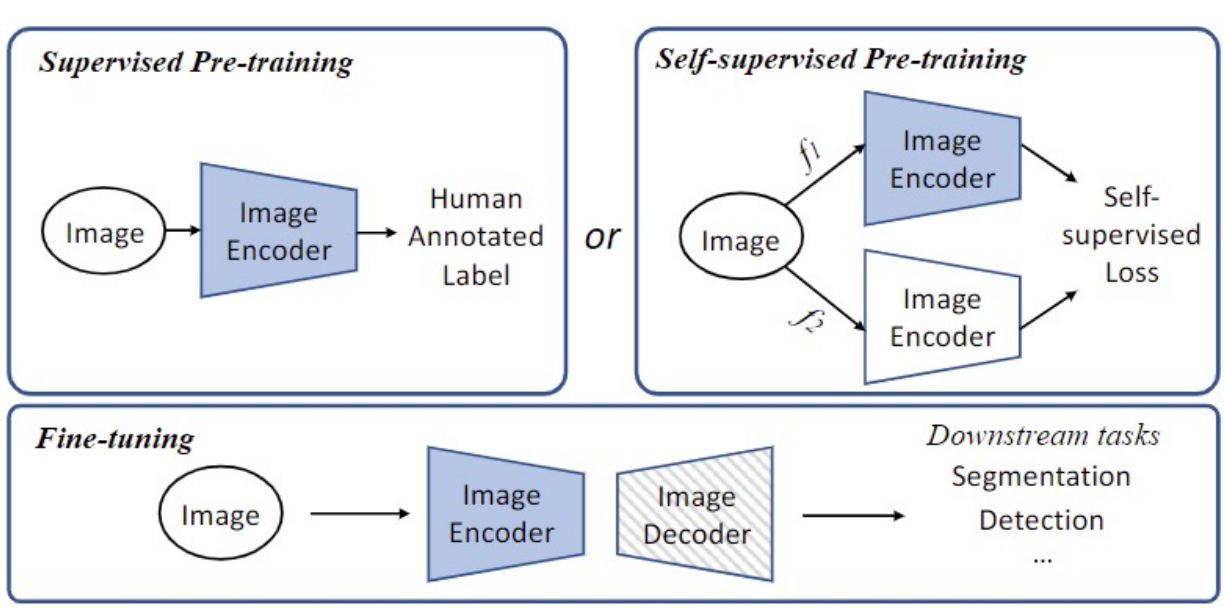
DenseCLIP: Language-Guided Dense Prediction with Context-Aware Prompting

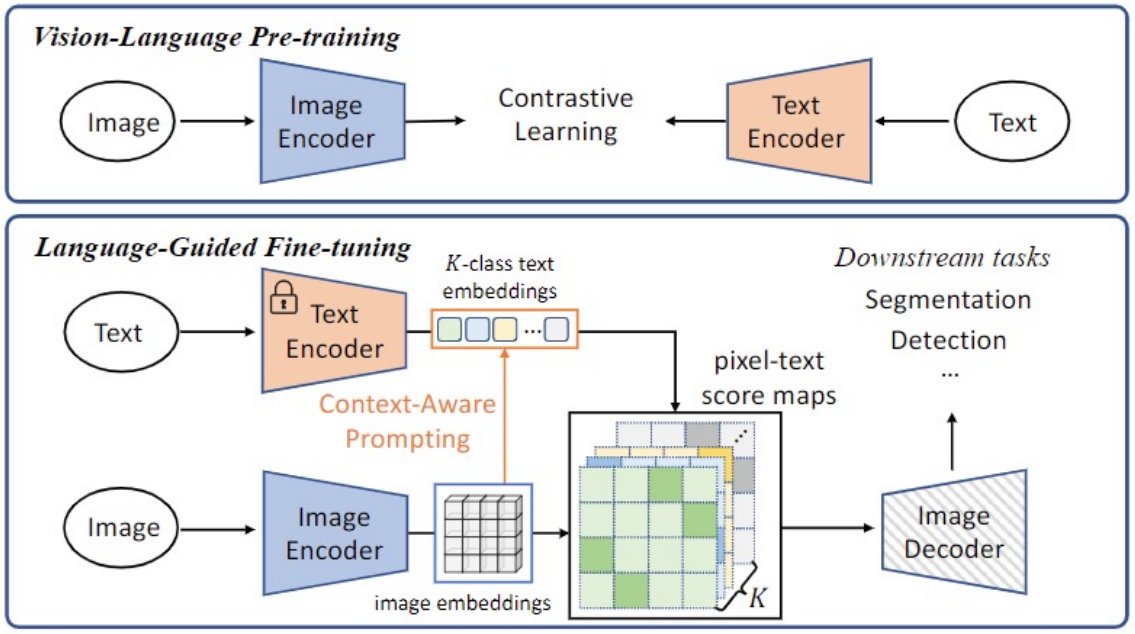
Yongming Rao[*,1], Wenliang Zhao[*,1], Guangyi Chen[2,3], Yansong Tang[4],
Zheng Zhu[1], Guan Huang[5], Jie Zhou[1], Jiwen Lu[†,1]
[1]Tsinghua University, [2]MBZUAI, [3]CMU, [4]University of Oxford, [5]PhiGent Robotics

韩坤洋
12/25/2021

Learning Representation

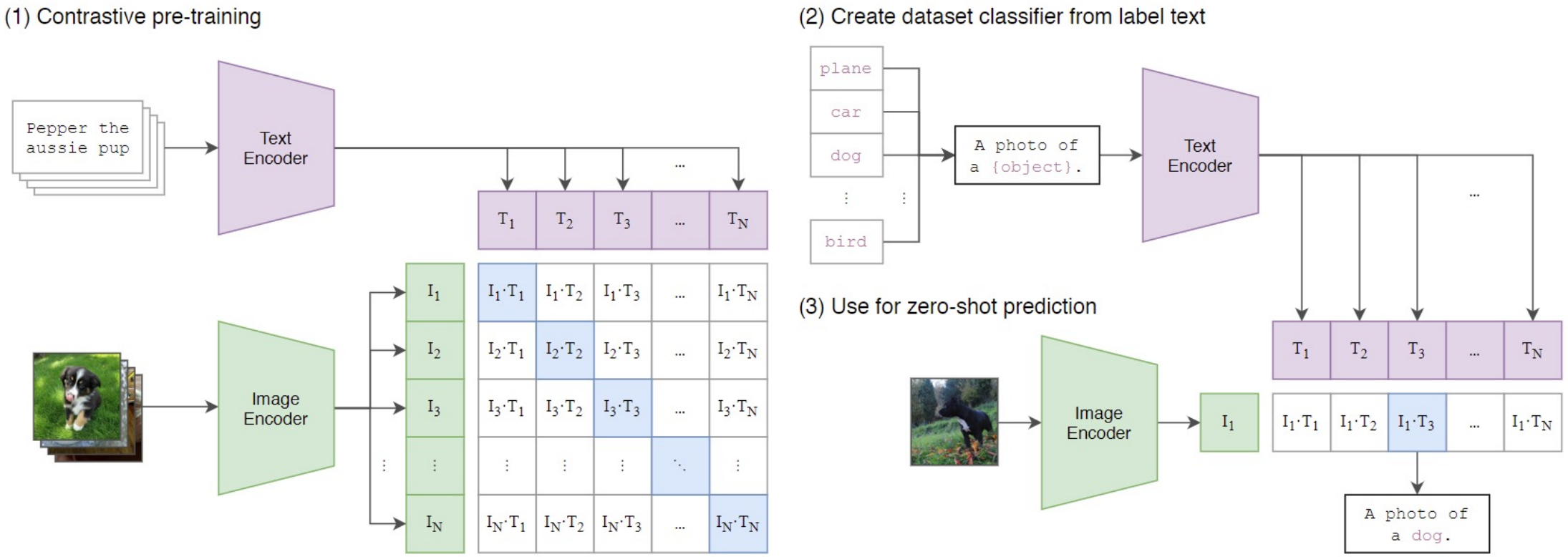


(a) Conventional Pre-training + Fine-tuning Paradigm

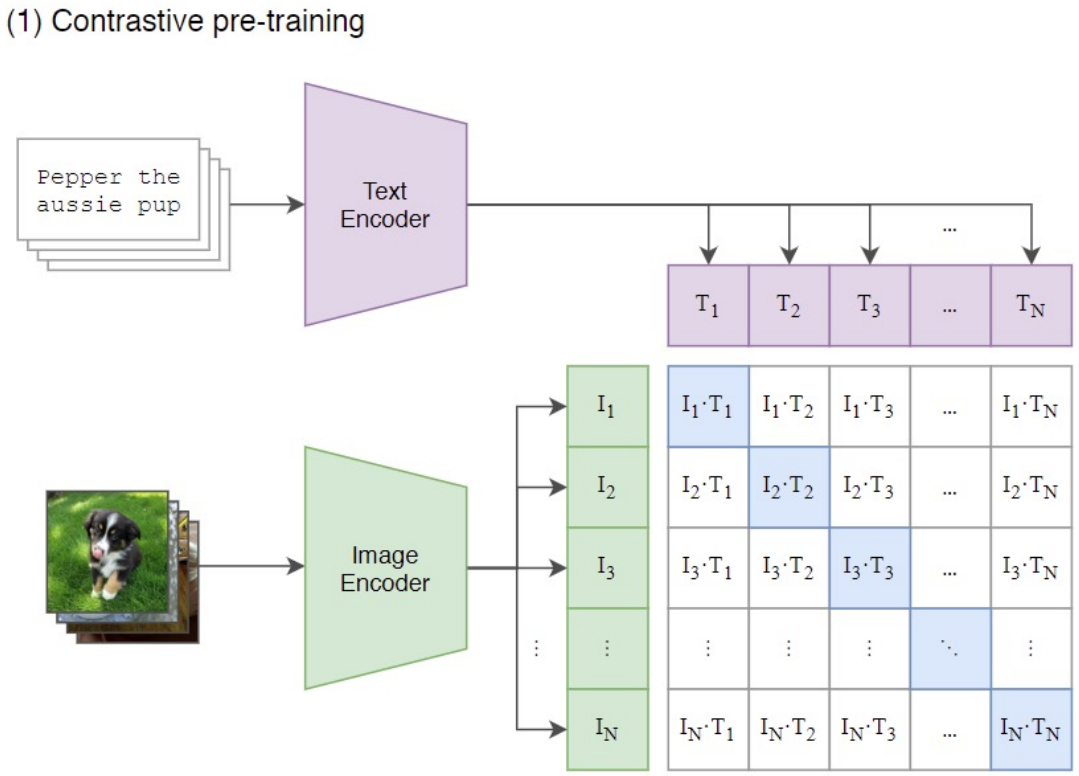


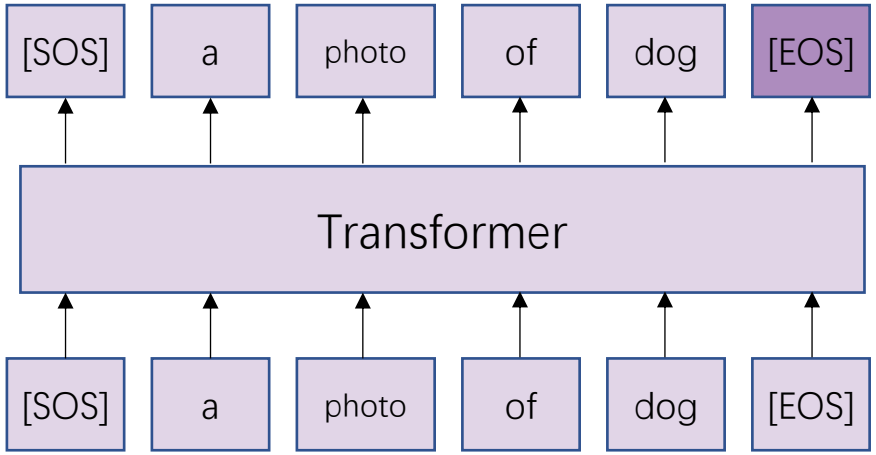
(b) DenseCLIP: CLIP Pre-training + Language-Guided Fine-tuning

CLIP: Connecting Text and Images

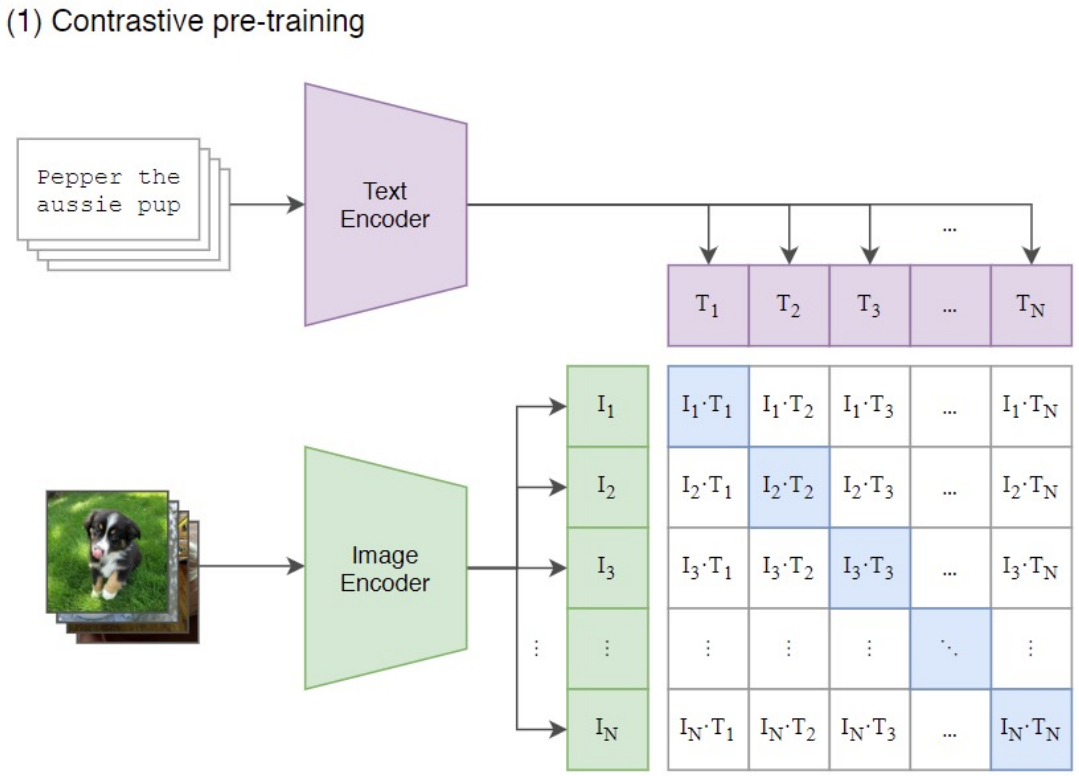


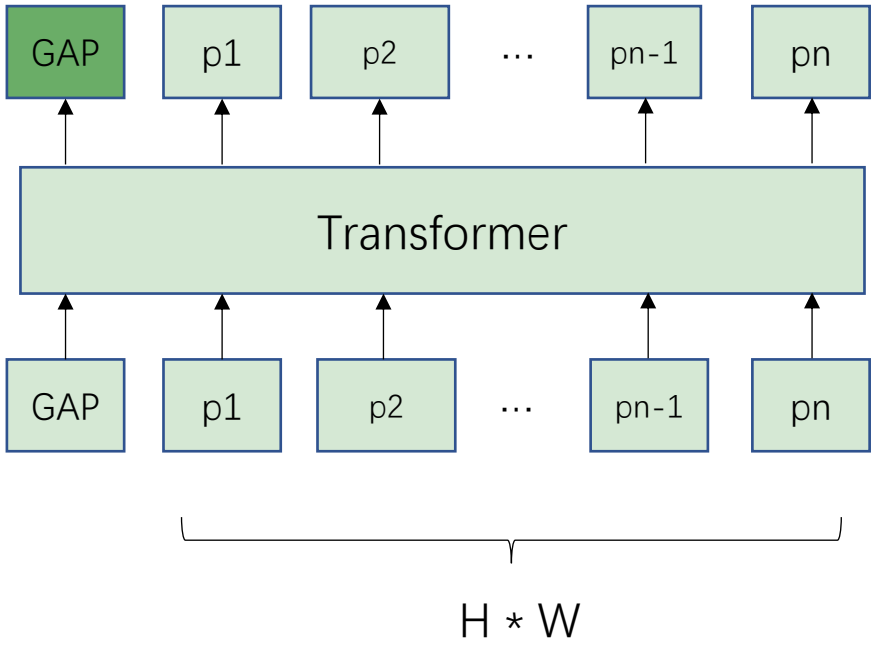
CLIP: Connecting Text and Images



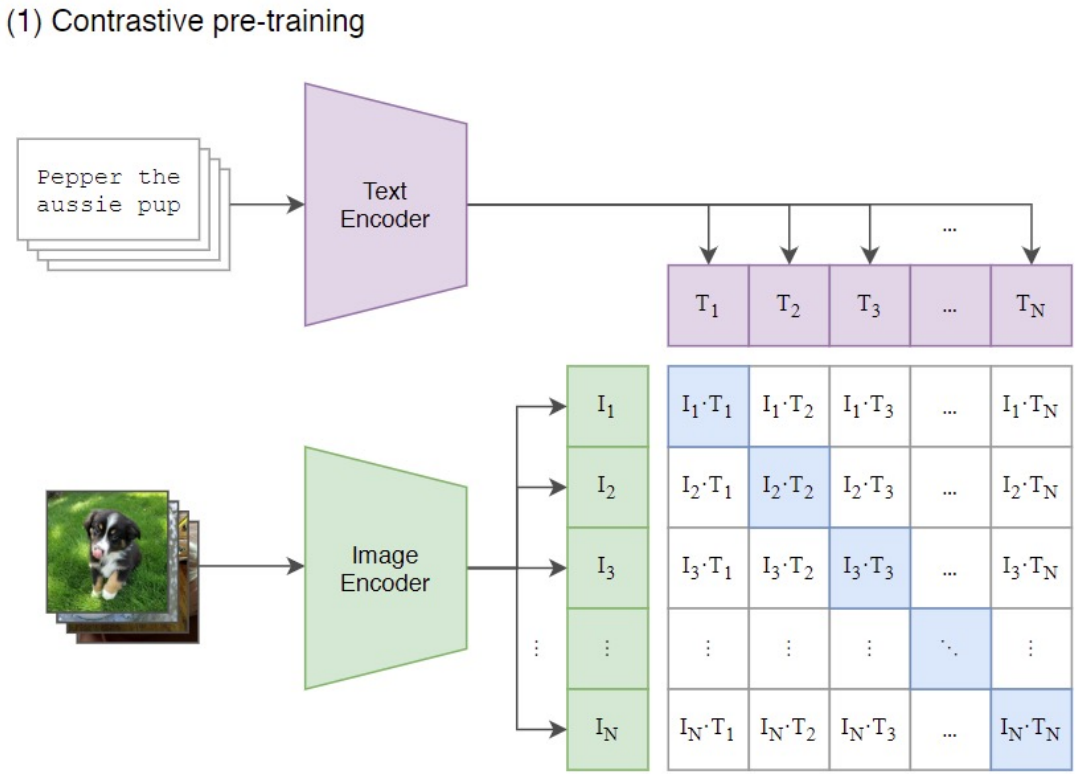


CLIP: Connecting Text and Images



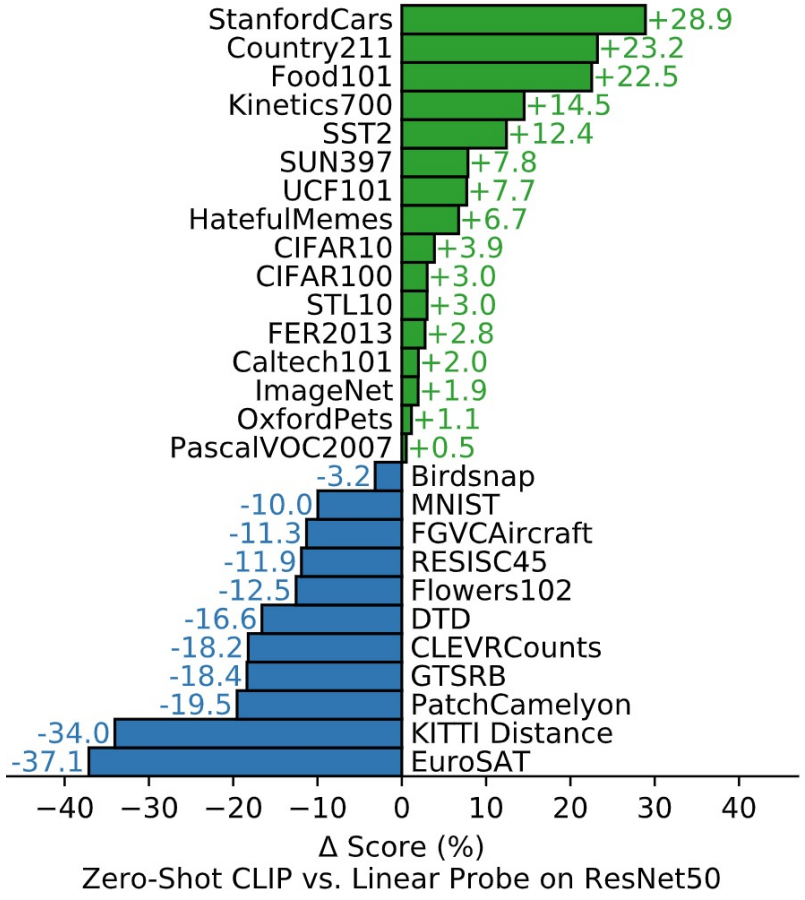


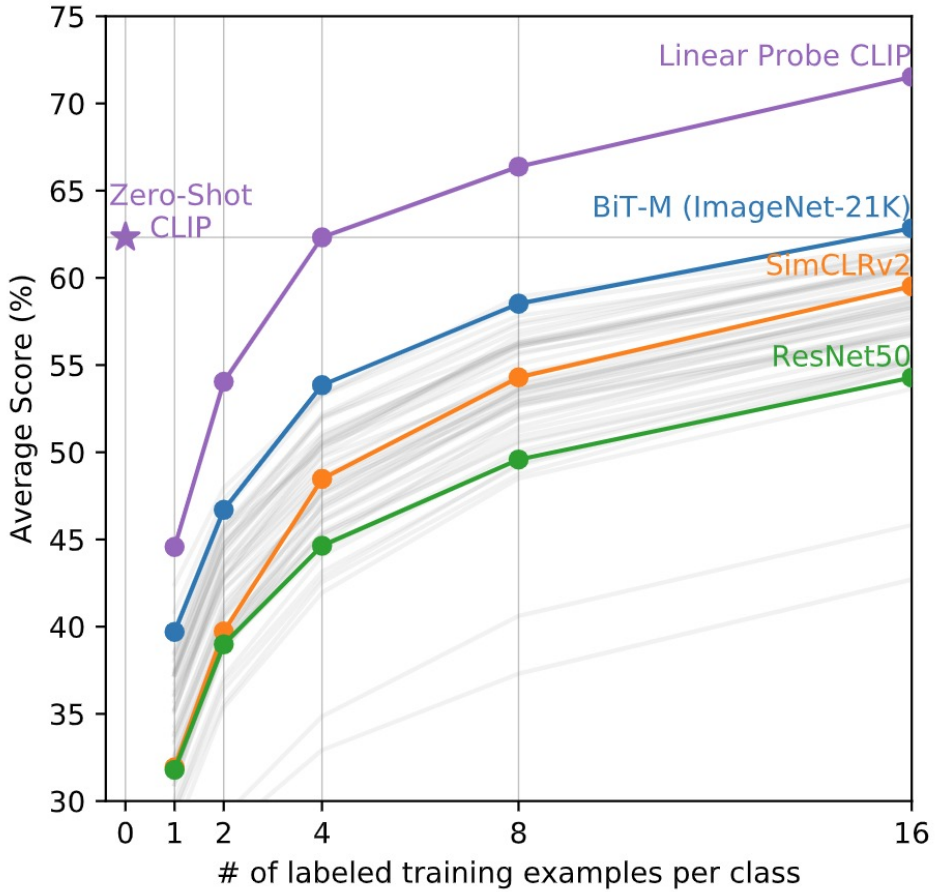
CLIP: Connecting Text and Images



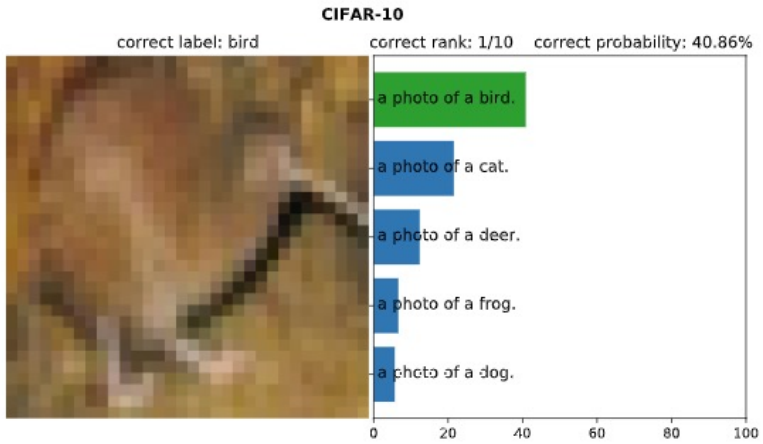

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned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

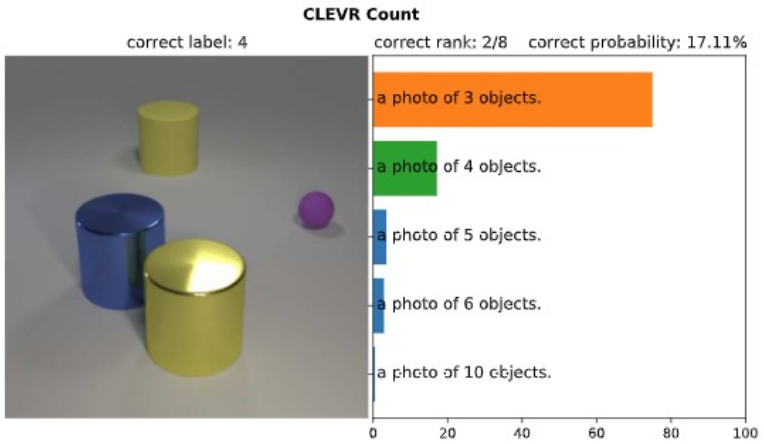
CLIP: Connecting Text and Images

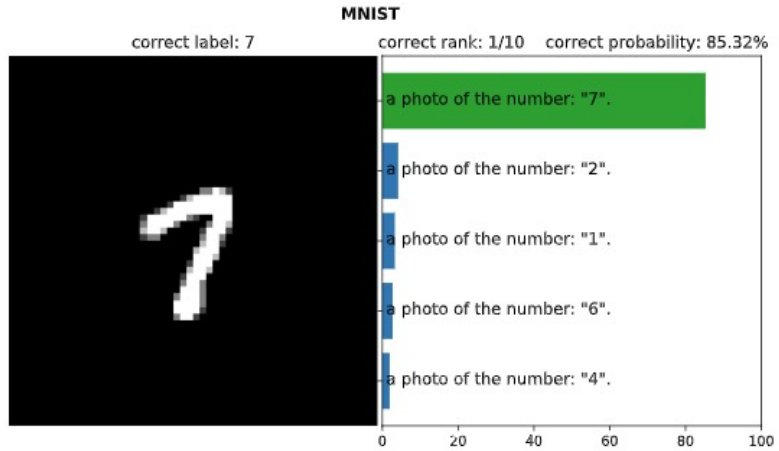


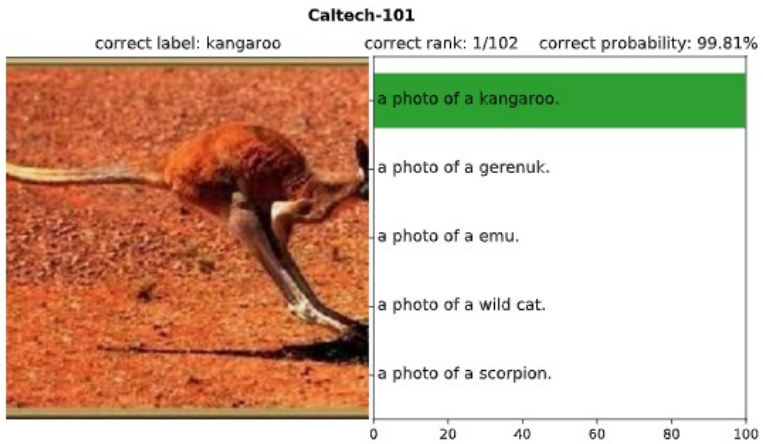


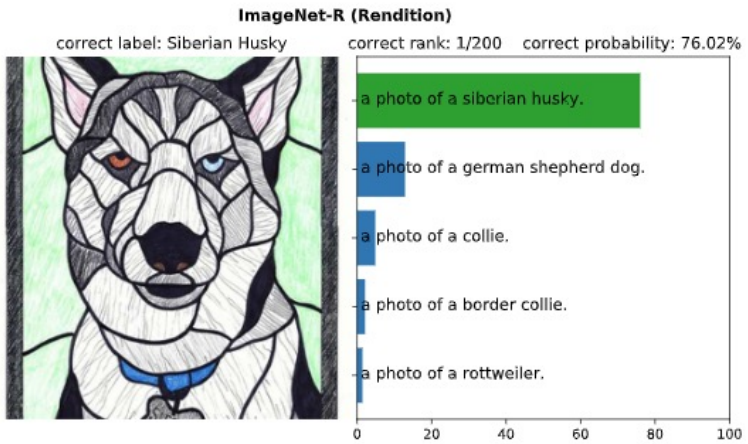
CLIP: Connecting Text and Images

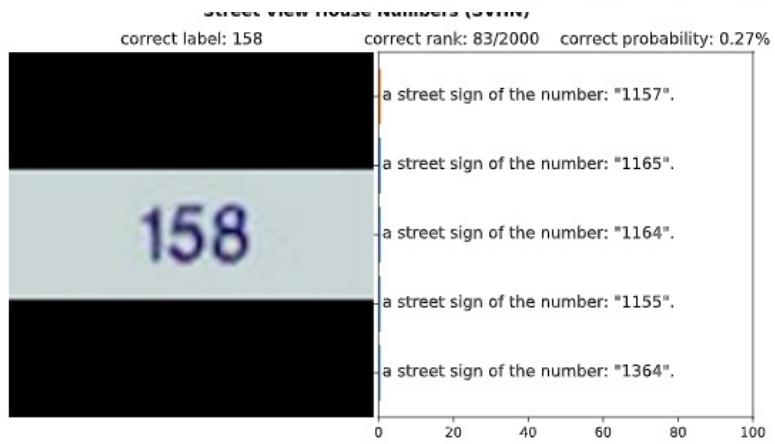










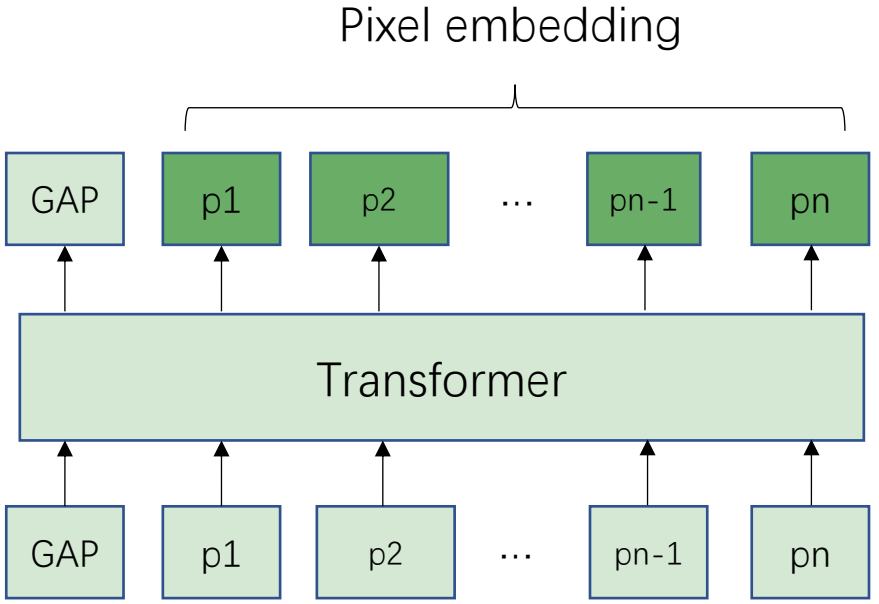


CLIP: Connecting Text and Images

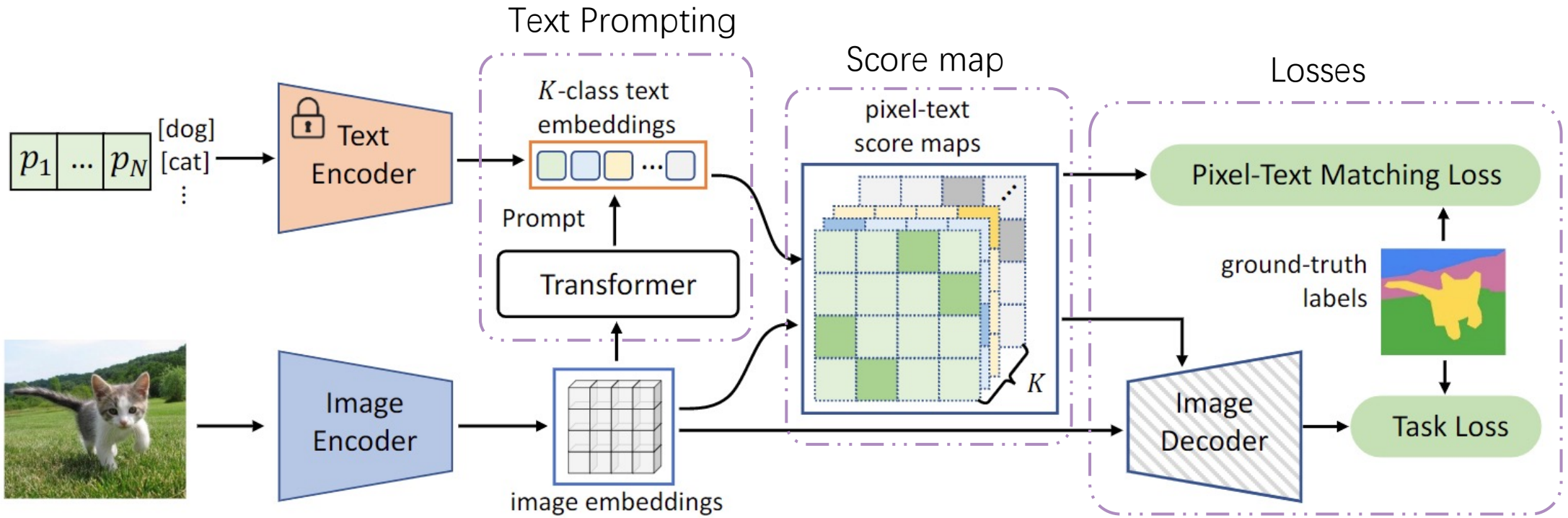
- Dataset
  - 400 million pairs (image, text)
  - Collected from Internet
- Training
  - 32 epochs
  - minibatch size of 32,768
  - 18 days on 592 V100 GPUs (RN50x64)
  - 12 days on 256 V100 GPUs (ViT-L/14)

Dense Clip

- Motivation
  - Clip, image-text problem
  - Dense clip, pixel-text matching problem

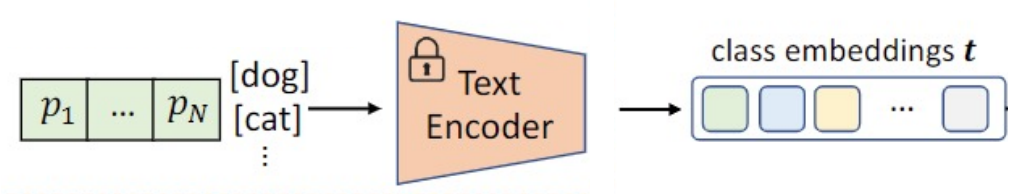


Dense Clip - pipeline



Dense Clip - Context-Aware Prompting

- Improve the text features
  - Instead of predefined templates, like “a photo of a [CLS]”
  - Use learnable textual contexts
- Pipeline
  - Dog (word) – tokenize → **[Dog] (vector)** – encoder → t (embedding)
- Classes
  - ADE20K, 150 categories, 150 * t

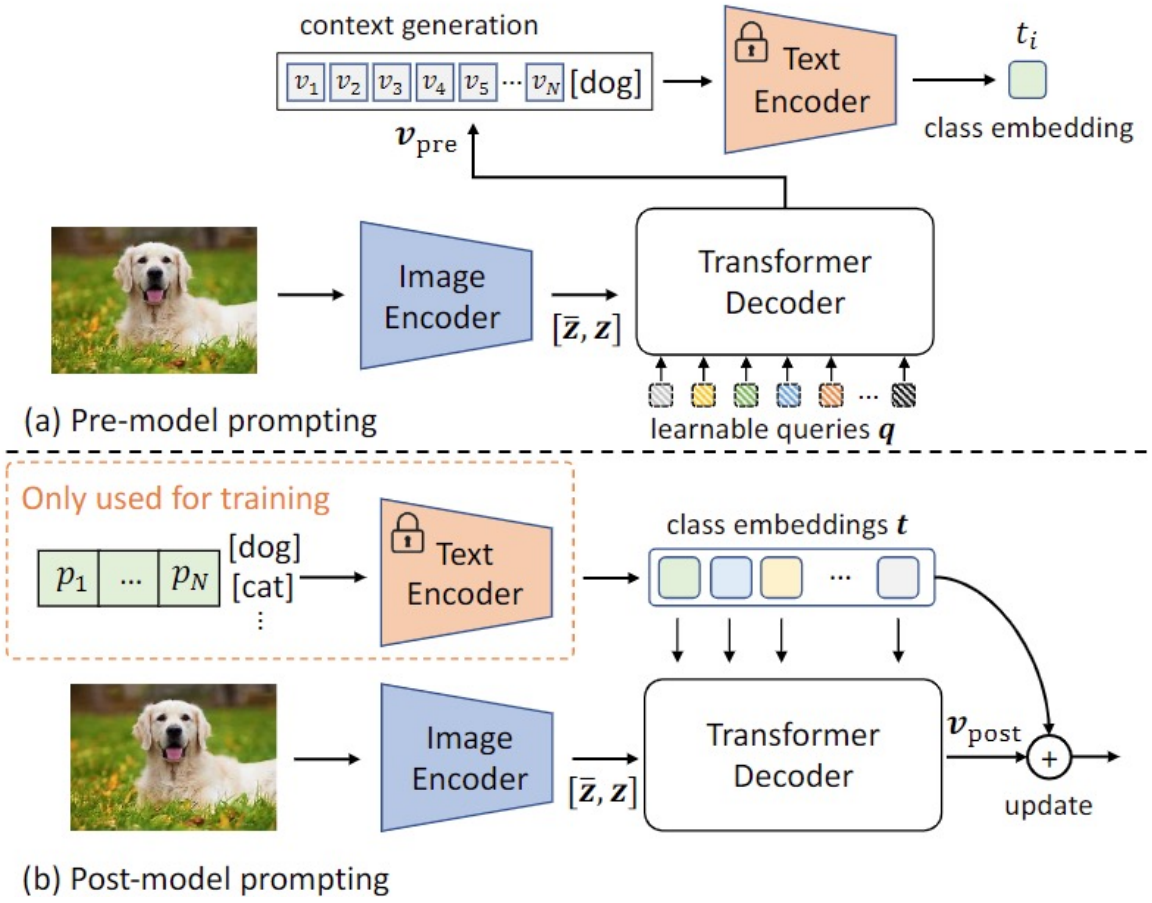


Dense Clip - Context-Aware Prompting

1. Pre-language-model prompting
   - K&V: vision feat, Q: learned queries
   - Output: text feat

2. Post-language-model prompting
   - K&V: vision feat, Q: text feat
   - Output: text feat

   - text feat = text feat + γ * text feat



Dense Clip - Context-Aware Prompting

1. Pre-language-model prompting
   - K&V: vision feat, Q: learned queries
   - Output: text feat

2. Post-language-model prompting
   - K&V: vision feat, Q: text feat
   - Output: text feat

   - text feat = text feat + γ * text feat

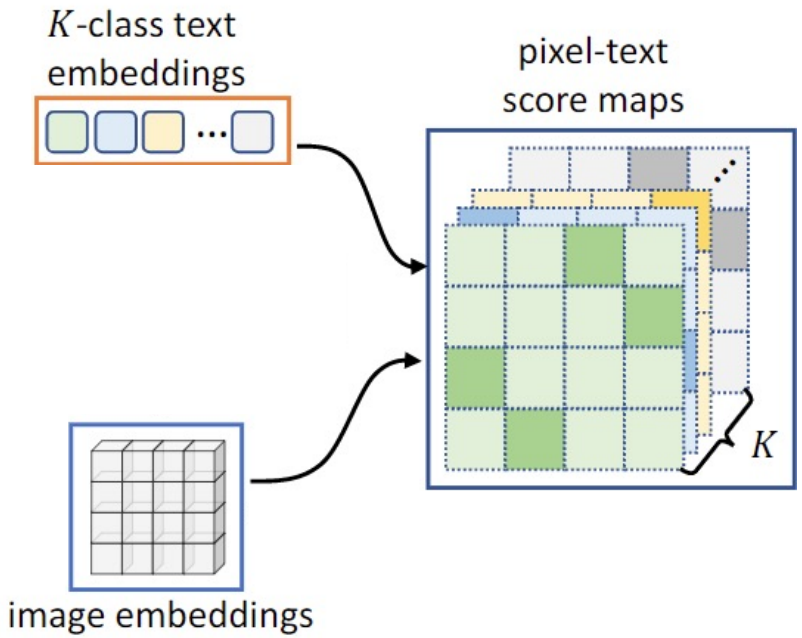
Table 2. **Ablation study.** We demonstrate that performing post-model vision-to-language prompting can yield the better performance with fewer extra FLOPs and parameters.

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

Dense Clip - Score map

- Pipeline
  - TE, K-class embedding, [K, C]
  - VE, Pixel embedding, [H*W, C]
  - S, Score map = VE * TE^T, [H*W, K]

1. Auxiliary loss
2. Concat as language priors
   - Reshape(Cat([VE, S])), [K + C, H, W]

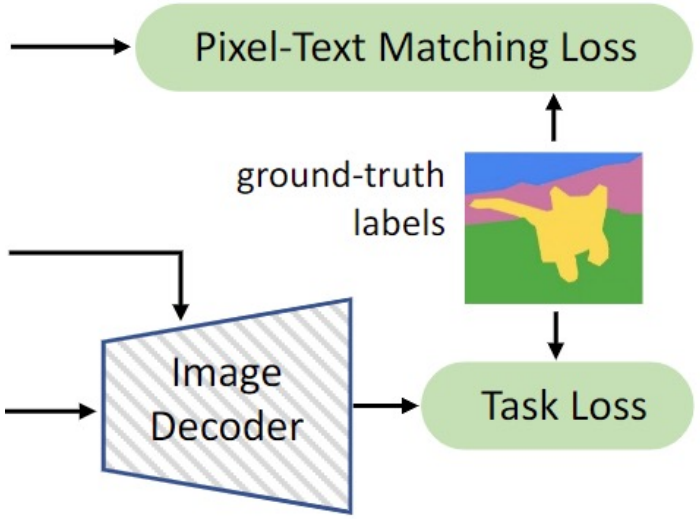


Dense Clip - Loss

- Auxiliary loss, Score map

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

- Task Loss
  - [K + C, H, W] – Decoder → Pred, [K, H, W]
  - Cross Entropy Loss



Dense Clip - Training

- Dataset, ADE20K
  - 150 categories
  - 20K training images, 2K val images
- Train setting
  - 80,000 iters
  - 32 minibatch size
  - 8 GPU
- Key modifications
  - AdamW
  - 0.1x lr

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 [31]	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+ [8]	ImageNet	42.7	43.8	711.5	43.7
	UperNet [47]	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 [31]	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+ [8]	ImageNet	44.6	46.1	1022.7	62.7
	UperNet [47]	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

Dense Clip - Result

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 [47]	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**$_{(+0.2)}$

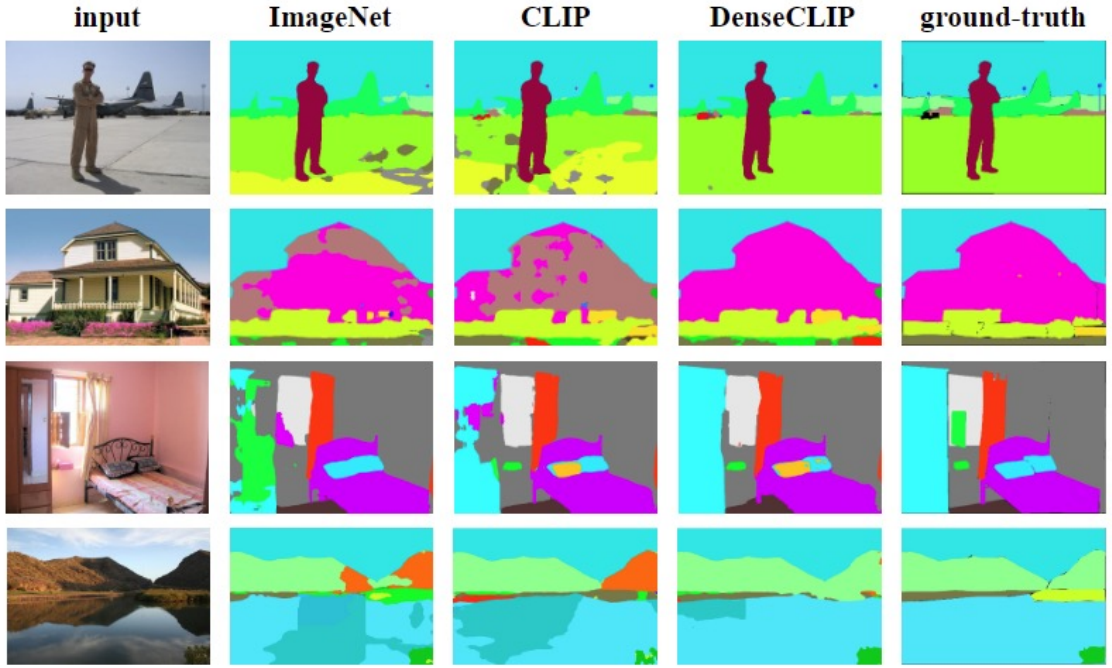


Figure 5. **Qualitative results on ADE20K.** We visualize the segmentation results on ADE20K validation set of our DenseCLIP based on ResNet-101 and two baseline models.