves to fit mask prediction, and the class prediction of masks can be easily obtained by comparing the distance/similarity between the `[SLS]` token and the CLIP text embedding of class names, denoted as $\mathbf{P} \in \mathbb{R}^{C\times N}$, where C is class number.

Segmentation map generation With the mask proposals $\mathbf{M} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times N}$ and the class prediction of masks $\mathbf{P} \in \mathbb{R}^{C\times N}$, we can compute the segmentation map:

$$\mathbf{S} = \mathbf{M} \times \mathbf{P}^{\mathrm{T}} \quad (4)$$

, where $\mathbf{S} \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times C}$. It is a standard output of semantic segmentation and is therefore compatible with mainstream semantic segmentation evaluation.

To train our model, we follow the practice of [7]. The mask generation is supervised with the dice loss $L_{\texttt{mask\_dice}}$ and binary cross-entropy loss $L_{\texttt{mask\_bce}}$. The mask recognition is supervised with the cross-entropy loss $L_{\texttt{cls}}$. The total loss is:

$$L_{\texttt{seg}} = \lambda_1 L_{\texttt{mask\_dice}} + \lambda_2 L_{\texttt{mask\_bce}} + \lambda_3 L_{\texttt{cls}} \quad (5)$$

The loss weight λ_1, λ_2, λ_3 are 5.0, 5.0, and 2.0, respectively. The gradient flow of SAN is shown in Fig. 2. With end-to-end training, the side adapter network can maximally adapt to the frozen CLIP model, thus the mask proposals and attention biases are *CLIP-aware*.

4. Experiments

In this section, we will first introduce the datasets and the evaluation protocol used in our experiments (Sec. 4.1). Then we will describe the implementation details of our experiments (Sec. 4.2). Finally, we will compare our method with the state-of-art methods (Sec. 4.3) and ablate the effectiveness of our method (Sec. 4.4).

4.1. Dataset and Evaluation Protocol

We conduct experiments on 6 datasets: COCO Stuff [4], ADE20K-150 [41], ADE20K-847 [41], Pascal Context-59 [25], Pascal Context-459 [25], and Pascal VOC [11]. Following the common practice [22, 33], all models are trained on the training set of COCO Stuff and evaluated on other datasets.

COCO Stuff It contains 164K images with 171 annotated classes, which are divided into the training set, the validation set, and the test set containing 118K, 5K, and 41K images, respectively. In our experiments, we use the full 118K training set as the training data by default.

ADE20K-150(ADE-150) It is a large-scale scene understanding dataset with 20K training images and 2K validation images, and a total of 150 annotated classes.

Dataset	Label Sim. to COCO Stuff
Pascal VOC	0.91
Pascal Context-59	0.86
Pascal Context-459	0.70
ADE20K-150	0.73
ADE20K-847	0.57

Table 1. The label-set similarity between validation datasets and training set (*i.e*. COCO Stuff). Measured by Hausdorff distance and cosine similarity based on CLIP text encoder.

ADE20K-847(ADE-847) It has the same images as ADE20K-150 but more annotated classes (847 classes), which is a challenging dataset for open-vocabulary semantic segmentation.

Pascal VOC(VOC) Pascal VOC contains 20 classes of semantic segmentation annotations, where the training set, and the validation set contain 1464, and 1449 images, respectively.

Pascal Context-59(PC-59) It is a dataset for semantic understanding which contains 5K training images, 5K validation images, and a total of 59 annotated classes.

Pascal Context-459(PC-459) It has the same images as Pascal Context-59 but more annotated classes (459 classes), which is also widely used in open-vocabulary semantic segmentation.

Dataset Analysis The relationship between the different datasets is a merely touched problem in the previous paper. To clarify and benefit our understanding of the open-vocabulary ability, we hereby give a straightforward analysis by computing the category similarity between other datasets and the training dataset COCO Stuff. We compute the similarity between two datasets with the Hausdorff Distance. For pairwise similarity computing, we extract the text embedding of each concept with the pretrained CLIP text encoder (ViT-L/14) and compute the cosine similarity. The results are presented in Tab. 1. Among the five validation datasets, Pascal VOC and Pascal Context-59 have a high similarity score of up to 0.9, which means they are better at measuring the *in-domain open-vocabulary ability* in terms of the visual categories. Moreover, Pascal Context-459, ADE20K-150, and ADE20K-847 have a lower similarity score, making them better evaluate the *cross-domain open-vocabulary ability*.

Evaluation Protocol Following the common practice [7, 12, 33], we use the mean of class-wise intersection over union (mIoU) to measure the performance of our models. For the system-level comparison, we report the mean and variance of 5 trials to ease the randomness. For the ablation study, we only report the average results of 2 trials for saving cost.

4.2. Implementation Details

Training Setting By default, the side adapter network consists of 8 transformer blocks with channel dimensions of 240, attention heads of 6, patch size of 16, and 100 query tokens. For ViT-B/16 CLIP model (pretrained on 224^2 resolution), we used the first 9 blocks for feature fusion and the last 3 blocks for mask recognition, the input resolution is 320^2. For ViT-L/14 model (pretrained on 336^2 resolution), we use the first 18 blocks for feature fusion and the last 6 blocks for mask recognition, and the input resolution is 448^2. For both ViT-B/16 and ViT-L/16, the input resolution of the side-adapter network is 640^2.

All models are trained on the training set of COCO Stuff dataset. The AdamW optimizer are used with the initial learning rate of 1e-4, weight decay of 1e-4, batch size of 32, and total 60K training iterations, During training, the learning rate is decayed with a poly schedule, with a power of 0.9. We also adopt the data augmentation [7, 22, 33] with random image resizing in the short-side range of [320,1024] and a crop size of 640^2.

4.3. System level comparison

In Tab. 2, we compare our method with other state-of-the-art methods. In comparison with other methods that also use the CLIP ViT models and COCO Stuff dataset, without using ensemble trick[2], our method surpasses other methods under the same setting with an average of +1.8 mIoU for CLIP ViT-B/16, and an average of +2.3 mIou for ViT-L/14, respectively. Further applying the ensemble trick increases the gap to an average of +2.9 mIoU and +3.7 mIoU for CLIP VIT-B/16 and ViT-L/14, respectively.

Notably, the improvements of our method are more pronounced on the ADE-847. As we discussed in Tab. 1, ADE-847 has fewer similar classes to COCO-Stuff, and we argue that the better performance on ADE-847 further affirms the stronger open-vocabulary recognition capability of our approach.

Furthermore, we compare with other methods: SimSeg [33], OvSeg [22], and MaskCLIP [10], which also use CLIP ViT models, in terms of trainable parameters, GFLOPs and inference time (FPS). For a fair comparison, we test all methods under the same environment: single Titan Xp GPU, Xeon E5 v2 CPU (32 core), 252G RAM, PyTorch 1.9.0, and CUDA 11.3. We use images of 640^2 reso-

[2]Previous works [22, 33] ensemble the predictions of the model fine-tuned on COCO Stuff with the predictions of frozen CLIP to get better performance. In our approach, we ensemble our model with the model fine-tuned on COCO Stuff.

Method	VL-Model	Training Dataset	ensemble.	ADE-847	PC-459	ADE-150	PC-59	VOC
Group-VIT [32]	rand. init.	CC12M+YFCC	no.	-	-	-	22.4	52.3
LSeg+ [12]	ALIGN RN101	COCO	no.	2.5	5.2	13.0	36.0	59.0
OpenSeg [12]	ALIGN RN101	COCO	no.	4.0	6.5	15.3	36.9	60.0
LSeg+ [12]	ALIGN EN-B7	COCO	no.	3.8	7.8	18.0	46.5	-
OpenSeg [12]	ALIGN EN-B7	COCO	no.	6.3	9.0	21.1	42.1	-
OpenSeg [12]	ALIGN EN-B7	COCO+Loc. Narr.	no.	8.8	12.2	28.6	48.2	72.2
SimSeg [33]	CLIP ViT-B/16	COCO	yes.	7.0	8.7	20.5	47.7	88.4
SimSeg†	CLIP ViT-B/16	COCO	yes.	6.9	9.7	21.1	51.9	91.8
OvSeg [22]	CLIP ViT-B/16	COCO	yes.	7.1	11.0	24.8	53.3	92.6
SAN(ours)	CLIP ViT-B/16	COCO	no.	**10.1** ± 0.23	**12.6** ± 0.44	**27.5** ± 0.34	**53.8** ± 0.57	**94.0** ± 0.21
MaskCLIP [10]	CLIP ViT-L/14	COCO	no.	8.2	10.0	23.7	45.9	-
SimSeg†	CLIP ViT-L/14	COCO	yes.	7.1	10.2	21.7	52.2	92.3
OvSeg [22]	CLIP ViT-L/14	COCO	yes.	9.0	12.4	29.6	55.7	94.5
SAN(ours)	CLIP ViT-L/14	COCO	no.	**12.4** ± 0.27	**15.7** ± 0.26	**32.1** ± 0.42	**57.7** ± 0.34	**94.6** ± 0.42

Table 2. Performance comparison with state-of-the-art methods. † SimSeg [33] trained with a subset of COCO Stuff in their paper. For a fair comparison, we reproduce their method on the full COCO Stuff with their officially released code. * RN101: ResNet-101 [14]; EN-B7: EfficientNet-B7 [29]; SAN ensemble. is the result using ensemble tricks, not the default setting.

Method	Param. (M)	GFLOPs	FPS
SimSeg	61.1	1916.7	0.8
OvSeg*	147.2	1916.7	0.8
MaskCLIP*	63.1	307.8	4.1
SAN(ours)	**8.4**	**64.3**	**15.2**

Table 3. Training and testing efficiency comparison with other methods. *Param.* stands for the total number of trainable parameters in the methods in millions. The input image is of 640 × 640 resolution. And the clip model is ViT-B/16. * no official code available yet and we re-implement their methods following the description in their papers. OvSeg [22] has similar structures to SimSeg [33] but it finetuned the whole CLIP model,resulting in much more trainable parameters.

lution for all models, and process a single image per inference. The results are summarized in Tab. 3. Our approach outperforms other methods in all aspects. We also visualize the the predictions with our best ViT-L/14 model in Sec.3 of the supplementary material.

4.4. Ablation Studies

We ablate the key design choices of our method on ADE-150. If not specified, the ViT-B/16 CLIP model and an 8-layer side adapter network with a feature dimension of 240 and an attention head of 6 are used as the default setting.

Importance of feature fusion. The key to SAN being lightweight is leveraging the strong features of the CLIP model. We experimentally illustrate the importance of feature fusion in Tab. 4. Without fusing the CLIP feature, the mIoU would drop from 27.8 to 21.1. In addition, we also noticed that fusing the feature of deeper layers (*e.g.* 9th layer) is better than fusing shallower layers (*e.g.* stem layer), and only fusing the feature of 9th layer can reach

Description.	Layers	mIoU
w/o. fusion	none	21.1
single-fusion	stem	20.0
	3rd layer	24.1
	6th layer	26.2
	9th layer	27.1
multi-fusion	{6,9}-layers	27.0
	{3,6,9}-layers	27.7
	{stem,3,6,9}-layers	**27.8**

Table 4. Different feature fusion strategies. The last 3 layers of ViT-B/16 are used for mask prediction in all experiments.

#Feature Fusion Layers	#Recognition Layers	mIoU
12	12	27.6
11	1	25.9
10	2	27.3
9	3	**27.8**
6	6	26.9
3	9	23.8

Table 5. The trade-off between the number of feature fusion layers and the number of mask prediction layers. *Note*: the 2nd row (*i.e.* the {12,12} setting) is the *twice-forward* baseline.

27.1 mIoU, which is +6.0 mIoU higher than baseline without feature fusion. This observation is consistent with the intuition that deeper features tend to be more semantic. In addition, fusing features from multiple layers can further improve performance compared to single-layer fusion by +0.8 mIoU.

To minimize the inference cost of CLIP, we adopt a *single-forward* design that the shallower layers are used for feature fusion, and other deeper layers are used for mask recognition, and thus a trade-off is required, which is examined in Tab. 5, and the best performance is achieved when

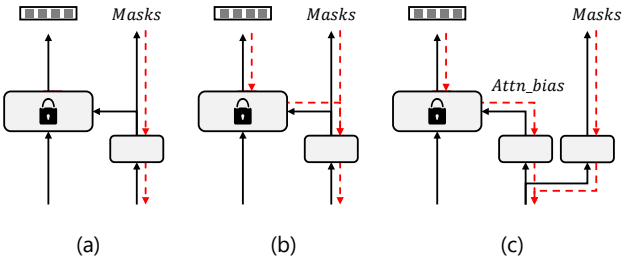

Figure 5. Design choice of mask prediction head. (a) Two-stage training with single head and blocking gradients from CLIP. (b) End-to-end training with single head (c) End-to-end training with decoupled head. The red dotted line indicates the gradient flow during training.

Description	Backbone	*CLIP-aware*	mIoU
SimSeg	ViT-B/16	no.	21.1
MaskCLIP	ViT-L/14	no.	23.7
two-stage training	ViT-B/16	no.	21.6
e2e training	ViT-B/16	yes.	**26.1** (+4.5)

Table 6. *Two-stage vs. end-to-end.* The significant improvement proves the importance of *CLIP-aware* mask prediction.

the first 9 layers are used for feature fusion and the last 3 layers for mask recognition. In addition, we also compared with the *twice-forward* baseline (2nd row in Tab. 5) and did not find a significant difference. [3]

Importance of *CLIP-aware* mask prediction. Unlike other two-stage frameworks [10, 33], our approach is an end-to-end training framework. We study the difference between the two frameworks in Tab. 6. As the attention bias branch must be trained through CLIP, for comparison, we use the mask proposals instead of the attention bias in the self-attention layers of CLIP. If the gradients from CLIP are blocked, the method degenerates into a two-stage framework, *i.e.*, the mask prediction is isolated from CLIP recognition. Otherwise, the method is a *single* head end-to-end training framework, and the mask prediction is *CLIP-aware*.

Tab. 6 shows the end-to-end training has *+4.5* mIoU improvements over two-stage baseline. Besides, we list the results of other two-stage methods as a reference, showing that our two-stage baseline can achieve reasonable performance.

Importance of the *decoupled* head. We study the effect of the *decoupled* head design in Tab. 7. Compared with the *single* head model, the *decoupled* head model has *+1.7* mIoU improvements. Note that both models are trained end-to-end, so their mask predictions are all *CLIP-aware*.

[3] We note a 0.2 mIoU gap which could arise from randomness.

Head	E2E Training	mIoU
single head	yes.	26.1
decoupled head	yes.	**27.8** (+1.7)

Table 7. Comparison on *single* head and *decoupled* head. With few additional parameters and flops, *decoupled* head improves a notable performance. All models are trained end-to-end.

Resolution.	GFLOPs	mIoU
192^2	39.4	25.3
224^2	44.3	26.3
320^2	64.3	**27.8**
448^2	106.3	26.1
640^2	213.4	24.6

Table 8. The influence of ViT-B/16 CLIP model input resolution. We vary CLIP input resolutions, while always using 640^2 images in the side-adapter network.

Description.	Resolution.	mIoU
fixed pos embed.	320^2	27.0
ft. pos embed.	320^2	**27.8**

Table 9. Fine-tuning the position embedding can improve the performance.

Asymmetric input resolution. We based on the officially released ViT CLIP models. They are designed for low-resolution input images (*e.g.* 224^2), while semantic segmentation requires high-resolution images. To resolve the conflicts on input resolution, we use low-resolution images for CLIP model and high-resolution images for SAN model. Tab. 8 shows how the different image resolutions of CLIP model affect performance. In addition, by default, we fine-tune the position embedding of CLIP model, its effects are shown in Tab. 9.

5. Conclusion

In this work, we presented the SAN framework for open-vocabulary semantic segmentation. Our framework succeeds in leveraging the features of the frozen CLIP model and an end-to-end pipeline to adopt the frozen CLIP model maximally. Notably, the proposed framework significantly outperforms the previous state-of-the-art methods on five semantic segmentation benchmarks with much fewer trainable parameters and much less computation cost.

Acknowledge. This work was supported by the National Science Fund for Distinguished Young Scholars of China (Grant No.62225603)

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur

Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *arXiv preprint arxiv:2204.14198*, 2022. 3

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015. 3

[3] Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *NeuralIPS*, 32:468–479, 2019. 3

[4] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *CVPR*, pages 1209–1218, 2018. 2, 5

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4):834–848, 2017. 2

[6] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. In *ECCV*. Springer, 2020. 3

[7] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. 2022. 2, 3, 4, 5, 6

[8] Bowen Cheng, Alexander G Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *arXiv preprint arXiv:2107.06278*, 2021. 4

[9] Jian Ding, Nan Xue, Gui-Song Xia, and Dengxin Dai. Decoupling zero-shot semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11583–11592, 2022. 2

[10] Zheng Ding, Jieke Wang, and Zhuowen Tu. Open-vocabulary panoptic segmentation with maskclip. *arXiv preprint arXiv:2208.08984*, 2022. 3, 5, 6, 7, 8

[11] Mark Everingham and John Winn. The pascal visual object classes challenge 2012 (voc2012) development kit. *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep*, 8:5, 2011. 2, 5

[12] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. 2022. 2, 3, 6, 7

[13] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021. 3

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 7

[15] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 3

[16] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799. PMLR, 2019. 3

[17] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. *arXiv preprint arXiv:2102.05918*, 2021. 2, 3

[18] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *EMNLP*, 2021. 3

[19] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *ICLR*, 2022. 2, 3

[20] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, pages 121–137. Springer, 2020. 3

[21] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *ACL*, 2021. 3

[22] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. *arXiv preprint arXiv:2210.04150*, 2022. 2, 3, 5, 6, 7

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 2

[24] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019. 3

[25] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, pages 891–898, 2014. 2, 5

[26] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021. 3

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 2, 3

[28] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019. 3

[29] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 7

[30] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. Cris: Clip-driven referring image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11686–11695, 2022. 3

[31] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, pages 8256–8265, 2019. 3

[32] Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. *arXiv preprint arXiv:2202.11094*, 2022. 7

[33] Mengde Xu, Zheng Zhang, Fangyun Wei, Yutong Lin, Yue Cao, Han Hu, and Xiang Bai. A simple baseline for open-vocabulary semantic segmentation with pre-trained vision-language model. In *ECCV*, pages 736–753. Springer, 2022. 2, 3, 5, 6, 7, 8

[34] Mohit Bansal Yi-Lin Sung, Jaemin Cho. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. In *arXiv*, 2022. 3

[35] Mohit Bansal Yi-Lin Sung, Jaemin Cho. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *CVPR*, 2022. 3

[36] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arxiv:2205.01917*, 2022. 2, 3

[37] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. Florence: A new foundation model for computer vision. *arXiv preprint arxiv:2111.11432*, 2021. 2, 3

[38] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *ECCV*, pages 698–714. Springer, 2020. 3

[39] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *ECCV*, 2022. 3

[40] Hang Zhao, Xavier Puig, Bolei Zhou, Sanja Fidler, and Antonio Torralba. Open vocabulary scene parsing. In *ICCV*, pages 2002–2010, 2017. 3

[41] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, pages 633–641, 2017. 2, 5

[42] Chong Zhou, Chen Change Loy, and Bo Dai. Denseclip: Extract free dense labels from clip. *arXiv preprint arXiv:2112.01071*, 2021. 3

[43] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021. 3