Paper Review

Extract Free Dense Labels from CLIP

(ECCV 2022)

2023. 3. 24 (Fri)

Chanyoung Kim

MaskCLIP

- **Extract Free Dense Labels from CLIP (Zhou et al., ECCV 2022)**
- MaskCLIP: Masked Self-Distillation Advances Contrastive Language-Image Pretraining (Dong and Zheng et al., arXiv 2022)

Overview

- This paper examine the intrinsic potential of **CLIP for pixel-level dense prediction**, specifically in **semantic segmentation**.
- With minimal modification, this paper shows that **MaskCLIP** yields compelling segmentation results on open concepts across various datasets **in the absence of annotations and fine-tuning**.
- By adding **pseudo labeling** and **self-training, MaskCLIP+** surpasses SOTA transductive zero-shot semantic segmentation methods by large margins

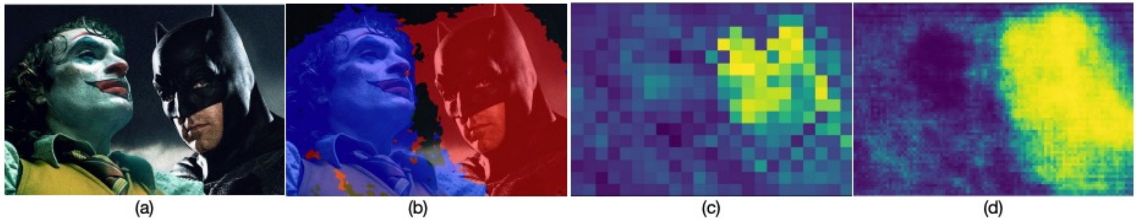
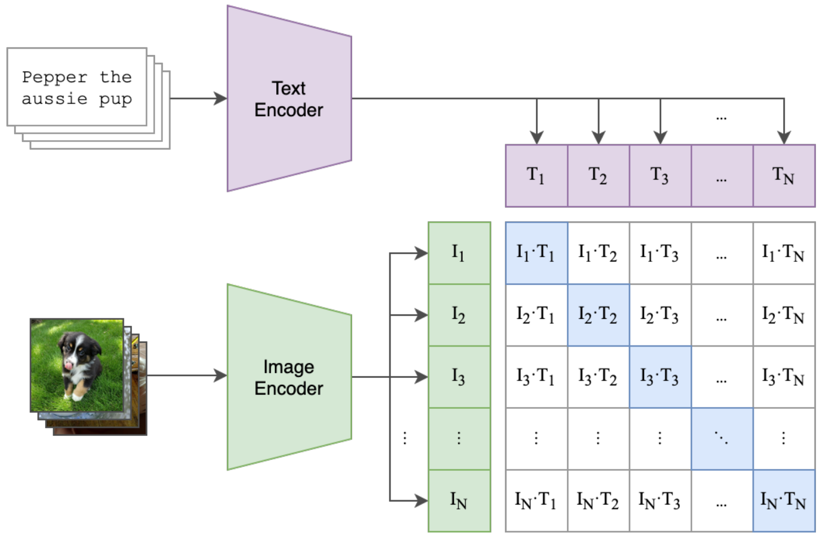


Fig. 1: Here we show the original image in (a), the segmentation result of MaskCLIP+ in (b), and the confidence maps of MaskCLIP and MaskCLIP+ for *Batman* in (c) and (d) respectively. Through the adaptation of CLIP, MaskCLIP can be directly used for segmentation of fine-grained and novel concepts (e.g., *Batman* and *Joker*) without any training operations and annotations. Combined with pseudo labeling and self-training, MaskCLIP+ further improves the segmentation result.

Background

- CLIP (Contrastive Language-Image Pre-Training)

(1) Contrastive pre-training



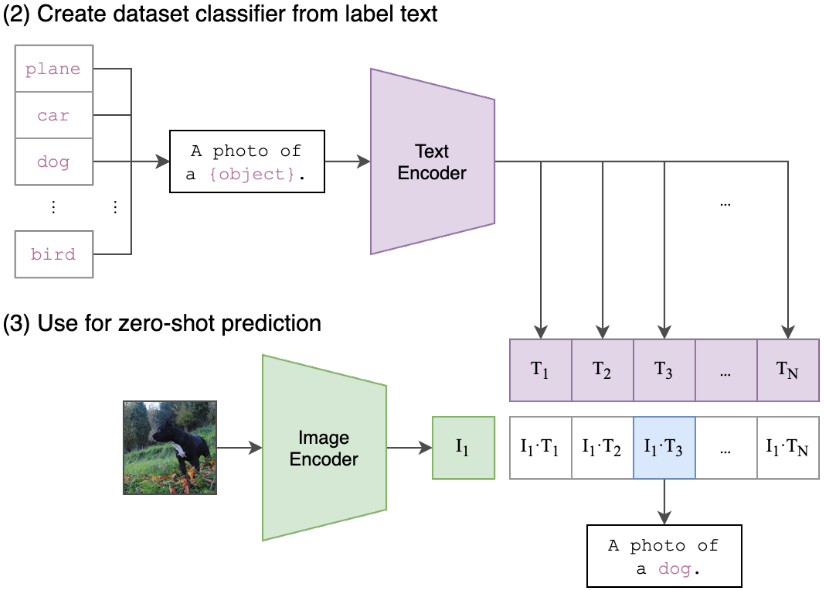

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

Background

- CLIP (Contrastive Language-Image Pre-Training)'s Downstream Tasks

Zero-shot Prediction



Text-Driven Image Editing

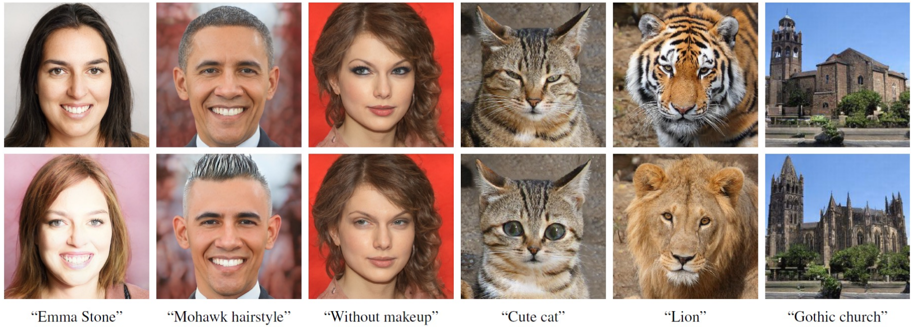


Figure 1. Examples of text-driven manipulations using StyleCLIP. Top row: input images; Bottom row: our manipulated results. The text prompt used to drive each manipulation appears under each column.

StyleCLIP (ICCV 2021)

Background

- CLIP (Contrastive Language-Image Pre-Training)'s Downstream Tasks

Object Detection

Detic (ECCV 2022)

Text-to-Image Generation

Stable Diffusion (CVPR 2022)

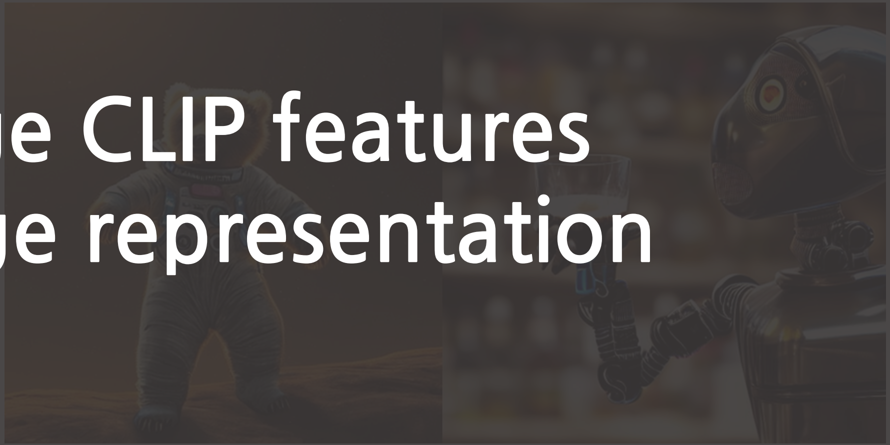
Background

- CLIP (Contrastive Language-Image Pre-Training)'s Downstream Tasks

Object Detection

Detic (ECCV 2022)

Text-to-Image Generation



Stable Diffusion (CVPR 2022)

Mainly leverage CLIP features as a global image representation

Motivation

CLIP
(Image + Language)

-> embed local image semantics in its features

Open vocabulary

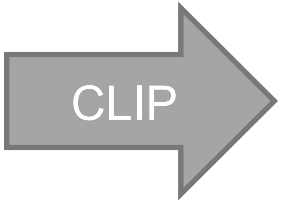
Captures rich contextual information

ex) relation of certain objects & priors of the spatial locations

Potential for dense prediction tasks

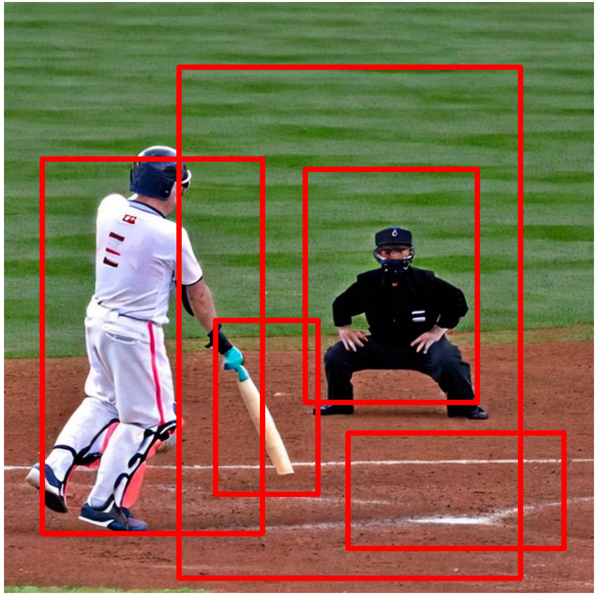
Motivation

The man at bat readies to swing at the patch while the umpire looks on



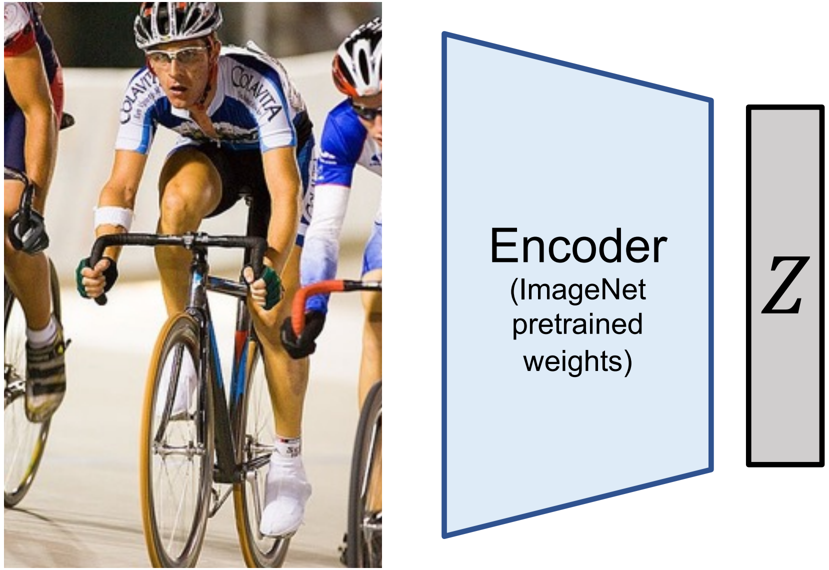
man, bat, swing, patch, man at bat, man at patch, man readies to swing,

Align Local Semantics



Methodology

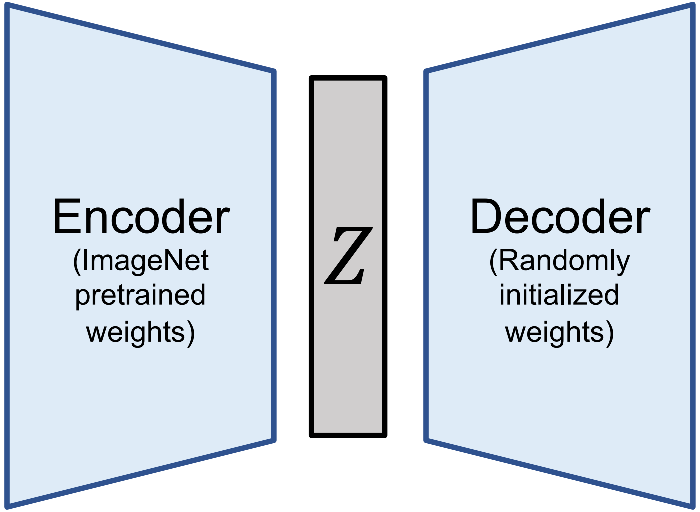
- A naïve solution



1) Initializing the backbone network with the ImageNet pre-trained weights

Methodology

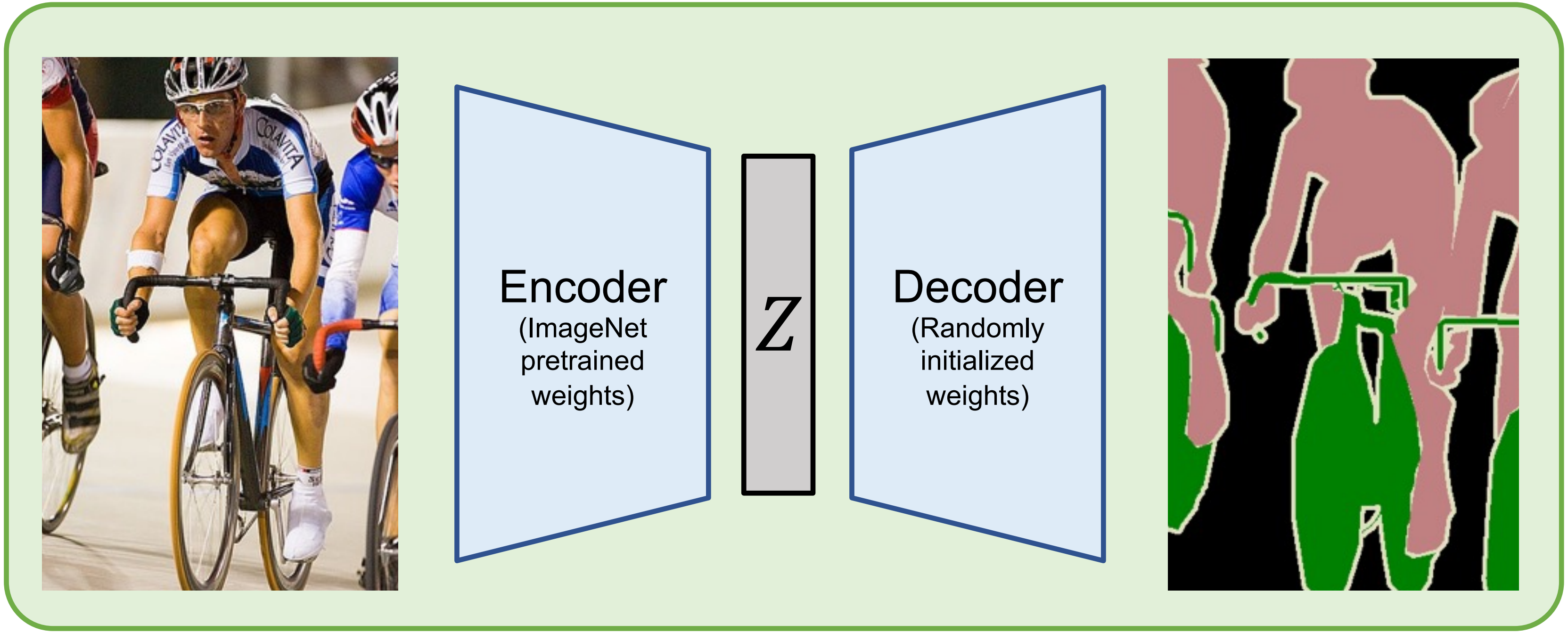
- A naïve solution



2) Adding segmentation-specific network modules with randomly initialized weights

Methodology

- A naïve solution



3) Jointly fine-tuning the backbone and newly added modules

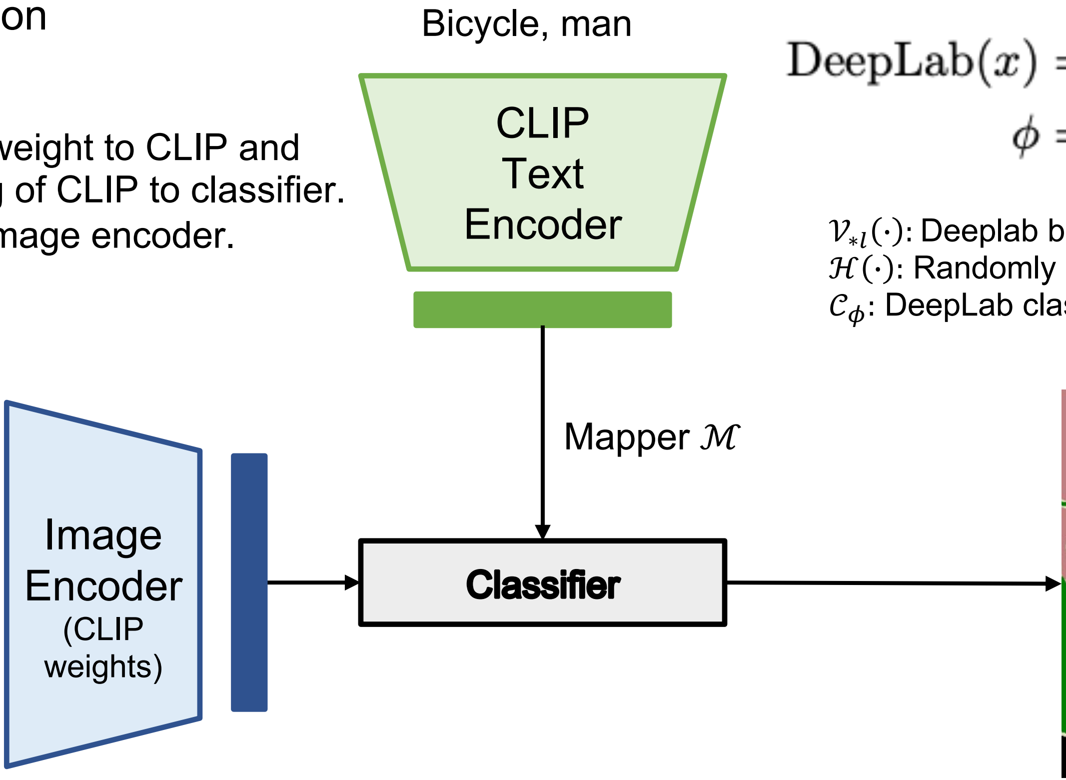
Methodology

- A naïve solution

Replace backbone weight to CLIP and map text embedding of CLIP to classifier. Then fine-tune the image encoder.

$$\mathrm{DeepLab}(x) = \mathcal{C}_\phi(\mathcal{H}(\mathcal{V}_{*l}(x))),$$
$$\phi = \mathcal{M}(t),$$

$\mathcal{V}_{*l}(\cdot)$: Deeplab backbone, ResNet
$\mathcal{H}(\cdot)$: Randomly initialized ASPP module
$\mathcal{C}_\phi$: DeepLab classifier, determined by the $\mathcal{M}$



Methodology

- A naïve solution

Replace backbone weight to CLIP and map text embedding of CLIP to classifier. Then fine-tune the image encoder.

Bicycle, man

CLIP Text Encoder

$$\mathrm{DeepLab}(x) = \mathcal{C}_\phi(\mathcal{H}(\mathcal{V}_{*l}(x))),$$
$$\phi = \mathcal{M}(t),$$

$\mathcal{V}_{*l}(\cdot)$: Deeplab backbone, ResNet
$\mathcal{H}(\cdot)$: Randomly initialized ASPP module
$\mathcal{C}_\phi$: DeepLab classifier, determined by the $\mathcal{M}$

Fails to segment well on unseen classes

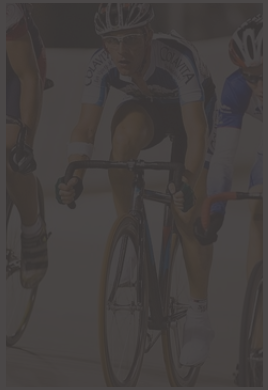


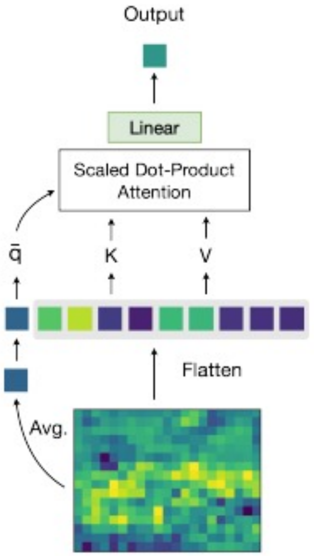
Image Encoder (CLIP weights)

Mapper $\mathcal{M}$

Classifier

Methodology

- MaskCLIP
  - Revisiting the image encoder of CLIP



CLIP's Global Attention Pooling Layer

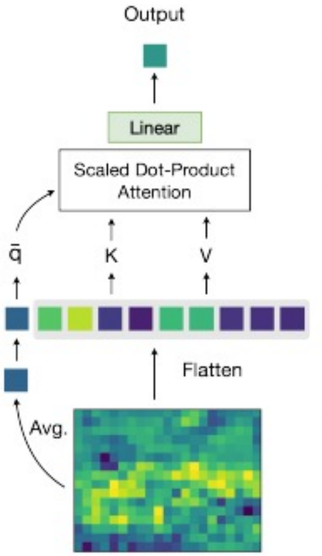
$$\mathrm{AttnPool}(\bar{q}, k, v) = \mathcal{F}(\sum_i \mathrm{softmax}(\frac{\bar{q}k_i^\top}{C})v_i)$$
$$= \sum_i \mathrm{softmax}(\frac{\bar{q}k_i^\top}{C})\mathcal{F}(v_i),$$
$$\bar{q} = \mathrm{Emb_q}(\bar{x}),\ k_i = \mathrm{Emb_k}(x_i),\ v_i = \mathrm{Emb_v}(x_i),$$

$\mathcal{F}(\cdot)$: Linear layer

Methodology

- MaskCLIP

  - Revisiting the image encoder of CLIP



CLIP's Global Attention Pooling Layer

$$\mathrm{AttnPool}(\bar{q}, k, v) = \mathcal{F}\left(\sum_i \mathrm{softmax}\left(\frac{\bar{q}k_i^\top}{C}\right)v_i\right)$$

$$= \sum_i \mathrm{softmax}\left(\frac{\bar{q}k_i^\top}{C}\right)\boxed{\mathcal{F}(v_i),}$$

$$\bar{q} = \mathrm{Emb_q}(\bar{x}),\ k_i = \mathrm{Emb_k}(x_i),\ v_i = \mathrm{Emb_v}(x_i),$$

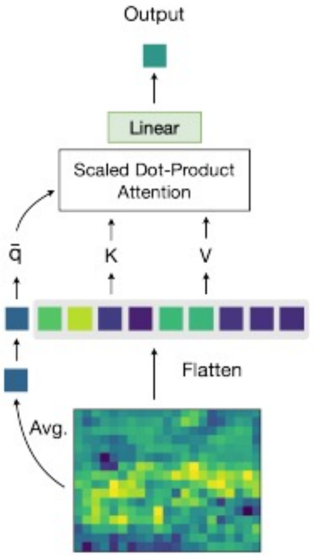
$\mathcal{F}(\cdot)$: Linear layer

Authors think that $\mathcal{F}(v)$ contains rich local semantics cooresponding to the token in text embeddings

Methodology

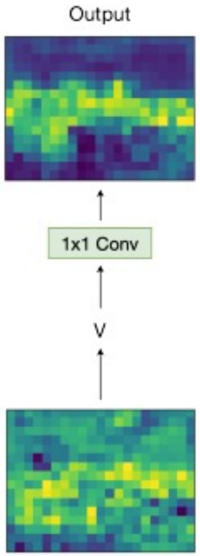
- MaskCLIP

  - Revisiting the image encoder of CLIP



CLIP's Global
Attention Pooling Layer

1) Remove the ***query*** and ***key*** embedding layer
2) Replace linear layers to 1X1 conv layers

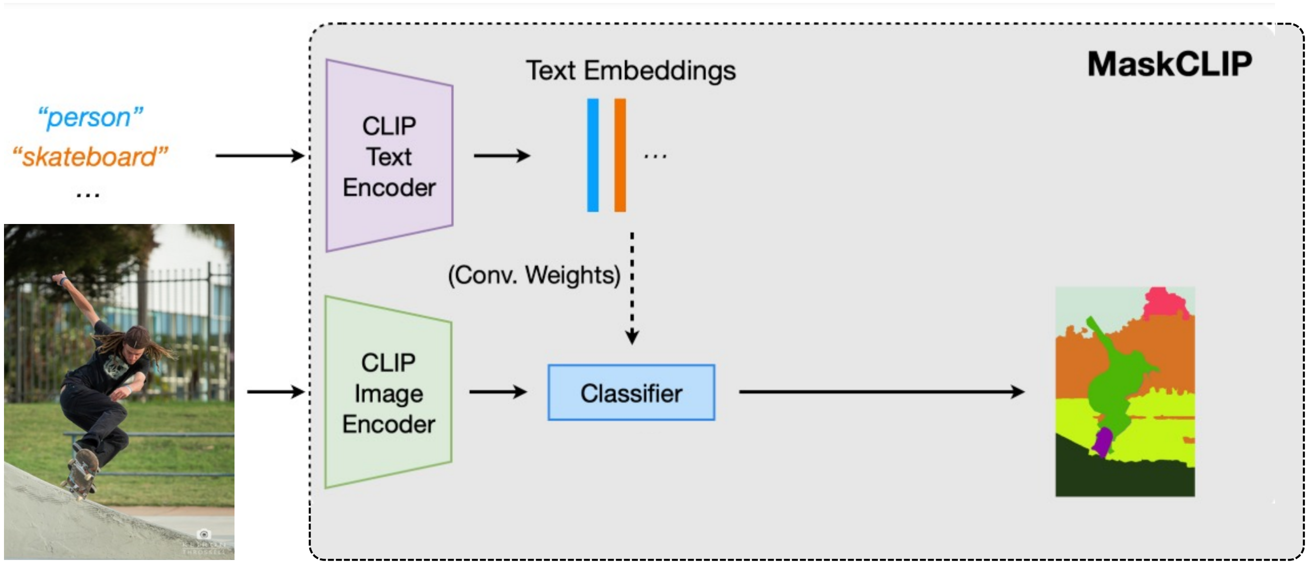


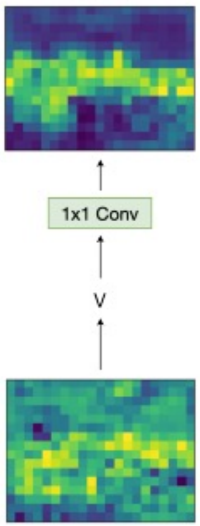
(Our Adaptation)

Methodology

- MaskCLIP

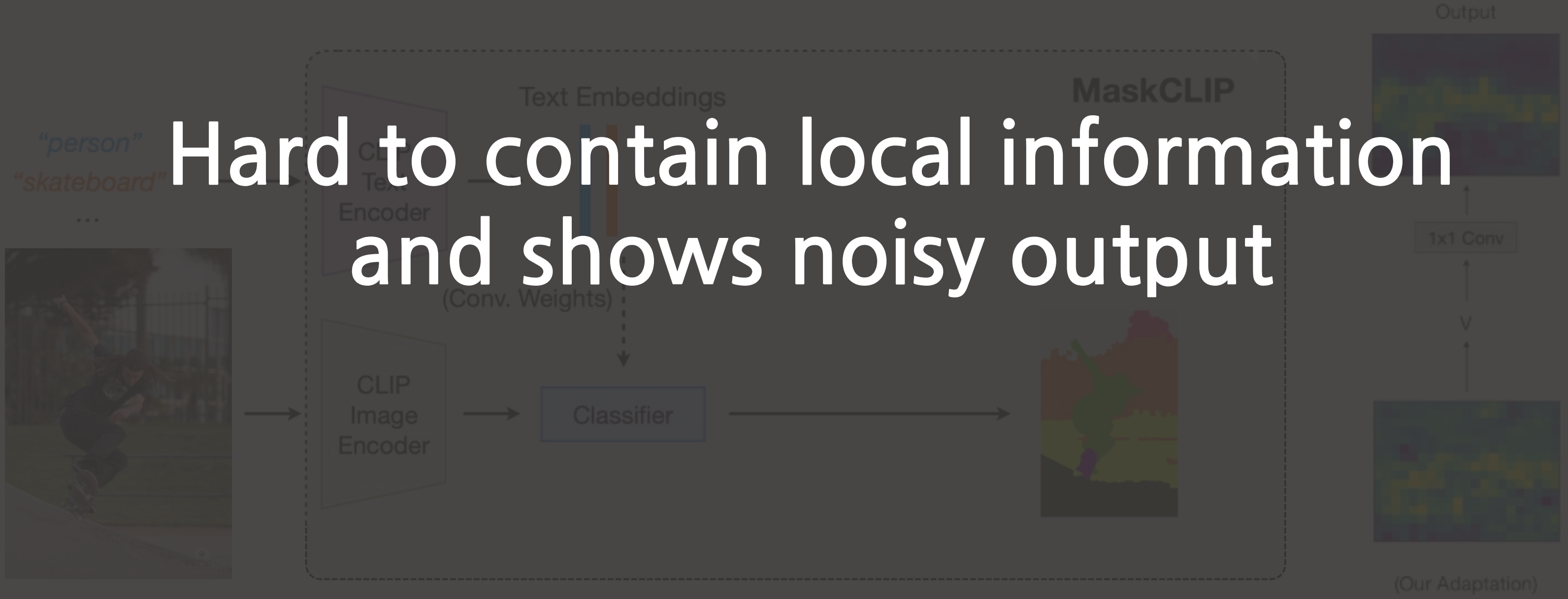
  - Revisiting the image encoder of CLIP





Methodology

- MaskCLIP
  - Revisiting the image encoder of CLIP



Hard to contain local information and shows noisy output

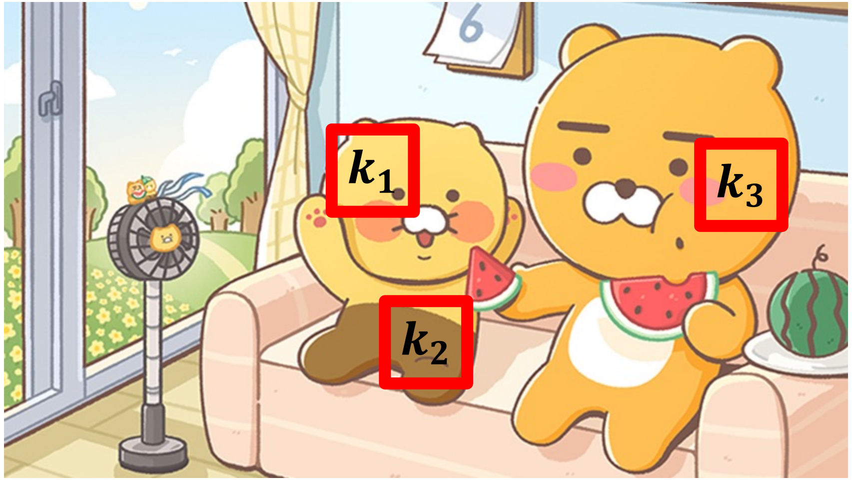
Methodology

- MaskCLIP
  - Refinement strategies (**Key smoothing** and prompt denoising)

Key features can be viewed as the descriptor of the corresponding patch

↓

Patches with similar key features should yield similar predictions



k_1 and k_2 similar
k_1 and k_3 not similar

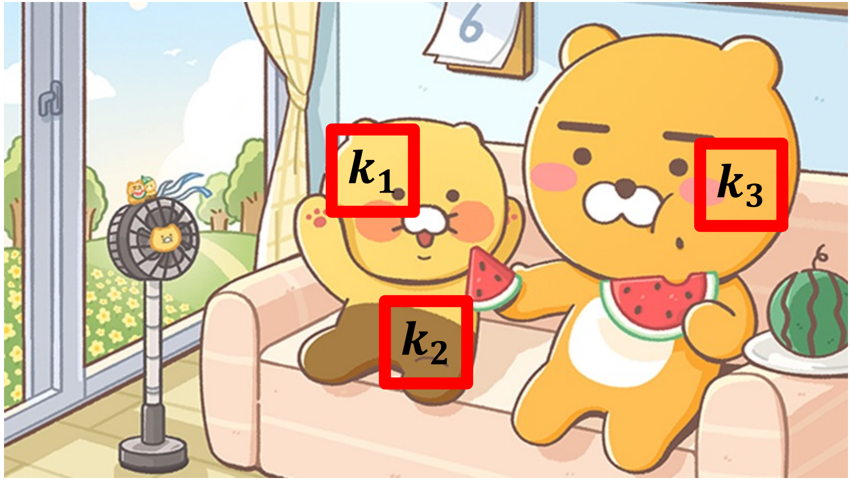
Methodology

- MaskCLIP
  - Refinement strategies (**Key smoothing** and prompt denoising)

Key features can be viewed as the descriptor of the corresponding patch

↓

Patches with similar key features should yield similar predictions



$$\text{pred}_i = \sum_j \cos\left(\frac{k_i}{\|k_i\|_2}, \frac{k_j}{\|k_j\|_2}\right)\text{pred}_i$$

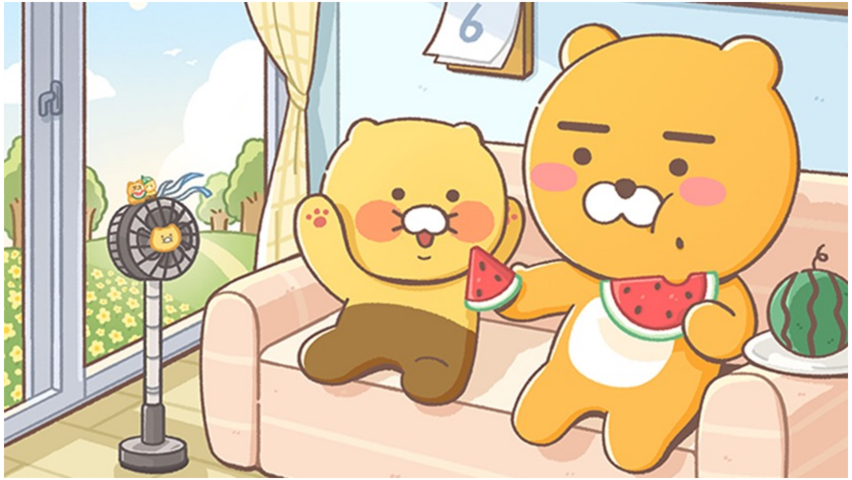
Methodology

- MaskCLIP
  - Refinement strategies (Key smoothing and **prompt denoising**)

A small proportion of the classes appear in a single image

↓

Degrades performance



choonsik, Lyan, Sofa, watermelon and calender

Removes the prompt with target class if its class confidence at all spatial locations is all less than a threshold t = 0.5

Methodology

- MaskCLIP
  - Multiple unique merits of MaskCLIP

MaskCLIP can be used as a **free segmentation annotator without any training**

Possesses the ability to segment **open vocabulary classes**, as well as **fine-grained classes**

Demonstrates great robustness to **natural distribution shift** and **input corruptions**

Methodology

- MaskCLIP
  - Multiple unique merits of MaskCLIP

Network architecture is rigid

MaskCLIP can be used as a **free segmentation annotator without any training**

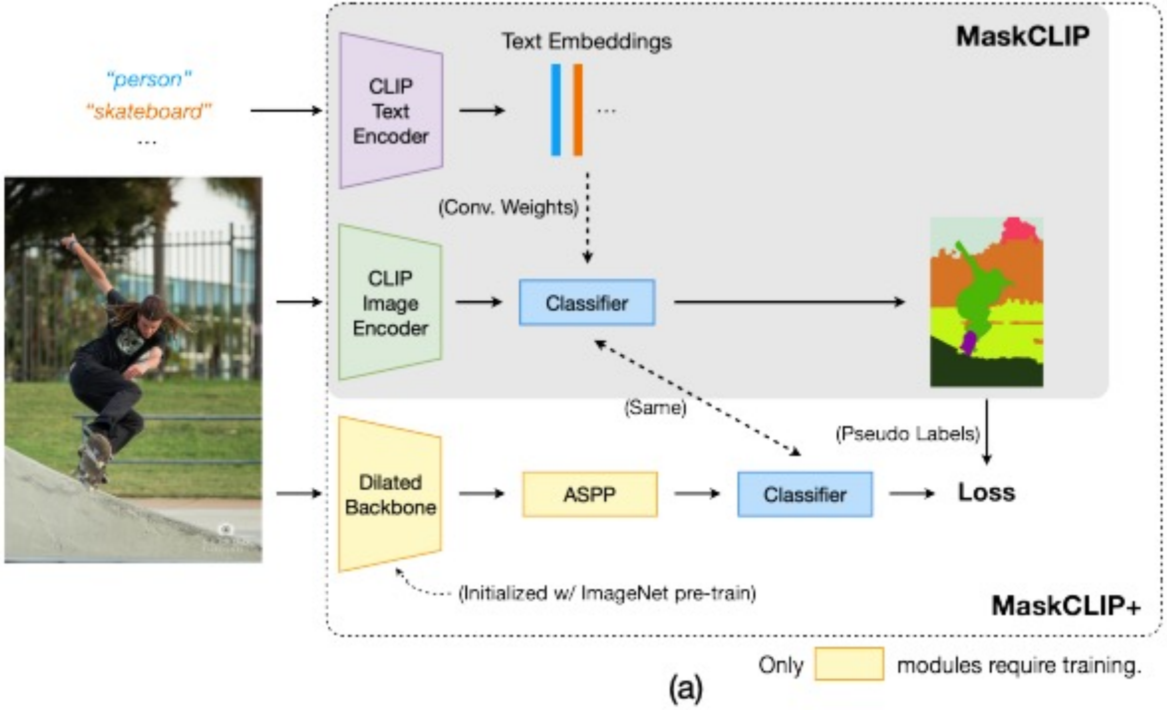
Possesses the ability to segment **open vocabulary classes** as well as **fine-grained classes**

Need advanced architectures tailored for segmentation

Demonstrates great robustness to **natural distribution shift** and **input corruptions**

Methodology

- MaskCLIP+
  - MaskCLIP-guided learning and self-training



- DeepLabV2 is used for target network (Backbone)
- Use the predictions of MaskCLIP as pseudo GT
- Use same classifier with that of MaskCLIP to preserve the ability for open vocabulary prediction
- When MaskCLIP+ outperform MaskCLIP, MaskCLIP+ generate pseudo labels for itself (self-training)

Experiments

- Quantitative Evaluation – Annotation-free segmentation

Table 1: **Annotation-free segmentation (mIoU).** **(a)** We evaluate the performance of MaskCLIP(+) on two standard datasets. For Pascal Context, we ignore the evaluation on the background class. The target model of MaskCLIP+ is Deeplabv2-ResNet101. KS and PD denote key smoothing and prompt denoising respectively. And they are not used in MaskCLIP+. **(b)** We test the robustness of MaskCLIP on Pascal Context under various types of corruption

(a)

Method	CLIP	PASCAL Context	COCO Stuff
Baseline	r50	8.3	4.6
	vit16	9.0	4.3
MaskCLIP	r50	18.5	10.2
	+KS	21.0	12.4
	+PD	19.0	10.8
	+KS+PD	21.8	12.8
	vit16	21.7	12.5
	+KS	23.9	13.8
	+PD	23.1	13.2
	+KS+PD	25.5	14.6
MaskCLIP+	r50	23.9	13.6
	vit16	31.1	18.0

(b)

Corruption	**level 1**		**level 5**	
	r50	vit16	r50	vit16
None	18.5	21.7	18.5	21.7
Gaussian Noise	13.7	19.6	2.1	6.8
Shot Noise	14.0	19.6	2.4	7.5
Impulse Noise	9.9	17.3	2.1	7.2
Speckle Noise	15.1	20.0	5.6	11.4
Gaussian Blur	17.4	21.6	4.3	14.1
Defocus Blur	15.7	20.8	6.6	15.5
Spatter	17.1	20.5	7.8	12.2
JPEG	15.7	20.8	7.6	14.5

Experiments

- Quantitative Evaluation – Zero-shot setting

Table 2: **Zero-shot segmentation performances.** ST stands for self-training. mIoU(U) denotes mIoU of the unseen classes. SPNet-C is the SPNet with calibration. On PASCAL Context, all methods use DeepLabv3+-ResNet101 as the backbone segmentation model and the rest two datasets use DeepLabv2-ResNet101. For MaskCLIP+, CLIP-ResNet-50 is used to generate pseudo labels

Method	PASCAL-VOC			COCO-Stuff			PASCAL-Context		
	mIoU(U)	mIoU	hIoU	mIoU(U)	mIoU	hIoU	mIoU(U)	mIoU	hIoU
Inductive									
SPNet	0.0	56.9	0.0	0.7	31.6	1.4	·	·	·
SPNet-C	15.6	63.2	26.1	8.7	32.8	14.0	·	·	·
ZS3Net	17.7	61.6	28.7	9.5	33.3	15.0	12.7	19.4	15.8
CaGNet	26.6	65.5	39.7	12.2	33.5	18.2	18.5	23.2	21.2
Transductive									
SPNet+ST	25.8	64.8	38.8	26.9	34.0	30.3	·	·	·
ZS3Net+ST	21.2	63.0	33.3	10.6	33.7	16.2	20.7	26.0	23.4
CaGNet+ST	30.3	65.8	43.7	13.4	33.7	19.5	·	·	·
STRICT	35.6	70.9	49.8	30.3	34.9	32.6	·	·	·
MaskCLIP+	**86.1**	**88.1**	**87.4**	**54.7**	**39.6**	**45.0**	**66.7**	**48.1**	**53.3**
	+50.5	+17.2	+37.6	+24.4	+4.7	+12.4	+46.0	+22.1	+29.9
Fully Sup.	·	88.2	·	·	39.9	·	·	48.2	·

Experiments

- Qualitative Evaluation – PASCAL Context

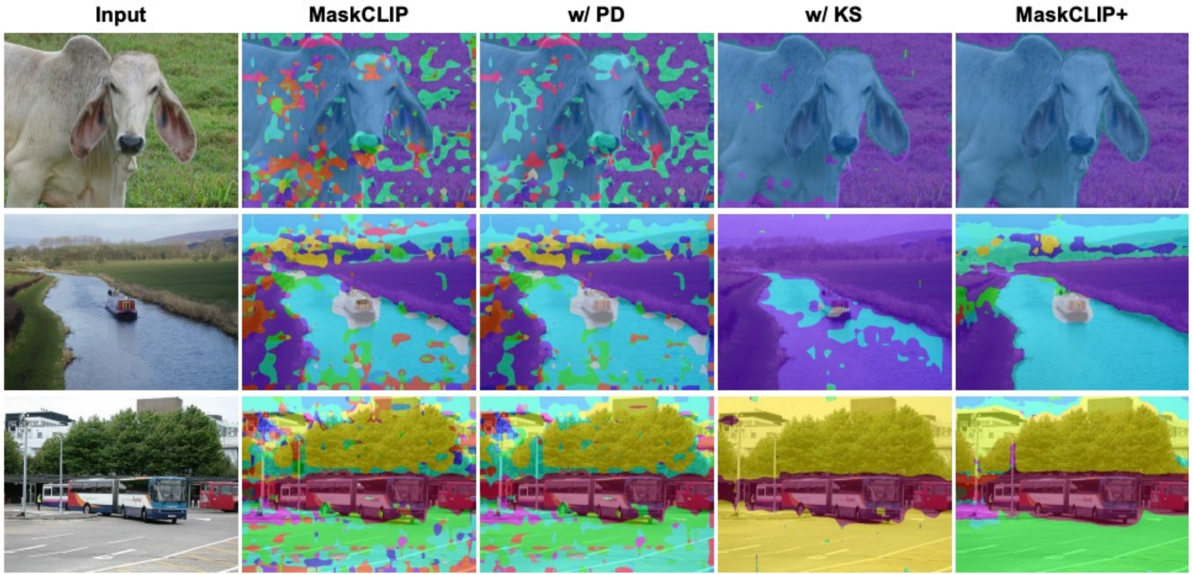


Fig. 3: **Qualitative results on PASCAL Context.** Here all results are obtained **without** any annotation. PD and KS refer to prompt denoising and key smoothing respectively. With PD, we can see some distraction classes are removed. KS is more aggressive. Its outputs are much less noisy but are dominated by a small number of classes. Finally, MaskCLIP+ yields the best results

Experiments

- Qualitative Evaluation – Web Images

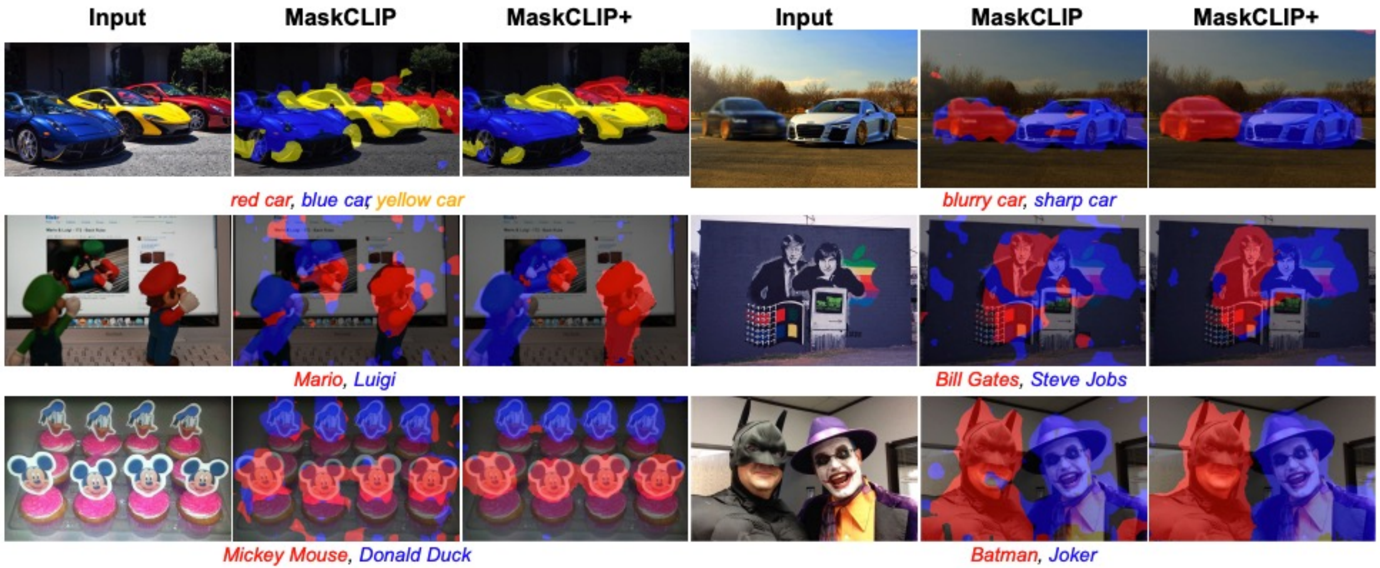


Fig. 4: **Qualitative results on Web images.** Here we show the segmentation results of MaskCLIP and MaskCLIP+ on various **unseen classes**, including fine-grained classes such as cars in different colors/imagery properties, celebrities, and animation characters. All results are obtained **without** any annotation

Experiments

- Ablation Study

Table 3: **Ablations of MaskCLIP+.** Experiments are performed on the PASCAL VOC dataset under the zero-shot setting

Method	mIoU(S)	mIoU(U)	mIoU	hIoU
Adapted DeepLabv2	83.4	3.7	63.5	7.0
+ MaskCLIP-Guided	**89.5**	72.8	85.3	80.3
+ Self-Training	88.8	**86.1**	**88.1**	**87.4**

Thank you

Any question?