DenseCLIP: Language-Guided Dense Prediction with Context-Aware Prompting

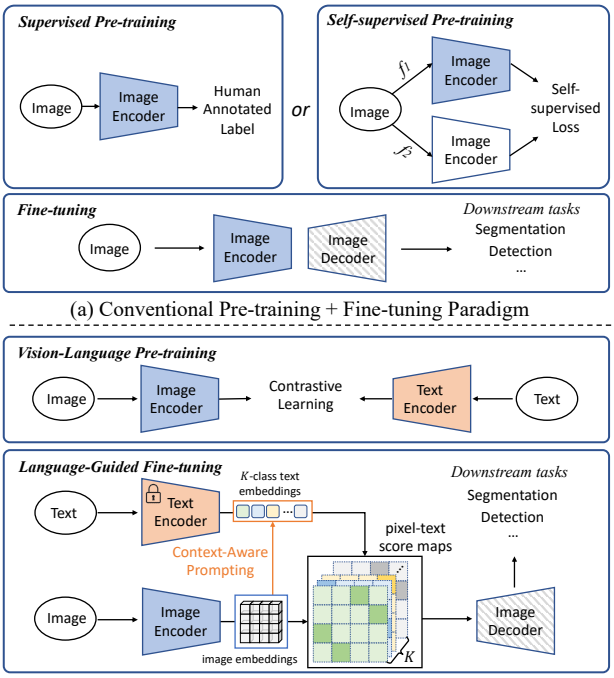
Yongming Rao*,[1], Wenliang Zhao*,[1], Guangyi Chen[1], Yansong Tang[1],
Zheng Zhu[1], Guan Huang[2], Jie Zhou[1], Jiwen Lu†,[1]
[1]Tsinghua University [2]PhiGent Robotics

Abstract

Recent progress has shown that large-scale pre-training using contrastive image-text pairs can be a promising alternative for high-quality visual representation learning from natural language supervision. Benefiting from a broader source of supervision, this new paradigm exhibits impressive transferability to downstream classification tasks and datasets. However, the problem of transferring the knowledge learned from image-text pairs to more complex dense prediction tasks has barely been visited. In this work, we present a new framework for dense prediction by implicitly and explicitly leveraging the pre-trained knowledge from CLIP. Specifically, we convert the original image-text matching problem in CLIP to a pixel-text matching problem and use the pixel-text score maps to guide the learning of dense prediction models. By further using the contextual information from the image to prompt the language model, we are able to facilitate our model to better exploit the pre-trained knowledge. Our method is model-agnostic, which can be applied to arbitrary dense prediction systems and various pre-trained visual backbones including both CLIP models and ImageNet pre-trained models. Extensive experiments demonstrate the superior performance of our methods on semantic segmentation, object detection, and instance segmentation tasks. Code is available at https://github.com/raoyongming/DenseCLIP.

1. Introduction

The "*pre-training + fine-tuning*" paradigm is recognized as one of the key discoveries that has largely pushed the state-of-the-art for various downstream computer vision tasks, including image classification [12, 22, 23], object detection [14, 35], semantic segmentation [6, 30], and action recognition [4]. Due to the high annotation and computation cost of the per-pixel prediction, pre-training is even more critical for dense prediction tasks. As illustrated in Figure 1 (a), the pre-training step is usually accomplished via supervised classification or self-supervised learning of the backbone model on large-scale datasets like ImageNet [11]. Then, a task-specific module like a detector or a segmentation decoder is added to the backbone and the whole model is fine-tuned on the target dataset with less training data [6, 35].



(a) Conventional Pre-training + Fine-tuning Paradigm

(b) DenseCLIP: CLIP Pre-training + Language-Guided Fine-tuning

Figure 1. Comparisons of the conventional "pre-training + fine-tuning" paradigm and our proposed *DenseCLIP*. The pre-training + fine-tuning paradigm directly applies the image pre-trained model as the initialization of encoder. Differently, DenseCLIP transfers the knowledge learned with image-text contrastive learning to dense prediction models by introducing a new pixel-text matching task and further using the contextual information from images to prompt pre-trained language model.

Different from conventional supervised and self-supervised pre-training methods only based on images, Contrastive Language-Image Pre-training (CLIP) [33] is a new framework to learn high-quality visual representation by exploring contrastive learning with large-scale noisy image-

*Equal contribution. †Corresponding author.

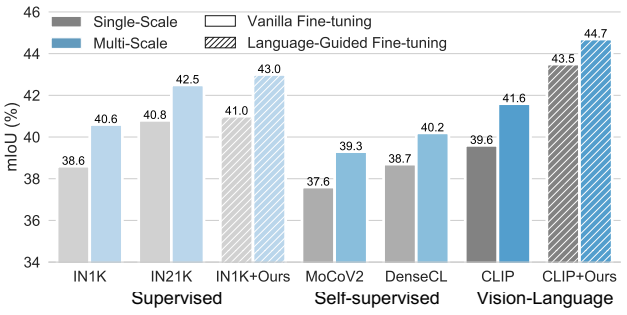


Figure 2. Results of different pre-training and fine-tuning strategies on the semantic segmentation task. We report the single-scale and multi-scale mIoU on ADE20K [59] of different pre-trained ResNet-50 [18] models, including supervised ImageNet1K [11] (IN1K) and ImageNet21K [11, 36] (IN21K), self-supervised MoCoV2 [9] and DenseCL [43], and vision-language model CLIP. Equipped with DenseCLIP, we show that large-scale vision-language pre-training can substantially improve the dense prediction performance (+4.9%/+4.1%) over the commonly used ImageNet pre-training.

text pairs. By exploiting the semantic relationships between the images and the associated texts, this new framework benefits from rich and semantic level supervision from texts while enjoying a broader and cheaper source of data. Thanks to the language supervision, models pre-trained via CLIP achieve impressive results on various visual classification tasks with no or very limited annotations [13, 41, 60].

Very recently, several efforts have been made to adopt the prompt engineering from NLP community [28] to better transfer the CLIP models to the downstream visual classification tasks. Several learning-based prompting methods [13, 51, 56, 60] are proposed to modify the output of the language model to better adapt to the new tasks. However, they mainly focus on transferring the CLIP model to classification tasks by performing image-text matching, which is much close to the original pre-training task. The problem of transferring the knowledge learning from image-text pairs to more complex dense prediction tasks and a more generic setting has barely been visited.

In this paper, we study how to fine-tune the pre-trained CLIP models to dense prediction tasks. Compared to conventional ImageNet pre-trained models, one distinct challenge is the gap between the upstream contrastive pre-training task and the downstream per-pixel prediction task, where the former involves instance-level representation of both images and texts, and the latter is only based on the visual information at the pixel level. To tackle this problem, we present a new language-guided dense prediction framework named *DenseCLIP*. As shown in Figure 1 (b), it is designed for various *Dense* prediction tasks by implicitly and explicitly leveraging the pre-trained knowledge from *CLIP* models. An implicit way to exploit the pre-trained knowledge is to directly fine-tune the models on the downstream datasets. Our results show that the CLIP models can outperform the conventional ImageNet pre-trained models with some modifications on hyper-parameters (see the CLIP result in Figure 2). But the straightforward way cannot fully exploit the potential of the CLIP models. Inspired by the original contrastive learning framework in CLIP, we propose to convert the original image-text matching problem in CLIP to a pixel-text matching problem and use the pixel-text score maps to guide the learning of dense prediction models explicitly. By further using the contextual information from the image to prompt the language model with a Transformer [40] module, we are able to facilitate our model to better exploit the pre-trained knowledge by optimizing the text embeddings.

Our method can be a plug-and-play module to improve the fine-tuning of CLIP pre-trained models on off-the-shelf dense prediction methods and tasks. By applying our method to the popular semantic segmentation framework semantic FPN [21] on the challenging ADE20K [59] dataset, we exhibit +4.9%, +4.7% and +2.3% mIoU improvement compared over ImageNet pre-trained models and +3.9%, +2.4% and +1.2% mIoU improvement compared to vanilla fine-tuning of a CLIP models based on ResNet-50, ResNet-101 [18] and ViT-B [12] respectively. We also observe significant improvements in object detection and instance segmentation tasks. Notably, we show a ResNet-101 model equipped with our method and a lightweight semantic FPN decoder can achieve 46.5% mIoU on ADE20K, which outperforms state-of-the-art solutions like DeepLabV3+ [7] and UperNet [45] with only 1/3 computation.

Moreover, our framework can also be applied to *any* backbone models by using the pre-trained language model to guide the training of dense prediction tasks. We observe significant improvements by applying DenseCLIP to ImageNet pre-trained ResNets [18] and recent Swin Transformers [29] with slight computation overhead. We expect our method to be a new and generic paradigm to improve dense prediction models with guidance from pre-trained language models.

2. Related Work

Pre-training and fine-tuning. The revolution of computer vision in the past decade has been driven by the "pre-training + fine-tuning" paradigm. Specifically, it first pre-trains models on large-scale datasets (*e.g.,* ImageNet [11], JFT [38], Kinetics [4], *etc.*) in a supervised learning [12, 18, 29, 34] or self-supervised learning manner [3, 8, 15, 16], and then fine-tunes the models on various downstream tasks. In NLP community, this framework has also been similarly and widely used [2] and recently evolves into a prompt paradigm [28], in which downstream tasks are reformulated to simulate the solved tasks in original pretraining process. Inspired by these works, we explore to transfer the knowledge in large-scale vision-language pre-trained models to the downstream dense prediction tasks.

Vision-language models. There have been a series of works on the interaction of computer vision and natural language processing fields, *e.g.,* text-to-image retrieval [44], image caption [48], visual question answering [1], referring segmentation [19,49,50] and so on. Among these works, vision-language pre-training has attracted growing attention during the past few years [24, 32, 37]. As a milestone, Radford *et al*. devise a large-scale pretraining model, named CLIP [33], which employs a contrastive learning strategy on a huge amount of image-text pairs, and shows impressive transferable ability over 30 classification datasets. Motivated by this work, a number of follow-ups have been proposed to improve the training strategy (*e.g.,* CoOp [60], CLIP-Adapter [13], Tip-adapter [56]) or apply it to other domains (*e.g.,* ActionCLIP [41]). However, there are very few attempts on performing dense prediction tasks via the CLIP model. The work most related to ours is CPT [51], which reformulates dense predictions into a fill-in-the-blank problem by jointly marking co-referential parts of both image and text in color. Differently, we consider a standard dense prediction setting in this paper, where we use pixel-text relationships to guide the training of dense prediction models and further optimize language embedding with image context with a context-aware prompting method.

Dense prediction. Compared with conventional instance-level classification problem, dense prediction tasks (*e.g.,* semantic segmentation [59], instance segmentation [27], object detection [27]) are more challenging as they requires to model the finer-grained representation at the pixel level or region level. Following the "pre-training + fine-tuning" paradigm, previous literatures have developed various dense prediction models like FCN [30], PSPNet [57], FPN [25], UperNet [45], and many others. To alleviate the heavy annotation cost in previous supervised pre-training settings, a number of self-supervised pre-training approaches have been proposed for dense prediction [3, 43, 46, 47]. Orthogonal to these prior arts, we introduce a new fine-tuning strategy that leverages the knowledge in the large-scale vision-language pre-trained model and uses the language information to guide the learning process.

3. Approach

3.1. Preliminaries: Overview of CLIP

We begin by reviewing the Contrastive Language-Image Pre-training (CLIP) [33] framework to illustrate the motivation of our method. CLIP consists of two encoders, including an image encoder (ResNet [18] or ViT [12]) and a text encoder (Transformer [40]). The goal of CLIP is to align the embedding spaces of visual and language during pre-training through a contrastive objective.

To learn more transferable pre-trained knowledge, CLIP collects 400 million image-text pairs for model training. To transfer knowledge of CLIP to downstream classification task, a simple yet effective way [33] is to construct a set of text prompts based on a template such as "`a photo of a [CLS]`.", where `[CLS]` can be replaced by the actual class names. Then given an image, one can use CLIP to compute the similarities between the image and the text prompts in the embedding space and the class with the highest score is regarded as the final prediction. Recently, several works [13, 60] have shown CLIP can obtain strong classification performance with few examples. Therefore, it raises an interesting question: *whether the impressive ability of CLIP can be transferred to more complex vision tasks like dense prediction?*

However, the extension is nontrivial. Firstly, how to leverage the visual-language pre-trained model in dense prediction tasks is a barely visited question. Although a simple solution is to only use the image encoder like a pre-trained 2D backbone, we argue that the language priors contained in the text encoder are also of great importance. Secondly, unlike the classification considered in [13, 60], transferring the knowledge from CLIP to dense prediction is more difficult due to the substantial gap between the upstream contrastive pre-training task and the downstream per-pixel prediction task, where the former considers instance-level representation of both images and texts, and the latter is only based on the visual information but expects pixel-level outputs.

3.2. Language-Guided Dense Prediction

To solve the above issues, we propose our language-guided dense prediction framework, which can better leverage the language priors in CLIP pre-trained models. The pipeline of our framework is shown in Figure 3. One of our important findings is that apart from the global image feature, we can also extract a language-compatible feature map from the last layer of the CLIP image encoder. To show this, we start by describing the architecture of the CLIP image encoder in detail. Take the ResNet [18] encoder for example, there are 4 stages in total and we denote the feature maps as $\{\mathbf{x}_i\}_{i=1}^4$. Different from the original ResNet [18], CLIP makes a small modification [33] by adding an attention pooling layer. Specifically, CLIP first performs global average pooling to $\mathbf{x}_4 \in \mathbb{R}^{H_4 W_4 \times C}$ to obtain a global feature $\bar{\mathbf{x}}_4 \in \mathbb{R}^{1 \times C}$, where H_4, W_4, C are the height, width and the number of channels of the feature maps from the 4-th stage of the backbone. The concatenated features $[\bar{\mathbf{x}}_4, \mathbf{x}_4]$ are then fed into an multi-head self-attention layer [40] (MHSA):

$$[\bar{\mathbf{z}}, \mathbf{z}] = \mathrm{MHSA}([\bar{\mathbf{x}}_4, \mathbf{x}_4]). \tag{1}$$

In the standard training process of CLIP, the global feature $\bar{\mathbf{z}}$ is used as the output of the image encoder while the other outputs $\mathbf{z}$ are usually neglected. However, we find $\mathbf{z}$ has two interesting properties: (1) $\mathbf{z}$ still retains sufficient spatial information thus can serve as a feature map. (2) since the

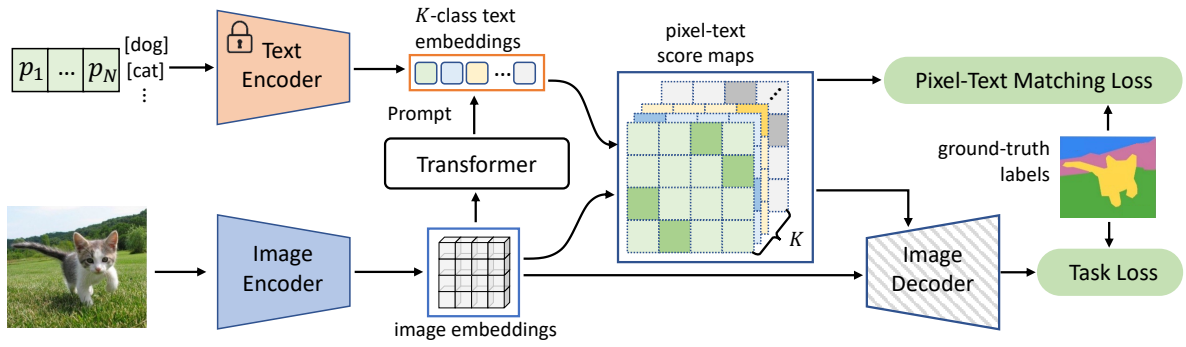


Figure 3. The overall framework of DenseCLIP. DenseCLIP first extracts the image embeddings and K-class text embeddings, and then calculates pixel-text score maps to convert the original image-text matching problem in CLIP to pixel-text matching for dense prediction. These score maps are fed into decoder and also supervised using the ground-truth labels. To better exploit the pre-trained knowledge, DenseCLIP uses the contextual information in images to prompt the language model with a Transformer module.

MHSA is symmetric to each input element, $\mathbf{z}$ might behave similarly to $\bar{\mathbf{z}}$, which aligns well with the language features. Based on the above observations, we can use $\mathbf{z}$ as a language-compatible feature map. It is also noted that for architectures like ViT [12], $\mathbf{z}$ can be obtained similarly by excluding the class token of outputs.

To obtain the text features, we can construct text prompts from the template "`a photo of a [CLS]`." with K class names, and the use CLIP text encoder to extract the features as $\mathbf{t} \in \mathbb{R}^{K \times C}$. We then compute the pixel-text score maps using the language-compatible feature map $\mathbf{z}$ and the text features $\mathbf{t}$ by:

$$\mathbf{s} = \hat{\mathbf{z}}\hat{\mathbf{t}}^{\top}, \quad \mathbf{s} \in \mathbb{R}^{H_4 W_4 \times K}, \tag{2}$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{t}}$ are the ℓ_2 normalized version of $\mathbf{z}$ and $\mathbf{t}$ along the channel dimension. The score maps characterize the results of pixel-text matching, which is one of the most crucial ingredients in our framework. Firstly, the score maps can be viewed as segmentation results with a lower resolution, and thus we can use them to compute an auxiliary segmentation loss. Secondly, we can concatenate the score maps to the last feature map to explicitly incorporate language priors, *i.e.*, $\mathbf{x}'_4 = [\mathbf{x}_4, \mathbf{s}] \in \mathbb{R}^{H_4 W_4 \times (C+K)}$. Our framework is model-agnostic because the modified feature maps can be directly used as usual in segmentation or detection with some minor modifications (*e.g.*, the input dimension of FPN [25]).

3.3. Context-Aware Prompting

Previous efforts [13, 60] have already proved that mitigating the domain gaps in visual or language can significantly improve the performance of CLIP models on downstream tasks. Therefore, instead of using the vanilla human predefined templates, we seek for other methods to improve the text features $\mathbf{t}$.

Language-domain prompting. Different from the original CLIP that uses human-designed templates like "`a photo of a [CLS]`." as text prompts, CoOp [60] introduces learnable textual contexts to achieve better transferability in downstream classification tasks by directly optimizing the contexts using back-propagation. Inspired by CoOp [60], we also use learnable textual contexts in our framework as a baseline, which only includes language-domain prompting. The input of the text encoder then becomes:

$$[\mathbf{p}, \mathbf{e}_k], \quad 1 \le k \le K, \tag{3}$$

where $\mathbf{p} \in \mathbb{R}^{N \times C}$ are the learnable textual contexts and $\mathbf{e}_k \in \mathbb{R}^{C}$ is the embedding for the name of the k-th class.

Vision-to-language prompting. Including descriptions of visual contexts can make the text more accurate. For example, "`a photo of a cat in the grass`." is more accurate than "`a photo of a cat`.". Therefore, we investigate how to use visual contexts to refine the text features. Generally, we can use the cross-attention mechanism in Transformer decoder [40] to model the interactions between vision and language.

We propose two different strategies of context-aware prompting, which is shown in Figure 4. The first strategy we consider is the pre-language-model prompting, or *pre-model prompting* for short. We pass the features $[\bar{\mathbf{z}}, \mathbf{z}]$ to a Transformer decoder to encode visual contexts:

$$\mathbf{v}_{\text{pre}} = \text{TransDecoder}(\mathbf{q}, [\bar{\mathbf{z}}, \mathbf{z}]), \tag{4}$$

where $\mathbf{q} \in \mathbb{R}^{N \times C}$ are a set of learnable queries and $\mathbf{v}_{\text{pre}} \in \mathbb{R}^{N \times C}$ are the extracted visual contexts. We replace the $\mathbf{p}$ in Equation (3) by the visual contexts $\mathbf{v}$ to form the input of the text encoder. Since the input of the text encoder is modified, we refer to this version as *pre-model prompting*.

Another choice is to refine the text features after the text encoder, namely *post-model prompting*. In this variant, we use CoOp [60] to generate text features and directly use them as the queries of the Transformer decoder:

$$\mathbf{v}_{\text{post}} = \text{TransDecoder}(\mathbf{t}, [\bar{\mathbf{z}}, \mathbf{z}]). \tag{5}$$

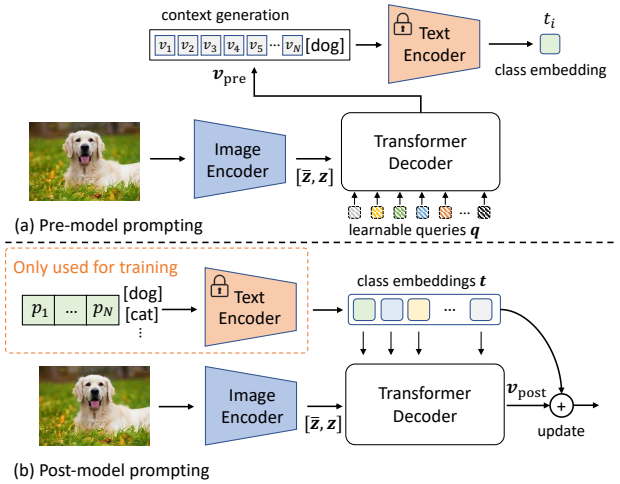


Figure 4. **Two different strategies of context-aware prompting.** The pre-model prompting directly uses the image contexts to generate the desired text inputs, while post-model prompting refines the class embedding instead.

This implementation encourage the text features to find most related visual clues. We then update the text features through a residual connection:

$$\mathbf{t} \leftarrow \mathbf{t} + \boldsymbol{\gamma}\mathbf{v}_{\text{post}}, \tag{6}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^C$ is a learnable parameter to control the scaling of the residual. $\boldsymbol{\gamma}$ is initialized with very small values (*e.g.*, 10^{-4}) to maximally preserve the language priors from the text features.

Although the two variants target the same goal, we prefer the post-model prompting for mainly two reasons: (1) The post-model prompting is efficient. The pre-model prompting requires extra forward passes of the text encoder during inference since its input is dependent on the image. In the case of post-model prompting, we can store the extracted text features after training and thus can reduce the overhead brought by the text encoder during inference. (2) Our empirical results show the post-model prompting can achieve better performance than pre-model prompting.

3.4. Instantiations

Semantic segmentation. As discussed in Section 3.2, our framework is model-agnostic and can be applied to any dense prediction pipelines. Moreover, we propose to use an auxiliary objective to make better use of our pixel-text score maps in segmentation. Since the score maps $\mathbf{s} \in \mathbb{R}^{H_4W_4 \times K}$ can be viewed as smaller segmentation results, we therefore compute a segmentation loss on it:

$$\mathcal{L}^{\text{seg}}_{\text{aux}} = \text{CrossEntropy}(\text{Softmax}(\mathbf{s}/\tau), \mathbf{y}), \tag{7}$$

where $\tau = 0.07$ is a temperature coefficient following [16] and $\mathbf{y} \in \{1, \dots, K\}^{H_4W_4}$ is the ground truth label. The auxiliary segmentation loss can help the feature map to recover its locality faster, which is beneficial to dense prediction tasks for both segmentation and detection.

Object detection & instance segmentation. In this case, we do not have ground truth segmentation labels. To construct a similar auxiliary loss as in segmentation, we use the bounding box and the label to build a binary target $\tilde{\mathbf{y}} \in \{0, 1\}^{H_4W_4 \times K}$. The auxiliary objective can be defined as a binary cross-entropy loss:

$$\mathcal{L}^{\text{det}}_{\text{aux}} = \text{BinaryCrossEntropy}(\text{Sigmoid}(\mathbf{s}/\tau), \tilde{\mathbf{y}}). \tag{8}$$

Applications to *any* backbone models. Another interesting usage of our framework is that we can replace the image encoder of CLIP with *any* backbones (*e.g.*, ImageNet pre-trained models and self-supervised models). Although there might be no strong relation between the outputs of the visual backbone and the text encoder, the backbone can learn better and faster with language guidance. In other words, we can leverage the language priors from the pre-trained text encoder to improve the performance of any pre-trained image backbone, which makes DenseCLIP a more generic framework to improve dense prediction with the natural language priors learned from large-scale pre-training.

4. Experiments

To evaluate the effectiveness of our DenseCLIP, we conduct extensive experiments on dense prediction tasks including semantic segmentation, object detection and instance segmentation. The following subsections describe the details of the experiments, results and analyses.

4.1. Semantic Segmentation

Setups. We start by evaluating our DenseCLIP on ADE20K [59], a challenging large-scale semantic segmentation dataset that covers a broad range of 150 categories. ADE20K contains 20K images for training and 2K images for validation. Following common practice [20, 45], we report the mIoU on the validation set. For fair comparisons, we also include the FLOPs and the number of parameters.

Implementation details. We experiment with the popular Semantic FPN [21] framework to evaluate our DenseCLIP. Specifically, we apply the pre-trained image encoder of the CLIP as the segmentation backbone, and directly use the Semantic FPN [21] as the decoder. We consider three kinds of image backbones including ResNet-50 [18], ResNet-101 [18], and ViT-B [12]. For language-domain prompting, we use a context length of 8. The Transformer decoder to extract visual contexts consists of 6 layers and we set the number of heads as 4. We fix the text encoder during training to preserve the natural language knowledge learned from large-scale pre-training. To reduce the computational costs, we project both the image embeddings and the text

Table 1. **Semantic segmentation results on ADE20K**. We compare the performance of DenseCLIP and existing methods when using the same backbone. We report the mIoU of both single-scale and multi-scale testing, the FLOPs and the number of parameters. The FLOPs are measured with 1024×1024 input using the `fvcore` library. The results show that our DenseCLIP outperforms other methods by large margins with much lower complexity. Our models and our baselines that are trained using identical settings are highlighted in gray.

Backbone	Method	Pre-train	mIoU (SS)	mIoU (MS)	GFLOPs	Params (M)
ResNet-50	FCN [30]	ImageNet	36.1	38.1	793.3	49.6
	EncNet [55]	ImageNet	40.1	41.7	565.6	36.1
	PSPNet [57]	ImageNet	41.1	41.9	716.2	49.1
	CCNet [20]	ImageNet	42.1	43.1	804.0	49.9
	DeeplabV3+ [7]	ImageNet	42.7	43.8	711.5	43.7
	UperNet [45]	ImageNet	42.1	42.8	953.2	66.5
	DNL [52]	ImageNet	41.9	43.0	939.3	50.1
	Semantic FPN [21]	ImageNet	38.6	40.6	227.1	31.0
	CLIP + Semantic FPN	CLIP	39.6	41.6	248.8	31.0
	DenseCLIP + Semantic FPN	CLIP	**43.5**	**44.7**	269.2	50.3
ResNet-101	FCN [30]	ImageNet	39.9	41.4	1104.4	68.6
	EncNet [55]	ImageNet	42.6	44.7	876.8	55.1
	PSPNet [57]	ImageNet	43.6	44.4	1027.4	68.1
	CCNet [20]	ImageNet	44.0	45.2	1115.2	68.9
	DeeplabV3+ [7]	ImageNet	44.6	46.1	1022.7	62.7
	UperNet [45]	ImageNet	43.8	44.8	1031.0	85.5
	OCRNet [54]	ImageNet	45.3	-	923.9	55.5
	DNL [52]	ImageNet	44.3	45.8	1250.5	69.1
	Semantic FPN [21]	ImageNet	40.4	42.3	304.9	50.0
	CLIP + Semantic FPN	CLIP	42.7	44.3	326.6	50.0
	DenseCLIP + Semantic FPN	CLIP	**45.1**	**46.5**	346.3	67.8
ViT-B	SETR-MLA-DeiT [58]	ImageNet	46.2	47.7	-	-
	Semantic FPN [21]	ImageNet	48.3	50.9	1037.4	100.8
	Semantic FPN [21]	ImageNet-21K	49.1	50.4	1037.4	100.8
	CLIP + Semantic FPN	CLIP	49.4	50.3	1037.4	100.8
	DenseCLIP + Semantic FPN	CLIP	**50.6**	**51.3**	1043.1	105.3

Table 2. **Ablation study.** We demonstrate that performing post-model vision-to-language prompting can yield the better performance with fewer extra FLOPs and parameters.

Pre-train	Language Prompt	V→L Prompt		mIoU (%)	FLOPs (G)	Params (M)
		pre	post			
ImageNet				38.6	227	31.0
CLIP				39.6 (+1.0)	249	31.0
CLIP	✓			42.1 (+3.5)	269	46.5
CLIP	✓	✓		42.9 (+4.3)	368	116.9
CLIP	✓		✓	**43.5** (+4.9)	269	50.2

embeddings to a lower dim (256) before the Transformer module. We empirically find that directly fine-tuning CLIP models to dense prediction with the default training strategies in [10] will lead to unsatisfactory results (only 21.9% mIoU on ADE20K, which is 15.6% lower than its ImageNet pre-trained counterpart). Therefore, two key modifications are made compared to the default configurations: (1) we use AdamW [31] instead of the default SGD inspired by recent progress in vision Transformers [29, 39, 42]; (2) to better preserve the pre-trained weights, we set the learning rate of the image encoder as $1/10$ of the other parameters. We also adopt the above training strategies to our baselines in ablation studies for fair comparisons (+1.1% mIoU over the ImageNet pre-trained ResNet-50 with the default settings in [10]).

Main results. We report the semantic segmentation results of our DenseCLIP with three different backbones on ADE20K in Table 1. We include the FLOPs, the number of parameters, and the mIoU in both single-scale (SS) and multi-scale (MS) testings. The experiments results show that for the same backbone, our DenseCLIP with a simple Semantic FPN can outperform the state-of-the-art methods that use more sophisticated decoders by large margins. Unlike previous works that use dilated backbones (ResNet-D8 [20, 53, 54, 57]), the ResNet encoder in DenseCLIP is more close to standard ResNet thus our DenseCLIP has much fewer FLOPs. Besides, our DenseCLIP is +4.9%, +4.7%, and +2.3% mIoU (SS) higher than the original ImageNet pre-trained baselines on ResNet-50, ResNet-101 and ViT-B backbones with acceptable extra computation cost. DenseCLIP is also +3.9%, +2.4%, and +1.2% mIoU higher than the vanilla fine-tuning strategy (CLIP + Semantic FPN).

Ablation studies. To further demonstrate the effects of different components of our DenseCLIP, we perform detailed ablation studies with the ResNet-50 [18] backbone and the results are shown in Table 2. Firstly, we show by adopting a better training strategy aforementioned the ResNet-50 baseline we implemented has a higher mIoU than [10] (38.6% *vs.* 37.5%). Secondly, we find that CLIP pre-trained ResNet-50 outperforms the ImageNet pre-trained one by 1%, which indicates that large-scale vision language pre-trained model can be better transferred to downstream vision tasks. To

better leverage the language priors, we adopt our language-guided with language-domain prompt and witness a significant performance boost (+2.5% mIoU). Finally, we compare the two methods to perform vision-language prompting to incorporate visual contexts. We find both the pre-model and post-model prompting can improve the performance, while the post-model prompting is better and more computationally efficient. Therefore, we choose the post-model prompting as the default configuration in all the rest experiments.

Effects of language-guided pre-training and fine-tuning. We compare the performance on ADE20K of different pre-training and fine-tuning strategies to better reveal the potential of language-guided paradigm, which is shown in Figure 2. We consider supervised pre-training on ImageNet1K [11] and ImageNet21K [11, 36], self-supervised pre-training via MoCoV2 [16] and DenseCL [43], and the vision-language pre-training. We show that the vision-language pre-trained model (CLIP) can outperform ImageNet1K pre-trained model by vanilla fine-tuning. Furthermore, through the language-guided fine-tuning with context-aware prompting, our DenseCLIP surpasses even the ImageNet21K pre-trained model. These promising results demonstrate that language-priors can largely facilitate vision models in downstream dense prediction tasks.

4.2. Object Detection and Instance Segmentation

Setups. We also conduct experiments to apply our DenseCLIP to object detection and instance segmentation tasks on COCO [27], which contains 118K training images and 5K validation images. We adopt two widely used frameworks, RetinaNet [26] and Mask R-CNN [17]. Following [17], we report the standard AP, AP at IoU=0.5/0.75, and cross-scale AP. For Mask R-CNN, we report both the mAPs for object detection and instance segmentation since these two tasks are performed simultaneously.

Implementation details. For object detection, we adopt ResNet-50 and ResNet-101 as backbones. We train all the models for 12 epochs using AdamW optimizer with batch size 16 as in [5]. Specifically for RetinaNet, we witness a super large loss at the start of training that makes the model hard to converge. Therefore, we use gradient clipping with a max ℓ_2 norm of 0.1 to protect the pre-trained weights.

Results analysis. The results using the RetinaNet [26] and the Mask R-CNN [17] are summarized in Table 3 and Table 4, respectively. For object detection with RetinaNet, we compare DenseCLIP with ImageNet1K pretrained model and vanilla CLIP fine-tuning on detection task. One can observe that DenseCLIP outperforms the ImageNet1K pre-trained model by +1.5% and +2.6% AP. Meanwhile, it also improves the vanilla fine-tuning strategy by +0.9% and +0.6% AP on both ResNet-50 and ResNet-101 backbones.

Table 3. **Object detection on COCO val2017 using RetinaNet [26] framework.** We compare our DenseCLIP framework to the vanilla fine-tuning of ImageNet/CLIP pre-trained models. We find DenseCLIP can better make use of the language priors to facilitate better training.

Model	FLOPs (G)	Params (M)	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RN50-IN1K [18]	239	38	36.3	55.3	38.6	19.3	40.0	48.8
RN50-CLIP [33]	265	38	36.9	57.7	39.1	22.5	40.7	47.1
RN50-DenseCLIP	285	60	**37.8**	**59.9**	**40.0**	**24.8**	**42.0**	**47.9**
RN101-IN1K [18]	315	57	38.5	57.6	41.0	21.7	42.8	50.4
RN101-CLIP [33]	341	57	40.5	61.6	43.4	25.6	44.6	51.3
RN101-DenseCLIP	360	78	**41.1**	**63.4**	**44.1**	**26.9**	**45.5**	**52.4**

For Mask R-CNN, we observe that DenseCLIP achieves consistent improvement on both object detection and instance segmentation tasks within an affordable computational budget. Especially for instance segmentation, our DenseCLIP outperforms the ImageNet1K pre-trained model with +2.9% and +2.5% mask AP on both ResNet50 and ResNet101 backbones and also outperforms the vanilla fine-tuning strategy with +0.8% and +0.7% mask AP. The significant improvements of DenseCLIP on the instance segmentation task suggest that our pixel-text matching is conceptually suitable for segmentation.

4.3. DenseCLIP for Any Visual Backbone

Previous experiments have demonstrated the effectiveness of our DenseCLIP framework. However, since DenseCLIP is specifically designed to leverage the visual-language relation contained in the pre-trained CLIP models, the generalization ability of DenseCLIP might be somehow doubted: *Is DenseCLIP only suitable to CLIP image encoders?* To answer this question, we perform experiments to verify whether our DenseCLIP can also perform well with other backbones. The extension is actually straightforward: we can simply replace the CLIP image encoder with any given 2D pre-trained image model. Although there are no strong correlations between the feature maps of the new backbone and the text features output by the CLIP text encoder, we hypothesize that if we preserve the language priors by freezing the text encoder as before, the text encoder will guide the backbone to better adapt to downstream tasks.

To verify the above assumption, we choose two representative 2D models including ResNet [18], the most widely used CNN model, and Swin [29], the recent state-of-the-art vision Transformer. Following the standard setting in [21] and [29], we use the Semantic FPN [21] framework for ResNet models and the UperNet [45] framework for Swin models. The experimental results are summarized in Table 5, where we report the mIoU on ADE20K of both the single-scale and multi-scale testing. We demonstrate that our DenseCLIP can consistently improve all the baseline models

Table 4. **Object detection and instance segmentation results on COCO val2017 using Mask R-CNN [17] framework.** Our DenseCLIP outperforms ImageNet/CLIP pre-trained baseline models, especially on the instance segmentation task.

Model	FLOPs (G)	Params (M)	AP^b	AP^b_{50}	AP^b_{75}	AP^b_S	AP^b_M	AP^b_L	AP^m	AP^m_{50}	AP^m_{75}	AP^m_S	AP^m_M	AP^m_L
RN50-IN1K [18]	275	44	38.2	58.8	41.4	21.9	40.9	49.5	34.7	55.7	37.2	18.3	37.4	47.2
RN50-CLIP [33]	301	44	39.3	61.3	42.7	24.6	42.6	50.1	36.8	58.5	39.2	18.6	39.9	51.8
RN50-DenseCLIP	327	67	**40.2**	**63.2**	**43.9**	**26.3**	**44.2**	**51.0**	**37.6**	**60.2**	**39.8**	**20.8**	**40.7**	**53.7**
RN101-IN1K [18]	351	63	40.0	60.5	44.0	22.6	44.0	52.6	36.1	57.5	38.6	18.8	39.7	49.5
RN101-CLIP [33]	377	63	42.2	64.2	**46.5**	26.4	46.1	54.0	38.9	61.4	41.8	20.5	42.3	55.1
RN101-DenseCLIP	399	84	**42.6**	**65.1**	**46.5**	**27.7**	**46.5**	**54.2**	**39.6**	**62.4**	**42.4**	**21.4**	**43.0**	**56.2**

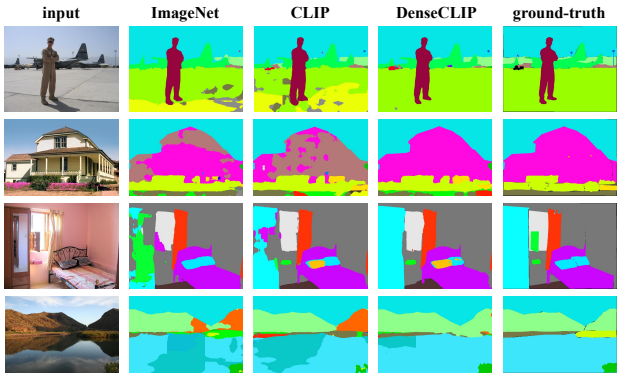


Figure 5. **Qualitative results on ADE20K.** We visualize the segmentation results on ADE20K validation set of our DenseCLIP based on ResNet-101 and two baseline models.

notably. Specifically, DenseCLIP can bring $\sim 2.5\%$ single-scale mIoU improvement for ResNet-50/101 with semantic FPN [21], and $\sim 0.8\%$ improvement for Swin-T/S with UperNet [45]. These results clearly show that our DenseCLIP can successfully guide any pre-trained 2D backbone by language priors to boost performance. Since the text encoder can be removed after training, our method provides a low-cost solution to improve arbitrary dense prediction models. Although these performances still lag behind our models with CLIP image encoders, the findings in this section provide a solution to generalize human knowledge learned from large-scale vision-language pre-training to a wider range of models. We expect this could be an interesting direction to connect vision and language researches in the future.

4.4. Visualization

To better demonstrate the superiority of DenseCLIP, we provide several qualitative results in Figure 5. We compare the segmentation maps of our method and the baselines and find DenseCLIP is better at identifying holistic objects.

5. Conclusion and Discussion

In this paper, we have presented a new framework, DenseCLIP, to transfer the knowledge from the vision-language pre-trained model (CLIP) to the downstream dense prediction tasks. DenseCLIP is a model-agnostic framework to use the pre-trained vision-language knowledge with the context-aware prompting strategy. The framework can be applied to various dense prediction tasks including semantic segmentation, object detection, and instance segmentation. We conducted extensive experiments to demonstrate the superior performance of our method.

Table 5. **Applying DenseCLIP to *any* backbone.** Image backbones (such as ImageNet pre-trained ResNet [18] and Swin [29]) equipped with our DenseCLIP benefit from the language priors and enjoy significant performance boost. We report mIoU on ADE20K dataset for both single-scale (SS) and multi-scale (MS) testing.

Decoder	Method	mIoU (SS) (%)	mIoU (MS) (%)
Semantic FPN [21]	RN50 [18]	38.6	40.6
	RN50 + DenseCLIP	**41.0**$_{(+2.4)}$	**43.0**$_{(+2.4)}$
	RN101 [18]	40.4	42.3
	RN101 + DenseCLIP	**43.0**$_{(+2.6)}$	**45.2**$_{(+2.9)}$
UperNet [45]	Swin-T [29]	44.5	45.8
	Swin-T + DenseCLIP	**45.4**$_{(+0.9)}$	**46.5**$_{(+0.7)}$
	Swin-S [29]	47.6	49.5
	Swin-S + DenseCLIP	**48.3**$_{(+0.7)}$	**49.7**$_{(+0.2)}$

Limitations & societal impact. Although our method has achieved substantial improvement in segmentation, we find the improvements on detection are not such significant. We conjecture that it is because the pre-trained CLIP image encoder lacks locality since there is no such constraint during the pre-training of CLIP while object-centered tasks can only provide less dense supervision. We believe DenseCLIP can be further improved by introducing the dense supervision during pre-training or better recovering the locality after pre-training. We develop a general method for dense prediction in this paper. Since our method is not for a specific application, it does not directly involve societal issues.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62125603, Grant U1813218, and in part by a grant from the Beijing Academy of Artificial Intelligence (BAAI).

References

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, pages 2425–2433, 2015. 3

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, and Melanie Subbiah et al. Language models are few-shot learners. In *NeurIPS*, 2020. 2

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2, 3

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308, 2017. 1, 2

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 7

[6] 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. 1

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pages 801–818, 2018. 2, 6

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020.

[9] Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *CoRR*, abs/2003.04297, 2020. 2

[10] MMSegmentation Contributors. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020. 6

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 1, 2, 7

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020. 1, 2, 3, 4, 5

[13] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021. 2, 3, 4, 11

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 1

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 2

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735, 2020. 2, 5, 7

[17] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 7, 8

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2, 3, 5, 6, 7, 8

[19] Ronghang Hu, Marcus Rohrbach, and Trevor Darrell. Segmentation from natural language expressions. In *ECCV*, pages 108–124, 2016. 3

[20] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, pages 603–612, 2019. 5, 6

[21] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, pages 6399–6408, 2019. 2, 5, 6, 7, 8

[22] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, pages 491–507. Springer, 2020. 1

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 1

[24] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L. Berg, Mohit Bansal, and Jingjing Liu. Less is more: Clipbert for video-and-language learning via sparse sampling. In *CVPR*, pages 7331–7341, 2021. 3

[25] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017. 3, 4

[26] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 7

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014. 3, 7

[28] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. 2

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2, 6, 7, 8

[30] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 1, 3, 6

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[32] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, pages 13–23, 2019. 3

[33] 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. 1, 3, 7, 8, 11

[34] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. In *NeurIPS*, 2021. 2

[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015. 1

[36] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021. 2, 7

[37] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: pre-training of generic visual-linguistic representations. In *ICLR*, 2020. 3

[38] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, pages 843–852, 2017. 2

[39] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357, 2021. 6

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 2, 3, 4

[41] Mengmeng Wang, Jiazheng Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *arXiv preprint arXiv:2109.08472*, 2021. 2, 3

[42] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021. 6

[43] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *CVPR*, pages 3024–3033, 2021. 2, 3, 7

[44] Zihao Wang, Xihui Liu, Hongsheng Li, Lu Sheng, Junjie Yan, Xiaogang Wang, and Jing Shao. CAMP: cross-modal adaptive message passing for text-image retrieval. In *ICCV*, pages 5763–5772, 2019. 3

[45] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, pages 418–434, 2018. 2, 3, 5, 6, 7, 8

[46] Zhenda Xie, Yutong Lin, Zhuliang Yao, Zheng Zhang, Qi Dai, Yue Cao, and Han Hu. Self-supervised learning with swin transformers. *arXiv preprint arXiv:2105.04553*, 2021. 3

[47] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *CVPR*, pages 16684–16693, 2021. 3

[48] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, volume 37, pages 2048–2057, 2015. 3

[49] Zhao Yang, Yansong Tang, Luca Bertinetto, Hengshuang Zhao, and Philip H.S. Torr. Hierarchical interaction network for video object segmentation from referring expressions. In *BMVC*, 2021. 3

[50] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip H.S. Torr. Lavt: Language-aware vision transformer for referring image segmentation. In *CVPR*, 2022. 3

[51] Yuan Yao, Ao Zhang, Zhengyan Zhang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv preprint arXiv:2109.11797*, 2021. 2, 3

[52] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *ECCV*, 2020. 6

[53] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 6

[54] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *ECCV*, 2020. 6

[55] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, pages 7151–7160, 2018. 6

[56] 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. *arXiv preprint arXiv:2111.03930*, 2021. 2, 3

[57] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017. 3, 6

[58] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 6

[59] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *IJCV*, 127(3):302–321, 2019. 2, 3, 5

[60] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021. 2, 3, 4, 11

Appendix: More Analysis

We provide more analyses of both the design of our model and the training strategies in detail in the section.

Effects of learning rate multipliers. As discussed in Section 4.1, we found that the optimal learning rate for CLIP models and conventional ImageNet pre-trained models are different. Here we further investigate the effects of learning rate multiplier for image encoder and text encoder in Table 6. We see both fixing the text encoder and using a lower learning rate for image encoder is beneficial to train the dense prediction model. Note that we observe a much lower performance (<30% mIoU) when directly fine-tuning CLIP models with $1.0\times$ learning rate for the image encoder, which suggests our language guided method can largely stabilize the training process and make the final results less sensitive to the learning rate configuration.

Table 6. Ablation study of the learning rate multiplier of text encoder and image encoder. We find freezing the text encoder and setting the lr multiplier of image encoder as 0.1 yields the best performance. The configuration used in our final models is highlighted in gray.

	text encoder	image encoder	mIoU (%)
lr multi	0.0	0.1	**43.5**
	0.0	1.0	42.6
	0.1	0.1	42.2

Effects of optimization of the textual contexts. Previous works [13, 60] on transferring CLIP models to downstream classification tasks have clearly shown the importance of adapting the textual contexts for different datasets and tasks. We show the effects of optimizing the textual contexts compared to the original prompting strategy proposed in [33] in Table 7. We see that although the learnable contexts will introduce additional computation during training (gradient computation for the text encoder), this strategy can bring notable improvement over the baseline. Therefore, we choose to add the learnable textual contexts for our models.

Table 7. Effects of optimization of the textual contexts. We compare the results of using the context optimization [60] with directly constructing textual prompts from human defined template and find learnable textual contexts can bring notable improvements. The configuration used in our final models is highlighted in gray.

Textual Context	mIoU (%)
`a photo of a [CLS].`	42.9
CoOp [60]	**43.5**

Effects of γ. Table 8 shows the effects of γ. We see a learnable γ initialized with small values can improve the final performance.

Table 8. Ablation study of the residual coefficient γ. The configuration used in our final models is highlighted in gray.

initial value of γ	γ learnable	mIoU (%)
10^{-4}	✗	42.6
10^{-4}	✓	**43.5**
1.0	✓	42.8